BIOMEDICAL ENGINEERING MSc PROGRAM
UNIVERSITY OF CRETE
TECHNICAL UNIVERSITY OF CRETE
FOUNDATION FOR RESEARCH AND TECHNOLOGY-HELLAS

# Application Grade Thesis

# Title: Deep radiotranscriptomic survival analysis for non-small cell lung cancer patients by utilizing machine learning methods

Student's Name: Nikoletta-Maria Koutroumpa

Supervisors' Name: Konstantinos Marias, Michalis Zervakis

2nd member committee name:

3rd member committee name:

Date of completion: 10 June 2022

*This dissertation is submitted as a partial fulfilment of the requirements for the Master's degree of Biomedical Engineering M.Sc. program*

# Acknowledgements

The Application Grade Thesis entitled: «Deep radiotranscriptomic survival analysis for non-small cell lung cancer patients by utilizing machine learning methods» was prepared under the supervision of Associate Professor of the Department of Electrical & Computer Engineering of Hellenic Mediterranean University, Konstantinos Marias and Professor of School of Electrical and Computer Engineering of Technical University of Crete, Michalis Zervakis. This Application Grade Thesis marks the completion of my studies at the Master Program «Biomedical Engineering» from the University of Crete, Technical University of Crete, and the Foundation for Research and Technology (FORTH-Hellas).

I would like to express my gratitude to both my supervisors, Assoc. Prof. Konstantinos Marias, and Prof. Michalis Zervakis for the assignment of such an interesting topic and for their guidance. My warmest thanks to Eleftherios Trivizakis, Ph.D. candidate researcher at the Medical School of University of Crete for the valuable help and the continuous support throughout the conduct and completion of this thesis.

Finally, I would like to thank my family and friends for their support in every step of my journey.

# Abstract

**Deep radiotranscriptomic survival analysis for non-small cell lung cancer patients by utilizing machine learning methods**

According to the World Health Organization, lung cancer is estimated to have the highest mortality rate worldwide. Lung cancer can be divided into two main categories: non-small cell lung carcinoma (NSCLC) and small cell lung carcinoma (SCLC), with the former being the most prevalent type of lung cancer, accounting for approximately 85% of cases. The majority of lung cancer cases are diagnosed after a symptom appears related to primary or metastatic disease. The progression of the disease is typically described using five stages, from 0 to IV. The accurate staging of lung cancer is essential to establishing a prognosis and selecting the optimal treatment. However, staging information is not necessarily predictive of the disease progression or the response to treatment. Several studies have investigated the relationship between image features and lung cancer. Radiomics refers to the extraction of a large number of features from medical images with the intent of creating mineable databases from radiological images. Image features can be used to reveal diagnostic, predictive, and prognostic associations in cancer patients via correlations with other response criteria like survival or response to treatment. The increase in deep learning methods has also paved the way for the extraction of high-dimensional deep features that could capture deeper the cancer information. Furthermore, advances in transcriptomics have provided genome-wide information on gene structure and gene function in order to reveal the mechanisms behind the biological processes of cancer.

In many cancer studies, the main outcome under assessment is the time to an event of interest. The event might be the death of the patient, or the recurrence of the disease after successful treatment. The modelling of time to event data is called survival analysis and it has been used in many areas, including the biomedical, social, and engineering sciences. Outcome modelling can be used for the identification of the prognostic signature of patients and the stratification according to their survival time into groups with different risks of experiencing the event. Several studies have been conducted that use single source data to investigate the survival of cancer patients, such as histologic, imaging, or molecular data.

This master thesis aims to investigate the synergetic properties of multi-view data sources such as radiomics, transcriptomics, and deep features, in developing machine learning models for survival analysis. The dataset used comprised of 211 Computer Tomography (CT) examinations, 130 RNA-seq vectors $(P_G)$ and clinical data with histology, genomic, semantic, survival and disease recurrence information. The intersection of the transcriptomic and imaging data was a subset of 115 patients and the patient cohort of survival included 40 subjects. Two commonly used machine learning methods have been examined for the classification of patients into low- and high-risk, random forest and support vector machine. The feature-fusion strategy included combining all features to perform

survival analysis and also combining only radiomics and deep features. The proposed deep radiotranscriptomic analysis resulted in a C-index $0.77 \pm 0.10$ using support vector machine with C-index in the range of 0.65 to 0.83. The C-index using random forest classifier was $0.74 \pm 0.11$, in the range of 0.63 to 0.81. Deep radiotranscriptomic analysis outperformed analyses comprised only of radiomics and deep features. In that case, random forest reached a C-index of $0.68 \pm 0.03$ and support vector machine a C-index of $0.73 \pm 0.07$. The deep features that resulted in the best predictions were mostly extracted from MobileNet, ResNet, DenseNet, and NasNet models. Combining imaging information in the form of radiomics and deep features and histologic in the form of transcriptomics improved classification metrics, such as C-index and better ranked the patients according to their risk of experiencing the event.

Parts of this work are included in the publication that is under review, entitled "Deep Radiotranscriptomics of Non-Small Cell Lung Carcinoma for Assessing High-Level Clinical Outcomes using Multi-View Analysis" conduced by Trivizakis Eleftherios, Koutroumpa Nikoletta-Maria, Souglakos John, Karantanas Apostolos, Zervakis Michalis E., Marias Kostas. Details regarding the selected parameters and the complete source code of the analysis are provided online at https://github.com/NikiKou/deep_radiotranscriptomics_survival_analysis.

**Keywords**

non-small cell lung cancer; deep features; radiotranscriptomics; radiomics; transcriptomics; survival analysis; machine learning; feature fusion

# Περίληψη

**Βαθιά μεταγραφωματική ανάλυση επιβίωσης ασθενών με μη-μικροκυτταρικό καρκίνο του πνεύμονα χρησιμοποιώντας μεθόδους μηχανικής μάθησης**

Σύμφωνα με τον Παγκόσμιο Οργανισμό Υγείας, ο καρκίνος του πνεύμονα αποτελεί τη μορφή καρκίνου με το υψηλότερο ποσοστό θνησιμότητας παγκοσμίως. Ο καρκίνος του πνεύμονα χωρίζεται σε δύο κατηγορίες, μη-μικροκυτταρικό και μικροκυτταρικό καρκίνο του πνεύμονα, με την πρώτη κατηγορία να κυριαρχεί με ποσοστό 85% των διαγνώσεων καρκίνου του πνεύμονα. Στην πλειοψηφία των ασθενών, η διάγνωση γίνεται μετά την εμφάνιση συμπτωμάτων που σχετίζονται με πρωταρχική ή μεταστατική ασθένεια. Η εξέλιξη της πάθησης χαρακτηρίζεται από πέντε στάδια, από 0 έως IV. Η άμεση διάγνωση και ανάλυση της ασθένειας είναι απαραίτητη για την επιλογή της κατάλληλης θεραπείας. Αρκετές μελέτες συσχετίζουν χαρακτηριστικά που προκύπτουν από ιατρικές εικόνες με τον καρκίνο του πνεύμονα. Η ραδιωμική (radiomics) αφορά στην εξαγωγή μεγάλου αριθμού χαρακτηριστικών από ιατρικές εικόνες με σκοπό τη δημιουργία βάσεων δεδομένων από δεδομένα απεικονιστικών μεθόδων. Τα χαρακτηριστικά της εικόνας μπορούν να χρησιμοποιηθούν για την εύρεση διαγνωστικών και προγνωστικών συσχετισμών σε ασθενείς με καρκίνο του πνεύμονα. Η διαθεσιμότητα δεδομένων ιατρικής εικόνας σε συνδυασμό με την αύξηση μεθόδων βαθιάς μάθησης (deep learning) άνοιξε το δρόμο για την εξαγωγή χαρακτηριστικών υψηλής ποιότητας που θα μπορούσαν να συμβάλλουν στην βαθύτερη κατανόηση της ασθένειας. Επιπλέον, η μεταγραφωματική (transcriptomics) παρέχει σημαντικές πληροφορίες για το γονιδίωμα, βοηθώντας στην κατανόηση των μηχανισμών πίσω από τις βιολογικές διεργασίες του καρκίνου.

Αρκετές μελέτες που σχετίζονται με τον καρκίνο στοχεύουν στην εύρεση του χρόνου μέχρι να εμφανιστεί το συμβάν του ενδιαφέροντος. Το συμβάν μπορεί να είναι ο θάνατος του ασθενούς ή η επανεμφάνιση της νόσου ύστερα από μία επιτυχή θεραπεία. Η μοντελοποίηση των δεδομένων χρόνου μέχρι την εμφάνιση του συμβάντος ονομάζεται ανάλυση επιβίωσης (survival analysis) και βρίσκει εφαρμογή στην βιοϊατρική, τη βιοστατιστική, καθώς και σε άλλες επιστήμες, όπως στη μηχανική. Αρκετές μελέτες χρησιμοποιούν δεδομένα από μία μόνο πηγή, όπως ιστολογικά δεδομένα, απεικονιστικά ή μοριακά, για την ανάλυση επιβίωσης ασθενών με καρκίνο.

Σκοπός της μεταπτυχιακής εργασίας είναι η ανάλυση επιβίωσης με χρήση μεθόδων μηχανικής μάθησης και χρησιμοποιώντας διαφορετικές πηγές δεδομένων, ραδιωμικής, μεταγραφωματικής και δεδομένων που προέκυψαν από την εφαρμογή μοντέλων βαθιάς μάθησης σε ιατρικές εικόνες (deep features). Το σύνολο των δεδομένων που χρησιμοποιήθηκε περιείχε 211 εικόνες αξονικής τομογραφίας, 130 φορείς RNA-seq και κλινικά δεδομένα με πληροφορίες ιστολογίας, γονιδιώματος, επιβίωσης και υποτροπής της νόσου. Από αυτά τα δεδομένα, ένα υποσύνολο με 40 ασθενείς χρησιμοποιήθηκε για την ανάλυση επιβίωσης.

Δύο μέθοδοι μηχανικής μάθησης έχουν χρησιμοποιηθεί ευρέως για την ταξινόμηση ασθενών σε περιπτώσεις υψηλού και χαμηλού κινδύνου, ο αλγόριθμος τυχαίων δασών (random forest) και οι μηχανές διανυσμάτων υποστήριξης (support vector machines). Δύο συνδυασμοί δεδομένων μελετήθηκαν, ο συνδυασμός όλων των δεδομένων (deep radiotranscriptomics) και ο συνδυασμός μόνο δεδομένων radiomics και deep features. Η προτεινόμενη ανάλυση με συνδυασμό όλων των δεδομένων, deep ratiotranscriptomics, οδήγησε σε C-index $0.77 \pm 0.10$ με μηχανές διανυσμάτων υποστήριξης και $0.74 \pm 0.11$, με τυχαία δάση. Με συνδυασμό μόνο των δεδομένων radiomics και deep features, οι μηχανές διανυσμάτων υποστήριξης κατέληξαν σε C-index $0.73 \pm 0.07$ και τα τυχαία δάση σε C-index $0.68 \pm 0.03$. Ο συνδυασμός όλων των χαρακτηριστικών οδήγησε σε μοντέλα με καλύτερη ικανότητα πρόβλεψης. Τα μοντέλα βαθιάς μάθησης που παρείχαν χαρακτηριστικά υψηλής ποιότητας ήταν τα MobileNet, ResNet, DenseNet και NasNet. Η μελέτη αυτή οδήγησε στο συμπέρασμα ότι η χρήση δεδομένων από διαφορετικές πηγές οδηγεί σε μοντέλα με καλύτερη πρόβλεψη της επικινδυνότητας της νόσου των ασθενών και σε καλύτερη κατηγοριοποίησή τους σε ασθενείς χαμηλού και υψηλού κινδύνου.

Τμήματα αυτής της εργασίας περιλαμβάνονται στη δημοσίευση με τίτλο “Deep Radiotranscriptomics of Non-Small Cell Lung Carcinoma for Assessing High-Level Clinical Outcomes using Multi-View Analysis” από τους Τριβιζάκης Ελευθέριος, Κουτρούμπα Νικολέττα Μαρία, Σουγκλάκος Ιωάννης, Καραντάνας Απόστολος, Ζερβάκης Μιχάλης Ε., Μαριάς Κώστας, η οποία βρίσκεται σε στάδιο αξιολόγησης για αποστολή σε επιστημονικό περιοδικό. Λεπτομέρειες σχετικά με παραμέτρους που επιλέχθηκαν και ο κώδικας για την ανάλυση είναι διαθέσιμα διαδικτυακά στο: https://github.com/NikiKou/deep_radiotranscriptomics_survival_analysis.

**Λέξεις κλειδιά**

Μη-μικροκυτταρικός καρκίνος του πνεύμονα, ραδιωμική, μεταγραφωματική, ανάλυση επιβίωσης, βαθιά μάθηση, μηχανική μάθηση

# Table of contents

## Table of Contents

# List of figures

# List of tables

# Chapter 1: Introduction

## Lung cancer: principles, diagnosis, and treatment

Cancer is a major public health problem with about 19.3 million new cancer cases occurred in 2020. Female breast cancer has surpassed lung cancer as the most common diagnosed cancer, followed by lung cancer [1]. In men, lung cancer and prostate cancer are the first and the second more frequently diagnosed cancer, respectively. According to the International Agency for Research on Cancer of the World Health Organization (WHO), lung cancer is the leading cause of cancer death worldwide [2]. The total number of new cases and new deaths for most cancers in 2020 in a global range [1] are depicted in Figure 1 and Figure 2. Smoking is the most common cause of lung cancers with 80% to 90% arising in cigarette smokers. A lifetime smoker has a 20-fold increased risk of developing lung cancer compared to a non-smoker. The pathogenesis of lung cancer involves the exposure of environmental carcinogens and intrinsic factors. Genetic variations and family health history may also be the cause of the disease. Mutations that have frequently been identified in tumors of lung cancer are in the epidermal growth factor receptor (EGFR) gene, which is present in adenocarcinomas [3].

The majority of lung cancer cases are diagnosed after a symptom appears related to primary or metastatic disease. The patient is evaluated when obtaining tissue for histologic diagnosis, determining the stage of the disease based on International TNM staging system, imaging such as computed tomography (CT) etc. To facilitate the prognostic decision and treatment, lung cancer is classified based on the histologic appearance into small cell lung cancer (SCLC) and non-small cell lung cancer (NSCLC). NSCLC are further classified into four major histologic classes, adenocarcinoma, squamous cell carcinoma, small cell carcinoma and large cell carcinoma. Also, adenosquamous carcinoma, carcinoid, and bronchial gland carcinoma are histologic classes of lung cancer with prevalence less than 5%. Adenocarcinomas have prevalence of 40% in the lung tumors and are histologically heterogeneous peripheral masses that metastasize early in the disease course. Squamous cell carcinomas are the second more common with 25% prevalence and are endobronchial masses that are centrally located. Small cell carcinomas are also centrally located and are associated with early extrathoracic metastases, including paraneoplastic syndrome. Lastly, large cell carcinomas are large peripheral masses with early metastases [3], [4].

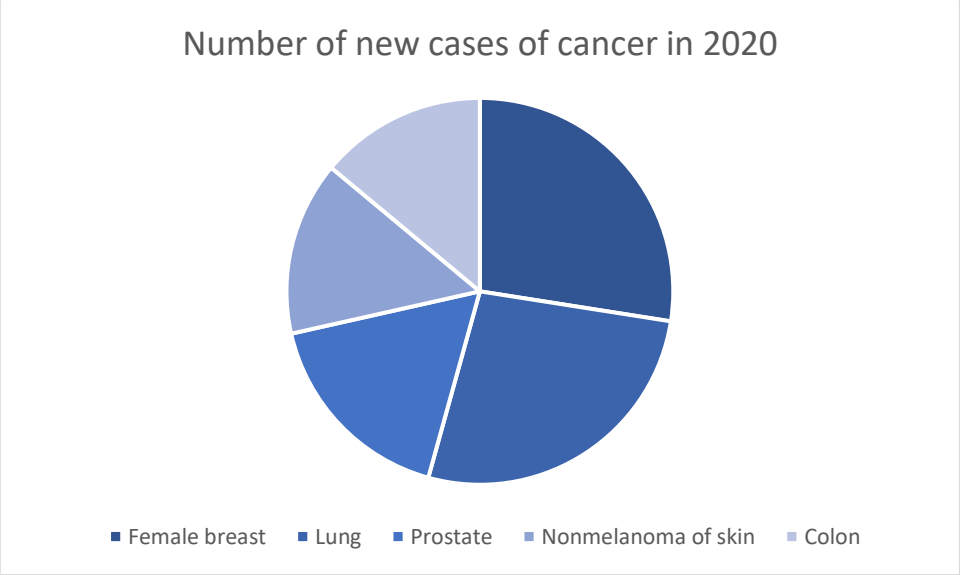


*Figure 1: Number of new cases for different cancers in 2020*

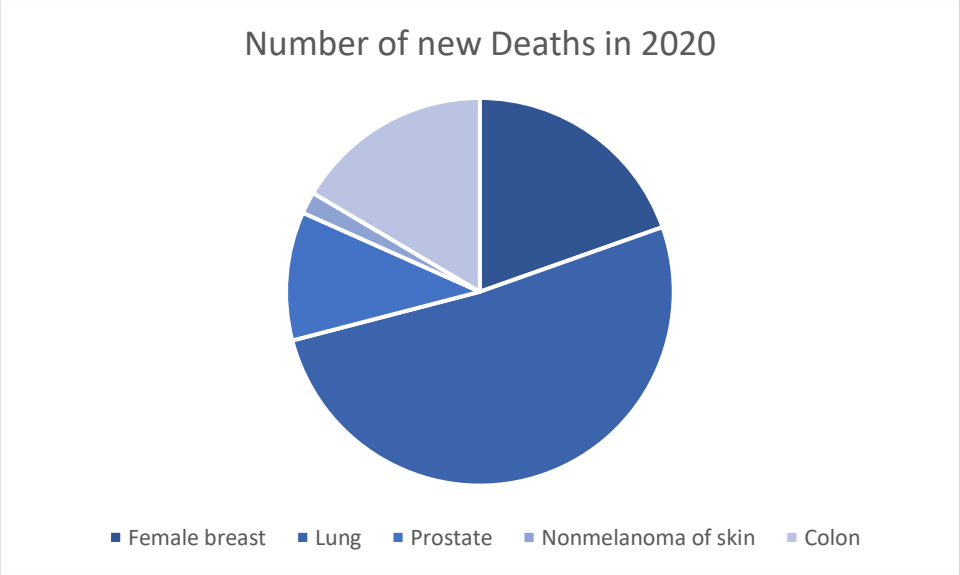


*Figure 2: Number of new deaths for different cancers in 2020*

As already stated, the diagnosis includes tissue diagnosis, staging, and functional evaluation. There is a variety of techniques for tissue diagnosis. In patients with non-small cell carcinomas, thoracotomy

is a convenient and least invasive technique. In general, a least invasive method possible should be used. If obtaining the tissue fails, a more invasive method is needed. Flexible bronchoscopy is another choice for patients with central tumors. Transthoracic needle aspiration appears to be more sensitive than bronchoscopy, especially in patients with peripheral lung tumors. After tissue diagnosis, the next step of cancer evaluation is the clinical staging, which is based on findings obtained before treatment using medical imaging modalities, such as computed tomography (CT) and positron emission tomography (PET). Integrated these modalities, CT/PET scanners sometimes appear to have better characteristics than CT or PET alone. After evaluation of the information obtained, the staging classification can be determined based on the type of tumor identified and the presence or absence of metastatic disease. The last step of diagnosis contains the functional evaluation of the patient. The performance and pulmonary status will determine the treatment options, the therapy option or the probability of surgery [5].

NSCLC accounts for 85% of all cases of lung cancer. The primary curative modality for patients with early-stage NSCLC is surgical resection, with either lobectomy or pneumonectomy, depending on the extent of the disease. For some patients which are not candidates for surgical resection, the treatment may include conventional radiotherapy or adjuvant chemotherapy. Stage III patents will be treated with combined therapy, with concurrent radiotherapy and chemotherapy. Another treatment which might benefit selected patients with advanced NSCLC that have specific mutations is molecular targeted therapy. Patients with limited stage SCLC will be treated with chemoradiotherapy with an intent to be cured, however, chemotherapy can prolong survival also in patients with extensive-stage SCLC [6].

## Artificial Intelligence in Medicine

Artificial Intelligence (AI) refers to a set of technologies that allow machines and computers to simulate human intelligence. AI is broadly used in both the technical and popular lexicon to encompass a spectrum of learning, including machine learning and deep learning. AI has gained wide success in a range of applications, such as speed recognition, computer vision, and natural language processing (NLP). In medicine, the increasing amount of available data in combination with the evolution in automation technology and the rapid development of computer hardware and software has created the ideal conditions for the development of AI systems. AI methods have been developed to analyze health related data, genetic data, as well as clinical and data encompassed in the biomedical literature. Regardless of the technique applied, the scope of AI technologies in medicine is to use advanced algorithms to understand health data, uncover hidden patterns, and assist in clinical decision-making [7].

Both machine learning and deep learning are subsets of AI. Machine learning enables machines to learn from data using statistical methods and to make predictions. Deep learning is a subdivision of machine learning which makes the computation of multilayer neural networks feasible. Advances in

these fields have the potential to revolutionize medicine by performing complex tasks that are currently assigned to specialists. Machine learning algorithms can increase diagnostic accuracy, improve treatment costs, reduce the probability of errors in diagnosis. However, effective use of AI-based technologies in medicine requires synergistic transdisciplinary competencies. For example, personalized care of oncology necessitates the collaboration of many disciplines, such as oncology, radiology, nuclear medicine. A combination of different disciplines can accelerate the effectiveness of AI-based applications [8]. Some of the potential applications of AI-based technologies in healthcare are shown in Figure 3, and involve diagnostics, medical image analysis, therapeutics, population health management, administration and regulation of big data in hospitals [9].

