CARDIFF UNIVERSITY

SCHOOL OF MEDICINE

MASTERS DISSERTATION

New Cancer Immunotherapy Targets by Dissection of Bulk RNA-seq and T-Cell Receptor Sequence Data

---

Submitted By :

Michalis Mylonas

C1964262

Supervisor: Dr. Barbara Szomolay

Acknowledgements

I would like to personally acknowledge and thank my supervisor Dr. Barbara Szomolay and Tom Whalley for this wonderful collaboration, for their constructive feedback, guidance as well as their comments and recommendations.

I have also received a lot of guidance and assistance throughout this course and I am deeply grateful and would like to thank Dr. Richard Anney, Dr. Marian Hamshere, Dr. Robert Andrews as well as my colleagues, for their support, understanding and effort to keep me going.

Declarations

Candidate's ID number: C1964262

Candidate's surname: Mylonas

Please circle appropriate value: Mr / ~~Miss~~ / ~~Ms~~ / ~~Mrs~~ / ~~Rev~~ / ~~Dr~~ / ~~Other please specify~~ .....

Candidate's full forename: Michalis

This work has not previously been accepted in substance for any degree and is not concurrently submitted in candidature for any degree.

October 19, 2020

Sign: 

Statement One

This dissertation is being submitted in partial fulfilment of the requirements for the degree of MSc in Bioinformatics.

October 19, 2020

Sign: 

Statement Two

This dissertation is the result of my own independent work/investigation, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A Bibliography is appended.

October 19, 2020

Sign: 

Statement Three

I hereby give consent for my dissertation, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

October 19, 2020

Sign: 

Statement Four - BAR ON ACCESS APPROVED

I hereby give consent for my dissertation, if accepted, to be available for photocopying and for inter-library loans after expiry of a bar on access approved by the Graduate Development Committee.

October 19, 2020

Sign: 

Abstract

Background

Cancer is known for its ability to trick our immune system, posing as normal tissue while wreaking havoc on the body. However, what if cancer cells produced 'tells'—subtle but distinctive features that showed their true nature? In this paper we present evidence of the importance of neoantigens in tumour samples and review different methods in which such results can be produced.

Methods

Carefully screened RNA-seq data-sets to identify genomic variants. Neoantigens were predicted using machine learning methods to identify and improve existing software model.

Results

By selection and prioritisation of candidate genes, PLIN4 and CCN3 were identified as being highly expressed for neoantigen selection. This showed a high recurrence number compared to other genes.

Conclusion

In individual patients, technological advances have been used to define neoantigens and T-cells. These strategies may as well result in false positives, however, personalised treatment with a particular interest on how these neoantigens can be leveraged to improve medical interventions will certainly lead a new era of cancer treatment.

Contents

List of Figures

List of Code Listings

List of Tables

List of Abbreviations

DCs	Dendritic cells
HLA	Human Leukocyte Antigen
LR	Linear Regression
MHC	Major Histocompatibility Complex
ML	Machine Learning
RMSE	RMSE
SPORE	Specialised Research Excellence Tissue Core
TCR	T-cell receptor
TRB	T Cell Receptor Beta Locus
VPN	Virtual Private Netowrk
WGS	Whole Genome Sequencing

Chapter 1

Introduction

1.1 Cancer Immunotherapy

Immuno oncology also called as cancer immunotherapy is a form of cancer treatment that leverages the strength of the body's own immune system to prevent, regulate and destroy cancer cells (Topalian *et al.*, 2011).

There are different ways in which these treatments can be delivered. One way, is to 're-train' the immune system to identify and defend itself against specific cancer cells. Another way, is to boost immune cells to aid in the elimination of tumour tissue and lastly, it can also supply the body with the necessary means to enhance an immune response (Diamandis *et al.*, 2016).

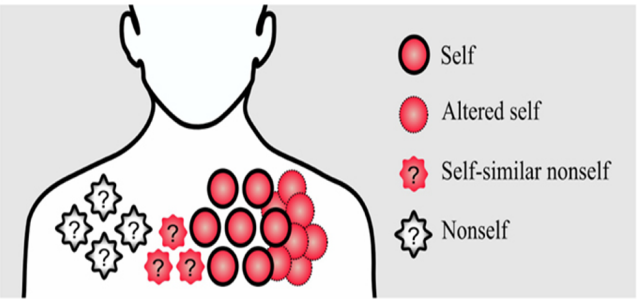
Although cancer immunotherapy comes in many forms, such as targeted adoptive cell transfer and tumour infecting viruses. The main focus of this paper is, neoantigens which are altered self peptides and can arise in tumour cells as a result of tumour mutation as shown in Figure 1.1b.

Figure 1.1a conveys different disturbances of the immune system. Firstly, pathogenic non-self disturbances can be recognised by non-self specific cells. These cells are also capable of developing pathogen-specific memory to recognise any peptides, including self-peptides. However, recognising altered self and self-similar cells is, to say the least, challenging as the immune system is unable to trigger an immune response against these threats.

In order to gain a better understanding of this field, Figure 1.1b has been provided to illustrate how neoantigens can support T-Cells in the identification of such disturbances and in extend against tumour tissue. As shown, T-Cell receptors (TCRs) are molecules located on the surface of the T-Cells and are mainly responsible for identifying harmful pathogens by actively serving as surveillance proteins (membrane-spanning proteins). Having said that, neoantigens can provide the necessary support to the immune system for identifying and eliciting an immune response against cancer tissue.

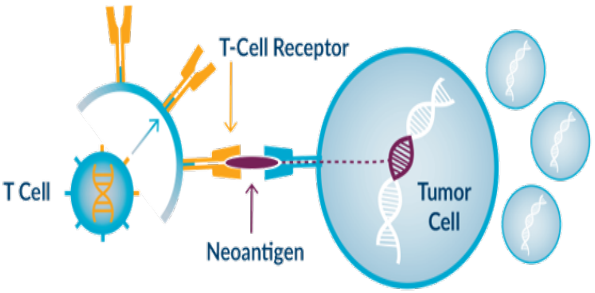
T-Cells and T-Cell Receptors

Disturbances of the immune system



(a) Schematic description of the different immune disturbances (Khailaie *et al.*, 2013).

Neoantigen Schematic



(b) T-Cells can identify the neoantigen signature (Chu *et al.*, 2018).

Figure 1.1: On the left side there is a "Snapshot" of the different immune disturbances. For comparison, the right side, depicts a schematic of T-Cells, T-Cell Receptors, tumour cells and shows where neoantigens are located in contrast to T-Cells and tumour cells.

