FH Salzburg

BACHELOR THESIS

On the Predictability of Remission in Colorectal Tumor Patients using Radiomics

submitted to the course of
Information Technology and Systems Management
Salzburg University of Applied Sciences

submitted by
Valentin Schweitzer

Head of school: FH-Prof. DI Dr. Gerhard Jöchtl
Supervisor: FH-Prof. Dr. Michael Gadermayr

Puch, on 2022-01-06

Declaration on oath

I hereby declare on oath that I have written this bachelor's thesis on my own and without external assistance, and have used no other source materials or aids than those listed. Furthermore, I hereby affirm that I have identified any text or content taken from the source materials used accordingly.

To date, this paper has neither been submitted to any other examination board, in this country or abroad, nor published in an identical or similar form.

2022-01-06
Date


Signature

Acknowledgements

I would like to thank everyone who so patiently helped me in writing this thesis.

This thesis would not have been possible without those that provided open access to their work and knowledge.

Abstract

Purpose To determine the predictability of pCR in colorectal tumor patients using statistical properties of associated MRI imaging data. Through the usage of a standardized radiomics library, beneficial radiomics features for a better-than-random classification performance are determined. The implementation of this setup is attempted through the sole use of widely available or replicable tools.

Experimental Setup The dataset featuring inconsistent tumor annotations was processed by a common semi-automatic annotation algorithm. An initial count of 1743 radiomics features was extracted using PyRadiomics. Each feature was then assigned a weighted "importance" score, depending on its contribution to a successful classification. Finally, the ideal importance threshold was determined.

Additionally, the possibility of favorable feature combinations was explored by creating groupings of related features. Models based on those groupings were evaluated by comparing their actual performance against their expected performance. Feature combinations performing unexpectedly well were marked as potentially beneficial in combination.

Model performance was evaluated using a combination of balanced accuracy and ROC curve metrics. Measures were taken over the average of 100 classifier models for any set of parameters.

Results A classifier model reaching a balanced accuracy of 0.697 (95.0 % Confidence Interval [0.683, 0.712]) and an AUC mean of 0.766 (95.0 % Confidence Interval: [0.753, 0.780]) has been constructed.

Favorable groupings for increased model accuracy have been found.

Conclusion In concordance with similar studies, the prediction of pCR in colorectal tumor patients based on radiomics features seems possible.

Studies utilizing a different classification model for comparable challenges consistently performed better than the RF-based classifier used here. Along with a strong difference in the amount of selected features, this result possibly calls the effectiveness of the feature selection algorithm introduced in this thesis into question.

Although feature groupings that perform unusually well together have been found as expected, low correlation between expected and achieved grouping performance points to flaws in either the evaluation of groupings or the earlier mentioned weighting of feature importances.

Contents

1 Introduction

This section gives an overview about the reasoning behind the creation of this thesis, along with a comparison of similar studies. For a practically oriented overview of this thesis, refer to Figure 1.

1.1 Motivation

Cancer is one of the leading causes of death in adults worldwide. Estimated to be the third most diagnosed type of cancer in 2020, colorectal cancer is both common and carries a high mortality rate [1]. The response to treatment varies between patients, where complete recovery is known as Pathological Complete Remission (pCR) [2]. The prediction of pCR aides in the choice of treatment of patients suffering from tumors [3]. Although this prediction has been the aim multiple studies, the methodology used is often difficult to reproduce, through the use of either undocumented or non-standard implementations.

1.2 Aim of this Thesis

The aim of this thesis is to investigate a connection between statistical features of common Magnetic Resonance Imaging (MRI) images and the pCR of a patient. A radiomics-based machine learning model in the shape of a Random Forest (RF)-based classifier, will be used in an attempt to predict pCR in colorectal tumor patients significantly better than trough random choice. To improve reproducibility, publicly accessible and well-document tools will be used. Where applicable, the idea behind custom implementations and their mode of operation shall be well-described, in order to be easily replicated.

1.3 State of Research

Numerous papers have been written about the analysis of tumors using radiomics, though the focus on colorectal tumors is comparatively sparse. As of time of writing, three other papers are known to focus on the prediction of pCR via radiomics features extracted from MRI scans of rectal tumors [3–5]. A comparison of methodology and results of these studies is shown in Table 1. All three studies were successful in separating pCR and non-pCR patients at a rate better than through simple guesses. Two of these studies employed either an RF-based model, or one including an RF classifier.

Study	Patients in Dataset	Classification Model	AUC	95.0 % Confidence Interval
[3]	222	SVM	0.976	[0.919, 0.971]
[4]	134	Mixed, including SVM and RF	0.910	[0.830, 0.980]
[5]	104	RF	0.712	—

Table 1: A comparison of studies on the prediction of pCR using MRI-based radiomics features. Where missing, data was not available from the original study.

Although all datasets consisted of MRI imaging data, significant differences in scan composition were shown. All studies included T_2 weighted scans, with [3] using additional Diffusion Weighted Imaging (DWI) data, where each weighting was recorded at two distinct points in time. [4] states access to additional T_1 weighted scans, though it focuses on the analysis of T_2 data.

Additionally, selection of Region of Interest (ROI) varied in methodology. While all studies used manual annotations, [5] utilized a combined approach of using both the largest available tumor slice and a volumetric representation of the entire tumor, while [4] and [5] opted for the former and latter approaches respectively.

Although automated and semi-automatic tumor annotation techniques are in development, all studies listed in Table 1 opted for manual annotation. As non-manual annotation technologies are, as of time of writing, not clearly standardized, a simple "a priori" shape-based algorithm has been used to semi-automatically create 3-dimensional annotations in the context of this thesis [6–8].

For the extraction of radiomics features, both [3] and [5] opted for differing custom implementations using MATLAB[1], where in both cases a description, of varying detail, of the features utilized, was published. In contrast, [4] chose MaZda [9], an image texture analysis toolkit. This shows the common intent of a transparent feature definition. Though this helps in increasing reproducibility [10, 11], two out of three studies used MATLAB for feature extraction, while [4] later relies on MATLAB for classification. This may hinder availability of the methods described in these studies, as MATLAB is, as of time of writing, commercial software [12]. Additionally, even well-documented custom implementations of radiomics features pose problems for the comparison between studies, as addressed by [10]. To alleviate problems associated with both inconsistent feature definitions and the potential unavailability of commercial software, this thesis is based on an open-source radiomics library, which largely follows Image Biomarker Standardisation Initiative (IBSI) recommendations for feature definitions.

Although using different sets of features and feature definitions, all studies shown use at least one common method of selecting features (Least Absolute Shrinkage and Selection Operator (LASSO) [13]). Additional commonality arises from the features selected, as feature groups chosen by each study consist largely of texture-based features, such as Co-occurrence of Local Anisotropic Gradient Orientations (CoLlAGe) and Gray-Level Matrices [3–5, 14].

As the only known study to utilize an identical classifier setup (a purely RF-based classifier) for the problem posed in this thesis (predicting pCR in colorectal tumor patients), [5] employs a combination of three feature selection variants. In contrast to these measures, a selection method based on the ability of RF classifiers to determine the contribution of individual features to the final classification performance is used. The choice of an RF classifier was based on the strength of such models to handle a large amount of features extracted from a relatively small sample size [15, 16]. Indeed, the number of patients in this study is consistently lower than all similar studies referenced here (compare Section 3.1 and Table 1).

[1] The MathWorks, Inc.

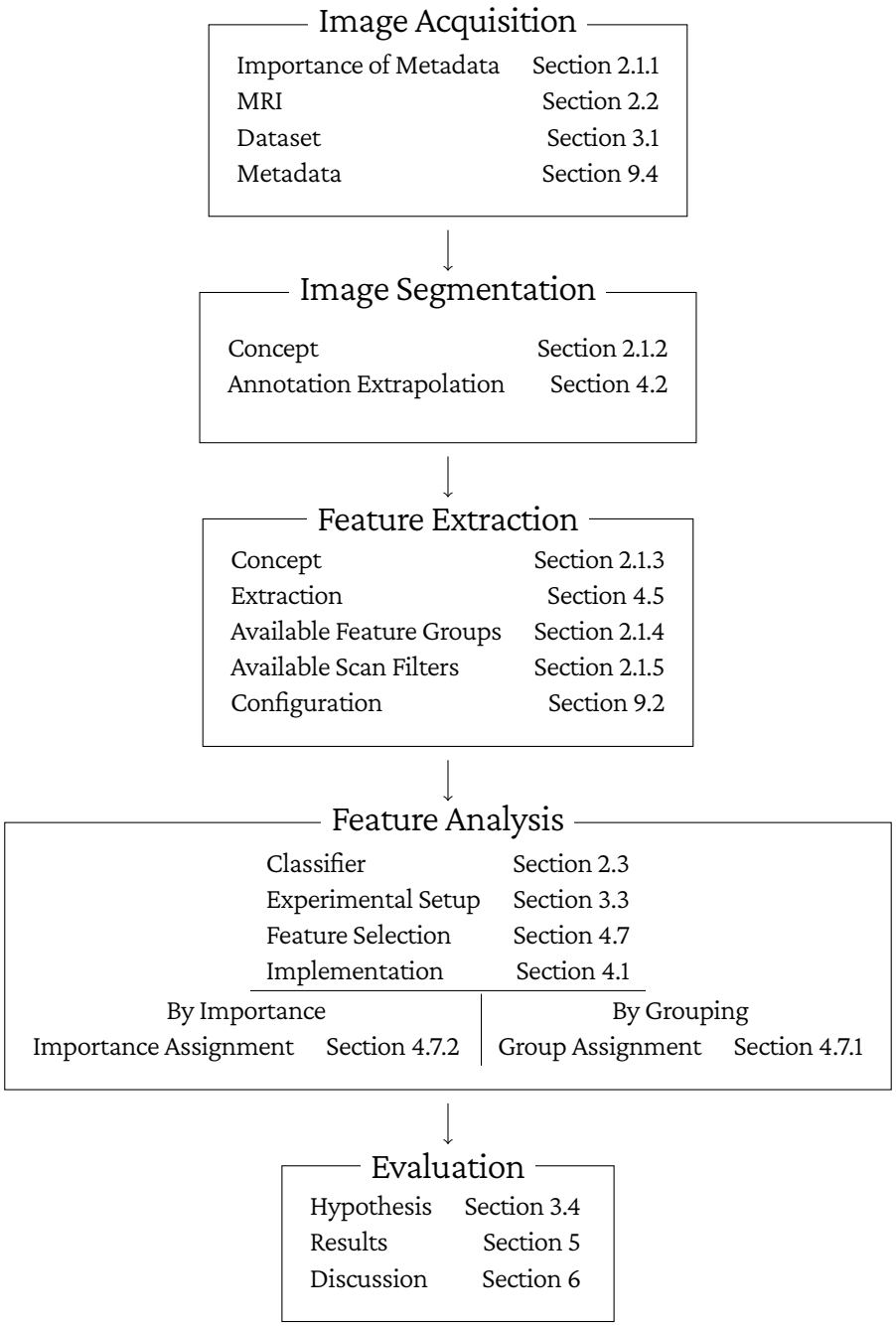


Figure 1: A rough overview of the sections that make up the central experiment this thesis covers. A summary of this setup is provided in Section 3.3.

2 Underlying Concepts

This section gives an overview of concepts that form the base of this experimental setup. First, radiomics is described in more detail in Section 2.1, along with the imaging technology whose results it analyzes, which is examined in Section 2.2. Second, the classification model used here and its evaluation are covered in Section 2.3.

2.1 Radiomics

Radiomics is an emerging field of study concerning the analysis of medical imaging data [17]. It aims to use the products of widely used and available imaging technologies to extract additional information about a patient's health through statistical analysis. It does so by the extraction of *features*, discrete pieces of information describing an image's content, which later can be fed into a classifier model. Such features describe properties such as the shape of a tumor inside a scan or the distributions of gray levels inside a specific area [11, 18, 19].

For using radiomics as a base for classification, four steps are required; acquiring image data to be analyzed, marking an ROI within those images, extracting radiomics features from an image and its newly generated ROI, and analyzing potential interrelationships between the extracted features and expectations in a patient's development [11, 20]. While these steps cover a rough outline of the entire radiomics workflow, some authors may choose to further divide these steps into more detailed subroutines, especially when focusing on one or more steps in particular (compare [10]).

2.1.1 Image Acquisition

For images to be analyzed, these images must be present in the first place. This can be accomplished any common or available medical imaging technology. The technology in focus here is MRI, the process of which is described in more detail in Section 2.2.

Though the focus of radiomics lies on the analysis of images themselves, metadata is not to be ignored. Imaging technologies generally are customizable. This means, the resulting image is a product of not only the patient scanned, but also of the scanner's configuration. This can result in images displaying differing characteristics taken of the same patient, even if they would be taken simultaneously. Of course, patients can also be treated differently before their scan, such as through the use of a contrast medium, which in turn changes the resulting image. Such unknown variables and the inconsistencies they cause should be avoided both in the initial dataset used to build and test a model, as well as in any new scans to be analyzed by this model. Any difference in the methodology of creating an image may cause a change in features, which in turn may adversely affect the classification of a scan.

For this reason, both the status of the patient to be scanned, and the scanner's configuration should be published, if possible [21]. This can be especially helpful if the original dataset of a classifier's model can not be published, as metadata may potentially be made public without harming patients' anonymity or breaking confidentiality.

2.1.2 Image Segmentation

In medical images, some sections are generally of more interest than others. A section of an image containing a tumor may hold more information about the wellbeing and probability of survival of a patient than its surroundings. For this reason, an area can be marked to be a Region of Interest (ROI), to be analyzed in more detail.

The level of detail in a segmentation is a trade-off. One of two common possibilities can be chosen; 2-dimensional and 3-dimensional annotation. A 3-dimensional annotation can enclose the entire tumor, yielding features that can better describe its overall structure. This type of annotation is comparatively slow though. When done manually, a tumor must be marked in every slice it crosses. In contrast, a 2-dimensional annotation can be kept to a single slice, saving time, but possibly losing information [6, 22].

The segmentation can be achieved by one of three methods: manual segmentation, semi-automatic segmentation and automatic segmentation. Manual segmentation is often considered the "ground truth", as it is usually done by medical professionals [6]. Though accurate, this method is very time-consuming. For this reason, semi-automatic processes aim to assist humans workers, lightening the load on medical staff or sometimes enabling a person with less or no medical training to perform a segmentation[6]. Lastly, removing the human factor entirely, automatic segmentation algorithms aim to require no intervention at all. Attempts at such processes include region-growing, from a manually set origin [6], expansion of a 2-dimensional annotation based on prior knowledge about organ shape [7] and the creation of a convolutional neural network, either fully autonomous [8] or based on available manual delineations [23].

Automatic and semi-automatic segmentation algorithms would greatly reduce work for medical professionals, while improving reproducibility by removing unquantifiable "human factors". As of time of writing though, while showing promising results, these technologies are still in development, having not reached a common standard and, in some cases, requiring human intervention, even if automatic[6, 8].

2.1.3 Feature Extraction

The scans that radiomics features are extracted from can largely be treated similar to conventional images. Although they may have some properties that appear unusual, such as uneven axis scaling and a third dimension, these irregularities can be corrected or accounted for. As an MRI scan can be represented in a matrix, all conventional matrix operations, such as image manipulation algorithms can be applied.

Features can be extracted by a combination of the MRI scan and its annotation and, in some cases, from the annotation itself. The detailed extraction of features varies between implementations. PyRadiomics, the library used here, largely follows the methods recommended by the IBSI [10, 24].

Some scan properties may not hold any information about the patient. Such properties, such as voxel dimensions, may instead be caused by other circumstances, such as scanner configuration, and may vary between scans. These differences may complicate the comparison between scans or may even be picked up as a "feature" itself, introducing unwanted biases. The process removing such inconsistencies between scans is known as *normalization* [25].

2.1.4 Radiomics Features

The information that is extracted from medical images in the process of radiomics is made up of *features*; a set of variables, each describing a discrete property of an image's content. These features are split into groups, where each group marks features dependent on similar image properties, such as the shape of the ROI or the values of pixels inside it. Here, an overview is given of feature groups, as defined in PyRadiomics.

First Order Statistics This category consists of features describing "[...] the distribution of voxel intensities within the image region defined by the mask [...]" [25, First Order Features], i.e. the differences in gray-levels among voxels inside the ROI. These features correspond largely to those defined in IBSI's "Intensity-based sta-

tistical features" [26, UHIW] and "Intensity histogram features" [26, ZVCW]. Though PyRadiomics both slightly changes the definition of features and adds additional custom features, these changes are noted in the documentation [25, First Order Features].

Shape-based (3D) and Shape-based (2D) These features describe the shape of the ROI. As such these features do not vary with the input scan, but solely by annotation. This means, this category is calculated only once, independent of how many input scan filters are used.

Gray Level Cooccurence Matrix The Gray Level Cooccurence Matrix (GLCM) describes the texture of an image. More accurately, it describes the frequency of two specific values (in this case colors, usually gray values) being present in two pixels in a certain distance from each other. The distance used here is described by the L_∞ *metric* or *Chebyshev distance*. At a distance of 1, this would be pixels directly touching each other with at least one corner. At this distance, the GLCM describes how often two colors, a and b, sit next to each other, for each possible combination of colors. For n possible colors (256 for an 8 bit color-depth, 65536 for a 16 bit color-depth, assuming grayscale), this results in $n \times n$ possible combinations, and therefore, and $n \times n$ sized GLCM. In this configuration, a GLCM is always symmetrical, as the combinations ab and ba appear an equal amount of times, as they are the same combination of colors. A GLCM can also describe directed combinations; two pixels with a certain combination of colors, situated in a specific angle in relation to each other. For each combination of distance and angle, a separate GLCM has to be calculated [25, 27, 28].

Gray Level Run Length Matrix The Gray Level Run Length Matrix (GLRLM) describes continuous strips of pixels sharing the same color along a certain direction. "Continuous" meaning one or more neighboring pixels along the direction in question. For an image with a length of n pixels in a certain direction and m possible gray-values per pixel, the GLRLM for that specific direction has the dimensions of $n \times m$ [25, 27].