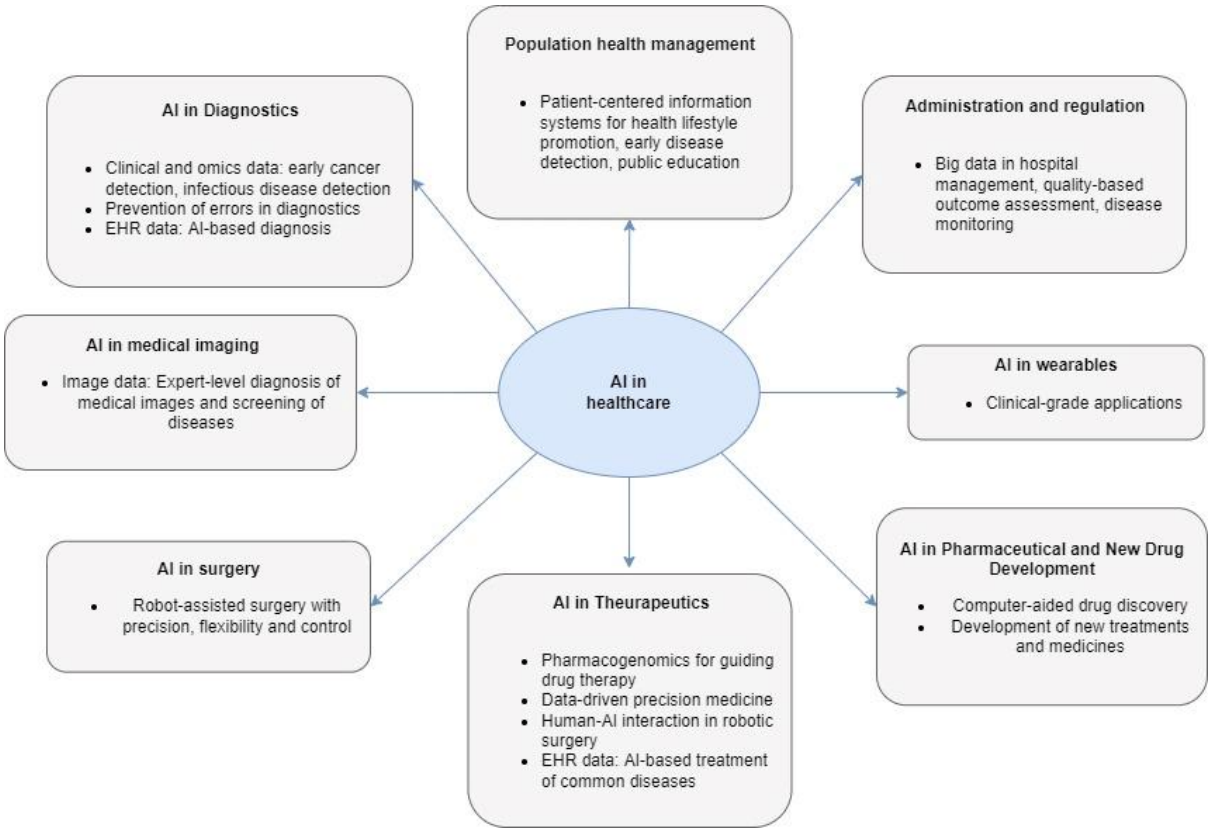


*Figure 3: Applications of AI in healthcare*

In addition to developing AI algorithms for medicine, their practical implementation is extremely difficult. The productization process requires managing and using big amount of data, integration into complex clinical workflows, transparency of the process, and compliance with regulatory frameworks.

Medicine in evolving to a more proactive practice, consisting of four main components, referred to as 4Ps: Predictive, Personalized, Preventive, and Participatory. The main driver of this change is the digitization of medical data, together with the development of new technologies, like machine learning tools, which enable people to interact, analyze, and extract information from these massive volumes of data like never before. Forecasts for the next decade set the bar in the range of billions of data points for each individual patient. As the complexity and dimensionality of these datasets keep growing, it becomes paramount to integrate big data analytics into medical practice and research. It is important to note that collection and analysis of data is not a novel concept in clinical research and medical practice. However, the real value of big data comes from analyzing very large volumes of them and it is the digitization factor in conjunction with the development of these cutting-edge technologies that provide us with new opportunities and demands both in terms of sophistication and depth. Data, collected from electronic health records (EHRs) and precision medicine platforms, such as image, genetic, omics, clinical, and wearable devices data, create large volumes that could be analyzed. For data to be characterized as "big data", several characteristics are required, including the so-called 5 "Vs": volume, velocity, variety, veracity, and value. The volume of data that must be processed by algorithms to be substantially large, in the order of petabytes, data to appear in some context at high velocities, to have a great variety (structured, unstructured data, and different formats), veracity or validity of data (are the data correct, do they have good quality) and finally to have value, meaning that the extracted information to be useful [10]. The use of big data requires planning and careful execution. The security and privacy of data, especially in the field of health care, are of high importance. The privacy of those patients whose data are being managed is critical, and relevant norms and regulations should be applied, such as anonymization and deidentification. With the scale of dissemination, confidentiality and privacy may need to be reimagined entirely. Cyber security measures are increasingly important for addressing the risks of wrong use of data, or inaccurate and inappropriate disclosures. In the data analysis phase, data might be distributed among several nodes. Thus, privacy preservation should be a requirement for the development of the algorithms [9], [10].

Different types of data have been used in AI for health. Most common data types include multi-omics data, clinical data and medical image data. Multi-omics refers to data that belong to the family of "-omics" data, such as genomics, proteomics, transcriptomics and epigenomics. Multi-omics approach joints these data and offer a comprehensive understanding of biological systems. The integration of data provides to machine learning models a multi-view approach, where in conventional single omics approach there is a separate view of machine learning model. The integration of data from single-nucleotide polymorphisms and mRNA gene expression has been used for the prediction of a quantitative phenotype using a Bayesian model. Also, different omics data such as mRNA, gene expression, and methylation have been integrated for identifying associations with clinical outcomes such as ovarian cancer survival. These are some examples that shows the promising results that have been achieved so far. However, there are many challenges regarding AI methods for multi-omics data analysis [11].

Another most common used type of data is medical images. Medical imaging is an important diagnostic tool for various diseases. There are many modalities that have been invented and are used,

including computed tomography (CT), magnetic resonance imaging (MRI), ultrasound (US) and positron emission tomography (PET) among the most common used. These modalities play a vital role in the detection of anatomical or functional information about body organs. Information gained from medical images plays an important role in the patient care process, its characterization, monitoring of the disease, treatment response etc. Medical image analysis aims to help radiologists and clinicians to make diagnosis more efficient. The computer aided diagnosis (CAD) started to develop in the 1980s and the goal was to offer a second opinion to assist radiologists in image interpretation. The first CAD commercial system ready to use was approved by the Food and Drug Administration (FDA) in 1998. CAD systems have been investigated for various applications, including lesion detection, patient's characterization, prognosis prediction. A CAD system is developed with machine learning methods. Conventional machine learning methods have been developed for image analysis to recognize patterns, detect abnormalities, and classify structures on images as normal or abnormal. Image processing and feature extraction techniques have been developed to characterize images. The features extracted can be used as input to a classifier and create a predictive model that can estimate the probability of an image to belong to a normal or abnormal state. Although the research in CAD systems has been increasing, only a few are used in the clinic. CAD systems developed with conventional machine learning techniques may not reach the best performance that is needed by clinicians. The growth of deep learning methods in many scientific areas has pave the way to CAD systems with higher performances that will meet the expectations for implementing CAD in clinical use [12], [13].

Deep learning uses complex multi-layer architectures to learn representations of data and model high level abstractions present in the data. There is a wide variety of deep learning architectures that are used in different applications, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), autoencoders and stacked autoencoders. CNNs are biologically inspired variants of multilayer perceptrons. They have been successfully used for pattern recognition tasks. CNNs look each time at small patches of the input image and use shared weights in each convolutional layer. They learn to extract relevant features from the input image and adjust the weights each time using a backpropagation algorithm. When they perform the convolution operation in each region of the entire image, they eventually create a feature map which provides insight into the internal representations for the specific input. CNNs learn features from underlying data. Their strength in their usage relies on the fact that the loss function provides an error signal that is backpropagated to the networks to improve the feature extraction and as a result, CNNs provide better representation of the input image. CAD systems have been advantages by the use of deep learning. The most common applications of deep learning in CAD systems are the classification of benign and malignant lesions in medical images, the classification of normal or disease patterns, the prediction of high risk and low risks patterns for prognosis of a disease, and the stratification of patients into high risk or low risk patients. CAD systems developed with deep learning algorithms also include image segmentation and classification of tumors or organs, feature extraction of tumor size or texture to characterize the patient or compare features of normal and abnormal cases to assess the treatment response or predict the recurrence of the disease. The large datasets that are available for chest CT, chest radiographs, and mammograms has led research to conduct several studies regarding lung cancer and breast cancer with the use of these datasets [12], [13].

## NSCLC radiotranscriptomics

Lung cancer is responsible for a large percentage of cancer-related deaths worldwide. The early diagnosis of the disease can improve the prognosis and increase the survival rate. Common imaging interpretation, for instance CT, MRI, or PET relies on the visual analysis in terms of shape, size, contrast enhancement of various regions of interest within the image. Radiomics involves the high-throughput extraction and analysis of large amounts of quantitative imaging features with the intent to create mineable databases from radiological images. Radiomics, with reference to genomics, was first introduced by Lambin et al. in 2012 [14]. Like other high-throughput techniques, labeled "-omics", radiomics targets on developing new imaging biomarkers to better understand the microbiology of cancer. The use of radiomics could provide further information about the biological constitution of a tissue or offer prognostic markers. Radiomics can be applied to any cancer-related imaging because it contains many potential information from medical images. Cancer imaging can be explored using radiomics and can be converted into high dimensional quantitative tumor intensity, shape, and text features. In oncological studies, extracted from tumor features that are obtained from radiological data (CT, PET scans) can be used to reveal diagnostic or prognostic associations in patients with correlation to other criteria like response to treatment or survival [15].

The process of radiomics involves the conversion of medical images into quantitative features and occurs through five steps, i) image acquisition and reconstruction, ii) segmentation of region of interest and rendering, iii) Feature extraction and feature qualification, iv) data sharing and building databases and lastly v) building predictive and prognostic models. The extracted features utilized in radiomic analysis refer to algorithms that can be used to describe regions within a radiologic image. Intensity-based, structural, texture-based and wavelet as the basic classes of algorithms that have been commonly used for this purpose. After feature extraction, several statistical models can be used to select the top features that correlate with the outcome of the study. The optimal features can then be used to build a classifier. Using a set of labeled instances, the classifier is trained to predict the outcome of instances in the unseen dataset with unknown labels. These labels may represent malignant or benign characterization of tumor, in case of diagnosis, or low-risk and high-risk patients in case of survival analysis. A range of classifiers can be used for prediction, including random forest, support vector machines, neural networks etc. [15].

The effectiveness of radiomics is based on the hypothesis that medical image analysis can quantify the underlying diseases. Despite the development of multi-modal imaging technologies and computer-aided diagnostic tools, imaging information at the tissue level may not be correlated with the underlying molecular and genetic disease biomarkers. For this reason, the joint effect of multi-scale pathophysiological disease biomarkers may help get closer to the vision of precision medicine. Radiogenomics/radiotranscriptomics analysis involves the combination of radiomic and genomic or transcriptomic information and aims to merge the imaging phenotype with the underlying molecular characteristics of a disease. "-Omics" data, including genomics and transcriptomics have been

increased with the advances in next-generation sequencing. Genomics study the structure, function, and inheritance of the genome, with a major part of it being the determination of the sequence of molecules that make up the genomic DNA content of an organism. Transcriptomics technologies are used to study an organism's transcriptome, the sum of all its RNA transcripts. Transcriptomics focuses on the gene expression at the RNA level and offers the genome-wide information of gene structure and gene function in order to reveal the mechanisms behind specific biological processes. The combination of radiomics and transcriptomic features aims to correlate the imaging and gene expression information and increase the predictive power of predictive models. Several studies have tried to associate image-derived features with molecular information, where both imaging (CT, MRI, PET/CT) and molecular or genetic information (miRNA, RNA-seq, DNA) are available [16], [17].

NSCLC radiogenomics and radiotranscriptomics analysis in literature mostly focuses to find the association between radiomic features and genomic signatures. Morgado et al. [18] investigated the relation between image phenotypes and the mutation status of Epidermal Growth Factor Receptor (EGFR) using radiomic features extracted from CT scans. Using linear Support Vector Machine, Elastic Net and Logistic Regression, authors showed that a comprehensive approach using a region of interest that include the lung with nodule could successfully predict the EGFR mutation status. Similarly, EGFR co-mutated with TP53 status was identified using a CT-derived radiomics approach. Zhu et al. [19] developed and validated a multiclass classification strategy to predict primary overlapping mutations involving TP53 and EGFR in advanced lung adenocarcinomas. The model could potentially be used as an important alternative marker for selecting the best responders to target therapy, since EGFR co-mutated with TP53 NSCLC patients could reduce responsiveness to treatment with tyrosine kinase inhibitors. However, these studies used radiomic features and correlated the genomic signatures. There are only few studies that combined radiomic and transcriptomic features [20], [21].

## Objectives of the study

In this study, a multi-view survival analysis will be developed to investigate the combination of radiomics, transcriptomics and deep features extracted from CT scans of NSCLC patients. Two classifiers will be examined for their ability to classify NSCLC to low- and high-risk patients, Random Forest, and Support Vector Machine. In Chapter 2, a review of studies about survival analysis of NSCLC patients is presenting. Survival analysis for lung cancer has been extensively studied over the years, using solely one type of features. Only a handful of studies have combined different features, such as radiomics and transcriptomics. This study aims to combine into one feature space, radiomics, transcriptomics and deep features and assess the predictive power of the developed models.

The analysis code was developed in Python programming language and packages from the scikit-survival library were used. The code is freely available on GitHub: https://github.com/NikiKou/deep_radiotranscriptomics_survival_analysis and in the Appendix. A part of this Thesis will be included in the survival section of the publication entitled: "Deep Radiotranscriptomics of Non-Small Cell Lung Carcinoma for Assessing High-Level Clinical Outcomes using Multi-View Analysis" conduced by Trivizakis Eleftherios, Koutroumpa Nikoletta-

Maria, Souglakos John, Karantanas Apostolos, Zervakis Michalis E., Marias Kostas. The publication is under peer-review for publishing in a journal. This Thesis differs from the publication in feature selection and feature integration parts. This study integreates radiomics, transcriptomics and deep features whereas publication takes into consideration also clinical data. Furthermore, more classifiers than random forest and support vector machine are examined in the publication.

# Chapter 2: State-of-the-art review

Cancer mortality remains significant despite advances in medicine, including personalized treatment strategies, combining surgery, chemotherapy, immunotherapy, and radiotherapy. Most patients with NSCLC are characterized with poor median overall survival. It is of high importance to discover and validate biomarkers that can predict outcomes and be sensitive to different treatment effects. Outcome modeling can enable the identification of prognostic signature and a risk stratification of patients with different cancer therapies. Accurate prediction of survival usually depends on multiple information including histologic, genetic, imaging and molecular information [22].

## NSCLC Survival Analysis using radiomics

Many studies found in the literature are focused on predicting the survival rate or survival time of a group of patients with NSCLC. Radiomics studies have shown promising results to decode the intra-tumoral heterogeneity and predict the progression of a disease and the therapy response of the patients. The radiomic features extracted from computed tomography (CT) and positron emission tomography (PET), as will be described below, were used to predict prognosis. However, different modalities reflect different aspect of tumor heterogeneity. By integrating the information of imaging modalities some studies ended up in improved prognostic values. Chaddad et al. [23] investigated the prediction of NSCLC patient survival outcomes based on radiomic texture and shape features. CT scans of 315 patients were retrieved and features automatically extracted from tumor image were computed. The gross tumor volume was computed for each scan and assigned to different NSCLC subtypes (i.e. large cell carcinoma, adenocarcinoma, squamous cell carcinoma or not otherwise specified). A total of 24 features were computed and used in combination with other information of patients to analyse their survival. The correlation between the survival time and the features was measured with Spearman's rank correlation, whereas Kaplan-Meier estimator and log-rank tests were developed to select features that were more related to patient survival outcomes. After feature selection, random forest algorithm was used to predict the patient's survival group. More specifically, a multivariate analysis using 24 radiomic features and 5 staging/demographic variables such as age, T, N, M and overall stage, were used as input to a random forest classifier, to classify the patients in two subgroups, one group with long survival time (above the median survival time) and one with low survival time (below the median survival time). The importance of each feature was also assessed. Correlations between radiomic features and survival led to the conclusion that CT imaging features, especially for patients with large cell carcinoma, primary tumor size and no lymph node metastasis, could be used as indicators of survival with accuracy at the range of 72%.

Another study conducted by Shen et al. [24] compared the 2D and 3D radiomic features prognostic performance differences of NSCLC. A collection of 588 NSCLC patients' pre-treatment CT images were used in this study. All patients' survival data were dichotomized by the cut-off of 2 years, where 1 indicated patients with larger than 2 years survival time and 0, patients with less than 2 years. Tumors' contours were segmented semi-automatically. Semi-automatically segmentation referred to radiologists identifying a proper point inside the lesion and after Toboggan based growing algorithm automatically segmenting lung lesion. A total of 1014 radiomics features were assessed, 507 2D features and 507 3D features. Both feature groups involved certain categories, first-order histogram statistics, Gray-Level Co-occurrence Matrix, Gray-Level Run-length Matrix and Fractal Dimension. Authors employed the univariate Cox regression model to achieve each feature's Harrell's concordance index (C-index) and those with higher C-index were considered with potential prognostic power. The stable and potential prognostic features were selected to construct the 2D group's and 3D group's indicators of classification. The prediction performance of the logistic classifier showed that 3D group's AUC was larger than the 2D's in the training cohort, but the 2D's was better in the validation cohort. Classified binary indicators were associated with censored continuous survival data for the survival analysis. Both 2D and 3D indicators achieved good results in the Kaplan-Meier analysis and C-index of the 2D model was higher than the 3D model. In conclusion, considering the cost of radiomic feature calculation and the better performance of 2D features, Shen et al. resulted in recommending 2D features for use in practical research.

The combination of the radiomics signatures extracted from 2D and 3D CT images was also studied [25]. The main purpose of the study was to develop a radiomics nomogram for the prediction of the survival of patients with NSCLC. A total of 975 features were extracted from 371 CT images. Using the LASSO regression model, the candidate features from 2D were reduced to ten variables and the 3D features were reduced to nine variables. To build the optimal radiomics signature, authors compared the prognosis performance of 2D and 3D feature groups. As discussed in the previous study, univariate Cox proportional hazards model was used and C-index and hazard ratio were calculated to evaluate the predictive accuracy of 2D, 3D or 2D plus 3D radiomics signature. The C-index of 3D was greater than of 2D, however both had a favorable predictive power for survival. The Kaplan-Meier analysis demonstrated that the combined 2D and 3D features could more effectively distinguish the patients into low-risk and high-risk groups. The new radiomics signature was built using the combined feature vector containing different feature groups. Finally, they evaluated the prognosis ability of the clinical TNM staging and radiomics nomogram. When the radiomics signature was combined with the TNM staging system and clinical information, the predictive power was significantly improved, showing that radiomics can improve the predictive accuracy of patients' survival. The conclusion of the study that with combined 2D and 3D features the predictive power was improved and that could be verified on the reason that 2D image under segmenting mainly contains the central region of the tumor. However, the tumor heterogeneity, is the outer structure of the tumor tissue and 3D image carries information on the peripheral surface of the tumor.

Another study focused on the quantitative assessment of heterogeneity by histogram analysis of tumor images. Bluthgen et al. [26] aimed to determine the potential of imaging analysis as an

independent predictor of survival and the potential of associating tumor heterogeneity with gene alterations. CT scans from 2009 till 2015 from 692 patients were reviewed, together with clinical and molecular data (KRAS, EGFR and ALK status) and 421 of them were used for histogram analysis. The prognostic value of sex, age of inclusion, smoker status, size tumor and CT histogram analysis parameters were measured in a univariate analysis. Multivariate analysis was performed by Cox regression models and Wilcoxon test was used to correlate the CT histogram analysis parameters with mutational status. In the training dataset of 313 patients, primary mass entropy was strongly associated with overall survival in the univariate analysis and remained an independent prognostic factor in a multivariate analysis. However, this result wasn't reproducible in the validation patient cohort.

De Jong et al. [27] investigated the prognostic value of the signature in two cohorts of stage IV NSCLC patients, EGFR and ALK wildtype, from 195 patients with their CT scans. Radiomic features were calculated for the primary tumor and the C-index of Cox regression model was calculated and compared to the signature performance of overall survival. The same group some years before published a prognostic radiomic signature for overall survival, consisting of 4 radiomic features: "first order statistics: Energy", which described the overall density of the tumor volume, "Shape: Compactness", a feature for quantifying the compactness of the tumor volume relative to the compactness of a sphere, "Gray level run length: Gray level non-uniformity" to measure intra-tumor heterogeneity and lastly "Wavelet Gray level run length: gray level non-uniformity" to describe the intra-tumor heterogeneity after wavelet decomposition of CT scan. The signature was trained on stage I-III NSCLC and in a second phase was used in stage IV patients. This study showed that stage IV patients with prognostic index, which was calculated for the radiomic signature, lower than the signature median had better overall survival compared to patients with higher prognostic index.

Another study by Zhang et al. [28] presented different strategies for improving predictive performance of radiomics-based prognosis for NSCLC. Also, these strategies were used to overcome some challenges such as unbalanced data, small sample sized and feature redundancy, which lead to low predictive accuracy. CT images of 112 patients with NSCLC were used to predict recurrence, death, and recurrence-free survival using a radiomics analysis. Initially, a large number of quantitative features were extracted from medical images, and 5 unsupervised feature reduction methods were investigated, including Principal Component Analysis, Independent Component Analysis, Zero Variance and Near Zero Variance. Eight classification algorithms were used to predict the endpoint event. Some of the algorithms examined were Random Forest, K-nearest Neighbour, Generalized linear model, Support Vector Machine, Neural networks. PCA showed the best performance in reducing the number of features, highlighting that unsupervised feature reduction methods maintain the interaction among features, benefiting the predictive model training process. Comparing the classifiers, random forest resulted in the highest predictive value. Down-sampling, up-sample and Synthetic Minority Over-sampling techniques were also applied to tackle the unbalanced data. Finally, an analysis of variance suggested that feature selection methods, data endpoints and classification models significantly affected the predictive accuracy, indicating that these factors should be investigated when building a cancer prognosis model.

The CT-based radiomic signature was used by Li et al. [29] for predicting progression-free survival in stage IV ALK-positive NSCLC patients. A total of 63 stage IV ALK-positive NSCLC patients who had received TKI crizotinib therapy and 105 stage IV EGFR-positive NSCLC patients were used for model construction and validation, respectively. The progression-free survival was counted from the start of treatment with TKI to the confirmation of disease progression or death. Other clinical characteristics, such as sex, age and smoking status were also recorded. Authors initially extracted 481 quantitative 3D features from CT scans of the patients from manually segmented tumor volumes of interest. Next step for Li et al. was to perform Pearson correlation analysis and the least absolute shrinkage and selection operator (LASSO) penalized Cox proportional hazards regression to reduce the number of radiomic features and select the critical ones. They assessed the potential association between the radiomic signature and progressive-free survival by Kaplan-Meier survival curves as well as log-rank tests. Results showed that the CT-based radiomic features could capture important information regarding the tumor phenotype. The prognostic performance for ALK-positive NSCLC patients in both training and validation cohort reached C-index to 0.74 and 0.72 respectively. The radiomic signature managed to stratify the patients into slow progression and rapid progression disease. Adding the clinical characteristics (sex, age and smoking status) did not benefit the model, indicating that radiomic signature alone could predict efficiently the prognosis in ALK-positive NSCLC patients treated with TKI crizotinib.

The early prediction of the tumor response of SCLC patients to chemotherapy with the use of CT-based radiomics signature was examined by Wei et al [30]. A total of 92 patients who received the standard first-line regimen of etoposide and cisplatin, were divided into two groups: response and no response patients. These patients also underwent CT examination and a total of 21 radiomics features were extracted from CT scans that conducted prior to and after two cycles of chemotherapy. Researchers established a predictive model using a binary logistic regression model. The results of the study showed that the performance of the radiomics signature to predict the chemotherapy efficacy were higher than the conventional model that used clinicopathological parameters. The outcome of this study was that radiomics models could effectively predict the therapy response.