1.2 Major Histocompatibility Complex (MHC)

The major histocompatibility complex was originally found as transplant antigens which primarily define tissue compatibility between individuals (A. *et al.*, 2009). The complex consists of three gene-classes. Genes from Classes I and II encode antigens, while genes from Class III mainly encode components of the complement system (Noris and Remuzzi, 2013). Classes II and I antigens are glycoproteins (Nakane *et al.*, 2019) which make T-Cells active in peptides.

T-Cells can be divided in two categories (Damoiseaux *et al.*, 1999); CD4 cells (Luckheeram *et al.*, 2012) which are known to bind to MHC class II genes and CD8 cells (Tsukumo *et al.*, 2018) which are related to MHC class I genes. Thus, from this point onwards, referring to T-Cells will essentially mean CD8 type cells.

To be more precise, they can be further explained in the following:

- Class I: as shown in both Table 1.1 and a graphical representation in Figure 1.2, class I genes express glycoproteins present on the surface of most nucleated cells. In addition, the presence of peptide antigens to T-cells is the main feature of these genes.
- Class II: express predominantly glycoproteins on APCs, where they introduce peptides to TH cells.
- Class III: encode numerous proteins with immunogenic functions made up of components of the complement and immune system.

Summary of Information of the MHC Class I Genes

Description	Explanation
Structure	MHC class I genes comprise of a membrane-spanning α chain(heavy chain) formed by MHC molecules, and α β chain created by the β2-microglobulin gene.
Types of APCs	MHC I have glycoproteins found in all nucleated cells of the body.
Semantic of Antigen	Class MHC I glycoproteins contain endogenous antigens derived from cytoplasm.
Peptide Size	MHC Class I contain 8-10 amino acid peptides
T-cells	Cytoxic T-cell lymphocytes
Co-receptor	Binding with CD8 co-receptors in T-cells
Enzymes	Cytosolic proteasome
Function	Targeting degradation of cells

Table 1.1: MHC class I gene detailed information.

Schematic Representation of MHC Class I

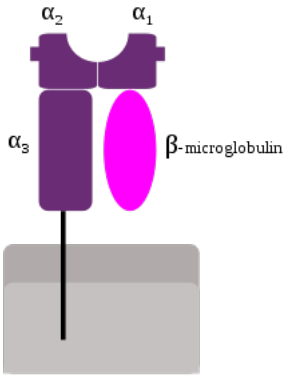


Figure 1.2: A schematic representation of MHC class I molecule, The molecule consists of three alpha-domains and one beta-two-microglobulin. The groove that binds peptides is located between the alpha-one and alpha-two domains (A. *et al.*, 2009).

Immunotherapy has become the core focus of cancer treatments. This type of treatment directs the immune system of a patient against cancer cells.

Cancer occurs from accumulating damage in DNA and gene mutations. Cancer cells contain mutations which do not occur in their healthy counterparts. Mutated gene products on major histocompatibility complex (MHC) molecules of tumour cells could be extracted and introduced as small neoantigens, and some can induce T-cell responses. These neoantigens, can then be considered as promising targets for developing tailored therapies.

Identifying them would ultimately enable a patient's immune system to locate and target cancer cells because of their distinctive nature while leaving healthy cells undisturbed. The first clinical trial to use neoantigens was conducted in 2015 (Rajasagi *et al.*, 2015). Since then, there has a been a steadily rising interest in neoantigen identification.

In addition to neoantigens, Tumor Associated Antigens (TAAs) are another target of immunotherapies (Zhang *et al.*, 2009). Unlike neoantigens, TAAs are unaltered self peptides that can be overexpressed on certain tumours (see Figure 1.1a). Although both TAAs and neoantigens are important targets of cancer therapies, studies favoured neoantigens for having more resources available in terms of bioinformatic approaches (Wagner *et al.*, 2018).

1.3 How Neoantigen Prediction Works

More and more evidence indicate that cancer immunotherapy is driven by immune responses of T-Cells to neoantigens (Blankenstein *et al.*, 2015), (Rosenberg *et al.*, 2015), (Wirth *et al.*, 2017). Even though there is a lot of research to be done, early studies suggest that immune response and identification of tumour cells which consist of particular peptides rely on the ability of the major histocompatibility complex class I genes to successfully bind with the protein and elicit an immune response.

Neoantigens could be developed through a wide range of methods, such as aberrant gene expression usually constrained to antigen-privileged tissues, viral aetiology, or tumour-specific enhancements of DNA which promote the formation of novel proteins.

The introduction of fairly inexpensive short-read sequencing, thorough neoantigen assessment focused on RNA-seq in comparison with other sequencing technologies (Yin *et al.*, 2019)(i.e. whole exome sequencing (WES)) has become viable. The main reason been that RNA-seq goes through RNA transcribed from DNA whereas WES looks at DNA in exonic regions. In addition, several cancer immunotherapy approaches seek to use such in-depth understanding of the neoantigen spectrum to develop and/or enhance T-Cell reactivity for immunotherapies (Liu *et al.*, 2017b).

Nevertheless, the evaluation and verification of the most suitable candidates for the neoantigens are in practice a very difficult and above all time-consuming challenge (Chen *et al.*, 2019). Typically, an approach is based on the individual patient-specific mutational burden: RNA-seq data were extensively analysed and used to predict neoantigens and their binding affinities to MHC I complex genes (A. *et al.*, 2009). This research seeks to facilitate this methodology from the perspective of precision medicine.

A Simplistic Immunity Cycle Diagram for Cancer.

