Implementation and validation of radiomics in lung cancer

Master Thesis in Medical Physics

by

Filip Bjurstrøm

Department of Physics and Technology
University of Bergen

June 01, 2022

Acknowledgements

First of all, I would like to extend my deepest gratitude towards my supervisor, Liv Bolstad Hysing, and co-supervisors Helge Pettersen and Erlend Hodneland for their invaluable guidance throughout this project. Thank you to Liv for a truly engrossing subject for this thesis, providing your excellent insights for the project and guiding the research. Thank you Helge, for sharing your expertise in programming and imaging, especially for helping me navigate the CT-DICOM jungle. Thank you to Erlend for sharing your experience within the field and helping with the finer mathematical details. Thank you all for your availability and great feedback on my progress, making this thesis possible.

Thank you to all my friends in room 534 at IFT for all the great moments, discussions and motivation. Also for your efforts in adopting me, against my will, into the engineering-crowd. A smaller thanks to the PhD guys down the hall for continuously reminding me that I have much yet to learn, but also for your genuine help and academic inspiration.

Finally, thank you to Camilla for your fantastic humour and enduring patience. Thank you for being here with me all this time and supporting me, I am forever grateful.

- Filip Johan Bjurstrøm
Bergen, June 1 2022

Abstract

Introduction: Cancer tumours exhibit spatial and temporal heterogeneity, and quantitative medical imaging has the potential to non-invasively capture complete tumour details where conventional invasive techniques are limited. Radiomics is an emerging field that focuses on the extraction of high-dimensional quantitative data from medical images and mining these large feature vectors for potential relationships with clinical objectives such as survival, histology or treatment response.
Materials and methods: A Python-platform was developed for calculating and analysing radiomic features using *PyRadiomics* on complete cohorts of patients with associated CT image- and segmentation data. This implementation was used to extract radiomic features from the publicly available Lung1 NSCLC cohort, which was used in a previous study to develop a prognostic radiomic signature. Features were also extracted from an internal cohort at HUH of LA-NSCLC patients to examine the viability of applying the signature on an independent dataset of higher-risk patients. Results from the previous study on the signature were replicated using the implementation for verifying correct extraction of radiomic features. The signature was also examined for correlation with tumour volume. The difference in feature value distributions between Lung1 and HUH was used as a measure for the viability of using the signature on the HUH cohort.
Results: The results from the study were reproduced successfully using the implementation, ensuring that features were extracted correctly. The features Energy, Gray-level non-uniformity (GLNU) and HLH Gray-level non-uniformity out of the four features comprising the prognostic radiomic signature showed predictive capabilities in relation to overall survival, but were also strongly correlated with the tumour volume from which they were calculated. The final feature, Compactness2, was not significantly correlated with volume and had low prognostic performance for overall survival. When comparing the distribution of feature values between Lung1 and HUH, Energy and GLNU had significantly similar distributions. Restricting Lung1 to only LA-NSCLC (LA-Lung1) patients and comparing to HUH improved the similarity of HLH GLNU between the two cohorts to be significant. Compactness2 did not show any similarity in the distribution of values when comparing HUH to both Lung1 and LA-Lung1.
Conclusion: The implementation developed by the candidate was able to calculate radiomic features reliably. Tumour volume was embedded in the radiomic signature due to the feature selection process in the original Lung1-study not accounting for the underlying mathematical definitions of selected features. The clinical relevance of the radiomic signature and comprising features cannot be assessed accurately due to the intrinsic influence of volume. The co-dependencies to tumour volume also disrupted meaningful assessment of signature transferability onto HUH. Further investigations into the dependencies of features with volume should be performed, and of the prognostic and translational potential after corrective measures have been done, e.g. normalisation.

List of Figures

List of Tables

Contents

1

Introduction

Historically and contemporary, medical images have been a subjective matter, where the knowledge and experience of the clinical observer are the key factors in making decisions regarding diagnosis and treatment. Radiomics is a relatively new field which is based on the calculation of predefined quantitative metrics from medical images or specialised matrices derived from these. The hypothesis is that these so-called radiomic features are able to non-invasively capture certain details and trends in segmented tumour volumes that usually not possible, nor practical, for the clinicians to investigate, and moreover relate these to clinical objectives such as survival, treatment response, histology, etc. in patient populations. For developing these models, potentially hundreds of features are extracted and then exhaustively mined for significant relationships with clinical objectives.

These additional parameters are hopefully able to assist clinical decision-making and further differentiate patient disease characteristics, facilitating more personalised medicine. Medial images such as CT, MRI and PET are captured extensively during the radiotherapy treatment planning process and thus forms an extensive database of images that can be mined for radiomic features with relevant clinical information. As radiomics only requires the medical images that are acquired routinely for treatment planning, it can potentially be seamlessly implemented into the clinical workflow and provide immediate feedback on a patient's radiomic profile to be used in conjunction with existing parameters for clinical aid.

Lung cancer is the second most common cancer type for both men and women, and the leading cause of cancer-related deaths both in Norway and globally (Sung et al., 2021). It is often diagnosed in late stages, with generally poor survival outlook. Cancer tumours in general also exhibit spatial and temporal heterogeneity (Aerts et al., 2014) and hence recent efforts in personalised medicine using invasive biopsies and genomic profiling have been limited due. CT is the most common modality for imaging lung cancer, providing good contrast between the tumour tissue and the air-filled lungs, which can be further enhanced using PET-CT. It moreover provides good spatial resolution for a detailed representation of tumour structure and texture, allowing accurate calculation of radiomic features.

In 2014, Aerts et al. published an extensive study showing potential for relating radiomic features with both staging and histology in non-small-cell lung cancer (NSCLC). Using a large database of several training and validation cohorts they also developed a four-feature prognostic radiomic "signature" which showed apparent predictive power of features describing tumour heterogeneity in both NSCLC and head-and-neck (H&N) cancer patient cohorts. The large scope and exhaustive methodology of the study gave confidence in the capabilities of radiomics, and subsequent research has extensively investigated the wide range of potential radiomic applications, from automatic classification of cancer histopathology to describing changes in tumours during the radiotherapy treatment process. As many are very optimistic regarding the results presented in the field, many also advocate for caution and scepticism, citing close relationships to tumour volume and limited implementation with current clinical conditions (Anagnostopoulos et al., 2022). Additionally, with the radiomic process consisting of several inter-disciplinary stages with a wide range of applied methodology, standardised approaches to the workflow for

yielding highly reproducible results is necessary. Aerts et al. provide extensive documentation of their analyses and results, moreover making the datasets used in their study publicly available on the cancer imaging archive (TCIA). The in-house MATLAB code they used for feature calculation and numerical details on the fitted models was not disclosed however, necessitating some guesswork when attempting to replicate their methods and results.

Objectives

The goal of this work is to use an independent implementation in Python to recreate the work done in Aerts et al., 2014 and moreover see if these results and methods are applicable to an independent dataset. Haukeland University Hospital (HUH) provided a set of locally advanced NSCLC (LA-NSCLC) patients to this project for evaluating the transferability of the radiomic signature from Lung1 dataset available on TCIA. Thus for this work, the following objectives were formulated:

1. Implement a Python platform for calculating radiomic features on multiple datasets using the free open-source software *PyRadiomics*
2. Validate this implementation by reproducing the results from Aerts et al., 2014 on their publicly available dataset, Lung1
3. Investigate if the radiomic signature proposed by Aerts et al. provides additional prognostic information to established clinical parameters
4. Determine if the radiomic signature proposed by Aerts et al. is transferable to an independent local dataset of higher risk patients (HUH)

2

Theory

2.1 Radiomics and the radiomic workflow

Radiomics is an emerging new field at the intercept between medical imaging, big data and quantitative image analysis. Due to the rapid proliferation and evolution of cancer cells, tumours present with diverse genetic and other molecular diversities, both between (inter-) and within (intra-) individual cancer tumours. Hence cell biopsies from one portion of a tumour might be limited in describing the entire tumour biology. Medical imaging is performed extensively during the radiotherapy treatment process and captures the entire three-dimensional tumour volume enabling radiomics to potentially describe the complete tumour biology, provided accurate imaging and applied methods. The hypothesis is that quantitative metrics, called *radiomic features*, derived from calculations on these digital medical images of tumour volumes are related to clinically relevant factors, such as staging, treatment response, histology, and more.

The goal of radiomics is to aid clinical decision-making, by providing clinicians with additional quantitative metrics that can be used in conjunction with the established clinical measures and qualitative evaluations. Some potential applications include providing additional information on tumour response, distinguish cancer tissue from benign tissue (Wibmer et al., 2015), association of tumour phenotype with prognosis (Aerts et al., 2014) and the prediction of adverse effects induced by cancer therapy (Colen et al., 2018).

Radiomics can potentially be implemented in parallel to the current clinical workflow where digital radiological images are routinely gathered for most cancer patients (Gillies et al., 2016) for diagnostic and/or treatment purposes. Especially in radiotherapy treatment, radiomic parameters deduced from images retrieved for planning purposes have the potential to immediately assist in decision-making processes regarding the subsequent treatment. This can moreover be done with minimal intrusion on patients and clinical personnel, as only the image data is required.

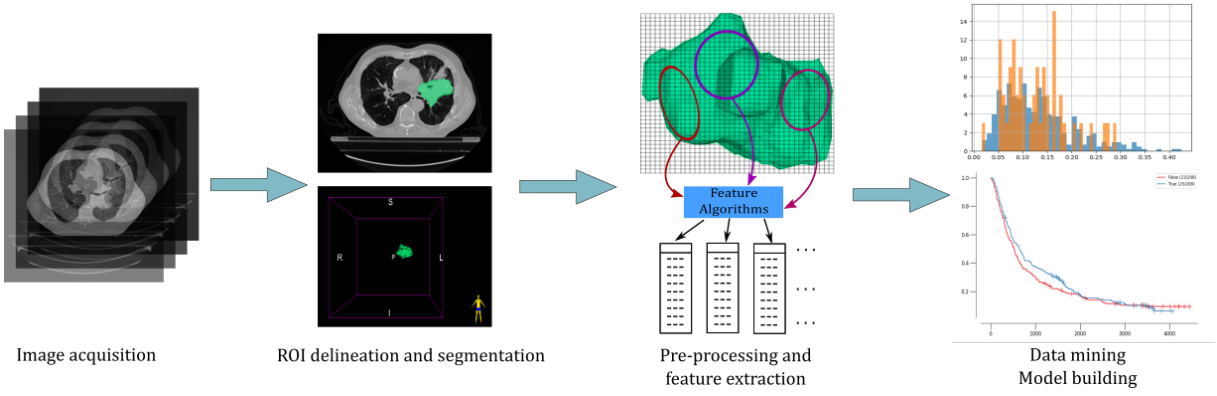


Figure 2.1: The radiomics workflow

2.1.1 Radiomics workflow

To ensure reliable and reproducible results, with reportable and preferably quantitative uncertainties, a consistent and standardized work pipeline for calculating radiomic features should be followed. With a standardised workflow together with well-defined relationships between workflow-steps and final result, the source of potential errors and uncertainties in developed models and methodology will be easier to identify.

The workflow for radiomics can be summarized in short by Figure 2.1. Radiological images are acquired of the patients body as a part of the treatment planning routine, usually MRI, CT or PET-CT in the case of e.g. lung cancer. Regions considered to contain tumour tissue or other pathological relevant information are identified on these images, and the voxels contained are extracted to form the volume that is used to calculate radiomic features. The volume is then preprocessed and features are calculated from the segmented tumour volume using various mathematical and image processing methods. Potentially hundreds of features can be calculated (Zwanenburg et al., 2020b) from a single volume, which through the final step of the workflow can be mined together with clinical data to build models.

In this section, we will describe the various steps of the radiomics pipeline in detail, from image acquisition and to the creation of models that can aid in clinical decision making.

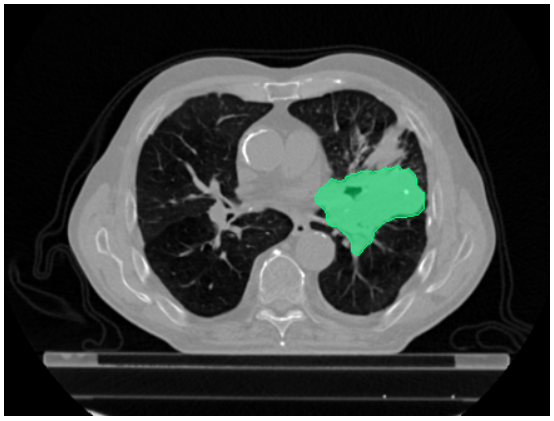
Image acquisition

During cancer treatment, radiological images are acquired of the patient for both diagnostic and treatment purposes. For diagnosis, images are used primarily for locating tumour position, extent and possible local- or distant metastases. Additionally, further imaging is done specifically for planning radiotherapy treatment, where the patient's treatment process is simulated and dose plans are calculated. Furthermore, images are taken during the treatment course for visual monitoring of progress and response, which are usually not delineated. The images that are delineated from the treatment process and of acceptable quality can be retrieved for radiomics research with minimal intrusion, provided institutional review board approval.

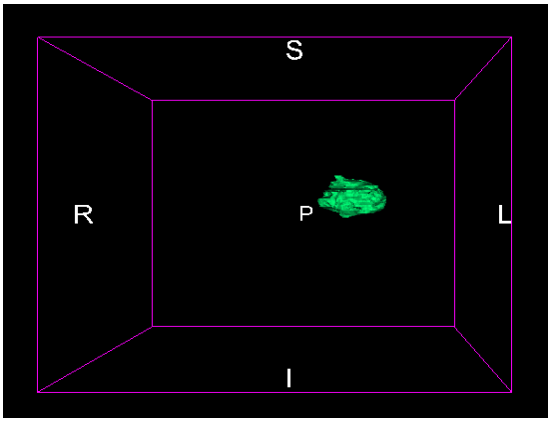
The three main modalities used for radiological imaging in cancer treatment are CT, MRI and PET-CT. A natural consequence of using different modalities are differences in working principles, acquisition standards and image reconstruction techniques, not considering possible differences in protocols between institutions. For reproducible results, we want inter-modality imaging to be as consistent as possible, with regards to spatial and intensity resolution, grey-level range, noise characteristics and segmentation. The Image Biomarker Standardisation Initiative (IBSI) is an ongoing collaboration for improving the reliability and quantising uncertainties of calculated radiomic features with regards to differences in modality, image processing scheme and image filters (Zwanenburg et al., 2020b). Extensive disclosure of imaging protocols in radiomic studies are essential for achieving reproducibility and comparability of results (Lambin et al., 2017).

ROI identification and segmentation

In order to capture phenotypic differences between different tumours, we need first to define which areas of an image contain tumour cells, and which do not. For the purposes of radiotherapy treatment, an oncologist will define several volumes for which the treatment dose will be distributed. The gross tumour volume (GTV) is the base volume defined, which includes the the tumour itself as seen on images and possibly detected by biopsies (ICRU, 2010). This can be surrounded by several safety margins and extended volumes, as for example the planned target volume (PTV), which accounts for additional subclinical disease and various uncertainties in



(a) A crossectional view of a patient's abdomen with a 2D contour of NSCLC tumour ROI highlighted.



(b) The entire ROI contour rendered in 3D by combining the contours from all crossections.

Figure 2.2: Segmented NSCLC GTV in both two- and three dimensions.

radiotherapy dose delivery. In the case of 4D images which vary with time, a specialised volume for the union of GTV positions at all time frames can be defined, to account for patient motion during the treatment process.

In practice (Aerts et al., 2014) it is reasonable to use the GTV as the ROI for radiomic calculations, as it gives a definite volume in which a discernible tumour is contained. There are three types of approaches to segmenting ROIs and other anatomical structures in medical imaging: manual, semi-automatic and automatic. Manual delineation is when a trained clinician draws the entirety of the structure contour "by hand", often resource-intensive and time-consuming. Semi-automatic delineation is when one uses the aid of specialized algorithms and software to delineate a structure, with the operator usually controlling some parameters of the algorithm such as initialisation and stopping conditions. Examples of semi-automatic methods include region-growing and active contours. Automatic delineations are performed entirely by a specialized algorithm, usually with techniques from machine learning such as neural networks.

Usually the segmentations or contours are given as piecewise linear boundaries defined around the circumference (2D) or surface (3D) of a particular structure (Figure 2.2), that aid observers in the precision delivery of radiotherapy dose. However for radiomics, it is most practical to convert these curves into binary masks that cover the entire area inside the contours, that can later be used via array multiplication as shown in section 2.4.2 to segment the ROI. Thus only voxels inside the ROI will keep their original intensity values, while all other pixels are set to zero and ensuring that the subsequent radiomic calculations are only performed on the ROI.

Image processing

Prior to radiomic feature calculation, extra precautions can be made in order to ensure highest achievable data quality and reproducibility. The image processing in radiomics is focused on homogenizing image characteristics such that images from different institutions, acquired with potentially different equipment and protocols, are as comparable as possible before feature calculation. The image parameters subject to pre-processing are for example voxel dimensions, grey-level intensities, intensity histograms, etc. (Timmeren et al., 2020). The specific pre-processing done depends on the characteristics of the image material and the intended radiomic applications.

Some techniques for radiomic image preprocessing include:

- *Intensity discretization*: Voxels inside the ROI are resampled into discrete intensity bins. The intensity range, number of bins, and bin width characterize the discretization performed on the ROI. This must be done before calculation of texture matrices that are dependent on discrete intensity values (Section 2.5.3). E.g. for texture matrices which describe the adjacency of voxels with equal intensities, will rarely find adjacent voxels of the same grey-level value if the intensity range is very wide or consists of floating point numbers.

- *Voxel resampling*: Some features are dependent on voxel dimensions, and thus images can be resampled to a common spatial resolution for more accurate comparison of image material with differing voxel dimensions (Shur et al., 2021).

- *Interpolation to isotropic voxel spacing*: Equal spacing of voxels in all directions is necessary for most texture feature sets to become rotationally invariant and allow for comparisons of data from different sources (Timmeren et al., 2020)

- *Range re-segmentation and intensity outlier filtering*: For CT, ROI voxels outside of some defined intensity range are removed e.g. excluding voxel describing air or bone inside a tumour ROI. MRI has arbitrary grey-level units, so intensity outlier filtering is done instead, where voxels are excluded if they fall outside a range based on the intensity mean and standard deviation in the ROI, e.g. $\mu \pm 3\sigma$ (Timmeren et al., 2020).

In practice, the preprocessing step is often implemented into the feature calculating software, as is the case for *PyRadiomics*[1], which is a free open-source package for calculating radiomic features in the Python programming language. The operator is free to control the parameters of the preprocessing by providing specific settings to the software implementation.

Feature extraction

After the appropriate considerations and preparations have been done with regards to ROI segmentation and image preprocessing, quantitative image features can finally be extracted.

Feature extraction is the process of taking a defined ROI and calculating quantitative descriptive scalars, based on the contained voxels' values, intensity histogram and relative spatial distribution, depending on the "category" of features we are calculating. An overview of the different feature classes and some relevant definitions are provided in Section 2.5. Features are calculated from either the digital images themselves, or from specialized matrices derived from these, e.g. texture matrices. Image processing filters such as gaussian-, wavelet-, or Laplacian of Gaussian (LoG) filters can be applied prior to feature extraction, for e.g. enhancing specific frequency- or edge details. Several features have been shown to be strongly co-dependent or even being derivable from one another (Zwanenburg et al., 2020a), thus in practice we need not calculate all the features available in the radiomic library.

Feature selection/Dimensionality reduction

The possible number of features that we can extract generally far exceeds the bounds of what is computationally practical for using in analyses. Moreover, a large number of variables will increase the probability of model overfitting. In addition to the fact that many features are derivatives of each other, some are also strongly correlated due to quantifying similar or equal aspects of the ROI, implying a high degree of redundancy among many features. For these reasons, feature selection, a form of dimensionality reduction, is essential for translating the

[1] https://github.com/AIM-Harvard/pyradiomics

large amounts of data into a generalizable model. By using the most robust, reproducible and informative features to represent the extended spectrum of highly correlated features, we can remove large amounts of redundancy.

Several measures can be taken to reduce the dimension of feature data. Features that show low robustness in regards to reproducibility, for example due to inter- or intra-observer variability should be excluded, in the case for manual and semi-automatic segmentation. The robustness of a feature when used in the controlled environment of, e.g. a phantom, should also be considered. Highly co-dependent features must also be handled for reducing redundancy. Correlation matrices can be a very useful tool for identifying feature clusters that are highly correlated, which is clearly exemplified in the correlation matrix of shape features shown in Figure 2.3. The features in each correlation cluster can be reduced down to a single representative feature to minimise redundancy.

In addition to excluding co-dependent and unstable features, we want to select the features that are the most informative for the intended application. A common approach is to use a machine learning (ML) model for assessing which features are most predictive with regards to the model objective. Features that exhibit strong dependencies with common clinical parameters such as tumour volume or cancer stage provide little additional information, and should also be considered for exclusion.

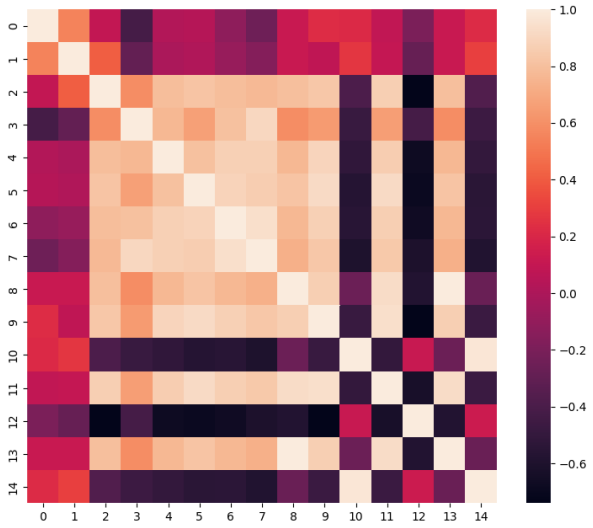


Figure 2.3: Correlation matrix of 16 shape features extracted from the Lung1 dataset

Analysis and modelling

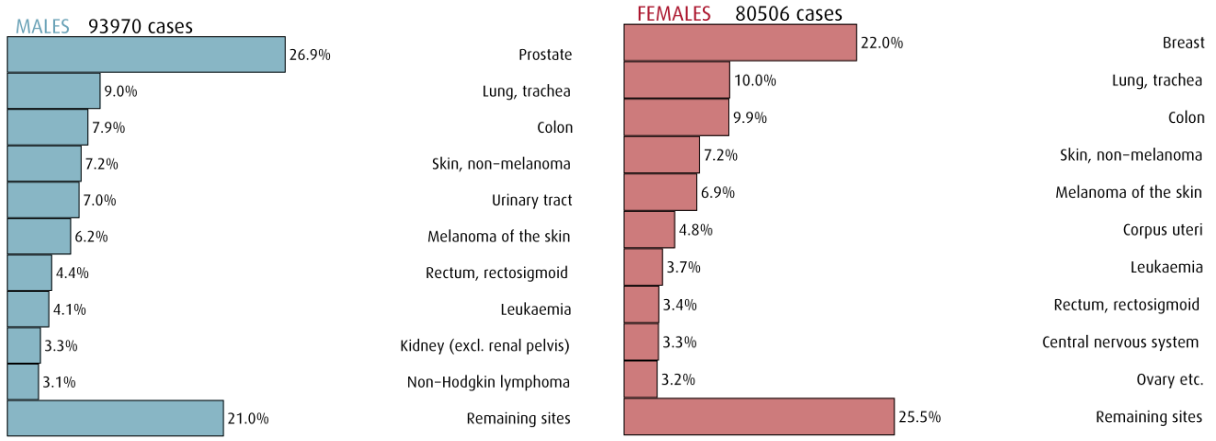
The central application of machine learning is the ability to take large quantities of potentially high dimensional as data inputs and learn certain trends that can be applied to predict some target on previously unseen data. Hence these methods are well-suited for application on radiomic data, and models can possibly be made more robust by incorporating patient parameters beyond the scope of imaging, such as genomic profiles, histology, patient history, etc. (Gillies et al., 2016).

After we have extracted the features, and performed feature selection, we are left with a smaller batch of robust, non-redundant and informative features, with which we can build a predictive model. The goal of the model is to use a given input, in this case a vector of selected features, to predict some target relating to the application of the model. The target can be either categorical for a classifier model, or a scalar value if it is a regression model. After the model has been trained, it must be validated on unseen data to evaluate its practical performance and

usefulness. This can be done internally by e.g. cross-validation, and optimally externally on independent data with similar prerequisites as the training data.

2.2 Non-small-cell lung cancer

In 2020, 35 515 new cancer cases were reported in Norway, with lung cancer being the second most common type of cancer for both men and women, behind prostate- and breast cancer respectively for the period 2016-2020 (Cancer registry of Norway, 2021). The relative frequencies of the ten most common cancer types for this period are shown in Figure 2.4. Though lung cancer is the second most common cancer type for both men and women, lung cancer is the largest contributor to cancer-related deaths, representing 20% of mortality in 2020 (Cancer registry of Norway, 2021). It is often diagnosed late and at an advanced stage, due to absence of early clinical symptoms and limited screening programs. For all stages combined, the 5-year relative survival was 24.6% and 30.7% (Helsedirektoratet, 2021) for men and women respectively in 2020, giving lung cancer the worst survival rate of the most common cancer types.



(a) (b)

Figure 2.4: Frequency of different cancer types in males and females for all ages in Norway over the period 2016-2020. Bar charts from Cancer registry of Norway, 2021.

In the recent decades, the individual lung cancer incidences for men and women have approached each other due to a on-setting decline in male incidence while it is still increasing in females, reflecting historical smoking trends. Thus, in contrast to the trend around the middle of the last century, the ratio of lung cancer cases between men and women is today 1:1 (Helsedirektoratet, 2021).

The risks of developing lung cancer is highly correlated with smoking habits, with 80%-90% of cases considered attributed within the Nordic countries (Helsedirektoratet, 2021), but is also diagnosed in people who have never smoked. Risk is also to a lesser degree associated with environmental or occupational factors such as radon or various industrial compounds. Note that age is a major factor of lung cancer cases, with risk rates increasing with increasing age. The proportion of patients 70 years and older account for over half of all patients diagnosed in recent years (Sagerup et al., 2012).

Histology

Lung cancer develops when cells in the lung undergo mutations that suppress the self-controlling mechanisms of the cell-cycle, leading to unrestricted cell division and formation of tumours. The disease is categorised into two main types: small-cell lung cancer (SCLC) and non-small-cell lung cancer (NSCLC), accounting for approximately 15% and 85% of cases respectively (Gridelli et al.,

2015). Small-cell lung cancer is characterized by small, rapidly dividing cells and an aggressive spread of the disease throughout the body. Non-small-cell lung cancer consists of larger cancer cells more prone to forming discrete tumours, and includes the following subtypes: squamous cell carcinoma; large cell carcinoma; adenocarcinoma, with each one containing further subtypes themselves. The classification of histologic subtypes of lung cancer with relative incidence rates are provided in Figure 2.5.

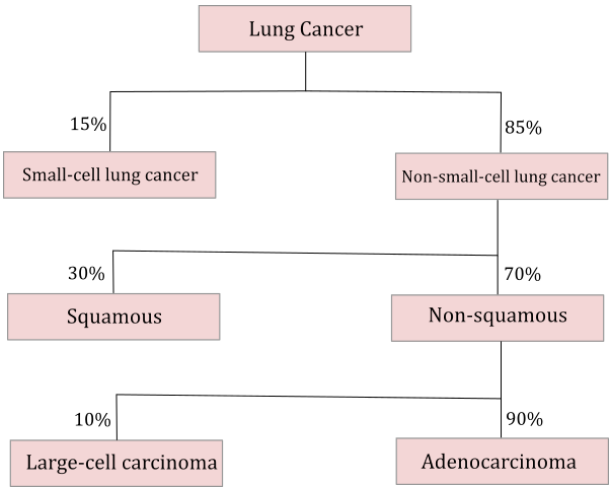


Figure 2.5: Relative incidence rates of histologic subtypes of lung cancer. Illustration and numbers reprinted from Gridelli et al., 2015.

With NSCLC being the most common type of lung cancer and also associated with the highest mortality rates compared to other cancer types, it is a field of great interest for improvement of treatment possibilities. Accurate determination of histologic subtype has become critical for administration of subtype-specific therapeutics and treatment options.

Lung tumours show diverse heterogeneity related to genetic signature and evolution, both within a single tumour and between a primary tumour and its metastases (Gridelli et al., 2015). Certain genetic mutations in the tumour can affect cell cycle proliferation and survival, and thus give indications of disease progress, aggressiveness and prognosis. The usual method of obtaining information on the molecular and genetic structure is by taking a small sample of tissue from the tumour, called a biopsy. Since these are only retrieved from a small portion of the lesion, they are limited in the describing the genetic and molecular variation across the entire tumour. More accurate descriptions of complete tumour heterogeneity can potentially allow for more personalised treatments suited for patient's individual cancer characteristics.

Staging

Accurate and consistent staging is crucial for clinicians across borders and institutions to refer to a common language for describing disease extent and appropriate treatment options. The Tumour-Node-Metastasis (TNM) staging system for malignant tumours defines the disease extent of lung cancer by assessing three components: primary tumour (T); regional lymph nodes (N) and distant metastases (M). T-stage describes the size of the primary tumour and its involvement into nearby tissues (Rosen et al., 2022). Lymph nodes are an important part of the immune system and operate as biological filters for especially cancer cells. The N-stage describe the degree in which the cancer disease has spread from the original tumour site to nearby (regional) lymph nodes. The M-stage describes whether the cancer has spread beyond the region of origin to other parts of the body. Classification via the TNM system guides the prediction

of prognosis, treatment selection and response assessment. The TNM classification with some definitions are summarized in Table 2.1. Note that the T- and M- stages are also divided further into sub-stages. For T-stages these detail primary tumour size and location, while describing distant metastases location and number for M-stages.

Table 2.1: T, N and M stages definitions summary, 8th edition. Adapted from Goldstraw et al., 2016.

T-stage	**Definition**
Tx	Tumour is proven but cannot be visualized by imaging or bronchoscopy
T0	No evidence of primary tumour
Tis	Carcinoma in situ, i.e. malignant tumour cells are present, but has not spread beyond point or tissue of origin
T1-T4	Increasing T-stage for increasing size of primary tumour. Location and local invasion extent also impacts magnitude of T-stage
N-stage	**Definition**
Nx	Regional lymph nodes cannot be evaluated
N0	No spread to regional lymph nodes
N1-N4	Increasing severity of regional lymph node involvement. Which specific group of lymph nodes affected largely determines N-stage
M-stage	**Definition**
M0	No distant metastasis
M1	Confirmed distant metastasis

The individual TNM stages are used to quantify the overall disease progress in a patient by grouping subsets of different TNM stages into what we can refer to as the TNM overall stage. The grouping has a scale of increasing disease severity from stage 0 to stage IV, including substages indicated by a trailing a or b. The TNM overall groups are summarized in Table 2.2. Characteristic of stage III NSCLC is that it presents with significant primary tumour extent and/or regional lymph node involvement. Hence it is often referred to as locally advanced non-small-cell lung cancer (LA-NSCLC).

Treatment

Following the Norwegian Directory of Health's guidelines (Helsedirektoratet, 2021), surgery and radiotherapy are the two main curative treatments of NSCLC in stages I-III. Chemotherapy alone is not curative but can be combined with one or both of the curative options to improve treatment outlook. For stage I, surgery is recommended alone for patients that are operable, while curative radiotherapy is an option for inoperable patients. These options are recommended together with chemotherapy in the case of stage II patients.

Regarding stage III, the optimal treatment approach depends on the independent T- and N-stages. Additionally, with prognostic outlook being poorer within this stage, curative and non-curative intent of treatment needs to be evaluated in relation to survival outlook. Some treatment options for stage IIIa include concurrent and sequential chemo-radiotherapy, or surgery in combination with one of these. With stage IIIb considered inoperable, concurrent or sequential

Table 2.2: TNM stage groupings, 8th edition. Adapted from Goldstraw et al., 2016.

Stage grouping	T-stage	N-stage	M-stage
0	Tis	N0	M0
Ia	T1a, T1b, T1c	N0	M0
Ib	T2a	N0	M0
IIa	T1, T1b	N1	M0
IIb	T1a, T1b, T1c	N1	M0
	T2	N1	M0
	T3	N0	M0
IIIa	T1, T2	N2	M0
	T3	N1	M0
	T4	N0, N1	M0
IIIb	T1, T2	N3	M0
	T3, T4	N2	M0
IIIc	T3, T4	N3	M0
IVa	Any T	Any N	M1a, M1b
IVb	Any T	Any N	M1c

chemo-radiotherapy with curative intent can be considered.

Palliative treatment with either radiotherapy or therapeutics is recommended for patients with either stage IV NSCLC or stage III with poor prognostic outlook. Genetic expressions of the disease are determining factors when deciding treatment methods or specific combinations treatments.

Imaging lung cancer

For identification of the primary tumour in the lungs, CT provides good contrast between the air in the lungs and the surrounding solid tissue, making abnormal lesions stand out. The origin of contrast in CT and planar X-ray imaging will be further explained in section 2.3. Additionally, with a good spatial resolution around 0.35mm (Smith et al., 2011), CT images can be used to accurately asses primary tumour size and extent, including that of local metastases.

Lymph metastases don't necessarily show a difference in size from healthy lymph nodes, hence PET or most commonly integrated PET-CT (Roth et al., 2014) with FDG tracer is used to identify local and distant metastases. This technique works by exploiting the metabolic differences between cancerous and healthy cells to acquire images that measure the relative uptake of the radioactive tracer in different parts of the body.

2.3 Computed Tomography

Computed tomography (CT) imaging uses a rotating X-ray source to acquire image projections at different angles in order to reconstruct a detailed cross-sectional image corresponding to the plane of rotation. By acquiring these cross sections at consecutive axial positions, the images can be stacked together into full three-dimension reconstruction of a patients internal anatomy. CT is an evolution of traditional planar X-ray imaging, which can only capture a two dimensional projected image of the target volume, and is suitable for examining bone structures at excellent resolution but provides limited contrast between different soft tissues.

2.3.1 Production of X-rays

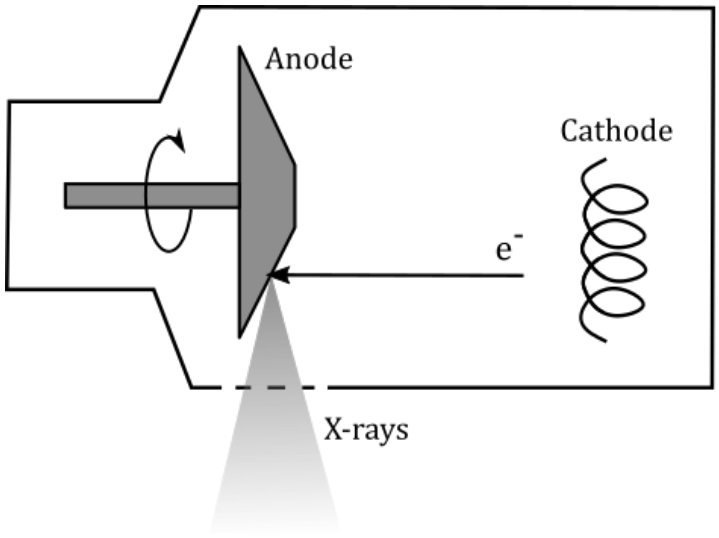


Figure 2.6: X-ray tube schematic.