Radiomics analysis of tumors of NSCLC patients has been widely used on single images modalities, and especially in CT scans, as it was presented previously. Integrating information from different imaging modalities may provide different characteristics on tumor heterogeneity. Amini et al. [31] compared the prognostic value of multi-modality multi-level fusion radiomics models. CT images and $^{18}$F-FDG PET images of 182 patients were collected from The Cancer Imaging Archive (TCIA) and were used for this study. Single-modality models were constructed initially, and then PET and CT information were integrated using image-and feature-level fusions to construct the multi-modality models. Cox proportional hazard regression was used for survival analysis. Image-level fusion was performed with feature extraction from fused PET and CT scans using to wavelet-based technique. The wavelet fusion outperformed other models resulting to C-index=0.71. In feature-level fusion, features were extracted from separate PET and CT scans and then two different strategies were developed, feature concatenation (ConFea) and feature averaging (AvgFea). Both approaches resulted in lower C-indices (0.58 and 0.64, respectively). Amini et al. concluded that multi-modality models showed increased

prognostic value compared to single-modality models. Also, when combining information obtained from CT and PET scans, image-level fusion showed superiority in predicting compared to feature-level fusion. One year later, the same group of researchers showed that image-level fusion multi-modality radiomics models outperformed clinical models as well [32].

The combination of different imaging modalities was also examined by Forouzannezhad et al [33]. In this study, they collected the survival outcomes for 45 patients with unresectable NSCLC enrolled on the FLARE-RT phase II trial of risk-adaptive chemoradiation. CT scans, FDG-PET and perfusion SPECT imaging before treatment and after treatment were performed. Shape, intensity, and texture-based features were extracted from the metabolic tumor volume resulting in 110 total features. Authors applied a multitask learning approach for prediction of overall survival, that was consisted of a fused Laplacian sparse group LASSO with component-wise gradient boosting survival regression. The study demonstrated that FDG-PET radiomics had the higher prognostic value with C-index equal to 0.71 compared to CT radiomics and SPECT radiomics with C-index 0.64 and 0.6, respectively. Multitask learning of FDG-PET radiomics outperformed also clinical imaging and conventional FDG-PET delta radiomics. The studies presented in this chapter highlighted the potential of radiomics models, especially multi-modality models to improve prognosis for NSCLC patients.

## NSCLC Survival Analysis using transcriptomics

Advances in genomics and the advent of next-generation sequencing has established massive genomic approaches. RNA sequencing for gene expression has created paradigm shifts in a variety of research fields. Gene and network expression signatures have been successfully used to predict cancer patients' survival. Different studies have shown that microarray measurements of gene expression whether alone or combined with clinical information could predict overall survival in lung adenocarcinoma [34], [35]. Gene expression signatures were also used to predict response to therapy. Zhu et al. [36] presented the first NSCLC prognostic gene expression signature generated from microarray studies using samples collected prospectively in a randomized phase III adjuvant cisplatin/vinorelbine trial. A 15-gene signature separated patients into high-risk and low-risk groups with significantly different survival. The interaction between risk groups and adjuvant cisplatin/vinorelbine was verified by quantitative polymerase chain reaction. They found that patients whose tumors were predicted to have a poor prognosis but who received chemotherapy, exhibited significantly better survival than the observed patients whose tumors had a poor prognosis signature and did not receive chemotherapy. In contrast, when the patient's tumor had a good prognosis signature and chemotherapy was administered, the patient did worse than patients with a good signature with no chemotherapy. The study showed that this gene expression signature could be a prognostic marker of NSCLC in early stage. For patients with poor prognoses, the tumor biomarker information suggested that they needed additional therapy, and that were likely to gain survival benefit from adjuvant cisplatin/vinorelbine.

Another study by Watza et al. [37] hypothesized that NSCLC-specific mechanisms of immune modulation could be detected by leveraging tumor transcriptomic profiling paired with patient outcome data and could identify the immune-centric gene networks that impact patients prognosis. Tumor transcriptomes and clinical characteristics were obtained from two distinct NSCLC cohorts. To test the immune-centric gene and pathway expression for association with patient survival, they constructed a parsimonious base survival model. Prognosis-guided gene and pathway analysis showed significant survival enrichments and the interleukin signaling pathway, was found to be enriched with prognostic signal and have the greatest impact on overall survival. From 430 genes in interleukin signaling pathway, subsequent leading-edge analysis identified 23 genes that were in both cohorts. These gene-pathway candidates were proposed as future targets for therapeutic and mechanistic studies to advance immunotherapy. Another prognostic model based on immune-related genes of lung squamous cell carcinoma was developed by Rui et al [38]. They performed univariate and multivariate Cox regression analysis to construct the differentially expressed immune-related genes (DEIRGs) to predict survival. The p-value between low- and high-risk subgroups was zero, indicating that the prediction model could accurately estimate lung squamous cell carcinoma prognosis. The relationship between prognostic model and immunocytes was further explored through immunocyte correlation analysis. They also performed immunocyte infiltration analysis, showing that dendritic cells and neutrophils were positively correlated with immune-related genes and played an important role within the immune microenvironment. The relationship of the transcriptional tumor immune microenvironment with prognosis of patients with lung adenocarcinoma was also examined by Chen et al [39]. They used gene set variation analysis to identify gene sets related to prognosis starting from 85 locally advanced lung adenocarcinoma samples. To quantify infiltrated immune cells, researchers employed the microenvironment cell-population counter method. Survival analysis with the log rank test demonstrated that antigen processing pathway enrichment was associated with better prognosis. Also, Cox proportional hazards models were used to identify risk factors and greater infiltration of neutrophils was identified as an independent risk factor for poor diagnosis.

## NSCLC Survival Analysis using deep features

Identifying predictive features from medical images of patients is the key concern for tumor classification and prognosis analysis. Traditional quantitative features, such as radiomics, have been successfully used for this purpose [23], [24], [27], [28]. However, recent studies use features extracted from a deep neural network to characterize cancers, which showed good classification performance. Advances in artificial neural networks (ANNs) and especially in convolutional neural networks (CNNs) have created a new way for extracting features of medical images and which could be used in different tasks. A study that explored deep learning applications in medical imaging for patient stratification was conducted by Hosny et al. [40]. Seven independent datasets across five institutions containing 1194 patients with NSCLC images with CT and treated with either radiotherapy or surgery was collected for analysis. For the patients treated with radiotherapy, a 3D CNN was trained end-to-end. The same network was used for the surgery dataset, using a transfer learning approach between

radiotherapy and surgery datasets. The deep learning networks were used for extracting imaging features and creating prognostic signatures for patients. These prognostic signatures were assessed on their ability to stratify patients into low and high mortality risk subgroups. Also, Kaplan-Meier curve analysis was used to evaluate the performance of network to stratify low and high-risk patients. CNN showed a significant prognostic power in predicting 2-year survival and outperformed models based on predefined tumor features. Hosny et al. finally identified regions withing and beyond the tumor, which had the largest contribution to the prognostic signature and the most contribution towards predictions, highlighting the importance of tumor-surrounding tissue in patient stratification.

A multimodal deep learning framework for NSCLC survival analysis, named DeepMMSA was proposed by Wu et al. [41]. DeepMMSA consisted of three modules, multimodal feature extraction (features from CT images and clinical records), multimodal feature fusion, and survival analysis. A group of 422 NSCLC patients from The Cancer Imaging Archive were considered to assess the framework. For these patients, pretreatment CT scans and clinical outcome data were available for analysis. Using a 3D-ResNet as network structure and CT images as inputs, image features were extracted. Authors also used clinical information, such as clinical TNM stage, histology, gender, age of patient. These features were then fused and used for survival analysis module. Results shown that there was a relationship between prognostic information and radiomic images. The majority of existing models could be employed for survival analysis, such as Kaplan-Meier model, Cox regression model, machine learning methods and deep-learning based methods. All of these methods could be used to analyze the input from multimodal features in the fusion layer.

The efficacy immune checkpoint inhibitor monotherapy was evaluated by He et al. [42] with the aid of CT images combined with deep learning. A group of patients who received anti-PD-1/PD-L1 monotherapy for advanced NSCLC and had undergone CT scans was included in this study. Progression-free survival (PFS) was calculated as the time from the start of immunotherapy till tumor progression or death, and the overall survival as the time from diagnosis till last follow-up or death. A survival network was developed during this study to obtain a risk vector of PFS risk and OS for patients through 3D tumor imaging. Then, the risk vectors were combined to explore more in depth the patient prognosis. At the same time, they constructed dual-task network to obtain the progressive disease score and partial response score of patients. The vectors obtained from the networks were fitted using the Cox regression model, in order to calculate the OS risk score and PFS risk score.

## NSCLC Survival Analysis with feature fusion

Many genomic biomarkers, such as DNA polymorphisms and RNA expression levels, have emerged in recent years of genomics. These biomarkers have been used for the diagnosis and the prognosis of different cancers, including NSCLC. Furthermore, with the advancement of image-processing technologies, such as MRI, CT, PET, PET-CT etc., and the extraction of quantitative imaging features from medical images, radiomics has emerged. “Radiogenomics”, a combination of radiomics and

genomics have been applied for both radiotherapy response and prognostic implications in patients with NSCLC [15]. Radiomics has already shown promise for predicting diagnosis, prognosis and response to therapy in lung cancer, with radiogenomics bridging the gap between computer-aided prognosis and personalized medicine. “Radiotranscriptomics” is a branch of radiogenomics, which combines radiomics with transcriptomics. However, it has not yet been fully explored the combination between mRNA, miRNA, expression levels and medical image features in NSCLC. A radiotranscriptomics study for prediction of radiotherapy response was conducted by Fan et al. [21]. They established radioresistant NSCLC cell lines to extract cell miRNA and performed microarray analysis. At the same time, 174 NSCLC patients were CT scanned. The radiomics texture features were extracted from their images and their miRNA serum was also obtained. After obtaining the optimal CT texture features, the LASSO model was used for feature selection and the features were combined to generate the radiotranscriptomic signatures. These signatures were used to predict the objective response rate, OS and PFS using logistic and Cox regression. The conclusion of this study was that the radiotranscriptomic signature could be an independent biomarker that could predict the radiotherapy response of NSCLC patients.

The combination of deep features with radiomics was explored by Paul et al. [43]. In this study, they used a transfer learning concept to predict the survival of patients with non-small cell adenocarcinoma lung cancer. They collected 40 lung cancer cases with the CT images and applied pretrained CNNs on ImageNet to extract deep features. Three different pretrained CNN models described in Chatfield et al.’s work were used (VGG-F architecture, VGG-M architecture, and VGG-S architecture) [44]. Relief-f, a simple selection algorithm for finding features with strong class dependencies and symmetric uncertainty feature selector, an algorithm that ranks the features by calculating the fitness between the features and the classes, were experimented in this study for selecting features. Also, for classification of patients to short-term and long-term survivors, four classifiers (Naïve Bayes, Nearest Neighbors, Decision Trees and Random Forests) were implemented. When they used traditional features for patients’ classification, the best accuracy of a decision tree classifier was 77.5%. With the use of deep features and a decision tree classifier, the accuracy was the same. However, with the combination of traditional quantitative features and the extracted deep neural network features, Paul et al. reached an accuracy of 90%. With single-slice approach, meaning by merging deep and traditional quantitative features extracted from single-tumor slice, the 90% accuracy was obtained using the VGG-F CNN architecture with the random forest classifier in a leave-one-out-cross validation with 10 features. With multiple-slice approach, the best accuracy was achieved using the random forest classifier, the symmetric uncertainty feature ranking algorithm and combining the best five features extracted from the VGG-F architecture and the five best traditional features.

# Chapter 3: Research methodology

In this study, three different classes of data, radiomics, transcriptomics and deep features will be examined in their ability to predict the survival of NSCLC patients. The fusion strategy includes combining all features to perform survival analysis and also combine only radiomics and deep features. The collected data will be pre-processed and after feature selection, the data will be used for the analysis. Two classifiers will be examined, Random Survival Forest and Survival Support Vector Machine. The workflow of this study is presented in Figure 4.

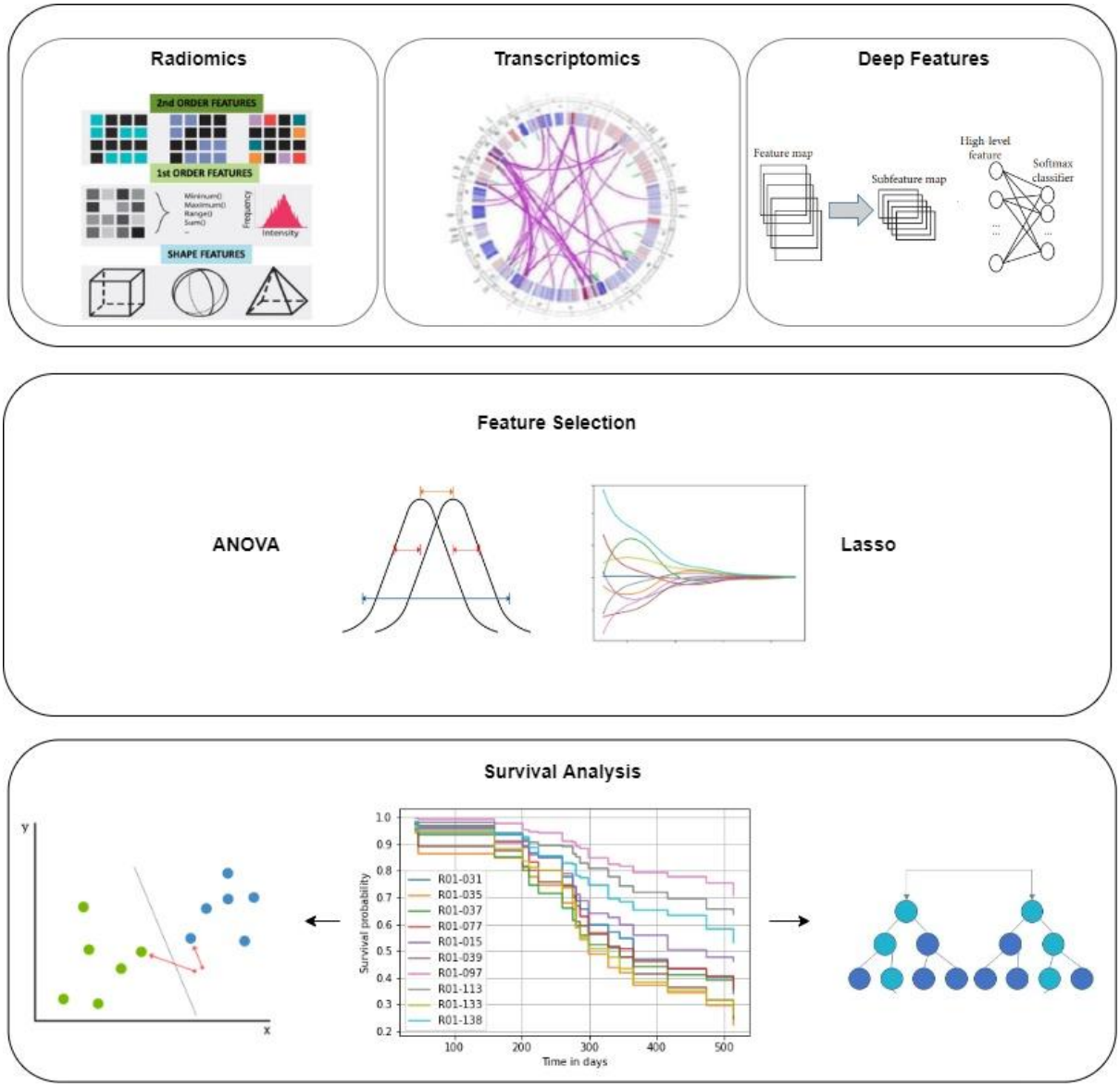


*Figure 4: Flow diagram of the proposed survival analysis*

## Dataset

The data used in this study was obtained from the NSCLC Radiogenomics dataset [45]. The dataset was created using clinical and imaging data of 211 subjects which were collected between 2008 and 2012. Subjects received preoperative CT and PET/CT scans at Stanford University Medical Centre and Palo Alto Veterans Affairs Healthcare System prior to survival treatment. The dataset comprises 211 Computer Tomography (CT) examinations, with 142 available ePad pixel-based lesion annotations [46] and 211 image markup standards [47]. There are additionally Fasting Fluorodeoxyglucose $^{18}$F-FDG PET/CT data available for 162 subjects, 130 RNA-seq vectors ($P_G$) and clinical data with histology, genomic, semantic, survival and disease recurrence information. The dataset is available at Cancer Imaging Archive [48].

From the total examinations, a subset of 142 CT has annotations on a pixel basis for the region of interest ($P_{ROI}$). The intersection of the transcriptomic and imaging data, dented by $P_T \cap P_{ROI}$, gives the subset $P_{RG}$ of 115 patients. The clinical data includes patients with characterization such as "Survival Status", either being alive or dead and "Time to Death" counted in days. The patient's labels were 0 when the time of survival was more than the median time of survival, indicating low risk patients and 1, when time of survival was less than the median time of survival, indicating high risk patients.

$$L_{SURVIVAL} = P_{ROI} \cap P_{TIME\ TO\ DEATH}$$

$$L_{SURVIVAL} = \begin{cases} 0\ if\ P_{TIME\ TO\ DEATH} > median(P_{TIME\ TO\ DEATH}) \\ 1\ if\ P_{TIME\ TO\ DEATH} < median(P_{TIME\ TO\ DEATH}) \end{cases} \quad Eq.\ 1$$

The median days of survival were computed to be 704. The patient cohort of survival $P_{SURVIVAL} = L_{SURVIVAL} \cap P_{RG}$) includes 40 subjects and was considered for the proposed analyses. A subset of 23 patients were characterized as high-risk patients and labeled with label 1 and the remaining 17 as low risk patients, with label 0.

## Feature extraction

### Radiomics

The radiomics analysis of the original CT examination retrieved from Trivizakis et al., 2021 [20] and resulted in 2996 imaging features. Texture features were calculated by the PyRadiomics framework version 2.2.0 [49], including Gray-Level Covariance (GLCM), Gray-Level Dependence Matrix (GLDM), Gray-Level Size Zone Matrix (GLSZM), Neighbouring Gray Tone Difference Matrix (NGTDM) etc. These kinds of matrices are methods for describing spatial pixel differences by studying the spatial correlation properties of gray scales and thus are the most capable of expressing the inhibition between different parts of tumor [49]. Shape features and first order features were also calculated, including elongation, flatness, 2D and 3D diameter, minimum, maximum, mean, range, kurtosis, skewness of gray level intensity and other statistical features. Exponential, gradient, Laplacian of Gaussian, square and wavelet filtering were applied to the original image in order to enrich the radiomic analysis. The general process of PyRadiomics platform is shown in Figure 5.

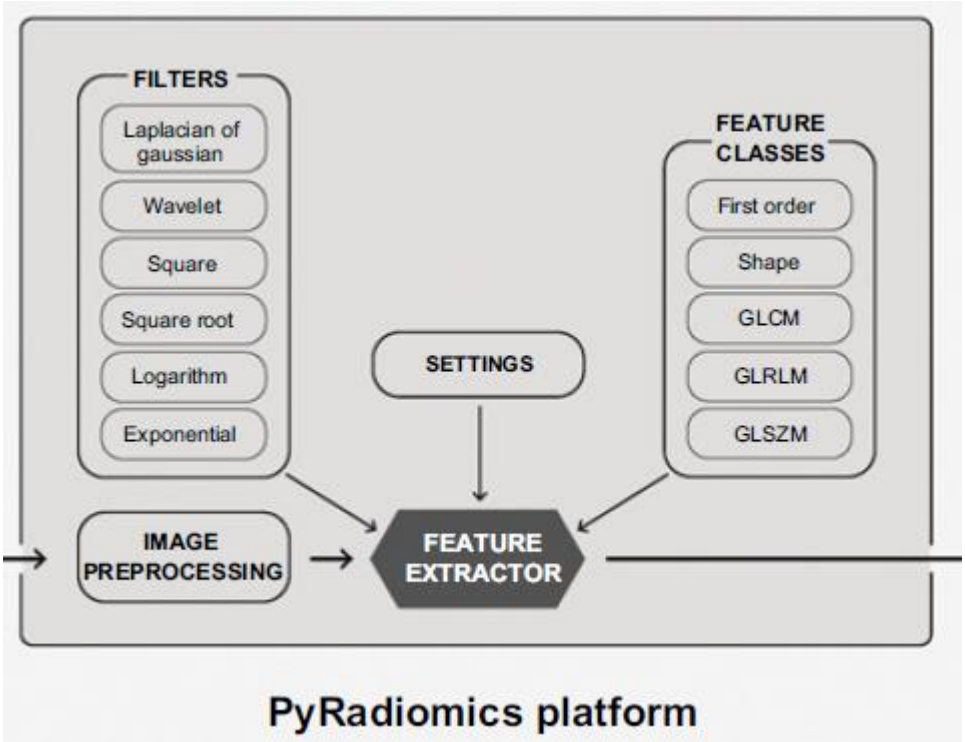


*Figure 5: Overview figure of the process of PyRadiomics [49]*

### Transcriptomics

The RNA-seq data for this study were downloaded from the NCBI GEO hosting database [50]. The RNA was extracted from the tissue and analysed with RNA sequencing technology. A total of 130 RNA-seq vectors were available for transcriptomic analysis. The transcriptomics comprised of 22126 values but after the removal of incomplete features, a transcriptomic signature of 5268 molecules per patient was examined.

### Deep features

Advances in Deep Learning have contributed in identifying, classifying and quantifying patterns in medical images [51]. The success of Convolutional Neural Networks is attributed to their ability to extract highly representative features among the network layers of filtering, and use these features in the last fully connected layers for pattern classification [52]. The deep features were retrieved from Trivizakis et al., 2021 [20]. The proposed "off-the-shelf" Transfer Learning (TL) strategy developed in that study, uses pretrained ImageNet [53] models to extract raw deep features from the last convolutional layer of the source model. Transfer Learning is a machine learning technique whereby a model is trained and developed for one task and is then re-used on a second related task. Usually, these tasks are called source and target tasks, respectively. It is usually applied where there is a new dataset smaller than the original dataset used to train the pre-trained model. The "off-the-shelf" strategy uses features from the source task without re-training the network and use them to train a third-party classify. In another Transfer Learning strategy, called fine-tunning, the internal representation has to be adapted with a new training process, but a part of the whole of the source model is transferred to the new model [54], [55].

Eighteen models were used for the extraction of deep features. Some of the most popular ones were ResNet [56], NasNet [57], DenseNet [58], Xception [59], VGG [60] and MobileNet [61]. All pretrained convolutional layers were transferred to the new model and the fully connected layers were removed in order to extract the deep features from the low-level filters. Different number of features were extracted per slice, depending on the architecture used. After the removal of features with zero variance, the raw features extracted of each model varied from 502 to 7952.

## Multi-learning with combined features

Three different categories of data were considered in this study: (a) radiomics, (b) transcriptomics and (c) deep features. The data sets were combined in order to understand the contribution of each category in the final outcome. Radiomics, transcriptomics and deep features were concatenated into a common feature space prior to classification indicating the first data view which all available data contributed to classification. The second data view was created with the concatenation of radiomics

and deep features into a common feature space. The acquisition of these data categories as shown in Figure 6 The results obtained using these data views are shown in Research findings / results.