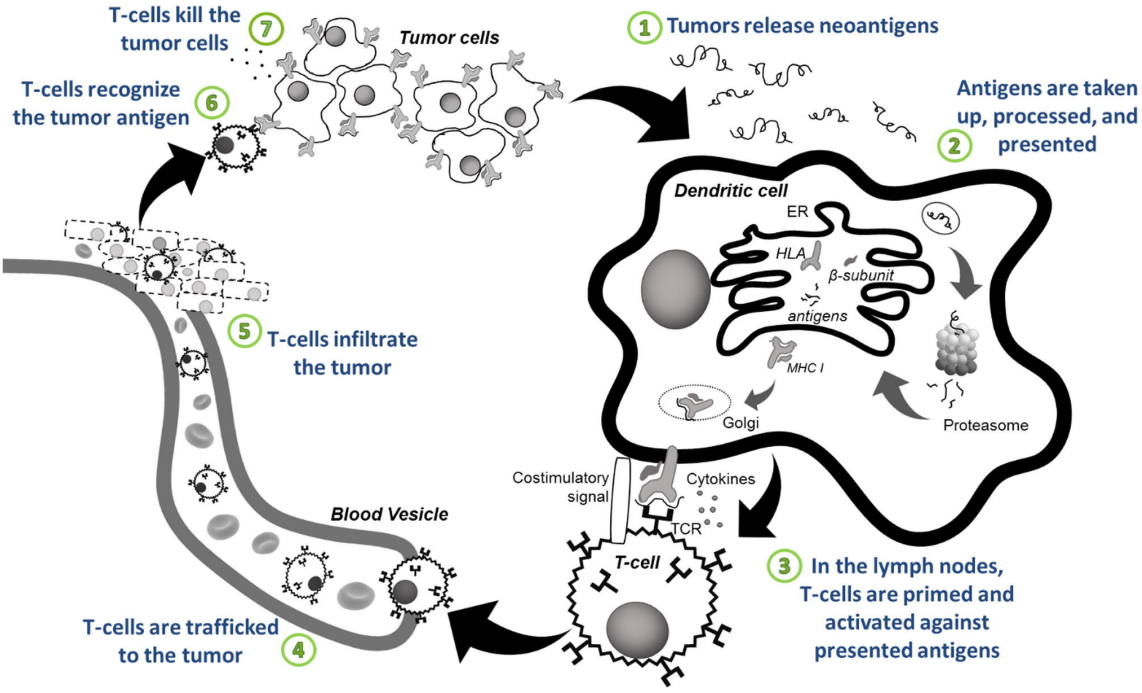


Figure 1.3: A simplistic Immunity Cycle Diagram for Cancer. The cancer produces antigens (1), which dendritic cells (DCs) (Patente *et al.*, 2019) bring in and process (2). T-cells are primed in the lymph nodes, and triggered against the epitopes displayed (3). T-cells (4) infiltrated (5) the tumour, these same antigens must be recognised on the cancer cells (6) to stimulate an immune response against the tumour (7) (Lee *et al.*, 2018)

Although neoantigen prediction is tailored to a particular patient tumour profile, many such steps need to be taken before they are identified as depicted in Figure 1.3. Through some in-silico approaches, researches are capable of using algorithms for searching and prioritising which neoantigens are causing the immune system to act most efficiently depending on the tumour profile of an individual. This will ultimately allow the extraction of unique mutations in tumour cells.

Chapter 2

Literature Review & Project Aims

2.1 Literature Review

To begin with, a critical literature review was performed with the Ovid Medline database (Ovid, 1946). Its purpose was to gain more information regarding the current state of knowledge in this field and to provide an overview of the key findings. A comprehensive mind-map of the search strategy can be found in Figure 6.1.

Cancer is by itself a challenging field as the rapid development of abnormal cells can also bypass several defence mechanisms which the immune system does not detect. In addition, antigens are often expressed on both normal cells and cancer cells, making the body susceptible to attacks at the wrong places. Not to mention the effects of other non-immunotherapy treatments such as radiotherapy and chemotherapy. Therefore, it is here that neoantigens shine.

Neoantigens are often defined as very valuable for cancer treatment, however, now there are more resources to focus on and explore this field. One reason why neoantigens were long regarded as important candidates is that they are exclusive to cancer cells. By comparison, antigens investigated for cancer immunotherapy were also expressed in normal cells, leaving healthy tissue susceptible to immune response.

Over the past five years, three independent studies ((Magrini *et al.*, 2015a), (Hu *et al.*, 2017), (Derhovanessian *et al.*, 2017)) reported on the experience of about two dozen people with prostate cancer who received experimental treatments containing neoantigen 'cocktails'. Once all the participants received the vaccinations, they made T-Cells which could recognise their cancers. Quite notably, every individual ended up receiving a unique collection of neoantigens to produce an appropriate combination of antigens. The study investigated specific tumour mutations and the immune system of every patient which makes it truly the first patient-specific vaccine (Hollingsworth and Jansen, 2019).

Malignant cell transformation is based upon DNA damage accumulation. Throughout what re-

searchers call neoantigen "era" (Wirth *et al.*, 2017), there have been discoveries that T-Cells also respond to neoantigens which occur because of damage in DNA. In addition, neoantigens recognition remains a critical driver of research for both the T-Cell checkpoint inhibitors and the T-Cell blockade therapy adopted as immunotherapy for cancer.

Thus, this project's main focus was on a variety of tools to improve neoantigen candidate prediction accuracy. Such tools are critically compared in order to define the best methods for conducting this research.

2.1.1 Reviewing HLA tools

Model Validation of Structures

Method	A	B	C	DQB1	DRB1
OptiType(%)	99.6	99.4	100.0	-	-
seq2HLA(%)	98.6	94.8	95.1	96.0	98.5
PHLAT(%)	99.4	93.4	94.3	96.0	98.5
HLAProfiler(%)	99.9	99.0	99.6	99.9	99.6
arcasHLA(%)	100.0	100.0	100.0	99.9	99.7

Table 2.1: First row contains all the HLA prediction methods. First column contains different HLA alleles. Columns 2-6 examine the accuracy of each tool on different parameters. Accuracy scores were taken from ArcasHLA publication paper (Orenbuch *et al.*, 2019).

Table 2.1 contains information on ArcasHLA for structure validation predicted via concordance of calls from 447 RNA samples and 69 individuals (Orenbuch *et al.*, 2019). This tool achieved an accuracy score of 100 per cent for Class I and over 99.7 per cent for Class II when running on the 1000 Genomes project (Auton *et al.*, 2015), outperforming other existing tools (Table 2.1). All in all, using this set of benchmarks, ArcasHLA delivers high accuracy rate for the HLA field (Orenbuch *et al.*, 2019).

2.1.2 TCR sequencing tools

Accuracy of Gene Segment Identification

Platform	Wrong V genes(%)	Wrong D genes(%)	Wrong J genes(%)
MiXCR	0.0	35.3	0.2
IMGT	0.6	21.6	9.3
Decombinator	3.8	-	2.3
IgBlast	0.0	28.5	0.0

Table 2.2: Columns two to four have information regarding variable (V), joining (J), and diversity (D) genes contained in TCR molecules. With that said, A comparative analysis was performed(Bolotin *et al.*, 2015) on V(D)J gene segment accuracy and alignment with known V, D, J genes for selected platforms on synthetic human T Cell Receptor Beta Locus(TRB) sequences. Each data-set contains 100,000 reads. Accuracy scores were taken from the MixCR publication paper (Bolotin *et al.*, 2015).