Gray Level Size Zone Matrix Similar to the GLRLM, the Gray Level Size Zone Matrix (GLSZM) describes stretches of neighboring pixels of the same value. Where it differs is, that it describes not just straight lines in a single direction, but any area of identical values touching. This creates a matrix of variable size, with one fixed dimension representing possible pixel values and a flexible dimension, determined by the size of the biggest pixel patch. The size of the matrix therefore grows with large stretches of areas with low contrast [25, 27].

Neighboring Gray Tone Difference Matrix The Neighboring Gray Tone Difference Matrix (NGTDM) describes the absolute difference between the value of a single pixel as compared to the average pixel value of its neighborhood. The neighborhood is considered to be every pixel inside a certain distance to the center pixel, but not that pixel itself. The matrix dimensions are formed by the amount of gray levels possible per pixel n and the number of possible differences to a pixel's neighborhood m, resulting in an $n \times m$ matrix [25, 26].

Gray Level Dependence Matrix The Gray Level Dependence Matrix (GLDM), also called the Neighboring Gray Level Dependence Matrix (NGLDM), describes the coarseness of an image [26, REKO]. When given a certain *coarseness level* and a distance, it gives the number of pixels connected to a center pixel within that distance, whose value does not differ by more than the coarseness level from that of the center pixel. These pixels are considered *dependent* on the center pixel. The number of these pixels is defined to be at least one, which can be interpreted as counting the center pixel to be dependent on itself. The dimensions of the matrix are determined by the number of possible gray values per pixel n and the maximum amount of dependent pixels in its neighborhood m, forming an $n \times m$ matrix [25, 26].

Although GLCM, GLRLM, GLSZM and NGTDM were originally designed for use in 2-dimensional image analysis, they have been adapted to work with 3-dimensional images for use in radiomics [26, 27].

2.1.5 Base Scan Filters

The MRI scan used to extract features from can be subjected to a selection of filters. These filters each yield a new image, generated from the base image but representing different properties; each of these forms the base for the extraction of the same set of features. Note, that even though the original scan is not explicitly listed here, it is still used for feature extraction [25].

Wavelet A wavelet filter splits an input image into multiple, smaller images each representing a filtered version of the original. These new images each display a certain frequency band along each direction (horizontally and vertically). A wavelet filter can be applied again to the output of a prior filter, using the image containing the lowest frequencies. The more often a wavelet filter is applied, the lower the frequencies in that image become [29, 30].

Laplacian of Gaussian Laplacian of Gaussian (LoG) is a filter, creating an image that shows the location of fast changes in value of neighboring pixels in the original image, i.e. edges. The scale of these edges, that is, how fast the value of pixels must change to be considered an edge, can be manipulated by choosing a different *Gaussian filter* [25, 31].

Square and SquareRoot These filters each apply a function to the value n of every pixel inside the original image. After calculating the square (n^2) or square root ($\sqrt{n}$) respectively, the result is scaled to fit the original range of gray values [25].

Logarithm and Exponential Similarly to "Square" and "SquareRoot", these filters each apply a mathematical operation to each individual value, scaling it down its original range afterwards. Both operations are applied to the absolute value of each voxel, where "Logarithm" returns the natural logarithm with a fixed value of one added to it, while "Exponential" returns e to the power of the absolute voxel value [25].

Gradient The gradient filter represents the change of value in each voxel based on the values of voxels surrounding it [25, 32–35].

LocalBinaryPattern2D and LocalBinaryPattern3D Local Binary Patterns (LBPs) provide information about pixel differences in a circular neighborhood of a central pixel. By using value differences between pixels instead of absolute values in a rotationally-symmetrical area, they provide information about an image's texture in a rotation and scale invariant way [25, 36].

As with Section 2.1.4, even though some filters were developed for use in 2-dimensional images, they have been adapted to accept 3-dimensional images[25].

2.1.6 Challenges in Radiomics

Datasets As a relatively new field of study, the research about radiomics is ongoing. Although the tools for the analysis of medical images are, by now, widely available, access to a usable dataset requires both access to expensive machinery and a sizable group of volunteers. Due to this, access to datasets for radiomics research are limited.

Furthermore, MRI scans contain a set of variables in their own right. Access to a dataset does not guarantee the possibility of reproducing other studies. If these studies did not disclose information about the creation of their dataset, differing scanner configurations may lead to inherently different scans, severely affecting classification [11].

Segmentation The ROI has a strong impact on the extraction and quality of radiomics features, as many of these features are based on characteristics of the ROI itself and its contents [24, 25]. As such, both the quality and consistency of annotating (i.e. marking an ROI) is of major importance and its improvement poses an ongoing challenge [11, 23]. It would be theoretically possible for all scans in a single study to be annotated by the same medical professional. Although this would keep the annotation somewhat consistent between scans, this can be very time-consuming [11]. [6] mentions the time requirement for manual segmentation varying "[...] from 60 to 1118 seconds ($\pm$18.5 minutes) for the manual delineation of primary tumors." [6, p. 828]. For studies involving more than a hundred patients, such as [23] and [8], annotating all scans would pose a significant challenge to a single person, while employing the help of multiple annotators would introduce additional inconsistency between ROIs. As such, attempts at providing an automatic, or semi-automatic, solution to scan segmentation are being developed, with the aim of lightening the workload of medical professionals, while improving consistency and reproducibility. As mentioned earlier, these technologies are still largely under development [6, 8, 11].

Feature Selection The details of radiomics feature extraction is, as of time of writing, a point of debate. Features simply are statistical properties of a scan and its annotation, an infinite amount of which, in theory, could be extracted. Combined with the ability to change the input scan by applying a variety of filters, the amount of features generated is again multiplied. Still, just because a feature can be extracted does not mean it serves a role in predicting treatment outcome. Therefore, a choice must be made as to which features are relevant to radiomics and should be analyzed further. This choice, so far, is not always consistent between implementations, leading to differing experimental setups between authors.

Feature Definition Even if a common set of radiomics features is chosen, the possibility of inconsistencies persists. Even while sharing a common name, features can be implemented using different methodology, leading to different values extracted from the same scan, using the "same" feature.

One attempt at defining a common feature set and establishing a standardized way of calculating each is the Image Biomarker Standardisation Initiative (IBSI). It aims to unify the way radiomics features are chosen and generated by providing a common standard and dataset to test implementations against [10].

2.2 Magnetic Resonance Imaging

Medical imaging technologies represent an important tool in medicine. They provide a look on, and inside, the human body that, due to the vast variability in technologies and their focus, bear a range of information about a patient's health. Some of these methods, utilizing the properties of magnetic fields, x-rays and γ-rays,

include Magnetic Resonance Imaging (MRI), Computed Tomography (CT) and Positron Emission Tomography (PET) scans. Of these methods, the first two focus on tissue structure, while the latter shows the activity of cellular processes, like the movement and processing of glucose [20, 37, 38]. As a technique showing the structure of organs and associated tumors, featuring "[...] excellent soft-tissue contrast and high spatial resolution (~ 1 mm)."[38, p. 15], MRI is a commonly used tool in the visualization of tumors and is the imaging technology focused on here.

2.2.1 Physical Basics

Like most living creatures, humans consist mostly of atoms. These atoms in part consist of charged, and sometimes additional uncharged, particles. Hydrogen-1 for example has two charged particles, one positively charged proton forming its core and a negatively charged electron orbiting that core. The proton, as with all other protons, has an intrinsic spin. As a spinning[2], charged particle, it generates both a magnetic and electric field around itself. When paired up, the magnetic field of such particles cancel out each other. As for the case of hydrogen, its uneven amount protons cause the nucleus to form a dipole. This makes hydrogen nuclei react to external magnetic fields. If an appropriately strong magnetic field is applied, hydrogen nuclei adopt one of two states: "parallel" and "anti-parallel". The probability of a particle being in either of the two states start out as near equal, while no external magnetic field is applied. As the the "anti-parallel" state requires more energy to maintain, the probability of a nucleus being in the "parallel" state rises with a stronger magnetic field. This relationship is can be described by the Boltzmann distribution, as described in [38, eq. 1.16]. This uneven distribution of parallel and anti-parallel oriented protons "creates" a magnetic field, as the anti-parallel protons can not cancel out the more abundant parallel protons' field. This magnetic field made up of individual protons' fields is referred to as M_0, while the strong external magnetic field causing this formation is called B_0[3]. The current alignment of M_0 is usually described as being along the "z" axis. The total magnetic field along the z axis is referred to as M_z, and currently, M_z makes up the entirety of M_0 [38, 39, 41].

The reason for the focus on hydrogen atoms comes from the human body's relatively high hydrogen content. Though other elements or their isotopes can be visualized using MRI, hydrogen is the most readily available. Not do humans consist of approximately 45.0 % to 60.0 % water by weight, but it is also contained in other materials such as fats and proteins [40–42].

Though M_0, under the sole influence of B_0, is a single magnetic field stretching along a single axis, the protons making up M_0 are not aligned exactly. Instead, the axes they align themselves along, or their magnetic vectors rotate around, M_0's vector at an angle. Although these individual magnetic vectors move, their movement is both at a common frequency and out of phase and therefore add up to a static overall magnetic field. This rotation, or "precession", of individual protons around their net magnetic vector is called the *Larmor precession*, while the speed at which they rotate is described by the *Larmor frequency*. The speed of this procession, and therefore its frequency, varies with the strength of the external magnetic field, as described in Equation 1. The Larmor frequency ω [MHz] is relative to the applied magnetic field's strength B [T] in the ratio of the *gyromagnetic factor* γ [$\frac{MHz}{T}$], a constant determined by the particle of interest.

$$\omega = \gamma B \tag{1}$$

M_0, as a static magnetic field, is difficult to measure. For a sensor, e.g. a coil, to pick up a signal from the protons' magnetic field, the field must be variable, either by changing strength or direction. To change M_0 from its current projection vector along z to a moving magnetic field, the precession of the protons it is made up of must

[2] "Spinning" in the context of particle physics does not imply physical movement, but denotes a particle with spin [39]

[3] Some authors, such as [40], use "O" (upper-case letter "o") instead of "0" (zero) to denote these fields.

be synchronized. These protons, in their state of Larmor precession, can be influenced by an electromagnetic pulse, oscillating at their Larmor frequency. Such a pulse, referred to as B_1 (the second external magnetic field), applied perpendicularly to the z axis creates a second axis around which the protons precess. This movement around a second axis leads the individual protons to sync up their rotation. Additionally, energy is transferred from B_1, rising some into the "anti-parallel" state. This leads M_0 to change; the synchronous precession of protons now creates a moving magnetic field, which, as more and more protons change to the "anti-parallel" state, shifts downward from the z axis. When positioned at a 90° angle, the magnetic vector lies along the x-y axis. This vector is known as M_{xy}. The stronger the electromagnetic pulse B_1 and the longer it is applied, the farther the shift of M_0. This, now moving, magnetic field does induct energy in the surrounding sensors and can be measured [38, 39, 41].

2.2.2 Obtaining Measurements

Though a measurable magnetic field is now available, the external static magnetic fields B_0 and B_1 affect the entire body. In turn, the resulting magnetic field M_0 originates from every affected hydrogen proton, which means only an overall magnetic field can be detected. To differentiate signals from individual sections of the body, the precession of protons in each area must be changed. Along each desired axis in the final MRI image, at least one property must change between protons. This gradual change along a dimension is called a *gradient*.

For 3-dimensional scans, such as those making up the dataset for this thesis, three gradients are required. These gradients are the *frequency encoding*, the *phase encoding* and the *slice selection* gradient.

First, the slice to be analyzed is chosen through the slice selection gradient. A "slice" in this context means a plane stretching the patient's body for which image data will be generated, reaching a certain thickness. This slice can be chosen along any axis, and is determined by the electromagnet creating the external electromagnetic field B_0. The slice selection gradient makes use of the fact, that the strength of B_0 changes with increasing distance from this electromagnet. A change in magnetic field strength in turn means, that the Larmor frequency of protons along the gradient changes as well (see Equation 1). As the creation of a signal relies on influencing protons through impulses matching their resonance frequency, an impulse of a single certain frequency now only excites the protons its corresponding area of the gradient. The broadness of the impulse's bandwidth determines, and rate of change of field strength along the slice selection gradient determine the thickness of the slice.

Perpendicular to the slice selection gradient, the frequency and phase encoding gradients are applied. These gradients affect the frequency and phase of proton's Larmor precession respectively. Through measuring reflected frequencies and the offset of their precessions, these gradients further limit the area a signal is received from. By intersecting all three gradients in a certain point, the resulting signal forms a single voxel. As the thickness of slices created by each gradient are independent of each other, a voxel may not form a perfect cube, leading to different resolutions along different axes [38, 39, 41].

2.2.3 T_1 and T_2

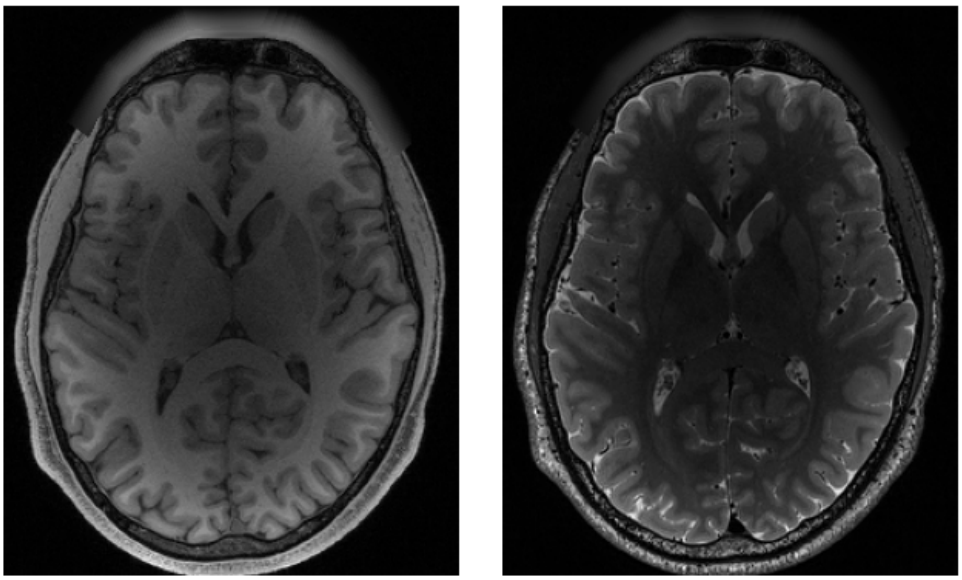
Through gradients, the area to sample a voxel from can now be chosen. This ability is only useful, if contrast can be made out between different voxels. In part, this contrast occurs naturally. The fewer protons are contained inside a voxel, the weaker the signal measured inside this voxel is. This property of tissue is referred to as *Proton Density (PD)*. Additional methods of creating contrast between materials is via their T_1 and T_2 *relaxation times*.

After application of B_1, M_0 is shifted entirely to the x-y plane. When B_1 subsides, protons shift back into their initial state, the state they were in when only B_0 was active.

First, as the resonant magnetic pulse loses influence, the synchronized precession breaks up, or loses *coherence*. As the angled rotating magnetic fields start to spread out in phase, they start to, again, counteract each other's magnetic field along the x-y plane, reducing M_{xy}. This interaction between protons is known as the *spin-spin relaxation* or T_2 *relaxation*. The time it takes M_{xy} to shrink to 37.0 % of its maximum level through a logarithmic fall is referred to as T_2[40].

Second, all protons, including those that changed their state to "anti-parallel" through the absorption of energy from the electromagnetic pulse, will begin dissipate their additional energy into their environment. The loss of energy causes protons to flip back to their initial state. As the distribution of proton states grows uneven again, individual magnetic fields do not cancel each other out anymore and M_z grows. This interaction, the transfer of energy from spinning particles to their environment, also called *lattice*, is known as the *spin-lattice relaxation* or T_1 *relaxation*. T_1 is defined as the time it takes M_z to grow back to 63.0 % of its initial value, following an exponential rise[40].

PD, T_1 and T_2 are intrinsic properties of tissues, not influenced by outside factors. Though they differ between materials, a change in material may affect some of these properties more than others. Therefore, singling out each of these factors can give different information about tissue. Such a difference between the signal strength given off by different types of tissue (which results in different colors between tissues) can be observed in Figure 2.



(a) A T_1 weighted scan.
$\mathrm{TR} = 2400.0$
$\mathrm{TE} = 2.14$

(b) A T_2 weighted scan.
$\mathrm{TR} = 3200.0$
$\mathrm{TE} = 565.0$

Figure 2: Two MRI scans of the same patient weighted differently. Both images have been created at a B_0 of 3.00 T. Pixel values have been scaled to fill the entire available range. Note the difference in tissue color between weightings. These images have been adapted from data provided by the Human Connectome Project [43].

As T_2 usually sets in earlier than T_1, each can be filtered out by a variation in scan parameters; *Echo Time (TE)* and *Repetition Time (TR)*. TE describes the time between the application of B_1 and the recording of a signal, while TR refers to the time in-between pulses of B_1. Due to T_1's and T_2's difference (up to multiple seconds [38, p. 16]) their contribution to the overall signal are affected significantly differently.

$$\text{MR Signal} \propto \text{PD} \quad \cdot \quad \underbrace{\left(1 - e^{-\frac{\text{TR}}{\text{T}_1}}\right)}_{\text{T}_1 \text{ Component}} \quad \cdot \quad \underbrace{e^{-\frac{\text{TE}}{\text{T}_2}}}_{\text{T}_2 \text{ Component}} \tag{2}$$

As shown in Equation. 2 [40, p. 26], the proportion of influence to the signal of T_1 rises with TR, while that of T_2 falls with a longer TE. By removing the other component's influence, an MRI scan can be *T_1 weighted* or *T_2 weighted* respectively [38, 39, 41].

2.2.4 The NIfTI-1.1 Format

A standardized representation of the data acquired through MRI facilitates saving, transmitting and editing that data. One standard covering this use case is the *NIfTI-1.1* file format. Along with the voxel scan data, associated metadata can be preserved using this format. Such metadata includes both metadata describing the image acquisition, e.g. slice thicknesses and relation between voxel and physical position, and added custom information about the preserved image data, such as annotations [44].