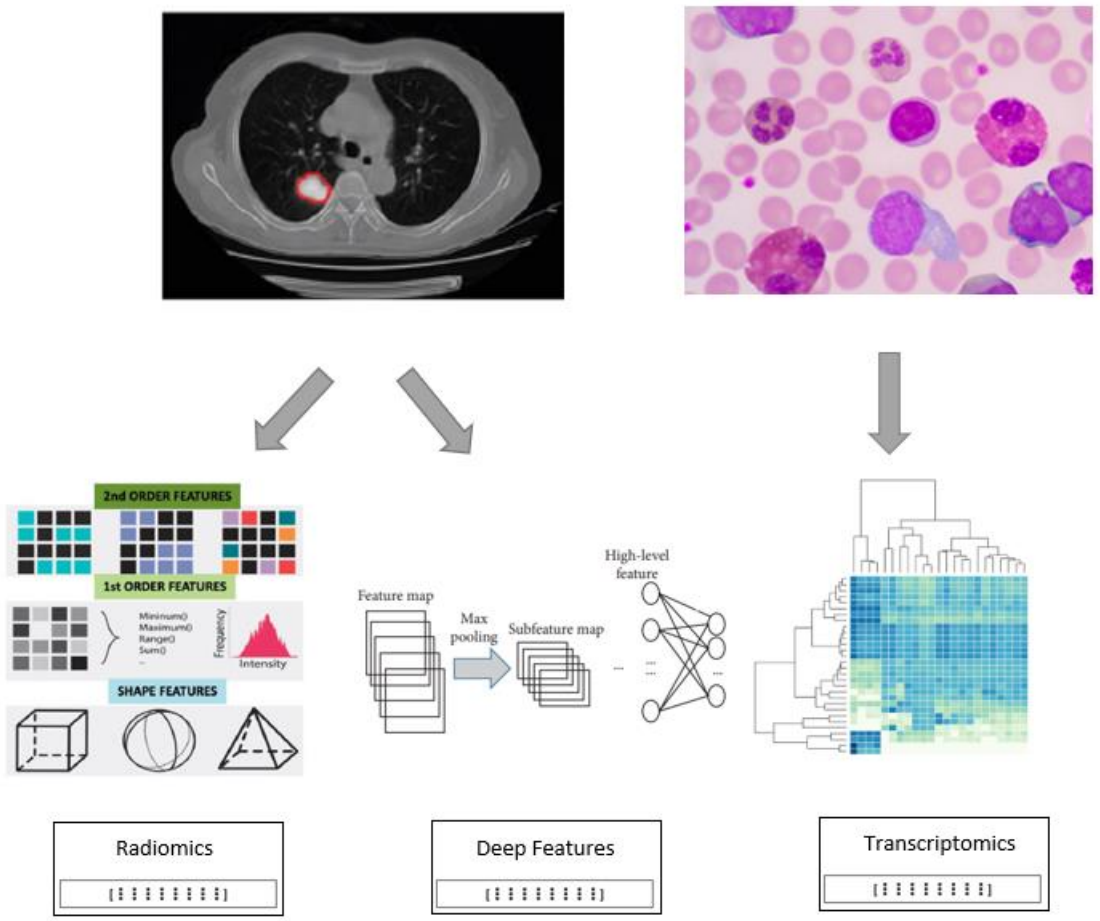


*Figure 6: Features retrieved from CT, PET/CT and RNA-seq data* [62]–[65]

## Data pre-processing

### Data cleaning

Data cleaning is usually the first task in data pre-processing. It is the process of detecting and then removing or correcting corrupt data and refers to identifying incorrect, inaccurate, or incomplete parts of the data [66]. When building a machine learning model, not all the features in the input data are useful to build a model. Some may give no value to the model and should be removed. Removing features whose variance does not meet a specific threshold, is a common approach to feature

selection. The features in this study have been selected with the Variance Threshold approach, and the features with low variance have been removed.

### Normalization

Normalizing data is of high importance in accelerating the machine learning process and improving predictions. Most tools normalize the data in the range of [0,1] or [-1,1], depending on the application. Generally, data normalization refers to the process of mapping the data variables from one range of values to another for each feature to contribute equally to the model training. There are some challenges appear affecting the normalization process when some data values are out of range (outliers), which are usually removed from the set. When the data are transformed in the range [0,1], the method is called normalization [67]. Another approach which generates features with zero mean and unit variance is called standardization and the standardized value $x_{new}$ of a sample $x$ with $\mu$ statistical mean and $\sigma$ standard deviation is calculates as shown in Eq. 2. The features in this study have been standardized as described above.

$$x_{new} = \frac{x - \mu}{\sigma} \qquad Eq.\ 2$$

## Feature Selection

Machine Learning methods have difficulty in dealing with high-dimensional data, so pre-processing of the data and reduction of features is essential. Feature selection is one of the most important techniques in the data pre-processing, it helps in understanding data, reducing computation requirement, improving the predictor performance and reducing the effect of curse of dimensionality [68], [69]. The focus of feature selection is selecting a subset of variables from the feature space which can efficiently describe the input data while reducing effects from noise or irrelevant variables and still provide good predictions [70]. Filter methods are independent of the learning algorithms, and they rely only on the characteristics of data to assess feature importance. The feature importance evaluation process can be either univariate or multivariate. In the univariate approach, each feature is ranked individually regardless of other features. The multivariate approach ranks multiple features in a batch way [71].

To reduce the number of features, a feature selection method based on the Analysis of Variance (ANOVA) method was used in this study. ANOVA is a statistical procedure that compares means of several samples [72]. The purpose of this is to determine whether the means from two or more samples of data come from the same distribution or not. Specifically, it tests the null hypothesis $H_0$ of

Eq. 3 against the alternative hypothesis $H_1$ where there are at least two group means that are statistically significantly different from each other (Eq. 4).

$$H_0 = \mu_1 = \mu_2 = \mu_3 = \ldots = \mu_k \qquad Eq.\ 3$$

$$H_1: \exists\ 1 \leq i, l \leq k: \mu_i \neq \mu_l \qquad Eq.\ 4$$

Where $\mu$ is the group mean, $i, l$ are two groups and $k$ is the number of groups.

The formula for the one-way ANOVA F-test statistic is shown in Eq. 5, where $\bar{Y}_i$ denotes the sample mean in the $i$-th group, $n_i$ is the number of observations in the $i$-th group, $\bar{Y}$ denotes the overall mean of the data, $Y_{ij}$ is the $j$-th observation in the $i$-th out of $k$ groups and $N$ is the overall sample size.

$$F = \frac{between - group\ variablity}{within - group\ variability} = \frac{\sum_{i=1}^{k} n_i (\bar{Y}_i - \bar{Y})^2 / (k-1)}{\sum_{i=1}^{k} \sum_{j=1}^{k} \left(Y_{ij} - \bar{Y}_i\right)^2 / (N-k)} \qquad Eq.\ 5$$

A combined analysis of the $p$ values with respect to their corresponding $F$-scores for radiomics, transcriptomics and deep features led to the selection of a subset of the most significant features. After feature selection with a univariate method, LASSO method was used for further feature selection.

Shrinkage methods minimize the residual sum of squares of the model using Ordinary Least Squares (OLS) [73]. The least absolute shrinkage and selection operator (LASSO) can be used for parameter estimation and variable selection and minimizes the absolute sum of the regression coefficients. LASSO is a particular case of the penalized least squares regression with L1-penalty function. It improves both prediction accuracy and model interpretability by combining the good qualities of subset selection and ridge regression. The LASSO estimate can be defined by:

$$\hat{\beta}^{lasso} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2$$

$$Subject\ to \sum_{j=1}^{p} |\beta_j| \leq t \qquad Eq.\ 6$$

## Survival Analysis

The statistical analysis of what are usually referred to as survival time, lifetime, or failure time is an important topic in many areas, including the biomedical, social, and engineering sciences. Survival analysis has been a very active research field for several decades and refers to a set of statistical methods for analyzing the time until an event occurs, such as death in biological organisms or failure in mechanical systems. In survival analysis, the outcome variable has both an event and a time value associated with it. When it is used in medical studies, the event of interest is usually death. However, in cancer studies, the event could be the time between response to a treatment till disease-free survival time or response to treatment till recurrence. It is crucial to specify what is the event of interest and the starting and finishing point of observation period [74].

Usually, only some individuals have experienced the event and, subsequently, survival times will be unknown for a subset of the group of study. This phenomenon is called censoring and is one aspect that creates specific difficulties relating to survival analysis. Survival analysis generally deals with censored data, that is, when the time to the event is not observed. There are three main reasons why this happens. First, the individual withdraws from the study and there is no information about him after a specific time, second, the individual does not experience the event till the study is over, or third, the individual is lost to follow-up during the study period. Also, censoring can be categorized in three different types, right, left and interval censoring. Right censoring, the most common type of censoring, occurs when the survival time is incomplete at the right side of the follow-up period. That can be occurred when the subject does not experience the event because the study ends earlier, or the subject leaves the study before the event. An example of right censoring is depicted in Figure 7. In contrast to right censoring, if the event of interest has already occurred, meaning that the survival time of the individual is less than or equal to the observed survival time, the censoring is called left censoring. The last type of censoring is the interval-censoring, in which we do not know the exact timing of the exposure, but the event occurs between two timepoints. As an example, when dealing with lifetime problems and knowing both the birth and death of the subjects, right censoring occurs for those subjects whose birth is known and there are still alive when the study ends. Left censoring, on the other hand, is when the lifetime of the subject is known to be less than a certain duration.

When the lifetime is less than a specific threshold may not be observed at all. This is called truncation and is deliberate and due to design study. When the entire study population experience the event of interest, the phenomenon is called right truncation. When the subjects have been at risk before entering the study, they are left truncated [75].

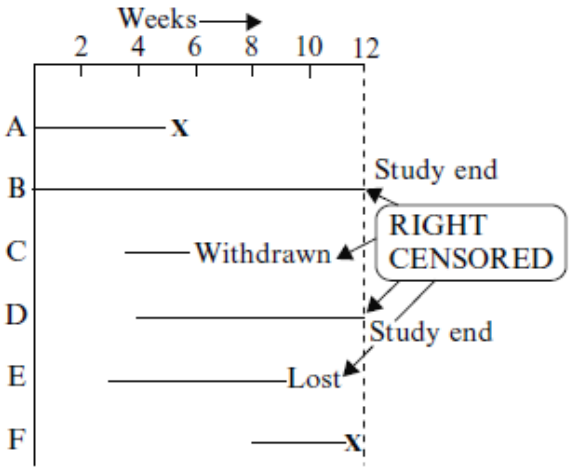


*Figure 7: Right censored individuals, true survival time is equal to or greater than the observed survival time* [75]

Two quantitative terms considered in any survival analysis are the survival function, denoted by $S(t)$, and the hazard function, denoted by $h(t)$. The survival function gives the probability that a person survives longer than some specified time $t$, that is, $S(t)$ gives the probability that the random variable $T$, exceeds the specified time $t$ (Eq. 7).

$$S(t) = P(T \geq t) = 1 - F(t) = \int f(x)dx$$

$$f(t) = \frac{dF(t)}{dt} = -\frac{dS(t)}{dt} \qquad \textit{Eq. 7}$$

where $T$ is a non-negative random variable of a person's survival time, $F(t)$ is the cumulative distribution function of $T$ with corresponding probability density function $f(t)$.

The survival function is essential in survival analysis, because knowing the probabilities for different values of $t$, provides crucial information on survival data. Time $t$ ranges from zero to infinity, where at time $t = 0$, is the start of the study and the probability of survival gets its higher value of one, and at time $t = \infty$, the survival probability decreases till eventually fall to zero. Usually, the survival function is represented as a step function, as is shown in Figure 8(ii).

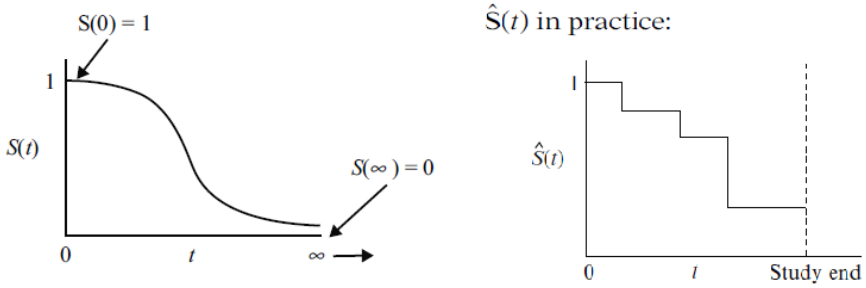


*Figure 8: Graphs representing survival function in a (i) smooth curve and (ii) step function.*

The hazard function, $h(t)$, is the instantaneous rate at which events occur, given no previous events. The hazard function is also referred to as mortality rate or risk in the health care field [76]. The hazard function $h(t)$ and the cumulative hazard function $H(t)$ are given by the Eq. 8. The hazard function gives the instantaneous potential per unit time for the event to occur, given that the individual has not experienced the event up to time $t$. The hazard function focuses on failing, on the occurrence of the event, whereas survival function focuses on not failing.

$$h(t) = \lim_{\Delta t \to 0} \frac{P(t \leq T < t + \Delta t | T \geq t)}{\delta t} = f(t)/S(t)$$

*Eq. 8*

$$H(t) = \int_0^t h(u)du = -\log\left(S(t)\right)$$

The survival probability can be estimated nonparametrically from observed times, using the Kaplan-Meier estimator [77]. The Kaplan-Meier method is a popular method to analyse "time-to-event" data. The outcome in Kaplan-Meier analysis often includes all-cause mortality. For $k$ patients, the events occur in distinct times $t_1 < t_2 < \cdots < t_k$. The probabilities of surviving from one interval of time to

the other may be multiplied together to give the cumulative survival probability. The probability $S(t_j)$ of not experiencing the event at time $t_j$ is calculated from $S(t_{j-1})$, which is the probability of being alive at time $t_{j-1}$, the number $n_j$ of patients alive just before time $t_j$ and the number of events $d_j$ at time $t_j$.

$$S(t_j) = S(t_{j-1})(1 - \frac{d_j}{n_j}) \qquad Eq.\ 9$$

The value of survival probability is constant between times of events, so the probability is a step function that changes value after each event. The Kaplan-Meier survival curve, the plot of survival probability against time, provides an overview of the data that can be used in survival analysis. The log rank test is a popular method to test the null hypothesis of no survival between two or more independent groups. It is a large-sample chi-square test that uses as its criterion a statistic that provides a comparison of all compared Kaplan-Meier survival curves. For each time point the observed number of events and the number of expected in each group are calculated. The number of expected events is calculated as the proportion of subjects who are at risk at a given time point multiplied by the total number of events at that point. The test is more likely to detect a difference between groups when the risk of an event is consistently greater for one group than another.

The Cox proportional Hazards model [78] is one of the most widely used methods for modelling survival data. For one explanatory variable in data of analysis, non-parametric methods like Kaplan-Meier, can be adequate if the groups that are in comparison are reasonably similar. However, the groups may differ in many aspects (age distributions, proportion of men and women etc). The analysis of different groups must be adjusted accordingly, otherwise the analysis may be confounded. The purpose of the Cox proportional hazards model is to evaluate simultaneously the effect of several factors on survival in a multivariate way. It allows analysts to examine factors that may influence the rate of an event occurrence at a specific point in time. The predictor variables, or factors, are also called covariates. The basic Cox model is described as:

$$h(t|\boldsymbol{Z}) = h_0(t)exp\ (\beta'\boldsymbol{Z}) \qquad Eq.\ 10$$

where $h_0(t)$ is the baseline hazard which may vary arbitrarily over time, $\boldsymbol{Z}$ is the covariate vector, and $\beta = (\beta_1, \beta_2, \dots, \beta_p)$ is a vector of covariate coefficients that are assumed to be constant. For two individuals with covariate vectors $\boldsymbol{Z}$ and $\boldsymbol{Z}^*$, the ratio of their hazard functions can be simplified to:

$$\frac{h(t|\boldsymbol{Z})}{h(t|\boldsymbol{Z}^*)} = \exp(\sum_{k=1}^{p} \beta_k(\boldsymbol{Z_k} - \boldsymbol{Z_k^*})) \qquad \textit{Eq. 11}$$

As it shown in Eq. 11, the ratio of the hazard rates of two covariate values is constant or proportional to the other and does not depend on time $t$. The Cox model can be described as a multiple linear regression of the logarithm of the hazard on the covariates $\boldsymbol{Z}$ with the baseline hazard being the intercept. The covariates act multiplicatively on a baseline hazard which may vary freely over time.

## Combining survival analysis with machine learning

At each time point that an event occurs, the process can actually be viewed as a classification problem, whether or not a certain subject will experience the event at the specific time. Cox proportional hazards model is the standard for analysing time-to-event data. However, there are several limitations in the method. Firstly, it assumes that hazard functions for any two individuals are proportional, it fails if features are highly correlated, its decision function is linear in the covariates and lastly, it is not applicable to data with more features than samples. Several machine learning models for survival analysis have been trained. After training the model, it can be used to predict the survival time of patients based on a given set of features. For a set of $n$ patients, we know the survival time $y_i$ of patient $i$, that is the exact time $c_i \geq 0$ of censoring (if the patient has not experienced the event $\delta_{\mathrm{i}} = 0$) and the time $t_i$ when the patient experienced the event ($\delta_{\mathrm{i}} = 1$). In this study, a random forest and a support vector machine will be evaluated as classifiers for survival analysis.

Support vector machines are classical machine learning techniques that can be used for classification and regression. The main goal in classification tasks is to find a hyperplane in an N-dimensional space (where N is the number of features) that distinctly classifiers the data points. Hyperplanes are decision boundaries that help classify the data points. Data points that fall on different side of the hyperplane can be attributed to different classes. Support vector machines are widely used because are effective in high dimensional spaces and they use a subset of training points in the decision function (called support vectors), so they are also memory efficient. There are two types, linear and non-linear. Linear support vector machines are used for linearly separable data and non-linear for non-linearly separated data, meaning that the dataset cannot be classified by using a straight line. In ranking-based linear support vector machines, the main objective is to recover the correct order of samples according to their relevance. In survival analysis, relevance indicates the survival time. The pairs of comparable samples that can be used for training can be defined as the set $P$ in Eq. 12 and $p = |P|$ defines the cardinality of this set, which is bounded by $O(n^2)$ space, where $n$ is the number of samples [79].

$$P = \{(i,j)|y_i > y_j \wedge \delta_j = 1\}_{i,j=1,\ldots,n} \qquad \textit{Eq. 12}$$

The objective function of ranking-based linear survival support vector machine is shown in Eq. 13.

$$f(w) = \frac{1}{2}w^T w + \frac{\gamma}{2}\sum_{i,j\in P} \max\left(0,1 - \left(w^T x_i - w^T x_j\right)\right)^2 \qquad \textit{Eq. 13}$$

where $w \in \mathrm{R}^d$ are the coefficients and $\gamma > 0$ is a regularization parameter. A new set of data points $X_{new}$ can be ranked with respect to their predicted survival time.

Another machine learning method used in survival analysis is random forests. Random forest is a commonly used algorithm, which combines the output of multiple decision trees to reach a single result. It is used both for classification and regression problems. The random forest is an extension of the bagging method, it utilizes bagging and feature randomness and creates uncorrelated forest of decision trees. While decision trees consider all possible feature splits, random forest selects only a subset of those features using feature randomness. The random forest is made up of a collection of decision trees, and each tree is comprised of a data sample drawn from the training set. The algorithm has three main hyperparameters, the node size, the number of trees, and the number of features sampled. There are many advantages in its uses, as it provides flexibility, makes easy to determine feature importance and has reduced risk of overfitting. However, it is time-consuming process, since it computes data for each individual decision tree, and is a more complex method compared to single decision tree method.

Random survival forests methodology extends random forest method. In classical random forest, randomization is introduced in two forms. A randomly drawn bootstrap sample is firstly used to grow a tree. Then, at each node of the tree, a randomly selected subset of covariates is used as candidate features for splitting. Averaging over trees enables random forest to approximate rich classes of functions and to maintain low generalization error. Extending random forest to right censored survival data is of high importance. Survival data are usually analysed with methods that rely on assumptions as proportional hazards. The methods used are parametric, so nonlinear effects of variables must be modelled by transformations. Difficulties as these are automatically handled with the used of random forests. In right censored survival data, the outcome is the survival time and the censoring status. Thus, the splitting criterion of growing a tree must involve the survival time and the censoring status. The predictive value for the terminal node in a tree and the predicted value from the forest must also properly incorporate the survival information [80].

The random survival forest algorithm starts with drawing $B$ bootstrap samples from the original data, which excludes on average 37% of the data, the so-called out-of-bag data. Then, it grows a survival tree for each bootstrap sample. At each node of the tree, it randomly selects $p$ candidate covariates.

The node is split using the candidate covariate that maximizes the survival difference between the daughter nodes. The trees are grown to full size under the constraint that a terminal node should have no less than a positive number $d_0$ of unique deaths. Next step is to calculate the cumulative hazard function for each tree, obtain the ensemble cumulative hazard function and use it to calculate the prediction error. In more detail, survival trees are binary trees grown by splitting tree nodes. A starting point of the tree is the root node, which is split into two daughter nodes using the survival criterion. Each node tree becomes homogeneous and is populated by cases with similar survival information. Each daughter node is also split into other two nodes, repeating the process until the termination node. The termination node is when no new daughter nodes can be formed. The survival times in terminal node $h$ is denoted as $(T_{1,h}, \delta_{1,h}), \dots, (T_{n(h),h}, \delta_{n(h),h})$. An individual $i$ is characterized as right censored if at time $T_{i,h}$ the event indicator $\delta_{i,h}$ is zero. The cumulative hazard function estimate for $h$ and for each case $i$ with $d$-dimensional covariate $x_i$ is defined below in Eq. 14 and Eq. 15, respectively:

$$\widehat{H}_h(t) = \sum_{t_{l,h} \le t} \frac{d_{l,h}}{Y_{l,h}} \qquad Eq.\ 14$$

$$H(t|x_i) = \widehat{H}_h(t), \qquad if\ x_i \in h \qquad Eq.\ 15$$

## Metrics

When dealing with binary dependent variables or continuous dependent variables that may be censored when the patients have not suffered the event, the usual mean squared error do not apply as a metric for the model. A concordance index (C-index) is widely used to measure the predictive discrimination and is applicable to ordinal outcomes and censored time until event response variables. The C-index is related to a rank correlation between observed and predicted outcomes. It was firstly introduced by Harrell et al. [81]. The C-index is related to the area under the ROC curve (AUC). Like for the AUC, C-index equal to 1 indicates perfect prediction accuracy and C-index equal to 0.5 indicates random prediction.

C-index is defined as the proportion of all usable patient pairs in which the predictions and the outcomes are concordant. The index measures the predictive power derived from a set of predictor covariates in a model. If the C-index is used to evaluate the prediction of time till the event, C-index is calculated by considering all possible pairs of individuals, when at least one of them has experienced the event. If the predicted survival time is larger than the actual survival time, for the patient who lived, the predictions for that pair are concordant with the outcome values. In other words, a pair is concordant, if the one with the higher estimated risk score has a shorter actual survival time. When

predicted risks are identical for a pair, 0.5 rather than 1 is added to the count of concordant pairs. A patient pair is unusable if both patients experienced the event at the same time or if one is still event-free but has not been followed long enough to determine the survival time, and the other experienced the event. The C-index is interpretable for the use as a misclassification probability, and for that reason is used as a metric for survival performance. Another reason for its use is the fact that it does not depend on a single fixed time for evaluation.