This Section is on TCR sequencing tools. Table 2.2 depicts four different tools that are capable of performing this type of sequencing. MixCR was the tool selected as it has shown the best accuracy among other tools. The performance and precision of the extraction is considerably more efficient than that of other tools in the field. In addition, the processing speed was significantly better with two and four orders of magnitude more efficiency (Bolotin *et al.*, 2015).

2.1.3 Neoantigen Prediction Tool Comparison

As there was an abundance of tools and approaches, it was necessary to undertake an in-depth research of the most common tools (Richters *et al.*, 2019) of bioinformatics approaches for RNA-sequencing. Thus, we have collated Tables 2.3 and 2.4 after extensive research and analysis, firstly to gain a better understanding of the output of each tool. Secondly, to compare each tool and define which one is best for this study.

Output Explained

Parameter	Description
Sample	vcf filename/patient identifier
Chr	Chromosome of mutation
Allelepos	Position of the mutation
Pos	Residue number (starting from 0)
Ref	Reference base at the position
Alt	Alternative base at the location
RefSeqID	Gene name and RefSeq ID separated by a colon. Multiple genes/RefSeq IDs separated by a comma.
HLA Allele	Allele name
Peptide	Amino acid sequence
Affinity	Predicted binding affinity in nanoMolar units.
Score	Raw score
Rank	Ranking of the predicted affinity
Bind Level	SB: strong binder, WB: weak binder

Table 2.3: The primary output file of neoantigens contains these columns. The first column is essentially a list of all the parameters found in the output of the neoPredPipe prediction tool. The second column gives a brief explanation of what each parameter means.

Output of Each Tool

Parameters	Neopredpipe	CloudNeo	Mupexi	pVACtools	TSNaD
Sample	✓	X	✓	✓	✓
Chr	✓	X	✓	✓	✓
AllelePos	✓	X	✓	✓	X
Pos	✓	X	✓	✓	✓
Ref	✓	X	X	X	X
Alt	✓	X	X	X	X
RefSeqID	✓	X	X	X	✓
HLA allele	✓	X	X	✓	X
Peptide	✓	X	✓	✓	✓
Binding Affinity	✓	X	✓	✓	X
Score	✓	X	✓	X	X
Rank	✓	X	✓	✓	X
BindLevel	✓	X	X	X	X

Table 2.4: Analysis of neoantigen and neoepitope prediction tools. The first row contains all the potential tools and the first column has information generated data from each tool as explained in Table 2.3. All the data were collated from each tool's, publication papers, readMe files and test data.

Table 2.4 conveys in detail the generated information of each tool. NeoPredPipe (Neoantigen Prediction Pipeline) is used for evaluating putative neoantigens for single and multi-regional tumour samples and their corresponding recognition ability. NeoPredPipe can provide researchers and clinicians with detailed information regarding predicted neoantigenic burdens in an efficient way, while at the same time providing substantial insights into tumour heterogeneity mainly due to somatic mutation.

In this particular scenario, neoPredpipe has outperformed other tools. That is mainly because of the machine learning algorithms that can perform a scoring for each predicted neoantigen. Furthermore, it can provide information on the binding level making it again an ideal tool for this project.

2.2 Project Aims

The aim of this project is to use bulk RNA-seq to explore potential antigens. As illustrated in Figure 2.1, this is made possible through a selection of tools.

Even though there is some variability, these are the basic steps in the prediction of neoantigens:

Figure 2.1: An overview of the neoantigen prediction approach. Yellow boxes made up the milestones of the project whereas light green boxes contain subsections of key milestones. Blue boxes contain the tools required during each phase of the project. These have been extensively discussed in Section 3.

Lately, scientists have been able to leverage RNA-seq tools for classifying neoantigens and delivering patient-specific vaccines for treating tumours. Neoantigens are predicted from matched tumour–normal sequencing data to produce a unique cancer vaccine. Accordingly, they need to be classified to their predicted ability to stimulate a response on T-Cells. A neoantigen prediction pipeline includes several steps such as the detection of somatic mutations, HLA typing for identifying patient-specific T-Cells, peptide processing, and binding prediction using machine learning (ML) of peptide-MHC I genes.

From Figure 2.1 methods and milestones can be distinguished to two main colours. Light blue stands for all the tools and methods used and yellow is used to convey each milestone. Key milestones can be explained below:

- To begin with, a set of five tools have been used for identifying genomic variants, applying quality control processing, filtering and variant calling. Initial format of the data was in FASTA files. After processing, the end result was a variant calling file (VCF).
- The next set of tools was used for predicting neoantigens (ArcasHla (Orenbuch *et al.*, 2019), neoPredPipe (Schenck *et al.*, 2019)). ArcasHLA was used for identifying patient specific variations (Hutchinson *et al.*, 2015). NeoPredPipe was used for predicting neoantigens.
- Machine learning (ML) methods were then used to improve predictions. Linear regression was used for proving further insights to relations between variables and KNeighoubrs was used for cross validating results.
- Next milestone contains tools for aligning and sequencing immune data (MixCR (Bolotin *et al.*, 2015), NetChop (Saxová *et al.*, 2003), SnpEff (Cingolani *et al.*, 2012)).
- In addition to ML analysis, more analysis was undertaken combining regression models and immune data.
- Finally, differentially expressed genes were identified to further aid the whole process (FeatureCounts (Liao *et al.*, 2013), DESeq2 (I. *et al.*, 2014)). FeatureCounts was used for counting reads and DESeq2 was then used for exploring identifying differentially expressed genes.

Chapter 3

Methods

3.1 Data-Sets

Tissue samples of prostatectomy were collected from the Baylor Prostate Specialised Research Excellence Tissue Core (SPORE) (O. *et al.*, 1992) for this study. A screenshot of the website where the samples have been taken from is shown in Figure 3.1.

Complete RNA samples were handled utilising Illumina mRNA-seq protocol for transcriptome sequencing(Kannan *et al.*, 2011). The transcriptome of twenty tumour cells and ten matched benign cells was analysed from individuals with prostate adenocarcinoma who did not undergo pre-operative treatment before prostatectomy to recognise possible neoantigens which can illicit a TCR response for prostate cancer. Additionally, there was not any demographic information provided.