2.3 Classification

The discipline of machine learning regularly revolves around the prediction of certain events or properties. Such predictions are made using records of already well-known events, and the factors leading up to them. For example, using historical weather data, a model can make guesses about tomorrow's average temperature. In turn, current temperatures, when compared to that same historical data, can also be used to determine the current season. When trying to continue the list of daily temperatures, a model is trying to continue a stream of continuous values by a continuous value. Although tomorrow's temperature may fall into a certain limited range, the possibilities to choose from are still infinite; this model is performing *regression*. Meanwhile, the model predicting seasons uses its available information to generate a result that is part of a predetermined limited set, such as "Spring", "Summer", "Fall" or "Winter"; this model is performing *classification*. Formally, classification can be interpreted as the mapping from one domain into the label domain; a domain consisting of a subset of $\mathbb{N}$[4]. In this case, classification maps the image domain, in the shape of MRI scans to two labels $\in \mathbb{N}$, such as $\{0, 1\}$, representing either non-pCR or pCR of the patient.

As the prediction of a patient's pCR is based on a classification model, this section is a short overview of its underlying concepts. Classification is the aim of a multitude of models. These include Support Vector Machines (SVMs), which try to separate classes by splitting features along multidimensional planes and decision trees, which use features to assemble multiple interlinked "questions" that sort classes through their answers. Decision trees form the base of the classifier model used here.

2.3.1 Decision Trees

The structure of a decision tree follows that of the widely used data structure; a collection of *nodes*, all connected by *edges* to, directly or through other nodes, a single *root node*, where no edge creates a loop. In a decision tree, a root node represents the first question of that tree. This question can be answered using that node's edges; each edge pointing away from a node contains an answer to the posed question. These edges then lead to additional nodes, the *children* of the node the edge originated from. If this new node has edges pointing away from it, it

[4] Note, that multidimensional classification is possible and not covered by this description. As the classification used here maps onto a single dimension, that use-case is treated here.

contains an additional question. As edges lead in a single defined direction, that is, away from the root node, decision trees are *directed trees*. Should a node be attached only by edges pointing to it, this node is a *leaf node*. When arriving at a leaf node, no further questions are raised. Instead, the content of the leaf node represents the final decision gained by answering all questions leading up to it. For the case of classification this final decision would be the assignment to a class [45].

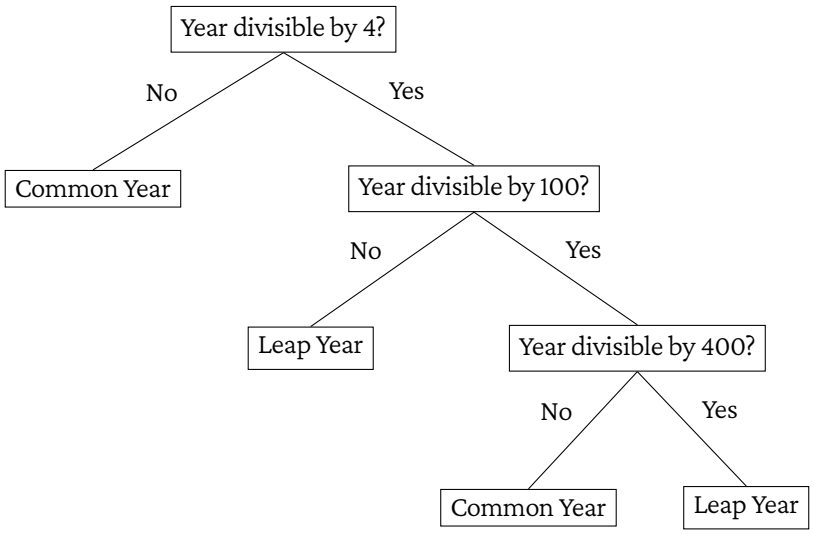


Figure 3: A simple decision tree, determining if a year is a leap year.

Such a decision tree is shown in Figure 3. The input for this tree would be a set of year numbers. Taking one of these numbers, its properties (or features) can be used to answer the question posed by the root node. Following the appropriate edge, and answering any following questions will finally lead to a leaf node. The leaf node reached will contain the classification of the input; the answer whether the year number is a leap year.

2.3.2 Constructing Decision Trees

In some cases, the construction of a decision tree may seem obvious; nodes can be constructed to clearly separate classes and the resulting classification is free of error. But as the size of datasets and the amount and complexity of their features rise, a clearly defined way of constructing decision trees is required. To determine along what quality to split a dataset, that is, through which question to determine the further progression through the tree, every possible new node must be evaluated through a common measure. This measure determines, which split is chosen to achieve the best overall performance for the classification tree. One such measure, the *purity* of a split will be utilized here. Purity describes how well a node, or its question, manages to split the dataset along its classes. If, after one node, no part of the input is lumped together with one of a different class, that node is completely pure. Inversely, the more mixed between classes the output of a node is, the more impure that node is. Pure nodes are generally preferred, as they lead to smaller, more effective trees. Sorting by purity, nodes are connected starting at the root and branching out into branches and leaves. The tree stops growing when either perfect purity is achieved, i.e. the entire set is correctly classified, no nodes are left or a predetermined maximum tree depth is reached [46–48].

2.3.3 Random Forests

A decision tree can in theory be used to classify any amount of data utilizing a feature vector of arbitrary length. Should a tree not achieve full purity, there will always be an error in classification. Inversely, if a decision tree achieves a perfect classification rate, this may mean it is overfitted on its training set. This means, as that model fits its training set, and only its training set, perfectly, it manages classification on any additional datasets considerably worse than non-overfitted models. Additionally, individual trees tend to be unstable; a change in a single node affects all of its children. Slight changes in the dataset, such as random noise, can strongly affect the construction and outcomes of a tree [46, 49].

To avoid this, a collection of trees, sometimes called a forest, can be used. By taking random samples of the training set per tree, selecting a small random subset of available features at each node and using the best (most "pure") parameter to split on, a group of decision trees are constructed. When classifying a member of the test set, that member passes through every tree, generating a prediction from each. Using those individual classifications, the member is then assigned to the class that received the most "votes", i.e. the most chosen class by individual trees. This ensemble of randomly generated trees forming a forest is known as a Random Forest (RF) classifier [15, 16].

The accuracy of a tree are measured during its construction by splitting off a portion of all available cases. This portion is then used to test already constructed trees. Additionally, this enables the determination of individual features' contribution to the final result. First, the reserved set is classified using the constructed forest, counting correct votes for each case. Then, a single variable is swapped randomly between cases and they are classified again. If performance suffers after swapping the variable, this variable contributes strongly to the forest's performance, making it more important [15, 16].

2.3.4 Evaluation

RFs, as with classifier models in general, can be evaluated using a set of metrics. The two methods explored here are the *confusion matrix* and the *Receiver Operating Characteristic (ROC) curve*, using hold-out evaluation. Note, that although the evaluation of a model does not by itself have any influence on the performance of a model, it has a strong influence on the perceived performance of a model. Depending on evaluation method, a model can be perceived to perform better or worse [50]. The methods treated here were chosen because of their relatively common use in similar studies [3, 5].

In the context of this thesis, all metrics described are calculated based on the process of hold-out evaluation. This involves splitting an available dataset into two parts; a *training set* and a *testing set*. The training is used in conjunction with known class labels to construct a classifier. The class labels are used to estimate and minimize the expected error rate of that classifier. Then, the testing set is passed to the model, stripped of labels. By measuring its performance in classifying the testing set, a model is evaluated [50, 51].

The confusion matrix contains information about every classification. For n classes in the dataset it forms an $n \times n$ matrix. In this matrix, the sum of all entries in row i represents the amount of cases assigned to class i by the classifier. Meanwhile, the sum of elements in row j represents the number of elements that class j actually contains. Every cell ij in the matrix represents how many cases of class j have been predicted to belong to class i. If $i = j$, cell ij contains *true* predictions. If $i \neq j$, all predictions in that cell are *false* [50].

For a binary classification, such as "pCR of a patient", classifications can be separated into four distinct groups. If a prediction accurately assigns a label ($i = j$), that prediction is true; is the prediction positive ("the patient will recover"), it is a True Positive (TP), if not ("the patient will not recover"), it is a True Negative (TN). In contrast, if the prediction fails ($i \neq j$), a positive assignment results in a False Positive (FP) ("the patient was expected to

recover, but did not"), while a negative prediction yields a False Negative (FN) ("the patient recovered, but was not expected to do so"). These relations between predicted and actual outcomes are shown in Table 2.

	Positive Prediction	**Negative Prediction**
Positive Case	*True Positive (TP)*	False Negative (FN)
Negative Case	False Positive (FP)	*True Negative (TN)*

Table 2: A confusion matrix for binary classification. The vertical axis going from top left to bottom right represents correct classifications, written in *italics*. Adapted from [50].

The total number of predictions in one of these groups reveals little about the performance of a classifier. A high total count of TP classifications may just as well stem from a large sample size as from a well constructed model. For this reason, a *True Positive Rate (TPR)* can be calculated. By setting in contrast the number of TP predictions to that of positive cases in the dataset, the probability of a positive case being recognized as such is found. This ratio, as shown in Equation 3, can be applied to any group; by dividing the number of true assignments to a class by that of the members of that class, the rate of recognition of that class is gained. The same can be calculated from false assignments, yielding the rate of misclassification [50, 51].

$$\text{True Positive Rate} = \frac{\text{True Positive Predictions}}{\text{Positive Cases}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

In the context of classification, two more metrics based on the confusion matrix are used; *sensitivity*, which equates to the TPR, and *specificity*, which represents the ratio of TN predictions to the sum of FP and TN predictions, also known as the True Negative Rate (TNR). Respectively, these measures describe which portion of actually positive and actually negative cases were recognized as such. A composite metric of these scores is the *balanced accuracy*, defined as in Equation 4 [50–52]. By combining information about both the recognition of positive and negative cases, it provides an overview of the model's total accuracy. This is especially useful, if the dataset is unbalanced, i.e. if one label is more abundant than the other. If a dataset consisted of 90.0 % positive cases, and only 10.0 % negative cases, a model labelling every case as "positive" would achieve both a "normal" accuracy of 0.9 and a TPR of 1.0. By giving TNR an equal weight in the total score, the accuracy (now balanced) would sink to 0.5.

$$\text{Balanced Accuracy} = \frac{\frac{\text{True Positive Predictions}}{\text{Positive Cases}} + \frac{\text{True Negative Predictions}}{\text{Negative Cases}}}{2} = \frac{\text{TPR} + \text{TNR}}{2} \tag{4}$$

Another metric for model performance can be constructed by seeking an ideal balance in the trade-off between the TPR and the False Positive Rate (FPR). This is possible, because an RF classifier does not solely put any case into one of two classes. Due to its assembly of voting trees, an RF classifier can return the (estimated) probability of a case being positive. By moving the threshold for the minimum probability to classify a case as "true", the sensitivity of a model can be raised or lowered. If the model would accept all cases as true (a threshold of 0.0), it would achieve a TPR of 1.0, while raising its FPR to 1.0. As the FPR is directly correlated to the TNR, this being $\text{TNR} = 1.0 - \text{FPR}$, this significantly lowers the balanced accuracy. To achieve the best possible trade-off between these two metrics, an ROC curve can be constructed, by plotting the TPR against the FPR. One such curve is shown in Figure 4, representing the results of differing classifier thresholds.

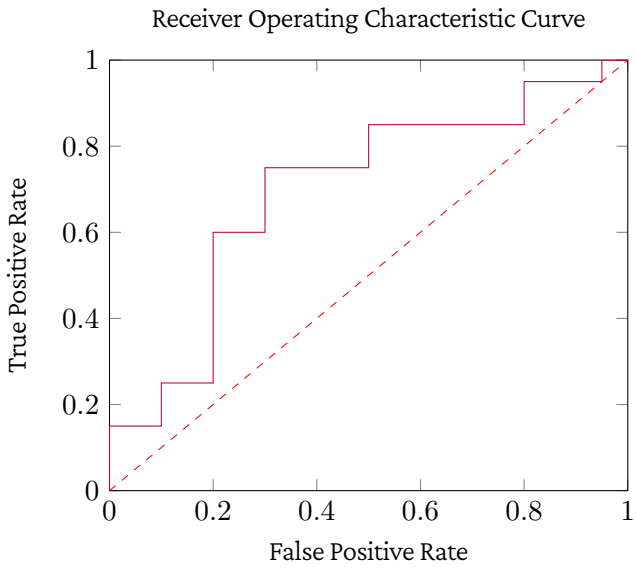


Figure 4: The ROC curve of a fictional classifier. The solid purple line shows the performance of the model, while the dashed red line shows the result of random guesses. The larger the area below the curve, i.e. the farther the model is away from random guessing, the bigger the AUC and the performance of the model. A curve running below the center line would suggest swapped labels.

The ROC curve shows the ability of a model to recognize a positive case as such, against the probability of misidentifying a negative case as positive. If these probabilities are equal, the model is guessing randomly. The total performance of the model at the full range of thresholds can be calculated by the deviation from random guesses, i.e. the center line. An ROC curve usually runs above this line, as the opposite would suggest swapped labels. Therefore, measuring all space below the curve yields a new metric, the so-called Area Under the Curve (AUC) [50, 51].

3 Materials and Methods

This section gives an overview about the dataset used for classification, as well as a conceptual overview of the experimental setup and the assumptions behind that setup. For a source-code focused view of the project, refer to Section 4.

3.1 Dataset

Though this thesis treats the analysis of a set of MRI scans, these scans were not produced by the author. The dataset was provided by the Institute of Radiology of the Salzburger Landeskliniken (SALK). It consists of MRI scans of 102 patients, where 24 reached pCR. Due to mismatches in geometry between annotation and scan, 9 scans could not be used. This leaves 93 patients, 22, or 23.4 %, of which achieved pCR. All scans were manually annotated.

The dataset may not be published, as it underlies an obligation of anonymity to the patients the scans were recorded of. Additional available metadata to that mentioned above is listed in the appendix under Section 9.4.

The exact definition of pCR is not always clear or consistent. In this context, it signifies absence of cancer tissue around the site of occurrence after treatment. This includes "[...] the total absence of neoplastic cells [tumors] in the rectal wall, mesorectum, and mesorectal lymph nodes" [2, p. 2823][53].

The dataset is not annotated consistently; most scans contain a 2-dimensional annotation, marking a slice of tumor, while 7 scans are annotated along all three axes. To avoid inconsistencies, all annotations were interpolated to form a 3-dimensional annotation from their largest slice, as described in 4.2.

3.2 Choice of Tools

As is popular with machine learning projects, this program is implemented in Python[5]. It offers the possibility of quick prototyping, intuitive and easily readable syntax and, most importantly, a large tool set for both radiomics and machine learning applications.

To address the earlier mentioned problem with the definition and extraction of radiomics features, a widely available and well documented library was chosen. PyRadiomics[6] covers, and is compatible to, most features defined by the IBSI, while documenting any differences in features definition or calculation (such as correcting for negative values, which may not be covered in the IBSI formula). Additionally, PyRadiomics offers tools to pre-process data, e.g. normalization of input images or passing input images through predetermined filters.

Next, the features extracted by PyRadiomics have to be processed. In the context of this project, this means to classify an input image into two categories: expected "pCR" or "non-pCR". As discussed earlier, a random forest classifier was chosen for this task. The toolkit used to implement this classification model is `sklearn`, which offers an implementation of a random forest classifier through its `sklearn.ensemble.RandomForestClassifier` class. It offers an accessible and easy to use interface and compatibility with other feature-rich data analysis tools (such as, but not limited to, `numpy`, `pandas` and `matplotlib`). Additionally, being an open-source project, the library is both cheap (i.e. free) to acquire and use (assuming no license violations) and easy to version and distribute, increasing reproducibility.

[5] More information available at https://www.python.org/.

[6] More information available at https://www.radiomics.io/pyradiomics.html.
Documentation available at https://pyradiomics.readthedocs.io/en/latest/.

As a connection between extraction and classification and analysis of results, a multitude of other libraries are required. Though important in their own rights, these libraries are not of special interest in the context of this thesis and will be addressed when relevant, instead of introducing them here.

3.3 Experimental Setup

As stated earlier, this thesis aims to determine, if pCR of a colorectal tumor patient can be predicted through a radiomics-based classification model, using MRI scans as input. To be considered successful, the model has to achieve a (balanced[7]) accuracy deviating significantly from 0.5, or 50.0 %.

The model chosen is an RF classifier. The model is trained on a set composed of 70.0 % of all pCR patients and an equal amount of non-pCR patients, both chosen randomly. The testing set consists of all remaining patients. Each patient is represented by an MRI scan accompanied by an annotation marking the contained tumor. Before feature extraction, this annotation is passed through a common extrapolation algorithm to create a 3-dimensional ROI. From this, radiomics features are extracted using PyRadiomics. A common extractor is used for all patients; its configuration file can be found in the appendix under Listing 2. The majority of available features are extracted, as a large fraction will be filtered out later.

Using these features, RF classification models are constructed. Each model is composed of 1000 estimators, each fully grown [54]. The performance for each model is evaluated over 100 instances of that model. These 100 models share their configuration, but are trained and tested using differently split datasets and receive different random-states.

With these classification models, two groups are determined; features that have a favorable impact on the accuracy of the model and feature groupings that may improve accuracy in combination. First, features are sorted by their contribution to a set of "sacrificial" models processing all available features. Then, features below a certain significance are dropped, to determine the optimal level of minimal importance. The remaining features are considered significant to the success of the classification model. If the model fed with the "ideal" set of features reliably performs better than random chance, the experiment is considered successful.

To determine possible synergies between features, models based on related features are constructed. Relation is considered in three categories; source image, feature group and feature extraction method. If a model based on a grouping performs better than the earlier "sacrificial" model, its performance is compared to the average "importance" of features making up that grouping. If the performance of that grouping is better than the expected performance for its average "importance", the grouping is considered "synergetic".

3.4 Hypothesis

A significant correlation between features extracted from annotated MRI scans and patient pCR is expected. This is based on the results of earlier studies (compare Section 1.3) investigating this relationship through comparable means.

As for accuracy, the filtering of less "helpful" features is expected to improve the final result. Especially the removal of zero-importance features should significantly improve performance through the removal of statistical "noise" [49]. Whereas the removal of low-importance features is expected to have a positive impact, the existence and detection of "synergetic" features is though to be possible, but not certain.

[7] Unless explicitly stated otherwise, all mentions of "accuracy" refer to "balanced accuracy".

4 Implementation

This section covers the details of implementing a program spanning the workflow of a machine learning model based on features extracted through a radiomics library. Overall, this can be roughly sectioned into three conceptual stages; preparation of the dataset, extraction of features and training and evaluation of the classifier model.