Since the evaluation of the predictive accuracy of a survival model is in terms of the C-index, it is natural to formulate the learning problem to directly maximize the C-index. As the C-index is invariant to any monotone transformation of the survival times, the model that learns by maximizing the C-index is considered as a ranking problem. The main goal of problems like that is to predict whether the survival time of one individual is larger than the survival time of the other individual. In ranking problems in machine learning, this is a $N - partite\ ranking\ problem$ where each data point is a different class. When we formulate the ranking problem, we can incorporate the censored data and use several ranking algorithms for survival analysis.

The C-index is calculated using the following steps:

i. Firstly, it forms all possible pairs over the dataset.
ii. It skips all the pairs whose shorter survival time is censored. Also, it skips pairs if the survival time is equal, meaning, for $i$ and $j$ individuals, if $T_i = T_j$.
iii. For the included pairs with different survival times $T_i \neq T_j$, it counts 1 if the shorter survival time has worse predicted outcome, and 0.5 if predicted outcomes are tied. For permissible pairs, when $T_i = T_j$ and indicate both deaths for individuals, it counts 1 if the predicted outcomes are tied, otherwise 0.5. When the same survival time does not indicate death, it counts 1 if the death has worse predicted outcome.
iv. Concordance denotes the sum over all permissible pairs. Permissible denotes the total number of permissible pairs.
v. The C-index is finally defined by $C = Concordance/Permissible$.

## Survival Analysis Library

Today, many successful ideas from machine learning have been adapted for time-to-event analysis, such as random forests, gradient boosting, and support vector machines. It is important to note that censored data does not only affect the training of the model, but also the evaluation of it, because held-out data will be subject to censoring too. For this study, `scikit-survival` library was used [82]. `scikit-survival` is an open-source Python package for time-to-event analysis fully compatible with `scikit-learn`, such that pre-processing and feature selection techniques within `scikit-learn` to be combined with the survival model. It provides implementations of many

machine learning techniques, such as penalized Cox model, Survival Support Vector Machine, and Random Survival Forest. Evaluation metrics range from simple rank correlation metrics, as C-index of Harrell et al., to time-dependent versions of well-known mean squared error and receiver operating characteristic curve. The documentation of the library contains installation instructions and a full description of the API. The library is distributed under the GPL-3 license with the source code available at `https://github.com/sebp/scikit-survival`. The biggest difference between time-to-event analysis and traditional machine learning techniques are the semantic predictions. In time-to-event analysis, predictions are usually arbitrary risk scores of experiencing or not the event, and not the actual time of experiencing the event, which is the input for training the model. For that reason, the evaluation of predictions is made by a measure of rank correlation between predicted risk scores and observed time points. The widely used Harrell's concordance index computes the ratio of correctly ordered-concordant pairs and is the default metric when using the model's `score()` function. For this study, `RandomSurvivalForest()` method and `FastSurvivalSVM()` were examined. For models that provide time dependent predictions, there are available two methods: `predict_survival_function()`, and `predict_cumulative_hazard_function()`. These methods return the survival plot and the cumulative hazard plot for examined dataset.

# Chapter 4: Research findings / results

In this study, the fusion of deep features, radiomics and transcriptomics is evaluated. Two data views are considered: (A) deep features with radiomics and (B) deep features, radiomics and transcriptomics. The fusion includes the concatenation of features into one common feature space. For the classification of low- and high-risk patients, two classifiers were employed, namely: (1) Random Survival Forest and (2) Survival Support Vector Machine. Fourfold cross-validation on a patient basis was applied to the original dataset for splitting the data into training and testing sets. For a total of 40 patients, each test fold contained 10 patients at a time.

## A1. Deep features and radiomics - Random Forest

A total of 2996 radiomics features and deep features varying from 502 to 7952 for 18 deep models were assessed for their prognosis performance. After univariate and multivariate feature selection, a total of 25 features were kept. From these, 23 of them were deep features and two radiomics. The best performance was obtained using deep model $MobileNet$. In the test cohort, random forest classifier achieved a C-index of $0.68 \pm 0.03$ $(0.65\ to\ 0.72)$ . The survival function and the cumulative hazard function were plotted for the fold with the best C-index and are presented in Figure 9 and Figure 10 respectively. The survival function can be interpreted as the probability of a patient to survive beyond a certain time t. At time 0, no patient has experienced the event, so all have a probability of 100% that they survive. The drop of curves reflects patients experiencing the event. The survival plot ends at median time of survival since all the events have taken place until this time. Survival plot can also show the probability of survival for each patient in a specific time point. Seven out of ten patients have been predicted with less than 50% survival probability until the median survival time. Patients that have less than 50% survival probability can be characterized as high-risk patients.

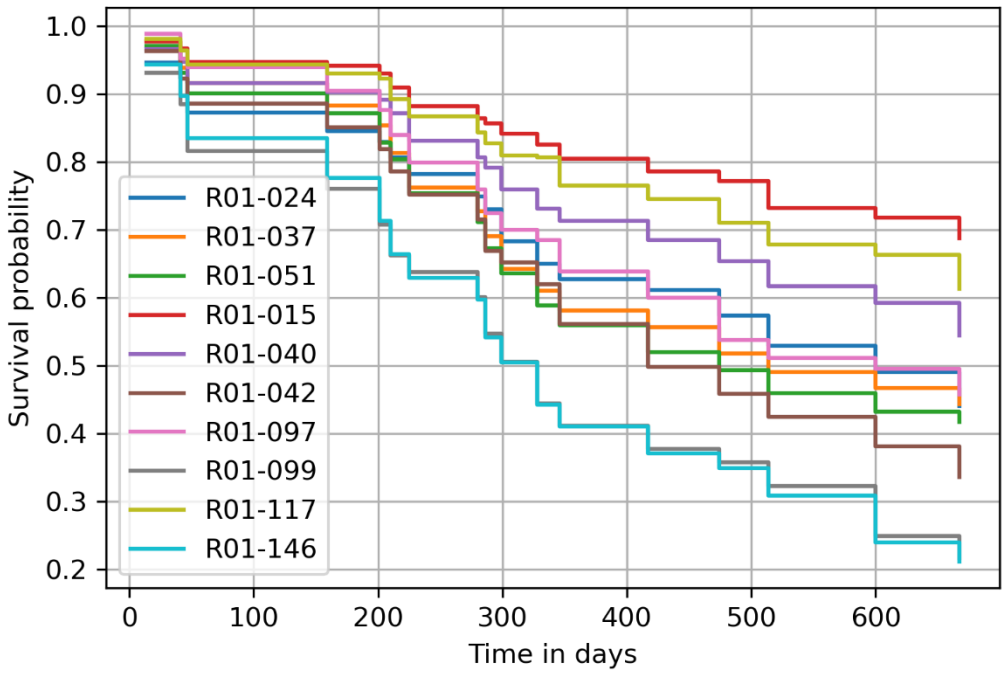


*Figure 9: Survival function using deep features and radiomics with random forest classifier*

The actual status and the predicted survival probability till the observed median survival time are shown in Table 1. The classifier predicted correctly as high-risk 5 patients and as low-risk 2 patients. The classifier failed to predict 3 patients, one high-risk that was classified as low-risk with probability higher than 50% (patient R01-015) and 2 low-risk classified as high-risk (patients R01-097 and R01-099).

*Table 1:* Survival probability using random forest classifier with data view of deep features and radiomics. The classifier correctly predicted 7 patients. The classifier failed to predict 3 patients, one high-risk patient that was classified as low-risk and 2 low-risk classified as high-risk.

| ***Patient*** | ***Actual status*** | ***Survival probability*** |
|---|---|---|
| $\boldsymbol{R01-037}$ | $High-risk$ | $< 50\%$ |
| $\boldsymbol{R01-042}$ | $High-risk$ | $< 50\%$ |
| $\boldsymbol{R01-051}$ | $High-risk$ | $< 50\%$ |
| $\boldsymbol{R01-146}$ | $High-risk$ | $< 50\%$ |

| | | |
|---|---|---|
| $\boldsymbol{R01-024}$ | $High-risk$ | $< 50\%$ |
| $\boldsymbol{R01-015}$ | $High-risk$ | $> 50\%$ |
| $\boldsymbol{R01-099}$ | $Low-risk$ | $< 50\%$ |
| $\boldsymbol{R01-117}$ | $Low-risk$ | $> 50\%$ |
| $\boldsymbol{R01-097}$ | $Low-risk$ | $< 50\%$ |
| $\boldsymbol{R01-040}$ | $Low-risk$ | $> 50\%$ |

For each tree in the ensemble, the cumulative hazard function for a patient with feature vector $x$ is calculated from all samples of the bootstrap sample that are in the same terminal node as $x$. Similarly, for cumulative hazard function, at time 0 all patients have cumulative hazard rate equal to 0. The hazard rate for each patient increases by increasing the time. The survival was predicted to be better for patients with cumulative hazard lower than 1, compared to patients with cumulative hazard greater than 1.

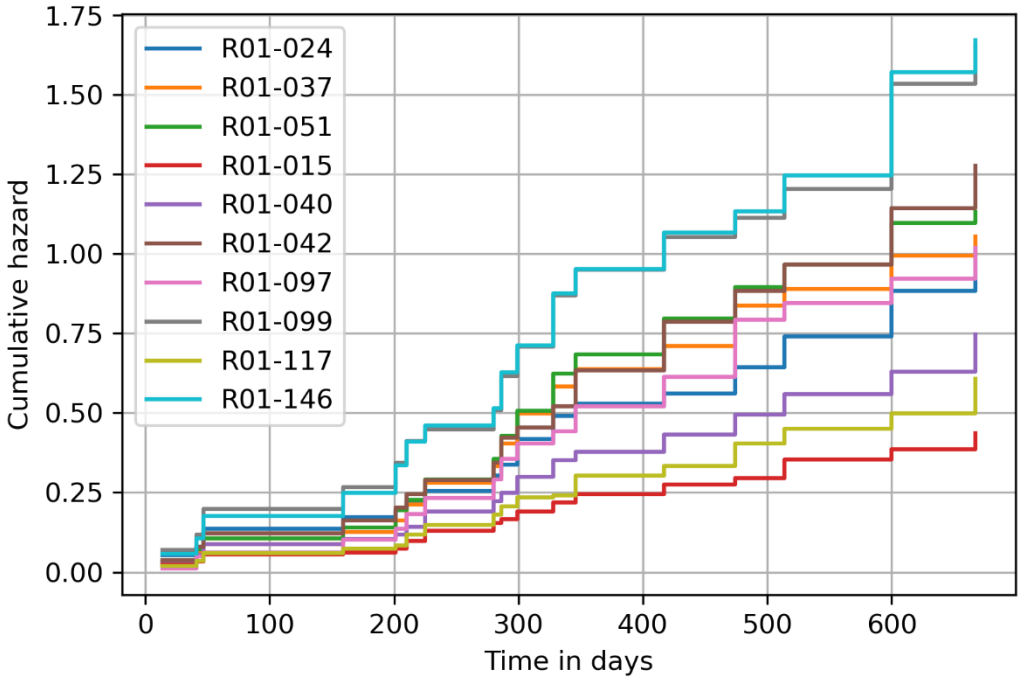


*Figure 10: Cumulative Hazard Function using deep features and radiomics with random forest classifier*

The best results in the testing folds achieved with $MobileNet$ model, resulting in a mean C-index of $0.68 \pm 0.03$ $(0.65 - 0.72)$. However, $ResNet$ and $DenseNet$ also achieved a good discrimination ability, compared to other deep models, as are shown in Table 2. $ResNet$ achieved a mean C-index of $0.61$, and $DenseNet$ achieved a lower mean C-index of $0.59$.

*Table 2: Best deep learning models for patient classification using random forest. MobileNet achieved the best C-index of 0.68. C-index are predicted for test set.*

| $\boldsymbol{Model\ name}$ | $\boldsymbol{Mean \pm std}$ | $\boldsymbol{Min\ C-Index}$ | $\boldsymbol{Max\ C-Index}$ |
|---|---|---|---|
| $\boldsymbol{ResNet}$ | $0.61 \pm 0.07$ | 0.57 | 0.66 |
| $\boldsymbol{MobileNet}$ | $\mathbf{0.68 \pm 0.03}$ | $\mathbf{0.65}$ | $\mathbf{0.72}$ |
| $\boldsymbol{DenseNet}$ | $0.59 \pm 0.10$ | 0.52 | 0.62 |

For prediction, an individual is dropped down each tree in the forest until it reaches a terminal node. Data in each terminal is used to estimate the survival and cumulative hazard function. A risk score can

be computed using random survival forest, which represents the expected number of events for one particular terminal node. The ensemble prediction is the average across all trees in the forest. The higher predicted risk scores indicate shorter survival whereas a lower risk score indicate longer survival of the patient. Table 3 presents the risk score for each individual predicted using random survival forest. As it was already stated, the classifier predicted incorrectly higher risk score for patients R01-097 and R01-099 and a lower risk score for patient R01-015.

*Table 3: Survival time and predicted risk scores for testing set with deep features and radiomics using random forest classifier.*

| $\boldsymbol{Patient}$ | $\boldsymbol{Survival\ time\ (days)}$ | $\boldsymbol{Risk\ score}$ |
|---|---|---|
| $\boldsymbol{R01-037}$ | 28 | 9.05 |
| $\boldsymbol{R01-042}$ | 42 | 8.57 |
| $\boldsymbol{R01-051}$ | 261 | 8.54 |
| $\boldsymbol{R01-146}$ | 276 | 12.16 |
| $\boldsymbol{R01-024}$ | 366 | 7.90 |
| $\boldsymbol{R01-015}$ | 430 | 3.19 |
| $\boldsymbol{R01-099}$ | 952 | 12.07 |
| $\boldsymbol{R01-117}$ | 1083 | 4.00 |
| $\boldsymbol{R01-097}$ | 1352 | 6.98 |
| $\boldsymbol{R01-040}$ | 2041 | 4.09 |

## A2. Deep features and radiomics - Support Vector Machine

After univariate and multivariate feature selection, 27 deep features and 4 radiomics were selected to predict the survival outcome of the patients. The best performance was obtained using deep model $DenseNet$. In the test cohort, support vector machine reached a C-index of $0.73 \pm 0.07$ $(0.68\ to\ 0.75)$. $ResNet$ and $NasNet$ models achieved good results, as shown in Table 4.

*Table 4: Best deep learning models for patient classification using support vector machine. DenseNet achieved the best C-index of 0.73. C-index are predicted for test set.*

| $\boldsymbol{Model\ name}$ | $\boldsymbol{Mean \pm std}$ | $\boldsymbol{Min\ C-Index}$ | $\boldsymbol{Max\ C-Index}$ |
|---|---|---|---|
| $\boldsymbol{ResNet}$ | $0.72 \pm 0.09$ | 0.59 | 0.75 |
| $\boldsymbol{DenseNet}$ | $\mathbf{0.73 \pm 0.07}$ | $\mathbf{0.68}$ | $\mathbf{0.75}$ |
| $\boldsymbol{NasNet}$ | $0.67 \pm 0.11$ | 0.56 | 0.77 |

A risk score can be computed using survival support vector machine, which represents the ranking of samples according to survival times. Table 5 presents the risk score for each individual predicted using survival support vector machine. The higher predicted risk scores indicate shorter survival whereas a lower risk score indicate longer survival of the patient. For example, patient R01-031 had the highest survival time and the lowest risk score was predicted. Similarly, patients R01-093 and R01-119 had the lowest survival time and for those, a high value risk score was predicted but not in the right order.

*Table 5: Survival time and predicted risk scores for testing set with deep features and radiomics using support vector machine.*

| $\boldsymbol{Patient}$ | $\boldsymbol{Survival\ time\ (days)}$ | $\boldsymbol{Risk\ score}$ |
|---|---|---|
| $\boldsymbol{R01-093}$ | 47 | 1.73 |
| $\boldsymbol{R01-119}$ | 159 | 2.25 |
| $\boldsymbol{R01-066}$ | 201 | 0.38 |
| $\boldsymbol{R01-106}$ | 225 | 0.72 |
| $\boldsymbol{R01-072}$ | 299 | 0.91 |
| $\boldsymbol{R01-017}$ | 474 | 0.61 |
| $\boldsymbol{R01-055}$ | 1165 | 0.90 |

| | | |
|---|---|---|
| $\boldsymbol{R01-077}$ | 1322 | 0.06 |
| $\boldsymbol{R01-101}$ | 1491 | 0.27 |
| $\boldsymbol{R01-031}$ | 1890 | $-2.79$ |

## B1. Deep radiotranscriptomics - Random Forest

A total of 2996 radiomics, 5268 transcriptomics and deep features varying from 502 to 7952 for 18 deep models were assessed for their prognosis performance. After univariate and multivariate feature selection, a total of 49 features were kept. From these, 21 of them were deep features, 2 radiomics and 26 transcriptomics. The best performance was obtained using deep model $ResNet$. In the test cohort, random forest classifier achieved a C-index of $0.74 \pm 0.11\ (0.63\ to\ 0.81)$ . The survival function and the cumulative hazard function were plotted for the fold with the best C-index and are presented in Figure 11 and Figure 12. The survival function can be interpreted as the probability of a patient to survive beyond a certain time t. At time 0, no patient has experienced the event, so all have a probability of 100% that they survive. The drop of curves reflects patients experiencing the event. The survival plot ends at time the median time of survival since all the events have taken place until this time. Survival plot can also show the probability of survival for each patient in a specific time point. Seven out of ten patients have been predicted with less than 50% survival probability until the median survival time. Patients that have less than 50% survival probability can be characterized as high-risk patients.

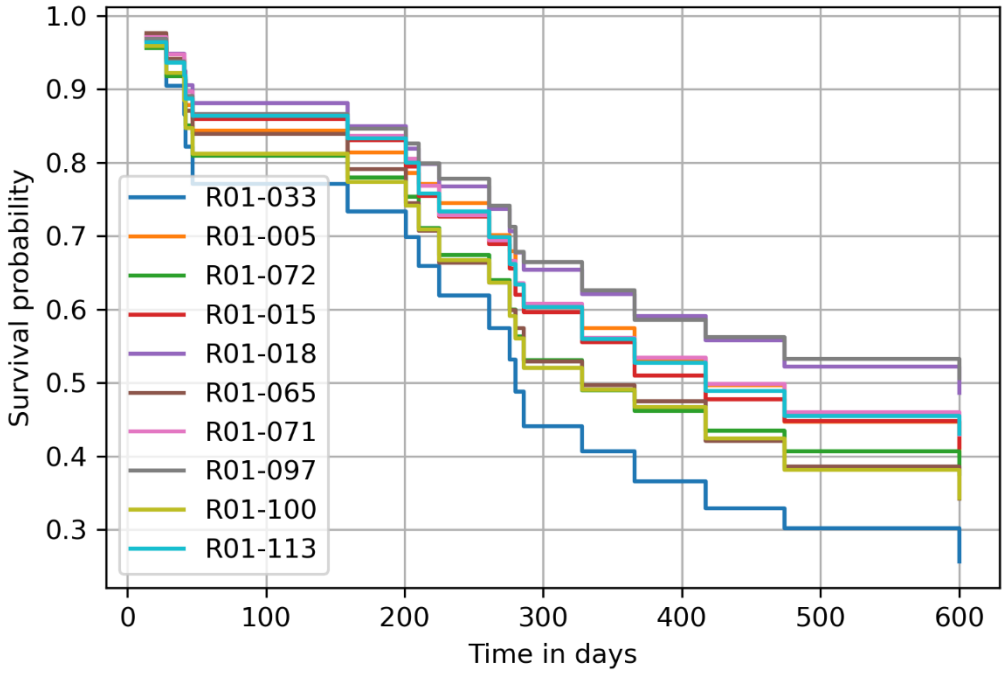


*Figure 11: Survival function using deep features, radiomics and transcriptomics with random forest classifier*

The actual status and the predicted survival probability till the observed median survival time are shown in Table 6. The classifier predicted correctly as high-risk 5 patients and as low-risk 2 patients. The classifier failed to predict 3 patients as low-risk (R01-005, R01-071 and R01-100) and predicted with lower survival probability.

*Table 6: Survival probability using random forest classifier with data view of deep features, radiomics and transcriptomics. The classifier correctly predicted 7 patients. The classifier failed to correctly predict 3 patients.*

| ***Patient*** | ***Actual status*** | ***Survival probability*** |
|---|---|---|
| $\boldsymbol{R01-072}$ | $High-risk$ | $< 50\%$ |
| $\boldsymbol{R01-065}$ | $High-risk$ | $< 50\%$ |
| $\boldsymbol{R01-015}$ | $High-risk$ | $< 50\%$ |
| $\boldsymbol{R01-033}$ | $High-risk$ | $< 50\%$ |

| | | |
|---|---|---|
| $\boldsymbol{R01-113}$ | $High-risk$ | $< 50\%$ |
| $\boldsymbol{R01-100}$ | $Low-risk$ | $< 50\%$ |
| $\boldsymbol{R01-018}$ | $Low-risk$ | $> 50\%$ |
| $\boldsymbol{R01-097}$ | $Low-risk$ | $> 50\%$ |
| $\boldsymbol{R01-071}$ | $Low-risk$ | $< 50\%$ |
| $\boldsymbol{R01-005}$ | $Low-risk$ | $< 50\%$ |

For each tree in the ensemble, the cumulative hazard function for a patient with feature vector $x$ is calculated from all samples of the bootstrap sample that are in the same terminal node as $x$. Similarly, for cumulative hazard function, at time 0 all patients have cumulative hazard rate equal to 0. The hazard rate for each patient increases by increasing the time. The survival was predicted to be better for patients with cumulative hazard lower than 1, compared to patients with cumulative hazard greater than 1.

The best results in the testing folds achieved with $ResNet$ model, resulting in a mean C-index of $0.74 \pm 0.11$ $(0.63\ to\ 0.81)$. However, $MobileNet$ and $DenseNet$ also achieved a good discrimination ability, compared to other deep models, as are shown in Table 7. $MobileNet$ achieved a mean C-index of $0.65$, and $DenseNet$ achieved a higher mean C-index of $0.71$.