The source of X-rays used in diagnostic imaging is called an X-ray tube (Figure 2.6), which is an evacuated vessel containing most importantly a cathode and a rotating anode. A tungsten cathode filament is heated by an electric current to a temperature at which electrons are ejected from the metal. These electrons are focused and accelerated towards a positively charged and rotating heavy-metal anode, which is usually made from tungsten or molybdenum. When the accelerated electrons hit the metal target, some of their kinetic energy is converted into X-rays.

There are two different mechanisms behind the production of X-rays when the electrons strike the metal target: bremsstrahlung and characteristic X-rays. Bremsstrahlung is the result of electrons passing close to a metal nucleus and being deflected and decelerated, such that kinetic energy is continuously converted into electromagnetic waves in the form of X-rays. Characteristic X-rays arise from atomic electrons in the metal being excited or ejected by the incoming free electrons, and the subsequent release of photons when they fall back down to a lower energy state.

These two mechanisms can be seen when measuring an X-ray energy spectrum, which is illustrated in Figure 2.7. Note the continuous curve arising from the bremsstrahlung, with sharp spikes superimposed that originate from the characteristic X-rays. The energy levels of atomic electrons are unique for each element, and hence so is the characteristic X-ray spectrum of that element.

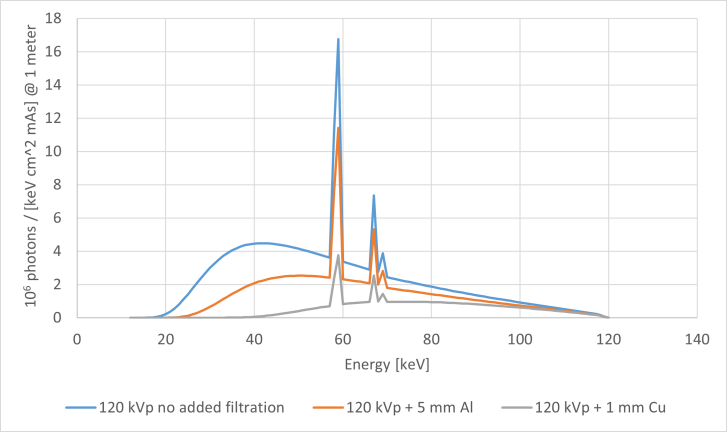


Figure 2.7: X-ray spectrum of differently attenuated beams.

2.3.2 Interaction with organic tissue

X-ray and subsequently CT imaging exploit the differences in photon-attenuation of different tissues in the body in order to construct images. With the energy ranges used in diagnostic X-ray imaging, there are two main interactions that the photons can have with the atoms in organic tissue: photoelectric effect and Compton scattering. These interactions cause the attenuation of X-rays in the body, which can then be described by using the linear attenuation coefficient.

Photoelectric absorption

This interaction (Figure 2.8) is the main source of contrast in diagnostic X-ray imaging, due to the difference in attenuation of different tissues and bone. When X-ray photons travel through the tissues in the body, they can be absorbed by atomic electrons, similarly to the creation of characteristic X-rays on the metal anode in the X-ray tube. However when a photon is released again by an electron dropping to a lower energy level in the atomic shells, that photon will usually have a lower energy and be completely absorbed before ever reaching the detector.

As different tissues and bone have different elemental compositions, they will also attenuate incident X-rays differently, allowing us to distinguish the various tissues by basing image contrast on the amount of X-rays detected. The most important elements present in organic tissue with regards to photoelectric attenuation are carbon, oxygen and calcium (Smith et al., 2011).

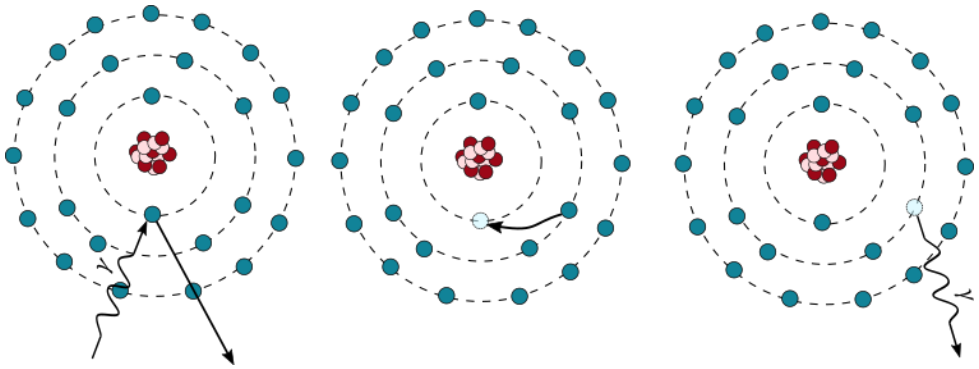


Figure 2.8: **Left**: An atomic electron absorbs an incident photon γ and is ejected. **Middle**: An electron changes energy level from an outer shell to the empty place in the inner shell. **Right**: The difference in binding energy between the two energy states is released as another photon.

The probability of the photoelectric interaction, P_{PE}, is proportional to the incident photon energy E, the effective atomic number Z_{eff} of the tissue, and the tissue density ρ:

$$P_{PE} \propto \rho \frac{Z_{eff}^3}{E^3} \tag{2.1}$$

The effective atomic number of soft tissue is 7.4 and 13.8 for bone, while the respective densities ρ are 1 and 1.85 (Smith et al., 2011), indicative of the good contrast between soft tissue and bone seen in both planar X-ray and CT imaging.

Compton scattering

Figure 2.9 shows the process of Compton scattering. The incident photon γ with energy E_γ interacts with a loosely bound outer atomic electron, such that the electron is ejected from the atom at an angle φ and the photon is scattered at angle θ with energy $E_{\gamma\prime}$ and momentum $p_{\gamma\prime}$.

The standard Compton equation gives the change in wavelength between the incident and the scattered photon (Smith et al., 2011):

$$\Delta\lambda = \frac{h}{m_e c}(1 - \cos\theta) \tag{2.2}$$

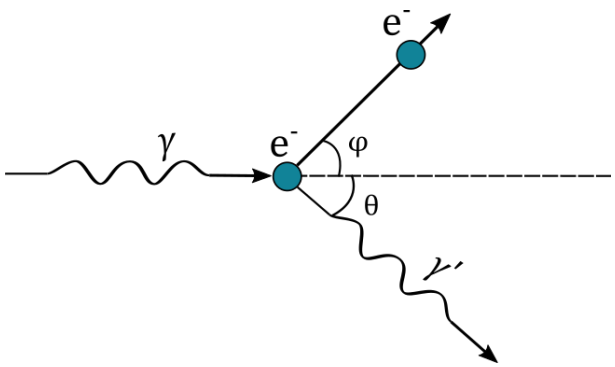


Figure 2.9: Compton scattering

Where m_e is the electron mass. This can then be used to calculated energy difference and thus the energy of the scattered photon:

$$E_{\gamma\prime} = \frac{E_\gamma}{1 + \left(\frac{E_\gamma}{m_e c^2}\right)(1 - \cos\theta)} \tag{2.3}$$

Examining eq. (2.3), note that E_γ is usually in the range 30-120 keV for diagnostic X-ray imaging (Maqbool, 2017), while the electron rest energy $m_e c^2$ is 511 keV. Hence $E_\gamma / m_e c^2$ in the second term of the denominator will be small, and the scattered photon energy is comparable to the incident energy. As a result, the scattered photon will often have enough energy to completely penetrate the human body and reach the detector.

The effect of Compton scattering in imaging is that photons will reach the detector at unintended angles and thus create random noise which will reduce the contrast to noise ratio (CNR) and lowering overall image quality. One solution to this effect is to use anti-scatter grids on the detector, allowing only photons travelling at intended angles to reach the detector.

Tissue linear attenuation coefficient

The total attenuation of X-rays through tissues of the body has been experimentally determined to be exponentially dependent on the distance travelled (Smith et al., 2011). We can express the intensity I of radiation passing through a material of thickness x as

$$I = I_0 e^{-\mu x} \tag{2.4}$$

Where I_0 is the initial radiation intensity and μ is the linear attenuation coefficient of the tissue, which is dependent on tissue type and the X-ray energy. The value of μ is equal to the individual

contributions from the effects that cause the attenuation, which in this case are the photoelectric effect and Compton scattering:

$$\mu = \mu_{PE} + \mu_{Compton} \tag{2.5}$$

2.3.3 CT WORKING PRINCIPLE

By rotating the X-ray source around the patient, we are able to measure the attenuation coefficient at exact positions inside the body and reconstruct an image based on the attenuation. This can be done using a technique called filtered back-projection.

FILTERED BACKPROJECTION

Consider Figure 2.10 as a simple model of a volume we want to image. The squares represent the individual units of volume (voxels) that we can discern at the detector, each having their own linear attenuation coefficient $\mu_{i,j}$ dependant on the tissue type in the voxel.

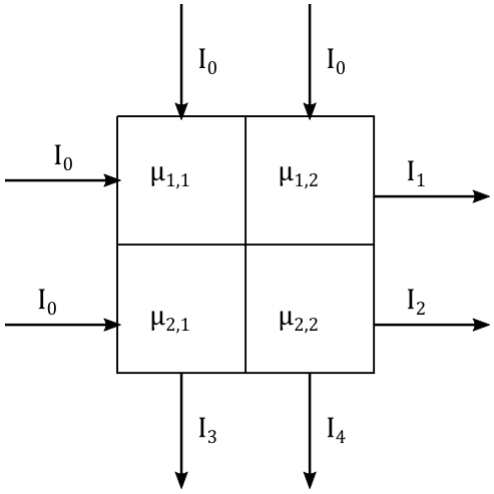


Figure 2.10

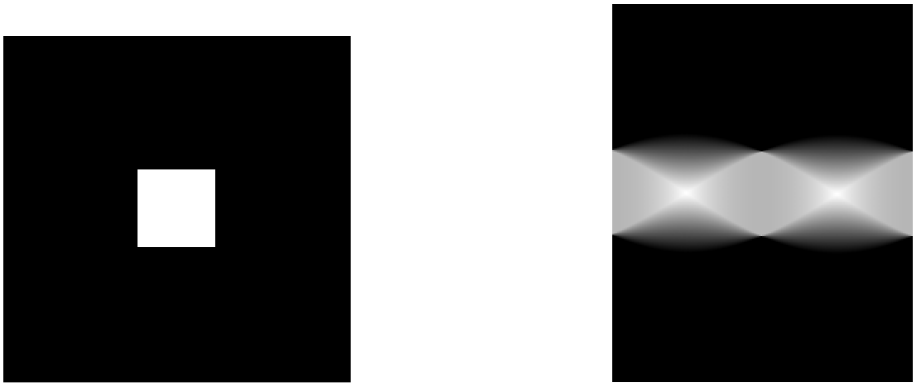
X-rays of initial intensity I_0 are transmitted through the volume, and are registered at the detector as I_1 and I_2 from the left at angle $\theta = 0$, and I_3 and I_4 from the top at angle $\theta = \pi/2$, after having passed through the volume. Let the width of each voxel be w. Considering the top left beam, it will pass through the voxels with attenuation coefficients $\mu_{1,1}$ and $\mu_{2,1}$, and following eq. (2.4):

$$I_3 = I_0 e^{-w(\mu_{1,1}+\mu_{2,1})} \tag{2.6}$$

Which will be the value measured at the detector, and the same logic applying for the other transmitted beams I_1, I_2 and I_4.

From this setup, we can collect the measured intensities in a new type of image known as a sinogram, shown in Figure 2.11. The intensity of pixels (θ, x) in the sinogram is simply the intensity measured at position x along the detector, at angle θ of the source-detector axis relative to some origin.

Backprojection is a technique used for reconstructing an image of the original structure from the measured sinogram. It is done by taking a single column from the sinogram, representing a specific one-dimensional projection on the detector, and repeating it into a complete image, at the angle θ which it was measured. We can think of this as "smearing" the projection across



(a) Original image (b) Sinogram

Figure 2.11: Image of a square and its sinogram[2].

an empty image. All the projections from the sinogram are then added together to form the reconstructed image. The accuracy of this image compared to the true structure that was imaged improves with the number of scan angles and reconstruction projections.

Figure 2.12 shows simple backprojection of a white square using different numbers of projections from the sinogram. This technique is referred to as "simple" due to not modifying the signal in any way before reconstruction. As the number of projections used for the reconstruction approaches the number of scans used to construct the sinogram, 180 in this case, the reconstructed image becomes increasingly accurate.

Note that there is a considerable blurring effect when using simple backprojection, due to the unmodified projection intensity being applied to the entire image and is a major limitation

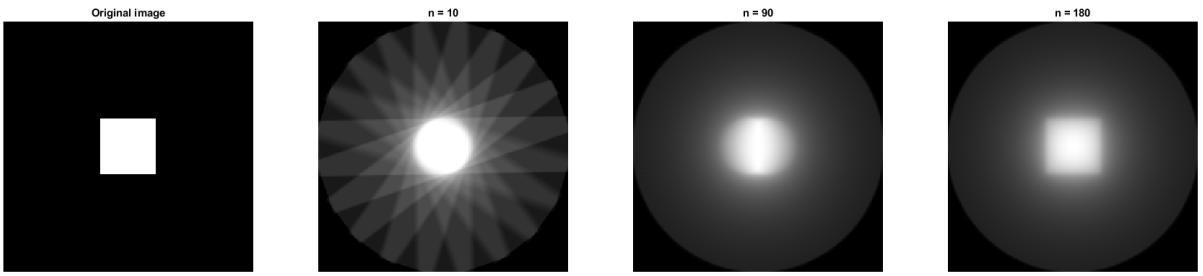


Figure 2.12: Simple backprojection of a square using different number, n, of projections at regular intervals of the sinogram[2].

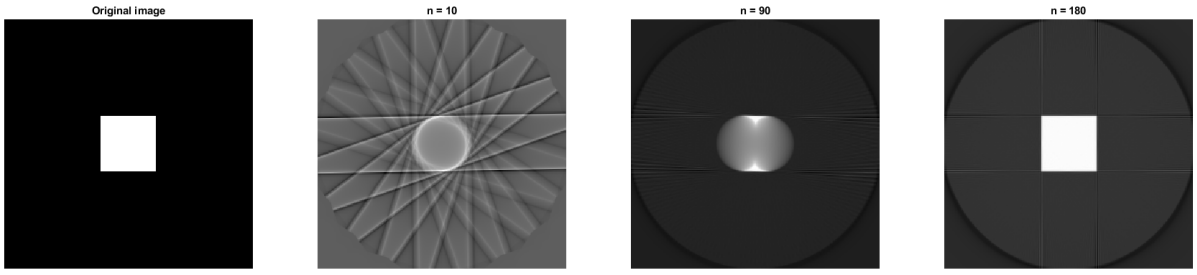


Figure 2.13: Filtered backprojection of a square using different number of projections[2].

[2]Images created with the CT basic reconstruction algorithms package for MATLAB (Sokol, 2021)

of this method. When applied to more complex structures, like the phantom in Figure 2.15, we can see it would be near impossible to distinguish finer details in the application of medical imaging (Figure 2.15c).

The solution to this is the technique of *filtered* backprojection, which applies an edge-enhancing (high-pass) filter to each individual projection, as illustrated in Figure 2.14, before the reconstruction is done. In Figure 2.13, the same square from Figure 2.12 is now reconstructed using filtered backprojection. It is evident from the $n = 10$ case that the individual projections have undergone edge-enhancement, and for $n = 180$ we have achieved a near perfect reconstruction of the original object. The improvement is especially substantial on the anatomically replicating Shepp-Logan phantom in Figure 2.15, where the previously obscured details inside the volume of (c) can now be clearly seen in (d).

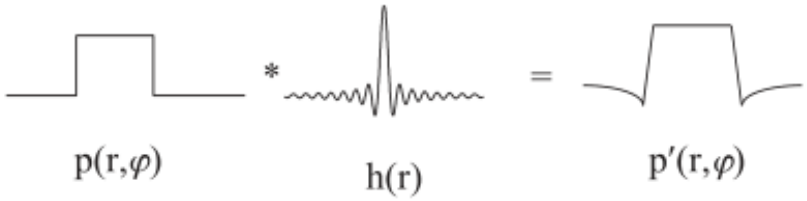
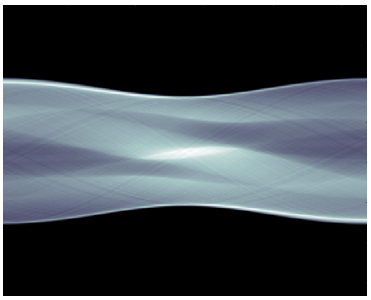


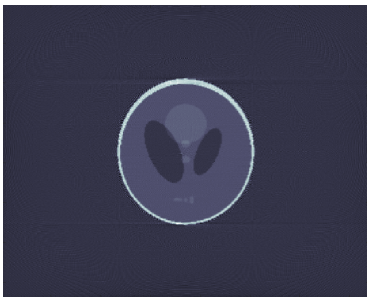
Figure 2.14: A high-pass filter $h(r)$ convoluted with a projection $p(r, \phi)$, resulting in the edge-enhanced projection $p\prime(r, \phi)$. Image from Smith et al., 2011.

(a) Shepp-Logan phantom.



(b) Sinogram of the phantom.

(c) Simple backprojection from the sinogram.



(d) Filtered backprojection from the sinogram.

Figure 2.15: Simple and filtered backprojection on a Shepp-Logan phantom[3].

[2]Images made with the CT reconstruction package for MATLAB (Bangert, 2022).

Hounsfield units

As outlined in the preceding descriptions of linear attenuation of X-rays and back-projection, the attenuation coefficients of a specific materials is key in acquiring CT images. However, we don't use a matrix of the linear attenuation values when displaying CT images. Instead we convert the values to a CT number instead, given in Hounsfield units and abbreviated with HU:

$$\text{HU} = 1000 \cdot \frac{\mu - \mu_{H_2O}}{\mu_{H_2O}} \tag{2.7}$$

Where μ_{H_2O} is the linear attenuation coefficient of water and μ is the coefficient of the specific voxel. HU are normalised to give the same value for the same tissue coefficients independent of the X-ray energy used for the measurement.

Table 2.3: The Hounsfield units of various organic materials, adapted from Smith et al., 2011.

Tissue type	**Hounsfield units**
Bone	1000 to 3000
Muscle	10 to 40
Water	0
Lipid	-50 to -100
Air	-1000
Brain, white matter	20 to 30
Brain, grey matter	35 to 45
Blood	40

From Table 2.3 we can see that bone will have a high HU value and thus appears bright white on CT images, analogous to how we see bone in planar X-ray imaging. Additionally, the air filled lungs will appear dark grey or black with -1000 HU, which makes abnormal lesions of tissue having higher HU stand out inside the lungs.

In translating voxel values from HU into intensity values used in digital images, the *bit depth* of the digital image is an important factor. The bit depth of an image is the number of bits k per voxel, and thus there are 2^k possible grey-scale values for each voxel. The CT numbers can have ranges of several thousand HU that need to be mapped to a k-bit grey-scale digital image. Mapping from HU to grey-scale is controlled by *windowing*, using two parameters called window- width and level. The window width is the range of CT numbers that are mapped to grey-scale while the window level is the centre position of this range.

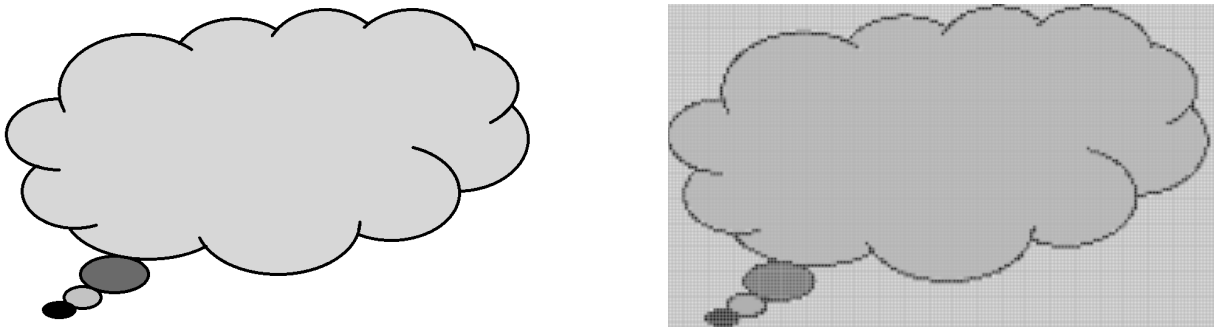
For imaging the lungs, typical window settings can include a window level of -500 and a window width of 1500, such that both air and soft tissue is represented as accurately as possible. While for brain imaging the settings might be level of 40 with a width of 80 in order to capture subtle difference in the soft tissue (Maqbool, 2017).

2.4 Digital images and processing

In this section we will develop the fundamental theory around digital images, and the parts from image processing that can be considered prerequisites to the theory behind radiomics. We will look at how images are represented and stored in the digital domain, and how this allows for the various algorithms and methods in image processing. We will also have a brief introduction to the wavelet transform, which in radiomics is used to analyse specific detail decompositions of an image.

2.4.1 Fundamentals

Let $f(x, y)$ be a mathematical, two-dimensional representation of a real world scene or object, such as the cloud in Figure 2.16a. A digital image is constructed by sampling f into a grid represented by a new function $I(m, n)$, of the discrete variables $m \in \{0, 1, ..., M-1\}$ and $n \in \{0, 1, ..., N-1\}$, seen in Figure 2.16b. The coordinate pairs (m, n) of small, constant-intensity squares that together make up the image, are know as picture-elements, or pixels. The quantity $M \times N$, i.e. the dimensions of the image matrix, is called the resolution of the image. The range of values that $I(m, n)$ can take, depends on the data type that is used to represent the specific image, the two most common being unsigned 8-bit integer (*uint8*) and real floating point number (*float*), where $I \in \{0, 1, ..., 255\}$ and $I \in [0, 1]$, respectively.



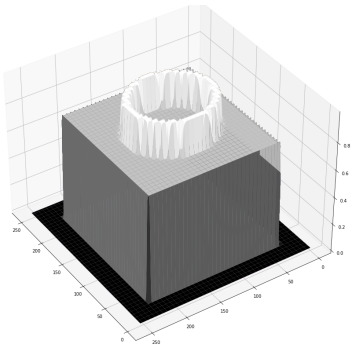
(a) A cloud

(b) The cloud sampled into a discrete digital image

Figure 2.16

A digital image can be represented in multiple ways. The most familiar is the collection of pixels into a grid, where the intensity is proportional to I at a given point, forming a traditional greyscale image, as seen in Figure 2.17a. The lowest values of the image intensity range are displayed as black, the highest as white, while the values in between being different shades of grey depending on their intensity value. Figure 2.17b shows a surface representation of the same image as in 2.17a. Here, the intensity value I is the height of a graph above the coordinates (m, n) in the discrete pixel grid that makes up the xy-plane. In Figure 2.17c is a matrix (or array) representation of the example image, which is simply constructed by arranging the function values of $I(m, n)$ into a matrix, where (m, n) denotes the position of matrix elements. This array representation is the most useful and common for computations and developing algorithms, and is how computers store digital images.

(a) A grey-level image



(b) The same gray-level image seen as the magnitude of $f(x, y)$ over an xy-plane

$$\begin{pmatrix} 0 & 0 & & \dots & & 0 \\ 0 & & \vdots & & & \\ & \dots & 0.7 & 0.7 & \dots & 0 \\ \vdots & & 0.7 & & \vdots & \\ & & \vdots & \dots & 1 & \dots \\ 0 & & & & \vdots & \end{pmatrix}$$

(c) The grey-level image as a matrix

Figure 2.17: Different image representations, made with inspiration from Gonzalez et al., 2007.

2.4.2 Image operations

Images are very often represented on matrix form and we can thus perform the common operations such as matrix- addition and multiplication on them. Additionally in the context of working with digital images on matrix form, it is very useful to perform element-wise multiplication of two matrices, also known as the *Hadamard product* and denoted with a $\circ$. Traditional matrix multiplication between two matrices $\mathbf{X}$ and $\mathbf{Y}$ is calculated as follows:

$$\begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix} = \begin{pmatrix} x_{11}y_{11} + x_{12}y_{21} & x_{11}y_{12} + x_{12}y_{22} \\ x_{21}y_{11} + x_{22}y_{21} & x_{21}y_{12} + x_{22}y_{22} \end{pmatrix} \tag{2.8}$$

which is not commutative, and requires the number of rows in the left matrix to equal the number of columns in the right matrix. The Hadamard product, or element-wise array multiplication is calculated in the following way:

$$\begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{pmatrix} \circ \begin{pmatrix} y_{11} & y_{12} \\ y_{21} & y_{22} \end{pmatrix} = \begin{pmatrix} x_{11}y_{11} & x_{12}y_{12} \\ x_{21}y_{21} & x_{22}y_{22} \end{pmatrix} \tag{2.9}$$

From now on, all arithmetic operations performed on matrices will be element-wise, unless explicitly stated otherwise.

Array multiplication is central to the concept of applying *masks* to images. Masks are binary images, i.e. the pixels have intensity either 0 or 1, that are used to segment out or highlight parts of an image. For example, if we want to only study a specific region of an image, we multiply it with a mask of the same dimensions, where the pixels in positions corresponding to the region of interest (ROI) have intensity 1, and all other pixels have intensity 0. By element-wise multiplication, we get an image where the ROI keeps all of its original intensities, but all pixels outside are set to 0 and will appear black. This process is exemplified in Figure 2.18.

The previous example shows one of several arithmetic operations between two images. But there is also a class of operations which concern themselves with the manipulation of individual pixels in a single given image, referred to as spatial operations. These can be categorized further into single-pixel operations, neighbourhood operations, and geometric spatial operations (Gonzalez et al., 2007).

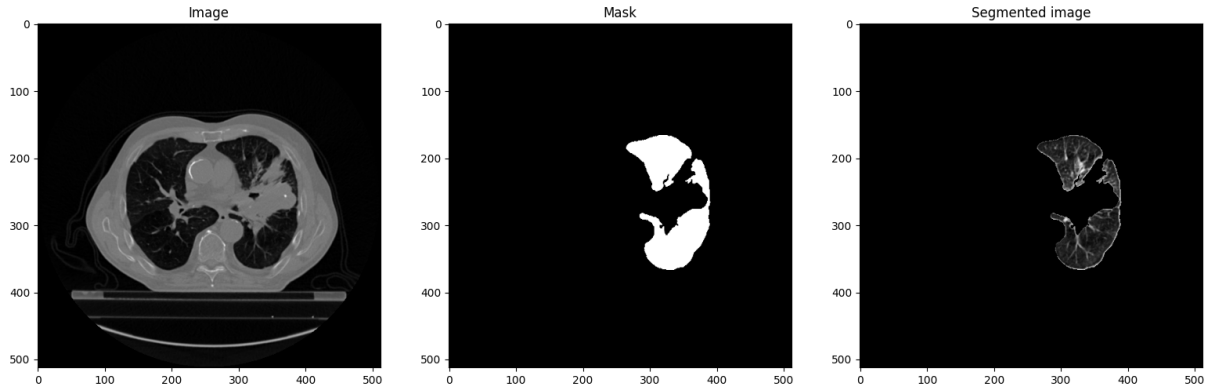


Figure 2.18: Left: A CT cross-section of a patient's abdomen. Middle: A binary mask of the left lung. Right: The result of multiplying the image and the mask.

Single-pixel operations manipulate the individual pixel intensities of an image. We can say that, for a pixel at coordinates (m, n) with intensity $p(m, n)$, the transformed pixel intensity $q(m, n)$ is given by

$$q(m, n) = T(p(m, n)) \tag{2.10}$$

where T is the transform function. We choose T depending on the goal of the intensity transformation. For example, if T is a logarithmic function, it will expand the range of dark pixels in an image, while restricting the variance in lighter values. Or if we want to highlight a specific range of intensities by mapping them to white, we can have a linear function with a *band* of values which map to the maximum intensity. A graph of such a function is shown in Figure 2.19, where intensities of the input and output pixels are given on the x- and y-axis, respectively.

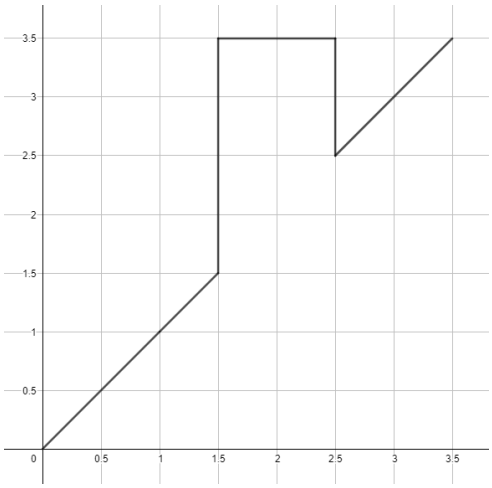


Figure 2.19: A function which highlights the intensities in the range $(1.5, 2.5)$. The intensity range is arbitrary and it is the shape of the graph that is important.

Neighbourhood operations give the intensity of a new pixel $q(m, n)$ in the output image dependent on some neighbourhood $S_{m,n}$ around the corresponding pixel $p(m, n)$ in the input image. A very common application of this is an averaging filter, where the value of q is given by the average of all pixel intensities in a square $n \times n$ neighbourhood around p. We can more

generally express this as the following equation:

$$q(m,n) = \frac{1}{A} \sum_{(u,v)\in S_{m,n}} p(u,v) \tag{2.11}$$

Where A is the area (i.e. number of pixels) of the neighbourhood $S_{m,n}$ around the pixel p, and $I(u,v)$ is the intensity at position (u,v) in the input image. The effect of taking a local mean of every pixel in an image is that it will lose some detail and appearing blurred, while potentially reducing the effects of random noise in the image.

Geometric spatial transformations, in contrast to the two previously discussed categories, affect the relative spatial distribution of pixels in the image. They are analogous to what we would consider to be mathematical transformations of an object in 2D-space, e.g. translation and rotation. We can say the new coordinates (m,n) in the output image are given by the transformation T on the old coordinates (u,v) in the input image:

$$(m,n) = T[(u,v)] \tag{2.12}$$

Note that in this case, T is a function of pixel position (u,v), and not of pixel intensity p at that position, as is the case for eq. (2.10). *Affine* transformations are one of the most common groups of geometric spatial transforms, and they are defined by the general expression (Gonzalez et al., 2007)

$$\begin{pmatrix} m & n & 1 \end{pmatrix} = \begin{pmatrix} u & v & 1 \end{pmatrix} \mathbf{T} = \begin{pmatrix} u & v & 1 \end{pmatrix} \begin{pmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{pmatrix} \tag{2.13}$$

Where $\mathbf{T}$ is a transformation matrix, where the values of its elements t_{ij} decide the type of transformation that it performs on the coordinates $\begin{pmatrix} u & v & 1 \end{pmatrix}$. An important property of 2D affine transformations is that they preserve points, straight lines and planes. A few examples of common affine transforms are given in Table 2.4.

Table 2.4: Some familiar affine transformations

Identity	Scaling	Translation	Reflection (vertical)	Rotation
$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{pmatrix}$	$\begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$

In performing a spatial transformation, the initial pixel coordinates (u,v) are moved to the new location (m,n), but an appropriate intensity value must also be assigned to transformed pixel. This can be done by various interpolation techniques, which use the intensities of the neighbouring pixels to (u,v) to assign an intensity value to (m,n).

2.5 Calculating radiomic features

Radiomic features are commonly divided into different feature classes according to their mathematical definition and which aspects of the ROI that they describe. The feature classes most relevant for this work are: first-order, shape, texture and wavelet features. There exist more groupings as well, but these are the ones most commonly featured in recent publications and an ongoing standardization initiative (Zwanenburg et al., 2020a). There has also been made an interclass distinction between so-called "semantic" and "agnostic" features (Gillies et al., 2016). Semantic features are the computerized quantification of the descriptors already used by radiologists in the clinic, e.g. shape and size. Agnostic features are those not necessarily within regular clinical vocabulary, nor practical utility of the clinic, but can provide previously unknown insight given the proper analytical tools.

2.5.1 First-order features

Also known as histogram features, these values are calculated from the histogram of voxel intensities in the ROI. Describing the distribution of intensity values using first-order statistics, but not accounting for positional relationships of the voxels. Some examples include common first-order statistics such as 10th- and 90th percentiles, mean and variance. In addition there are some more sophisticated features such as kurtosis and skewness, which describe the actual shape of the ROI histogram (Mayerhoefer et al., 2020).

Examples Let $\mathbf{X} \in D \subset \mathbb{R}^3$ be an image matrix containing N voxels, and D be the image domain. As position in the matrix is irrelevant, we can leave X_i to be the intensity of the i-th voxel in the matrix.

$$\text{mean} = \frac{1}{N}\sum_{i}^{N} X_i \tag{2.14}$$

$$\text{energy} = \sum_{i}^{N} (X_i)^2 \tag{2.15}$$

The mean is as we know it from statistics, while the energy gives a measure on the total amount of intensity in the ROI.

2.5.2 Shape-based features

The description of a tumour's shape is already central within radiology, where volume and local invasion extent plays large roles in diagnosis and treatment of lung cancer. Shape features are an effort to quantify the three-dimensional size and shape of the ROI, making them so-called semantic features. This group moreover includes less clinically relevant metrics relating to shape such as tumour sphericity and elongation. Analogous to first-order features, the mathematical definitions of shape features rely on the common shape and size descriptors of volume and surface area. Note that shape features are not dependent on voxel intensities and hence in practice only the ROI segmentation mask is need for their calculation.

Examples Let V and A be the respective calculated volume and surface area of an ROI and N the number of voxels in the segmentation. Then (Zwanenburg et al., 2020a),

$$\text{sphericity} = \frac{\pi^{\frac{1}{3}}(6V)^{\frac{2}{3}}}{A} \tag{2.16}$$

$$\text{compactness2} = 36\pi\frac{V^2}{A^3} \tag{2.17}$$

Both sphericity and compactness compares the shape of the ROI to that of a sphere (the most compact shape) and hence are strongly correlated to each other. Both volume and surface are can be calculated either via the voxel-volume or the mesh-volume of the segmented structure. The former is calculated as N times the single-voxel volume, while the latter uses a mesh technique where the ROI is represented in 3D space as a surface of connected polygons. For large volumes with hundreds or thousands of voxels the two techniques give practically equivalent calculation, but for smaller volumes (tens to hundreds of voxels) voxel-volume will often overestimate the volume compare to the mesh-based method (Zwanenburg et al., 2020a).

2.5.3 Texture features

Texture features measure the spatial relationships between pixel intensities, in contrast to the first-order features which only provide information on the distribution of grey-levels in an image. There are several ways to describe texture and define texture features, but the one most relevant for this project is the grey-level run-length matrix (GLRM). This matrix was originally designed to investigate differences in terrain depicted on aerial photographs in a 1975 paper (Galloway, 1975).

In the GLRM, denoted $\mathbf{R}(i, j|\theta)$, the matrix element (i, j) is the frequency of intensity value i, occurring j consecutive times in a run along the direction θ in the image matrix (Galloway, 1975). As we are considering discrete intensity levels, the images will therefore need to be discretised by resampling the image pixels into equally spaced bins according to intensity values.

As an example, take the image, $\mathbf{M}$, represented by the following matrix:

$$\mathbf{M} = \begin{pmatrix} 5 & 1 & 3 & 4 & 5 \\ 5 & 1 & 1 & 1 & 2 \\ 2 & 4 & 3 & 3 & 1 \\ 2 & 2 & 3 & 2 & 1 \\ 2 & 5 & 4 & 2 & 2 \end{pmatrix} \tag{2.18}$$

Suppose we want to find the horizontal ($\theta = 0$) GLRM. The first element, i.e. in position $(1, 1)$, will be the number of times pixels with intensity of 1 appear 1 time in succession along the horizontal direction, which we can se happens one time each in the first, third and fourth row. Also, two consecutive values of 3 appear only a single time and thus element $(3, 2)$ has the value 1. In this way we calculate the following GLRM:

$$\mathbf{R}(0) = \begin{pmatrix} 3 & 0 & 1 & 0 & 0 \\ 4 & 2 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{2.19}$$

This definition is expanded straightforward to three dimensions, with θ being each of the 13 directions around a voxel in three-dimensional space.