4.1 Code Structure

To represent the dataset, a class exists representing one single patient. This class, `TestCase`, consists of an MRI scan, the corresponding annotation, the pCR status of the patient and the metadata associated with the scan. On instantiation, the existence of both input images is checked, along with save-files from previous executions. These files preserve information from prior executions, enabling computationally intense tasks to be skipped, if related settings have not been changed. If a related save-file exists, information is restored from that file; if not, the file is created after the information becomes available. Save-files are identified in a way that aims to preserve the context they were created in; this method and its benefits and issues are treated in Section 4.3. The metadata these files, and in turn the class itself, consists of two sets of data.

Firstly, a second annotation. This second annotation is generated from the original annotation through an algorithm described in more detail in Section 4.2. This algorithm, applied consistently to every original annotation is an attempt to even out inconsistencies in the original dataset through a simple, easily reproduced method.

Lastly, the radiomics feature vector. This contains a list of feature identifiers and their respective value. The identifier unambiguously assigns each feature to the source image it was extracted from (or, the filter that was applied to that image), the feature group it belongs to and the features individual name (a textual representation of the feature extraction method). An overview about filters and feature groups is given in Section 2.1.4, whereas more detail is given in [25].

The dataset consists of more than one patient. Additionally, to fulfill their full function, each instance of a patient's representation depends on shared resources, such as the feature extractor's configuration. To manage the entirety of test cases and keep common data synchronized between them, the `TestCaseCollection` class is utilized.

The main focus of this class is to provide access to consistent sub-samples of the total dataset and to collections of their feature vectors. As such, it manages the central feature extractor shared between `TestCases`. This enables it to create random splits in the dataset, as described in Section 4.6, and return their combined feature vectors in a format parsable by `sklearn`'s classifier classes.

Now that the dataset can be split into multiple sub-sets, which provide input data for a classifier, such a model can be created. The connection between this input data and the model is offered by the `ClassificationRun` class. It accepts a `TestCaseCollection`, information about desired sample ratios (see Section 4.6) and a filter, which selects the features actually used for training the model specific to this execution (see Section 4.7). After creating and training its classifier, `ClassificationRun` then stores the model's evaluation, i.e. the testing data, along with generated predictions and their statistical descriptors, such as balanced accuracy.

Although this evaluation data is already valuable on its own, it focuses only on a single split of the total dataset, and on a single classifier model. Depending on which instance of a `ClassificationRun` is chosen, this could artificially make the model seem better or worse than what would be representative by accidentally (or intentionally) "cherry-picking" results. For this reason, a model is evaluated by calculating its accuracy as a mean over multiple executions, each featuring a different training and testing set, along with a different Random Forest makeup, determined by differing random generator seeds. As such, each grouping of `ClassificationRuns` is

managed by a `ClassificationRunCollection`.

A `ClassificationRunCollection` is representative of one "experimental setup"; if the effect of differing hyper-parameters or different feature filters should be compared, it is done using this class. Like the `ClassificationRun` instances it contains, it offers information about the setup's performance. Similar to the `TestCaseCollection` class, it keeps shared data consistent, such as dataset splitting information and feature filters. It also accepts a seed parameter for passing down to `ClassificationRun` instances. A collection of these instances is only useful, as either their seed or parameters differ. As parameters are explicitly being kept consistent between `ClassificationRun` instances, the seed received from the overarching `ClassificationRunCollection` is counted up from its base value for each new `ClassificationRun` instance. This ensures different, but reproducible, random-states.

Now with all classifications done, information each instance is preserved. Metrics such as ROC curve, targets (actual case labels) and predictions are saved to then be used to evaluate the model. This enables later analyzation by additional metrics and their visualization, without the need to repeat the classification with the same parameters.

4.2 Annotation Extrapolation

While all MRI scans in the dataset consist of 3-dimensional data, most are segmented along only 2 axes, forming a flat slice in the scan. An example would be an annotation that cover only the sagittal plane. Note, that the annotation file for a 2-dimensional annotations is still a 3-dimensional image, but the ROI is marked only in one slice, as can be seen in Figure 5.

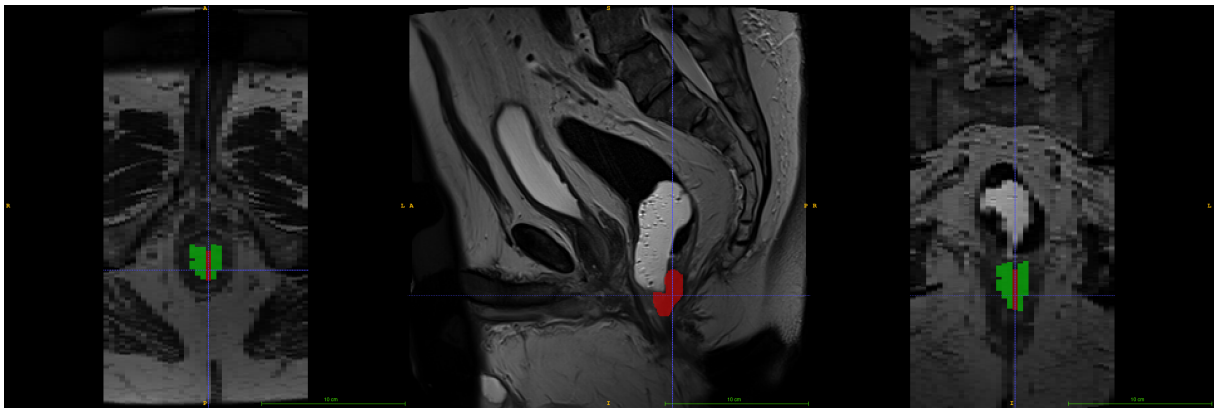


Figure 5: A manual tumor annotation. A 2-dimensional annotation is marked in red. As it is formed by voxels, it still possesses a non-zero width. If combined with the annotation marked in green, it forms a 3-dimensional annotation. From right to left: axial plane, sagittal plane and coronal plane.

For ease of understanding, 2-dimensional annotations will be treated as 2-dimensional images, if not noted otherwise. This mixture of annotation types poses a set of problems; requirements of radiomics features and consistency. A group of PyRadiomics' features, namely "Shape Features (3D)" rely on a 3-dimensional ROI. With a largely 2-dimensional dataset, these features would need to be excluded. Furthermore, it has been suggested, that analysis of complete tumors may be more representative of a tumor's features than analysis of its biggest slice, especially in colorectal tumors [22, 55]. In contrast, some scans in the dataset are annotated along 3 dimensions. This inconsistency in input data may lead to varying ratios of 2D and 3D annotated data in subsamples, strongly impacting reproducibility.

4.2.1 Creation of 2D annotations

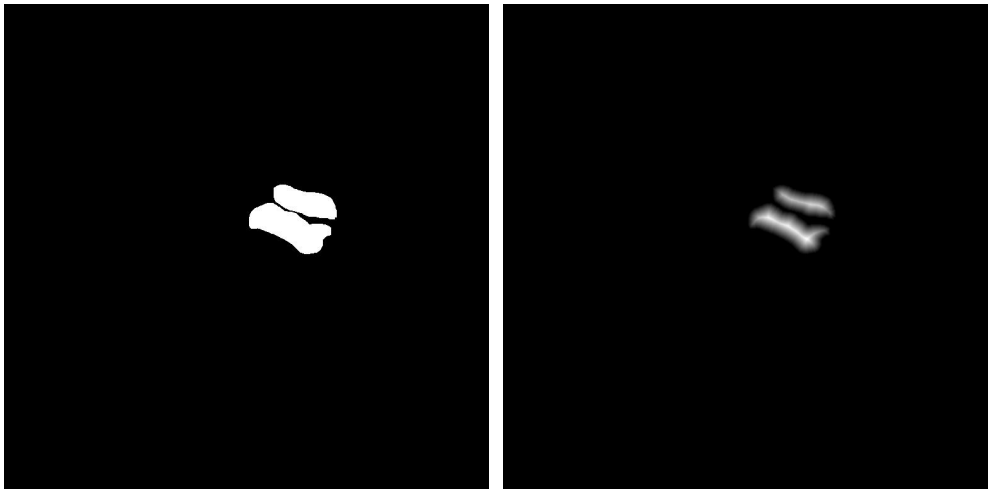
As all 3D annotations in the dataset were created manually, they can neither be recreated automatically nor reliably, without involving the original annotators. To avoid a mix of manual and automated 3D annotations, the same 3D reconstruction algorithm will be applied to all annotations. Manual 3D annotations will therefore be treated as 2D annotations and must be reduced by one axis.

To lose as little information as possible, while still dropping information along the third axis, the largest area along two axes in the 3D annotation is used. To determine this slice of the original annotation, the annotation is split into 2-dimensional slices along each axis. For each slice, all marked voxels marked as part of the ROI can be counted. As the axes of these voxels may be unevenly scaled, the voxel count must be multiplied with a correction factor to account for the different axis scales. The largest area after application of the correction factor is the largest 2D-slice by area, and therefore the new 2D annotation.

4.2.2 Creation of 3D annotations

Now, 3D annotations can be reconstructed from the fully 2D annotated dataset. The algorithm used here was chosen to be simple to implement and reproduce. It is based on "inverting" a 2-dimensional euclidean distance transformation.

The distance transform assigns each pixel in an image a value based on the pixel's distance to the nearest pixel with a non-zero color value [56]. Here, the distance is determined by the L_2 norm, adjusted for potentially uneven scaled axes. As demonstrated in Figure 6 using this transformation on a 2-dimensional annotation yields the distance of each pixel inside the ROI to the nearest pixel outside the ROI. This property will henceforth simply be referred to as a pixel's "2D distance".



(a) Base image. White areas mark the ROI.

(b) Euclidean distance transform implemented by [56]. Lighter colors indicate a greater distance.

Figure 6: A 2-dimensional ROI annotation. The contrast of both images has been increased to allow for easier viewing.

To create a 3-dimensional annotation, new layers of voxels along the new third axis will be added. The distance of each new voxel in relation to every pixel in the original ROI is determined. Should the "2D distance" of a pixel in the ROI be larger than the distance between that pixel and the new voxel, the new voxel is added to the new

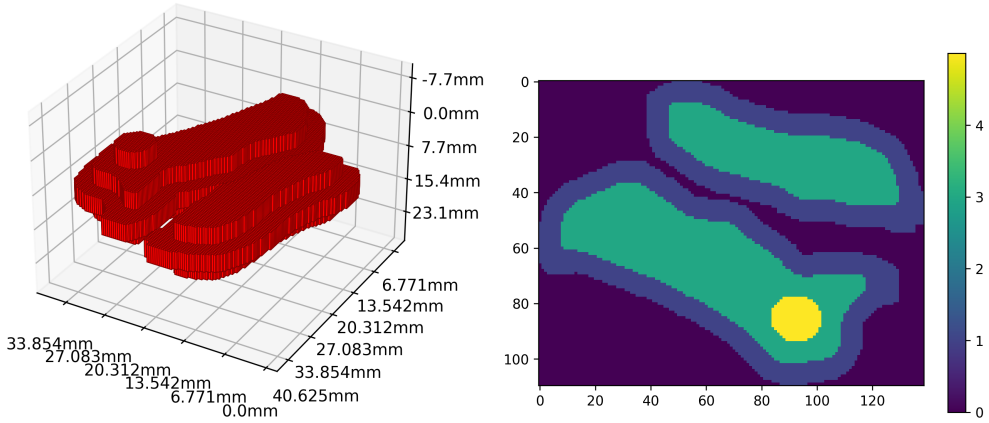
ROI. A pseudo-code implementation of this algorithm is shown in Listing 1.

```
annotation_2d = get_largest_slice(annotation_original)
annotation_2d_distances = euclidean_distance(annotation_2d)

for voxel_new in annotation_new:
    for pixel_old in annotation_2d:
        if euclidean_distance(voxel_new, pixel_old) <= annotation_2d_distances[pixel_old]:
            annotation_new[voxel_new] = 1
            break
```

Listing 1: 3D reconstruction.

This leads to a new 3-dimensional annotation, symmetrical along the initial 2-dimensional annotation slice. As can be seen in Figure 7, the thickness of the annotation increases with the "2D distance" of the underlying pixels. Should the 2-dimensional annotation along the base layer be circular, the 3-dimensional annotation will approach a sphere.



(a) A 3-dimensional plot of the annotation. Note the uneven resolutions along different axes.

(b) The 3-dimensional annotation projected back onto the original 2-dimensional annotation. The shape is symmetrical along the 2-dimensional base layer.

Figure 7: The 3-dimensional annotation generated from a 2-dimensional annotation. Both plots are cropped to the ROI.

4.2.3 Optimization of 3D-reconstruction

The algorithm described above is inefficient for large images. Its complexity can be described by $\mathcal{O}(n \cdot m)$, where n is the total number of voxels in the new annotation and m is the number of voxels (or pixels, as it only uses two axes) in the original 2-dimensional annotation. As tumors (and therefore tumor annotations) cover only small parts of the total 3D image, most pixels in the new annotation (n) are checked[8] for no reason; the distance of the majority of voxels to the ROI is far greater than the "2D-distance" of any pixels in the original annotation.

The following changes in the algorithm defined earlier aim to reduce the amount of unnecessary distance measurements. Instead of checking every new voxel in the new annotation for its distance to a pixel in the original annotation, a cube of influence is calculated for that pixel. In the center of this cube of influence lies a pixel in the original annotation. The size of the cube is determined by the pixel's "2D-distance"; the "2D-distance" is

[8] "Checked" refers a voxel's distance to a pixel in the original annotation being calculated and compared to that pixel's "2D-distance".

divided by each axis' scale. This, rounded down to the next whole number, gives the "radius" of the cube along each axis. For example, a pixel may possess a "2D-distance" of 10 mm. For axis scales of 3 mm, 3 mm and 12 mm, the cube of influence would measure $7\times 7\times 1$ voxels. These dimensions result from the radii (3, 3, 0) being applied in both directions (e.g. "forward" and "backward") and the initial voxel being added in the center. Each voxel in this cube is now checked to be closer to the original pixel than the pixel's "2D-distance". If so, the voxel is added to the new annotation. Should the voxel already be part of the new annotation (because it was already assigned this value through comparison to an earlier pixel), it is not checked again.

4.2.4 Rationalization

As of time of writing, the (semi-)automatic segmentation of tumors is a problem without a generally agreed-on solution. Current approaches include Deep-Learning (DL) based approaches such as [23] and [8]. To train DL models accurately, a certain amount of input data is required, both as ground truth and training data. In [23], 300 3-dimensionally annotated MRI scans are used for training, validation and testing, while [8] uses a total of 140 scans. These quantities of scans far outweigh the total count of 7 3-dimensionally annotated scans available here.

Another common approach is semi-automatic segmentation. [6] proposes both an automatic and a semi-automatic method of segmentation. While lightening the burden on medical professionals, both of these methods rely on prior medical knowledge, or at the very least, human interaction. As the relevant medical information is not available in this project, no approach described by [6] can be utilized. Different semi-automatic segmentation methods, even if they may not depend on any medical training, are still, by definition, based on human interaction. As one the main goals of the 3D-reconstruction described here is to be easily reproducible, introducing a human factor may be detrimental this objective.

4.3 Reducing Redundant Work

Save-files are a persistent storage of information between program executions, that are used to avoid redundant work. This section gives an overview about their management.

Though it may be possible to identify a save-file through its content, no additional identifier are present in the actual file content. Instead, identifiers are included in each file's name. This enables these files to be processed without the necessity of filtering out meta-information that is not relevant beyond locating the save-file in the first place. The pattern of these identifiers varies between use cases.

Interpolated annotations, as created in Section 4.2, are stored as valid NIfTI-1.1 annotations. Their name is rather simple, consisting of the identifier (i.e. the file name) of the MRI scan file they belong to, with a suffix representing the method of extrapolation. As this method has changed during development, it was necessary to differentiate between them, as different methods produced different extrapolations. Should a new method be employed in the future, this extrapolator-identifier can change, although it can be reused if the change consists of mere optimization, which does not change the final result.

Feature vectors depend on three factors; the input MRI scan, its annotation and the extractor's configuration. Therefore, to be able to tell if a save-file is relevant to the current execution, these factors must be known, not only from the running program, but also from the save-file. For this reason, the file name contains a representation of the path to both the scan file and the annotation used (e.g. the extrapolated annotation) and a representation of the extractor's configuration. "A representation" in this case means an SHA-256 hash. A hash guarantees consistency between filenames in that it explicitly defines the length of the resulting name, while the hash's hexadecimal representation guarantees a limited set of legal characters. SHA-256 meanwhile is seen as an ac-

ceptable trade-off between performance and probability of a collision, even for comparatively large datasets.

An additional trade-off is chosen in how to arrive at these hashes. It is assumed, that NIfTI-1.1 files adhere to a strict structure and are not changed in-place, at least not without the user's explicit permission. It is therefore also assumed, that the same path always points to the same file, both in name and content. Treating the file path as a representation of the entire file massively cuts down on the amount of data to be hashed, increasing performance, as the permissible file size for raster-image formats is assumed to be magnitudes larger than the average permissible path length. In case a file is changed in-place, this can be signaled by the user, which forces the program to re-generate any save-files that would otherwise have been used.

The extractor configuration meanwhile is parsed before hashing, extracting only information that affects the generation of features. This strips out properties of the configuration such as comments and formatting, which may change without meaningfully affecting the configuration itself. Therefore, should the configuration be changed in a way only relevant to the user, such as through reformatting, the existing save-file is still recognized.

4.4 Reproducibility

Some components of this program rely on (pseudo-[9])randomness. Though this randomness is required, it poses a problem for both comparability between results and reproducibility of prior experiments. Results may vary between classifiers; if they do not operate on the same training data, it is difficult to tell if the change in input or differing hyperparameters caused that variation.

For this reason, every function that introduces randomness, such as the splitting of the dataset or the choice of features for a tree in a random forest, accepts a random seed. This random seed is a "starting point" for pseudo-random number generators, such as the *Mersenne Twister* used by Python's own `random` library. Supplying the same seed to the generator is bound to yield the same sequence of numbers every time, enabling later reproduction [57, 58].