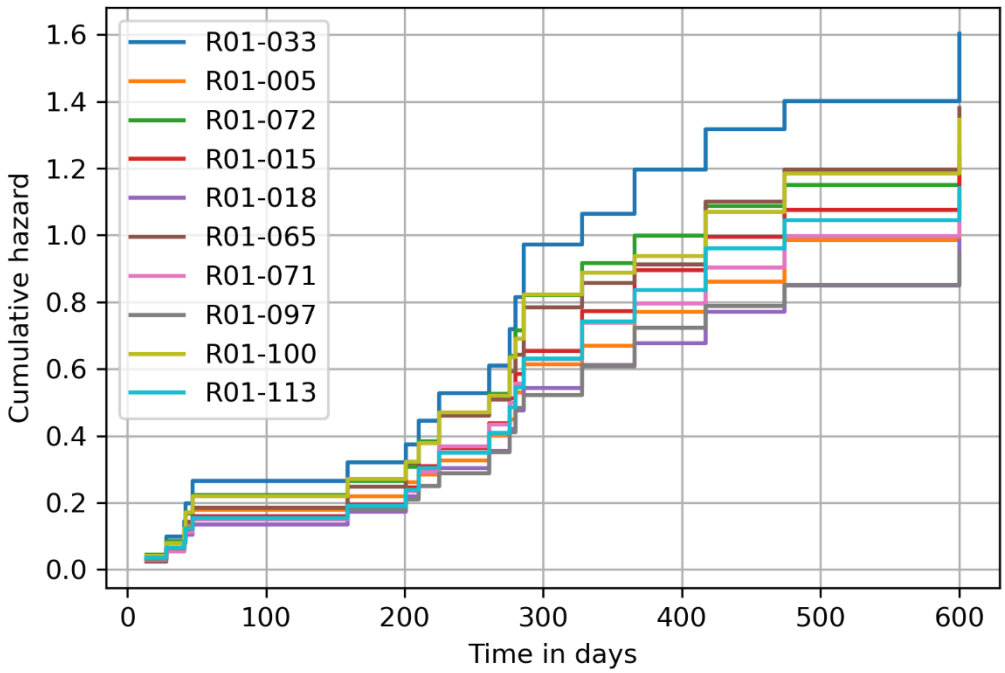


*Figure 12: Cumulative Hazard Function using deep features, radiomics and transcriptomics with random forest classifier*

*Table 7: Best deep learning models for patient classification using random forest. ResNet achieved the best C-index of 0.74. C-index are predicted for test set.*

| $\boldsymbol{Model\ name}$ | $\boldsymbol{Mean \pm std}$ | $\boldsymbol{Min\ C-Index}$ | $\boldsymbol{Max\ C-Index}$ |
|---|---|---|---|
| $\boldsymbol{ResNet}$ | $\mathbf{0.74 \pm 0.11}$ | $\mathbf{0.63}$ | $\mathbf{0.81}$ |
| $\boldsymbol{MobileNet}$ | $0.65 \pm 0.08$ | 0.61 | 0.72 |
| $\boldsymbol{DenseNet}$ | $0.71 \pm 0.09$ | 0.64 | 0.79 |

A risk score can be computed using random survival forest, which represents the expected number of events for one particular terminal node. The ensemble prediction is the average across all trees in the forest. The higher predicted risk scores indicate shorter survival whereas a lower risk score indicate longer survival of the patient. Table 8 presents the risk score for each individual predicted using random survival forest. As it was already stated, the classifier predicted incorrectly higher risk score for patients R01-100, R01-071 and R01-005. For these three patients, the predicted risk score should be lower than predicted and decreasing as the survival time increases.

*Table 8: Survival time and predicted risk scores for testing set with deep features, radiomics and transcriptomics using random forest classifier.*

| $\boldsymbol{Patient}$ | $\boldsymbol{Survival\ time\ (days)}$ | $\boldsymbol{Risk\ score}$ |
|---|---|---|
| $\boldsymbol{R01-072}$ | 299 | 10.18 |
| $\boldsymbol{R01-065}$ | 346 | 9.88 |
| $\boldsymbol{R01-015}$ | 430 | 9.73 |
| $\boldsymbol{R01-033}$ | 514 | 11.11 |
| $\boldsymbol{R01-113}$ | 667 | 8.34 |
| $\boldsymbol{R01-100}$ | 867 | 10.17 |
| $\boldsymbol{R01-018}$ | 1176 | 7.07 |
| $\boldsymbol{R01-097}$ | 1352 | 7.01 |
| $\boldsymbol{R01-071}$ | 1425 | 8.17 |
| $\boldsymbol{R01-005}$ | 1456 | 8.03 |

## B2. Deep radiotranscriptomics - Support Vector Machine

After univariate and multivariate feature selection, 13 deep features, 2 radiomics and 21 transcriptomics features were selected to predict the survival outcome of the patients. The best performance was obtained using deep model $MobileNet$. In the test cohort, support vector machine reached a C-index of $0.77 \pm 0.10$ $(0.65\ to\ 0.83)$. $ResNet$ and $NasNet$ models achieved good results, as shown in Table 9.

*Table 9: Best deep learning models for patient classification using support vector machine. MobileNet achieved the best C-index of 0.77. C-index are predicted for test set.*

| $\boldsymbol{Model\ name}$ | $\boldsymbol{Mean \pm std}$ | $\boldsymbol{Min\ C-Index}$ | $\boldsymbol{Max\ C-Index}$ |
|---|---|---|---|
| $\boldsymbol{ResNet}$ | $0.72 \pm 0.15$ | 0.53 | 0.79 |
| $\boldsymbol{MobileNet}$ | $\mathbf{0.77 \pm 0.10}$ | $\mathbf{0.65}$ | $\mathbf{0.83}$ |
| $\boldsymbol{NasNet}$ | $0.65 \pm 0.13$ | 0.55 | 0.72 |

A risk score can be computed using survival support vector machine, which represents the ranking of samples according to survival times. Table 10 presents the risk score for each individual predicted using survival support vector machine. The higher predicted risk scores indicate shorter survival whereas a lower risk score indicate longer survival of the patient. For example, patient R01-037 had the lowest survival time and survival support vector machine ranked the patient with the highest risk score. Similarly, patient R01-039 had the second lower survival time and a high risk score was predicted. Generally, we can see a decrease in the risk score for patients with high survival time. However, in some cases as for patients R01-097 and R01-026 the algorithm failed to rank the patients in the correct order, since R01-026 had the highest survival time should get the lowest risk score.

*Table 10: Survival time and predicted risk scores for testing set with deep features, radiomics and transcriptomics using support vector machine.*

| $\boldsymbol{Patient}$ | $\boldsymbol{Survival\ time\ (days)}$ | $\boldsymbol{Risk\ score}$ |
|---|---|---|
| $\boldsymbol{R01-037}$ | 28 | 2.44 |
| $\boldsymbol{R01-039}$ | 41 | 0.93 |
| $\boldsymbol{R01-146}$ | 276 | 0.50 |
| $\boldsymbol{R01-029}$ | 286 | 0.17 |
| $\boldsymbol{R01-128}$ | 328 | 0.77 |

| | | |
|---|---|---|
| $\boldsymbol{R01-065}$ | 346 | $-0.35$ |
| $\boldsymbol{R01-100}$ | 867 | $-1.98$ |
| $\boldsymbol{R01-078}$ | 1133 | $-0.75$ |
| $\boldsymbol{R01-097}$ | 1352 | $-5.18$ |
| $\boldsymbol{R01-026}$ | 2356 | $-3.25$ |

# Chapter 5: Discussion and analysis of findings

NSCLC patients are usually characterized with median overall survival time. It is of high importance to predict the survival of patients according to their radiomic or transcriptomic signatures. Outcome modelling can enable us to identify the prognostic signature of patients and stratify them according to their survival time. However, the survival depends on multiple information, and it is difficult to be predicted, since histologic, genetic, imaging, and clinical information play an important role in the overall survival of cancer patients. Several efforts have used single source data to investigate the survival of NSCLC. The goal of this study is to examine the synergistic use of high dimensional and high throughput data for identifying the survival of patients.

In this study, we compared the prognostic power of two data views, a combination of radiomics and deep features and a combination of radiomics, transcriptomics and deep features. A set of 40 patients with extracted radiomic and deep features and an obtained transcriptomic signature was examined. For these patients, the exact date of death was known. The median survival time was calculated and the patients that had time till death greater than the median survival time was considered as low-risk patients. Individuals with time till death lower than the median survival time were characterized as high-risk patients. As a result, a subset of 23 patients were labeled with 1 as high-risk patients and the remaining 17 as low risk patients, with label 0. Two classifiers were trained for the stratification of patients, random survival forest and survival support vector machine. The data were first pre-processed using a variance threshold and features with low variance were removed. The remaining features were standardized in values [-1,1] and were further decreased with univariate and multivariate feature selection. Performing ANOVA analysis and LASSO, the most important features for prediction were collected from each data type. The features were then concatenated into one common feature space prior to classification.

The selected features' prognostic power was compared using random forest and support vector machine. In machine learning methods for survival analysis, the predictions are arbitrary risk scores of experiencing or not the event. The evaluation of the predictions is made by a measure of rank correlation between predicted risk scores and observed time points. For that reason, C-index was used as a metric of the performance of the models. The results from both classifiers and data views are shown in Table 11. The radiomics and deep features analyses using both classifiers achieved a performance at the lower end of the spectrum of metrics. The mean C-index was $0.68 \pm 0.03$ and $0.73 \pm 0.07$ for random survival forest and survival support vector machine, respectively. Using a 4-fold cross validation method, the C-index in each fold varied from $0.65$ to $0.72$ for random forest classifier and from $0.68$ to $0.75$ for support vector machine. A slight increase of C-index was observed when transcriptomics was also used for the stratification of patients. Using random forest classifier, the prediction resulted in a C-index of $0.74 \pm 0.11$ and for each fold the C-index varied from $0.63$ till $0.81$. Support vector machine obtained the best results of C-index $0.77 \pm 0.10$ in a range of $0.65$ to $0.83$.

*Table 11: Results with different data views and classifiers*

| $\boldsymbol{Experiment}$ | $\boldsymbol{C-index}$ |
|---|---|
| $\boldsymbol{Deep\ features\ and\ radiomics\ -\ Random\ Forest}$ | $0.68 \pm 0.03\ (0.65\ to\ 0.72)$ |
| $\boldsymbol{Deep\ features\ and\ radiomics\ -\ Support\ Vector\ Machine}$ | $0.73 \pm 0.07\ (0.68\ to\ 0.75)$ |
| $\boldsymbol{Deep\ radiotranscriptomics\ -\ Random\ Forest}$ | $0.74 \pm 0.11\ (0.63\ to\ 0.81)$ |
| $\boldsymbol{Deep\ radiotranscriptomics\ -\ Support\ Vector\ Machine}$ | $0.77 \pm 0.10\ (0.65\ to\ 0.83)$ |

For each experiment, a risk score for patients in the unseen data was computed. The risk score should increase for patients with high survival time and decrease for low-risk patients. Also, both machine learning techniques that were used in this study could be described as a ranking problem, the model learns to assign samples with shorter survival times a higher risk score. A clear decrease of risk score with the increase of survival time could be observed with Deep radiotranscriptomics data and support vector machine. The algorithm calculated the highest score to the patients with the lowest survival and a low value of risk score for low-risk patients. However, the order of the patients' survival was not always correct. For example, for a patient with 1352 days of survival the predicted risk score was -5.18 whereas for patient with 2356, the predicted risk score was -3.25, indicating that the second patient was more likely to experience the event earlier than the first.

Another interesting finding was regarding the deep models that provided deep features with the best prognostic power. Eighteen deep models were used for the extraction of deep features. In most cases, MobileNet, ResNet, DenseNet and NasNet were the four deep models from which using these deep features, the C-index of survival models got the higher values. The C-index of 0.77 with support vector machine and the C-index of 0.68 with random forest was achieved with deep features extracted from MobileNet. Deep radiotranscriptomics survival analysis using random forest had the highest C-index of 0.74 with deep features from ResNet, and survival analysis using radiomics and deep features extracted from DenseNet, with support vector machine classifier got the highest C-index of 0.73.

# Chapter 6: Conclusion and recommendations

This master thesis was aimed to compare the performance of machine learning models for survival analysis that have been developed using different data types. The deep radiotranscriptomics framework outperform the performance of models based solely on imaging information. Both classifiers had approximately the same performance. More machine learning models can be examined for classification of patients in a future work. Also, further study may focus on specific subsets of patients for specific markers.

# References


[1] H. Sung *et al.*, "Global Cancer Statistics 2020: GLOBOCAN Estimates of Incidence and Mortality Worldwide for 36 Cancers in 185 Countries," *CA. Cancer J. Clin.*, vol. 71, no. 3, pp. 209–249, 2021, doi: 10.3322/caac.21660.

[2] "World Health Organization." https://covid19.who.int/ (accessed Apr. 25, 2022).

[3] K. M. Latimer and T. F. Mott, "Lung cancer: Diagnosis, treatment principles, and screening," *Am. Fam. Physician*, vol. 91, no. 4, pp. 250–256, 2015.

[4] J. D. Minna, J. A. Roth, and A. F. Gazdar, "Focus on lung cancer," *Cancer Cell*, vol. 1, no. 1, pp. 49–52, 2002, doi: 10.1016/s1535-6108(02)00027-2.

[5] L. G. Collins, C. Haines, R. Perkel, and R. E. Enck, "Lung cancer: Diagnosis and management," *Am. Fam. Physician*, vol. 75, no. 1, pp. 56–63, 2007.

[6] S. M. Gadgeel, S. S. Ramalingam, and G. P. Kalemkerian, "Treatment of Lung Cancer," *Radiol. Clin. North Am.*, vol. 50, no. 5, pp. 961–974, 2012, doi: 10.1016/j.rcl.2012.06.003.

[7] T. B. Murdoch and A. S. Detsky, "The inevitable application of big data to health care," *JAMA - J. Am. Med. Assoc.*, vol. 309, no. 13, pp. 1351–1352, 2013, doi: 10.1001/jama.2013.393.

[8] Z. Ahmad, S. Rahim, M. Zubair, and J. Abdul-Ghafar, "Artificial intelligence (AI) in medicine, current applications and future role with special emphasis on its potential and promise in pathology: present and future impact, obstacles including costs and acceptance among pathologists, practical and philosoph," *Diagn. Pathol.*, vol. 16, no. 1, pp. 1–16, 2021, doi: 10.1186/s13000-021-01085-4.

[9] J. He, S. L. Baxter, J. Xu, J. Xu, X. Zhou, and K. Zhang, "The practical implementation of artificial intelligence technologies in medicine," *Nat. Med.*, vol. 25, no. 1, pp. 30–36, 2019, doi: 10.1038/s41591-018-0307-0.

[10] A. Alonso-Betanzos and V. Bolón-Canedo, "Big-Data Analysis, Cluster Analysis, and Machine-Learning Approaches BT - Sex-Specific Analysis of Cardiovascular Function," P. L. M. Kerkhof and V. M. Miller, Eds. Cham: Springer International Publishing, 2018, pp. 607–626.

[11] F. Wang and A. Preininger, "AI in Health: State of the Art, Challenges, and Future Directions," *Yearb. Med. Inform.*, vol. 28, no. 1, pp. 16–26, 2019, doi: 10.1055/s-0039-1677908.

[12] S. M. Anwar, M. Majid, A. Qayyum, M. Awais, M. Alnowami, and M. K. Khan, "Medical Image Analysis using Convolutional Neural Networks: A Review," *J. Med. Syst.*, vol. 42, no. 11, pp. 1–13, 2018, doi: 10.1007/s10916-018-1088-1.

[13] G. Lee and H. Fujita, *Deep learning in medical image analysis, Challenges and Applications*, vol. 7, no. 4. 2020.

[14] P. Lambin *et al.*, "Radiomics: extracting more information from medical images using advanced feature analysis.," *Eur. J. Cancer*, vol. 48, no. 4, pp. 441–446, Mar. 2012, doi: 10.1016/j.ejca.2011.11.036.

[15] R. Thawani *et al.*, "Radiomics and radiogenomics in lung cancer: A review for the clinician," *Lung Cancer*, vol. 115, no. October 2017, pp. 34–41, 2018, doi: 10.1016/j.lungcan.2017.10.015.

[16] R. Lowe, N. Shirley, M. Bleackley, S. Dolan, and T. Shafee, "Transcriptomics technologies," *PLoS Comput. Biol.*, vol. 13, no. 5, pp. 1–23, 2017, doi: 10.1371/journal.pcbi.1005457.

[17] E. Trivizakis *et al.*, "Artificial intelligence radiogenomics for advancing precision and effectiveness in oncologic care (Review)," *Int. J. Oncol.*, vol. 57, no. 1, pp. 43–53, 2020, doi: 10.3892/ijo.2020.5063.

[18] J. Morgado *et al.*, "Machine Learning and Feature Selection Methods for EGFR Mutation Status Prediction in Lung Cancer," *Applied Sciences* , vol. 11, no. 7. 2021, doi: 10.3390/app11073273.

[19] Y. Zhu *et al.*, "A computed tomography (CT)-derived radiomics approach for predicting primary co-mutations involving TP53 and epidermal growth factor receptor (EGFR) in patients with advanced lung adenocarcinomas (LUAD)," *Ann. Transl. Med.*, vol. 9, no. 7, pp. 545–545, 2021, doi: 10.21037/atm-20-6473.

[20] E. Trivizakis, I. Souglakos, A. H. Karantanas, and K. Marias, "Deep radiotranscriptomics of non-small cell lung carcinoma for assessing molecular and histology subtypes with a data-driven analysis," *Diagnostics*, vol. 11, no. 12, 2021, doi: 10.3390/diagnostics11122383.

[21] L. Fan, Q. Cao, X. Ding, D. Gao, Q. Yang, and B. Li, "Radiotranscriptomics signature-based predictive nomograms for radiotherapy response in patients with nonsmall cell lung cancer: Combination and association of CT features and serum miRNAs levels," *Cancer Med.*, vol. 9, no. 14, pp. 5065–5074, 2020, doi: 10.1002/cam4.3115.

[22] D. Hui, "Prognostication of survival in patients with advanced cancer: Predicting the unpredictable?," *Cancer Control*, vol. 22, no. 4, pp. 489–497, 2015, doi: 10.1177/107327481502200415.

[23] A. Chaddad, C. Desrosiers, M. Toews, and B. Abdulkarim, "Predicting survival time of lung cancer patients using radiomic analysis.," *Oncotarget*, vol. 8, no. 61, pp. 104393–104407, 2017, doi: 10.18632/oncotarget.22251.

[24] C. Shen *et al.*, "2D and 3D CT Radiomics Features Prognostic Performance Comparison in Non-Small Cell Lung Cancer.," *Transl. Oncol.*, vol. 10, no. 6, pp. 886–894, 2017, doi: 10.1016/j.tranon.2017.08.007.

[25] L. Yang *et al.*, "Development of a radiomics nomogram based on the 2D and 3D CT features to predict the survival of non-small cell lung cancer patients.," *Eur. Radiol.*, vol. 29, no. 5, pp. 2196–2206, 2019, doi: 10.1007/s00330-018-5770-y.

[26] V. M. Bluthgen *et al.*, "Prognostic value of histogram analysis in advanced non-small cell lung cancer: A radiomic study," *Oncotarget*, vol. 9, no. 2, pp. 1906–1914, 2018, doi: 10.18632/oncotarget.22316.

[27] E. E. C. de Jong *et al.*, "Applicability of a prognostic CT-based radiomic signature model trained on stage I-III non-small cell lung cancer in stage IV non-small cell lung cancer.," *Lung Cancer*, vol. 124, pp. 6–11, 2018, doi: 10.1016/j.lungcan.2018.07.023.

[28] Y. Zhang, A. Oikonomou, A. Wong, M. A. Haider, and F. Khalvati, "Radiomics-based Prognosis Analysis for Non-Small Cell Lung Cancer," *Sci. Rep.*, vol. 7, p. 46349, 2017, doi: 10.1038/srep46349 CO - SRCEC3.

[29] H. Li *et al.*, "CT-Based Radiomic Signature as a Prognostic Factor in Stage IV ALK-Positive Non-small-cell Lung Cancer Treated With TKI Crizotinib: A Proof-of-Concept Study.," *Front. Oncol.*, vol. 10, p. 57, 2020, doi: 10.3389/fonc.2020.00057.

[30] H. Wei *et al.*, "Application of computed tomography-based radiomics signature analysis in the prediction of the response of small cell lung cancer patients to first-line chemotherapy," *Exp. Ther. Med.*, pp. 3621–3629, 2019, doi: 10.3892/etm.2019.7357.

[31] M. Amini *et al.*, "Multi-Level PET and CT Fusion Radiomics-based Survival Analysis of NSCLC Patients," *2020 IEEE Nucl. Sci. Symp. Med. Imaging Conf. NSS/MIC 2020*, pp. 0–3, 2020, doi: 10.1109/NSS/MIC42677.2020.9507759.

[32] M. Amini *et al.*, "Multi-level multi-modality (PET and CT) fusion radiomics: Prognostic modeling for non-small cell lung carcinoma," *Phys. Med. Biol.*, vol. 66, no. 20, 2021, doi: 10.1088/1361-6560/ac287d.

[33] P. Forouzannezhad *et al.*, "Multitask Learning Radiomics on Longitudinal Imaging to Predict Survival Outcomes following Risk-Adaptive Chemoradiation for Non-Small Cell Lung Cancer," *Cancers (Basel).*, vol. 14, no. 5, 2022, doi: 10.3390/cancers14051228.

[34] D. G. Beer *et al.*, "Gene-expression profiles predict survival of patients with lung adenocarcinoma," *Nat. Med.*, vol. 8, no. 8, pp. 816–824, 2002, doi: 10.1038/nm733.

[35] K. Shedden *et al.*, "Gene expression-based survival prediction in lung adenocarcinoma: A multi-site, blinded validation study," *Nat. Med.*, vol. 14, no. 8, pp. 822–827, 2008, doi: 10.1038/nm.1790.

[36] C. Q. Zhu *et al.*, "Prognostic and predictive gene signature for adjuvant chemotherapy in resected non-small-cell lung cancer," *J. Clin. Oncol.*, vol. 28, no. 29, pp. 4417–4424, 2010, doi: 10.1200/JCO.2009.26.4325.

[37] D. Watza *et al.*, "Prognostic modeling of the immune-centric transcriptome reveals interleukin signaling candidates contributing to differential patient outcomes," *Carcinogenesis*, vol. 39, no. 12, pp. 1447–1454, 2018, doi: 10.1093/carcin/bgy119.

[38] R. Li *et al.*, "Identification of a Prognostic Model Based on Immune-Related Genes of Lung Squamous Cell Carcinoma," *Front. Oncol.*, vol. 10, no. September, 2020, doi: 10.3389/fonc.2020.01588.

[39] Y. Chen *et al.*, "Transcriptional characterization of the tumor immune microenvironment and its prognostic value for locally advanced lung adenocarcinoma in a Chinese population," *Cancer Manag. Res.*, vol. 11, pp. 9165–9173, 2019, doi: 10.2147/CMAR.S209571.

[40] A. Hosny *et al.*, "Deep learning for lung cancer prognostication: A retrospective multi-cohort radiomics study," *PLoS Med.*, vol. 15, no. 11, pp. 1–25, 2018, doi: 10.1371/journal.pmed.1002711.

[41] Y. Wu, J. Ma, X. Huang, S. H. Ling, and S. Weidong Su, "DeepMMSA: A Novel Multimodal Deep Learning Method for Non-small Cell Lung Cancer Survival Analysis," in *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 2021, pp. 1468–1472, doi: 10.1109/SMC52423.2021.9658891.

[42] B.-X. He *et al.*, "Deep learning for predicting immunotherapeutic efficacy in advanced non-small cell lung cancer patients: a retrospective study combining progression-free survival risk and overall survival risk," *Transl. Lung Cancer Res.*, vol. 0, no. 0, pp. 0–0, 2021, doi: 10.21037/tlcr-22-244.

[43] R. Paul *et al.*, "Deep Feature Transfer Learning in Combination with Traditional Features Predicts Survival among Patients with Lung Adenocarcinoma," *Tomography*, vol. 2, no. 4, pp. 388–395, 2016, doi: 10.18383/j.tom.2016.00211.