Data-Sets Snapshot

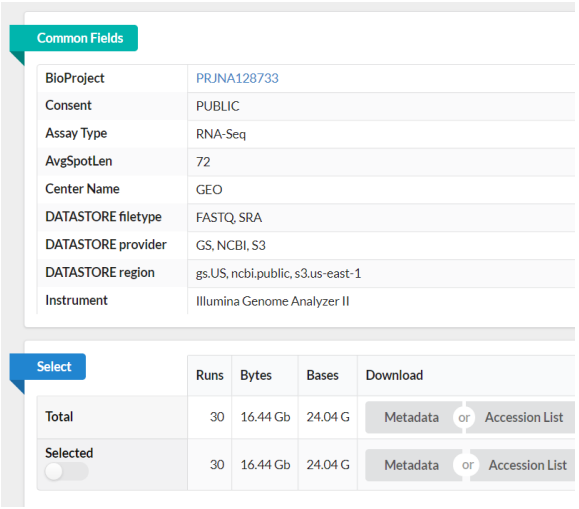


Figure 3.1: An overview of the data-sets. Data has been collated from the Baylor Prostate Specialised Research Excellence (SPORE) (O. *et al.*, 1992). Link directly to the data-sets can be found here.

3.2 Identification of Genomic Variants

3.2.1 Bowtie

Bowtie 2 is an ultra-fast and lightweight tool for aligning sequencing reads to long reference sequences. In this particular project Bowtie version 2.1.0 was used. In the following Listing (3.1) "bowtie2-build" parameter builds an index from a set of DNA sequences and returns a set of six files while at the same time selecting automatically the best settings for an efficient run.

```
1 ## loading modules
2 module load bowtie/2.1.0
3 module load tophat/2.0.10
4
5 ## building genome index mapping command
6 bowtie2-build ${myDir}/../resources/${genomeName} ${myDir}/../resources/${
      genomeIndex}
```

Code Listing 3.1: Bowtie2 was used for aligning sequencing reads and outputting a set of six files.

3.2.2 Tophat2

RNAseq data files for prostate cancer were obtained as FASTQ files. Quality-controll was conducted using FastQC. Reads have been mapped to reference genome GRCh37.p87 using tophat version 2.1.1 (Daehwan *et al.*, 2013). The parameter "–transcriptome-index" option here is used to point to a directory where the transcriptome data files will be stored. For efficiency purposes, the parameter "–num-threads 20" was used to select the number of twenty threads for processing.

```
1 ## run tophat mapping command
2 tophat2 --num-threads 20 --transcriptome-index ${myDir}/../resources/${
      gtfIndex} \
3 -o ${myDir}/../output/${sampleID}/tophat \
4 ${myDir}/../resources/${genomeIndex} \
5 ${myDir}/../input/${sampleID}_1.fastq.gz \
6 ${myDir}/../input/${sampleID}_2.fastq.gz
```

Code Listing 3.2: Running tophat mapping command.

3.2.3 Samtools

Using SAMtools version 1.4 (Li *et al.*, 2009), SAM files have been converted to BAM files, and the coordinates were sorted. BAM files have been processed according to best practices for the bioinformatic characterisation of neoantigens (Richters *et al.*, 2019).

3.2.4 Opposum

Opposum version 2.0 was used for pre-processing and filtering RNA-seq data prior to variant calling. The following arguments were used ProperlyPaired=True and MapCutoff=40 (Oikkonen and Lise, 2017).

The following commands were used to efficiently revised the output and prepare it for the next set of tools.

```
1 grep -v "^##" ${SampleID}.vcf | cut -f1,2,4,5,10 | sed 's/\t/,/g'| \
2 awk '{gsub(/:([^\n]+)$/,"")}{print}' \
3 | awk '{gsub(/Sample1/,"GEN")}{print}' \
4 > allNeoantigens.csv
```

Code Listing 3.3: Efficiently editing the whole file to obtain only the most valuable data.

Line 10 indicates how all the data was extracted.

```
1 cut -f1,2,4,5,10
```

Code Listing 3.4: Extracting the most important columns.

The following command removes everything after the colon sign (:) and until the end of the line.

```
1 awk '{gsub(/:([^\n]+)$/,"")}{print}'
```

Code Listing 3.5: Remove everything after the colon sign (:).

Additionally, gsub was used to add a new column name (GENES) to reflect new data.

```
1 awk '{gsub(/Sample1/,"GENES")}{print}'
```

Code Listing 3.6: Using gsub to add a new column name.

3.2.5 HLA Typing

Identification of MHC class I genes was completed with HLA typing and in particular, arcasHLA version 1.1.3 (Orenbuch *et al.*, 2019) was used. ArcasHLA is aligning reads from the RNA-sequencing data from each individual patient and then maps these alleles to a list of already known HLA alleles. At this point, all abundance is computed individually for each gene. The genotype that enhances the amount of associated reads is extracted by finding the most prevalent regions.

```
## running arcasHLA
./arcasHLA-master/arcasHLA extract ../output/${sampleID}/tophat/${sampleID}.
    bam \ # input
--paired -o ../output/${sampleID}/arcasHLA/${sampleID} # output

./arcasHLA-master/arcasHLA genotype ../output/${sampleID}/arcasHLA/${
    sampleID}.extracted.1.fq.gz \ # input-1
../output/${sampleID}/arcasHLA/${sampleID}.extracted.2.fq.gz \ # input-2
-o ../output/${sampleID}/arcasHLA/ \ # output
-t 20 -v # specify threads and verbose parameter

## merge all files and create a tab separated output
./arcasHLA-master/arcasHLA merge
--i ../output/${sampleID}/arcasHLA/ \ # input
--o ../output/${sampleID}/arcasHLA/ -v # output
```

Code Listing 3.7: HLA typing for MHC class I genes using arcasHLA

3.2.6 NeoPredPipe

Predicting potential neoantigens which would eventually bind to MHC I complex genes was completed with the NetMHCpan version 4.0 (Hoof *et al.*, 2009) and the NeoPrepPipe version 1.0 (Schenck *et al.*, 2019). Only neoantigens of nine amino acids were considered. NeoPredPipe was used with twenty tumour samples and ten benign samples. Figure 3.1 represents the workflow of predicting neoantigens. The first stage consists of the identification of non-synonymous variants from a VCF file. To this end, ANNOVAR (Li *et al.*, 2010) was employed for processing samples and for the prioritisation of variants. Then, NetMHCpan was used as mentioned above. From this point onwards, neoPredPipe utilises various algorithms (Schenck *et al.*, 2019) for delivering candidate information and methods for predicting scores and identifying neoantigens.