Thus, the grey-level run-length matrix gives a directionality-based measure of texture in the ROI. There are also other matrices that quantify the texture of an image, such as the grey-level

co-occurrence matrix (GLCM) and grey-level size-zone matrix (GLSZM), which measures the number of pixel intensity pairs along different directions and in interconnected neighbourhoods, respectively (Mayerhoefer et al., 2020).

Once we have obtained the grey level run-length matrix, we can then calculate the textural feature-values, which use the number of runs and intensities found in the GLRLM as central metrics in the feature definitions for describing texture. As the texture matrix is dependent on direction, there can be up to 13 different texture matrices for a 3D image that are need for the calculation of features. In *PyRadiomic*, the standard method for combing the results from each of these texture matrices is to calculate the feature value on each separately and then use the mean of these as the final feature value.

Examples Let $p(i,j|\theta)$ be the (i,j)th element in the grey level run-length matrix $\mathbf{R}$ for the direction θ, N_g the number of discrete intensity levels, and N_r the number of run lengths (Galloway, 1975).

$$\text{Grey level non-uniformity} = \frac{\sum_{i=1}^{N_g}\left[\sum_{j=1}^{N_r} p(i,j|\theta)\right]^2}{\sum_{i=1}^{N_g}\sum_{j=1}^{N_r} p(i,j|\theta)} \tag{2.20}$$

$$\text{Short run emphasis} = \frac{\sum_{i=1}^{N_g}\sum_{j=1}^{N_r}\left[\frac{p(i,j|\theta)}{j^2}\right]}{\sum_{i=1}^{N_g}\sum_{j=1}^{N_r} p(i,j|\theta)} \tag{2.21}$$

2.5.4 Wavelet features

Wavelet features are in reality a subgroup of the *transform-based* feature group, but we focus on wavelet features as they are of direct importance later on. Wavelet features are extracted from the so-called wavelet decompositions of an image, which are the output of a wavelet-transform.

The wavelet transform, by applying high-and low-pass filters in succession on the rows and columns of a two-dimensional image I, yields four *wavelet decompositions* of the original image: I_{LL}; I_{LH}; I_{HL}; I_{HH}, which are shown in Figure 2.20. Where the subscript of the decomposition denotes the combination of high-pass (H) and low-pass filtering (L) applied to the original image I. So for example, I_{LH} is the decomposition resulting from high-pass filtering in the x-direction (rows), followed by low-pass filtering in the y-direction (columns). On a three-dimensional image, the transform yields eight decompositions, one for every H and L combination in the tree spatial directions.

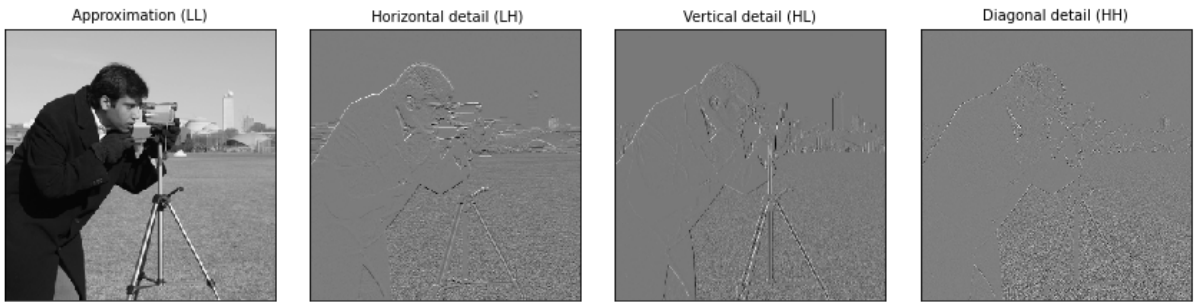


Figure 2.20: Two-dimensional wavelet decompositions of an image. The approximation decomposition on the left (LL), and the three detail compositions on the right (LH, HL, HH)[5].

[5]Made with the pywavelets package for python, https://github.com/PyWavelets/pywt/tree/master/pywt

The different resulting decompositions are often downsampled in the application of image compression, and highlight different aspects of the original image that was transformed. I_{LL} will be a version of the original image that is downsampled by a factor of two, also referred to as the approximation decomposition. The remaining I_{LH}, I_{HL}, I_{HH} highlight the horizontal, vertical and diagonal local intensity changes respectively, and are known as the detail decompositions. Note that a wavelet transform can be applied to an image without downscaling of the image resolution, known as an undecimated wavelet transform (Aerts et al., 2014), yielding decompositions of the same resolution as the original image.

Thus, wavelet features are said to be any features, e.g. the first-order or texture features we have previously defined, that are calculated on the wavelet decompositions of an image. Note that shape features are in calculated from the segmentation mask of a three-dimensional image, and are not affect by the changes in individual pixel intensity that wavelet transforms incur. Wavelet features are used to increase the dimensionality of the minable feature vectors, with the hypothesis that they can distinguish certain edge- or frequency details in the ROI that may be of predictive value.

2.6 Radiomics for phenotypic and prognostic classification

In 2014, Aerts et al. published what can be considered a seminal study for the field of radiomics, inducing great interest in its many applications. By analysing several large cohorts of patients with lung- or head-and-neck cancer they found a large number of radiomic features with prognostic power across independent datasets and moreover between different cancer types, providing evidence that radiomics potentially capture phenotypical characteristics in tumours. Additionally through radiogenomic analyses they also associated a radiomic signature describing intra-tumour heterogeneity with underlying gene-expression patterns, in addition to survival outcome.

Their workflow on the patient cohorts included in the study is shown in Figure 2.21. Using

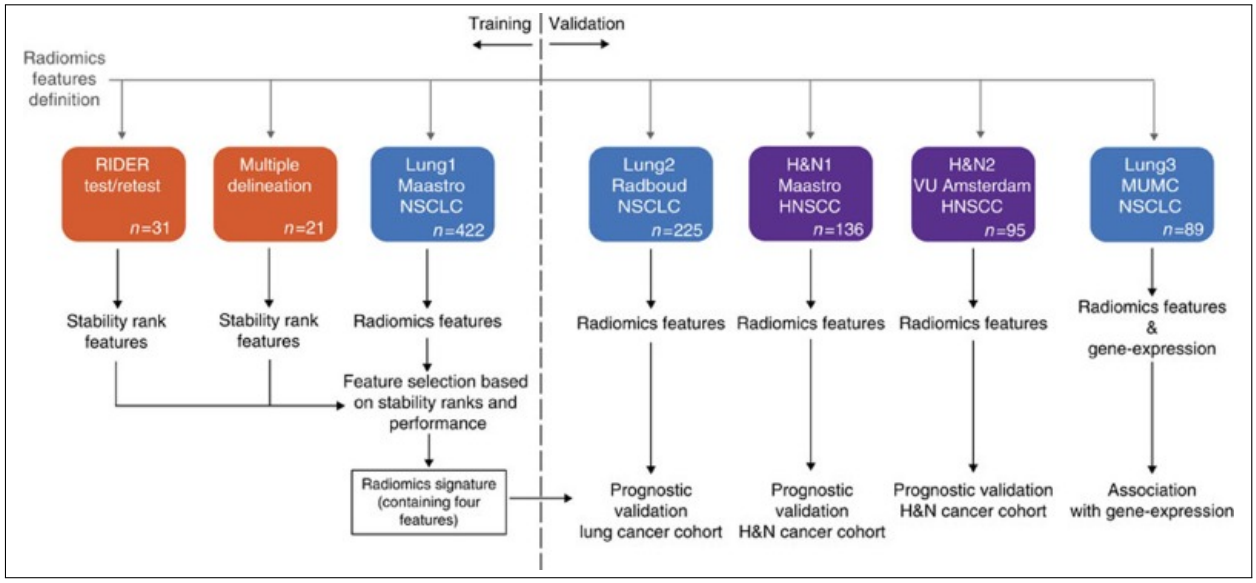


Figure 2.21: The workflow in the study by Aerts et al. and its relation to patient cohorts. From Aerts et al., 2014.

440 predefined quantitative image features describing tumours via: intensity, i.e. first-order features; shape; texture; and multiscale wavelet. They first evaluated the stability of these features with regards to test-retest CT scans and different delineations by independent radiation oncologists on the *RIDER* ($n = 31$, NSCLC) and *Multiple delineation* ($n = 21$, NSCLC) CT datasets respectively. All features were ranked from the most (rank 1) to the least stable (rank 440) feature based on extraction from these two cohorts.

The same features were then extracted from the Lung1 dataset and analysed using unsupervised clustering, revealing three main clusters of patients with similar radiomic expression patterns. Comparing these clusters with clinical parameters (Table 3.1) showed significant association with T-stage and overall TNM stage, and histology with squamous cell carcinoma showing a higher presence in one of the clusters, while N- and M-stages showed no relation to the expression patterns.

To investigate the prognostic performance of the radiomic features, the authors first performed Kaplan-Meier (KM) survival analysis on features extracted from Lung1, H&N1 and H&N2 cohorts. The median value of each feature was calculated from Lung1 and used as a threshold on the validation sets to ensure a completely independent validation without the need for retraining. In short, the median threshold derived from Lung1 gave a significant survival difference for 54%, 38% and 42% of the features on the Lung2, H&N1 and H&N2 validation cohorts respectively, and 66 (15%) of the features showed a significant survival difference in all three validation sets.

They also built a so-called prognostic radiomic signature from the Lung1 dataset, intended for reducing the dimensionality of the hundreds of features into a quantity which can be used generally for the assessment of a patient's prognosis. In summary, this signature consisted of the single most stable feature from each of the categories first-order, shape, texture and wavelet texture. The four features comprising this radiomic signature are summarised in Section 2.6. These were then used in a Cox proportional hazards regression model for prediction of survival, which was trained on the Lung1 cohort and validated on the Lung2, H&N1 and H&N2 cohorts using the concordance index (CI). The validation scores of the model on these data sets are shown in Table D.1. The signature showed what they describe as a good performance on the Lung2 set (CI = 0.65, $p = 2.91 \times 10^{-9}$), and high performance on H&N1 (CI = 0.69, $p = 7.99 \times 10^{-7}$) and H&N2 (CI = 0.69, $p = 3.53 \times 10^{-6}$) (Aerts et al., 2014).

Table 2.5: The four features comprising the prognostic radiomic signature in Aerts et al., 2014.

Energy	**Compactness**	**Grey-level Non-uniformity**	**HLH Grey-level Non-uniformity**
The sum of the squared intensities, see eq. (2.15)	Describing how compact the tumour shape is, also related to how spherical it is, see eq. (2.17)[4]	A measure of intra-tumour heterogeneity, calculated from the GLRLM, see eq. (2.20)	The grey-level non-uniformity of the HLH wavelet decomposition of the image

In comparing with the performance of volume and TNM-staging, they show that the radiomic signature performs in general better than these metrics alone. Furthermore, when combining the signature with volume and TNM-staging separately, the performance was comparable to that of the signature alone in all validation sets.

The large scope and exhaustive methodology of this study sparked great interest in the

[4]Referred to only as "Compactness" in the main article, but the supplementary information shows it is specifically Compactness2 as defined in eq. (2.17)

field. A common trend in subsequent studies is the use of stability as a consideration during the feature selection stage, and designing a model on a "signature" consisting of the best performing features both with regards to stability and predictive capabilities. Shortly after, Leijenaar et al., 2015 externally validated the signature on a large independent H&N cancer cohort and finding similar promising results for this signature.

3

Materials and method

In this thesis, a system for sorting individual patients with the related metadata and parameters into customizable groups was implemented in Python by the candidate and further adapted to be used with the *PyRadiomics* package for calculation of radiomic features. Data for the project consisted of two patient populations with associated CT images, where one is openly available from the The Cancer Imaging Archive (TCIA) and the other was provided by Haukeland University Hospital (HUH). Both datasets underwent manual quality assurance prior to radiomic feature extraction, ensuring consistency in the image material and thus correct calculation of quantitative image features. The QA process and discovered problems with the data are described further in section 3.2.1.

Calculated features were tabulated and sorted according to patient groups and the feature categories outlined in section 2.5. The independently calculated features from the Lung1 cohort was then used to evaluate the implementations ability to calculated radiomic features by comparing derived results to those of Aerts et al. The same process for radiomic feature calculation was then employed on the HUH data to investigate if the methodology in Aerts et al., 2014 is transferable to an independent dataset. The entirety of the Python code for this project is shown Appendix A and openly available in the following github repository along with calculated feature data: https://github.com/filipjb/NSCLC_radiomics.

3.1 Datasets

Lung1

The Lung1 dataset is a collection of FDG PET-CT images of 422 NSCLC patients (stages I to III) from the Maastro clinic in Maastricht, The Netherlands. The image data along with clinical parameters (Table 3.1) for each patient is openly available on The Cancer Imaging Archive (TCIA)[1]. The Lung1 dataset as referred to in this thesis is not exactly identical to the one uploaded to TCIA, as some patients with unfit or corrupted image material were excluded from this project. This is elaborated on further in section 3.2.1. The Clinical statistics of the Lung1 set used for this project is provided in Table 4.1a.

The following information is retrieved from the supplementary information to the article by Aerts et al. The patients underwent treatment with either radiotherapy alone ($n = 196$), or with chemo-radiotherapy ($n = 296$). The study was approved by the institutional review board and carried out in accordance with Dutch law. The FDG PET-CT images were taken on Siemens

Table 3.1: Clinical parameters for patients in Lung1

`Patient ID`	`Age`	`T-stage`	`N-stage`	`M-stage`
`Overall TNM-stage`	`Histology`	`Gender`	`Survival time`	`Survival status`

[1] https://wiki.cancerimagingarchive.net/display/Public/NSCLC-Radiomics

Biograph (SOMATOM Sensation-16 with an ECAT ACCEL PET SCANNER) for radiotherapy treatment planning, with a spiral CT with or without intravenous contrast covering the complete thoracic region. The gross tumour volume (GTV) and selected anatomical structures were delineated as a 3D volume (Patil, 2020) by a radiation oncologist on fused PET-CT images. All images in this set had 3mm slice thickness and 512×512 resolution with $0.977 \times 0.977\text{mm}^2$ listed as pixel dimensions.

HUH

The dataset provided by Haukeland University Hospital (HUH), hereby referred to as the HUH set, consisted of image material and clinical data from a cohort of LA-NSCLC patients treated with radiotherapy and either concurrent or sequential chemotherapy at HUH from October 2019 to August 2021. The data was collected as part of a research project approved by the Regional Committee for Medical Research Ethics of Western Norway (REK 2019/749), where all patients gave informed consent prior to participation. All DICOM data related to each patient was anonymized before being provided to the candidate. A limitation of this dataset is that follow-up data on survival time and death are unavailable. Subsequently only comparisons to the calculated feature values in Lung1 could be made, and not survival trends. With the exception of survival time and survival status, all clinical parameters in Table 3.1 also accompanied HUH.

All imaging for the study was performed at HUH on a Big Bore CT scanner (Phillips Healthcare) using a scanning protocol designed for LA-NSCLC treatment planning (Appendix B) using the settings shown in Table 3.2. The pixel dimensions in each slice were given as 0.9766×0.9766 mm^2.

Table 3.2: CT acquisition parameters for HUH set.

Peak potential	120kVp
Filter Type	Standard (B)
Scan option	Helix
Slice thickness	2mm

For this set, 10-phase free breathing CT images (4DCT) taken for free breathing (FB) radiotherapy planning were used as the basis for contours on CT images taken with contrast (CT thorax). The internal GTVs (IGTVs), divided into primary (IGTVp) and nodal (IGTVn) tumour volumes, included the union of all GTV positions at all ten phases of the 4DCT, and were delineated by the responsible oncologist following ESTRO guidelines (Nestle et al., 2018). All images have the same 512×512 resolution as the Lung1 set, but differ by having 2mm slice thickness instead of 3mm.

3.2 Implementation

For handling a potentially large number of patients with large amounts of image-, clinical-, and metadata, an object-oriented approach in Python was used to organize patients and associated data by creating two custom classes. Appendix A.1 shows the complete code for these classes.

The `Patient` class collects all data related to a specific patient including the clinical parameters outlined in Table 3.1, potentially relevant metadata from the patient's DICOM-files, and furthermore returning images and segmentations to the use. Due to the differences in both directory- and DICOM-structure in the two datasets, three methods were implemented into the

class for returning image material to the user. `get_TCIA_images` and `get_TCIA_segmentations` is used to read and return Lung1 images and segmentations, respectively. `get_haukeland_data` handles retrieval of images and segmentations simultaneously, as the file structure of the HUH set allowed for this. The `StudyGroup` class then collects all patients within a specific dataset, e.g. Lung1, such that statistics, and potentially radiomic calculations, can be performed on the entire patient population collected in an instance of that object.

3.2.1 Quality control of image material

The functions in *PyRadiomics* for calculating radiomic features take pairs of three-dimensional image- and binary mask arrays as input. Thus, before feature calculation, quality control was performed to ensure that the data had been translated properly from the DICOM format into image array representations in Python compatible with the software.

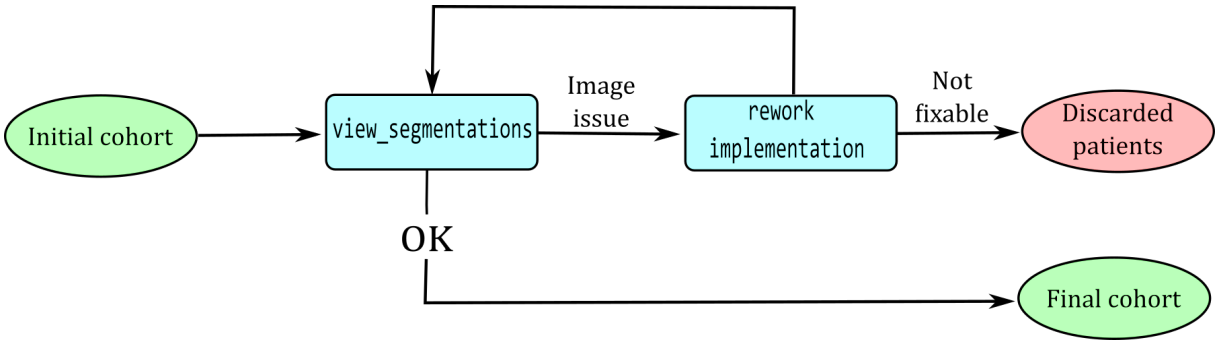


Figure 3.1: Workflow for manually controlling the state of the image material.

The candidate's primary supervisor was responsible for the following quality assurance and exclusion of patients from the initial HUH cohort ($N = 33$). Two patients were excluded due to only having a mediastinal target volume and no GTV, thus radiomic features could not be calculated from the tumour volume. Three patients had tumours that changed substantially in position between the various CT acquisitions, making translation of the IGTV from the 4DCT to CT thorax unviable and thus these patients were excluded. Finally three more patients were excluded due to incomplete delineations, making the final dataset provided to the candidate consist of 25 patients.

To ensure agreement between CT images and segmentations, a function[2] (Appendix A.4) which allows scrolling through three-dimensional images in a conventional `matplotlib.pyplot` window was implemented into the `Patient` class. This function was used by the candidate to view every patient in both image sets and control that segmentations were aligned correctly (Figure 4.1) with easily identifiable anatomical structures, e.g. the lungs, indicating that both CT image- and segmentation matrices had been returned correctly by the appropriate `Patient`-functions. For patients where this was not the case, the issue with the data was categorized and evaluated if it could be solved by adding functionality to the code. If the issue was of a nature where it could not be solved on a general basis, the patient was excluded from the final dataset. The workflow for the quality assurance of images is shown in Figure 3.1.

3.2.2 Feature calculation

Calculation and sorting of feature data was performed using the same process for both datasets, employing the workflow shown in Figure 3.2. Appendix A.2 shows the code written by the

[2]https://matplotlib.org/3.3.0/gallery/event_handling/image_slices_viewer

candidate for using *PyRadiomics* together with the functionality from `Patient` and `StudyGroup` to extract features on the two patient cohorts. It contains functions for calculating each of the following feature types for all patients in a provided `StudyGroup`: first-order, shape, grey-level co-occurrence matrix (GLCM), grey-level run-length matrix (GLRLM) and the GLRLM after being decomposed by an HLH wavelet transform.

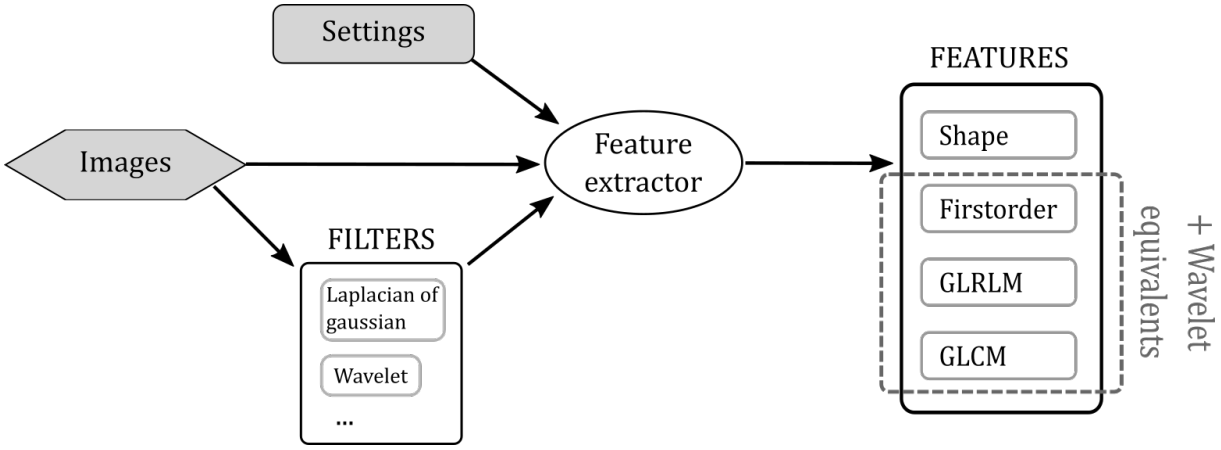


Figure 3.2: Workflow of the implementation using *PyRadiomics*.

These functions take in a `StudyGroup` instance and relevant parameters for adjusting dataset-specific settings for handling the image files. Logging to a .txt file is set up, and feature values of the function-specific feature-type are calculated for each patient in the group, which are then tabulated and saved to a .csv file. Due to correlation with other features, as discussed in section 2.1, specific features are disabled by default in *PyRadiomics* as they can be derived from other features. Some of these disabled features were manually enabled to maintain consistency with the work done by Aerts et al. and especially with regards to the radiomic signature that they developed, which contains the initially disabled feature Compactness2. Two preprocessing steps were done to the CT images prior to feature calculation. Shifting the intensity range to start at 0 for all images, to be consistent across all patients in both cohorts is done by the image-retrieving functions in the `Patient` class. Furthermore, resampling of voxel intensities into equally spaced bins of 25 HU is performed by the *PyRadiomics* feature extractors.

All feature values were calculated on the primary tumour volume in both Lung1 and HUH, labelled GTV-1 and IGTVp respectively in the segmentation data. Table 3.3 lists all features calculated within each class, with a total of 73 feature values being calculated from the primary tumour volume of each patient. The feature extractor calculates GLRLM features on the texture matrices in all 13 spatial directions and returns the mean of these as the feature value. For calculating wavelet features, the CT images from the patients were first transformed by a three-dimensional undecimated (stationary) wavelet transform using the `stwn` function with the "coif1" wavelet from the pywavelets package. The transformed images were then passed together with the segmentations masks into the feature extractors following the standard workflow (Figure 4.3). Note that, as discussed in section 2.5, wavelet features are the same features within a certain class, only that they are calculated after the image has been transformed. Thus the GRLM features and the wavelet HLH GLRLM share the same name, but are distinguished by the prefix HLH. As the wavelet-part of the prognostic signature in Aerts et al., 2014 is from the HLH-decomposed glrlm feature class, only the HLH GLRLM subset of wavelet features was calculated due to time considerations.

Table 3.3: Features computed on each patient for both groups, N = 90.

First order, n = 19
10Percentile, 90Percentile, Energy, Entropy, InterquartileRange, Kurtosis, Maximum, MeanAbsoluteDeviation, Mean, Median, Minimum, Range, RobustMeanAbsoluteDeviation, RootMeanSquared, Skewness, TotalEnergy, Uniformity, Variance, StandardDeviation
Shape, n = 15
Elongation, Flatness, LeastAxisLength, MajorAxisLength, Maximum2DDiameterColumn, Maximum2DDiameterRow, Maximum2DDiameterSlice, Maximum3DDiameter, MeshVolume, MinorAxisLength, Sphericity, SurfaceArea, SurfaceVolumeRatio, VoxelVolume, Compactness2
GLCM, n = 24
Autocorrelation, ClusterProminence, ClusterShade, ClusterTendency, Contrast, Correlation, DifferenceAverage, DifferenceEntropy, DifferenceVariance, Id, Idm, Idmn, Idn, Imc1, Imc2, InverseVariance, JointAverage, JointEnergy, JointEntropy, MCC, MaximumProbability, SumAverage, SumEntropy, SumSquares
(HLH) GLRLM, n = 16×2
GrayLevelNonUniformity, GrayLevelNonUniformityNormalized, GrayLevelVariance, HighGrayLevelRunEmphasis, LongRunEmphasis, LongRunHighGrayLevelEmphasis, LongRunLowGrayLevelEmphasis, LowGrayLevelRunEmphasis, RunEntropy, RunLengthNonUniformity, RunLengthNonUniformityNormalized, RunPercentage, RunVariance, ShortRunEmphasis, ShortRunHighGrayLevelEmphasis, ShortRunLowGrayLevelEmphasis

3.3 Quality control of implementation

To ensure that radiomic features had been calculated correctly by the implementation, they were used to reproduce selected results from Aerts et al., 2014. Only methods and results involving exclusively the Lung1 cohort were reproduced, as the datasets used for stability ranking and validation were not included as a part of this thesis.

In their original publication, Aerts et al. presented Kaplan-Meier (KM) curves for the survival probabilities of individuals with high and low feature values for each of the four features in the proposed radiomic signature: Energy; Compactness2; Gray-level non-uniformity (GLNU); HLH Gray-level non-unifomrity. There is however no raw source data available for the survival model or any of the plots that were presented in the article, so an external method for quantitatively comparing results was employed. The online tool *WebPlotDigitizer* (Rohatgi, 2021) was used to extract the coordinates from images of the Kaplan-Meier curves for the aforementioned four features, and for the combined radiomic signature, which were presented in the original publication. Details on the use of this tool are shown in Appendix C.1. These coordinates enabled plotting of the KM-curves from the article together with those resulting from the implementation's calculations (Section 4.2.1).

The authors' creation of a Cox proportional hazards (Cox-PH) regression model for prediction of survival on the four-feature signature in the Lung1 cohort was also replicated. In fitting this model, individuals were split according to their feature value being above or below the group median, as was done in the KM-analysis. The importance of the individual features on hazard

(survival probability) is evaluated by their regression coefficients, while the performance of the overall model combining all four features is evaluated using a log-likelihood ratio test and the concordance index. The regression coefficients from the Cox-PH model was used as weights in a linear combination of the four features into a single signature feature value for each patient. KM-curves of this total signature for patients in Lung1 were compared (Figure 4.9) to those from the article using the same method as for each of the individual features shown in Section 4.2.1. The proportional hazards assumption of the Cox-PH model was confirmed to be satisfied by the built-in functionality of the lifelines[3] Python package.

For investigating dependencies between the radiomic signature and primary tumour volume, each of the four features and the combined signature were plotted against tumour volume (Figure 4.10) for both Lung1 and HUH. Correlation of the features with volume was evaluated with the Pearson correlation coefficient (PCC) computed on the Lung1 data.

3.4 Comparing features between the cohorts

To investigate if the results and conclusion of Aerts et al. regarding the four-feature radiomic signature are transferable to an independent local dataset, the features calculated on HUH were compared to those from Lung1. The distribution of calculated values for all features were compared between the two cohorts using the two-sided Kolmogorov-Smirnov (KS) test for the similarity between two unknown distributions. The null hypothesis of the two-sided KS-test is that the two provided datasets were sampled from the same distribution. Histograms were plotted for the volume distributions in the two cohorts using `VoxelVolume`, which was calculated alongside the other radiomic shape features (Table 3.3) and is the sum of all voxel volumes contained in the segmentation mask, multiplied with voxel volume. The `MeshVolume` was also extracted among the shape features, which calculates ROI volume by constructing intersection tetrahedrons based on the segmentation mask (Zwanenburg et al., 2020a), and gave practically identical values to those of `VoxelVolume`. Upon the discovery of a high-volume outlier in the HUH set, the KS-test was also performed with this individual removed from the dataset to examine if it significantly impacted the similarity of feature distributions.

For the four features in the radiomic signature, histograms were also plotted in the same manner as the `VoxelVolume`, with and without the large-volume outlier. The comparisons with HUH were also performed using the subset of Lung1 containing only patients diagnosed with locally-advanced disease (referred to as LA-Lung1), by removing all patients diagnosed with overall stage II and lower. This was done to examine the impact of the more general patient population in Lung1 compared to HUH, on the distribution of feature values.

[3] https://lifelines.readthedocs.io/en/latest/fitters/regression/CoxPHFitter.html

4

Results

4.1 Implementation

Figure 4.1 is an image of the window resulting from the `view_segmentions` function that is a part of the `Patient` class. By relying on the same raw data passed to the functions in *PyRadiomics* to retrieve image-segmentation pairs and displaying delineations to the user, it provides a simple method for both ensuring the robustness of the central functions handling the images and allowing the user to control the accuracy of the image material received from TCIA and HUH.

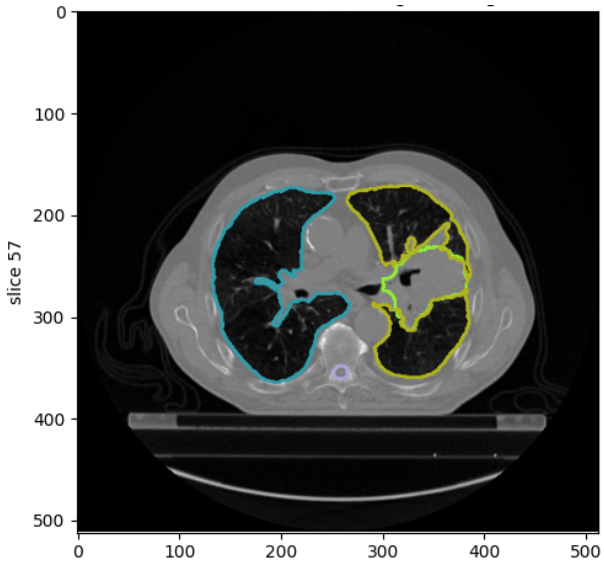


Figure 4.1: How `view_segmentations` shows delineations of the segmented structures, useful for QA of image data and of the implementation. The entirety of the imaged volume is navigated by scrolling. GTV in green, left lung in yellow, right lung in blue, and spinal cord in purple.

Figure 4.2 is a flowchart describing how `view_segmentations` uses the central functions in the Patient class for retrieving images. Inside the `view_segmentations` function itself, the binary masks provided are converted into coloured outlines which are then overlaid on top of the CT images before calling the `IndexTracker` on these images, giving the output window shown in Figure 4.1. The `IndexTracker` was retrieved from the matplotlib documentation[1] and adapted by the candidate for working with image matrices that have an additional colour dimension, and can be used to easily view all CT-slices comprising the volume. Separate functions were written for data from TCIA and HUH, due to differences in file structure.

[1] https://matplotlib.org/stable/gallery/event_handling/image_slices_viewer.html

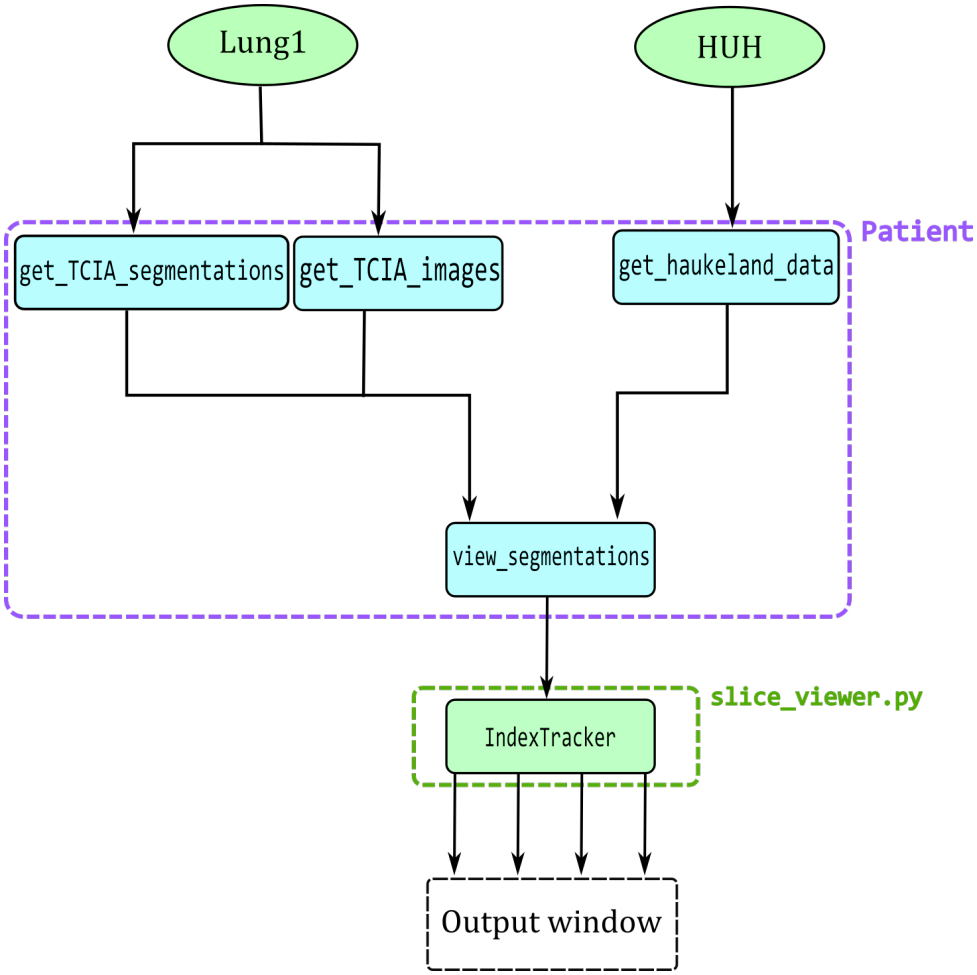


Figure 4.2: The interactions of functions in the `Patient` class together with image data and `slice_viewer.py` (Appendix A.4) for viewing segmentations on CT images.