[44] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: delving deep into convolutional nets," in *Proceedings of the British Machine Vision Conference 2014*, 2019, pp. 1–12.

[45] S. Bakr *et al.*, "A radiogenomic dataset of non-small cell lung cancer," *Sci. Data*, vol. 5, pp. 1–9, 2018, doi: 10.1038/sdata.2018.202.

[46] D. L. Rubin, M. U. Akdogan, C. Altindag, and E. Alkim, "Epad: An image annotation and analysis platform for quantitative imaging," *Tomography*, vol. 5, no. 1, pp. 170–183, 2019, doi: 10.18383/j.tom.2018.00055.

[47] D. S. Channin, P. Mongkolwat, V. Kleper, and D. L. Rubin, "The annotation and image mark-up project," *Radiology*, vol. 253, no. 3, pp. 590–592, 2009, doi: 10.1148/radiol.2533090135.

[48] K. Smith, "NSCLC Radiogenomics," 2021. https://wiki.cancerimagingarchive.net/display/Public/NSCLC+Radiogenomics (accessed Apr.

20, 2022).

[49] J. J. M. Van Griethuysen *et al.*, "Computational radiomics system to decode the radiographic phenotype," *Cancer Res.*, vol. 77, no. 21, pp. e104–e107, 2017, doi: 10.1158/0008-5472.CAN-17-0339.

[50] S. Bark, O. Gevaert, and S. . Plevritis, "Identification of Relationships between Molecular and Imaging Phenotypes in Non-Small Cell Lung Cancer using Radiogenomics Map," 2018. https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE103584 (accessed Apr. 20, 2022).

[51] D. Shen, G. Wu, and H.-I. Suk, "Deep Learning in Medical Image Analysis," *Annu Rev Biomed Eng.*, vol. 19, pp. 221–248, 2017, doi: 10.1146/annurev-bioeng-071516-044442.

[52] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 1717–1724, 2014, doi: 10.1109/CVPR.2014.222.

[53] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255, doi: 10.1109/CVPR.2009.5206848.

[54] M. Hussain, J. J. Bird, and D. R. Faria, "A Study on CNN Transfer Learning for Image Classification BT - Advances in Computational Intelligence Systems," 2019, pp. 191–202.

[55] R. Mormont, P. Geurts, and R. Maree, "Comparison of deep transfer learning strategies for digital pathology," *IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work.*, vol. 2018-June, no. June, pp. 2343–2352, 2018, doi: 10.1109/CVPRW.2018.00303.

[56] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[57] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning Transferable Architectures for Scalable Image Recognition," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 8697–8710, 2018, doi: 10.1109/CVPR.2018.00907.

[58] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 2261–2269, 2017, doi: 10.1109/CVPR.2017.243.

[59] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1800–1807, 2017, doi: 10.1109/CVPR.2017.195.

[60] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[61] A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” 2017, [Online]. Available: http://arxiv.org/abs/1704.04861.

[62] Illumina, “High-impact discovery through gene expression and regulation,” pp. 1–26, 2022.

[63] Z. Lai and H. Deng, “Medical image classification based on deep features extracted by deep model and statistic feature fusion with multilayer perceptron,” *Comput. Intell. Neurosci.*, vol. 2018, 2018, doi: 10.1155/2018/2061516.

[64] A. Kissopoulou, J. Jonasson, T. L. Lindahl, and A. Osman, “Next generation sequencing analysis of human platelet polyA+ mRNAs and rRNA-depleted total RNA,” *PLoS One*, vol. 8, no. 12, 2013, doi: 10.1371/journal.pone.0081809.

[65] R. El Ayachy *et al.*, “The Role of Radiomics in Lung Cancer: From Screening to Treatment and Follow-Up,” *Front. Oncol.*, vol. 11, no. May, pp. 1–14, 2021, doi: 10.3389/fonc.2021.603595.

[66] S. Wu, “A review on coarse warranty data and analysis,” *Reliab. Eng. Syst. Saf.*, vol. 114, pp. 1–11, 2013, doi: https://doi.org/10.1016/j.ress.2012.12.021.

[67] K. K. Al-jabery, T. Obafemi-Ajayi, G. R. Olbricht, and D. C. Wunsch II, *Computational Learning Approaches to Data Analytics in Biomedical Applications*. 2020.

[68] V. Kumar, “Feature Selection: A literature Review,” *Smart Comput. Rev.*, vol. 4, no. 3, 2014, doi: 10.6029/smartcr.2014.03.007.

[69] G. Chandrashekar and F. Sahin, “A Survey on Feature Selection methods,” *Comput. Electr. Eng.*, vol. 40, no. 1, pp. 16–28, 2014, doi: https://doi.org/10.1016/j.compeleceng.2013.11.024.

[70] I. Guyon and A. Elisseeff, “An Introduction to Variable and Feature Selection,” *J. Mach. Learn. Res. 3*, vol. 3, pp. 1157–1182, 2003.

[71] J. Li *et al.*, “Feature Selection: A Data Perspective,” *ACM Comput. Surv.*, vol. 50, no. 6, Dec. 2017, doi: 10.1145/3136625.

[72] E. Ostertagová and O. Ostertag, “Methodology and Application of Oneway ANOVA,” *Am. J. Mech. Eng.*, vol. 1, no. 7, pp. 256–261, 2013, doi: 10.12691/ajme-1-7-21.

[73] R. Muthukrishnan and R. Rohini, “LASSO: A feature selection technique in predictive modeling for machine learning,” *2016 IEEE Int. Conf. Adv. Comput. Appl. ICACA 2016*, pp. 18–20, 2016, doi: 10.1109/ICACA.2016.7887916.

[74] T. G. Clark, M. J. Bradburn, S. B. Love, and D. G. Altman, “Survival Analysis Part I: Basic concepts and first analyses,” *Br. J. Cancer*, vol. 89, no. 2, pp. 232–238, 2003, doi: 10.1038/sj.bjc.6601118.

[75] D. G. Kleinbaum and M. Klein, *Statistics for Biology and Health - Survival Analysis: A Self-Learning Text*, Third Edit. 2012.

[76] M. Tableman and J. S. Kim, *Survival Analysis Using S: Analysis of Time-to-Event Data*, First edit.

Chapman and Hall/CRC, 2003.

[77] E. L. Kaplan and P. Meier, “Nonparametric Estimation from Incomplete Observations,” *J. Am. Stat. Assoc.*, vol. 53, no. 282, pp. 457–481, May 1958, doi: 10.2307/2281868.

[78] D. R. Cox, “Regression Models and Life-Tables,” *J. R. Stat. Soc. Ser. B*, vol. 34, no. 2, pp. 187–202, Jan. 1972, doi: https://doi.org/10.1111/j.2517-6161.1972.tb00899.x.

[79] S. Pölsterl, N. Navab, and A. Katouzian, “Fast training of support vector machines for survival analysis,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9285, pp. 243–259, 2015, doi: 10.1007/978-3-319-23525-7_15.

[80] H. Ishwaran, U. B. Kogalur, E. H. Blackstone, and M. S. Lauer, “Random survival forests,” *Ann. Appl. Stat.*, vol. 2, no. 3, pp. 841–860, 2008, doi: 10.1214/08-AOAS169.

[81] F. E. J. Harrell, K. L. Lee, and D. B. Mark, “Multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors.,” *Stat. Med.*, vol. 15, no. 4, pp. 361–387, Feb. 1996, doi: 10.1002/(SICI)1097-0258(19960229)15:4<361::AID-SIM168>3.0.CO;2-4.

[82] S. Pölsterl, “Scikit-survival: A library for time-to-event analysis built on top of scikit-learn,” *J. Mach. Learn. Res.*, vol. 21, pp. 1–6, 2020.

# Appendices

## Code A1

```
import os
import sys
import pandas as pd
import numpy as np
import pickle as pkl
import matplotlib.pyplot as plt
import statistics
from sklearn.feature_selection import SelectFromModel
from sklearn.feature_selection import VarianceThreshold
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import f_classif as fc
from sklearn.feature_selection import SelectKBest as kbest
from sksurv.metrics import concordance_index_censored
import scipy.stats as st
from sklearn.model_selection import StratifiedKFold
from sksurv.ensemble import RandomSurvivalForest

#load radiomics data
radiomics_pd =
pkl.load(open("features_for_overall_survival_median/radiomics_pandas.pkl", "rb"))

#load deep feature
deep_avg =
pkl.load(open("features_for_overall_survival_median/deep_features_avg.pkl", "rb"))
deep_max =
pkl.load(open("features_for_overall_survival_median/deep_features_max.pkl", "rb"))

#merge deep features vectors (avg and max) to a single feature vector
deep={}
for net in list(deep_avg.keys()):
    deep[net] = deep_avg[net].merge(deep_max[net], left_index = True, right_index
= True)

# import labels
with open('features_for_overall_survival_median/days_OS_dictionary.pkl', 'rb') as
f:
    OS_days = pkl.load(f)
    OS_days_pd= pd.DataFrame([OS_days.keys(), OS_days.values()]).T
    OS_days_pd.columns= ['Patient', 'Survival_in_days']

with open('features_for_overall_survival_median/binary_OS_dictionary.pkl', 'rb')
as f:
    OS_binary = pkl.load(f)
    OS_binary_pd= pd.DataFrame([OS_binary.keys(), OS_binary.values()]).T
```

```
    OS_binary_pd.columns= ['Patient', 'Status']

final_OS = pd.merge(OS_binary_pd, OS_days_pd, on="Patient", how="left")
final_OSS = final_OS.set_index('Patient')

print("Data cleaning with VarianceThreshold")
thresholder_rad = VarianceThreshold(threshold=0.0)
thresholder_deep = VarianceThreshold(threshold=0.0)

radiomics_selected = thresholder_rad.fit(radiomics_pd)
mask_rad = thresholder_rad.get_support()
radiomics_ = radiomics_pd.loc[:,mask_rad]

print("Z-normalization")
radiomics_transformed = StandardScaler().fit_transform(radiomics_)

# Transform radiomics data to DataFrame
radiomics = pd.DataFrame(data=radiomics_transformed, index=radiomics_.index,
columns=radiomics_.columns)

def apply_feature_selection(df, labels, cutoff_pvalue=0.05):
    X=[]
    for key in list(df.index):
        X.append(df.loc[key])
    X = np.array(X)
    y = np.hstack(labels)

    selector = kbest(fc, k="all")
    best_features = selector.fit_transform(X, y)
    f_scores, p_values = fc(X, y)
    critical_value = st.f.ppf(q=1-cutoff_pvalue, dfn=len(np.unique(y))-1,
dfd=len(y)-len(np.unique(y)))

    best_indices=[]
    for index, p_value in enumerate(p_values):
        if f_scores[index]>critical_value and p_value<cutoff_pvalue:
            best_indices.append(index)
    print("Best ANOVA features:" + str(len(best_indices)))

    if len(best_features)>0:
        best_columns = np.array(list(df.columns))[best_indices]
        best_features = np.array(list(df[best_columns].values))
    else:
        best_columns = np.array(list(df.columns))
        best_features = np.array(list(df.values))

    try:
        sel_ = SelectFromModel(Lasso(alpha=0.01))
        sel_.fit(best_features, y)
        selected_features_bool = sel_.get_support()
        final_selected=[]
        final_features=[]
        for index, feat_id in enumerate(best_columns):
```

```
            if selected_features_bool[index]:
                final_selected.append(feat_id)
        final_selected = np.array(final_selected)
    except:
        print("No features left after Lasso")
        final_selected = best_columns

    print("Best Lasso features: "+str(len(final_selected)))
    return final_selected


pids = np.array(list(OS_binary.keys()),dtype=str)
f_labels = np.array(list(OS_binary.values()))
sss = StratifiedKFold(n_splits=4,shuffle=True)
kfolds = []
for train_index, test_index in sss.split(pids,f_labels):
    kfolds.append([pids[train_index],pids[test_index]])

for index,split in enumerate(kfolds):
    print(split[0])

results = {}
for model_name in deep.keys():
    print(model_name)
    deep_selected = thresholder_deep.fit(deep[model_name])
    mask_deep = thresholder_deep.get_support()
    deep_ = deep[model_name].loc[:,mask_deep]

    deep_ = StandardScaler().fit_transform(deep[model_name])
    deep_df = pd.DataFrame(data=deep_,index=deep[model_name].index,
columns=deep[model_name].columns)

    #Keep specific patients with known labels
    patients_split = []
    for patients in list(radiomics_pd.index):
        if patients in list(deep_df.index):
            patients_split.append(patients)

    deep_final = deep_df.loc[patients_split]

    Concordance_index = []
    for index,split in enumerate(kfolds):
        tr_split = []
        tst_split = []
        for key in list(radiomics.index):
            if key in list(split[0]):
                tr_split.append(key)
            elif key in list(split[1]):
                tst_split.append(key)

        binary_labels = []
        for pid in list(radiomics.loc[tr_split].index):
            if pid in final_OS.values:
```

```
                binary_labels.append(OS_binary[pid])


        days_labels = []
        for pid in list(radiomics.loc[tr_split].index):
            if pid in final_OS.values:
                days_labels.append(OS_days[pid])

        try:
            radiomics_feat = apply_feature_selection(radiomics.loc[tr_split],
binary_labels)
            deep_feat = apply_feature_selection(deep_final.loc[tr_split],
binary_labels)
        except:
            print('something went wrong')
            continue

        path="results/RF_rad_deep/"+model_name+"_RF_nsplit"+str(index)
        os.mkdir(path)
        pd.DataFrame(radiomics_feat).to_csv(path+"/selected_radiomics.csv")
        pd.DataFrame(deep_feat).to_csv(path+"/selected_deep.csv")

        selected_radiomics = {}
        for key in list(radiomics.index):
            selected_radiomics[key] =
radiomics[radiomics_feat].loc[key].to_numpy()

        selected_deep = {}
        for key in list(deep_final.index):
            selected_deep[key] = deep_final[deep_feat].loc[key].to_numpy()


        combined_patterns_rad_deep = {}
        for key in list(selected_deep.keys()):
            try:
                combined_patterns_rad_deep[key] =
np.concatenate((selected_radiomics[key], selected_deep[key]))
            except:
                print(key)
                continue


        x_pd=pd.DataFrame.from_dict(combined_patterns_rad_deep, orient='index')
        X_train = x_pd.loc[tr_split]
        X_test = x_pd.loc[tst_split]
        y_train = final_OSS.loc[tr_split]
        y_test = final_OSS.loc[tst_split]

        # give structure to y
        struct_arr_train =
y_train.astype({'Status':'?','Survival_in_days':'<f8'}).dtypes
        y_train_np = np.array([tuple(x) for x in y_train.values],
dtype=list(zip(struct_arr_train.index,struct_arr_train)))
```

```
        struct_arr_test =
y_test.astype({'Status':'?','Survival_in_days':'<f8'}).dtypes
        y_test_np = np.array([tuple(x) for x in y_test.values],
dtype=list(zip(struct_arr_test.index,struct_arr_test)))

        rsf = RandomSurvivalForest(n_estimators=1000,
                                   min_samples_split=10,
                                   min_samples_leaf=6,
                                   n_jobs=-1)


        rsf.fit(X_train,y_train_np)

        score_ = rsf.score(X_test,y_test_np)

        print(score_)
        Concordance_index.append(score_)

        pred = rsf.predict(X_test)
        predictions = np.round(pred, 3)
        pd.DataFrame(predictions).to_csv(path+"/predictions_RF_rad_deep.csv")
        pd.DataFrame(y_test_np).to_csv(path+"/labels_test_RF_rad_deep.csv")

        surv = rsf.predict_survival_function(X_test, return_array=True)
        hazard = rsf.predict_cumulative_hazard_function(X_test, return_array=True)

        for i, s in enumerate(surv):
            plt.step(rsf.event_times_, s, where="post", label=str(i))
        plt.ylabel("Survival probability")
        plt.xlabel("Time in days")
        plt.legend(X_test.index)
        plt.title("CI %.4f" % score_)
        plt.grid(True)
        plt.savefig(path+"/plot_performance_model for" + model_name + "and nsplit"
+ str(index) + ".png",dpi=300)
        plt.clf()

        for i, s in enumerate(hazard):
            plt.step(rsf.event_times_, s, where="post", label=str(i))
        plt.ylabel("Cumulative hazard")
        plt.xlabel("Time in days")
        plt.legend(X_test.index)
        plt.title("CI %.4f" % score_)
        plt.grid(True)
        plt.savefig(path+"/plot_performance_hazard_model for" + model_name + "and
nsplit" + str(index) + ".png",dpi=300)
        plt.clf()

    try:
        print('List of possible CI:', Concordance_index)
        print('\nMaximum CI That can be obtained from this model
is:',max(Concordance_index))
        print('\nMinimum CI:',min(Concordance_index))
```

```
        print('\nMean CI:',statistics.mean(Concordance_index))
        print('\nStandard Deviation is:', statistics.stdev(Concordance_index))
    except:
        print('only one CI points')
        continue

    results["rad_deep for model "+model_name] = pd.Series({"Maximum
CI":max(Concordance_index),"Minimum CI":min(Concordance_index),"Overall
CI":statistics.mean(Concordance_index),"Standard
Deviation":statistics.stdev(Concordance_index)})

final_results = pd.DataFrame.from_dict(results, orient="index")
final_results.to_csv("results/RF_rad_deep/results_RF_rad_deep.csv")
```

## Code A2

```
import os
import sys
import pandas as pd
import numpy as np
import pickle as pkl
import matplotlib.pyplot as plt
import statistics
from sklearn.feature_selection import SelectFromModel
from sklearn.feature_selection import VarianceThreshold
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import f_classif as fc
from sklearn.feature_selection import SelectKBest as kbest
from sksurv.metrics import concordance_index_censored
from sksurv.svm import FastSurvivalSVM
import scipy.stats as st
from sklearn.model_selection import StratifiedKFold

#load radiomics data
radiomics_pd =
pkl.load(open("features_for_overall_survival_median/radiomics_pandas.pkl", "rb"))

#load deep feature
deep_avg =
pkl.load(open("features_for_overall_survival_median/deep_features_avg.pkl", "rb"))
deep_max =
pkl.load(open("features_for_overall_survival_median/deep_features_max.pkl", "rb"))

#merge deep features vectors (avg and max) to a single feature vector
deep={}
for net in list(deep_avg.keys()):
    deep[net] = deep_avg[net].merge(deep_max[net], left_index = True, right_index
= True)
```

```
# import labels
with open("features_for_overall_survival_median/days_OS_dictionary.pkl", 'rb') as
f:
    OS_days = pkl.load(f)
    OS_days_pd= pd.DataFrame([OS_days.keys(), OS_days.values()]).T
    OS_days_pd.columns= ['Patient', 'Survival_in_days']

with open("features_for_overall_survival_median/binary_OS_dictionary.pkl", 'rb')
as f:
    OS_binary = pkl.load(f)
    OS_binary_pd= pd.DataFrame([OS_binary.keys(), OS_binary.values()]).T
    OS_binary_pd.columns= ['Patient', 'Status']

final_OS = pd.merge(OS_binary_pd, OS_days_pd, on="Patient", how="left")
final_OSS = final_OS.set_index('Patient')

print("Data cleaning with VarianceThreshold")
thresholder_rad = VarianceThreshold(threshold=0.0)
thresholder_deep = VarianceThreshold(threshold=0.0)

radiomics_selected = thresholder_rad.fit(radiomics_pd)
mask_rad = thresholder_rad.get_support()
radiomics_ = radiomics_pd.loc[:,mask_rad]

print("Z-normalization")
radiomics_transformed = StandardScaler().fit_transform(radiomics_)

# Transform radiomics data to DataFrame
radiomics = pd.DataFrame(data=radiomics_transformed, index=radiomics_.index,
columns=radiomics_.columns)

def apply_feature_selection(df, labels, cutoff_pvalue=0.05):
    X=[]
    for key in list(df.index):
        X.append(df.loc[key])
    X = np.array(X)
    y = np.hstack(labels)

    selector = kbest(fc, k="all")
    best_features = selector.fit_transform(X, y)
    f_scores, p_values = fc(X, y)
    critical_value = st.f.ppf(q=1-cutoff_pvalue, dfn=len(np.unique(y))-1,
dfd=len(y)-len(np.unique(y)))

    best_indices=[]
    for index, p_value in enumerate(p_values):
        if f_scores[index]>critical_value and p_value<cutoff_pvalue:
            best_indices.append(index)
    print("Best ANOVA features:" + str(len(best_indices)))

    if len(best_features)>0:
        best_columns = np.array(list(df.columns))[best_indices]
        best_features = np.array(list(df[best_columns].values))
```

```
    else:
        best_columns = np.array(list(df.columns))
        best_features = np.array(list(df.values))

    try:
        sel_ = SelectFromModel(Lasso(alpha=0.01))
        sel_.fit(best_features, y)
        selected_features_bool = sel_.get_support()
        final_selected=[]
        final_features=[]
        for index, feat_id in enumerate(best_columns):
            if selected_features_bool[index]:
                final_selected.append(feat_id)
        final_selected = np.array(final_selected)
    except:
        print("No features left after Lasso")
        final_selected = best_columns

    print("Best Lasso features: "+str(len(final_selected)))

    return final_selected

pids = np.array(list(OS_binary.keys()),dtype=str)
f_labels = np.array(list(OS_binary.values()))
sss = StratifiedKFold(n_splits=4,shuffle=True)
kfolds = []
for train_index, test_index in sss.split(pids,f_labels):
    kfolds.append([pids[train_index],pids[test_index]])

for index,split in enumerate(kfolds):
    print(split[0])

results = {}
for model_name in deep.keys():
    print(model_name)
    deep_selected = thresholder_deep.fit(deep[model_name])
    mask_deep = thresholder_deep.get_support()
    deep_ = deep[model_name].loc[:,mask_deep]

    deep_ = StandardScaler().fit_transform(deep[model_name])
    deep_df = pd.DataFrame(data=deep_,index=deep[model_name].index,
columns=deep[model_name].columns)

    #Keep specific patients with known labels
    patients_split = []
    for patients in list(radiomics_pd.index):
        if patients in list(deep_df.index):
            patients_split.append(patients)

    deep_final = deep_df.loc[patients_split]

    Concordance_index = []
    for index,split in enumerate(kfolds):
```

```
        tr_split = []
        tst_split = []
        for key in list(radiomics.index):
            if key in list(split[0]):
                tr_split.append(key)
            elif key in list(split[1]):
                tst_split.append(key)

        binary_labels = []
        for pid in list(radiomics.loc[tr_split].index):
            if pid in final_OS.values:
                binary_labels.append(OS_binary[pid])

        days_labels = []
        for pid in list(radiomics.loc[tr_split].index):
            if pid in final_OS.values:
                days_labels.append(OS_days[pid])

        try:
            radiomics_feat = apply_feature_selection(radiomics.loc[tr_split],
binary_labels)
            deep_feat = apply_feature_selection(deep_final.loc[tr_split],
binary_labels)
        except:
            print('something went wrong')
            continue

        path="results/SVM_rad_deep/"+model_name+"_SVM_nsplit"+str(index)
        os.mkdir(path)
        pd.DataFrame(radiomics_feat).to_csv(path+"/selected_radiomics.csv")
        pd.DataFrame(deep_feat).to_csv(path+"/selected_deep.csv")

        selected_radiomics = {}
        for key in list(radiomics.index):
            selected_radiomics[key] =
radiomics[radiomics_feat].loc[key].to_numpy()

        selected_deep = {}
        for key in list(deep_final.index):
            selected_deep[key] = deep_final[deep_feat].loc[key].to_numpy()

        combined_patterns_rad_deep = {}
        for key in list(selected_deep.keys()):
            try:
                combined_patterns_rad_deep[key] =
np.concatenate((selected_radiomics[key], selected_deep[key]))
            except:
                print(key)
                continue

        x_pd=pd.DataFrame.from_dict(combined_patterns_rad_deep, orient='index')
        X_train = x_pd.loc[tr_split]
        X_test = x_pd.loc[tst_split]
```

```
        y_train = final_OSS.loc[tr_split]
        y_test = final_OSS.loc[tst_split]

        # give structure to y
        struct_arr_train =
y_train.astype({'Status':'?','Survival_in_days':'<f8'}).dtypes
        y_train_np = np.array([tuple(x) for x in y_train.values],
dtype=list(zip(struct_arr_train.index,struct_arr_train)))
        struct_arr_test =
y_test.astype({'Status':'?','Survival_in_days':'<f8'}).dtypes
        y_test_np = np.array([tuple(x) for x in y_test.values],
dtype=list(zip(struct_arr_test.index,struct_arr_test)))

        estimator = FastSurvivalSVM(alpha=0.1, max_iter=1000, tol=1e-5,
random_state=0)
        estimator.fit(X_train,y_train_np)
        score_ = estimator.score(X_test,y_test_np)

        print(score_)
        Concordance_index.append(score_)

        pred = estimator.predict(X_test)
        predictions = np.round(pred, 3)
        pd.DataFrame(predictions).to_csv(path+"/predictions_SVM_rad_deep.csv")
        pd.DataFrame(y_test_np).to_csv(path+"/labels_test_SVM_rad_deep.csv")

    try:
        print('List of possible CI:', Concordance_index)
        print('\nMaximum CI That can be obtained from this model
is:',max(Concordance_index))
        print('\nMinimum CI:',min(Concordance_index))
        print('\nMean CI:',statistics.mean(Concordance_index))
        print('\nStandard Deviation is:', statistics.stdev(Concordance_index))
    except:
        print('only one CI points')
        continue

    results["rad_deep for model "+model_name] = pd.Series({"Maximum
CI":max(Concordance_index),"Minimum CI":min(Concordance_index),"Overall
CI":statistics.mean(Concordance_index),"Standard
Deviation":statistics.stdev(Concordance_index)})

final_results = pd.DataFrame.from_dict(results, orient="index")
final_results.to_csv("results/SVM_rad_deep/final_results_SVM_rad_deep.csv")
```