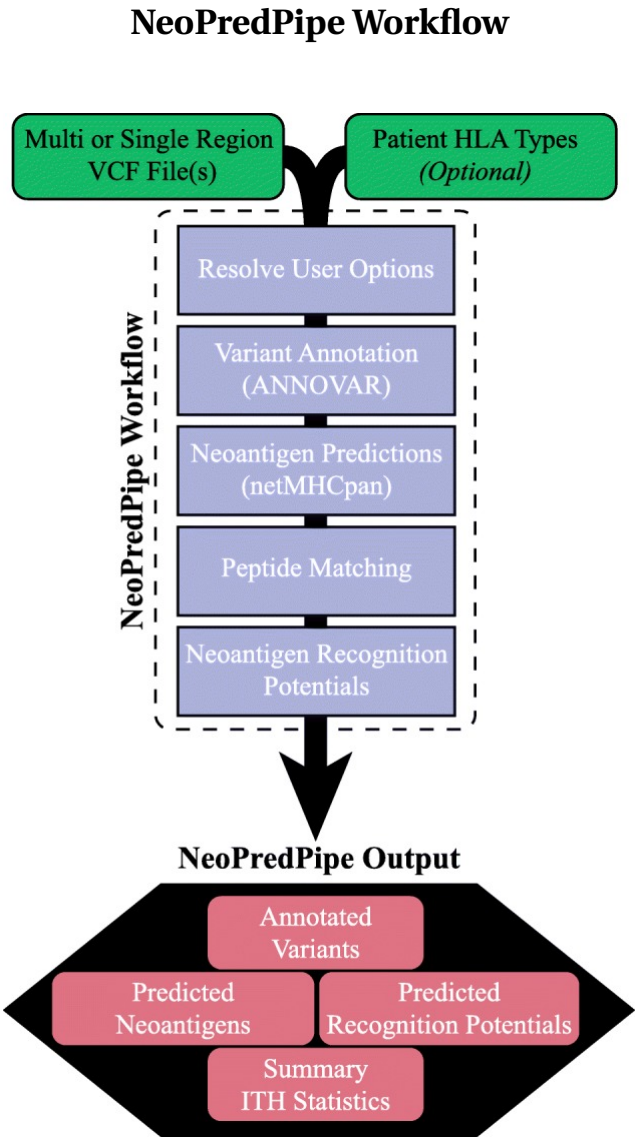


Figure 3.2: NeoPredPipe workflow distinguishing between client (green) phase, implementation processes (purple) and summary statistics (red) for the output.

3.3 Neoantigen Prediction Using ML Methods

This Section contains the ML methods and tools that have been used for this study. The main aim was to improve the accuracy of the model. In order to achieve this, some fundamental objectives have been set. First of all, select the most appropriate regression model and then achieve the best possible prediction accuracy using various cross-validation models. If time permits, introduce more parameters and run the whole procedure again with the aim to improve the output.

3.3.1 Scikit-Learn

Scikit-learn can perhaps be called one of Python's most widely use library. It includes several effective machine learning and statistical modelling resources such as classification, regression, clustering, and reduction in dimensionality(Pedregosa *et al.*, 2011).

Scikit-learn comes equipped with a number of components. Here is a list of them:

- **Supervised learning algorithms:** Generalised Linear Models (e.g. Linear Regression), Support Vector Machines (SVM), decision trees, and a lot more can be found in the scikit-learn toolbox.
- **Cross-validation:** There are many different cross-validation methods to check the accuracy of the generated models. The cross-validation model used in this study 'k-fold'. Coding examples and explanations of the code can be found in Section 6.3.4.
- **Unsupervised learning algorithms:** This can include principal component analysis clustering and unsupervised neural networks.

3.4 Combining VCF and TCR based predictions

3.4.1 MixCR

This is another tool for next-generation sequencing analysis. Listing 3.8 contains the MixCR version 2.1.3 (Bolotin *et al.*, 2015). In particular, MixCR was used for the alignment of the immune sequencing data. The entire code for the MixCR pipeline can be split into the following three Listings.

The first Listing (3.8) is just for aligning the raw FASTA files. The output of this step is a non human readable format. However, is the format required for the next step of the analysis.

```
1 ## run align on the raw fastq
2 ./mixcr align  \
3 ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
      fasta # input
4 ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
      vdjca # output
```

Code Listing 3.8: MixCR on the the FASTQ data for reading alignments

Then, Listing 3.9 is the step for assembling clones. The command MixCR assemble will create a report file from the output of the alignment file of the previous step. This step is computationally expensive but it is necessary to create a binary file.

```
1 ## run assemble on the vdjca
2 ./mixcr assemble \
3 ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
      vdjca \
4  ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
      clns
```

Code Listing 3.9: MixCR on the the FASTQ data for assembling clones

The final step is for producing a file in a txt format that contains all the clones from the alignment files. An an optional step, the keyword 'pretty' can be added as show in Listing 3.10 line 7 to obtain a better looking report which is also comma separated. Finally the complete code for MixCR can be found under appendix subsection 6.3.2.

```
1 ## run export on the clones
2 ./mixcr exportClones \
3 ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
      clns \
4 ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}_
      aligned.txt
5
6 ## Optionally run export on the clns with a csv file format output
7 ./mixcr exportClonesPretty \
8 ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
      clns \
9 ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}_
      aligned.csv
```

Code Listing 3.10: MixCR export command to export all the obtained clones

3.4.2 NetChop

The proteasome has a vital role in vertebrate immune responses. It is generating the bulk of peptides which can be introduced on T cells through destroying inter-cellular peptides from tumour samples and their healthy counterparts.

The next step was on acquiring an extra insight into the specificity of the proteasome partly because of the involvement of the proteasome in the production of neoantigens and in addition, because of the necessity of it to predict which neoantigens will be produced from both self and non-self peptides.

In Listing 3.11 netChop version 3.1 (Saxová *et al.*, 2003) was used for predicting the proteasomal cleavage sites and was combined with the output of NeoPredPipe to produce candidate neoantigens. The use of netChop fortunately is straight forward. The second line of the Listing calls the tool with the next line consisting of the input. Line four contains the code that specifies the output directory.

```
## predicting proteasomal cleavage sites
netchop-3.1/netchop \
${myDir}/../output/${sampleID}/getfasta/${sampleID}.fasta \
> ${myDir}/../output/${sampleID}/netchop/${sampleID}.out
```

Code Listing 3.11: Prediction of proteasomal cleavage sites.