During quality assurance of the Lung1 cohort's image material using this functionality together with the workflow described in Figure 3.1, issues with the images of six patients with the following patient IDs were discovered: LUNG1-014, LUNG1-021, LUNG1-085, LUNG1-095, LUNG1-128, LUNG1-194. The issue with these patients, except for LUNG1-128, was that the number of segmentation slices did not match with the number of CT image slices and thus making the two matrices incompatible. This issue seems to originate from the DICOM files themselves that are present on TCIA, with the version history[2] of the dataset indicating a prevalence of flawed image material. For LUNG1-128, the authors state in the same version history that this patient was a preoperative case that was included for completeness and hence does not have any associated delineation data. The details in incoherence between image- and segmentation matrices varied across the aforementioned patients and did not allow for a general solution to the issue, hence these 6 patients were excluded from the final Lung1 cohort used in this project. In addition to the aforementioned 6 patients, around 20 more had various smaller issues which needed to be accounted for by the candidate in the implementation. Most often the problem was the CT slice matrices presenting in a random order or being shifted in relation to the corresponding segmentation masks when sorted by filename, and had to be corrected using the slice-location parameter. The same QA process was applied to the HUH set and discov-

[2]https://wiki.cancerimagingarchive.net/display/Public/NSCLC-Radiomics

Table 4.1: Clinical statistics of the two cohorts, given in relative frequencies and number of subjects in parentheses.

(a) Lung1 (N = 416)

Sex	
Male	69.0% (287)
Female	31.0% (129)
T stage	
T1	22.1% (92)
T2	37.5% (156)
T3	12.3% (51)
T4	27.6% (115)
Tx	0.5% (2)
N stage	
N0	40.1% (167)
N1	5.3% (22)
N2	33.9% (141)
N3	20.0% (83)
Nx	0.7% (3)
TNM overall stage	
Stage I	22.1% (92)
Stage II	9.6% (40)
Stage IIIa	26.7% (111)
Stage IIIb-c	41.6% (173)

(b) HUH (N = 25)

Sex	
Male	64.0% (16)
Female	36.0% (9)
T stage	
T1	16.0% (4)
T2	20.0% (5)
T3	40.0% (10)
T4	24.0% (6)
Tx	0.0% (0)
N stage	
N0	8.0% (2)
N1	8.0% (2)
N2	72.0% (18)
N3	12.0% (3)
Nx	0.0% (0)
TNM overall stage	
Stage I	0.0% (0)
Stage II	0.0% (0)
Stage IIIa	44.0% (11)
Stage IIIb-c	56.0% (14)

ered no such problems, hence no patients from the initial cohort provided by the candidate's supervisor were excluded.

Tables 4.1a and 4.1b show the relative frequencies of clinical parameters in the two cohorts involved in this project, calculated using the built-in functionality of the `StudyGroup` class. They give the same statistics as Supplementary Table 4 from Aerts et al., 2014, and thus gives a basis for comparison not only between the two cohorts in this thesis, but also basis for comparison with the initial Lung1 cohort from TCIA, used by the original authors. No patients in either cohort were diagnosed with stage IV NSCLC.

Comparing this Lung1 cohort with the original, the difference in the individual parameters ranges from 0.0% to 0.5%. Note that the authors of the original 2014 study on Lung1 cite using the contemporary 7th edition TNM classification system (Compton et al., 2012), where stage IIIc had not yet been introduced. The HUH cohort however was sampled from late 2019 to mid 2021 and used the 8th edition TNM system, hence some patients in this cohort are staged with IIIc. As these would most likely be classified into IIIb following the 7th edition, stage IIIb and IIIc have been combined into the entry IIIb-c in Table 4.1.

Recall that the HUH cohort consists only of patients diagnosed with LA-NSCLC, which is

reflected somewhat in the distributions of T and N stages, with the later stages generally having a higher relative frequency compared to Lung1. However, both cohorts share some similar characteristics, such as a high N2 and N3 frequency within the N stage, and a comparable fraction of T4 cases. From Table 2.2, we can see that if the N stage is larger than 2 or if there is significant primary tumour involvement with or without some nodal spread, the disease is classified as locally advanced (stage III). This is reflected in HUH, as 72% of the patient are staged with N2, and 64% with either T3 or T4. Then the small fractions classified with N0 and N1 are most likely connected with cases of T $\geq$ 3 in order to be diagnosed as LA-NSCLC.

Figure 4.3 is a flowchart describing how the implementation calculates features in a provided `StudyGroup` instance using the functions in `feature_extraction.py` (Appendix A.2) together with the functionality of the `Patient` (Appendix A.1) instances. The wavelet filtering is in reality performed inside each specific `calculated_feature-class_features`, but is shown as it is in the figure for clarity regarding its role in feature calculation. Note the similarity in the upper halves between this schematic and Figure 4.2, where they both use the same output of the functions in the `Patient` class, supporting how `view_segmentations` was used for quality assurance prior to radiomic calculations.

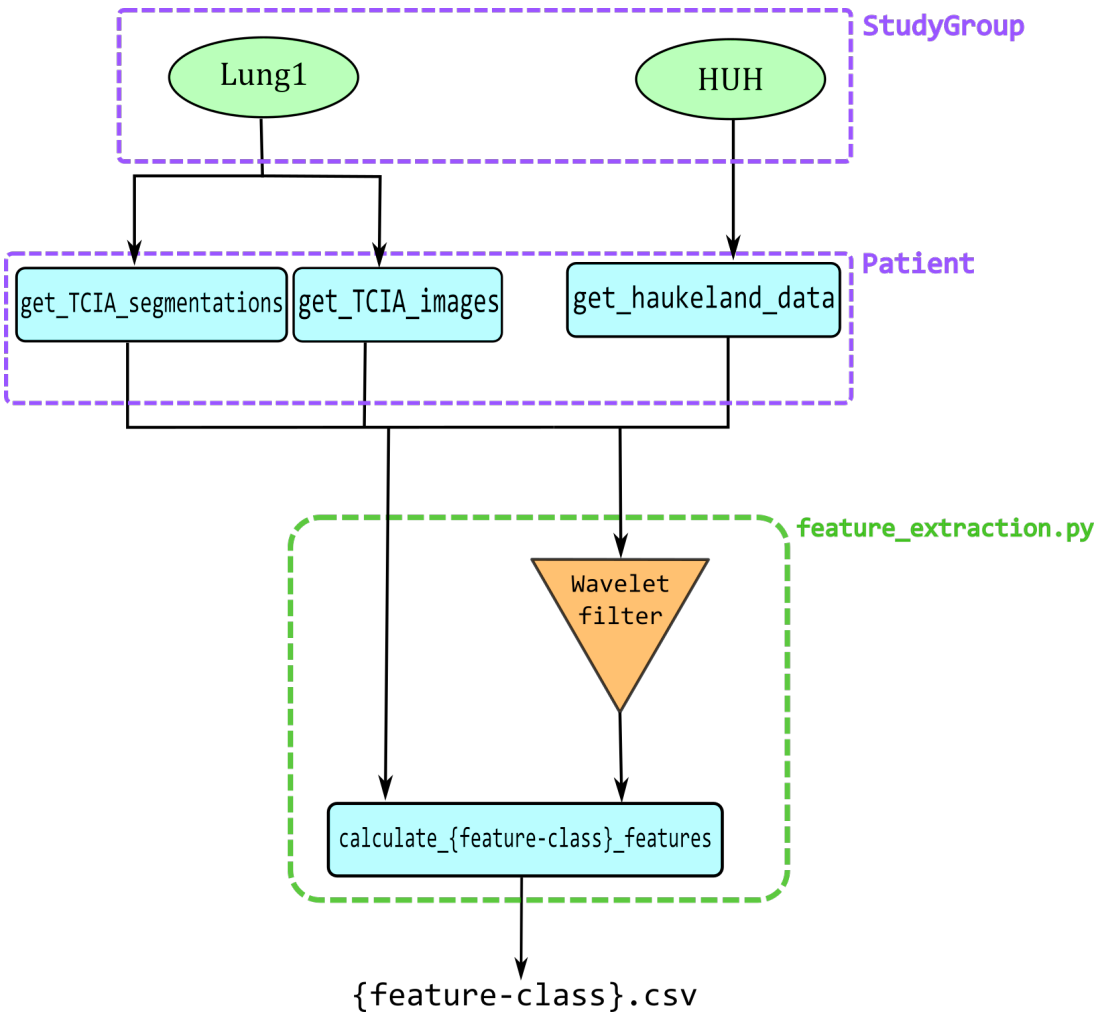


Figure 4.3: How the functions for feature calculation interacts with the `StudyGroup` and `Patient` classes for extracting radiomic features.

4.2 Implementation validation

The implementations ability to calculate radiomic features was validated by using the calculated features to reproduce the results presented in Aerts et al., 2014 developed from the Lung1 data set.

4.2.1 Kaplan-Meier analysis

Figures 4.4 to 4.7 shows the Kaplan-Meier curves from the original article (Supplementary Figure 1) plotted in red, together with the equivalent curves developed by the candidate in blue, for the four features comprising the radiomic signature proposed by the authors. The two survival groups were split according to individuals' feature value being above or below the cohort median. The p-values resulting from a Log-rank test between the implementation's curves (Blue) are provided below each plot, where the null hypothesis is that the two survival functions are

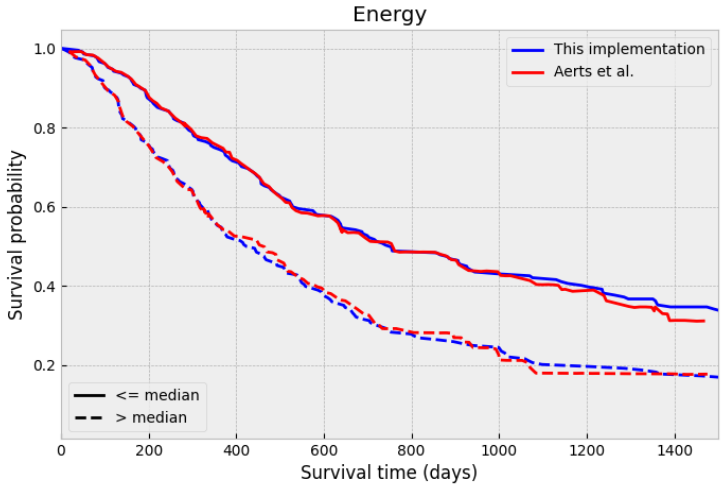


Figure 4.4: Energy, Log-rank test: $p = 0.00045$

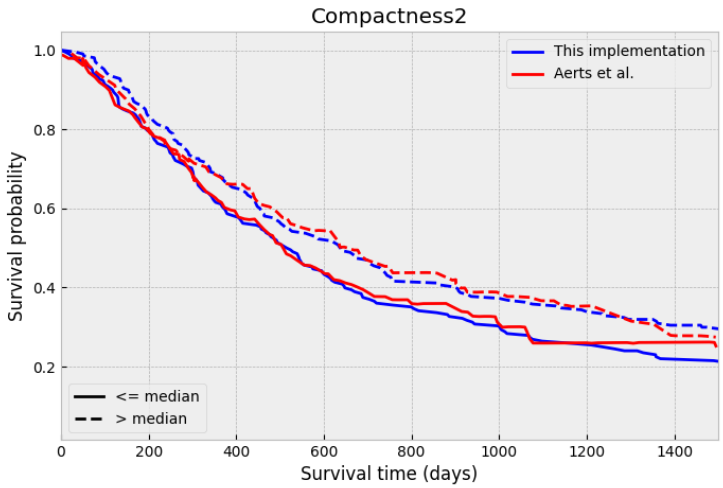


Figure 4.5: Compactness, Log-rank test: $p = 0.251$

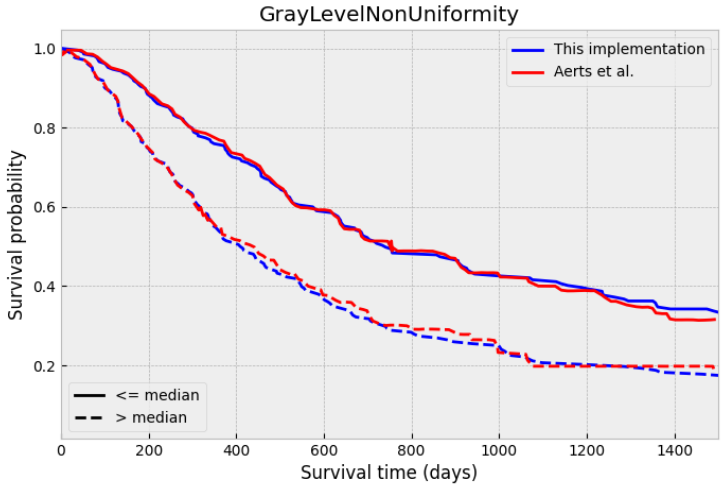


Figure 4.6: GLNU, Log-rank test: $p = 0.00044$

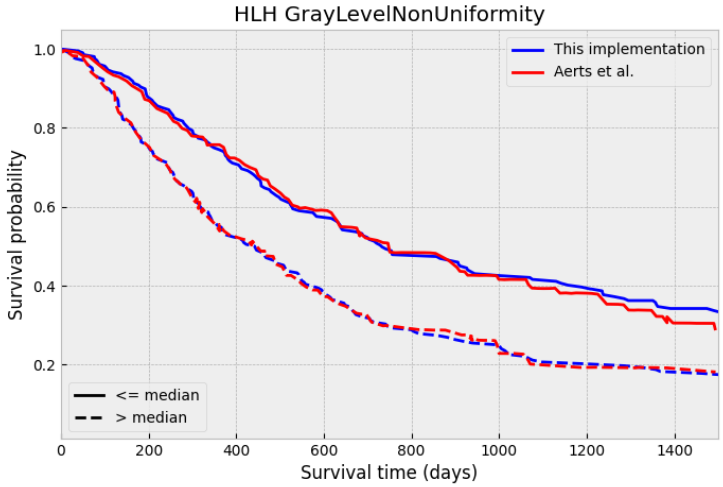


Figure 4.7: HLH GLNU, Log-rank test: $p = 0.00135$

the same.

Note that the curves labelled as "Aerts et al." are reconstructed from images of these curves in the original article, and thus contain minor imperfections such as small local spikes seen in some sections. They are however still an accurate representation of the shape and location of the curves presented in the original article.

These plots show clear agreement between the KM-curves in Aerts et al. and the independently developed curves of this thesis. The Log-rank test shows significant difference in survival between high- and low feature-value individuals in Lung1 for Energy, GLNU and HLH GLNU, but no significant difference in survival probability for Compactness.

4.2.2 Cox proportional hazards model

Equivalent to what was done in the original study, a Cox proportional hazards model (Appendix A.3) was fit onto the Lung1 cohort using the four-feature signature as binary exposure variables. When fitting the model, the covariates were made binary in the same fashion as in the Kaplan-Meier analysis, by splitting around the median. The resulting regression coefficients and related statistics are provided in Table 4.2, while Figure 4.8 is a plot of the coefficients together with 95% confidence intervals. The hazard coefficients, `exp(coef)`, gives the baseline hazard change for one unit change in the related covariate.

Table 4.2: The coefficients resulting from fitting a Cox proportional hazards model on Lung1

covariate	coef	exp(coef)	coef lower 95%	coef upper 95%	p
Energy	0.368	1.445	-0.306	1.042	0.285
Compactness2	0.028	1.028	-0.202	0.257	0.813
GLNU	0.316	1.372	-0.308	0.94	0.321
HLH GLNU	-0.297	0.743	-1.048	0.454	0.439

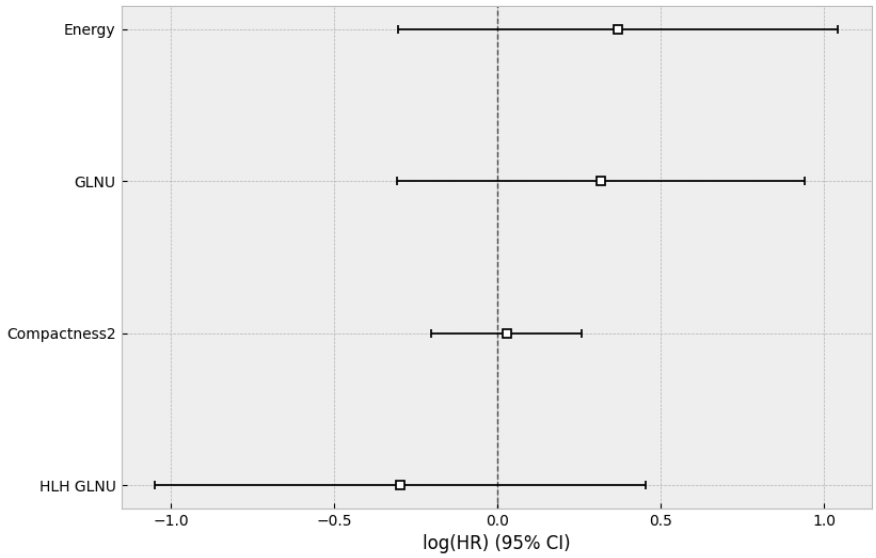


Figure 4.8: Plot of the model coefficients with 95% confidence intervals.

According to the hazard coefficients in the fitted Cox-PH model, all features except for compactness induce a considerable change in hazard between patients with high- and low- feature values. $p > 0.05$ for all covariates however, meaning that we cannot say with certainty that they are different from 0. Moreover, the confidence interval of the Compactness2 coefficient is centred very narrowly around 0 compared to the other three features, implying it has much less impact on the hazard compared to the other features. This coincides with the finding from the Kaplan-Meier analysis where Compactness2 shows very low difference in survival times between patients with high and low compactness values.

The total model performance on the fitted data is indicated by the log-likelihood ratio test and the concordance index (CI) which was calculated using the built-in functionality of the model fitter, the result of which are shown in Table 4.2. The test compares the existing model with

Table 4.3: Performance of the overall Cox-PH model.

test statistic	p	-log2(p)	concordance
13.416	0.009	6.731	0.572

all covariates against the trivial model with no covariates, i.e. the baseline hazard, and shows that the total model with all covariates combined provides a significant difference in survival outcome.

A single signature variable was created via a linear combination of the four features using the calculated Cox-PH weights, and KM-curves were again fitted using the median as a discriminator and plotted together with those extracted from Aerts et al., 2014 (Figure 4.9). The agreement of the total signature is equivalent to the behaviour exhibited by each individual feature in Section 4.2.1.

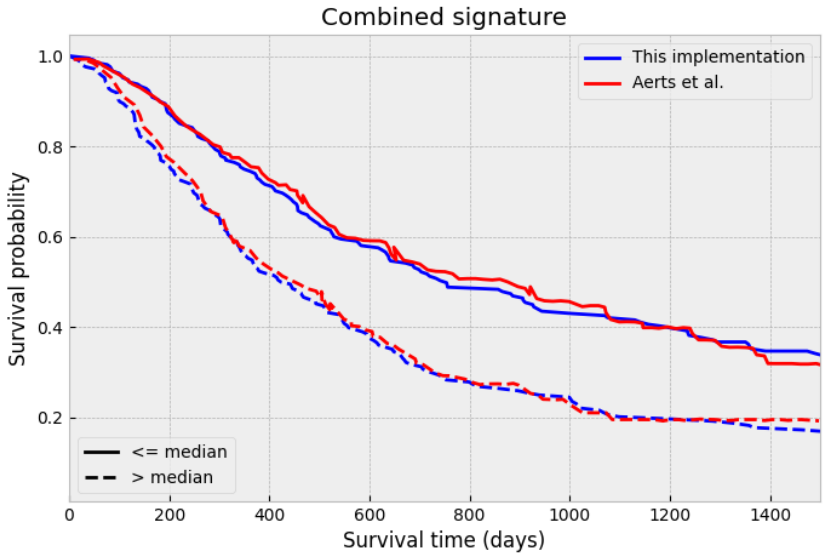
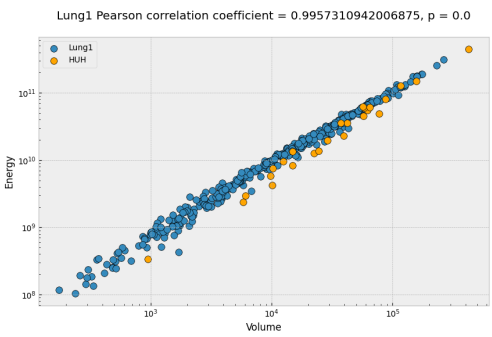


Figure 4.9: Kaplan-Meier curve of Lung1 for the combined radiomic signature.

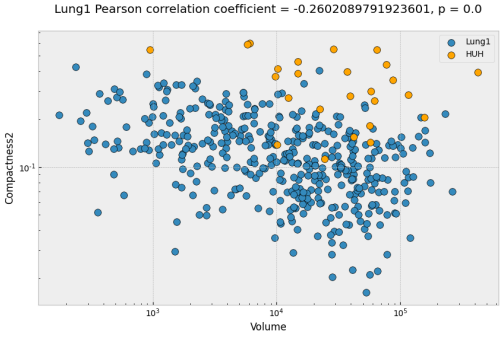
4.3 Correlation with volume

In Figure 4.10 are plots of feature value against the corresponding volume for all patients in Lung1 (blue) and HUH (orange), for each individual feature and the combined radiomic signature from Section 4.2.2. The Pearson correlation coefficient (PCC) is calculated on Lung1 and provided above each plot together with the p-value. Both axes in all plots are logarithmic to give a more uniform spread of points. Volume is the calculated `VoxelVolume`.

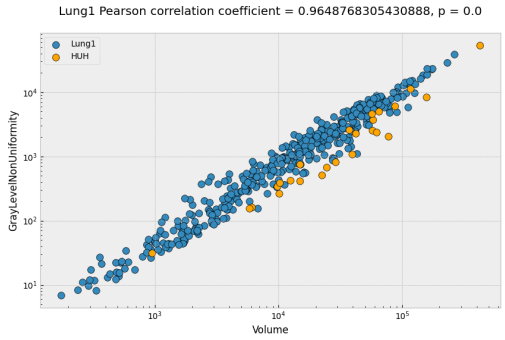
The calculated correlation coefficients show that Energy, GLNU and HLH GLNU all have a strong linear correlation with primary tumour volume (PCC ≥ 0.95). Note the similarity between the plots of Energy (Figure 4.10a) and the total prognostic signature (Figure 4.10e), additionally having very similar correlation coefficients that only differ from the ninth decimal



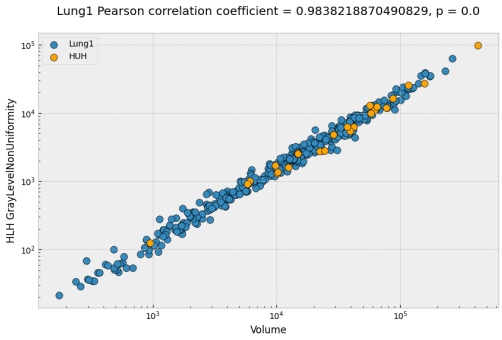
(a) Energy



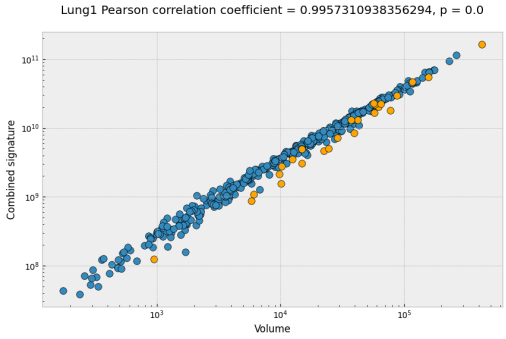
(b) Compactness2



(c) GrayLevelNonUniformity



(d) HLH GrayLevelNonUniformity



(e) The combined signature

Figure 4.10: Log-log plots of feature-value against volume for each feature in the prognostic signature.

place. Figure 4.11 is a plot of energy against the combined radiomic signature for both cohorts, as for the feature-volume plots above. It clearly shows a practically perfect correlation between the two features (PCC calculated from Lung1 values).

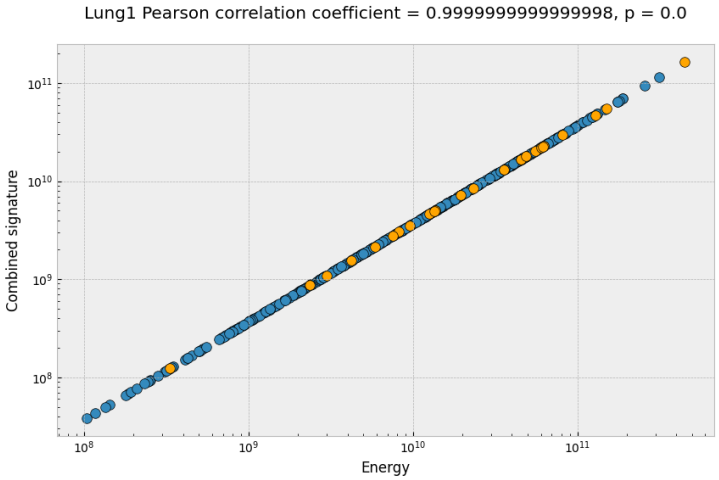
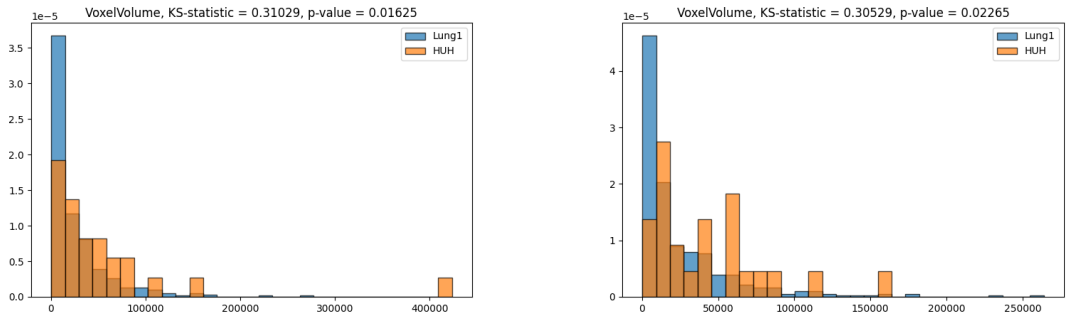


Figure 4.11: Correlation between energy and the radiomic signature

4.4 Comparing feature distributions

For examining the stability of feature calculation across independent datasets, the distribution of the calculated signature features were examined for the two cohorts. The comparison to HUH was done both with all patients in the Lung1 cohort, and also with just the patients in Lung1 that were diagnosed as locally advanced (LA-Lung1).

4.4.1 All stages



(a) With large-volume outlier in HUH.

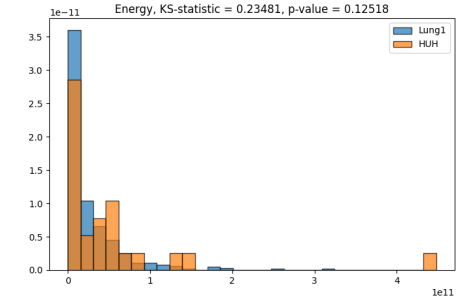
(b) Without outlier.

Figure 4.12: Distribution of `VoxelVolume` values in Lung1 and HUH.

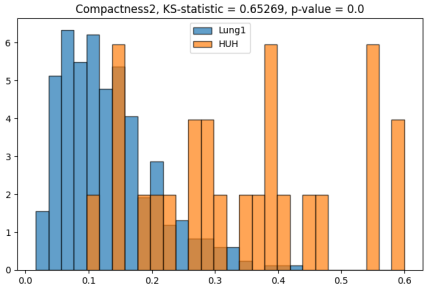
Figure 4.12a shows histograms of the primary tumour volume in both cohorts. Note that the histograms are normalised in order to show them comparatively on the same figure. The plots are created from the metric `VoxelVolume`, which was calculated alongside other radiomic shape features (Table 3.3). Figure 4.12b shows the same distribution but with the high-volume outlier in HUH removed. Above the histograms are also the results of a two-sided Kolmogorov-Smirnov (KS) test. From the test results, $p = 0.016$ and $p = 0.023$ with and without the outlier respectively, indicating that the volume distributions of the two cohorts are significantly different in both cases.

For the features in the radiomic signature, Figure 4.13 reveal that the underlying distributions are significantly similar between the cohorts for Energy (Figure 4.13a) and Grey-level non-uniformity (GLNU) (Figure 4.13c), having $p = 0.125$ and $p = 0.196$ respectively. For the wavelet decomposed GrayLevelNonUniformity (HLH GLNU), HUH seems to be slightly more weighted towards larger values of this feature compared to Lung1, and $p = 0.02 < 0.05$ implies that the underlying distributions are significantly different. Regarding Compactness2 (Figure 4.13b), the two distributions are clearly dissimilar, where Lung1's histogram is a skewed bell curve, while for HUH the feature values seem to be somewhat uniformly distributed and with generally higher values of compactness. This disagreement is also reflected in other calculated features that are either fully or somewhat correlated with compactness, as illustrated in Figure C.3, but also present in shape features which are not correlated with compactness (Figure C.4). Moreover, the distribution of compactness values is dissimilar from the other three signature features within the Lung1 set itself, which seem to be exponentially decreasing from a maximum closest to 0.

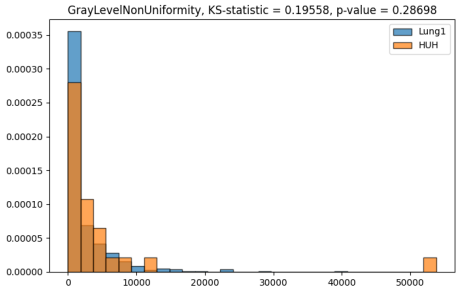
The large-volume outlier is also present in Figures 4.13a, 4.13c and 4.13d for Energy, GLNU and HLH GLNU respectively. This is because these features are calculated using a sum of all



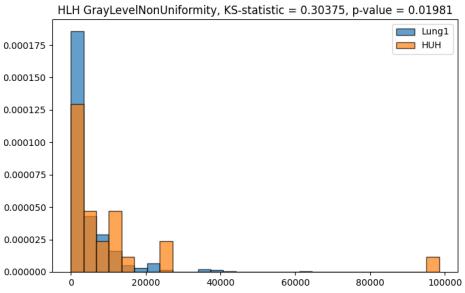
(a) Energy



(b) Compactness2

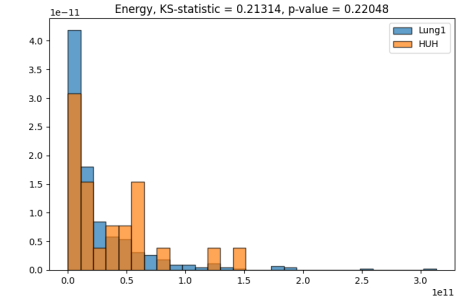


(c) GLNU

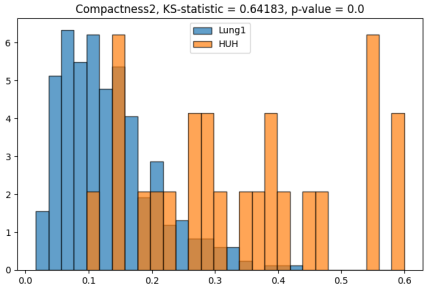


(d) HLH GLNU

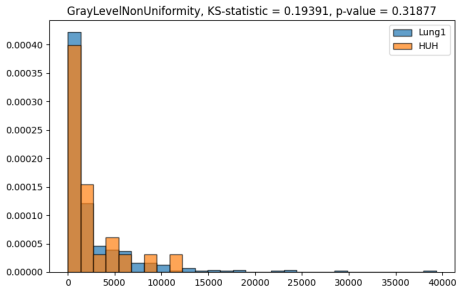
Figure 4.13: The distribution of feature values for the four features comprising the radiomic signature, across the Lung1 and HUH cohorts. KS-test results above each plot.



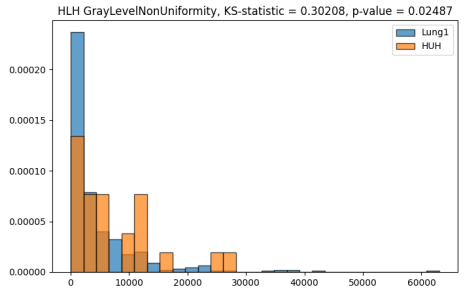
(a) Energy



(b) Compactness2



(c) GLNU



(d) HLH GLNU

Figure 4.14: The same histograms as in Figure 4.13, but with the large-volume outlier removed.

voxel values inside the segmented volume and thus a larger volume implies a large feature value. There is no outlier for HUH Compactness2 however, since compactness is calculated as the ratio of tumour volume to surface area (Equation (2.17)). As was done for the volume, the histograms of the four features were calculated again with the outlier removed to examine its impact. These are shown in Figure 4.14. The two previous features still had similar distributions with the removal of this individual but did not induce a change large enough in HLH GLNU to bring the p-value above 0.05.

The two-sided Kolmogorov-Smirnov test was also performed on all the calculated features to determine how many features had significantly similar distributions when comparing the patient cohorts. Out of the 66 features calculated (GLCM not included), 28 had p-values higher than 0.05. The ten features with highest p-values values across all feature groups are plotted in the bar chart in Figure 4.15, with p-values next to each bar. Appendix C.4 contains the results of KS-tests on all features within each feature class, for the complete Lung1 set compared to HUH.