## Code B1

```
import os
import sys
import pandas as pd
import numpy as np
import pickle as pkl
import matplotlib.pyplot as plt
import statistics
from sklearn.feature_selection import SelectFromModel
from sklearn.feature_selection import VarianceThreshold
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import f_classif as fc
from sklearn.feature_selection import SelectKBest as kbest
from sksurv.metrics import concordance_index_censored
import scipy.stats as st
from sklearn.model_selection import StratifiedKFold
from sksurv.ensemble import RandomSurvivalForest



#load radiomics and transcriptomics data
radiomics_pd =
pkl.load(open("features_for_overall_survival_median/radiomics_pandas.pkl", "rb"))
transcriptomics_pd =
pkl.load(open("features_for_overall_survival_median/transcriptomics_pandas.pkl","r
b"))

#load deep feature
deep_avg =
pkl.load(open("features_for_overall_survival_median/deep_features_avg.pkl", "rb"))
deep_max =
pkl.load(open("features_for_overall_survival_median/deep_features_max.pkl", "rb"))

#merge deep features vectors (avg and max) to a single feature vector
deep={}
for net in list(deep_avg.keys()):
    deep[net] = deep_avg[net].merge(deep_max[net], left_index = True, right_index
= True)

# import labels
with open('features_for_overall_survival_median/days_OS_dictionary.pkl', 'rb') as
f:
    OS_days = pkl.load(f)
    OS_days_pd= pd.DataFrame([OS_days.keys(), OS_days.values()]).T
    OS_days_pd.columns= ['Patient', 'Survival_in_days']

with open('features_for_overall_survival_median/binary_OS_dictionary.pkl', 'rb')
as f:
    OS_binary = pkl.load(f)
    OS_binary_pd= pd.DataFrame([OS_binary.keys(), OS_binary.values()]).T
    OS_binary_pd.columns= ['Patient', 'Status']

final_OS = pd.merge(OS_binary_pd, OS_days_pd, on="Patient", how="left")
```

```
final_OSS = final_OS.set_index('Patient')

print("Data cleaning with VarianceThreshold")
thresholder_rad = VarianceThreshold(threshold=0.0)
thresholder_tran = VarianceThreshold(threshold=0.0)
thresholder_deep = VarianceThreshold(threshold=0.0)

radiomics_selected = thresholder_rad.fit(radiomics_pd)
transcriptomics_selected = thresholder_tran.fit(transcriptomics_pd)
mask_rad = thresholder_rad.get_support()
mask_tran = thresholder_tran.get_support()
radiomics_ = radiomics_pd.loc[:,mask_rad]
transcriptomics_ = transcriptomics_pd.loc[:,mask_tran]

print("Z-normalization")
radiomics_transformed = StandardScaler().fit_transform(radiomics_)
transcriptomics_transformed = StandardScaler().fit_transform(transcriptomics_)

# Transform radiomics, transcriptomics data to DataFrame
radiomics = pd.DataFrame(data=radiomics_transformed, index=radiomics_.index,
columns=radiomics_.columns)
transcriptomics =
pd.DataFrame(data=transcriptomics_transformed,index=transcriptomics_.index,
columns=transcriptomics_.columns)

def apply_feature_selection(df, labels, cutoff_pvalue=0.05):
    X=[]
    for key in list(df.index):
        X.append(df.loc[key])
    X = np.array(X)
    y = np.hstack(labels)

    selector = kbest(fc, k="all")
    best_features = selector.fit_transform(X, y)
    f_scores, p_values = fc(X, y)
    critical_value = st.f.ppf(q=1-cutoff_pvalue, dfn=len(np.unique(y))-1,
dfd=len(y)-len(np.unique(y)))

    best_indices=[]
    for index, p_value in enumerate(p_values):
        if f_scores[index]>critical_value and p_value<cutoff_pvalue:
            best_indices.append(index)
    print("Best ANOVA features:" + str(len(best_indices)))

    if len(best_features)>0:
        best_columns = np.array(list(df.columns))[best_indices]
        best_features = np.array(list(df[best_columns].values))
    else:
        best_columns = np.array(list(df.columns))
        best_features = np.array(list(df.values))

    try:
        sel_ = SelectFromModel(Lasso(alpha=0.01))
```

```
        sel_.fit(best_features, y)
        selected_features_bool = sel_.get_support()
        final_selected=[]
        final_features=[]
        for index, feat_id in enumerate(best_columns):
            if selected_features_bool[index]:
                final_selected.append(feat_id)
        final_selected = np.array(final_selected)
    except:
        print("No features left after Lasso")
        final_selected = best_columns

    print("Best Lasso features: "+str(len(final_selected)))
    return final_selected


pids = np.array(list(OS_binary.keys()),dtype=str)
f_labels = np.array(list(OS_binary.values()))
sss = StratifiedKFold(n_splits=4,shuffle=True)
kfolds = []
for train_index, test_index in sss.split(pids,f_labels):
    kfolds.append([pids[train_index],pids[test_index]])

for index,split in enumerate(kfolds):
    print(split[0])

results = {}
for model_name in deep.keys():
    print(model_name)
    deep_selected = thresholder_deep.fit(deep[model_name])
    mask_deep = thresholder_deep.get_support()
    deep_ = deep[model_name].loc[:,mask_deep]

    deep_ = StandardScaler().fit_transform(deep[model_name])
    deep_df = pd.DataFrame(data=deep_,index=deep[model_name].index,
columns=deep[model_name].columns)

    #Keep specific patients with known labels
    patients_split = []
    for patients in list(radiomics_pd.index):
        if patients in list(deep_df.index):
            patients_split.append(patients)

    deep_final = deep_df.loc[patients_split]

    Concordance_index = []
    for index,split in enumerate(kfolds):
        tr_split = []
        tst_split = []
        for key in list(radiomics.index):
            if key in list(split[0]):
                tr_split.append(key)
            elif key in list(split[1]):
```

```
                tst_split.append(key)

        binary_labels = []
        for pid in list(radiomics.loc[tr_split].index):
            if pid in final_OS.values:
                binary_labels.append(OS_binary[pid])

        days_labels = []
        for pid in list(radiomics.loc[tr_split].index):
            if pid in final_OS.values:
                days_labels.append(OS_days[pid])

        try:
            radiomics_feat = apply_feature_selection(radiomics.loc[tr_split],
binary_labels)
            transcriptomics_feat =
apply_feature_selection(transcriptomics.loc[tr_split], binary_labels)
            deep_feat = apply_feature_selection(deep_final.loc[tr_split],
binary_labels)
        except:
            print('something went wrong')
            continue

        path="results/RF_rad_trans_deep/"+model_name+"_RF_nsplit"+str(index)
        os.mkdir(path)
        pd.DataFrame(radiomics_feat).to_csv(path+"/selected_radiomics.csv")
        pd.DataFrame(transcriptomics_feat).to_csv(path+"/selected_transcriptomics.
csv")
        pd.DataFrame(deep_feat).to_csv(path+"/selected_deep.csv")

        selected_radiomics = {}
        for key in list(radiomics.index):
            selected_radiomics[key] =
radiomics[radiomics_feat].loc[key].to_numpy()

        selected_transcriptomics = {}
        for key in list(transcriptomics.index):
            selected_transcriptomics[key] =
transcriptomics[transcriptomics_feat].loc[key].to_numpy()

        selected_deep = {}
        for key in list(deep_final.index):
            selected_deep[key] = deep_final[deep_feat].loc[key].to_numpy()


        combined_patterns_rad_trans_deep = {}
        for key in list(selected_deep.keys()):
            try:
                combined_patterns_rad_trans_deep[key] =
np.concatenate((selected_radiomics[key], selected_transcriptomics[key],
selected_deep[key]))
            except:
                print(key)
```

```
                continue

        x_pd=pd.DataFrame.from_dict(combined_patterns_rad_trans_deep,
orient='index')
        X_train = x_pd.loc[tr_split]
        X_test = x_pd.loc[tst_split]
        y_train = final_OSS.loc[tr_split]
        y_test = final_OSS.loc[tst_split]

        # give structure to y
        struct_arr_train =
y_train.astype({'Status':'?','Survival_in_days':'<f8'}).dtypes
        y_train_np = np.array([tuple(x) for x in y_train.values],
dtype=list(zip(struct_arr_train.index,struct_arr_train)))
        struct_arr_test =
y_test.astype({'Status':'?','Survival_in_days':'<f8'}).dtypes
        y_test_np = np.array([tuple(x) for x in y_test.values],
dtype=list(zip(struct_arr_test.index,struct_arr_test)))

        rsf = RandomSurvivalForest(n_estimators=1000,
                                   min_samples_split=10,
                                   min_samples_leaf=6,
                                   n_jobs=-1)


        rsf.fit(X_train,y_train_np)

        score_ = rsf.score(X_test,y_test_np)

        print(score_)
        Concordance_index.append(score_)

        pred = rsf.predict(X_test)
        predictions = np.round(pred, 3)
        pd.DataFrame(predictions).to_csv(path+"/predictions_RF_rad_trans_deep.csv"
)
        pd.DataFrame(y_test_np).to_csv(path+"/labels_test_RF_rad_trans_deep.csv")

        surv = rsf.predict_survival_function(X_test, return_array=True)
        hazard = rsf.predict_cumulative_hazard_function(X_test, return_array=True)

        for i, s in enumerate(surv):
            plt.step(rsf.event_times_, s, where="post", label=str(i))
        plt.ylabel("Survival probability")
        plt.xlabel("Time in days")
        plt.legend(X_test.index)
        plt.title("CI %.4f" % score_)
        plt.grid(True)
        plt.savefig(path+"/plot_performance_model for" + model_name + "and nsplit"
+ str(index) + ".png",dpi=300)
        plt.clf()

        for i, s in enumerate(hazard):
```

```
            plt.step(rsf.event_times_, s, where="post", label=str(i))
        plt.ylabel("Cumulative hazard")
        plt.xlabel("Time in days")
        plt.legend(X_test.index)
        plt.title("CI %.4f" % score_)
        plt.grid(True)
        plt.savefig(path+"/plot_performance_hazard_model for" + model_name + "and
nsplit" + str(index) + ".png",dpi=300)
        plt.clf()

    try:
        print('List of possible CI:', Concordance_index)
        print('\nMaximum CI That can be obtained from this model
is:',max(Concordance_index))
        print('\nMinimum CI:',min(Concordance_index))
        print('\nMean CI:',statistics.mean(Concordance_index))
        print('\nStandard Deviation is:', statistics.stdev(Concordance_index))
    except:
        print('only one CI points')
        continue

    results["rad_trans_deep for model "+model_name] = pd.Series({"Maximum
CI":max(Concordance_index),"Minimum CI":min(Concordance_index),"Overall
CI":statistics.mean(Concordance_index),"Standard
Deviation":statistics.stdev(Concordance_index)})

final_results = pd.DataFrame.from_dict(results, orient="index")
final_results.to_csv("results/RF_rad_trans_deep/results_RF_rad_trans_deep.csv")
```

## Code B2

```
import os
import sys
import pandas as pd
import numpy as np
import pickle as pkl
import matplotlib.pyplot as plt
import statistics
from sklearn.feature_selection import SelectFromModel
from sklearn.feature_selection import VarianceThreshold
from sklearn.linear_model import Lasso
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import f_classif as fc
from sklearn.feature_selection import SelectKBest as kbest
from sksurv.metrics import concordance_index_censored
from sksurv.svm import FastSurvivalSVM
import scipy.stats as st
```

```
from sklearn.model_selection import StratifiedKFold

#load radiomics and transcriptomics data
radiomics_pd =
pkl.load(open("features_for_overall_survival_median/radiomics_pandas.pkl", "rb"))
transcriptomics_pd =
pkl.load(open("features_for_overall_survival_median/transcriptomics_pandas.pkl","r
b"))

#load deep feature
deep_avg =
pkl.load(open("features_for_overall_survival_median/deep_features_avg.pkl", "rb"))
deep_max =
pkl.load(open("features_for_overall_survival_median/deep_features_max.pkl", "rb"))

#merge deep features vectors (avg and max) to a single feature vector
deep={}
for net in list(deep_avg.keys()):
    deep[net] = deep_avg[net].merge(deep_max[net], left_index = True, right_index
= True)

# import labels
with open("features_for_overall_survival_median/days_OS_dictionary.pkl", 'rb') as
f:
    OS_days = pkl.load(f)
    OS_days_pd= pd.DataFrame([OS_days.keys(), OS_days.values()]).T
    OS_days_pd.columns= ['Patient', 'Survival_in_days']

with open("features_for_overall_survival_median/binary_OS_dictionary.pkl", 'rb')
as f:
    OS_binary = pkl.load(f)
    OS_binary_pd= pd.DataFrame([OS_binary.keys(), OS_binary.values()]).T
    OS_binary_pd.columns= ['Patient', 'Status']

final_OS = pd.merge(OS_binary_pd, OS_days_pd, on="Patient", how="left")
final_OSS = final_OS.set_index('Patient')

print("Data cleaning with VarianceThreshold")
thresholder_rad = VarianceThreshold(threshold=0.0)
thresholder_tran = VarianceThreshold(threshold=0.0)
thresholder_deep = VarianceThreshold(threshold=0.0)

radiomics_selected = thresholder_rad.fit(radiomics_pd)
transcriptomics_selected = thresholder_tran.fit(transcriptomics_pd)
mask_rad = thresholder_rad.get_support()
mask_tran = thresholder_tran.get_support()
radiomics_ = radiomics_pd.loc[:,mask_rad]
transcriptomics_ = transcriptomics_pd.loc[:,mask_tran]

print("Z-normalization")
radiomics_transformed = StandardScaler().fit_transform(radiomics_)
transcriptomics_transformed = StandardScaler().fit_transform(transcriptomics_)
```

```
# Transform radiomics, transcriptomics data to DataFrame
radiomics = pd.DataFrame(data=radiomics_transformed, index=radiomics_.index,
columns=radiomics_.columns)
transcriptomics =
pd.DataFrame(data=transcriptomics_transformed,index=transcriptomics_.index,
columns=transcriptomics_.columns)

def apply_feature_selection(df, labels, cutoff_pvalue=0.05):
    X=[]
    for key in list(df.index):
        X.append(df.loc[key])
    X = np.array(X)
    y = np.hstack(labels)

    selector = kbest(fc, k="all")
    best_features = selector.fit_transform(X, y)
    f_scores, p_values = fc(X, y)
    critical_value = st.f.ppf(q=1-cutoff_pvalue, dfn=len(np.unique(y))-1,
dfd=len(y)-len(np.unique(y)))

    best_indices=[]
    for index, p_value in enumerate(p_values):
        if f_scores[index]>critical_value and p_value<cutoff_pvalue:
            best_indices.append(index)
    print("Best ANOVA features:" + str(len(best_indices)))

    if len(best_features)>0:
        best_columns = np.array(list(df.columns))[best_indices]
        best_features = np.array(list(df[best_columns].values))
    else:
        best_columns = np.array(list(df.columns))
        best_features = np.array(list(df.values))

    try:
        sel_ = SelectFromModel(Lasso(alpha=0.01))
        sel_.fit(best_features, y)
        selected_features_bool = sel_.get_support()
        final_selected=[]
        final_features=[]
        for index, feat_id in enumerate(best_columns):
            if selected_features_bool[index]:
                final_selected.append(feat_id)
        final_selected = np.array(final_selected)
    except:
        print("No features left after Lasso")
        final_selected = best_columns

    print("Best Lasso features: "+str(len(final_selected)))

    return final_selected

pids = np.array(list(OS_binary.keys()),dtype=str)
f_labels = np.array(list(OS_binary.values()))
```

```
sss = StratifiedKFold(n_splits=4,shuffle=True)
kfolds = []
for train_index, test_index in sss.split(pids,f_labels):
    kfolds.append([pids[train_index],pids[test_index]])

for index,split in enumerate(kfolds):
    print(split[0])

results = {}
for model_name in deep.keys():
    print(model_name)
    deep_selected = thresholder_deep.fit(deep[model_name])
    mask_deep = thresholder_deep.get_support()
    deep_ = deep[model_name].loc[:,mask_deep]

    deep_ = StandardScaler().fit_transform(deep[model_name])
    deep_df = pd.DataFrame(data=deep_,index=deep[model_name].index,
columns=deep[model_name].columns)

    #Keep specific patients with known labels
    patients_split = []
    for patients in list(radiomics_pd.index):
        if patients in list(deep_df.index):
            patients_split.append(patients)

    deep_final = deep_df.loc[patients_split]

    Concordance_index = []
    for index,split in enumerate(kfolds):
        tr_split = []
        tst_split = []
        for key in list(radiomics.index):
            if key in list(split[0]):
                tr_split.append(key)
            elif key in list(split[1]):
                tst_split.append(key)

        binary_labels = []
        for pid in list(radiomics.loc[tr_split].index):
            if pid in final_OS.values:
                binary_labels.append(OS_binary[pid])


        days_labels = []
        for pid in list(radiomics.loc[tr_split].index):
            if pid in final_OS.values:
                days_labels.append(OS_days[pid])

        try:
            radiomics_feat = apply_feature_selection(radiomics.loc[tr_split],
binary_labels)
            transcriptomics_feat =
apply_feature_selection(transcriptomics.loc[tr_split], binary_labels)
```

```
            deep_feat = apply_feature_selection(deep_final.loc[tr_split],
binary_labels)
        except:
            print('something went wrong')
            continue

        path="results/SVM_rad_trans_deep/"+model_name+"_SVM_nsplit"+str(index)
        os.mkdir(path)
        pd.DataFrame(radiomics_feat).to_csv(path+"/selected_radiomics.csv")
        pd.DataFrame(transcriptomics_feat).to_csv(path+"/selected_transcriptomics.
csv")
        pd.DataFrame(deep_feat).to_csv(path+"/selected_deep.csv")

        selected_radiomics = {}
        for key in list(radiomics.index):
            selected_radiomics[key] =
radiomics[radiomics_feat].loc[key].to_numpy()

        selected_transcriptomics = {}
        for key in list(transcriptomics.index):
            selected_transcriptomics[key] =
transcriptomics[transcriptomics_feat].loc[key].to_numpy()

        selected_deep = {}
        for key in list(deep_final.index):
            selected_deep[key] = deep_final[deep_feat].loc[key].to_numpy()

        combined_patterns_rad_trans_deep = {}
        for key in list(selected_deep.keys()):
            try:
                combined_patterns_rad_trans_deep[key] =
np.concatenate((selected_radiomics[key], selected_transcriptomics[key],
selected_deep[key]))
            except:
                print(key)
                continue

        x_pd=pd.DataFrame.from_dict(combined_patterns_rad_trans_deep,
orient='index')
        X_train = x_pd.loc[tr_split]
        X_test = x_pd.loc[tst_split]
        y_train = final_OSS.loc[tr_split]
        y_test = final_OSS.loc[tst_split]

        # give structure to y
        struct_arr_train =
y_train.astype({'Status':'?','Survival_in_days':'<f8'}).dtypes
        y_train_np = np.array([tuple(x) for x in y_train.values],
dtype=list(zip(struct_arr_train.index,struct_arr_train)))
        struct_arr_test =
y_test.astype({'Status':'?','Survival_in_days':'<f8'}).dtypes
        y_test_np = np.array([tuple(x) for x in y_test.values],
dtype=list(zip(struct_arr_test.index,struct_arr_test)))
```

```
        estimator = FastSurvivalSVM(alpha=0.1, max_iter=1000, tol=1e-5,
random_state=0)
        estimator.fit(X_train,y_train_np)
        score_ = estimator.score(X_test,y_test_np)

        print(score_)
        Concordance_index.append(score_)

        pred = estimator.predict(X_test)
        predictions = np.round(pred, 3)
        pd.DataFrame(predictions).to_csv(path+"/predictions_SVM_rad_trans_deep.csv
")
        pd.DataFrame(y_test_np).to_csv(path+"/labels_test_SVM_rad_trans_deep.csv")

    try:
        print('List of possible CI:', Concordance_index)
        print('\nMaximum CI That can be obtained from this model
is:',max(Concordance_index))
        print('\nMinimum CI:',min(Concordance_index))
        print('\nMean CI:',statistics.mean(Concordance_index))
        print('\nStandard Deviation is:', statistics.stdev(Concordance_index))
    except:
        print('only one CI points')
        continue

    results["rad_trans_deep for model "+model_name] = pd.Series({"Maximum
CI":max(Concordance_index),"Minimum CI":min(Concordance_index),"Overall
CI":statistics.mean(Concordance_index),"Standard
Deviation":statistics.stdev(Concordance_index)})

final_results = pd.DataFrame.from_dict(results, orient="index")
final_results.to_csv("results/SVM_rad_trans_deep/final_results_SVM_rad_trans_deep.
csv")
```