3.4.3 SnpEff

SnpEff, is used for the fast categorisation of variant effects in genome sequences. SnpEff mainly annotates variants depending on their genomic positions and once a genome is sequenced, it can predict coding effects.

Similarly with netChop, SnpEff (Cingolani *et al.*, 2012) is another easy to use tool. The code can be broken down into four sections. The first part is calling the tool while the second part is about the genome assembly of the FASTA file. Lines four and five make up the input and the output directories.

```
## categorising variants
java -Xmx4g -jar snpEff/snpEff.jar \
GRCh37.75 # specify genome assembly \
${myDir}/../output/${sampleID}/platypus/${sampleID}.vcf \ # input
> ${myDir}/../output/${sampleID}/snpEff/${sampleID}.vcf #output
```

Code Listing 3.12: Categorisation of variant effects.

3.5 Refining neoantigen prediction

The diversity of T-cell receptors analysis was carried out using previously published MiXCR derived TCR reads — MiXCR is a TCR analytical tool for data inference from RNA-seq. A number of computational methods for identifying neoantigens were formed based on the identification and quantification of MHC class I genes. Computational methods relied primarily on ML techniques, such as linear regression(LR) trained on large HLA-binding peptide data-sets.

3.5.1 Iterating Through Regression Models

The following code was implemented for selecting the best regression model. The first part of the code selects the best possible regression model by the r square output. The second part is for selecting the best cross-validation algorithm by the r square result. Then there is a recursive algorithm for finding the best possible value given the training the model has been through.

```
# Main script calling the base functions of the pipeline
def main():
    # Define the desired r square value
    target_value = 0.38

    # Get data and create plot
    df = plot_()

    # Regression and cross-validation testing
    (r_square, y, predicted, df2, c) = convert_to_features_format(df)

    # Recursive loop until desire target has been achieved
    while r_square < target_value:
        print(r_square)
        main()
```

Code Listing 3.13: This part of the code describes all workings in Python to loop through the code until the desire outcome has been achieved

3.6 Differentially Expressed Genes

3.6.1 Feature Count

This next tool is used for quantifying reads that have been generated from RNA-seq in terms of genomic features. Essentially, the featureCounts tool maps reads to a single location and counts them. On top of that, it is a very well equipped tool that incorporates many strategies such as chromosome hashing and feature blocking to best assign reads while maintaining high efficiency. Thus, it was selected as it is a very accurate and fast tool while at the same time being relatively easy to understand and use (Liao *et al.*, 2013).

Listing 3.14, demonstrates the required code for implementing featureCounts. There are options such as:

- -T n to specify the number of cores to be used
- -s n in case of "reversely" stranded data
- -a specify GTF path
- -o name of the output directory

```
1 ## run featureCounts
2
3 featureCounts \ # run the tool
4 -T 4 # specify 4 cores \
5 \-s 2 # these data are "reverse"ly stranded
6 -a ${myDir}/../resources/${gtfIndex}.gtf # GTF directory \
7 -o ${myDir}/../output/${sampleID}/featurecounts/${sampleID}.markdup.
      featurecount # output directory \
8 ${myDir}/../output/${sampleID}/featurecounts/${sampleID}.markdup.sorted.bam #
      input directory
```

Code Listing 3.14: This part of the code describes all workings for quantifying reads that have been generated.

3.6.2 DESeq2

DESeq2 is mainly used to perform an internal normalisation across all gene samples (I. *et al.*, 2014). This is achieved through an end to end gene-level RNA sequencing workflow using Bio-conductor packages as described in Listing under appendix in Section 6.3.4. This includes the alignment of FASTQ files to the reference genome and the preparation of a count matrix which tallies the amount of RNA sequencing fragments for each gene within each individual. Consequently, exploratory analysis is performed for quality assessment and to further explore any relations between samples.

In this particular scenario, DESeq2 was used for normalising the reads across five tumour samples and five healthy samples. This generated p values for each sample and log2 Fold change values as described in results..

The following code includes the most important parts that are required to generate this output. The full code can be found in the appendix under Listing 6.4.

```
# Create a DeSEQ2 design matrix

exptDesign = data.frame(
  row.names = colnames(rawData),
  condition = targets$treatment
)

# Create a DeSEQ2 experimental object

exptObject <- DESeqDataSetFromMatrix(
  countData = rawData,
  colData = exptDesign,
  design = ~ condition
)

# Run the analysis

analysisObject = DESeq(exptObject)
‘‘‘
```

Code Listing 3.15: This part of the code describes all workings in R required for implementing differential analysis.

Chapter 4

Results

Previous studies ((Magrini *et al.*, 2015a), (Hu *et al.*, 2017), (Derhovanessian *et al.*, 2017)) have described the role played by neoantigens in cancer cells and have defined a variety of tools (Richters *et al.*, 2019) for predicting neoantigens. This study has sought to gain a better understanding of what defines high-scoring (ranked by predicted binding affinity) neoantigens.

To begin with, neoPredPipe (Schenck *et al.*, 2019) was used for tumour analysis in order to appreciate the potential of neoantigens. This resulted in Table (4.1). Predictions were limited to individuals with only tumour tissues. There is also information on the count of neoantigens for each one of them with a classification of how many neoantigens are weak and strong binders.

The last column of the Table (see 4.1) represents the average ranking score for each patient. This was included to demonstrate that the predicted number of neoantigens varies a lot. There is a count of 123 for the least predicted neoantigens and as many as 3150 for the most predicted neoantigens. These may sound a lot but according to a recent study (Schumacher and Schreiber, 2015), the likelihood of neoantigen formation is roughly one mutation per megabase. In other words, these mutations are classified as regularly occurring (see Figure 6.2).

High-scoring neoantigens were then defined in order to filter out the low-quality ones. This resulted in a much smaller data-set which indeed makes it easier to further explore the potential of neoantigens.

Although not observable in Table 4.1 there is a pattern among the patients even though the ranking score was approximately the same among individuals. A better way to understand this pattern is Figure 4.1.