4.5 Feature Extraction

To improve reproducibility, an IBSI-based extraction approach has been chosen. The extraction was accomplished through PyRadiomics' `radiomics.featureextractor.RadiomicsFeatureExtractor` class. A single instance of this extractor was shared between scans, ensuring consistent extractor parameters. The configuration containing these parameters is a customized version of PyRadiomics' provided example configuration [25]; it can be found in its entirety in the appendix under Listing 2.

4.6 Sampling

Sampling is used to split the dataset into two groups: the training set and the testing set. The training set acts as a means to build the classifier model, which then attempts to predict pCR for each patient in the testing set.

Although the dataset is imbalanced with non-pCR patients outweighing pCR patients, the training set is selected to contain an equal amount of both scans. The amount of scans is therefore determined by the class with the fewest members, i.e. pCR patients. For a list of class member counts c, containing $|c|$ entries, the size of the training set is described in Equation 5, where p represents the percentage of scans to be allocated to the training set.

[9] All "randomness" mentioned in this thesis refers to pseudo-randomness, if not explicitly stated otherwise.

$$|\text{Training Set}| = \left\lfloor \frac{\min(c) \cdot p}{100} \right\rfloor \cdot |c| \tag{5}$$

Note, that even if p is set at 100 %, this would not select the entire dataset, if its classes are unbalanced. This is a limitation of this method of splitting the dataset, as it is assured to create a balanced subset.

4.7 Feature Filtering

In some cases, it is not desirable to use every feature associated with an MRI scan. For this reason, a feature-filtering functionality has been implemented. Filters take the form of instances of the *FeatureFilter* class. Each is instantiated with its associated filter function, which act similar to Python's own `filter` function [58]. Apart from value and name based filters, logical connectives such as AND and OR, ensuring functional completeness. As these filters can accept other filters as arguments, a filter graph can be represented by a single nested filter instance.

4.7.1 Filtering by Grouping

Features can be grouped by 3 categories. Firstly, the input image the feature was generated from. A feature can be extracted from an annotation or from the annotation combined with the MRI scan, to which a list of filters can be applied. Each filter enables the same features to be extracted again from a now different image.

The second category consists of the feature groups as defined by PyRadiomics. These groups are mostly a logical grouping of features describing similar image properties. The groups containing geometric features stand out as being extracted not from the MRI scan or one of its filtered variants, but from the annotation itself. When filtered to contain only feature from a single group, all feature sets apart from those consisting of geometric features will therefore contain features extracted from multiple inputs, due to input image filters.

Lastly, features can be grouped by their associated extraction method. An extraction method here refers to the equation or algorithm trough which a feature is calculated. This creates the smallest groups, as the number of feature extraction methods is larger than the number of filters. Additionally, features are not grouped together with similar other features anymore, as was the case when grouping by feature group.

The idea of creating classifier models based on a specific grouping of features is to search for feature groups with elevated impact in predicting pCR and for synergies between features. "Synergy" here means two, or more, features perform better when combined, than the sum of their impact would indicate. To test this, features with possible synergies must be included in the same tree(s) forming the random forest. For this reason, the set of possible combinations was limited by limiting the set of features to form these combinations.

$$\text{Possible Combinations} = \sum_{r=1}^{1743} \frac{1743!}{(1743-r)!r!} \tag{6}$$

Testing every possible combination of 1743 features, the count of which is shown in Equation6, is considered unreasonable, as creating this amount of models would require significant computational power, the scale of which is not available in this project. For this reason, the groupings described above are chosen with the assumption, that features are best combined with other "similar" features.

4.7.2 Filtering by Importance

With an input vector of 1743 features, it is not expected that each feature affects the final classification decision equally, or at all. A list of the "importance" of each feature, i.e. the impact its value will have on that final decision, can be generated after a random forest is built. If importance varies between features, those features can be sorted by this property. Then, by defining a threshold and dropping all features below that threshold, the list can be shrunk until only relatively important features remain.

It is expected, that a significant portion of all available features has only a minor impact on the final result, if any at all. Although not helpful for our purpose, these features still must be considered in building our random forest and may be chosen to form decision trees, introducing additional but ultimately pointless nodes into our decision trees. This may negatively impact the performance of the model [49]. For this reason, features of low importance will be filtered out.

To set a threshold, the distribution of feature importances must be known beforehand. This can be achieved by creating a "sacrificial" model; a model that is passed the same parameters as the final model, but not filtering out any features. From this model, the ranking of importance for each possible input feature can be extracted.

The creation of a model is a partially randomized process. The training and testing set is composed of randomly chosen scans and during random forest construction, their features are selected randomly to form trees. This means, that the accuracy of a model may naturally vary. To avoid basing the decision about a feature's relevancy to the overall result by a single "sacrificial" model, which may represent an outlier, the importance is determined by a weighted average over multiple models. To reduce the impact of inaccurate models on the total priority of features, each model is weighted by its accuracy, adjusted to conform to a range of "0" to "1", for the worst and best possible result respectively.

$$\text{Weighted Importance} = \sum_{n=1}^{m} \frac{b_n \cdot i_n}{m} \tag{7}$$

Assuming the importance of a feature is calculated over $m = 100$ models, where b_n is the (adjusted) accuracy of model n and i_n is the importance of this specific feature in this specific model, the weighing can be described as in Equation 7. As this importance is not normalized, care must be taken to calculate this value over the same number of models for each feature; for this reason, all features are included in the "sacrificial" models.

5 Results

As generated using the "sacrificial" model (compare Section 4.7.2), the distribution of feature importance, i.e. the weighted positive contribution of each feature to the total classification, is shown in Figure 8. Note the peak at, or near, zero.

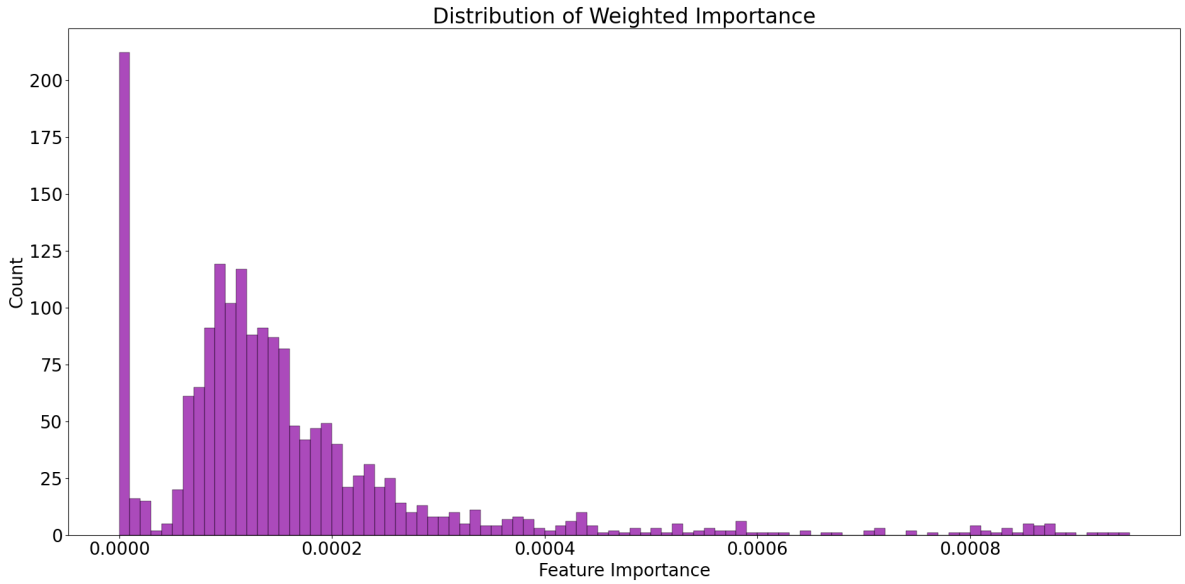


Figure 8: The distribution of feature importance. The importance has been calculated over 100 models. A bin-width of $1.00{\cdot}10^{-5}$ has been used in this plot. A large fraction of features has little or no (positive) effect on classification, while the majority of features have a similar, low impact.

As features are weighted not only by their total share of importance, but also by the performance of the associated model, an importance of near zero does not necessarily imply that a feature does not impact the classification at all. Instead, a feature could in theory harm classification. Therefore, it seems surprising, that removal of such "low-importance" features alone does not significantly improve the accuracy of the model. Figure 9 shows mean classifier accuracy and count of included features in relation to the minimum level of importance per feature.

Without filtering, the average accuracy of the model sits at a base level of 0.636. The first filter level removes features with an importance below $9.45{\cdot}10^{-6}$. Though removing these features increases average accuracy, the resulting improvement of approximately 1.46 % to 0.645 is barely noticeable. Furthermore, this improvement proves to be only temporary, with the first large monotonic increase in average accuracy beginning at the tenth filter level (with a value of $8.50{\cdot}10^{-5}$), filtering out 25.0 % of available features.

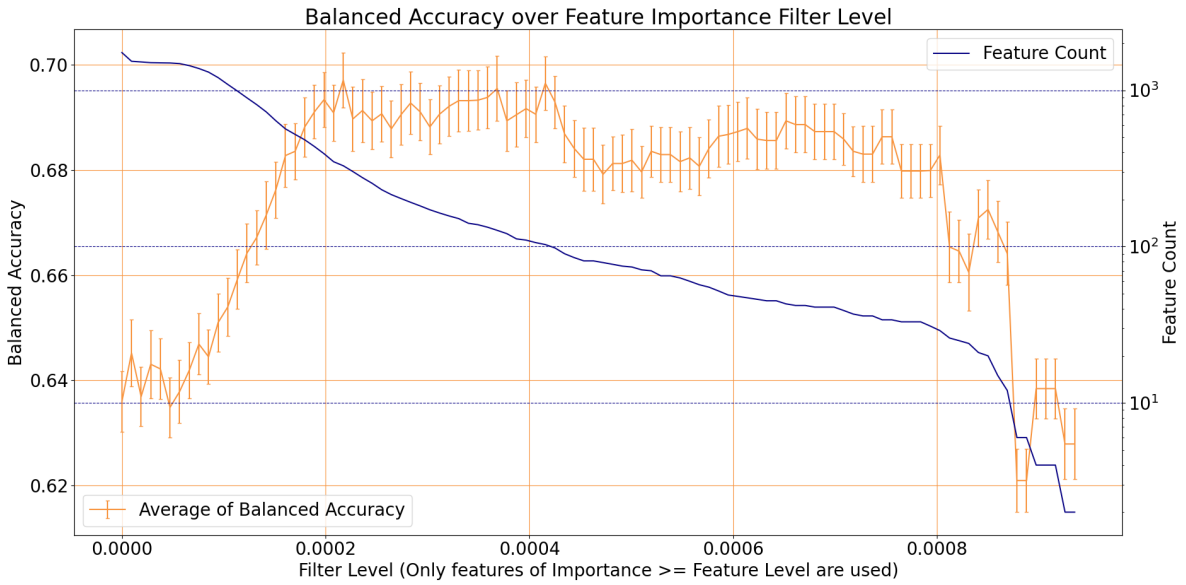


Figure 9: Amount of features per model, mean and variance of accuracy of this model over filter level. Note the logarithmic y-axis for Feature Count.

The overall peak in accuracy is reached shortly after the steady rise at a filter level of $2.17 \cdot 10^{-4}$ with a mean accuracy of 0.697 (95.0 % Confidence Interval [0.683, 0.712]) From here, accuracy stays relatively stable until reaching a second, slightly lower peak at a filter level of $4.16 \cdot 10^{-4}$ at an accuracy of 0.697 (95.0 % Confidence Interval [0.682, 0.711]).

Calculating the ROC and its associated AUC for the importance filter, yielding the "ideal" balanced accuracy over 100 models, results in the curves shown in Figure 10. On average, a model achieves mean AUC of 0.766 (95.0 % Confidence Interval: [0.753, 0.780]).

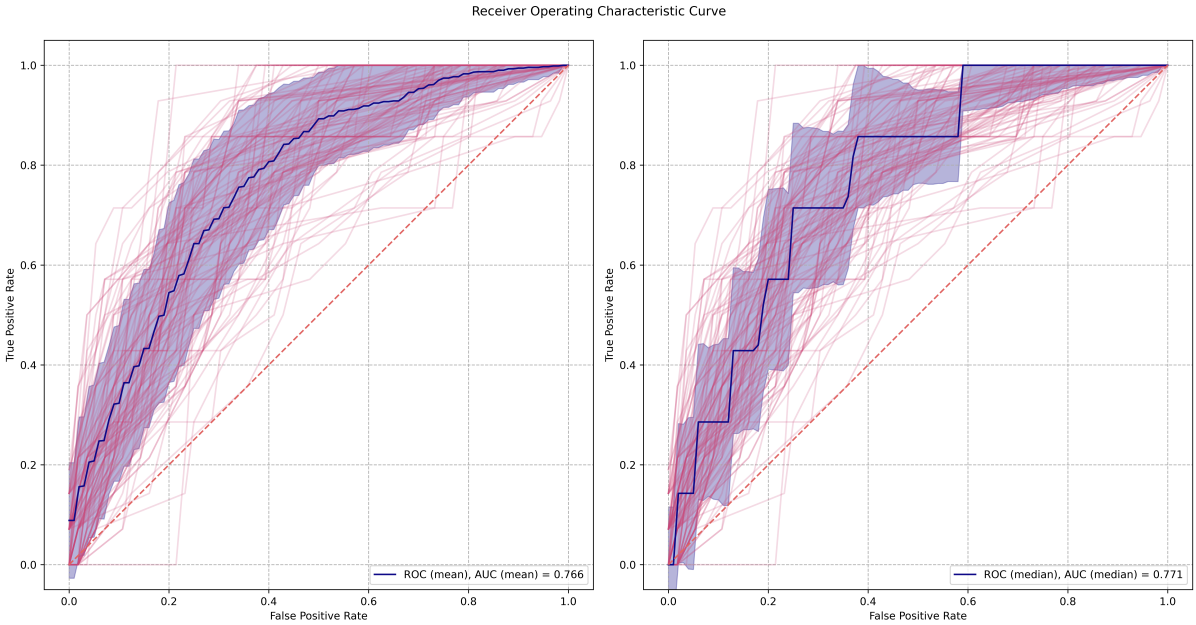


Figure 10: ROC curves for an importance filter level of $2.17{\cdot}10^{-4}$. Darker curves show the average, while the shading denotes standard deviation. Semitransparent curves show individual ROCs.

While features were treated as a single, all-containing set in the above classifications, grouping features shows clear differences in the effect of individual feature groups on a successful classification. "Synergy" inside a grouping was measured by the ratio of the balanced accuracy the model trained on that group reached (evaluated over 100 instances) to the average of importance of features in that group. The importance was measured as described in Section 4.7.2, comparing features against all available. Groupings deemed "synergetic" were those that achieved unusually high synergy. As expectations about differences in these synergy levels were unclear, an arbitrary condition has been chosen; synergy is "unusual" if lies outside the 95.0 % confidence interval of a linear regression over all groupings of the same category. It must be noted that there are grounds to suspect this methodology is flawed, as discussed in Section 6. Outlier groupings can be seen in Figure 11, sitting outside the shaded area marking the confidence interval.

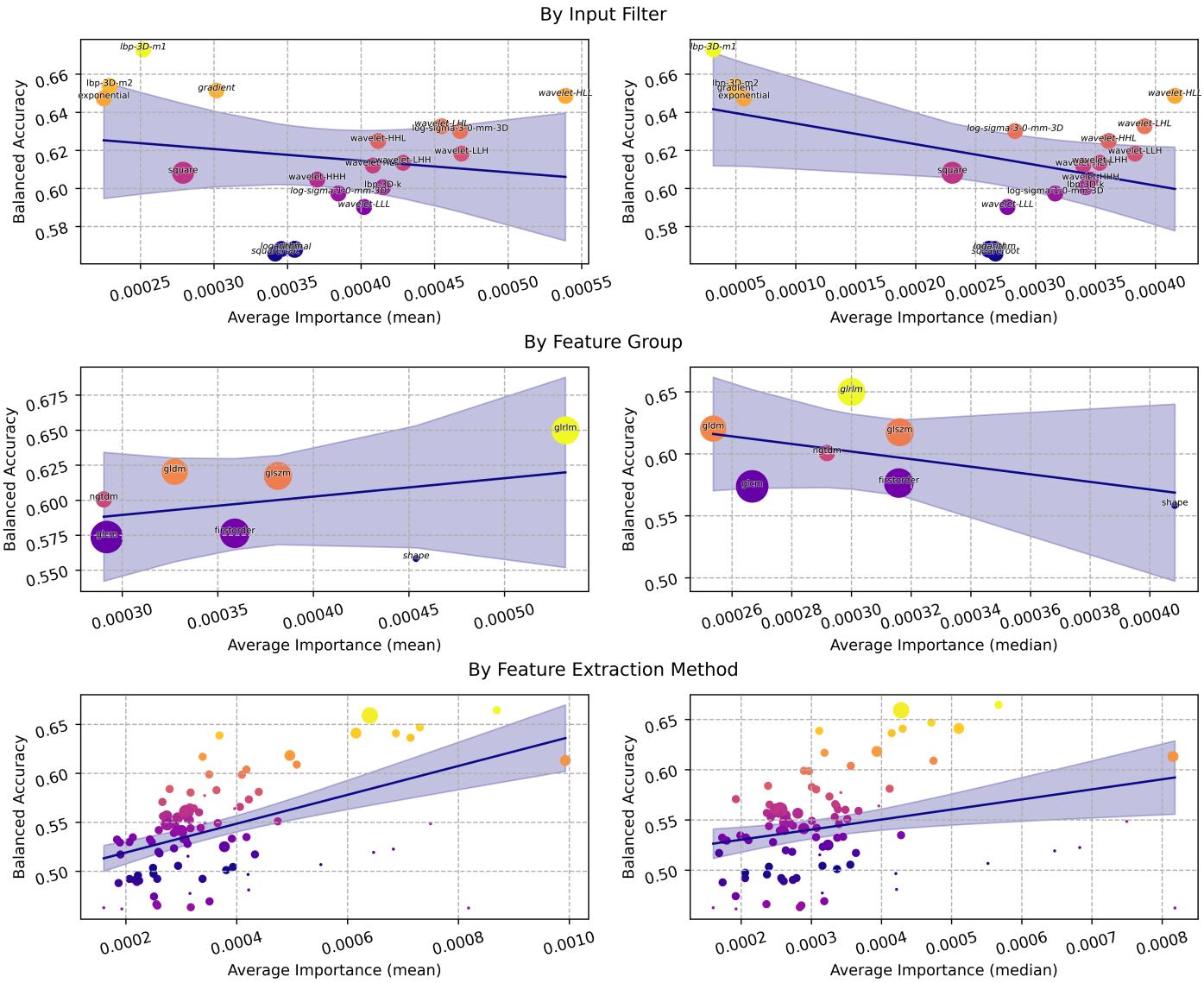


Figure 11: The balanced accuracy per feature grouping in relation to that grouping's average importance per feature. Larger dots denote larger groups. The line represents a linear regression, with the lightly shaded area marking the 95.0 % confidence interval. Groupings that fall outside the confidence interval are labelled in *italics*. Labels for feature method groups have been omitted in favor of legibility. The full information these plots are based on can be found in the appendix under Section 9.5.