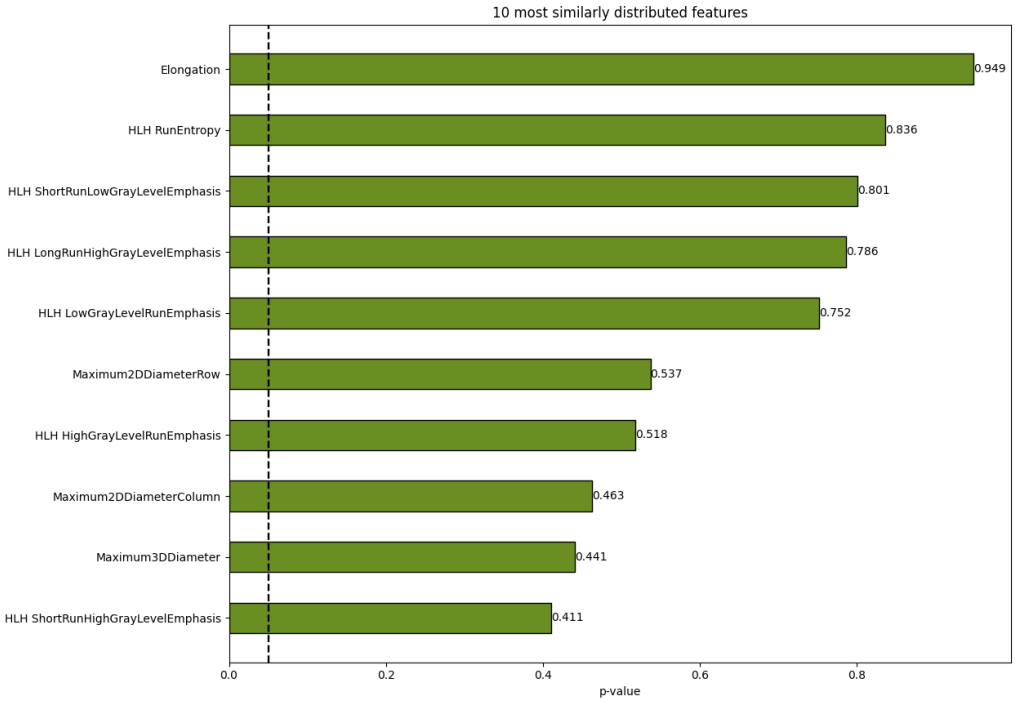
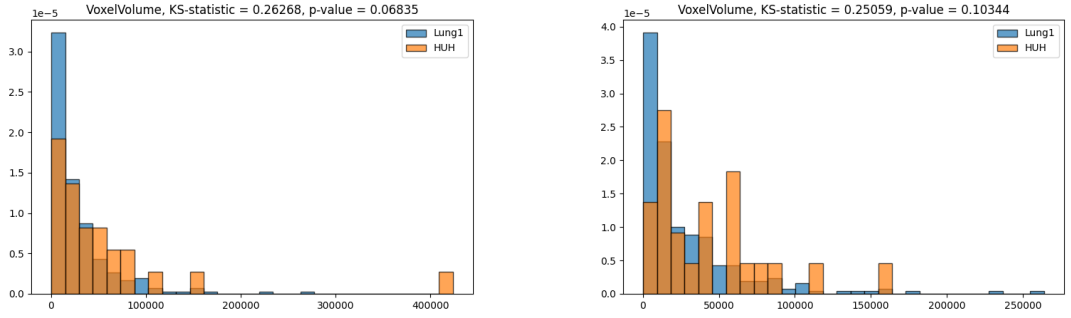


Figure 4.15: The 10 features with the largest p-value from the KS-test on Lung1 and HUH, out of the calculated features. Calculated without outlier patient. Vertical line is $p = 0.05$

4.4.2 Locally advanced stages

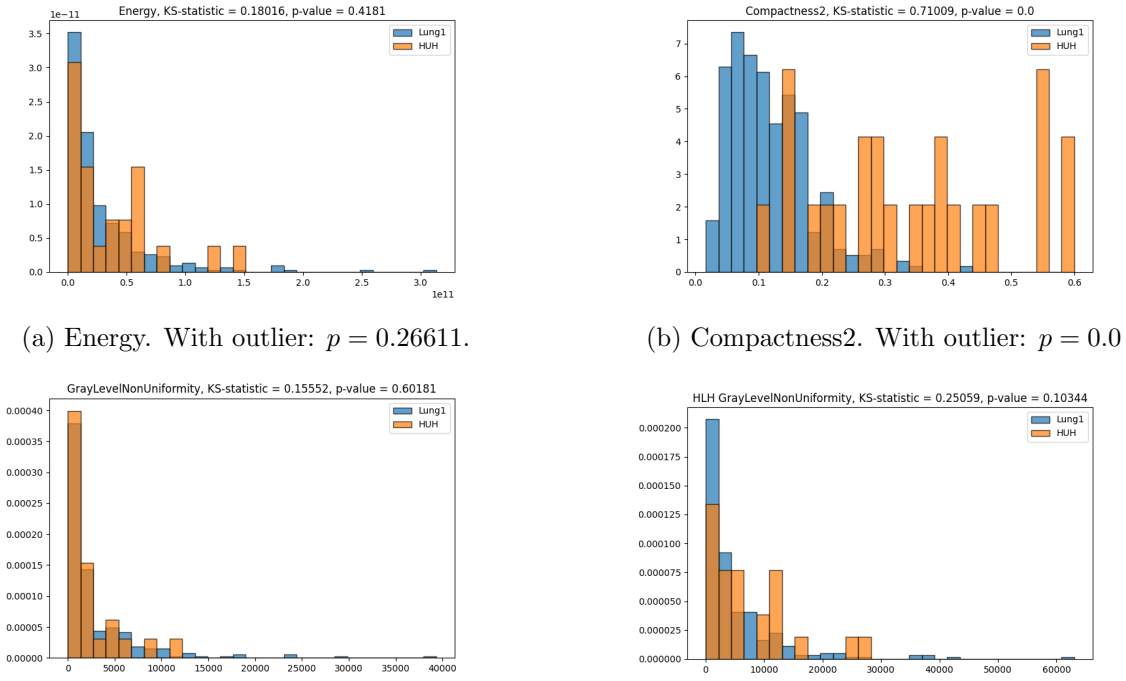
The impact of staging differences between the two patient populations was examined by repeating the comparisons in Section 4.4.1, but including only the locally-advanced patients in Lung1 (LA-Lung1). In this section are comparisons for volume (Figure 4.16) and feature distributions (Figure 4.17) between HUH and LA-Lung1. The histograms for the radiomic signature features



(a) With outlier.

(b) Without outlier.

Figure 4.16: Distribution of `VoxelVolume` values for LA-NSCLC patients in Lung1 and HUH.



(a) Energy. With outlier: $p = 0.26611$.

(b) Compactness2. With outlier: $p = 0.0$

(c) GLNU. With outlier: $p = 0.56407$

(d) HLH GLNU. With outlier: $p = 0.08862$

Figure 4.17: The same histograms as in Figure 4.13, but with only LA-NSCLC patients from Lung1.

in Figure 4.17 were all calculated with the large-volume outlier removed, while p-values for the KS-test with this patient included are given below each plot.

With LA-Lung1 only containing locally advanced patients, the test evaluates the two volume distributions to be equal both with and without the large volume patient. The test moreover indicates that the HLH GLNU distributions are equal both with and without the outlier, which was not the case for the complete Lung1 set. The ten features with largest p-values from the KS-test in comparing LA-Lung1 and HUH are plotted in Figure 4.18. 33 out of 66 features had p-values above 0.05.

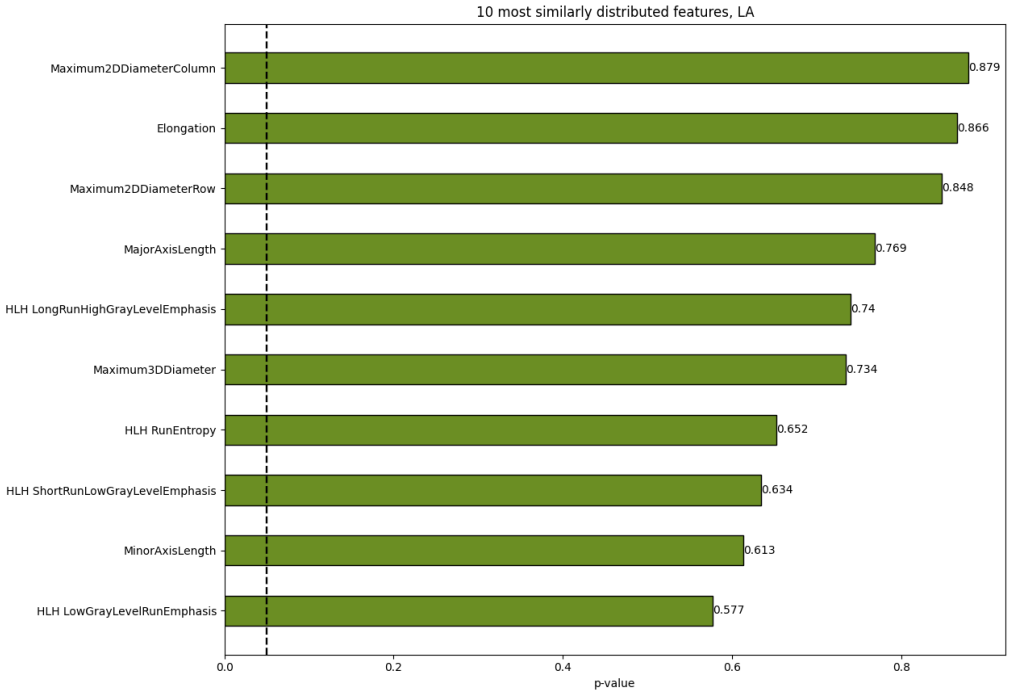


Figure 4.18: Same plot as in Figure 4.15, but comparing LA-Lung1 with HUH.

4.4.3 Volume outlier

Note the high-volume outlier present for HUH in the left figure, which originates from a patient with a particularly large primary tumour volume (shown in Figure 4.19) compared to the others in the cohort. This patient had a primary tumour volume of 809 cm^3, while the volume mean and median for HUH is 110 cm^3 and 71.2 cm^3, respectively (calculated from `VoxelVolume`). KS-tests were performed on the data sets with and without this patient removed from HUH to determine if it had a significant impact on the test results. Removal of this patient did not impact test outcome on any of the features in radiomic signature, nor on the `VoxelVolume`.

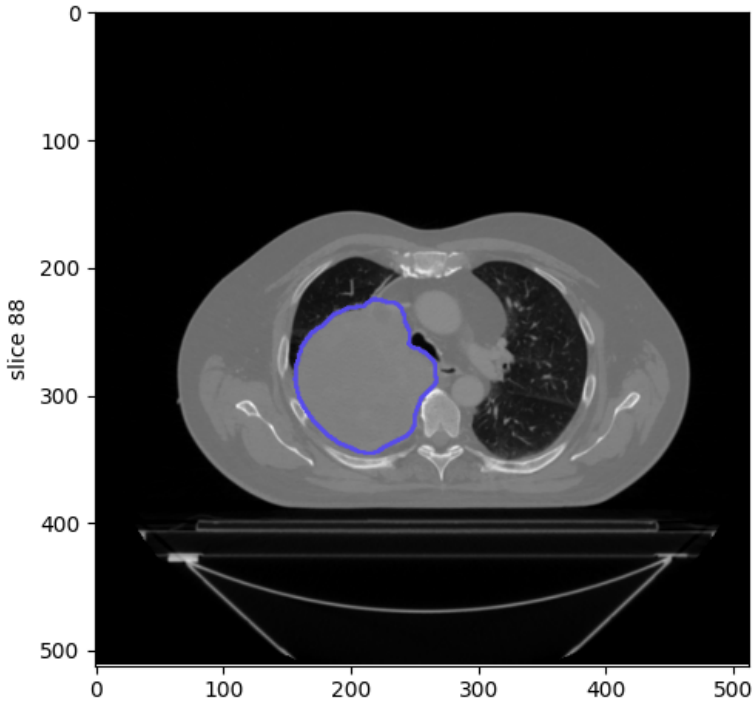


Figure 4.19: CT slice of the patient with a large volume primary tumour, outlined in blue.

5

Discussion

5.1 The prognostic signature

In their 2014 study, Aerts et al. provided extensive descriptions of their large assortment of results, but disclose few specific details regarding the trained model, in-house source code, and underlying numerical calculations. *PyRadiomics* was released some years later as an open-source implementation of the methods from the authors' original in-house Matlab code (Welch et al., 2019; van Griethuysen et al., 2017). With radiomics involving numerous variable factors from a clinical to scientific level, external validation of results and investigations into the applied methods are essential for identifying potential inconsistencies or errors. The prognostic signature by Aerts et al. was internally validated both in an independent lung cancer- and two head and neck (H&N) cancer cohorts, showing translational potential of the signature across different cancer pathologies. In a 2015 paper, Leijenaar et al. externally validated the signature on an independent H&N cancer cohort from the Princess Margaret Cancer Center and found comparable prognostic results to those of Aerts et al.

5.1.1 Reproducing radiomic signature results

For comparing Kaplan-Meier curves from this work to those presented in Aerts et al., 2014, the online tool *WebPlotDigitizer* was used for reconstruction of these curves from images in the publication. While some concerns may be raised at the accuracy of reconstructing curves using only simple images, the tool did in fact show a high degree of accuracy in aligning coordinate points with appropriate sections of curves in the images (See Figure C.1). There are however some small errors in the coordinates, which can most clearly be seen on Figure 4.9, with small upwards spikes at certain points, while a true survival curve is always strictly decreasing. However, the magnitude of these errors is quite small and the overall shape of the curve is still represented accurately. Due to being extracted from images, the curves labelled as "Aerts et al." in the Kaplan-Meier plots in Section 4.2 of this thesis do not represent true survival curves such as those from the implementation, but still provide an accurate retelling of the original curve shape and position. Hence they can be used for comparisons and validation of the survival curves developed by the candidate on the same data.

Independently calculated Kaplan-Meier curves were constructed by the candidate with the same methodology outlined in Aerts et al. and plotted together with those from the article (Section 4.2.1). The strong agreement of these curves provide compelling evidence that the implementation is able to compute radiomic features on the Lung1 dataset equivalent to those calculated by Aerts et al. Hence the results developed in this work can be used as supplementary or substitute data for otherwise missing or ambiguous details in the original paper. As in the study, a Cox proportional hazards model was fitted on the signature using the features Energy, Compactness2, GLNU, and HLH GLNU as exposure variables. It was this model that they fitted on the Lung1 dataset and validated on the independent Lung2, H&N1 and H&2 datasets. These validations could not be reproduced as the validation sets were not included

in this project, nor could HUH be used as a validation set due to the lack of follow-up data on patients. So to examine if the Cox-PH model fitted on Lung1 by the candidate was equivalent to that of Aerts et al., the Kaplan-Meier curves for the total radiomic signature presented in the article were reproduced using the same methodology as the authors (Figure 4.9), involving the Cox-PH regression coefficients. These curves for the overall signature agree very similarly to those presented in Section 4.2.1 and hence strongly indicates that the Cox-PH model (Table 4.2) developed is equivalent to the one in the original study. This also shows that, although not explicitly stated by the authors, that the KM-curves presented in the article for illustrating the survival performance of the combined radiomic signature were constructed by a linear combination of the four feature values using the Cox-PH coefficients as weights. Any discrepancies in the previous KM-curves can most likely be attributed to small errors in *WebPlotDigitizer* and from the small difference of six excluded patients between the Lung1 cohort used in this work and the one in the original study.

5.1.2 Signature performance and validity

Aerts et al. performed a large-scale study with extensive internal validation on, most centrally, the prognostic significance of their radiomic signature consisting of: Energy; Compactness2; Gray-level non-uniformity; and HLH Gray-level non-uniformity. From their KM plots equivalent to those presented in Section 4.2.1, they gather that "Features describing [primary tumour heterogeneity] were associated with worse survival in all four datasets" and that "Patients with more spherical/compact primary tumours had better survival probability". The KM analysis performed in this work is consistent with the results presented in the article, where a significant difference in survival between high and low values was shown for the features Energy, GLNU and HLH GLNU. The Cox-PH model fitted using the four features also gives some indication that these are related to overall survival in the related hazard ratios (Table 4.2), but due to all p-values being larger than 0.05 this cannot be said for certain. Note also that the Cox-PH model Table 4.2 assigns a negative value to the regression coefficient of HLH GLNU meaning that an increase in feature value leads to a decrease in hazard (higher survival probability). This is opposite to what is implied by the HLH GLNU KM-plots in this work and in Aerts et al., 2014 where a higher value leads to lower survival probability. The reason for this disagreement between the KM analysis and Cox-PH model is unknown, but can be indicative of a flaw in fitting of the Cox-PH model.

In this work we also successfully recreated the KM-curves for the combined radiomic signature (Figure 4.9) presented in Aerts et al., 2014, using the coefficients from the Cox-PH model fitted on Lung1 to combine the four features into a single variable. In Aerts et al., 2014 these are used to demonstrate the performance of the total radiomic signature on the data sets. There is a flaw in using this method for illustrating the survival trends relating to this combined signature feature however, being the difference in magnitude between the four features that are combined. The energy values[1] are most often in the range 10^9 to 10^{12} while (HLH) GLRLM values are usually around 10^3 and compactness is between 0 and 1. Hence the combined feature value is almost solely determined by the Energy value. This is indicated by Figure 4.11 which shows a perfect correlation between Energy- and signature values constructed with this method, with Pearson correlation coefficient (PCC) ≈ 0.99. Figure C.10 shows the implementation's KM-curves of Energy plotted together with the signature curves from Aerts et al., 2014. The close fit of these gives strong indication that the authors employed this linear combination with no prior normalisation for displaying the performance of the radiomic signature with KM-curves. Hence these curves for the combined signature are in fact displaying the survival differences in

[1] https://github.com/filipjb/NSCLC_radiomics/tree/master/feature_files

relation to the Energy feature with negligible influence from the other three features. Note that this does not have any impact on the other results from the Cox-PH model, which used the model itself for predictions, but illustrates that using this linear combination for illustrating the combined signature performance is invalid.

The definitions of both Energy (eq. (2.15)) and GLNU Equation (2.20) imply a certain dependency of these feature to the tumour volume they are calculated on. Energy is the sum of squared intensities in the ROI, both dependent on the intensity value and total number of voxels. The numerator in the GLNU definition is dependent on the number of run lengths squared in the Gray-level run-length matrix (GLRLM), which is in turn dependent on the dimensions of the image (or ROI) that the GLRLM is calculated from. Due to how the features in the signature are defined, the correlations with volume of each individual feature and the combined signature were investigated (Section 4.3). Figure 4.10 shows that Energy, GLNU and HLH GLNU are all strongly correlated with volume in both Lung1 and HUH, confirming the deductions made from the definitions of these features. This explains the trend seen in Figures 4.4 to 4.7, where it is these three features that are significantly prognostic. Tumour volume already being an established prognostic factor in the clinic, which makes assessing the supplementary clinical value of this signature difficult as it is primarily capturing the delineated volume. The one feature that was not directly correlated with tumour volume, Compactness2, showed no significant difference in survival (Figure 4.5) with $p = 0.251$ for testing the equality of the survival functions.

These issues of the radiomic signature developed by Aerts et al. were also outlined in Welch et al., 2019, showing exposed vulnerabilities of the applied radiomic method and highlighting safeguards for ensuring accuracy of radiomic signatures. In the study, they refitted a model on Lung1 and validated on an independent H&N cancer data set using the same prognostic signature, and found that tumour volume alone was equally prognostic as the radiomic signature. They also determined co-linearities in the same three signature features (Energy, GLNU and HLH GLNU) stemming from their strong dependency on tumour volume. They also fitted the model after randomly permuting voxel positions in the CT image, removing any intensity- and texture-based context from the ROIs while still preserving volume and shape information. With the voxel-randomised model they achieved equivalent prognostic performance both from the individual features and combined radiomic model, indicating that the three features calculated from intensity (Energy, GLNU and HLH GLNU) did not capture prognostically relevant information from intensity and texture, but rather describing the delineated volume. Welch et al. also showed in their extensive study that when volume-normalizing the confounded features they lose the specific predictive capabilities visualised in Figures 4.4 to 4.7, on their validation data. The fact that the radiomic model in Welch et al., 2019 was not able to outperform a model with tumour volume alone, implies that this particular radiomic signature is not able to provide additional prognostic information beyond known clinical metrics.

This does not necessarily imply that radiomic signatures could not be significantly prognostic under different constructions and applications, but the involvement of volume needs to be determined. Interestingly, Fave et al., 2017 investigated two radiomic models for the prediction of both overall survival and time to distant metastasis and found that adding Compactness2 to both of these increased their respective predictive performance as measured by the concordance index. Additionally, Yuan et al., 2017 created a radiomic signature of 20 features which outperformed a volumetric model significantly, although potential volume dependencies or co-linearities of this signature were not investigated.

These discoveries highlight, as Welch et al. notes in their article, the need for safeguards throughout the radiomics workflow for ensuring standardization with careful selection and testing of features, for ensuring accurate interpretation of model behaviour and performance. It is

important that, although this methodology is often applied to high-dimensional data with many abstracted metrics, that we exercise caution in viewing the solutions and models as a "black box". In addition to rigorous training and extensive validation, we must carefully consider underlying feature definitions, co-linearities and dependencies to known clinical factors, where applicable. The inclusion of GLRLM texture features into the radiomic library should also be done with caution and detailed focus on proper normalisation with regards to ROI volume and other varying factors. These features were originally designed for distinguishing terrains depicted on aerial photographs in the 70's (Galloway, 1975) on a much more homogeneous image set which allowed for accurate description of texture characteristics. The large differences in ROI dimensions in radiomics appears to induce a high degree of volume dependence in some of these features, while in the original study the uniform computational basis allowed for an accurate comparison of texture characteristics.

5.1.3 Feature selection

Three out of four features selected for the radiomic signature in Aerts et al., 2014 were strongly correlated with volume which makes evaluating the supportive potential of such a model in the clinic difficult. Investigating the feature selection process can provide insight into how volume became a dominant factor in the radiomic signature, and what measures can be made for ensuring the selection of stable, predictive and clinically relevant features.

In their study, Aerts et al. used an approach favouring experimentally stable features, rather than the more traditional method of feature selection via predictive power and low redundancy. In a field where new quantitative image features are conjured up at a high rate, it is difficult to evaluate both predictive significance and noise-vulnerability of the features on a general basis. Since features susceptible to random or potential systematic errors are most likely poor predictors, it makes sense to perform feature selection starting at the fundamental image acquisition level. The authors ranked their calculated features from least to most stable based on test-retest and inter-observer variations, and found evidence that stable features are better predictors, despite not using any prognostic information in the ranking. Their results showing the relationship between stability rank and prognostic performance are given in Figure C.9. Hence stability based feature selection seems viable for both ensuring prognostic significance and reproducible methodology. This hypothesis is also explored by the others, where many later studies use test-retest and/or inter-observer delineation stability as part of its radiomic feature selection (Papanikolaou et al., 2020; Hosny et al., 2018). An expansion of this concept with future potentially longitudinal applications, is presented in Timmeren et al., 2017, where they examined feature selection using detectable changes in features during the treatment course. Stability must also be reproducible itself, and the effects of clinical- and hardware specific variations must be described in order to build a standardized methodology. Traverso et al., 2018 performed a large meta-study on 41 papers reporting repeatability and reproducibility of radiomic features. In CT imaging, they gather that first-order and shape features were in general more repeatable than texture features. Slice thickness resampling and differing reconstruction algorithms strongly degraded feature reproducibility.

Stability is however not the final or only solution to feature selection. Due to the nature of stability ranking, it is very likely that many if not most of the top-ranking features, at least within the same class, are highly correlated. This is especially an increasing concern with higher dimensionality data. Aerts et al. focused on the 100 most stable features, and from these selected the single best prognostically performing feature from each of the four feature groups to create the prognostic signature. Their stated objective with this approach was to remove redundancy within the radiomic data. The issue with this approach is that although correlations between

features in the same group are eliminated, the possible dependencies of features between groups are not accounted for. It also seems that the focus on the 100 most stable features have eliminated most of the prognostic shape features, as Compactness2 shows negligible differences for survival both in the KM-analysis (Figure 4.5) and Cox-PH model (Table 4.2). The poor performance of Compactness2 and its potential origin is not elaborated upon in the article. Although this method led to a poor-performing shape feature, stability should still be a high priority in the early stage of dimensionality reduction. If prognostic, but unstable features are chosen, they can potentially be even less useful outside the training set due to overfitting. Primary tumour volume is already known to be a significant prognostic factor for lung carcinomas, and should be stable with respect to inter-observer delineations as most tumours stand in contrast to the surrounding dark lung tissue on CT images (e.g. Figure 4.1). Hence it is hard to believe that no volume-dependent features were among the top 100 features considered by Aerts et al., 2014 for the signature. It is possible the authors wanted to avoid any clear connections with volume in order to show the potential of radiomics on its own, and also be able to compare to a volume-only model in their validation and hence chose to not focus on these.

Moreover, the dimension of the final prognostic signature of Aerts et al. seems low compared to many other studies performed in recent years, where signatures with over 10 (Yuan et al., 2017) and even up to over 40 features (Hosny et al., 2018) have been used, depending on the models used for prediction. The reasoning for this might be due to the early position of the study, with little previous research performed in the field, the authors might have wanted to show the general potential of radiomics and not fall into the pitfalls of overfitting. There is no reason to rule out that there is additional prognostic information in multiple uncorrelated features within the same feature class, and should also be considered for inclusion in a radiomics model. The number of features should still be selected carefully however, keeping overfitting in mind. Wu et al., 2016 investigated 24 feature selection methods together with 3 classification models for histology prediction, where signatures 5-15 features in size together with Relief-based selection algorithm or Naive Bayes classifier performed very well, but the study shows that performance varies immensely depending on signature dimension, selection method, and classifier.

A sensible multi-stage methodology for radiomic feature selection seems to be a natural progression of the foundation laid by Aerts et al. First, the most stable features with regards to varying clinical parameters such as delineations and image acquisition should be chosen, increasing reproducibility. There is good evidence that a large number of features exhibit desirable stability, such that the remaining dimensionality is large enough to perform careful feature selection. For example, van Griethuysen et al., 2017 found as many as 535 features with intra-class correlation coefficient (ICC) ≥ 0.8 between four independently delineated datasets. Then redundancy can be eliminated via conventional correlation analyses, and from these the most predictive features can be chosen using established regression- and/or machine learning models. The most common machine learning method applied in radiomics is LASSO regression (Song et al., 2020), which has shown good performance for selecting predictive features (Dai et al., 2021). Yuan et al., 2017 used such a multi-stage approach which used the 100 most stable features with ICC ≥ 0.9 regarding inter-observer variation before redundancy- and prediction-based selection was done using nearest neighbour clustering and Relief feature selection (Urbanowicz et al., 2018), respectively.

5.2 Application on an independent dataset

The locally advanced HUH cohort is a population of higher risk NSCLC patients compared to Lung1 which consists of a more general patient population with all TNM-stages excluding stage IV. The translational potential of the radiomic signature onto an independent dataset, and how the difference in staging affects the calculated values were examined in Section 4.4. The size of HUH (N = 25) is much smaller than Lung1 (N = 416) and should be kept in mind when comparing the distribution of features in the two cohorts. HUH is also considerably smaller than the internal validation sets used in Aerts et al., 2014 which ranged from N = 95 to N = 225.

5.2.1 The impact of delineation procedure

A central difference between Lung1 and HUH with regard to radiomic feature calculation, is the delineation protocol and objective. HUH segmentations stem from IGTV delineations drawn on 4D free-breathing CT-images, while Lung1 GTVs were delineated on the basis of 3D PET-CT. As the IGTV consists of the tumour location and extent at all time instances during the patient breath-cycle for allowing accurate planning of free-breathing radiotherapy, it will most likely be a much more general and larger volume than a conventional stationary GTV. In addition to being larger, we can expect the shape to be somewhat more spherical or elliptical as any protrusions or specific surface details will move over a larger volume during breathing, which the IGTV needs to include. These differences are most likely to blame for the deviations in compactness distribution between the Lung1 and HUH, seen in Figure 4.13b.

The distributions in the two datasets were also compared with Lung1 restricted to only local-advanced patients, referred to as LA-Lung1. Restriction of Lung1 to LA-Lung1 did not in general significantly change the distribution of compactness feature values when comparing to HUH and hence it is unlikely that the large differences in distributions can be attributed to disease progression but rather to differing delineation protocols. In Figure C.3 are also histograms of sphericity and the surface-volume ratio, where the former is directly correlated with compactness and the latter relies on a similar definition. These show the same difference in distributions, i.e. the IGTV in HUH being generally more spherical (higher sphericity/lower surface-volume ratio) than the GTV in Lung1. The shape features Flatness and LeastAxisLength, that are not directly correlated with compactness, are shown in Figure C.4 and also present with a notably different distribution in HUH than in Lung1. These results indicate that shape features, especially the ones discussed, are severely dependent on delineation protocol. Lafata et al., 2018 examined the impacts of simulated noise and FB-motion on radiomic features, where all examined shape feature were highly susceptible to increasing motion, with many intensity- and texture features also displaying large changes from baseline values under FB segmentations.

Future work should include exploring translation of features from Lung1 to an LA-NSCLC cohort with a similar delineation protocol, such that differing segmentations do not undermine the differences in actual tumour volume and staging. Additional, it could be interesting to explore methods for potentially quantifying the impact of delineation protocols on radiomic features.

5.2.2 Volume considerations

As we have discovered, the three features in the signature not including compactness are strongly co-dependent on tumour volume and in fact poor descriptors of intra-tumour heterogeneity, contrary to the initial hypothesis. Hence the analyses presented in Section 4.4 give little information on the transferability of non-shape features due to the influence of volume on feature values.

The fit of the signature in the two cohorts did improve when Lung1 was restricted to LA-Lung1, since HLH GLNU had significant overlap between these two distributions, as opposed to the regular Lung1 population. This does not signify that LA-NSCLC exhibit more similar texture or first-order characteristics however, as the increased fit can be directly attributed to reducing Lung1 to a patient population with a narrower range of generally larger primary tumour volumes, which this feature is directly related to. Future prospects should be to volume-normalise affected features and measure if the significant similarity in the distributions between the two data sets is retained.

Between Lung1 and HUH, 28 out of 66 features used in this project had similar distributions, while this number increased to 33 for comparison between LA-Lung1 and HUH. It is possible that restricting treatment groups to more homogeneous pathology and staging could make a larger collection of radiomic features transferable between them, but with an unknown number of features being potentially correlated with tumour volume we can not say this conclusively with the investigations performed in this work. Again volume normalisation and voxel randomisation as done in Welch et al., 2019 are options for discovering potential false positives with regards to transferring radiomic models onto new data.

5.2.3 Texture and noise

It should also be noted that when comparing the complete Lung1 to HUH, the GLNU distributions were not significantly different ($p = 0.29$) while for the wavelet transformed HLH GLNU this was not the case ($p = 0.020$). This could be an indication of wavelet texture features being more susceptible to differences in noise- and intensity characteristics between different CT scanners. A wavelet transform decomposes an image into frequency detail decompositions which, in addition to highlighting directional details also amplify the noise in an image, potentially magnifying differences in scanner-specific noise prior to feature calculation. For example, Solomon et al., 2012 showed that the noise texture of CT scanners is highly dependent on the specific convolution kernel used during reconstruction. In Figure 4.10, GLNU (PCC = 0.96) is more strongly correlated with volume after being wavelet transformed into HLH GLNU (PCC = 0.98). This might indicate that the transform magnifies random noise in the image to such a degree that the original textural differences are lost, only leaving the dependency to volume. Bagher-Ebadian et al., 2017 showed that radiomic features on CT images of head and neck cancer were susceptible to high-pass filtering, which is the class of filters that the edge-enhancing detail decompositions in Figure 2.20 are a part of. Additionally, Traverso et al., 2018 report that slice thickness resampling and reconstruction algorithms strongly degraded feature reproducibility. This result could potentially be measured in the feature values of Lung1 and HUH seeing as they originate from different scanners, but the dominant influence of volume encumber such considerations.

The translational ability of features contoured with equal protocols should be explored further and viewed in the context of standardization efforts within radiomics. Also, vulnerabilities of certain delineation protocols and their direct effects on resulting features should be determined. Noise-characteristics between images from independent institutions and hardware can potentially have a large impact on texture and wavelet-texture features, and a standing question is if wavelet features can reliably enhance intra-tumour heterogeneous texture detail without losing information to image noise.

5.3 Implementation functionality

In this thesis, a platform for data-handling and radiomic feature calculation using *PyRadiomics* was implemented in Python. Its main aspects are custom classes for the sorting of patients and associated images, together with a multitude of independent functions design to work either on the aforementioned classes or on tabulated data for radiomic feature calculation and subsequent analysis. Overall, the implementation has performed its purpose well, but there is room for improvement both in simplicity for the user and in the overall architecture.

It seems feasible that the feature-calculating functions in `feature_extraction.py` (Appendix A.2) could be integrated into the `StudyGroup` class (Appendix A.1), and thus calculate features for the entire cohort. Then feature values could be assigned to each patient contained, and with an additional tabulated structure for the entire group, equivalent to the required input of must functions in `analysis.py` (Appendix A.3). A similar integration of the analysis functions could also be considered, but this would potentially involve more time-investment and experience-gathering on the author's part for performing, e.g. analysis and shared plots across instances of the `Patient` and `StudyGroup` classes. Despite the incoherent structure and with some sections of the implementation made redundant by later developments, it has performed very well as a tool for the candidate to collect image data and perform radiomic feature calculations with subsequent analysis. Note that the implementation's final design was not a result of carefully prepared planning or an end-goal strategy but rather a natural progression, where many smaller objectives and issues were solved with contemporary experience and knowledge of the author, along with trial and error. In the future, clear overarching goals regarding functionality and design could improve the structure and simplicity of the implementation.

It should also be noted that there is a *PyRadiomics*-extension for the Python-based free, open source medical imaging software *3DSlicer*[2]. Although not tested by the author, it appears to provide an integrated user experience in 3DSlicer, along with a graphical user interface for painless calculation of radiomic features.

[2]https://www.slicer.org/

5.4 Image material

Some attention should also be directed at the state of the Lung1 image material retrieved from TCIA. Around 25 patients had unexpected issues in their associated CT image files, ranging from randomly ordered CT images when sorted by filename, to either missing or empty segmentation slices. As described in Section 3.2.1, most problems regarding inconsistent image order could be fixed using slice location, but for missing or empty segmentations the patient was excluded. The corrections applied to the images by the implementation, did not affect the subsequent included images in any way that should impact the results of radiomic calculations. Moreover, the difference in results when comparing to Aerts et al., 2014 after the exclusion of six patients should be near negligible, due to the large size of Lung1. These unforeseen and poorly documented issues with publicly available data underline the importance of quality assurance of such image material.

Other studies using this dataset moreover report inconsistent numbers on cohort size, e.g. both Patil et al., 2016 and Hosny et al., 2018 report N = 317 on the dataset retrieved from TCIA, while Wu et al., 2016 state having initially N = 198 patients in Lung1 but do not cite if the data came from TCIA or some other source. It is unclear whether the inconsistent cohort sizes were due to exclusions within each study or from changes on the source material on TCIA. Absent reports in studies on excluded patients and image material from public data is problematic especially with regards to reproducing or validating studies that use the same publicly available data. More transparency on the usage of, and modifications done to public data, must be pursued in order to develop consistent and accurate understandings of both the methodology and results within the field.

The need for standardisation across several stages of the radiomic process is one of the most repeated arguments within the field for ensuring credibility in radiomic results. The sharing of consistent and transparent datasets is an important steppingstone for standardization in the radiomic workflow. Adherence to a broadly recognised standard such as the FAIR principles for scientific data management and stewardship (Wilkinson et al., 2016) can help legitimise radiomics within the scientific community and additional interested parties.

6

Conclusion

In this work, an independent implementation for calculating radiomic features using *PyRadiomics* was developed and applied to the NSCLC Lung1 and LA-NSCLC HUH cohorts. The results in Aerts et al., 2014 on Lung1 were successfully recreated and thus validating the implementation's functionality. It was shown that the prognostic significance in 3 out of 4 features in the radiomic signature was most likely exclusively due to the strong co-dependencies to volume. The feature that was not correlated with volume, Compactness2, showed no significant relation to overall survival in NSCLC patients. The three volume-confounded features also showed indication of reliable transferability between Lung1 and HUH, but this can likely be attributed to the calculation of volume being highly stable between hardware and institutions. Compactness2 presented with large differences in distribution of feature values between the two datasets. This difference can likely be attributed to the 4D-CT delineation protocol used for the HUH dataset giving a more generous and spherical tumour segmentation than in Lung1.

It is probable that volume has been the driving factor in the prognostic performance of the radiomic features examined in this work, and moreover influenced the investigations made for determining transferability between Lung1 and HUH. Exact measures on volume influence on prognostic and translational results was not a part of this project however, but would be a natural progression forward. Examining the performance of the signature with both volume-normalised features and randomised image intensity patterns could reveal the real performance and supplementary prognostic potential of the radiomic signature. An interesting avenue would also be to examine and determine the effect of differing delineation procedures on the different feature classes. Shape features appear to be particularly vulnerable, but it seems possible that intensity- and texture features could be affect as well, where e.g. 4D-CT IGTV contours often include non-tumour regions outside of the conventional GTV.

The main goal of radiomics is to provide supplementary information to known clinical parameters for aid in decision-making, but the intrinsic presence of volume in this radiomic signature made evaluation of its prognostic relevance and transferability unviable. The origin of the signature's volume-dependence can be traced back to the feature selection process and the "black box" approach to model development and validation. Necessary investigations into feature co-dependencies and other safeguards should be incorporated into the radiomic workflow at the feature selection stage together with ongoing standardisation initiatives.