6 Discussion

This thesis has shown the construction and evaluation of an RF-based machine learning classifier for the prediction of pCR in colorectal tumor patients. Considering the performance of this classifier, the prediction of pCR through MRI-based radiomics features seems, on average, possible.

The assignment of a weighted average of importance to individual features helped to improve the model. It must be considered that, as the testing set was involved in the selection of features, results generated through the classification of this set may present a biased evaluation of the total model performance. Due to its involvement in feature filtering, testing data was not completely "unknown" during the construction of the classifier model. This suggests, that a more representative model accuracy can be obtained using a training/validation/test set split, as to not weigh feature selection in favor of a "known" test set [51].

Although the usage of importance-based filtering raised the overall accuracy, the relation between filter levels and model performance did not conform to expectations. As stated in Section 3.4, filtering by importance was thought to immediately improve accuracy, as a large amount of unnecessary or non-discriminative features was removed from the dataset. Instead, performance climbed only with higher filter levels. Although more features were dropped than initially expected, the number of features used is still significantly higher than in comparable studies. The filter level for optimal balanced accuracy leaves a set of 329 features, compared to the 30 identifiers used in [3], 18 used in [4] and 4 used in [5]. As two of these studies [3, 4] achieved higher AUC scores than the model created here, both at its best performance and with a significantly smaller feature set, this may indicate the usage of a flawed feature selection method. It must be noted, that both of these studies relied on classification methods, that do not match an RF model exactly. When taking into consideration the study with a matching classifier model, [5], a similar AUC score was achieved[10], which calls this conclusion into question.

Another conflict arises in the determination of importances used here when compared to the performance of grouping-based models. Initially, it was assumed that these metrics share a positive covariance, that is, a group of features deemed more important on average is expected to perform better together. If a linear relationship is assumed, the simple linear regression describing this relationship, as seen in Figure 11, calls this into question. While an equal number of positive and negative correlations show up, none are statistically significant.

Two possible conclusions are drawn here to possibly explain this unexpected behavior. Firstly, the metric of importance, as defined in the context of this thesis, may not accurately describe how individual features contribute to the final result. An argument against this is posed by the improvement to model performance caused by importance-based filtering. Still, this improvement is not proven to arise solely (or mostly) from the removal of less important features, rather than from the reduction of features in general. The latter could be argued using the conclusion of [59], which states that an abundance in features in relation to training cases can negatively affect model performance.

Secondly, the possibility of an ill-fitting analysis. The connection between group importance may exist, but not be accurately characterized by a simple linear regression. This would imply the necessity of using either non-linear regression or a more careful examination of outliers. Although this circumstance is believed to be possible, its further exploration has been intentionally omitted in order to avoid accidental cherry-picking.

Having taken possible weaknesses of the metric defined in Section 4.7.1 and used in the analysis of Figure 11 into consideration, its results may still be of interest in the classification of MRI imaging data. When grouping features by the filter applied to the original imaging data, "LBPs" consistently achieve unexpectedly high synergy, both when measured by mean and median importance. This is consistent with [36], which introduced this metric as a tool for the analysis of tissue texture. Other filters such as "gradient" and "LoG" achieved high synergy in

[10] The classification scores for the model presented in this thesis is a composite value. In a single model instance, both better and worse results are possible.

either mean or median importance analysis, but failed to reproduce this result in the other. Wavelet filters, describing spatial frequencies across the input image, showed consistently high synergy for "HLL". Other wavelet configurations achieved unremarkable or even very low synergy. The only filters consistently resulting in a remarkably low synergy are "logarithm" and "squareroot", along with the "original", unfiltered image data.

When combining features by their respective feature group, as defined by PyRadiomics [24], the only groupings to stand out are "GLRLM" and "shape", performing better and worse than expected in one averaging category respectively. The low synergy in spite of high average importance in the "shape" group may indicate redundant information between members of that group. If so, individual features may achieve a high importance on their own, but do worse than expected in combination. Should this be true, it calls the effectiveness of the annotation interpolation algorithm used in this thesis (see Section 4.2) into question.

Both because of this category's large amount of groups and of the regression's comparatively small confidence interval, grouping by feature extraction method shows an unusual amount of outliers. In fact, the vast majority of available groupings, measured by both mean and median importance, are outliers, with 80 and 76 groupings out of a total of 99 respectively. Generally, outliers above the confidence interval largely consist of texture-based features, agreeing in part with the earlier groupings categories. In general, texture features appear to be important for successful classification, both in agreement with the findings of this thesis and with earlier works [3, 4]. In contrast, shape-based features dominate positions of high average importance but low synergy. This may lead to the assumption, that these features do not hold relevant information for classification. It must be considered though, that due to the non-existence of annotation-based filters, these groups largely consist of a single feature.

The grouping of features by category has shown groupings of both high accuracy and high synergy. Though no grouping has achieved better accuracy than the overall model, as described in Section 4.7.2, groupings like "lbp-3D-m1" or "ZonePercentage" show significant performance with both a smaller feature count and low assigned "importance". This may indicate the possibility of higher model accuracy and improved build duration through the choice of a more rigorous feature selection method.

Overall, a model that, on average, successfully predicts pCR has been constructed. Although techniques to improve model accuracy have been implemented, their exact effects are not yet fully understood and require further investigation.

7 References

[1] H. Sung, J. Ferlay, R. L. Siegel, M. Laversanne, I. Soerjomataram, A. Jemal, *et al.*, "Global Cancer Statistics 2020: GLOBOCAN Estimates of Incidence and Mortality Worldwide for 36 Cancers in 185 Countries," *CA: A Cancer Journal for Clinicians*, vol. 71, no. 3, pp. 209–249, 2021. DOI: https://doi.org/10.3322/caac.21660.

[2] L. Zorcolo, A. S. Rosman, A. Restivo, M. Pisano, G. R. Nigri, A. Fancellu, *et al.*, "Complete Pathologic Response after Combined Modality Treatment for Rectal Cancer and Long-Term Survival: A Meta-Analysis," *Annals of Surgical Oncology*, vol. 19, no. 9, pp. 2822–2832, Sep. 2012, ISSN: 1534-4681. DOI: 10.1245/s10434-011-2209-y.

[3] Z. Liu, X.-Y. Zhang, Y.-J. Shi, L. Wang, H.-T. Zhu, Z. Tang, *et al.*, "Radiomics Analysis for Evaluation of Pathological Complete Response to Neoadjuvant Chemoradiotherapy in Locally Advanced Rectal Cancer," *Clinical Cancer Research*, vol. 23, no. 23, pp. 7253–7262, 2017, ISSN: 1078-0432. DOI: 10.1158/1078-0432.CCR-17-1038.

[4] X. Yi, Q. Pei, Y. Zhang, H. Zhu, Z. Wang, C. Chen, *et al.*, "MRI-Based Radiomics Predicts Tumor Response to Neoadjuvant Chemoradiotherapy in Locally Advanced Rectal Cancer," *Frontiers in Oncology*, vol. 9, Jun. 2019. DOI: 10.3389/fonc.2019.00552.

[5] J. T. Antunes, A. Ofshteyn, K. Bera, E. Y. Wang, J. T. Brady, J. E. Willis, *et al.*, "Radiomic Features of Primary Rectal Cancers on Baseline T_2-Weighted MRI Are Associated With Pathologic Complete Response to Neoadjuvant Chemoradiation: A Multisite Study," *Journal of Magnetic Resonance Imaging*, vol. 52, no. 5, pp. 1531–1541, March 2020. DOI: 10.1002/jmri.27140.

[6] M. M. van Heeswijk, D. M. Lambregts, J. J. van Griethuysen, S. Oei, S.-X. Rao, C. A. de Graaff, *et al.*, "Automated and Semiautomated Segmentation of Rectal Tumor Volumes on Diffusion-Weighted MRI: Can It Replace Manual Volumetry?" *International Journal of Radiation Oncology, Biology, Physics*, vol. 94, no. 4, pp. 824–831, March 2016, ISSN: 0360-3016. DOI: 10.1016/j.ijrobp.2015.12.017.

[7] S. S. Mahdavi, N. Chng, I. Spadinger, W. J. Morris, and S. E. Salcudean, "Semi-automatic segmentation for prostate interventions," *Medical Image Analysis*, vol. 15, no. 2, pp. 226–237, April 2011. DOI: 10.1016/j.media.2010.10.002.

[8] S. Trebeschi, J. J. M. van Griethuysen, D. M. J. Lambregts, M. J. Lahaye, C. Parmar, F. C. H. Bakers, *et al.*, "Deep Learning for Fully-Automated Localization and Segmentation of Rectal Cancer on Multiparametric MR," *Scientific Reports*, vol. 7, no. 1, p. 5301, Jul. 2017, ISSN: 2045-2322. DOI: 10.1038/s41598-017-05728-9.

[9] P. M. Szczypiński, M. Strzelecki, A. Materka, and A. Klepaczko, "MaZda—a software package for image texture analysis," *Computer Methods and Programs in Biomedicine*, vol. 94, no. 1, pp. 66–76, April 2009. DOI: 10.1016/j.cmpb.2008.08.005. [Online]. Available: https://doi.org/10.1016/j.cmpb.2008.08.005.

[10] A. Zwanenburg, M. Vallières, M. A. Abdalah, H. J. W. L. Aerts, V. Andrearczyk, A. Apte, *et al.*, "The Image Biomarker Standardization Initiative: Standardized Quantitative Radiomics for High-Throughput Image-based Phenotyping," *Radiology*, vol. 295, no. 2, pp. 328–338, 2020, PMID: 32154773. DOI: 10.1148/radiol.2020191145.

[11] V. Kumar, Y. Gu, S. Basu, A. Berglund, S. A. Eschrich, M. B. Schabath, *et al.*, "Radiomics: the process and the challenges," *Magnetic Resonance Imaging*, vol. 30, no. 9, pp. 1234–1248, 2012, Quantitative Imaging in Cancer, ISSN: 0730-725X. DOI: https://doi.org/10.1016/j.mri.2012.06.010.

[12] The MathWorks, Inc., *MathWorks®*, 2021. [Online]. Available: https://www.mathworks.com/ (visited on 12/28/2021).

[13] Y. Kim and J. Kim, "Gradient lasso for feature selection," in *Proceedings of the Twenty-First International Conference on Machine Learning*, ser. ICML '04, Banff, Alberta, Canada: Association for Computing Machinery, 2004, p. 60, ISBN: 1581138385. DOI: 10.1145/1015330.1015364.

[14] P. Prasanna, P. Tiwari, and A. Madabhushi, "Co-occurrence of local anisotropic gradient orientations (collage): A new radiomics descriptor," eng, *Scientific reports*, vol. 6, pp. 37 241–37 241, November 2016, ISSN: 2045-2322. DOI: 10.1038/srep37241.

[15] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, October 2001, ISSN: 1573-0565. DOI: 10.1023/A:1010933404324.

[16] L. Breiman and A. Cutler, *Random Forests*. [Online]. Available: https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm (visited on 12/03/2021).

[17] S. S. F. Yip and H. J. W. L. Aerts, "Applications and limitations of radiomics," *Physics in Medicine and Biology*, vol. 61, no. 13, R150–R166, Jun. 2016. DOI: 10.1088/0031-9155/61/13/r150.

[18] P. Lambin, R. T. Leijenaar, T. M. Deist, J. Peerlings, E. E. de Jong, J. van Timmeren, *et al.*, "Radiomics: the bridge between medical imaging and personalized medicine," *Nature Reviews Clinical Oncology*, vol. 14, no. 12, pp. 749–762, October 2017. DOI: 10.1038/nrclinonc.2017.141.

[19] T. P. Coroller, V. Agrawal, E. Huynh, V. Narayan, S. W. Lee, R. H. Mak, *et al.*, "Radiomic-Based Pathological Response Prediction from Primary Tumors and Lymph Nodes in NSCLC," *Journal of Thoracic Oncology*, vol. 12, no. 3, pp. 467–476, March 2017. DOI: 10.1016/j.jtho.2016.11.2226.

[20] P. Lambin, E. Rios-Velazquez, R. Leijenaar, S. Carvalho, R. G. van Stiphout, P. Granton, *et al.*, "Radiomics: Extracting more information from medical images using advanced feature analysis," *European Journal of Cancer*, vol. 48, no. 4, pp. 441–446, 2012, ISSN: 0959-8049. DOI: https://doi.org/10.1016/j.ejca.2011.11.036.

[21] S. Sanduleanu, H. Woodruff, E. de Jong, J. Van Timmeren, A. Jochems, L. Dubois, *et al.*, "Tracking tumor biology with radiomics: A systematic review utilizing a radiomics quality score," *Radiotherapy and Oncology*, vol. 127, May 2018. DOI: 10.1016/j.radonc.2018.03.033.

[22] F. Ng, R. Kozarski, B. Ganeshan, and V. Goh, "Assessment of tumor heterogeneity by CT texture analysis: Can the largest cross-sectional area be used as an alternative to whole tumor analysis?" *European Journal of Radiology*, vol. 82, no. 2, pp. 342–348, February 2013. DOI: 10.1016/j.ejrad.2012.10.023.

[23] H.-T. Zhu, X.-Y. Zhang, Y.-J. Shi, X.-T. Li, and Y.-S. Sun, "Automatic segmentation of rectal tumor on diffusion-weighted images by deep learning with U-Net," *Journal of Applied Clinical Medical Physics*, vol. 22, pp. 324–331, 9 2021. DOI: 10.1002/acm2.13381.

[24] J. J. M. van Griethuysen, A. Fedorov, C. Parmar, A. Hosny, N. Aucoin, V. Narayan, *et al.*, "Computational Radiomics System to Decode the Radiographic Phenotype," *Cancer Research*, vol. 77, no. 21, e104–e107, 2017. DOI: 10.1158/0008-5472.CAN-17-0339.

[25] J. van Griethuysen, A. Fedorov, N. Aucoin, J.-C. Fillion-Robin, A. Hosny, S. Pieper, *et al.*, *pyradiomics*. [Online]. Available: https://pyradiomics.readthedocs.io/en/latest/ (visited on 12/07/2021).

[26] A. Zwanenburg, S. Leger, M. Vallières, and S. Löck, "Image Biomarker Standardisation Initiative," arXiv preprint arXiv:1612.07003v11, 2019.

[27] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621, 1973. DOI: 10.1109/TSMC.1973.4309314.

[28] G. Thibault, B. FERTIL, C. Navarro, S. Pereira, N. Lévy, J. Sequeira, *et al.*, "Texture Indexes and Gray Level Size Zone Matrix Application to Cell Nuclei Classification," November 2009.

[29] P. Moulin, "Chapter 6 - Multiscale Image Decompositions and Wavelets," in *The Essential Guide to Image Processing*, A. Bovik, Ed., Boston: Academic Press, 2009, pp. 123–142, ISBN: 978-0-12-374457-9. DOI: https://doi.org/10.1016/B978-0-12-374457-9.00006-8.

[30] B. Vidakovic and P. Mueller, "Wavelets for kids: A tutorial introduction," Duke University, Tech. Rep., 1991.

[31] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society of London. Series B. Biological Sciences*, vol. 207, no. 1167, pp. 187–217, February 1980. DOI: 10.1098/rspb.1980.0020.

[32] *SimpleITK*, version 2.2.0.dev164. [Online]. Available: https://simpleitk.org/doxygen/latest/html/.

[33] R. Beare, B. Lowekamp, and Z. Yaniv, "Image Segmentation, Registration and Characterization in R with SimpleITK," *Journal of Statistical Software*, vol. 86, no. 8, 2018. DOI: 10.18637/jss.v086.i08.

[34] Z. Yaniv, B. C. Lowekamp, H. J. Johnson, and R. Beare, "SimpleITK image-analysis notebooks: A collaborative environment for education and reproducible research," vol. 31, no. 3, pp. 290–303, November 2017. DOI: 10.1007/s10278-017-0037-8.

[35] B. C. Lowekamp, D. T. Chen, L. Ibáñez, and D. Blezek, "The Design of SimpleITK," *Frontiers in Neuroinformatics*, vol. 7, 2013. DOI: 10.3389/fninf.2013.00045.

[36] T. Ojala, M. Pietikäinen, and T. Mäenpää, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[37] K. Doi, "Diagnostic imaging over the last 50 years: research and development in medical imaging science and technology," *Physics in Medicine and Biology*, vol. 51, no. 13, R5–R27, Jun. 2006. DOI: 10.1088/0031-9155/51/13/r02.

[38] X. Zhang, N. Smith, and A. Webb, "Medical Imaging," in *Biomedical Information Technology*, ser. Biomedical Engineering, D. D. Feng, Ed., Burlington: Academic Press, 2008, pp. 3–27, ISBN: 978-0-12-373583-6. DOI: https://doi.org/10.1016/B978-012373583-6.50005-0.

[39] R. Mann, *An Introduction to Particle Physics and the Standard Model*. CRC Press, November 2009. DOI: 10.1201/9781420083002.

[40] M. Elmaoğlu and A. Çelik, *MRI Handbook*. Springer US, 2012. DOI: 10.1007/978-1-4614-1096-6.

[41] R. H. Hashemi, W. G. Bradley, and C. J. Lisanti, *MRI: the basics*, 3rd ed. Philadelphia, PA: Lippincott Williams & Wilkins, 2010, OCLC: ocn498978042, ISBN: 9781608311156.

[42] D. A. Schoeller, "Changes in total body water with age," *The American Journal of Clinical Nutrition*, vol. 50, no. 5, pp. 1176–1181, November 1989. DOI: 10.1093/ajcn/50.5.1176.

[43] D. C. V. Essen, S. M. Smith, D. M. Barch, T. E. Behrens, E. Yacoub, and K. Ugurbil, "The WU-Minn Human Connectome Project: An overview," *NeuroImage*, vol. 80, pp. 62–79, October 2013. DOI: 10.1016/j.neuroimage.2013.05.041.

[44] M. Jenkinson, *NIfTI-1 Data Format*, October 2007. [Online]. Available: https://nifti.nimh.nih.gov/nifti-1/ (visited on 12/02/2021).

[45] A. T. Berztiss, "Trees as data structures," Department of Computing Science, University of Wollongong, Tech. Rep. 79-2, 1979. [Online]. Available: https://ro.uow.edu.au/compsciwp/3.

[46] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees*. Taylor & Francis, 1984, ISBN: 9780412048418.

[47] L. E. Raileanu and K. Stoffel, "Theoretical Comparison between the Gini Index and Information Gain Criteria," *Annals of Mathematics and Artificial Intelligence*, vol. 41, no. 1, pp. 77–93, May 2004. DOI: 10.1023/b:amai.0000018580.96245.c6.

[48] S. Suthaharan, "Decision Tree Learning," in *Machine Learning Models and Algorithms for Big Data Classification*, Springer US, 2016, pp. 237–269. DOI: 10.1007/978-1-4899-7641-3_10.

[49] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Springer series in statistics). Springer, 2009, ISBN: 9780387848846.

[50] N. Japkowicz and M. Shah, *Evaluating Learning Algorithms*. Cambridge University Press, 2009. DOI: 10.1017/cbo9780511921803.

[51] J. D. Kelleher, B. Mac Namee, and A. D'Arcy, *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies*. The MIT Press, Jul. 2015, p. 624, ISBN: 978-0262029445.

[52] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, "The Balanced Accuracy and Its Posterior Distribution," in *2010 20th International Conference on Pattern Recognition*, IEEE, August 2010. DOI: 10.1109/icpr.2010.764.

[53] A. C. Cazador, R. F. Coll, F. O. Pujol, A. M. Grillo, M. P. de Palol, N. G. Romeu, *et al.*, "Clinical and Oncological Results of the Pathological Complete Response in Rectal Cancer After Neoadjuvant Treatment," *Cirugía Española (English Edition)*, vol. 91, no. 7, pp. 417–423, August 2013. DOI: 10.1016/j.cireng.2013.11.003.

[54] J. du Boisberranger, J. V. den Bossche, L. Estève, T. J. Fan, A. Gramfort, O. Grisel, *et al.*, *scikit-learn*. [Online]. Available: https://scikit-learn.org/stable/index.html.

[55] L. Geewon, L. H. Yun, K. E. Sook, and J. W. Kyoung, "Radiomics and imaging genomics in precision medicine," *Precision and Future Medicine*, vol. 1, no. 1, pp. 10–31, 2017. DOI: 10.23838/pfm.2017.00101. [Online]. Available: http://www.pfmjournal.org/journal/view.php?number=9.

[56] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[57] M. Matsumoto and T. Nishimura, "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator," *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30, January 1998, ISSN: 1049-3301. DOI: 10.1145/272991.272995.

[58] Python Software Foundation, *The Python Language Reference*, version 3.10.1, December 14, 2021. [Online]. Available: https://docs.python.org/3/reference/ (visited on 12/14/2021).

[59] D. Foley, "Considerations of Sample and Feature Size," *IEEE Transactions on Information Theory*, vol. 18, no. 5, pp. 618–626, 1972. DOI: 10.1109/TIT.1972.1054863.

8 Glossary

AUC	Area Under the Curve
CoLlAGe	Co-occurrence of Local Anisotropic Gradient Orientations
CT	Computed Tomography
DL	Deep-Learning
DWI	Diffusion Weighted Imaging
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GLCM	Gray Level Cooccurence Matrix
GLDM	Gray Level Dependence Matrix
GLRLM	Gray Level Run Length Matrix
GLSZM	Gray Level Size Zone Matrix
IBSI	Image Biomarker Standardisation Initiative
LASSO	Least Absolute Shrinkage and Selection Operator
LBP	Local Binary Pattern
LoG	Laplacian of Gaussian
MRI	Magnetic Resonance Imaging
NGLDM	Neighboring Gray Level Dependence Matrix
NGTDM	Neighboring Gray Tone Difference Matrix
pCR	Pathological Complete Remission
PD	Proton Density
PET	Positron Emission Tomography
RF	Random Forest
ROC	Receiver Operating Characteristic
ROI	Region of Interest
SALK	Salzburger Landeskliniken
SVM	Support Vector Machine
TE	Echo Time
TN	True Negative
TNR	True Negative Rate

TP	True Positive
TPR	True Positive Rate
TR	Repetition Time

9 Appendix

This section contains complimentary details to information introduced in the main body of this thesis. It is referred to, if data provided here may potentially, but not necessarily, be relevant to the reader.

9.1 Additional MRI Image Credit

To illustrate the difference between weightings in MRI scans, a comparison between T_1 and T_2 weighted images of a human brain were shown in Figure 2. Data were provided in part by the Human Connectome Project, WU-Minn Consortium (Principal Investigators: David Van Essen and Kamil Ugurbil; 1U54MH091657) funded by the 16 NIH Institutes and Centers that support the NIH Blueprint for Neuroscience Research; and by the McDonnell Center for Systems Neuroscience at Washington University. The MRI scan in question was taken of patient 100206.

9.2 PyRadiomics Extractor Configuration

The configuration in Listing 2 is supplied to PyRadiomics' `radiomics.featureextractor.RadiomicsFeatureExtractor` class, providing extraction parameters.

```
  setting:
  binWidth: 25
  label: 1
  interpolator: sitkBSpline
  resampledPixelSpacing: null
  weightingNorm: null
  normalize: true
imageType:
  Original: {}
  LoG:
    sigma:
      - 1
      - 3
  Wavelet:
    binWidth: 10
  Square: {}
  SquareRoot: {}
  Logarithm: {}
  Exponential: {}
  Gradient: {}
  LBP3D: {}
featureClass:
  shape: null
  firstorder: []
  glcm:
    - Autocorrelation
    - JointAverage
    - ClusterProminence
    - ClusterShade
    - ClusterTendency
    - Contrast
    - Correlation
    - DifferenceAverage
    - DifferenceEntropy
    - DifferenceVariance
    - JointEnergy
```

```
37      - JointEntropy
38      - Imc1
39      - Imc2
40      - Idm
41      - Idmn
42      - Id
43      - Idn
44      - InverseVariance
45      - MaximumProbability
46      - SumEntropy
47      - SumSquares
48    glrlm: null
49    glszm: null
50    gldm: null
51    ngtdm: null
```

Listing 2: PyRadiomics configuration file. This file is based on the default PyRadiomics configuration file[25]. The majority of features is enabled, to be selectively filtered out later. Some features are disabled, as recommended in the PyRadiomics documentation. Examples for reasons to disable certain features are obsolescense or redundance in combination with other features [25].

9.3 Software Versions

The experimental setup presented here depends on a collection of software distributions. To enable replication of this setup, the versions of all relevant tools are listed in Table 3, where Table 4 gives information about individual software libraries used in conjunction with the Python programming language.

Software	Version
Python	3.9.7
ITK-SNAP	3.8.0

Table 3: Versions of Software Tools used in the development of this thesis. Tools that do not have a direct impact on model performance and which are not specifically referenced, such as text editors, are omitted.

Library	Version
cycler	0.10.0
docopt	0.6.2
imageio	2.9.0
joblib	1.0.1
kiwisolver	1.3.2
matplotlib	3.4.3
networkx	2.6.2
nibabel	3.2.1
numpy	1.21.2
packaging	21.0
Pillow	8.4.0
pykwalify	1.8.0
pyparsing	2.4.7
pyradiomics	3.0.1
python-dateutil	2.8.2
PyWavelets	1.1.1
ruamel.yaml	0.17.14
ruamel.yaml.clib	0.2.6
scikit-image	0.18.3
scikit-learn	0.24.2
scipy	1.7.1
SimpleITK	2.1.0
six	1.16.0
sklearn	0.0
threadpoolctl	2.2.0
tifffile	2021.8.8

Table 4: Versions of Python libraries used in the development of this thesis.

9.4 MRI Metadata

Here, additional information the MRI scans making up the total dataset is provided. All MRI scans have been T_2 weighted, with a 16 bit color depth. Information about Voxel and Scan dimensions can be found in Table 5.

Voxel Dimensions	Scan Dimensions	Number of Scans
0.42253520 × 0.42253520 × 3.86000000	640 × 640 × 25	34
0.33854166 × 0.33854166 × 3.85000000	768 × 768 × 26	9
0.87500000 × 0.87500000 × 3.75000000	320 × 320 × 38	5
0.48828130 × 0.48828130 × 5.20000000	512 × 512 × 22	3
0.48828130 × 0.48828130 × 5.20000030	512 × 512 × 22	3
0.62500000 × 0.62500000 × 3.75000000	384 × 512 × 34	3
0.75000000 × 0.75000000 × 4.40000000	320 × 320 × 40	3
0.31250000 × 0.31250000 × 3.50000000	768 × 768 × 36	2
0.31413612 × 0.31413612 × 3.85000000	864 × 864 × 30	2
0.33854166 × 0.33854166 × 3.85000010	768 × 768 × 26	2
0.62500000 × 0.62500000 × 5.00000000	384 × 512 × 26	2
0.65104170 × 0.65104170 × 3.30000000	384 × 384 × 30	2
0.78125000 × 0.78125000 × 6.50000000	320 × 320 × 25	2
0.93750000 × 0.93750000 × 3.60000010	320 × 320 × 30	2
0.31413612 × 0.31413612 × 3.84999970	864 × 864 × 30	1
0.33678755 × 0.33678755 × 3.85000000	800 × 800 × 26	1
0.33854166 × 0.33854166 × 3.84999970	768 × 768 × 26	1
0.39062500 × 0.39062500 × 4.40000000	512 × 512 × 23	1
0.42253520 × 0.42253520 × 3.85999970	640 × 640 × 25	1
0.42253520 × 0.42253520 × 3.86000010	640 × 640 × 25	1
0.42857143 × 0.42857143 × 3.86000000	560 × 560 × 25	1
0.43750000 × 0.43750000 × 5.20000030	512 × 512 × 22	1
0.48828130 × 0.48828130 × 5.19999930	512 × 512 × 22	1
0.50781250 × 0.50781250 × 5.20000000	512 × 512 × 22	1
0.52083300 × 0.52083300 × 3.60000420	384 × 384 × 25	1
0.52734380 × 0.52734380 × 5.00000000	512 × 512 × 23	1
0.54455450 × 0.54455450 × 4.40000000	512 × 512 × 28	1
0.62500000 × 0.62500000 × 3.85000010	512 × 512 × 30	1
0.71875000 × 0.71875000 × 3.60000000	320 × 320 × 27	1
0.71875000 × 0.71875000 × 4.80000000	320 × 320 × 28	1
0.78125000 × 0.78125000 × 3.60000010	320 × 320 × 27	1
0.78125000 × 0.78125000 × 6.00000000	320 × 320 × 30	1
0.97656250 × 0.97656250 × 6.25000000	256 × 256 × 19	1

Table 5: Voxel and scan dimensions along with the number of scans conforming to these dimensions.

9.5 Grouping Accuracy versus Grouping Importance

This data makes up the base of Figure 11. Data points are sectioned into three groupings; grouping by input image filter, by feature group and by feature extraction method. The theory behind these groupings is covered in detail in Section 4.7.1.

Outliers, listed in Section 9.5.1, denote feature groupings that are either unexpectedly "synergetic", or unusually not so. An outlier is defined as being located outside the 95 % confidence interval of a linear regression of accuracy over average importance per feature, fitted over all groupings of that category.

9.5.1 Outliers

Table 6 lists all feature groupings that performed either significantly better or worse than expected. The table contains groupings of all categories. Groupings split by category along with the exact values of both average importance and accuracy of that grouping can be found in the following section.

Grouping Category	Average Function	Position relative to Confidence Interval	Name of Grouping
By Input Filter	Mean	Below	squareroot
By Input Filter	Mean	Below	original
By Input Filter	Mean	Below	logarithm
By Input Filter	Mean	Below	log-sigma-1-0-mm-3D
By Input Filter	Mean	Below	wavelet-LLL
By Input Filter	Mean	Above	wavelet-LHL
By Input Filter	Mean	Above	gradient
By Input Filter	Mean	Above	lbp-3D-m1
By Input Filter	Mean	Above	wavelet-HLL
By Input Filter	Median	Below	squareroot
By Input Filter	Median	Below	original
By Input Filter	Median	Below	logarithm
By Input Filter	Median	Below	wavelet-LLL
By Input Filter	Median	Above	log-sigma-3-0-mm-3D
By Input Filter	Median	Above	wavelet-HHL
By Input Filter	Median	Above	wavelet-LHL
By Input Filter	Median	Above	lbp-3D-m1
By Input Filter	Median	Above	wavelet-HLL
By Feature Group	Mean	Below	shape

Grouping Category	Average Function	Position relative to Confidence Interval	Name of Grouping
By Feature Group	Median	Above	glrlm
By Feature Extraction Method	Mean	Below	Flatness
By Feature Extraction Method	Mean	Below	Maximum3DDiameter
By Feature Extraction Method	Mean	Below	DependenceVariance
By Feature Extraction Method	Mean	Below	RunLengthNonUniformityNormalized
By Feature Extraction Method	Mean	Below	SurfaceVolumeRatio
By Feature Extraction Method	Mean	Below	Skewness
By Feature Extraction Method	Mean	Below	InterquartileRange
By Feature Extraction Method	Mean	Below	MeanAbsoluteDeviation
By Feature Extraction Method	Mean	Below	RootMeanSquared
By Feature Extraction Method	Mean	Below	Maximum2DDiameterColumn
By Feature Extraction Method	Mean	Below	Sphericity
By Feature Extraction Method	Mean	Below	LargeDependenceEmphasis
By Feature Extraction Method	Mean	Below	DifferenceVariance
By Feature Extraction Method	Mean	Below	Imc2
By Feature Extraction Method	Mean	Below	SizeZoneNonUniformityNormalized
By Feature Extraction Method	Mean	Below	MajorAxisLength
By Feature Extraction Method	Mean	Below	LeastAxisLength
By Feature Extraction Method	Mean	Below	Maximum2DDiameterSlice
By Feature Extraction Method	Mean	Below	90Percentile
By Feature Extraction Method	Mean	Below	JointEnergy
By Feature Extraction Method	Mean	Below	Variance
By Feature Extraction Method	Mean	Below	Mean
By Feature Extraction Method	Mean	Below	MaximumProbability
By Feature Extraction Method	Mean	Below	RunPercentage
By Feature Extraction Method	Mean	Below	HighGrayLevelRunEmphasis
By Feature Extraction Method	Mean	Below	DifferenceEntropy
By Feature Extraction Method	Mean	Below	RobustMeanAbsoluteDeviation
By Feature Extraction Method	Mean	Below	MinorAxisLength
By Feature Extraction Method	Mean	Below	Maximum2DDiameterRow
By Feature Extraction Method	Mean	Below	ShortRunHighGrayLevelEmphasis
By Feature Extraction Method	Mean	Below	JointEntropy
By Feature Extraction Method	Mean	Below	SurfaceArea
By Feature Extraction Method	Mean	Below	ShortRunEmphasis
By Feature Extraction Method	Mean	Below	LargeDependenceHighGrayLevelEmphasis
By Feature Extraction Method	Mean	Above	Busyness
By Feature Extraction Method	Mean	Above	LongRunLowGrayLevelEmphasis
By Feature Extraction Method	Mean	Above	SmallAreaHighGrayLevelEmphasis
By Feature Extraction Method	Mean	Above	GrayLevelNonUniformityNormalized
By Feature Extraction Method	Mean	Above	RunVariance
By Feature Extraction Method	Mean	Above	InverseVariance
By Feature Extraction Method	Mean	Above	VoxelVolume
By Feature Extraction Method	Mean	Above	Range
By Feature Extraction Method	Mean	Above	Contrast
By Feature Extraction Method	Mean	Above	Autocorrelation
By Feature Extraction Method	Mean	Above	SmallAreaLowGrayLevelEmphasis
By Feature Extraction Method	Mean	Above	SizeZoneNonUniformity
By Feature Extraction Method	Mean	Above	LongRunHighGrayLevelEmphasis
By Feature Extraction Method	Mean	Above	LargeAreaEmphasis
By Feature Extraction Method	Mean	Above	DifferenceAverage
By Feature Extraction Method	Mean	Above	GrayLevelNonUniformity
By Feature Extraction Method	Mean	Above	Coarseness
By Feature Extraction Method	Mean	Above	DependenceNonUniformityNormalized
By Feature Extraction Method	Mean	Above	DependenceNonUniformity
By Feature Extraction Method	Mean	Above	SmallDependenceLowGrayLevelEmphasis
By Feature Extraction Method	Mean	Above	MeshVolume
By Feature Extraction Method	Mean	Above	ZoneEntropy
By Feature Extraction Method	Mean	Above	ClusterProminence
By Feature Extraction Method	Mean	Above	GrayLevelVariance
By Feature Extraction Method	Mean	Above	Minimum
By Feature Extraction Method	Mean	Above	Energy
By Feature Extraction Method	Mean	Above	Elongation
By Feature Extraction Method	Mean	Above	Strength
By Feature Extraction Method	Mean	Above	Complexity
By Feature Extraction Method	Mean	Above	Idmn
By Feature Extraction Method	Mean	Above	RunEntropy
By Feature Extraction Method	Mean	Above	SumEntropy
By Feature Extraction Method	Mean	Above	Id
By Feature Extraction Method	Mean	Above	LargeAreaHighGrayLevelEmphasis
By Feature Extraction Method	Mean	Above	TotalEnergy
By Feature Extraction Method	Mean	Above	Idn
By Feature Extraction Method	Mean	Above	LargeAreaLowGrayLevelEmphasis
By Feature Extraction Method	Mean	Above	LongRunEmphasis
By Feature Extraction Method	Mean	Above	ZonePercentage
By Feature Extraction Method	Mean	Above	Median
By Feature Extraction Method	Mean	Above	SmallAreaEmphasis
By Feature Extraction Method	Mean	Above	ZoneVariance
By Feature Extraction Method	Mean	Above	ClusterTendency
By Feature Extraction Method	Mean	Above	Idm
By Feature Extraction Method	Mean	Above	DependenceEntropy
By Feature Extraction Method	Mean	Above	10Percentile
By Feature Extraction Method	Median	Below	Flatness
By Feature Extraction Method	Median	Below	Maximum3DDiameter
By Feature Extraction Method	Median	Below	DependenceVariance
By Feature Extraction Method	Median	Below	RunLengthNonUniformityNormalized
By Feature Extraction Method	Median	Below	SurfaceVolumeRatio
By Feature Extraction Method	Median	Below	Skewness
By Feature Extraction Method	Median	Below	LowGrayLevelRunEmphasis
By Feature Extraction Method	Median	Below	InterquartileRange
By Feature Extraction Method	Median	Below	MeanAbsoluteDeviation
By Feature Extraction Method	Median	Below	RootMeanSquared
By Feature Extraction Method	Median	Below	Maximum2DDiameterColumn
By Feature Extraction Method	Median	Below	Sphericity
By Feature Extraction Method	Median	Below	LargeDependenceEmphasis
By Feature Extraction Method	Median	Below	DifferenceVariance
By Feature Extraction Method	Median	Below	Imc2