Bibliography

Aerts, H. J. W. L., Velazquez, E. R., Leijenaar, R. T. H., Parmar, C., Grossmann, P., Carvalho, S., Cavalho, S., Bussink, J., Monshouwer, R., Haibe-Kains, B., Rietveld, D., Hoebers, F., Rietbergen, M. M., Leemans, C. R., Dekker, A., Quackenbush, J., Gillies, R. J., & Lambin, P. (2014). Decoding tumour phenotype by noninvasive imaging using a quantitative radiomics approach. *Nature Communications*, *5*, 4006. https://doi.org/10.1038/ncomms5006

Aerts, H. J. W. L., Wee, L., Rios Velazquez, E., Leijenaar, R. T. H., Parmar, C., Grossmann, P., Carvalho, S., Bussink, J., Monshouwer, R., Haibe-Kains, B., Rietveld, D., Hoebers, F., Rietbergen, M. M., Leemans, C. R., Dekker, A., Quackenbush, J., Gillies, R. J., & Lambin, P. (2019). *Data from NSCLC-radiomics* [Type: dataset]. https://doi.org/10.7937/K9/TCIA.2015.PF0M9REI

Anagnostopoulos, A. K., Gaitanis, A., Gkiozos, I., Athanasiadis, E. I., Chatziioannou, S. N., Syrigos, K. N., Thanos, D., Chatziioannou, A. N., & Papanikolaou, N. (2022). Radiomics/radiogenomics in lung cancer: Basic principles and initial clinical results [Number: 7 Publisher: Multidisciplinary Digital Publishing Institute]. *Cancers*, *14*(7), 1657. https://doi.org/10.3390/cancers14071657

Bagher-Ebadian, H., Siddiqui, F., Liu, C., Movsas, B., & Chetty, I. J. (2017). On the impact of smoothing and noise on robustness of CT and CBCT radiomics features for patients with head and neck cancers. *Medical Physics*, *44*(5), 1755–1770. https://doi.org/10.1002/mp.12188

Bangert, M. (2022). *CT reconstruction package*. Retrieved February 10, 2022, from https://www.mathworks.com/matlabcentral/fileexchange/34608-ct-reconstruction-package

Cancer registry of Norway. (2021). *Cancer in norway 2020 - cancer incidence, mortality, survival and prevalence in norway*. Cancer Registry of Norway. Oslo. https://www.kreftregisteret.no/Generelt/Rapporter/Cancer-in-Norway/cancer-in-norway-2020/

Colen, R. R., Fujii, T., Bilen, M. A., Kotrotsou, A., Abrol, S., Hess, K. R., Hajjar, J., Suarez-Almazor, M. E., Alshawa, A., Hong, D. S., Giniebra-Camejo, D., Stephen, B., Subbiah, V., Sheshadri, A., Mendoza, T., Fu, S., Sharma, P., Meric-Bernstam, F., & Naing, A. (2018). Radiomics to predict immunotherapy-induced pneumonitis: Proof of concept. *Investigational New Drugs*, *36*(4), 601–607. https://doi.org/10.1007/s10637-017-0524-2

Compton, C. C., Byrd, D. R., Garcia-Aguilar, J., Kurtzman, S. H., Olawaiye, A., & Washington, M. K. (Eds.). (2012). *AJCC cancer staging atlas*. Springer. https://doi.org/10.1007/978-1-4614-2080-4

Dai, H., Lu, M., Huang, B., Tang, M., Pang, T., Liao, B., Cai, H., Huang, M., Zhou, Y., Chen, X., Ding, H., & Feng, S.-T. (2021). Considerable effects of imaging sequences, feature extraction, feature selection, and classifiers on radiomics-based prediction of microvascular invasion in hepatocellular carcinoma using magnetic resonance imaging [Place: Shatin Publisher: Ame Publ Co WOS:000631988400014]. *Quantitative Imaging in Medicine and Surgery*, *11*(5), 1836–1853. https://doi.org/10.21037/qims-20-218

Fave, X., Zhang, L., Yang, J., Mackin, D., Balter, P., Gomez, D., Followill, D., Jones, A. K., Stingo, F., Liao, Z., Mohan, R., & Court, L. (2017). Delta-radiomics features for the prediction of patient outcomes in non–small cell lung cancer [Number: 1 Publisher: Nature Publishing Group]. *Scientific Reports*, *7*(1), 588. https://doi.org/10.1038/s41598-017-00665-z

Galloway, M. M. (1975). Texture analysis using gray level run lengths. *Computer Graphics and Image Processing*, *4*(2), 172–179. https://doi.org/10.1016/S0146-664X(75)80008-6

Gillies, R. J., Kinahan, P. E., & Hricak, H. (2016). Radiomics: Images are more than pictures, they are data. *Radiology*, *278*(2), 563–577. https://doi.org/10.1148/radiol.2015151169

Goldstraw, P., Chansky, K., Crowley, J., Rami-Porta, R., Asamura, H., Eberhardt, W. E. E., Nicholson, A. G., Groome, P., Mitchell, A., Bolejack, V., International Association for the Study of Lung Cancer Staging and Prognostic Factors Committee, Advisory Boards, and Participating Institutions, & International Association for the Study of Lung Cancer Staging and Prognostic Factors Committee Advisory Boards and Participating Institutions. (2016). The IASLC lung cancer staging project: Proposals for revision of the TNM stage groupings in the forthcoming (eighth) edition of the TNM classification for lung cancer. *Journal of Thoracic Oncology: Official Publication of the International Association for the Study of Lung Cancer*, *11*(1), 39–51. https://doi.org/10.1016/j.jtho.2015.09.009

Gonzalez, R. C., & Woods, R. E. (2007, August 31). *Digital image processing* (3rd edition). Pearson.

Gridelli, C., Rossi, A., Carbone, D. P., Guarize, J., Karachaliou, N., Mok, T., Petrella, F., Spaggiari, L., & Rosell, R. (2015). Non-small-cell lung cancer [Bandiera_abtest: a Cg_type: Nature Research Journals Number: 1 Primary_atype: Reviews Publisher: Nature Publishing Group Subject_term: Cancer genetics;Non-small-cell lung cancer;Targeted therapies Subject_term_id: cancer-genetics;non-small-cell-lung-cancer;targeted-therapies]. *Nature Reviews Disease Primers*, *1*(1), 1–16. https://doi.org/10.1038/nrdp.2015.9

Helsedirektoratet. (2021, December). *Nasjonalt handlingsprogram med retningslinjer for diagnostikk, behandling og oppfølging av lungekreft, mesoteliom og thymom* (No. 22). Helsedirektoratet. Oslo. Retrieved January 27, 2022, from https://www.helsedirektoratet.no/retningslinjer/lungekreft-mesoteliom-og-thymom-handlingsprogram#apiUrl

Hosny, A., Parmar, C., Coroller, T. P., Grossmann, P., Zeleznik, R., Kumar, A., Bussink, J., Gillies, R. J., Mak, R. H., & Aerts, H. J. W. L. (2018). Deep learning for lung cancer prognostication: A retrospective multi-cohort radiomics study [Publisher: Public Library of Science]. *PLOS Medicine*, *15*(11), e1002711. https://doi.org/10.1371/journal.pmed.1002711

ICRU. (2010). Report 83. *Journal of the International Commission on Radiation Units and Measurements*, *10*(1), NP. https://doi.org/10.1093/jicru/10.1.Report83

Lafata, K., Cai, J., Wang, C., Hong, J., Kelsey, C. R., & Yin, F.-F. (2018). Spatial-temporal variability of radiomic features and its effect on the classification of lung cancer histology. *Physics in Medicine and Biology*, *63*(22), 225003. https://doi.org/10.1088/1361-6560/aae56a

Lambin, P., Leijenaar, R. T. H., Deist, T. M., Peerlings, J., de Jong, E. E. C., van Timmeren, J., Sanduleanu, S., Larue, R. T. H. M., Even, A. J. G., Jochems, A., van Wijk, Y., Woodruff, H., van Soest, J., Lustberg, T., Roelofs, E., van Elmpt, W., Dekker, A., Mottaghy, F. M., Wildberger, J. E., & Walsh, S. (2017). Radiomics: The bridge between medical imaging and personalized medicine [Bandiera_abtest: a Cg_type: Nature Research Journals Number: 12 Primary_atype: Reviews Publisher: Nature Publishing Group Subject_term: Cancer genomics;Cancer imaging;Medical imaging;Personalized medicine Subject_term_id:

cancer-genomics;cancer-imaging;medical-imaging;personalized-medicine]. *Nature Reviews Clinical Oncology*, *14*(12), 749–762. https://doi.org/10.1038/nrclinonc.2017.141

Leijenaar, R. T. H., Carvalho, S., Hoebers, F. J. P., Aerts, H. J. W. L., van Elmpt, W. J. C., Huang, S. H., Chan, B., Waldron, J. N., O'sullivan, B., & Lambin, P. (2015). External validation of a prognostic CT-based radiomic signature in oropharyngeal squamous cell carcinoma. *Acta Oncologica (Stockholm, Sweden)*, *54*(9), 1423–1429. https://doi.org/10.3109/0284186X.2015.1061214

Maqbool, M. (2017, November 11). *An introduction to medical physics*. Springer.

Mayerhoefer, M. E., Materka, A., Langs, G., Häggström, I., Szczypiński, P., Gibbs, P., & Cook, G. (2020). Introduction to radiomics. *Journal of Nuclear Medicine: Official Publication, Society of Nuclear Medicine*, *61*(4), 488–495. https://doi.org/10.2967/jnumed.118.222893

Nestle, U., De Ruysscher, D., Ricardi, U., Geets, X., Belderbos, J., Pöttgen, C., Dziadiuszko, R., Peeters, S., Lievens, Y., Hurkmans, C., Slotman, B., Ramella, S., Faivre-Finn, C., McDonald, F., Manapov, F., Putora, P. M., LePéchoux, C., & Van Houtte, P. (2018). ESTRO ACROP guidelines for target volume definition in the treatment of locally advanced non-small cell lung cancer. *Radiotherapy and Oncology*, *127*(1), 1–5. https://doi.org/10.1016/j.radonc.2018.02.023

Papanikolaou, N., Matos, C., & Koh, D. M. (2020). How to develop a meaningful radiomic signature for clinical use in oncologic patients. *Cancer Imaging*, *20*, 33. https://doi.org/10.1186/s40644-020-00311-4

Patil, R., Mahadevaiah, G., & Dekker, A. (2016). An approach toward automatic classification of tumor histopathology of non-small cell lung cancer based on radiomic features. *Tomography (Ann Arbor, Mich.)*, *2*(4), 374–377. https://doi.org/10.18383/j.tom.2016.00244

Patil, R. B. (2020). *Prognostic and prediction modelling with radiomics for non-small cell lung cancer* (Doctoral Thesis). Maastricht University. Maastricht. https://doi.org/10.26481/dis.20201006rp

Rohatgi, A. (2021). *Webplotdigitizer: Version 4.5*. https://automeris.io/WebPlotDigitizer

Rosen, R. D., & Sapra, A. (2022, February 17). *TNM classification* [Publication Title: StatPearls [Internet]]. StatPearls Publishing. Retrieved May 26, 2022, from https://www.ncbi.nlm.nih.gov/books/NBK553187/

Roth, J. A., Hong, W. K., & Komaki, R. U. (2014). *Lung cancer*. John Wiley & Sons, Incorporated. Retrieved January 12, 2022, from http://ebookcentral.proquest.com/lib/bergen-ebooks/detail.action?docID=1658805

Sagerup, C. M. T., Småstuen, M., Johannesen, T. B., Helland, A., & Brustugun, O. T. (2012). Increasing age and carcinoma not otherwise specified: A 20-year population study of 40,118 lung cancer patients. *Journal of Thoracic Oncology: Official Publication of the International Association for the Study of Lung Cancer*, *7*(1), 57–63. https://doi.org/10.1097/JTO.0b013e3182307f7e

Shur, J. D., Doran, S. J., Kumar, S., ap Dafydd, D., Downey, K., O'Connor, J. P. B., Papanikolaou, N., Messiou, C., Koh, D.-M., & Orton, M. R. (2021). Radiomics in oncology: A practical guide [Publisher: Radiological Society of North America]. *RadioGraphics*, *41*(6), 1717–1732. https://doi.org/10.1148/rg.2021210037

Smith, N. B., & Webb, A. (2011). *Introduction to medical imaging: Physics, engineering and clinical applications* [Book Title: Introduction to medical imaging : physics, engineering and clinical applications]. University Press.

Sokol, M. (2021, May 25). *CT basic reconstruction algorithms* [original-date: 2020-10-16T22:00:27Z]. Retrieved March 1, 2022, from https://github.com/sokolmarek/CTBasicReconstruction

Solomon, J. B., Christianson, O., & Samei, E. (2012). Quantitative comparison of noise texture across CT scanners from different manufacturers. *Medical Physics*, *39*(10), 6048–6055. https://doi.org/10.1118/1.4752209

Song, J., Yin, Y., Wang, H., Chang, Z., Liu, Z., & Cui, L. (2020). A review of original articles published in the emerging field of radiomics. *European Journal of Radiology*, *127*, 108991. https://doi.org/10.1016/j.ejrad.2020.108991

Sung, H., Ferlay, J., Siegel, R. L., Laversanne, M., Soerjomataram, I., Jemal, A., & Bray, F. (2021). Global cancer statistics 2020: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA: a cancer journal for clinicians*, *71*(3), 209–249. https://doi.org/10.3322/caac.21660

Timmeren, J. E. v., Cester, D., Tanadini-Lang, S., Alkadhi, H., & Baessler, B. (2020). Radiomics in medical imaging—"how-to" guide and critical reflection. *Insights into Imaging*, *11*(1), 91. https://doi.org/10.1186/s13244-020-00887-2

Timmeren, J. E. v., Leijenaar, R. T. H., van Elmpt, W., Reymen, B., & Lambin, P. (2017). Feature selection methodology for longitudinal cone-beam CT radiomics. *Acta Oncologica (Stockholm, Sweden)*, *56*(11), 1537–1543. https://doi.org/10.1080/0284186X.2017.1350285

Traverso, A., Wee, L., Dekker, A., & Gillies, R. (2018). Repeatability and reproducibility of radiomic features: A systematic review. *International Journal of Radiation Oncology, Biology, Physics*, *102*(4), 1143–1158. https://doi.org/10.1016/j.ijrobp.2018.05.053

Urbanowicz, R. J., Meeker, M., La Cava, W., Olson, R. S., & Moore, J. H. (2018). Relief-based feature selection: Introduction and review. *Journal of Biomedical Informatics*, *85*, 189–203. https://doi.org/10.1016/j.jbi.2018.07.014

van Griethuysen, J. J., Fedorov, A., Parmar, C., Hosny, A., Aucoin, N., Narayan, V., Beets-Tan, R. G., Fillion-Robin, J.-C., Pieper, S., & Aerts, H. J. (2017). Computational radiomics system to decode the radiographic phenotype. *Cancer Research*, *77*(21), e104–e107. https://doi.org/10.1158/0008-5472.CAN-17-0339

Welch, M. L., McIntosh, C., Haibe-Kains, B., Milosevic, M. F., Wee, L., Dekker, A., Huang, S. H., Purdie, T. G., O'Sullivan, B., Aerts, H. J. W. L., & Jaffray, D. A. (2019). Vulnerabilities of radiomic signature development: The need for safeguards. *Radiotherapy and Oncology*, *130*, 2–9. https://doi.org/10.1016/j.radonc.2018.10.027

Wibmer, A., Hricak, H., Gondo, T., Matsumoto, K., Veeraraghavan, H., Fehr, D., Zheng, J., Goldman, D., Moskowitz, C., Fine, S. W., Reuter, V. E., Eastham, J., Sala, E., & Vargas, H. A. (2015). Haralick texture analysis of prostate MRI: Utility for differentiating non-cancerous prostate from prostate cancer and differentiating prostate cancers with different gleason scores. *European Radiology*, *25*(10), 2840–2850. https://doi.org/10.1007/s00330-015-3701-8

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., ... Mons, B. (2016). The FAIR guiding principles for scientific data management and stewardship [Number: 1 Publisher: Nature Publishing Group]. *Scientific Data*, *3*(1), 160018. https://doi.org/10.1038/sdata.2016.18

Wu, W., Parmar, C., Grossmann, P., Quackenbush, J., Lambin, P., Bussink, J., Mak, R., & Aerts, H. J. W. L. (2016). Exploratory study to identify radiomics classifiers for lung cancer histology. *Frontiers in Oncology*, *6*. Retrieved May 3, 2022, from https://www.frontiersin.org/article/10.3389/fonc.2016.00071

Yuan, M., Zhang, Y.-D., Pu, X.-H., Zhong, Y., Li, H., Wu, J.-F., & Yu, T.-F. (2017). Comparison of a radiomic biomarker with volumetric analysis for decoding tumour phenotypes of

lung adenocarcinoma with different disease-specific survival. *European Radiology*, *27*(11), 4857–4865. https://doi.org/10.1007/s00330-017-4855-3

Zwanenburg, A., Leger, S., Vallières, M., & Löck, S. (2020a). Image biomarker standardisation initiative - reference manual. *Radiology*, *295*(2), 328–338. https://doi.org/10.1148/radiol.2020191145

Zwanenburg, A., Vallières, M., Abdalah, M. A., Aerts, H. J. W. L., Andrearczyk, V., Apte, A., Ashrafinia, S., Bakas, S., Beukinga, R. J., Boellaard, R., Bogowicz, M., Boldrini, L., Buvat, I., Cook, G. J. R., Davatzikos, C., Depeursinge, A., Desseroit, M.-C., Dinapoli, N., Dinh, C. V., ... Löck, S. (2020b). The image biomarker standardization initiative: Standardized quantitative radiomics for high-throughput image-based phenotyping. *Radiology*, *295*(2), 328–338. https://doi.org/10.1148/radiol.2020191145

Appendices

A

Python code

A.1 Patient classes

Listing A.1: patient_classes.py

```
import os
import matplotlib.pyplot as plt
import numpy as np
import pydicom as dicom
import re
import cv2
import glob
from skimage.draw import polygon
from slice_viewer import IndexTracker


class Patient:

    def __init__(self, patientID, age, T_stage, N_stage, M_stage, overall_stage,
                 histology, gender, survival_time, deadstatus):
        self.patientID = patientID
        self.age = age
        self.T_stage = T_stage
        self.N_stage = N_stage
        self.M_stage = M_stage
        self.overall_stage = overall_stage
        self.histology = histology
        self.gender = gender
        self.survival_time = survival_time
        self.deadstatus = deadstatus

        # If needed, feature values can be assigned to the patient objects
        self.firstorder_features = None
        self.shapebased_features = None
        self.GLCM_features = None
        self.GLRLM_features = None
        self.HLH_features = None

    def __repr__(self):
        return str(self.patientID)

    @property
    def firstorder_features(self):
        return self.__firstorder_features

    @property
    def shapebased_features(self):
        return self.__shapebased_features

    @property
    def GLCM_features(self):
        return self.__GLCM_features

    @property
    def GLRLM_features(self):
        return self.__GLRLM_features
```

```
52
53      @property
54      def HLH_features(self):
55          return self.__HLH_features
56
57      @firstorder_features.setter
58      def firstorder_features(self, features):
59          self.__firstorder_features = features
60
61      @shapebased_features.setter
62      def shapebased_features(self, features):
63          self.__shapebased_features = features
64
65      @GLCM_features.setter
66      def GLCM_features(self, features):
67          self.__GLCM_features = features
68
69      @GLRLM_features.setter
70      def GLRLM_features(self, features):
71          self.__GLRLM_features = features
72
73      @HLH_features.setter
74      def HLH_features(self, features):
75          self.__HLH_features = features
76
77      # The methods to load dicoms from patients make assumption on the TCIA or HUH individual
78      # file structures to explicitly navigate the directories
79      def get_TCIA_images(self, path):
80          os.chdir(path)
81          # Managing if the user has provided a faulty or wrong path
82          try:
83              os.chdir(str(self.patientID))
84          except FileNotFoundError as e:
85              print(e, "\n")
86              print(f"\nError: The specified path is not a directory containing"
87                    f" the expected patient-ID: {self.patientID}")
88          else:
89              # Navigating TCIA file structure
90              os.chdir(os.listdir(os.getcwd())[0])
91              os.chdir(os.listdir(os.getcwd())[0])
92              images_dict = {}
93              # Retrieving images
94              for filename in os.listdir(os.getcwd()):
95                  dataset = dicom.dcmread(filename)
96                  location = dataset.SliceLocation
97                  image_array = dataset.pixel_array
98                  # Fixing images not being consistent across patients with pixel intensity range
99                  if np.min(image_array) < 0:
100                     image_array = image_array - np.min(image_array)
101                 images_dict.update({location: image_array})
102
103             # Sorting the dictionary by the numerical value of the keys, i.e. slice positions
104             sort = {k: v for k, v in sorted(images_dict.items(), key=lambda item: -item[0])}
105             final_array = np.array(list(sort.values()))
106
107             return final_array
108
109     def get_haukeland_data(self, path, structure="GTVp"):
110         os.chdir(os.path.join(path, str(self.patientID)))
111         ct_dict = dict()
112         masks = dict()
113
114         ct_filelist = glob.glob(os.path.join(os.getcwd(), r"CT*.dcm"))
115         rs_filename = glob.glob(os.path.join(os.getcwd(), r"RS*.dcm"))
116         if not ct_filelist or not rs_filename:
117             raise FileNotFoundError
118
119         for n in range(len(ct_filelist)):
120
121             # Handling CT-images
```

```
            ct = dicom.dcmread(ct_filelist[n])
            ct_dict.update({ct.ImagePositionPatient[2]: ct.pixel_array})

            # Handling segmentations
            # Extracting patient position from ct dicom
            patient_x = ct.ImagePositionPatient[0]
            patient_y = ct.ImagePositionPatient[1]
            patient_z = ct.ImagePositionPatient[2]
            ps = ct.PixelSpacing[0]

            seq = dicom.dcmread(rs_filename[0])
            # Finding the contournumber of the selected structure, such that we can extract it from ROIContourSequence
            structureNames = [seq.StructureSetROISequence[i].ROIName for i in range(len(seq.StructureSetROISequence))]
            contourNumber = [i for i, item in enumerate(structureNames) if re.search(structure, item)][0]

            ds = seq.ROIContourSequence[contourNumber].ContourSequence

            totalMask = np.zeros([ct.pixel_array.shape[0], ct.pixel_array.shape[1]])
            for element in ds:
                # If the UID of the contour matches the UID of the sequence, we retrieve the contour:
                if element.ContourImageSequence[0].ReferencedSOPInstanceUID == ct.SOPInstanceUID:
                    contour = np.array(element.ContourData)
                    # Each contoured point is stored sequentially; x1, y1, z1, x2, y2, z2, ...,
                    # so the array is reshaped thus the contour variable contains the coordinates of the
                    # contour line around the structure
                    contour = np.reshape(contour, (len(contour) // 3, 3))
                    # Make the contour into a mask:
                    contourMask = np.zeros([ct.pixel_array.shape[0], ct.pixel_array.shape[1]])
                    r, c = polygon((contour[:, 0] - patient_x) / ps, (contour[:, 1] - patient_y) / ps,
                                   contourMask.shape)
                    contourMask[r, c] = 1
                    totalMask += np.fliplr(np.rot90(contourMask, axes=(1, 0)))

            masks.update({patient_z: totalMask > 0})

        # Sorting the ct dict by image slice position:
        sorted_dict = {k: v for k, v in sorted(ct_dict.items(), key=lambda item: -item[0])}
        ct_images = np.array(list(sorted_dict.values()))

        # Sorting patient contours by slice position:
        sorted_contours = {k: v for k, v in sorted(masks.items(), key=lambda item: -item[0])}
        ct_masks = np.array(list(sorted_contours.values())).astype(np.uint8)

        return ct_images, ct_masks

    def get_TCIA_segmentations(self, path):
        os.chdir(path)
        try:
            os.chdir(str(self.patientID))
        except FileNotFoundError as e:
            print(f"\nError: The specified path is not a directory containing"
                  f" the expected patient-ID: {self.patientID}")
        else:
            os.chdir(os.listdir(os.getcwd())[0])
            for dirname in os.listdir(os.getcwd()):
                if re.search("Segmentation", dirname):
                    os.chdir(os.path.join(os.getcwd(), dirname))
            # TODO Perhaps add some handling here for if file not found

            filename = os.listdir(os.getcwd())[0]
            dataset = dicom.dcmread(filename)
            if dataset["PatientID"].value == self.patientID:
                pass
            else:
```

```
                print(f"Error: Patient object ID ({self.patientID}), does "
                      f"not correspond with the patient ID in the provided"
                      f"dataset ({dataset['PatientID'].value})")
                quit()

            segmentation_dict = {}
            for entry in dataset["SegmentSequence"]:
                segmentation_dict.update({entry["SegmentDescription"].value: None})

            total_array = dataset.pixel_array
            length, rows, cols = np.shape(total_array)

            # The array is split into equal sections, each section being the number of
            # images in the total segmentation array divided by the number of segmentations
            split_array = total_array.reshape(len(segmentation_dict), -1, rows, cols)

            for keyword in segmentation_dict:
                index = list(segmentation_dict.keys()).index(keyword)
                # The assigned array is flipped along the slice axis to correspond with CT image order
                segmentation_dict[keyword] = np.flipud(split_array[index, :, :, :])

            return segmentation_dict

    # A method for returning only the GTV segmentations, for calculating radiomics
    def get_TCIA_GTV_segmentations(self, path, return_dict=False):
        gtv_dict = {}
        segmentations = self.get_TCIA_segmentations(path)
        for volume in segmentations:
            match = re.search("GTV", volume)
            if match:
                gtv_dict.update({volume: segmentations[volume]})
        if gtv_dict == {}:
            print(f"Error: Found no segmentations of patient {self.patientID}"
                  f" tagged with 'GTV'\n")
            quit()
        if return_dict:
            return gtv_dict
        else:
            return gtv_dict["GTV-1"]

    # Method for viewing delineations on top of the CT images
    def view_segmentations(self, path, pathtype="TCIA", window_width=550, window_height=550):

        if pathtype == "TCIA":
            segmentations = self.get_TCIA_segmentations(path)
            ct_images = self.get_TCIA_images(path)
            print(f"Showing segmentations of patient {self.patientID}")
            print(f"Segmented volumes are: {list(segmentations.keys())}")

            ct_rgb_images = []
            for image in ct_images:
                image = cv2.normalize(
                    image, None, alpha=0, beta=255, norm_type=cv2.NORM_MINMAX, dtype=cv2.CV_8U
                )
                image = cv2.cvtColor(image, cv2.COLOR_GRAY2RGB)
                ct_rgb_images.append(image)
            ct_rgb_images = np.array(ct_rgb_images)

            for volume in segmentations:
                bw_array = segmentations[volume]
                rgb = [np.random.randint(0, 255), np.random.randint(0, 255), np.random.randint(0, 255)]
                for i in range(len(bw_array)):
                    bw_image = bw_array[i, :, :]
                    image = ct_rgb_images[i, :, :, :]
                    contours, _ = cv2.findContours(bw_image, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
                    ct_rgb_images[i, :, :, :] = cv2.drawContours(image, contours, -1, rgb, 2)

```

```
            # Viewing the result in the slice-viewer
            fig, ax = plt.subplots(1, 1)
            tracker = IndexTracker(ax, ct_rgb_images)
            fig.canvas.mpl_connect("scroll_event", tracker.on_scroll)
            fig.canvas.set_window_title(self.patientID)
            mngr = plt.get_current_fig_manager()
            mngr.resize(window_width, window_height)

            plt.show()

        elif pathtype == "HUH":
            ct_images, segmentations = self.get_haukeland_data(path, structure="GTV")
            segmentations = segmentations.astype(np.uint8)
            ct_rgb_images = []
            for image in ct_images:
                image = cv2.normalize(
                    image, None, alpha=0, beta=255, norm_type=cv2.NORM_MINMAX, dtype=cv2.CV_8U
                )
                image = cv2.cvtColor(image, cv2.COLOR_GRAY2RGB)
                ct_rgb_images.append(image)
            ct_rgb_images = np.array(ct_rgb_images)

            rgb = [np.random.randint(0, 255), np.random.randint(0, 255), np.random.randint(0, 255)
    ]

            for i in range(len(segmentations)):
                bw_image = segmentations[i, :, :]
                image = ct_rgb_images[i, :, :, :]
                contours, _ = cv2.findContours(bw_image, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
                ct_rgb_images[i, :, :, :] = cv2.drawContours(image, contours, -1, rgb, 2)

            fig, ax = plt.subplots(1, 1)
            tracker = IndexTracker(ax, ct_rgb_images)
            fig.canvas.mpl_connect("scroll_event", tracker.on_scroll)
            fig.canvas.set_window_title(self.patientID)
            mngr = plt.get_current_fig_manager()
            mngr.resize(window_width, window_height)

            plt.show()

        else:
            print("Error: Unrecognized pathtype")
            quit()

    def __str__(self):
        return f"{self.patientID}"


class StudyGroup:

    def __init__(self, groupID):
        self.patients = []
        self.groupID = groupID

    def __str__(self):
        result = "Group: \n"
        for patient in self.patients:
            result += str(patient) + "\n"
        return result

    def __getitem__(self, item):
        return self.patients[item]

    def __len__(self):
        return len(self.patients)

    def index(self, item):
        return self.patients.index(item)

    def add_patient(self, new_patient):
```

```
322         self.patients.append(new_patient)
323
324     def remove_patient(self, patientid):
325         for patient in self.patients:
326             if patient.patientID == patientid:
327                 self.patients.remove(patient)
328
329     def remove_multiple_patients(self, patients: list):
330         for name in patients:
331             self.remove_patient(name)
332
333     @staticmethod
334     # A private function for making other methods more compact
335     def __return_clinical_values(read_line):
336         if read_line[1] == "NA" or read_line[1] == "":
337             age = "NA"
338         else:
339             age = float(read_line[1])
340
341         if read_line[2] == "NA" or read_line[2] == "":
342             t_stage = "NA"
343         else:
344             t_stage = int(read_line[2])
345
346         if read_line[3] == "NA" or read_line[3] == "":
347             n_stage = "NA"
348         else:
349             n_stage = int(read_line[3])
350
351         if read_line[4] == "NA" or read_line[4] == "":
352             m_stage = "NA"
353         else:
354             m_stage = int(read_line[4])
355
356         overall_stage = str(read_line[5])
357
358         histology = str(read_line[6])
359
360         gender = str(read_line[7])
361
362         if read_line[8] == "NA" or read_line[8] == "":
363             survival_time = "NA"
364         else:
365             survival_time = int(read_line[8])
366
367         if read_line[9] == "NA" or read_line[9] == "":
368             deadstatus = "NA"
369         else:
370             deadstatus = int(read_line[9])
371
372         return age, t_stage, n_stage, m_stage, overall_stage, histology, gender, survival_time,
    deadstatus
373
374     # A method that can add HUH patients to the group using the directory containing the image
    data
375     # instead of csv
376     def add_HUH_patients(self, path):
377         os.chdir(path)
378         for dirname in os.listdir(os.getcwd()):
379             patient = Patient(dirname, None, None, None, None, None, None, None, None, None)
380             self.patients.append(patient)
381
382     def add_all_patients(self, path, pathtype="TCIA"):
383         file = open(path, "r")
384         for line in file.readlines()[1:]:
385             line = line.strip()
386             if pathtype == "TCIA":
387                 line = line.split(",")
388             elif pathtype == "HUH":
389                 line = line.split(";")
```

```
390             patient_id = str(line[0])
391             age, t, n, m, o, hist, g, st, dead = self.__return_clinical_values(line)
392             patient = Patient(patient_id, age, t, n, m, o, hist, g, st, dead)
393             self.patients.append(patient)
394         file.close()
395 
396     def add_specific_patients(self, path, patientnames: list):
397         file = open(path, "r")
398         for line in file.readlines()[1:]:
399             line = line.split(",")
400 
401             patient_id = str(line[0])
402             if patient_id in patientnames:
403                 age, t, n, m, o, hist, g, st, dead = self.__return_clinical_values(line)
404                 patient = Patient(patient_id, age, t, n, m, o, hist, g, st, dead)
405                 self.patients.append(patient)
406         file.close()
407 
408     def mean_age(self):
409         result = 0
410         invalid = 0
411         for patient in self.patients:
412             if patient.age == "NA":
413                 invalid += 1
414             else:
415                 result += np.floor(patient.age)
416         return result/(self.size() - invalid)
417 
418     def age_range(self):
419         ages = []
420         for patient in self.patients:
421             if patient.age != "NA":
422                 ages.append(patient.age)
423         return [min(ages), max(ages)]
424 
425     def size(self):
426         return len(self.patients)
427 
428     def relative_frequency_males(self):
429         result = 0
430         invalid = 0
431         for patient in self.patients:
432             if patient.gender == "NA":
433                 invalid += 1
434             elif patient.gender == "male":
435                 result += 1
436         return (result/(self.size() - invalid))*100
437 
438     def relative_frequency_females(self):
439         result = 0
440         invalid = 0
441         for patient in self.patients:
442             if patient.gender == "NA":
443                 invalid += 1
444             elif patient.gender == "female":
445                 result += 1
446         return (result/self.size())*100
447 
448     def relative_frequency_Tstages(self):
449         T1 = 0
450         T2 = 0
451         T3 = 0
452         T4 = 0
453         Tx = 0
454         for patient in self.patients:
455             if patient.T_stage == 1:
456                 T1 += 1
457             elif patient.T_stage == 2:
458                 T2 += 1
459             elif patient.T_stage == 3:
```

```
                T3 += 1
            elif patient.T_stage == 4:
                T4 += 1
            # NA entries are assumed to refer to Tx stages
            else:
                Tx += 1
        return [
            T1 / self.size()*100, T2 / self.size()*100,
            T3 / self.size()*100, T4 / self.size()*100,
            Tx / self.size()*100
        ]

    def relative_frequency_Nstages(self):
        N0 = 0
        N1 = 0
        N2 = 0
        N3 = 0
        Nx = 0
        for patient in self.patients:
            if patient.N_stage == 0:
                N0 += 1
            elif patient.N_stage == 1:
                N1 += 1
            elif patient.N_stage == 2:
                N2 += 1
            elif patient.N_stage == 3:
                N3 += 1
            # NA entries are assumed to refer to Nx stages
            else:
                Nx += 1
        return [
            N0 / self.size() * 100, N1 / self.size() * 100,
            N2 / self.size() * 100, N3 / self.size() * 100,
            Nx / self.size() * 100
        ]

    def relative_frequency_TNM(self):
        stage1 = 0
        stage2 = 0
        stage3a = 0
        stage3b = 0
        invalid = 0
        for patient in self.patients:
            if patient.overall_stage == "I":
                stage1 += 1
            elif patient.overall_stage == "II":
                stage2 += 1
            elif patient.overall_stage == "IIIa" or patient.overall_stage == "IIIA":
                stage3a += 1
            elif patient.overall_stage == "IIIb" or patient.overall_stage == "IIIB":
                stage3b += 1
            # There is a single patient which has "NA" overall stage in Lung1, and
            # coincidentally has a T stage of 5, which seems to have netted the
            # patient of being placed in the overall stage IIIb when the statistics
            # have been calculated in Aerts et al.
            else:
                stage3b += 1
        return [
            stage1 * 100 / (self.size() - invalid), stage2 * 100 / (self.size() - invalid),
            stage3a * 100 / (self.size() - invalid), stage3b * 100 / (self.size() - invalid)
        ]

    def print_statistics(self):
        T = self.relative_frequency_Tstages()
        N = self.relative_frequency_Nstages()
        TNM = self.relative_frequency_TNM()

        print("Males: ", self.relative_frequency_males())
        print("Females: ", self.relative_frequency_females())

```

```
        print("Mean age", self.mean_age())
        print("Age range", self.age_range())
        print()
        print("T1:", T[0])
        print("T2:", T[1])
        print("T3:", T[2])
        print("T4:", T[3])
        print("Tx:", T[4])
        print("T Sum:", sum(T))
        print()
        print("N0:", N[0])
        print("N1:", N[1])
        print("N2:", N[2])
        print("N3:", N[3])
        print("Nx:", N[4])
        print("N Sum:", sum(N))
        print()
        print("TNM:", np.array(TNM))
        print("TNM sum:", sum(TNM))


# This block is for debugging
if __name__ == '__main__':

    lung1_path = r"C:\Users\filip\Downloads\radiomics_data\NSCLC-Radiomics"
    huh_path = r"C:\Users\filip\Downloads\radiomics_data\HUH_data"
    csv_path = r"NSCLC Radiomics Lung1.clinical-version3-Oct 2019.csv"
    lung1_disq = ["LUNG1-014", "LUNG1-021", "LUNG1-085", "LUNG1-095", "LUNG1-128", "LUNG1-194"]
    huh_disq = ["26_radiomics_HUH", "27_radiomics_HUH", "28_radiomics_HUH"]

    lung1: StudyGroup = StudyGroup("lung1")
    lung1.add_all_patients(csv_path, pathtype="TCIA")
    lung1.remove_multiple_patients(lung1_disq)

    huh = StudyGroup("huh")
    huh.add_all_patients("HUH_clinical.csv", pathtype="HUH")
    huh.remove_multiple_patients(huh_disq)

    huh[13].view_segmentations(huh_path, pathtype="HUH")
```

A.2 Feature calculation

Listing A.2: feature_extraction.py

```
import logging
import os

import numpy as np
import pandas as pd
import radiomics

from patient_classes import Patient, StudyGroup
from radiomics import firstorder, shape, glcm, glrlm
import SimpleITK as sitk
import pywt

# Settings for the feature extractors, resampling is disabled by default so last two
# options can be ignored
settings = {'binWidth': 25,
            'interpolator': sitk.sitkBSpline,
            'resampledPixelSpacing': None}


def calculate_firstorder_features(patient_group, filepath, filetype, struc="GTVp", mute=True):
    # Logging:
    current_dir = os.path.dirname(os.path.realpath(__file__))
    log_file = os.path.join(current_dir, rf"feature_files\{patient_group.groupID}_firstorder_log.
    txt")
```

```
    handler = logging.FileHandler(filename=log_file, mode="w")
    formatter = logging.Formatter("%(levelname)s:%(name)s: %(message)s")
    handler.setFormatter(formatter)
    radiomics.logger.addHandler(handler)
    radiomics.logger.setLevel(logging.DEBUG)

    # A list for constructing the final total dataframe that will be returned to the user
    dataframes = list()

    # Looping through every patient in the group to calcualate each patients featurevalues
    for patient in patient_group:
        # Retrieving images and segmentations from the patient
        if filetype == "TCIA":
            images = sitk.GetImageFromArray(patient.get_TCIA_images(filepath))
            masks = sitk.GetImageFromArray(patient.get_TCIA_GTV_segmentations(filepath))
        elif filetype == "HUH":
            images, masks = patient.get_haukeland_data(filepath, structure=struc)
            images, masks = sitk.GetImageFromArray(images), sitk.GetImageFromArray(masks)
        else:
            print("Error: Uncrecognized filetype")
            quit()

        print(f"\nCalculating first-order features for patient {patient}")

        # Enabling features and extracting firstorder radiomic feature values
        firstorder_features = firstorder.RadiomicsFirstOrder(images, masks, **settings)
        firstorder_features.enableAllFeatures()
        # Standard deviation is not enabled by enableAllFeatures due to correlation with other features
        firstorder_features.enableFeatureByName("StandardDeviation", True)
        firstorder_features.execute()

        if not mute:
            for featurename in firstorder_features.featureValues.keys():
                print(f"Computed {featurename}: {firstorder_features.featureValues[featurename]}")
        # Turning the dict into a dataframe
        df = pd.DataFrame(firstorder_features.featureValues, index=[patient_group.index(patient)])
        df.insert(0, "PatientID", patient.patientID)
        # And appending the dataframe to the list of all dataframes
        dataframes.append(df)
    # Concatenating the list of dataframes into a single dataframe containing all features of all patients
    features_df = pd.concat(dataframes)
    # Returning the final dataframe
    pd.DataFrame.to_csv(
        features_df, os.path.join(current_dir, rf"feature_files\{patient_group.groupID}_firstorder.csv")
    )


def calculate_shape_features(patient_group, filepath, filetype, struc="GTVp", mute=True):

    current_dir = os.path.dirname(os.path.realpath(__file__))
    log_file = os.path.join(current_dir, rf"feature_files\{patient_group.groupID}_shape_log.txt")
    handler = logging.FileHandler(filename=log_file, mode="w")
    formatter = logging.Formatter("%(levelname)s:%(name)s: %(message)s")
    handler.setFormatter(formatter)
    radiomics.logger.addHandler(handler)
    radiomics.logger.setLevel(logging.DEBUG)

    # a list for constructing the final total dataframe that will be retured to the use
    dataframes = list()

    # Looping through every patient in the group and calculating each patients feature values
    for patient in patient_group:
        # Retrieving images and segmentations from the patient
        if filetype == "TCIA":
            images = sitk.GetImageFromArray(patient.get_TCIA_images(filepath))
            masks = sitk.GetImageFromArray(patient.get_TCIA_GTV_segmentations(filepath))
        elif filetype == "HUH":
```

```
            images, masks = patient.get_haukeland_data(filepath, structure=struc)
            images, masks = sitk.GetImageFromArray(images), sitk.GetImageFromArray(masks)
        else:
            print("Error: Uncrecognized filetype")
            quit()

        print(f"Calculating shape features for patient {patient}")

        # Enabling features and extracting the patient's radiomic shape features
        shape_features = shape.RadiomicsShape(images, masks, **settings)
        shape_features.enableAllFeatures()
        # Compactness is not enabled by enableAllFeatures due to relation to other features
        shape_features.enableFeatureByName("Compactness2", True)
        shape_features.execute()

        if not mute:
            for featurename in shape_features.featureValues.keys():
                print(f"Computed {featurename}: {shape_features.featureValues[featurename]}")

        df = pd.DataFrame(shape_features.featureValues, index=[patient_group.index(patient)])
        df.insert(0, "PatientID", patient.patientID)
        dataframes.append(df)

    features_df = pd.concat(dataframes)
    pd.DataFrame.to_csv(
        features_df, os.path.join(current_dir, rf"feature_files\{patient_group.groupID}_shape.csv"
    )
    )


def calculate_GLCM_features(patient_group, filepath, filetype, struc, mute=True):

    current_dir = os.path.dirname(os.path.realpath(__file__))
    log_file = os.path.join(current_dir, rf"feature_files\{patient_group.groupID}_GLCM_log.txt")
    handler = logging.FileHandler(filename=log_file, mode="w")
    formatter = logging.Formatter("%(levelname)s:%(name)s: %(message)s")
    handler.setFormatter(formatter)
    radiomics.logger.addHandler(handler)
    radiomics.logger.setLevel(logging.DEBUG)

    dataframes = list()

    for patient in patient_group:
        if filetype == "TCIA":
            images = sitk.GetImageFromArray(patient.get_TCIA_images(filepath))
            masks = sitk.GetImageFromArray(patient.get_TCIA_GTV_segmentations(filepath))
        elif filetype == "HUH":
            images, masks = patient.get_haukeland_data(filepath, structure=struc)
            images, masks = sitk.GetImageFromArray(images), sitk.GetImageFromArray(masks)
        else:
            print("Error: Uncrecognized filetype")
            quit()

        print(f"Calculating GLCM features for patient {patient}")

        glcm_features = glcm.RadiomicsGLCM(images, masks, **settings)
        glcm_features.enableAllFeatures()
        glcm_features.execute()

        if not mute:
            for featurename in glcm_features.featureValues.keys():
                print(f"Computed {featurename}: {glcm_features.featureValues[featurename]}")

        df = pd.DataFrame(glcm_features.featureValues, index=[patient_group.index(patient)])
        df.insert(0, "PatientID", patient.patientID)
        dataframes.append(df)

    features_df = pd.concat(dataframes)
    pd.DataFrame.to_csv(
        features_df, os.path.join(current_dir, rf"feature_files\{patient_group.groupID}_GLCM.csv")
```

```
    )


def calculate_GLRLM_features(patient_group, filepath, filetype, struc, mute=True):

    current_dir = os.path.dirname(os.path.realpath(__file__))
    log_file = os.path.join(current_dir, rf"feature_files\{patient_group.groupID}_GLRLM_log.txt")
    handler = logging.FileHandler(filename=log_file, mode="w")
    formatter = logging.Formatter("%(levelname)s:%(name)s: %(message)s")
    handler.setFormatter(formatter)
    radiomics.logger.addHandler(handler)
    radiomics.logger.setLevel(logging.DEBUG)

    dataframes = list()

    for patient in patient_group:
        if filetype == "TCIA":
            images = sitk.GetImageFromArray(patient.get_TCIA_images(filepath))
            masks = sitk.GetImageFromArray(patient.get_TCIA_GTV_segmentations(filepath))
        elif filetype == "HUH":
            images, masks = patient.get_haukeland_data(filepath, structure=struc)
            images, masks = sitk.GetImageFromArray(images), sitk.GetImageFromArray(masks)
        else:
            print("Error: Uncrecognized filetype")
            quit()

        print(f"Calculating GLRLM features for patient {patient}")

        glrlm_features = glrlm.RadiomicsGLRLM(images, masks, **settings)
        glrlm_features.enableAllFeatures()
        glrlm_features.execute()

        if not mute:
            for featurename in glrlm_features.featureValues.keys():
                print(f"Computed {featurename}: {glrlm_features.featureValues[featurename]}")

        df = pd.DataFrame(glrlm_features.featureValues, index=[patient_group.index(patient)])
        df.insert(0, "PatientID", patient.patientID)
        dataframes.append(df)

    features_df = pd.concat(dataframes)
    pd.DataFrame.to_csv(
        features_df, os.path.join(current_dir, rf"feature_files\{patient_group.groupID}_GLRLM.csv"
    )
    )


def calculate_HLHGLRLM_features(patient_group, filepath, filetype, struc, mute=True):

    current_dir = os.path.dirname(os.path.realpath(__file__))
    log_file = os.path.join(current_dir, rf"feature_files\{patient_group.groupID}_HLH_GLRLM_log.
    txt")
    handler = logging.FileHandler(filename=log_file, mode="w")
    formatter = logging.Formatter("%(levelname)s:%(name)s: %(message)s")
    handler.setFormatter(formatter)
    radiomics.logger.addHandler(handler)
    radiomics.logger.setLevel(logging.DEBUG)

    dataframes = list()

    for patient in patient_group:
        # CT images are not made into sitk images yet, as the wavelet transform uses numpy array
        if filetype == "TCIA":
            images = patient.get_TCIA_images(filepath)
            masks = patient.get_TCIA_GTV_segmentations(filepath)
        elif filetype == "HUH":
            images, masks = patient.get_haukeland_data(filepath, structure=struc)
        else:
            print("Error: Uncrecognized filetype")
            quit()
```

```
228         # Transform must have even dimensional images to work, so if the number of slices
229         # is not even, the image array is padded with an extra black slice
230         slices, rows, cols = np.shape(images)
231         if slices % 2 != 0:
232             images = np.append(images, [np.zeros([rows, cols])], axis=0)
233             masks = np.append(masks, [np.zeros([rows, cols])], axis=0)
234 
235         print(f"Calculating HLH texture features for patient {patient}")
236 
237         # Taking the stationary (undecemated) wavelet transform of the ct images,
238         # which returns a list of 8 dicts, one dict for each decomposition
239         decomp = pywt.swtn(images, "coif1", level=1)[0]
240         # We are interested in the HLH (i.e. dad) decomposition of the image, as it is the one
241         # used for the radiomic signature in the study
242         HLH = decomp["dad"]
243 
244         wavelet_images = sitk.GetImageFromArray(HLH)
245         sitkmasks = sitk.GetImageFromArray(masks)
246 
247         # The size of each decomposition is the same as the original, so we can use the same
248         # masks for feature calculation
249         glrlm_wavelet = glrlm.RadiomicsGLRLM(wavelet_images, sitkmasks, **settings)
250         glrlm_wavelet.enableAllFeatures()
251         glrlm_wavelet.execute()
252 
253         # Looping through the dict and changing featurenames to differentiate from regular texture
    featurenames
254         new_dict = dict()
255         for featurekey, featureval in glrlm_wavelet.featureValues.items():
256             new_dict.update({"HLH " + featurekey: featureval})
257             if not mute:
258                 print(f"Computed HLH {featurekey}: {featureval}")
259 
260         # Using the new dict as the basis for forming the complete dataframe
261         df = pd.DataFrame(new_dict, index=[patient_group.index(patient)])
262         df.insert(0, "PatientID", patient.patientID)
263         dataframes.append(df)
264 
265     features_df = pd.concat(dataframes)
266     pd.DataFrame.to_csv(
267             features_df, os.path.join(current_dir, rf"feature_files\{patient_group.groupID}
    _HLH_GLRLM.csv")
268         )
```

A.3 Analysis

Listing A.3: feature_analysis.py

```
1  import matplotlib.pyplot as plt
2  from matplotlib.cm import ScalarMappable
3  import pandas as pd
4  from lifelines import KaplanMeierFitter
5  from lifelines.statistics import logrank_test
6  import numpy as np
7  from lifelines import CoxPHFitter
8  import re
9  from resampy import resample
10 from scipy.stats import ks_2samp, cramervonmises_2samp, mannwhitneyu, pearsonr
11 from matplotlib.legend import Legend
12 from lifelines.utils import k_fold_cross_validation
13 from sklearn import tree, linear_model
14 from sklearn.model_selection import train_test_split, cross_validate, cross_val_score
15 from sklearn.preprocessing import StandardScaler
16 
17 
18 lung1_firstorder = pd.read_csv(r"feature_files\lung1_firstorder.csv")
19 lung1_shape = pd.read_csv(r"feature_files\lung1_shape.csv")
20 lung1_glrlm = pd.read_csv(r"feature_files\lung1_GLRLM.csv")
```

```
lung1_hlh = pd.read_csv(r"feature_files\lung1_HLH_GLRLM.csv")

huh_firstorder = pd.read_csv(r"feature_files\HUH_firstorder.csv")
huh_shape = pd.read_csv(r"feature_files\HUH_shape.csv")
huh_glrlm = pd.read_csv(r"feature_files\HUH_GLRLM.csv")
huh_hlh = pd.read_csv(r"feature_files\HUH_HLH_GLRLM.csv")

huh_clinical = pd.read_csv(
    r"HUH_clinical.csv", delimiter=";"
)

lung1_clinical = pd.read_csv(
    r"NSCLC Radiomics Lung1.clinical-version3-Oct 2019.csv"
)
disq_lung1 = ["LUNG1-014", "LUNG1-021", "LUNG1-085", "LUNG1-095", "LUNG1-194", "LUNG1-128"]
for i in disq_lung1:
    lung1_clinical = lung1_clinical.drop(lung1_clinical[lung1_clinical.PatientID == i].index[0])
# 14_radiomics_HUH is large volume outlier
disq_huh = ["26_radiomics_HUH", "27_radiomics_HUH", "28_radiomics_HUH"]
for i in disq_huh:
    huh_clinical = huh_clinical.drop(huh_clinical[huh_clinical.PatientID == i].index[0])

lung1_firstorder = pd.merge(lung1_firstorder, lung1_clinical, on="PatientID", how="outer")
lung1_shape = pd.merge(lung1_shape, lung1_clinical, on="PatientID", how="outer")
lung1_glrlm = pd.merge(lung1_glrlm, lung1_clinical, on="PatientID", how="outer")
lung1_hlh = pd.merge(lung1_hlh, lung1_clinical, on="PatientID", how="outer")

huh_firstorder = pd.merge(huh_firstorder, huh_clinical, on="PatientID", how="inner")
huh_shape = pd.merge(huh_shape, huh_clinical, on="PatientID", how="inner")
huh_glrlm = pd.merge(huh_glrlm, huh_clinical, on="PatientID", how="inner")
huh_hlh = pd.merge(huh_hlh, huh_clinical, on="PatientID", how="inner")


def compare_km(feature: str):
    if feature == "Energy":
        ref_over = pd.read_csv("automeris_coords/lung1energy_overmedian.csv", delimiter=";",
    decimal=",",
                               header=None)
        ref_under = pd.read_csv("automeris_coords/lung1energy_undermedian.csv", delimiter=";",
    decimal=",",
                                header=None)
        df = lung1_firstorder

    if feature == "Compactness2":
        ref_over = pd.read_csv("automeris_coords/lung1compactness_overmedian.csv", delimiter=";",
    decimal=",",
                               header=None)
        ref_under = pd.read_csv("automeris_coords/lung1compactness_undermedian.csv", delimiter=";"
    , decimal=",",
                                header=None)
        df = lung1_shape

    if feature == "GrayLevelNonUniformity":
        ref_over = pd.read_csv("automeris_coords/lung1glnu_overmedian.csv", delimiter=";", decimal
    =",",
                               header=None)
        ref_under = pd.read_csv("automeris_coords/lung1glnu_undermedian.csv", delimiter=";",
    decimal=",",
                                header=None)
        df = lung1_glrlm

    if feature == "HLH GrayLevelNonUniformity":
        ref_over = pd.read_csv("automeris_coords/lung1hlhglnu_overmedian.csv", delimiter=";",
    decimal=",",
                               header=None)
        ref_under = pd.read_csv("automeris_coords/lung1hlhglnu_undermedian.csv", delimiter=";",
    decimal=",",
                                header=None)
        df = lung1_hlh

```

```
    lin1, lin2 = plot_km(df, feature, df[feature].median(), "Lung1")

    lin3, = plt.plot(ref_over[0], ref_over[1], color="red", linestyle="--")
    lin4, = plt.plot(ref_under[0], ref_under[1], color="red")

    plt.gca().legend([lin2, lin4], ["This implementation", "Aerts et al."], loc=1)

    leg = Legend(plt.gca(), [lin4, lin3], ["<= median", "> median"], loc=3)
    plt.gca().add_artist(leg)

    plt.title(f"{feature}")
    lines = leg.get_lines()
    for line in lines:
        line.set_color("black")


def plot_km(dataframe, parameter: str, threshold, groupname: str, xlim=1500):
    group1 = dataframe[dataframe[parameter] > threshold]
    group2 = dataframe[dataframe[parameter] <= threshold]

    t1 = group1["Survival.time"]
    t2 = group2["Survival.time"]
    e1 = group1["deadstatus.event"]
    e2 = group2["deadstatus.event"]

    kmf = KaplanMeierFitter()
    fig, ax = plt.subplots()

    kmf.fit(t1, e1)
    l1, = plt.plot(kmf.survival_function_.index, kmf.survival_function_["KM_estimate"], color="blue",
                   linestyle="--", label="> median")

    kmf.fit(t2, e2)
    l2, = plt.plot(kmf.survival_function_.index, kmf.survival_function_["KM_estimate"], color="blue",
                   label="<= median")

    # TODO if logrank test still unreliable, this test of our data can be used to quantify difference
    lr_result = logrank_test(t1, t2, e1, e2)
    pval = lr_result.p_value

    plt.gca().set_xlim(0, xlim)
    plt.gca().legend([l2, l1], ["<= median", "> median"])
    plt.legend()
    plt.title(f"{groupname} {parameter}, Logrank P-value = {pval.__round__(5)}")
    plt.ylabel("Survival probability")
    plt.xlabel("Survival time (days)")

    fig.set_figwidth(8)
    fig.set_figheight(5)

    return l1, l2


def signature_cox_model(modeltype="radiomics", mute=False):
    # --------------- dfs for radiomics model ---------------- #
    energy = pd.DataFrame(lung1_firstorder["Energy"] > lung1_firstorder["Energy"].median())
    comp = lung1_shape["Compactness2"] > lung1_shape["Compactness2"].median()
    text = lung1_glrlm["GrayLevelNonUniformity"] > lung1_glrlm["GrayLevelNonUniformity"].median()
    wave = lung1_hlh["HLH GrayLevelNonUniformity"] > lung1_hlh["HLH GrayLevelNonUniformity"].median()
    time = lung1_firstorder["Survival.time"]
    event = lung1_firstorder["deadstatus.event"]

    # ---------------- dfs for basic clinical moodel ------------- #
    age = pd.DataFrame(lung1_firstorder["age"].round())
    sex = pd.get_dummies(lung1_firstorder["gender"]).drop("male", axis=1).rename(columns={"female": "gender"})
```

```
148
149     stage = list()
150     for row in lung1_firstorder["Overall.Stage"]:
151         if row == "I":
152             stage.append({"Overall.stage": 1})
153         elif row == "II":
154             stage.append({"Overall.stage": 2})
155         elif row == "IIIa" or row == "IIIb":
156             stage.append({"Overall.stage": 3})
157         else:
158             stage.append({"Overall.stage": pd.NA})
159     stage = pd.DataFrame(stage)
160
161     # -------------- dfs for tnm model --------------- #
162     t = pd.DataFrame(lung1_firstorder["clinical.T.Stage"]).astype(int)
163     n = lung1_firstorder["Clinical.N.Stage"]
164     m = lung1_firstorder["Clinical.M.Stage"]
165
166     volume = pd.DataFrame(lung1_shape["VoxelVolume"])
167
168     if modeltype == "radiomics":
169         df = energy.join([comp, text, wave, time, event])
170         # Renaming for brevity
171         df.rename(columns={"GrayLevelNonUniformity": "GLNU", "HLH GrayLevelNonUniformity": "HLH
    GLNU"}, inplace=True)
172
173     elif modeltype == "clinical":
174         df = age.join([sex, stage, time, event])
175         df = df.dropna()
176
177     elif modeltype == "tnm":
178         print(t)
179         print(n)
180         print(m)
181         df = t.join([n, m, stage, time, event])
182         df = df.dropna()
183
184     elif modeltype == "volume":
185         df = volume.join([time, event])
186
187     else:
188         print("No valid modeltype")
189         quit()
190
191     fitter = CoxPHFitter()
192     fitter.fit(df, duration_col="Survival.time", event_col="deadstatus.event")
193
194     if not mute:
195         fitter.print_summary(decimals=3)
196         fitter.plot()  # Plots regression coefficients with 95% confidence intervals
197         plt.show()
198
199     return fitter, df
200
201
202 def plot_signature_km():
203     df = pd.DataFrame(lung1_firstorder["Energy"])
204     df = df.join(
205         [lung1_shape["Compactness2"], lung1_glrlm["GrayLevelNonUniformity"], lung1_hlh["HLH
    GrayLevelNonUniformity"]]
206     )
207
208     huh_df = pd.concat(
209         [huh_firstorder["Energy"], huh_shape["Compactness2"], huh_glrlm["GrayLevelNonUniformity"],
210          huh_hlh["HLH GrayLevelNonUniformity"]], axis=1)
211
212     cph, train = signature_cox_model(modeltype="radiomics", mute=True)
213     weights = cph.params_
214
215     combined = pd.DataFrame(df["Energy"] * weights["Energy"] + df["Compactness2"] * weights["
```

```
    Compactness2"]
                            + df["GrayLevelNonUniformity"] * weights["GLNU"]
                            + df["HLH GrayLevelNonUniformity"] * weights["HLH GLNU"])

    huh_combined = pd.DataFrame(huh_df["Energy"] * weights["Energy"] + huh_df["Compactness2"] *
    weights["Compactness2"]
                            + huh_df["GrayLevelNonUniformity"] * weights["GLNU"]
                            + huh_df["HLH GrayLevelNonUniformity"] * weights["HLH GLNU"])

    combined = combined.join([lung1_firstorder["Survival.time"], lung1_firstorder["deadstatus.
    event"]])
    # Reuturning this to allow combined signature to be used in other functions
    total_combined = combined

    threshold = combined[0].median()
    group1 = combined[combined[0] > threshold]
    group2 = combined[combined[0] <= threshold]

    t1 = group1["Survival.time"]
    t2 = group2["Survival.time"]
    e1 = group1["deadstatus.event"]
    e2 = group2["deadstatus.event"]

    kmf = KaplanMeierFitter()
    fig, ax = plt.subplots()

    kmf.fit(t1, e1)
    lin1, = plt.plot(kmf.survival_function_.index, kmf.survival_function_["KM_estimate"],
                     color="blue", linestyle="--", label="> median")

    kmf.fit(t2, e2)
    lin2, = plt.plot(kmf.survival_function_.index, kmf.survival_function_["KM_estimate"],
                     color="blue", label="<= median")

    ref_over = pd.read_csv("automeris_coords/lung1overall_overmedian.csv", delimiter=";", decimal=
    ",", header=None)
    ref_under = pd.read_csv("automeris_coords/lung1overall_undermedian.csv", delimiter=";",
    decimal=",", header=None)
    lin3, = plt.plot(ref_over[0], ref_over[1], color="red", linestyle="--")
    lin4, = plt.plot(ref_under[0], ref_under[1], color="red")

    plt.gca().set_xlim(0, 1500)
    plt.gca().legend([lin2, lin1], ["<= median", "> median"])
    plt.legend()
    plt.title(f"Combined signature")
    plt.ylabel("Survival probability")
    plt.xlabel("Survival time (days)")

    plt.gca().legend([lin2, lin4], ["This implementation", "Aerts et al."], loc=1)
    leg = Legend(plt.gca(), [lin4, lin3], ["<= median", "> median"], loc=3)
    plt.gca().add_artist(leg)
    lines = leg.get_lines()
    for line in lines:
        line.set_color("black")

    fig.set_figwidth(8)
    fig.set_figheight(5)

    return total_combined, huh_combined, lin3, lin4


# Comparing the histogram of a feature value across the two cohorts
def compare_histograms(df1, df2, featurename):
    if re.match("LUNG1", df1.iloc[0].PatientID):
        label1 = "Lung1"
        label2 = "HUH"
    else:
        label1 = "HUH"
        label2 = "Lung1"

```

```
    df1 = df1[featurename]
    df2 = df2[featurename]

    minimum = min(df1.min(), df2.min())
    maximum = max(df1.max(), df2.max())
    # Kolmogorov-Smirnov test
    stat, p = ks_2samp(df1, df2)

    binning = np.linspace(minimum, maximum, 30)
    fig, ax = plt.subplots()
    ax.hist(df1, density=True, alpha=0.7, bins=binning, edgecolor="black", label=label1, color="blue")
    ax.hist(df2, density=True, alpha=0.7, bins=binning, edgecolor="black", label=label2, color="orange")
    ax.set_xlabel("Feature value")
    ax.set_ylabel("Relative frequency")
    fig.set_figwidth(8)
    fig.set_figheight(5)
    plt.title(f"{featurename}, KS-statistic = {stat.__round__(5)}, p-value = {p.__round__(5)}")

    plt.legend()
    plt.show()


def test_featuregroup(df1, df2, k=None, log=False, tight=False):
    df1 = df1.drop(["PatientID", "age", "Overall.Stage", "Histology", "gender", "deadstatus.event",
                    "Survival.time", "Unnamed: 0", "Clinical.M.Stage", "clinical.T.Stage", "Clinical.N.Stage"], axis=1)
    df2 = df2.drop(["PatientID", "age", "Overall.Stage", "Histology", "gender", "deadstatus.event",
                    "Survival.time", "Unnamed: 0", "Clinical.M.Stage", "clinical.T.Stage", "Clinical.N.Stage"], axis=1)

    result = dict()
    for col in df1:
        col1 = df1[col]
        col2 = df2[col]
        stat, p = ks_2samp(col1, col2)
        result.update({col: p})
    result = pd.DataFrame(result, index=[0]).transpose()

    if k is not None:
        result = result.nlargest(k, columns=0)
    result = result.sort_values(by=0)

    fig, ax = plt.subplots()
    if log:
        plt.xscale("log")

    cc = list(map(lambda x: 'indianred' if x < 0.05 else 'olivedrab', result[0]))
    bars = ax.barh(result.index, result[0], edgecolor="black", color=cc)

    plt.xlabel("p-value")
    ax.bar_label(bars)
    plt.axvline(x=0.05, linewidth=1.7, color="black", linestyle="--")

    if tight:
        plt.tight_layout()
    plt.show()
    return result


def test_all_features(k=10, lung1_la=False):
    if lung1_la:
        lung1_df = la_lung1_firstorder.merge(la_lung1_shape)
        lung1_df = lung1_df.merge(la_lung1_glrlm)
        lung1_df = lung1_df.merge(la_lung1_hlh)

    else:
```

```
        lung1_df = lung1_firstorder.merge(lung1_shape)
        lung1_df = lung1_df.merge(lung1_glrlm)
        lung1_df = lung1_df.merge(lung1_hlh)

    lung1_df = lung1_df.drop(["PatientID", "age", "Overall.Stage", "Histology", "gender", "deadstatus.event",
                              "Survival.time", "Unnamed: 0", "Clinical.M.Stage", "clinical.T.Stage",
                              "Clinical.N.Stage"], axis=1)
    huh_df = huh_firstorder.merge(huh_shape)
    huh_df = huh_df.merge(huh_glrlm)
    huh_df = huh_df.merge(huh_hlh)

    huh_df = huh_df.drop(["PatientID", "age", "Overall.Stage", "Histology", "gender", "deadstatus.event",
                          "Survival.time", "Unnamed: 0", "Clinical.M.Stage", "clinical.T.Stage", "Clinical.N.Stage"],
                         axis=1)

    result = dict()
    for col in lung1_df:
        col1 = lung1_df[col]
        col2 = huh_df[col]
        stat, p = ks_2samp(col1, col2)
        result.update({col: p})

    result = pd.DataFrame(result, index=[0]).transpose()
    result = result.nlargest(k, columns=0)
    result = result.sort_values(by=0)

    cc = list(map(lambda x: 'indianred' if x < 0.05 else 'olivedrab', result[0]))

    fig, ax = plt.subplots()
    bars = ax.barh(result.index, result[0].round(3), height=0.5, edgecolor="black", color=cc)
    plt.xlabel("p-value")
    ax.bar_label(bars)
    plt.axvline(x=0.05, linewidth=1.7, color="black", linestyle="--")
    if lung1_la:
        plt.title(f"{k} most similarly distributed features, LA")
    else:
        plt.title(f"{k} most similarly distributed features")
    plt.tight_layout()
    return result


def thresholded_histograms(df, feature: str, clinical: str):
    thresh = df[feature].median()
    df1 = df[df[feature] <= thresh]
    df2 = df[df[feature] > thresh]

    df1 = df1[clinical]
    df2 = df2[clinical]

    n = df.nunique(axis=0)[clinical]
    binning = np.arange(-0.5, n + 1, 1)
    lab = [f"{feature} <= median", f"{feature} > median"]

    plt.hist([df1, df2], bins=binning, color=["b", "r"], rwidth=0.5, label=lab)
    plt.title(f"{clinical} above and below the median value of {feature}")
    plt.xlabel(clinical)
    plt.ylabel("n")
    plt.legend()
    plt.show()


def regressor_selection(df_list: list):
    lst = list()
    for featuregroup in df_list:
        x = featuregroup.drop(
            labels=["Unnamed: 0", "Survival.time", "PatientID", "Overall.Stage", "Histology", "
```

```
    deadstatus.event", "gender",
                    "clinical.T.Stage", "Clinical.N.Stage", "Clinical.M.Stage", "age"],
            axis=1
        )
        lst.append(x)
    X = pd.concat(lst, axis=1)

    # Removing duplicate columns
    X = X.loc[:, ~X.columns.duplicated()]
    Y = df_list[0]["Survival.time"]
    X_train, X_val, Y_train, Y_val = train_test_split(X, Y, random_state=666, train_size=0.8)

    for name, values in X_train.iteritems():
        X[name] = (values - values.mean()) / values.std()

    print(X_train)
    model = linear_model.LassoCV(cv=5, max_iter=1000000)
    model.fit(X_train, Y_train)
    alpha = model.alpha_
    lasso_best = linear_model.Lasso(alpha=alpha, max_iter=1000000)
    lasso_best.fit(X_train, Y_train)
    print(list(zip(lasso_best.coef_, X)))

    print("Score: ", lasso_best.score(X_val, Y_val))
    Y_pred = lasso_best.predict(X_val)
    plt.scatter(X_val.index, Y_val)
    plt.scatter(X_val.index, Y_pred)
    plt.show()


def separate_LA_lung1():
    new_firstorder = lung1_firstorder[
        (lung1_firstorder["Overall.Stage"] != "I") & (lung1_firstorder["Overall.Stage"] != "II")
        ]

    new_shape = lung1_shape[
        (lung1_shape["Overall.Stage"] != "I") & (lung1_shape["Overall.Stage"] != "II")
        ]

    new_glrlm = lung1_glrlm[
        (lung1_glrlm["Overall.Stage"] != "I") & (lung1_glrlm["Overall.Stage"] != "II")
        ]

    new_hlh = lung1_hlh[
        (lung1_hlh["Overall.Stage"] != "I") & (lung1_hlh["Overall.Stage"] != "II")
        ]

    return new_firstorder, new_shape, new_glrlm, new_hlh


def feature_correlation(featurename_x, featurename_y, log=False):

    try:
        lung1_feat_x = lung1_firstorder[featurename_x]
        huh_feat_x = huh_firstorder[featurename_x]
    except KeyError:
        pass
    try:
        lung1_feat_y = lung1_firstorder[featurename_y]
        huh_feat_y = huh_firstorder[featurename_y]
    except KeyError:
        pass
    try:
        lung1_feat_x = lung1_shape[featurename_x]
        huh_feat_x = huh_shape[featurename_x]
    except KeyError:
        pass
    try:
        lung1_feat_y = lung1_shape[featurename_y]
        huh_feat_y = huh_shape[featurename_y]
```

```
    except KeyError:
        pass
    try:
        lung1_feat_x = lung1_glrlm[featurename_x]
        huh_feat_x = huh_glrlm[featurename_x]
    except KeyError:
        pass
    try:
        lung1_feat_y = lung1_glrlm[featurename_y]
        huh_feat_y = huh_glrlm[featurename_y]
    except KeyError:
        pass
    try:
        lung1_feat_x = lung1_hlh[featurename_x]
        huh_feat_x = huh_hlh[featurename_x]
    except KeyError:
        pass
    try:
        lung1_feat_y = lung1_hlh[featurename_y]
        huh_feat_y = huh_hlh[featurename_y]
    except KeyError:
        pass

    lung1corr = pearsonr(lung1_feat_x, lung1_feat_y)

    fig, ax = plt.subplots()

    if log:
        plt.xscale("log")
        plt.yscale("log")

    ax.scatter(lung1_feat_x, lung1_feat_y, edgecolors="black", s=70, label="Lung1")
    ax.scatter(huh_feat_x, huh_feat_y, edgecolors="black", s=70, color="orange", label="HUH")
    plt.legend()
    plt.xlabel(f"{featurename_x}")
    plt.ylabel(featurename_y)
    fig.set_figwidth(10)
    fig.set_figheight(6)
    plt.title(f"Lung1 Pearson correlation coefficient = {lung1corr[0]}, p = {lung1corr[1].round(4)
    }\n")


def signaturefeature_correlation(featurename, log=False):

    try:
        lung1_feature = lung1_firstorder[featurename]
        huh_feature = huh_firstorder[featurename]
    except KeyError:
        pass
    try:
        lung1_feature = lung1_shape[featurename]
        huh_feature = huh_shape[featurename]
    except KeyError:
        pass
    try:
        lung1_feature = lung1_glrlm[featurename]
        huh_feature = huh_glrlm[featurename]
    except KeyError:
        pass
    try:
        lung1_feature = lung1_hlh[featurename]
        huh_feature = huh_hlh[featurename]
    except KeyError:
        pass

    lung1_sig, huh_sig = plot_signature_km()
    lung1_sig = lung1_sig[0]
    huh_sig = huh_sig[0]

    lung1corr = pearsonr(lung1_sig, lung1_feature)
```

```
549
550     fig, ax = plt.subplots()
551     if log:
552         plt.xscale("log")
553         plt.yscale("log")
554
555     ax.scatter(lung1_feature, lung1_sig, edgecolors="black", s=70, label="Lung1")
556     ax.scatter(huh_feature, huh_sig, edgecolors="black", s=70, color="orange", label="HUH")
557     plt.xlabel(f"{featurename}")
558     plt.ylabel("Combined signature")
559     fig.set_figwidth(10)
560     fig.set_figheight(6)
561     plt.title(f"Lung1 Pearson correlation coefficient = {lung1corr[0]}, p = {lung1corr[1].round(4)
        }\n")
562
563
564 def compare_energy_signature_km():
565     l1, l2 = plot_km(lung1_firstorder, "Energy", lung1_firstorder["Energy"].median(), "Lung1")
566     _, _, lin1, lin2 = plot_signature_km()
567     plt.clf()
568     ax = plt.gca()
569     ax.set_xlim(0, 1500)
570     plt.xlabel("Survival time (days)")
571     plt.ylabel("Survival probability")
572     ax.plot(lin1.get_data()[0], lin1.get_data()[1], label="Signature > median", lw=3)
573     ax.plot(lin2.get_data()[0], lin2.get_data()[1], label="Signature < median", lw=3)
574     ax.plot(l1.get_data()[0], l1.get_data()[1], label="Energy > median", lw=2, color="blue")
575     ax.plot(l2.get_data()[0], l2.get_data()[1], label="Energy < median", lw=2, color="orange")
576
577     plt.legend()
578     plt.show()
579
580
581 la_lung1_firstorder, la_lung1_shape, la_lung1_glrlm, la_lung1_hlh = separate_LA_lung1()
582
583
584 if __name__ == '__main__':
585     plt.style.use("bmh")
586     #cph, train = signature_cox_model(modeltype="radiomics", mute=True)
587
588     #tree = regressor_selection([lung1_firstorder, lung1_shape, lung1_glrlm, lung1_hlh], regtype="
        tree")
589
590     #test_featuregroup(lung1_hlh, huh_hlh, log=True, tight=True)
591
592     #regressor_selection([lung1_firstorder, lung1_shape, lung1_glrlm, lung1_hlh])
593     #signaturevolume_correlation(log=True)
594     #featurevolume_correlation("Compactness2", "HLH GrayLevelNonUniformity", log=True)
595     #plot_km(lung1_hlh, feature, lung1_hlh[feature].median(), "Lung1")
596
597     #compare_histograms(lung1_firstorder, huh_firstorder, feature)
598     #plt.show()
599     #featurevolume_correlation("GrayLevelNonUniformityNormalized", "VoxelVolume", log=True)
600     #compare_energy_signature_km()
601     feature_correlation("RunLengthNonUniformity", "VoxelVolume", log=True)
602     plt.show()
```