Grouping Category	Average Function	Position relative to Confidence Interval	Name of Grouping
By Feature Extraction Method	Median	Below	SizeZoneNonUniformityNormalized
By Feature Extraction Method	Median	Below	MajorAxisLength
By Feature Extraction Method	Median	Below	LowGrayLevelZoneEmphasis
By Feature Extraction Method	Median	Below	LeastAxisLength
By Feature Extraction Method	Median	Below	Maximum2DDiameterSlice
By Feature Extraction Method	Median	Below	90Percentile
By Feature Extraction Method	Median	Below	JointEnergy
By Feature Extraction Method	Median	Below	Variance
By Feature Extraction Method	Median	Below	Mean
By Feature Extraction Method	Median	Below	MaximumProbability
By Feature Extraction Method	Median	Below	RunPercentage
By Feature Extraction Method	Median	Below	Imc1
By Feature Extraction Method	Median	Below	HighGrayLevelRunEmphasis
By Feature Extraction Method	Median	Below	DifferenceEntropy
By Feature Extraction Method	Median	Below	RobustMeanAbsoluteDeviation
By Feature Extraction Method	Median	Below	MinorAxisLength
By Feature Extraction Method	Median	Below	Maximum2DDiameterRow
By Feature Extraction Method	Median	Below	ShortRunHighGrayLevelEmphasis
By Feature Extraction Method	Median	Below	JointEntropy
By Feature Extraction Method	Median	Below	SurfaceArea
By Feature Extraction Method	Median	Below	ShortRunEmphasis
By Feature Extraction Method	Median	Below	LargeDependenceHighGrayLevelEmphasis
By Feature Extraction Method	Median	Above	LongRunLowGrayLevelEmphasis
By Feature Extraction Method	Median	Above	SmallAreaHighGrayLevelEmphasis
By Feature Extraction Method	Median	Above	GrayLevelNonUniformityNormalized
By Feature Extraction Method	Median	Above	RunVariance
By Feature Extraction Method	Median	Above	InverseVariance
By Feature Extraction Method	Median	Above	VoxelVolume
By Feature Extraction Method	Median	Above	Contrast
By Feature Extraction Method	Median	Above	Autocorrelation
By Feature Extraction Method	Median	Above	SmallAreaLowGrayLevelEmphasis
By Feature Extraction Method	Median	Above	SizeZoneNonUniformity
By Feature Extraction Method	Median	Above	LongRunHighGrayLevelEmphasis
By Feature Extraction Method	Median	Above	LargeAreaEmphasis
By Feature Extraction Method	Median	Above	DifferenceAverage
By Feature Extraction Method	Median	Above	GrayLevelNonUniformity
By Feature Extraction Method	Median	Above	Coarseness
By Feature Extraction Method	Median	Above	DependenceNonUniformityNormalized
By Feature Extraction Method	Median	Above	DependenceNonUniformity
By Feature Extraction Method	Median	Above	MeshVolume
By Feature Extraction Method	Median	Above	ZoneEntropy
By Feature Extraction Method	Median	Above	ClusterProminence
By Feature Extraction Method	Median	Above	GrayLevelVariance
By Feature Extraction Method	Median	Above	Minimum
By Feature Extraction Method	Median	Above	Energy
By Feature Extraction Method	Median	Above	Strength
By Feature Extraction Method	Median	Above	Idmn
By Feature Extraction Method	Median	Above	SumEntropy
By Feature Extraction Method	Median	Above	Id
By Feature Extraction Method	Median	Above	LargeAreaHighGrayLevelEmphasis
By Feature Extraction Method	Median	Above	Idn
By Feature Extraction Method	Median	Above	LargeAreaLowGrayLevelEmphasis
By Feature Extraction Method	Median	Above	LongRunEmphasis
By Feature Extraction Method	Median	Above	ZonePercentage
By Feature Extraction Method	Median	Above	Median
By Feature Extraction Method	Median	Above	SmallAreaEmphasis
By Feature Extraction Method	Median	Above	ZoneVariance
By Feature Extraction Method	Median	Above	ClusterTendency
By Feature Extraction Method	Median	Above	Idm
By Feature Extraction Method	Median	Above	DependenceEntropy
By Feature Extraction Method	Median	Above	10Percentile

Table 6: Groupings with unusually high or low "synergy". Those outside a 95 % Confidence Interval are considered outliers, where those performing better than expected are marked as "Above", while those performing worse than expected are labelled "Below".

9.5.2 Feature Grouping Performances

This section contains exact information about average importance, balanced accuracy and feature counts per grouping. Split by grouping category, Tables 7, 8 and 9 contain results for Input Image Filter, Feature Group and Feature Extraction Method groupings respectively. The data shown here forms the basis of Figure 11 and the associated discussion.

Grouping Name	Average Importance (mean)	Average Importance (median)	Balanced Accuracy	Feature Count
squareroot	0.000341704565399872	0.000266577969971479	0.565803571428571	91
original	0.000354917081698337	0.000261187760791617	0.568035714285714	105
logarithm	0.000345929141200555	0.000265853408533981	0.568214285714286	91
wavelet-LLL	0.000402072161452621	0.000276593490165019	0.590267857142857	91
log-sigma-1-0-mm-3D	0.000384638623273095	0.000316377087987373	0.597321428571429	91
lbp-3D-k	0.000414960374545107	0.00034170954195838	0.600625	91
wavelet-HHH	0.000370210030304343	0.000345888440136271	0.604642857142857	91
square	0.000279055112816945	0.000230604481671522	0.608214285714286	182
wavelet-HLH	0.000408122412609522	0.000339668522268521	0.612053571428572	91
wavelet-LHH	0.000428648739534644	0.000353465971157847	0.613482142857143	91
wavelet-LLH	0.000468245403758965	0.000382907832285733	0.618035714285571	91
wavelet-HHL	0.000411653667688676	0.000360961327310767	0.624910714285714	91
log-sigma-3-0-mm-3D	0.000467394071131184	0.000282871066317851	0.630178571428571	91
wavelet-LHL	0.000454730328222036	0.000390905069919834	0.632678571428571	91
exponential	0.000225155134390635	$5.68065789285015 \cdot 10^{-5}$	0.647321428571429	91
wavelet-HLL	0.000538989618678287	0.000416208609120368	0.648660714285714	91
gradient	0.000301757715918419	$4.97519742161806 \cdot 10^{-5}$	0.651428571428571	91
lbp-3D-m2	0.000229107937737779	$4.96886164714694 \cdot 10^{-5}$	0.653839285714285	91
lbp-3D-m1	0.000251987043357795	$3.11536672615141 \cdot 10^{-5}$	0.672946428571428	91

Table 7: Input Image Filter groupings alongside their average (mean and median) importance per feature, balanced accuracy reached by that group and number of features included.

Grouping Name	Average Importance (mean)	Average Importance (median)	Balanced Accuracy	Feature Count
shape	0.00045380555129263	0.000408408002069951	0.558392857142857	14
glcm	0.000291909052522544	0.000266757140080705	0.573660714285714	418
firstorder	0.000358882355981309	0.000315884424174178	0.576339285714286	342
ngtdm	0.000290326959296795	0.000291818624390038	0.600535714285714	95
glszm	0.000381474889504424	0.000316198002653161	0.617321428571428	304
gldm	0.000327477592478678	0.000253692504681163	0.620267857142857	266
glrlm	0.000531956079924822	0.000300015340885009	0.649821428571429	304

Table 8: Feature Group groupings alongside their average (mean and median) importance per feature, balanced accuracy reached by that group and number of features included.

Grouping Name	Average Importance (mean)	Average Importance (median)	Balanced Accuracy	Feature Count
LeastAxisLength	0.000192753020082659	0.000192753020082659	0.461285714285571	1
MajorAxisLength	0.000818285991585234	0.000818285991585234	0.462321428571429	1
SurfaceVolumeRatio	0.000160202667383835	0.000160202667383835	0.462589285714286	1
90Percentile	0.000317235886688426	0.00028364629145112	0.463125	19
Imc2	0.000256897953937047	0.000285880963573963	0.464910714285714	19
RunLengthNonUniformityNormalized	0.000255177233088193	0.000236130345513413	0.466160714285714	19
MeanAbsoluteDeviation	0.000351015317044901	0.000318526129453791	0.469107142857143	19
LargeDependenceHighGrayLevelEmphasis	0.000250813756179078	0.000192291163937927	0.474107142857143	19
MinorAxisLength	0.000315710812684378	0.000315710812684378	0.477232142857143	1
Maximum2DDiameterRow	0.00042158515736195	0.00042158515736195	0.480803571428571	1
LargeDependenceEmphasis	0.000186926328610863	0.000173633057949301	0.487857142857143	19
DifferenceVariance	0.000219852824714896	0.000260734025123532	0.489017857142857	19
DifferenceEntropy	0.00022240587193821	0.000273790754542432	0.490267857142857	19
JointEnergy	0.000216494359313307	0.000258034710540827	0.490982142857143	19
RunPercentage	0.000206661533045109	0.000205661705993437	0.492053571428571	19
Variance	0.000338397847686762	0.00027895287094278	0.492142857142857	19
MaximumProbability	0.000256652841901275	0.000257210029909079	0.492142857142857	19
JointEntropy	0.00022185944901707	0.000236873802028507	0.495714285714286	19
SurfaceArea	0.000420555567317798	0.000420555567317798	0.496607142857143	1
ShortRunHighGrayLevelEmphasis	0.000249840783482713	0.000205688270192137	0.497410714285714	19
RobustMeanAbsoluteDeviation	0.000381076857149254	0.000336670521582876	0.501071428571429	19
ShortRunEmphasis	0.000249101705165469	0.000239649557233807	0.503928571428571	19
InterquartileRange	0.000393146338396516	0.000315940752820833	0.504285714285714	19
SizeZoneNonUniformityNormalized	0.000294251197492025	0.000355621297018621	0.505446428571428	19
Sphericity	0.000551854706848811	0.000551854706848811	0.506696428571429	1
Flatness	0.000312163315905046	0.000312163315905046	0.515267857142857	1
LargeDependenceLowGrayLevelEmphasis	0.000190257994243145	0.000168590788961113	0.517053571428571	19
Skewness	0.000432913663509079	0.000363663609762297	0.517142857142857	19
HighGrayLevelRunEmphasis	0.000260786975960082	0.00027274977895857	0.518214285714286	19
Maximum3DDiameter	0.000646915606479321	0.000646915606479321	0.519196428571428	1
LowGrayLevelRunEmphasis	0.000258821347480441	0.000263650370320852	0.519732142857143	19
Maximum2DDiameterColumn	0.000682633477372861	0.000682633477372861	0.5225	1
Imc1	0.000287000408236279	0.000316358566440093	0.523303571428571	19
Mean	0.000377818305773541	0.000323827583827372	0.525	38
ShortRunLowGrayLevelEmphasis	0.00031136249442259	0.000245491144220808	0.527857142857143	19
SmallDependenceEmphasis	0.000189245644981453	0.000179573630877548	0.529375	19
SmallDependenceHighGrayLevelEmphasis	0.000206436275378163	0.000177526480431356	0.529732142857143	19
HighGrayLevelEmphasis	0.000246752882712125	0.000210576409339904	0.529821428571429	19
LowGrayLevelZoneEmphasis	0.000307892386525893	0.000336974512880458	0.532232142857143	19
RunEntropy	0.000184160172690058	0.000172995948945838	0.532321428571429	19
LowGrayLevelEmphasis	0.000243486977797869	0.000206808052467671	0.5325	19
HighGrayLevelZoneEmphasis	0.000303816979171O9	0.000306803247037094	0.532767857142857	19

Grouping Name	Average Importance (mean)	Average Importance (median)	Balanced Accuracy	Feature Count
RootMeanSquared	0.000391352315616504	0.000345336251110605	0.533214285714286	19
SmallDependenceLowGrayLevelEmphasis	0.00021262113701135	0.000198871824381	0.534553571428571	19
DependenceVariance	0.000417664975956439	0.000427940102875833	0.534821428571429	19
Entropy	0.000286725210583047	0.000262149940506973	0.539017857142857	19
SumSquares	0.000303241793860893	0.000255152000779382	0.539375	19
Uniformity	0.000291830012009122	0.00025928152390139	0.540625	19
Maximum	0.00030042817501506	0.000289285862560469	0.541428571428571	42
Range	0.000259447254962939	0.00025943399495331 2	0.541964285714286	19
TotalEnergy	0.000290822506333389	0.000307575549195901	0.542142857142857	19
JointAverage	0.000316772127095928	0.000239749562141409	0.543839285714286	19
Correlation	0.000335040800279584	0.000273971110896472	0.544285714285715	19
Busyness	0.000275929924815069	0.000339548339165088	0.545982142857143	19
Complexity	0.000289137696603458	0.000269678357950421	0.546964285714286	19
GrayLevelVariance	0.00027454611188747	0.000263326621196735	0.548214285714286	57
Maximum2DDiameterSlice	0.000749680265164642	0.000749680265164642	0.548482142857143	1
ClusterShade	0.000365756449491964	0.000300875321723022	0.549107142857143	19
Elongation	0.000342553699408943	0.000342553699408943	0.549464285714286	1
Kurtosis	0.000474110119530288	0.000351373545555505	0.550892857142857	19
ZoneVariance	0.000315602668535249	0.000242235016879686	0.552321428571428	19
SmallAreaLowGrayLevelEmphasis	0.000311547274277065	0.000333465446861615	0.552946428571429	19
Contrast	0.000293322112029501	0.000262840102410839	0.554464285714286	38
Idn	0.000309447215329252	0.000253575613407252	0.556071428571428	19
SumEntropy	0.000266992668900244	0.00024685254663712	0.556160714285714	19
Idmn	0.000307506887250623	0.000239556361227461	0.55625	19
GrayLevelNonUniformityNormalized	0.000273730197239957	0.00028079005274026	0.556607142857143	38
InverseVariance	0.000310352977602218	0.000259717600370205	0.556964285714286	19
DifferenceAverage	0.000312805776869658	0.000236635982440911	0.556964285714286	19
Coarseness	0.00030543849255068	0.000367342738201371	0.559017857142857	19
SmallAreaEmphasis	0.000332147735297015	0.000348605427210781	0.560089285714286	19
Id	0.00030904435 41503	0.000255542706374966	0.560357142857143	76
Strength	0.00030387150388422	0.000337932534687941	0.560535714285714	19
Idm	0.000307612483907285	0.000248533080285071	0.560982142857143	38
MeshVolume	0.000396260436822103	0.000396260436822103	0.563928571428571	1
Minimum	0.000322590164718114	0.000335782790267707	0.564196428571429	19
Autocorrelation	0.000315489946764868	0.000240520495818609	0.565267857142857	19
DependenceEntropy	0.000405783740605248	0.000334131628138018	0.565714285714286	19
LargeAreaHighGrayLevelEmphasis	0.000266070141636204	0.000192487504489202	0.570714285714286	19
10Percentile	0.000421905097723375	0.000325904314573446	0.573035714285714	19
VoxelVolume	0.000342122993679242	0.000342122993679242	0.577232142857143	1
ClusterTendency	0.00031679300339271	0.000307009874425878	0.580357142857143	19
DependenceNonUniformityNormalized	0.000439815981558903	0.000411743635551193	0.581071428571429	19
SmallAreaHighGrayLevelEmphasis	0.000363525535204834	0.000300776402024004	0.582767857142857	19
LargeAreaEmphasis	0.000279042845484471	0.000238397411459287	0.583928571428571	19
Median	0.000409351901406 95	0.000296801888753792	0.598571428571429	19
ClusterProminence	0.000350372592706282	0.000289438101256778	0.598928571428571	19
Energy	0.000417510890953797	0.000356320277750621	0.603839285714286	19
ZoneEntropy	0.000508343410352827	0.00047416607690 1191	0.608928571428571	19
RunLengthNonUniformity	0.000993044024156794	0.000815615846414965	0.613125	38
LargeAreaLowGrayLevelEmphasis	0.000338568351568341	0.00031884628186976	0.616875	19
SizeZoneNonUniformity	0.000495934005879104	0.000393519839808611	0.618214285714285	38
LongRunEmphasis	0.000713574165999038	0.000414606006678722	0.636339285714286	19
ZonePercentage	0.000368873418216379	0.000311370909530323	0.638660714285714	19
LongRunHighGrayLevelEmphasis	0.000687213072786737	0.000430207313367466	0.640803571428571	19
DependenceNonUniformity	0.000615258035982897	0.000510192739561261	0.641071428571429	38
LongRunLowGrayLevelEmphasis	0.000730322376983643	0.00047111136767 1612	0.646964285714286	19
GrayLevelNonUniformity	0.000640204165474765	0.000428002612759955	0.659196428571429	95
RunVariance	0.000869185422734627	0.000567049499388 05	0.664553571428572	19

Table 9: Feature Extraction Method groupings alongside their average (mean and median) importance per feature, balanced accuracy reached by that group and number of features included.