A.4 Slice viewer

Listing A.4: slice_viewer.py

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 # A scroll-wheel controlled sliceviewer that views 3d numpy arrays
5 # taking the 1st dimension of the arrays as the different slices
6 # retrieved from:
7 # https://matplotlib.org/3.3.0/gallery/event_handling/image_slices_viewer.html
```

```
# Adapted to work with rgb images


class IndexTracker:
    def __init__(self, ax, X):
        self.ax = ax
        self.X = X

        if len(np.shape(X)) == 4:
            self.slices, rows, cols, chnls = X.shape
        if len(np.shape(X)) == 3:
            self.slices, rows, cols = X.shape

        self.ind = self.slices//2

        if len(np.shape(X)) == 4:
            self.im = ax.imshow(self.X[self.ind, :, :, :])
        if len(np.shape(X)) == 3:
            self.im = ax.imshow(self.X[self.ind, :, :])

        self.update()

    def on_scroll(self, event):
        if event.button == 'up':
            self.ind = (self.ind + 1) % self.slices
        else:
            self.ind = (self.ind - 1) % self.slices
        self.update()

    def update(self):
        if len(np.shape(self.X)) == 4:
            self.im.set_data(self.X[self.ind, :, :, :])
        if len(np.shape(self.X)) == 3:
            self.im.set_data(self.X[self.ind, :, :])

        self.ax.set_ylabel('slice %s' % self.ind)
        self.im.axes.figure.canvas.draw()
```

B

HUH CT ACQUISITION PROTOCOL

Studieprotokoll: PulmDIBH

Kategori: Fagprosedyrer/Annet	Gyldig fra/til:17.12.2021/17.12.2023
Organisatorisk plassering: HVRHF/Helse Bergen HF/Avdeling for kreftbehandling og medisinsk fysikk	Versjon: 4.00
Godkjenner: Helga Gripsgård	Prosedyre
Dok. ansvarlig: Ove Dalseid	Dok.id: D60879

Pasientforberedelse	• Pasienten skal puste normalt (helst jevnt og rolig uten dype åndedrag), og ligge i ro gjennom alle skan. • Informere pasienten om at vi skal ta flere CT-opptak • Info om iv kontrast hvis rekvirert
Fiksering	• Thoraxfix m gul pute • Madrass, 8 + evt 1 eller 2 klosser under knær • Armer godt sammen, unngå A og B på thoraxfix • VCD skjerm brukes ved DIBH.
Leie	• Rygg

Merknad og utstyr

- 4D belte som festes på pasienten
- Tilpasse pitch 3 nivåer under ant pustesykluser pr minutt, se tabell.
- RGSC-boks på sternum for DIBH

Iv kontrast

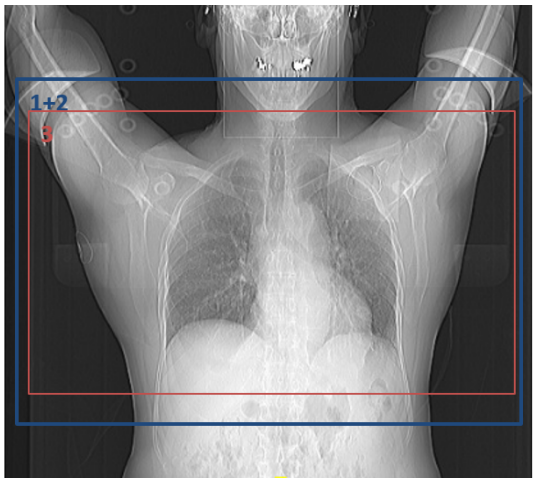
- Hvis ok GFR

Annet

- RGSC boks på sternum
- DIBH: Øve litt først, ha på VCD skjerm, skrive opp A i set up note og kort. Bedre med lavere A og holde pusten hele scannet enn høy A og ikke klare det. Navne pustekurven: datoDIBH. Hvis ikke pas klarer DIBH, skriv dette *Import Images-task* , ta deretter kun 4D og CT thorax.
- 4D: Sjekk kvalitet i PulmoViewer.
- Se neste side →

Scanområde

1. DIBH (3 DIBH ved først fremmøte.1 DIBH ved studie fremøte uke1+uke3) (max 20 s, må ha med hele lungevolumet). Gating-vindu 3mm.
2. 4DCT samme omr som scann 1 (max 120 s, må ha med hele lungevolum), skal brukes til doseplanlegging
3. CT Thorax: Kjevevinkel til binyrer (med iv kontrast)



Ref.pkt sentralt i thorax med perle på SSD.
Se på sidebildet at man dekker lunge i front

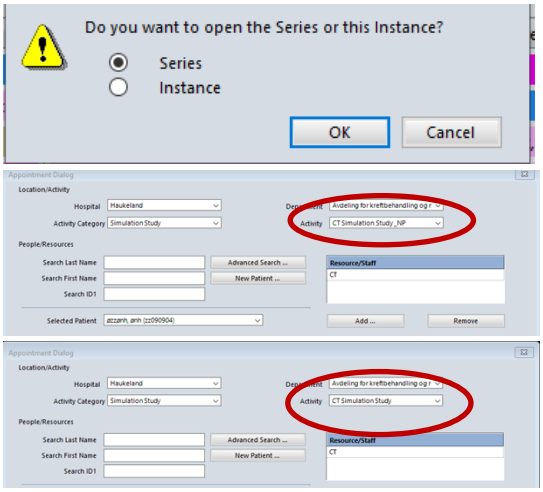
Protokoll: Studie: PulmDIBH / Studie: PulmDIBH + K

Iv kontrast	80 ml Omnip 350 mgI/ml		
Flowrate	3.0 ml/s		
Startdelay	35		
Rekonstruksjoner	Eclipse	DMA	Kryssreferanser
2/2	x	x	CT for doseplan med kontrast Bruk av trykksprøyte ved IV kontrast på gammel CT 4D rekonstruksjon CT2 Pulmonary viewer CT2 TumorLOC CT2
Faser (10)	x		
DIBH	x		
Exam summary		x	

- Disse pasientene skal ta disse bildene på nytt (DIBH, 4D CT og CT thorax uten kontrast, velg Studie: PulmDIBH på CT konsollen) i uke 1 + uke 3 av behandling. På DIBH må amplitude må være lik som på CT for doseplan, så det lages ny amplitudekurve som navnes med datoDIBH. A fra CT for doseplan står i strålekortet.

- For pasienter som er inkludert av lege:

  Endre aktivitetskode i Appointment scheduling



- Åpne i appointment scheduling. Velg den CT timen som er satt opp i behandlingsuke 1. Velg Series.
- Endre Activity fra CTSimulation Study_NP til CT simulation study.

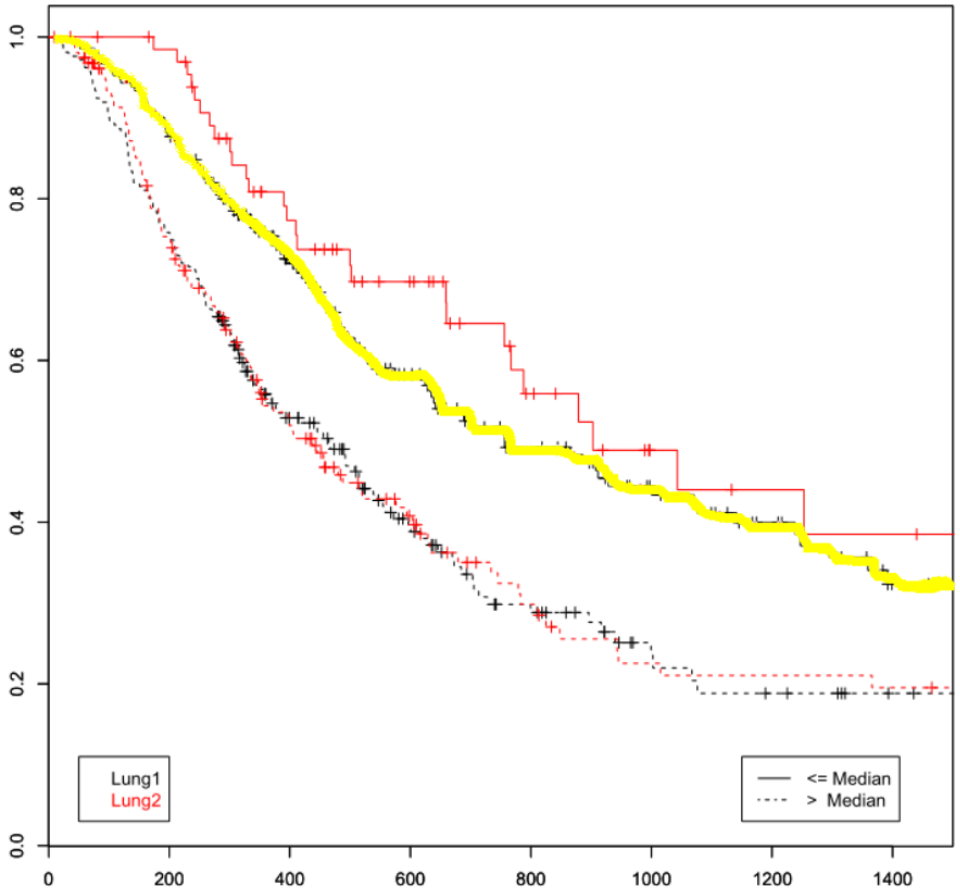
- Dersom pasienten ønsker å tenke på om de skal være med i studie, skal vi be om å få ta CT opptak som om de er inkludert og velge Studie:PulmDIBH/+K på CT. Skriv i så fall i oppstartstimen at behandlende personell etterspør samtykkeskjema. Ved oppstart av strålebehandling skal stråleterapeutene etterspørre samtykkeskjema. Ved samtykke ved oppstart må stråleterapeut endre aktivitetskoden som beskrevet over.
-
- Ved fremmøte uke 1 + uke 3 må en legge inn wegx kode 99 på CTtimen.

C

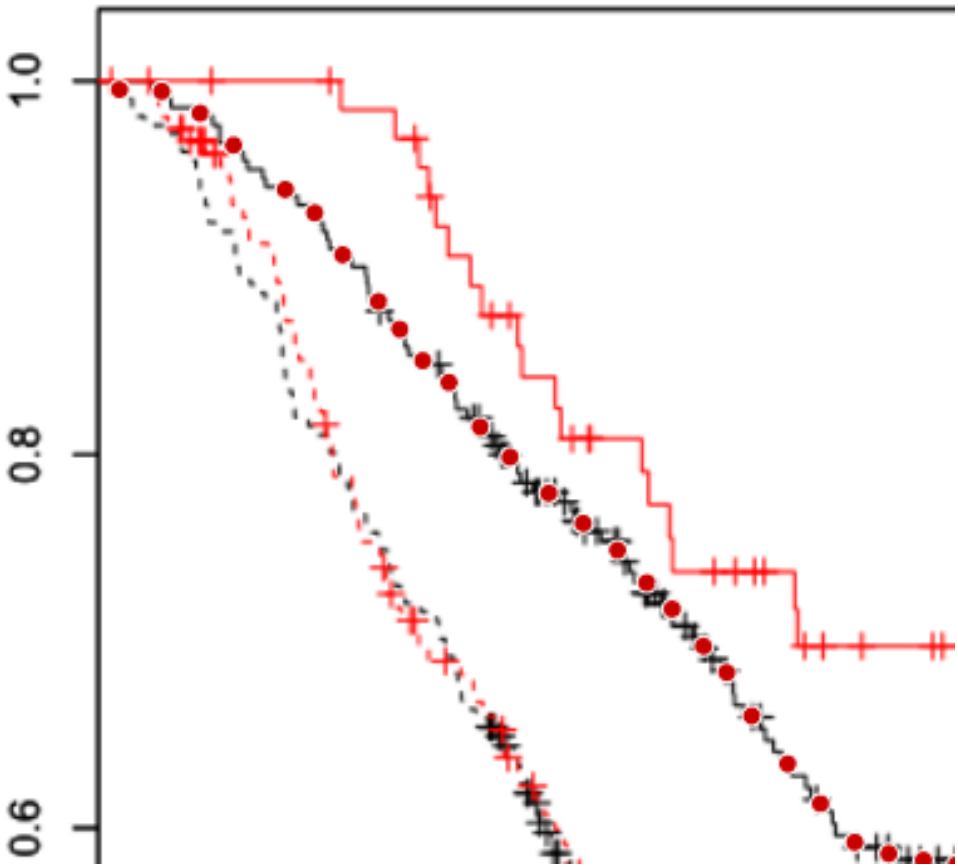
Supplementary figures

C.1 Extraction of curve coordinates from images

Below are images illustrating the process of using *WebPlotDigitizer* in extracting coordinate points from images of curves. The plot used in these example images is the Kaplan-Meier curve for energy from Aerts et al., Supplementary First, two points are marked on both the x- and y-axes to be used as a reference for the point coordinates. Figure 1. Figure C.1a shows how the pen tool is used to mark the curve from which we wish to extract coordinates, and Figure C.1b shows how the points align with the curve in the image.



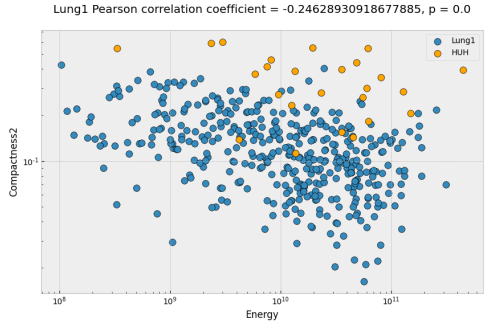
(a) The pen tool is used to mark the curve from which we want to retrieve coordinates.



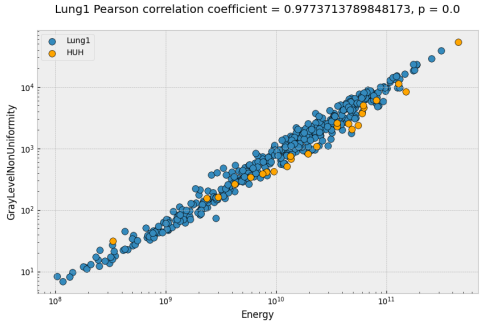
(b) The red points along the black curve are those resulting from running the tool after the marking done in Figure a.

Figure C.1: Images exemplifying the use of *WebPlotDigitizer*.

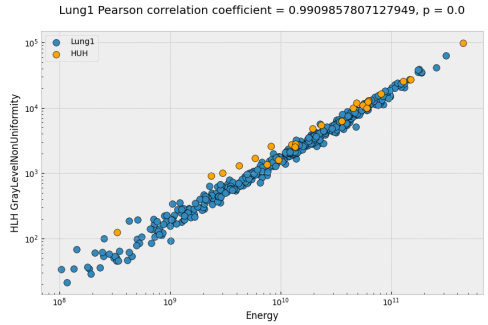
C.2 Correlation between features in the signature



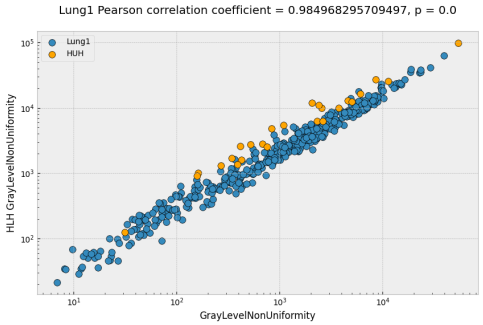
(a) Energy - Compactness



(b) Energy - GLNU



(c) Energy - HLH GLNU



(d) GLNU - HLH GLNU

Figure C.2: Correlation plots of some combinations of features within the radiomic signature.

C.3 Feature histograms

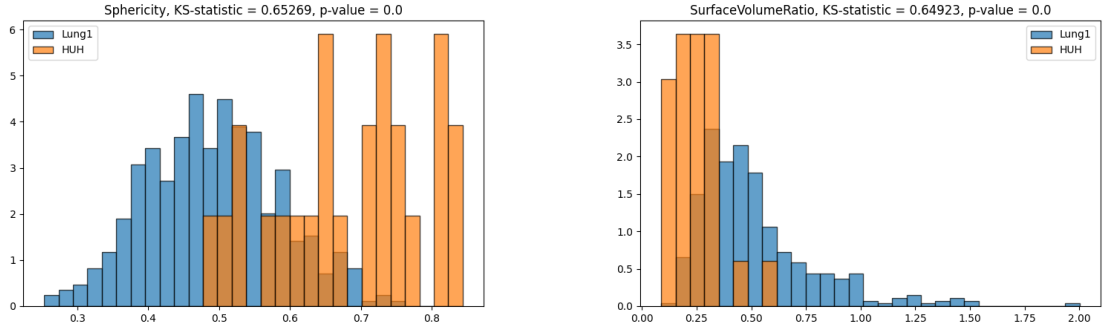


Figure C.3: Histograms of calculated features which are correlated with Compactness2.

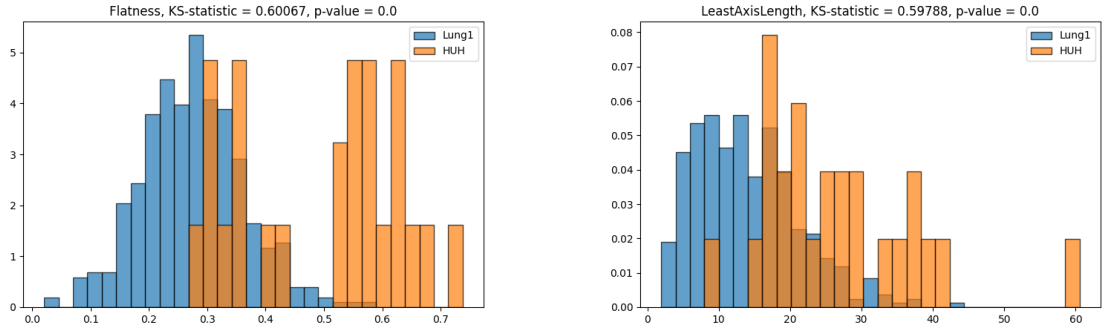


Figure C.4: Other shape features which have different distributions between the two cohorts.

C.4 Kolmogorov-Smirnov test results

Below are plots of the test results from the Kolmogorov-Smirnov test for all features within each feature group. The x-axis is in logarithmic scale.

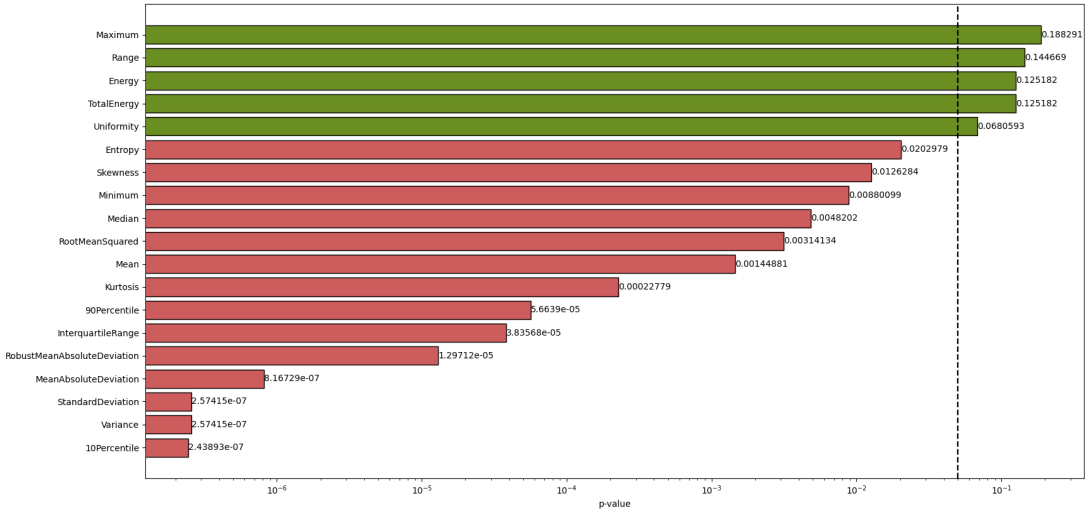


Figure C.5: First-order KS-results

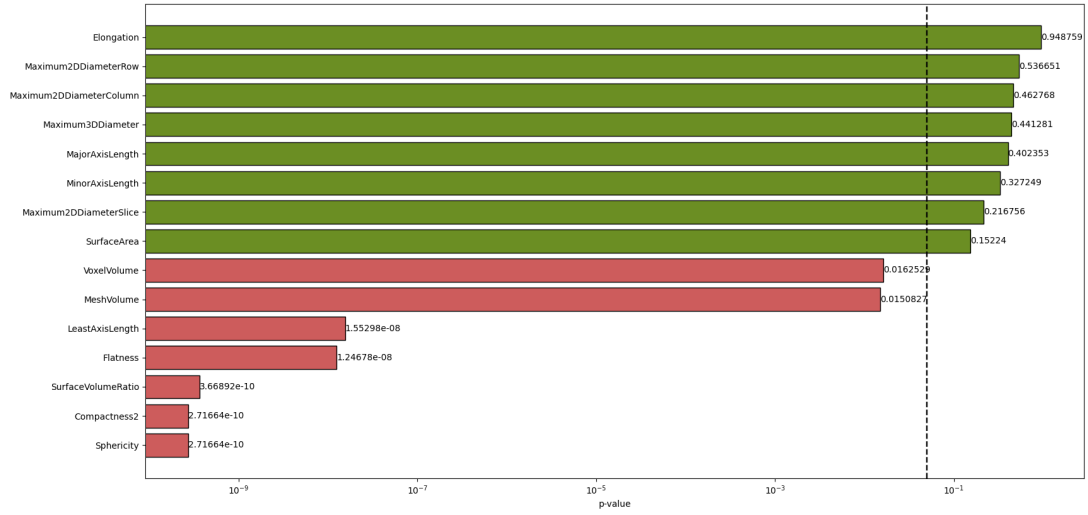


Figure C.6: Shape KS-results

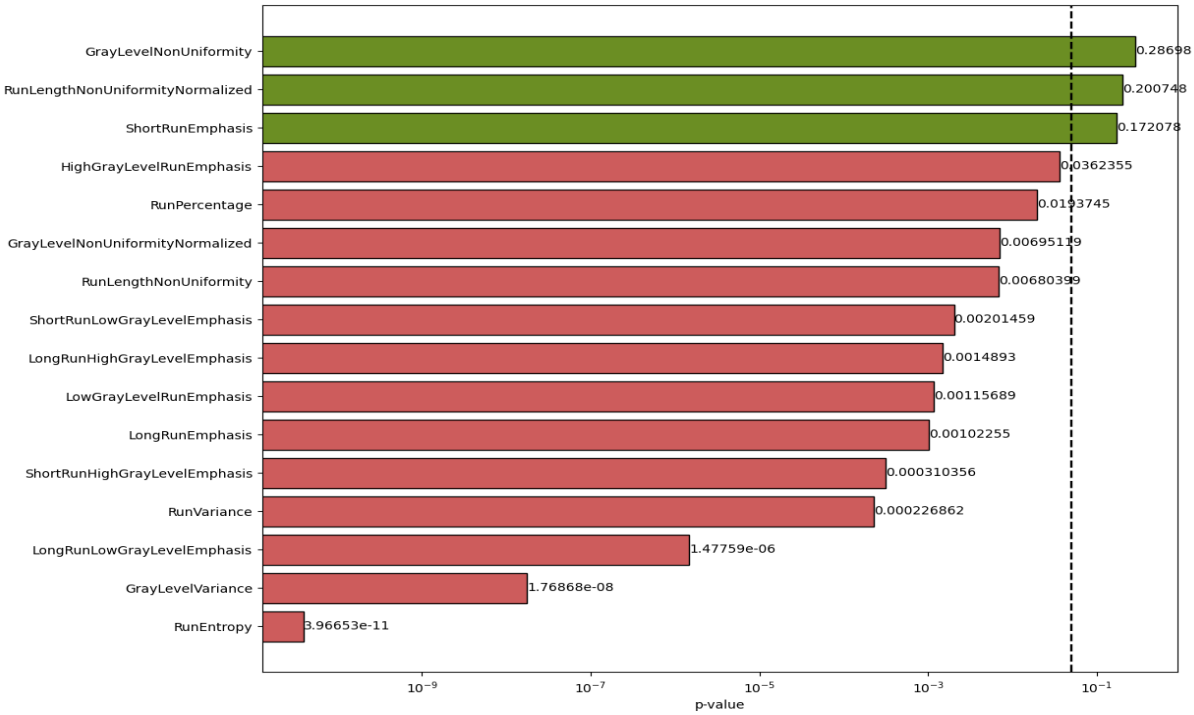


Figure C.7: Grey-level run-length matrix (GLRLM) KS-results

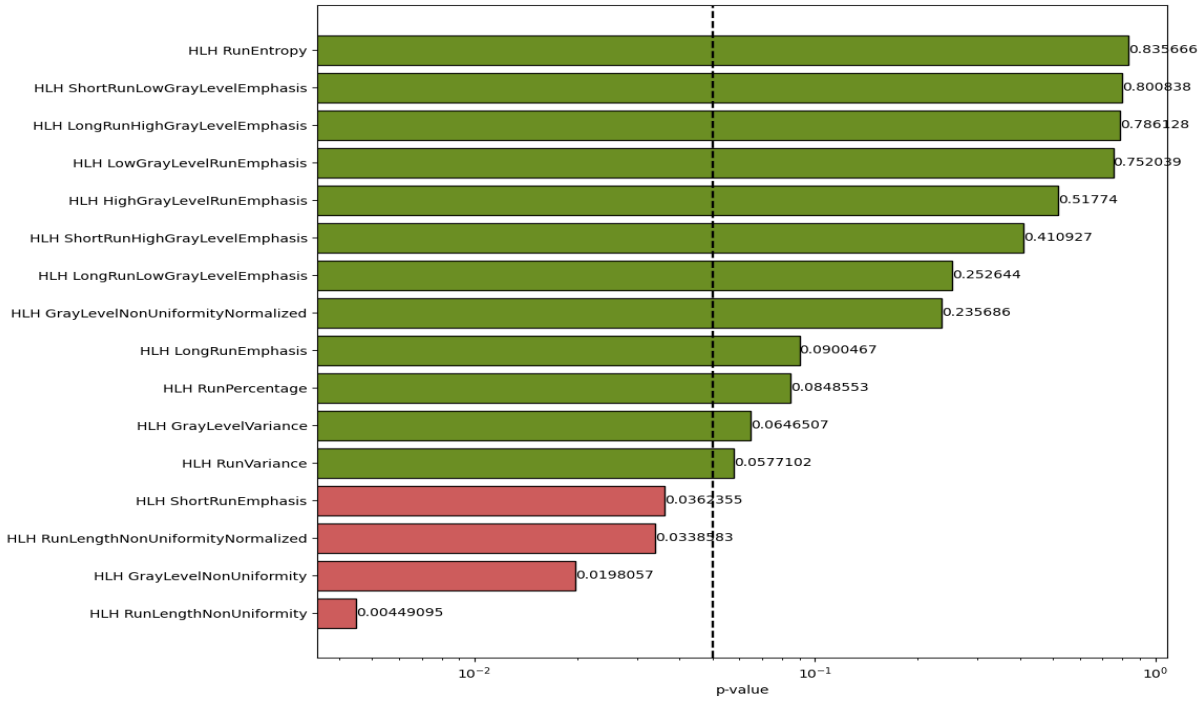
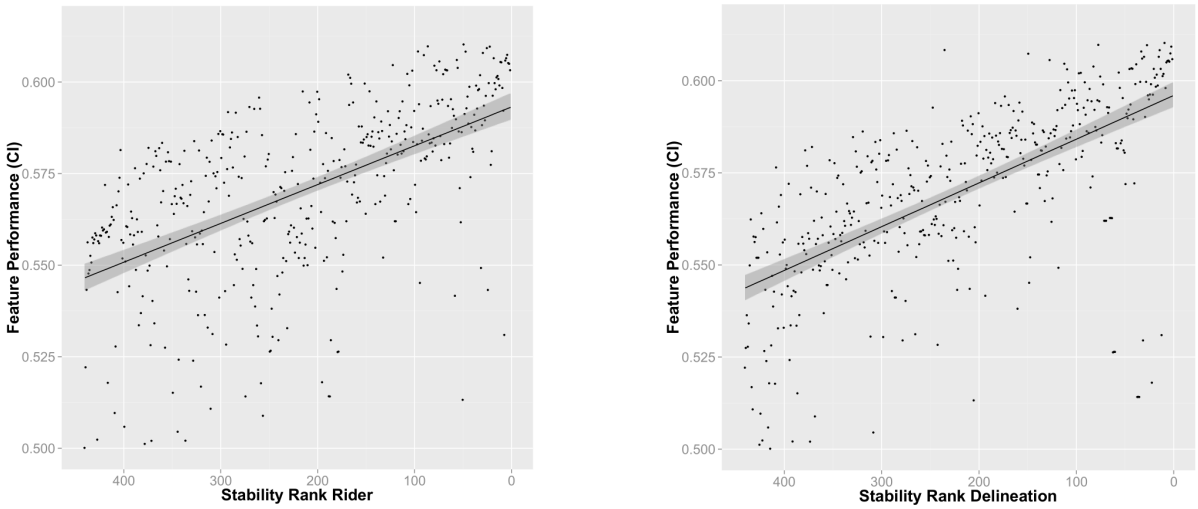


Figure C.8: HLH GLRLM KS-results

C.5 Feature stability plots

Below are the plots of feature stability ranking presented in Aerts et al., 2014. Features are ranked from least stable (rank 440) to the most stable (rank 1).



(a) Inter-observer delineation

(b) Test-retest

Figure C.9: Stability rank of radiomic features plotted against prognostic performance (Concordance index). Stability rank of 1 is the most stable feature. From Aerts et al., 2014.

C.6 Signature- and Energy KM-curves

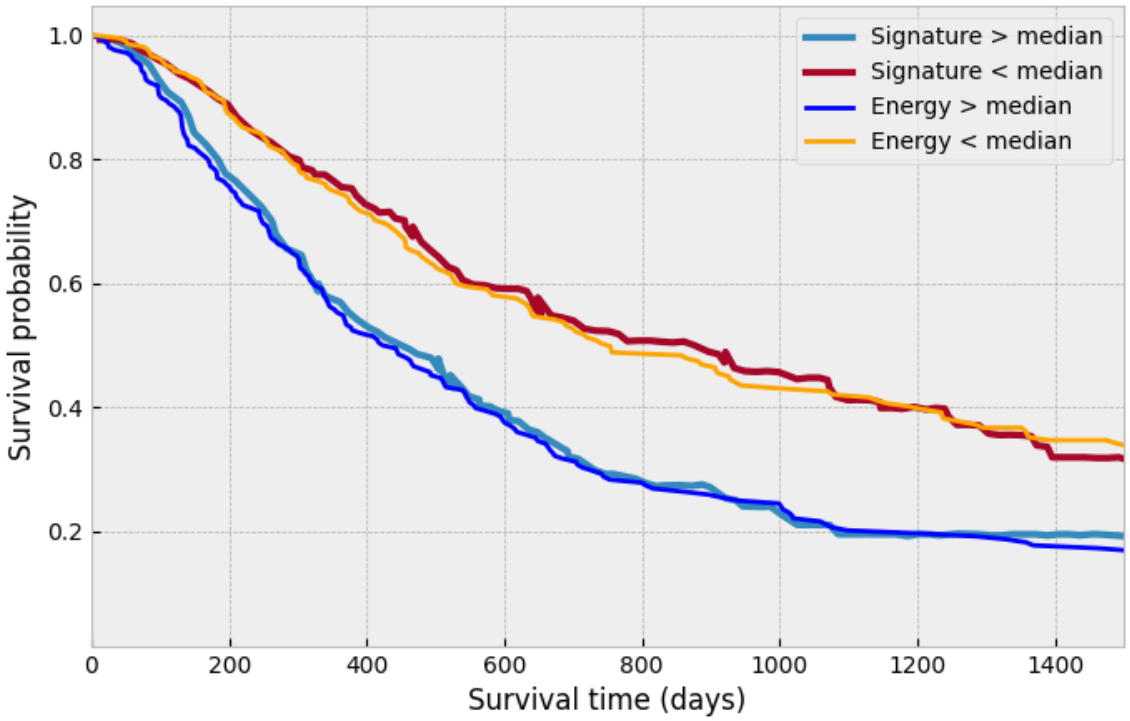


Figure C.10: Kaplan-Meier curves of Energy and the combined signature on the Lung1 set

D

Supplementary tables

D.1 Validation results of radiomic signature

Table D.1: Performance of the radiomic signature model on the three validation sets after being fitted on Lung1. Reprinted from Aerts et al., 2014.

Dataset	TNM	Volume	Radiomics	TNM-radiomics	Volume-radiomics
Lung2	0.60	0.63	0.65	0.64	0.65
H&N1	0.69	0.68	0.69	0.70	0.69
H&N2	0.66	0.65	0.69	0.69	0.68