NeoPrePipe Summary Statistics

Patient	Neonatigens	WB	SB	Scoring(%)
SRR057629	786	518	268	0.911
SRR057630	786	546	240	0.944
SRR057631	1700	1188	512	0.932
SRR057632	518	342	176	0.896
SRR057633	1844	1266	578	0.919
SRR057634	480	318	162	0.890
SRR057635	450	276	174	0.877
SRR057636	1404	960	444	0.926
SRR057637	2156	1472	686	0.925
SRR057638	2426	1710	716	0.938
SRR057639	2880	2064	816	0.955
SRR057640	1838	1278	560	0.909
SRR057641	2918	2060	858	0.946
SRR057642	3110	2154	956	0.928
SRR057643	1582	1072	510	0.901
SRR057644	3150	2222	928	0.926
SRR057645	1202	856	346	0.947
SRR057646	226	158	68	0.926
SRR057648	1962	1322	640	0.909

Table 4.1: Values of each patient for predicted neoantigens, the sum of WB(weak binders) and the sum of SB(Strong Binders). Fifth column contains the average scoring for each patient.

In addition to Table 4.1 the summary of the neoPredPipe tool is observable here as a visualisation of the key results. Firstly, Figure 4.1 shows how many strong and weak binders were predicted for each sample.

It is worth mentioning that one of the samples with tumour tissue(Sample SRR057647) did not show any predicted neoantigens and for that reason, it has been removed from the output. Further analysis on this from the initial publication (Kannan *et al.*, 2011) did not result in any meaningful insights.

Having said that, there seems to be a cluster of samples with fewer predicted neoantigens and another cluster with patients having too many predicted neoantigens. This among other things can be attributed to the degree of genetic instability on the tumour tissue (Peng *et al.*, 2019).

Count of SB And WB Per Each Sample

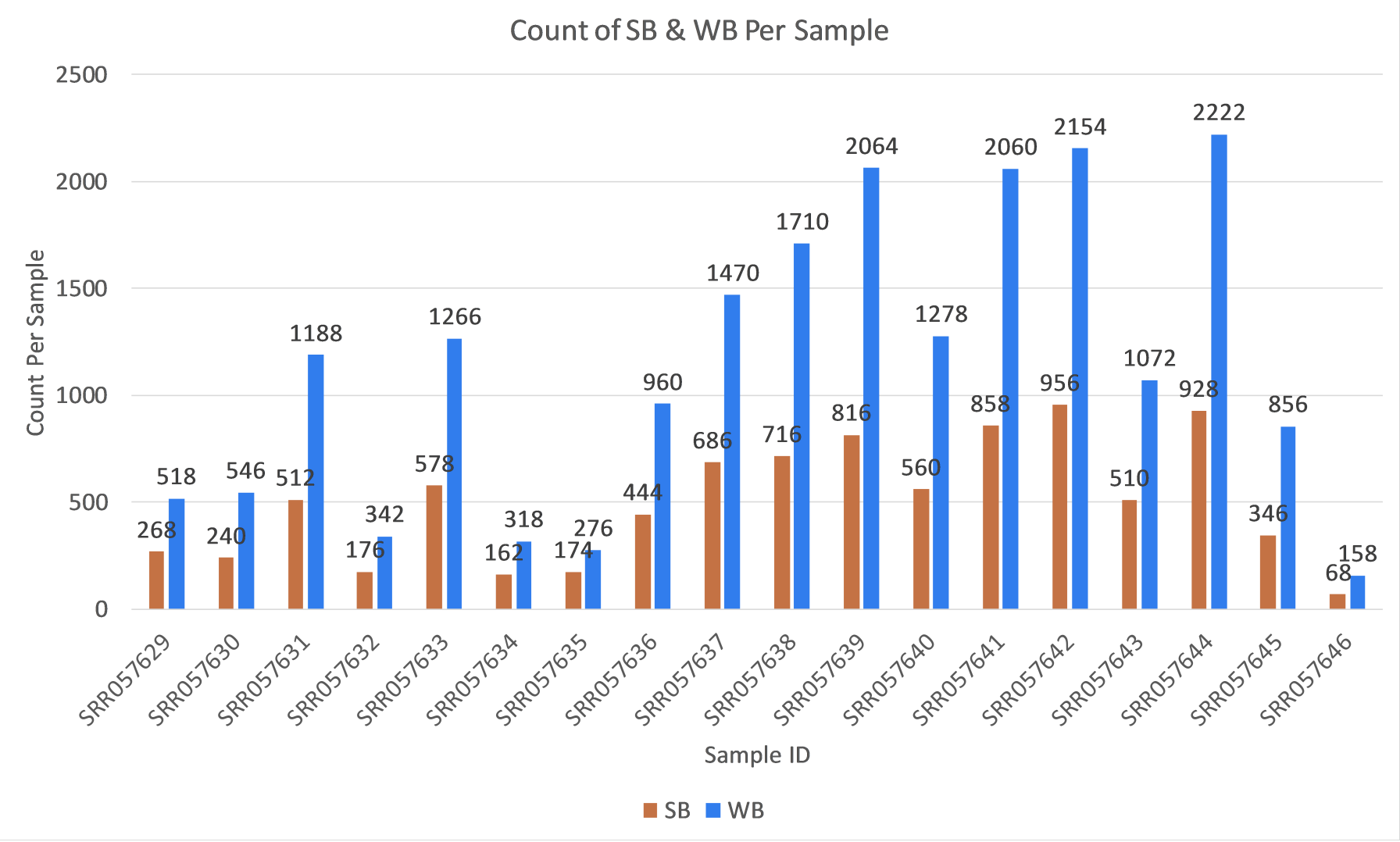


Figure 4.1: Count per each individual. The x-axis contains each individual and the y-axis contains how many counts each sample has. The blue bars stand for strong binders(SB) and the orange ones stand for weak binders(WB).

4.1 High-Scoring Neoantigens

A Figure (4.2) of the highest-scoring neoantigens has been collated in this Section. Due to the lack of significant results from the summary statistics of the neoPredPipe tool, it was necessary to explore the dataset from a different point of view. One way was to have a count of the most occurring genes among the samples. As shown in Figure 4.2, the most prevalent gene is PLIN4 (Richardson *et al.*, 2011).

PLIN4 is associated with lipogenesis (a metabolic process to synthesis fat (Kersten, 2001)). There has been a very limited number of studies that show this gene to be related to prostate cancer (Blanc *et al.*, 2019).

Interestingly though, different pharmaceutical companies have shown interest in patenting such programs or algorithms and their findings. Additional research has revealed that PLIN4 has already been patented by a pharmaceutical company in this patent (Fritsch *et al.*, 2016). More information can be found in this preliminary publication (Fritsch *et al.*, 2014). Thus, certain evidence suggest that there is indeed significant potential in researching neoantigens.

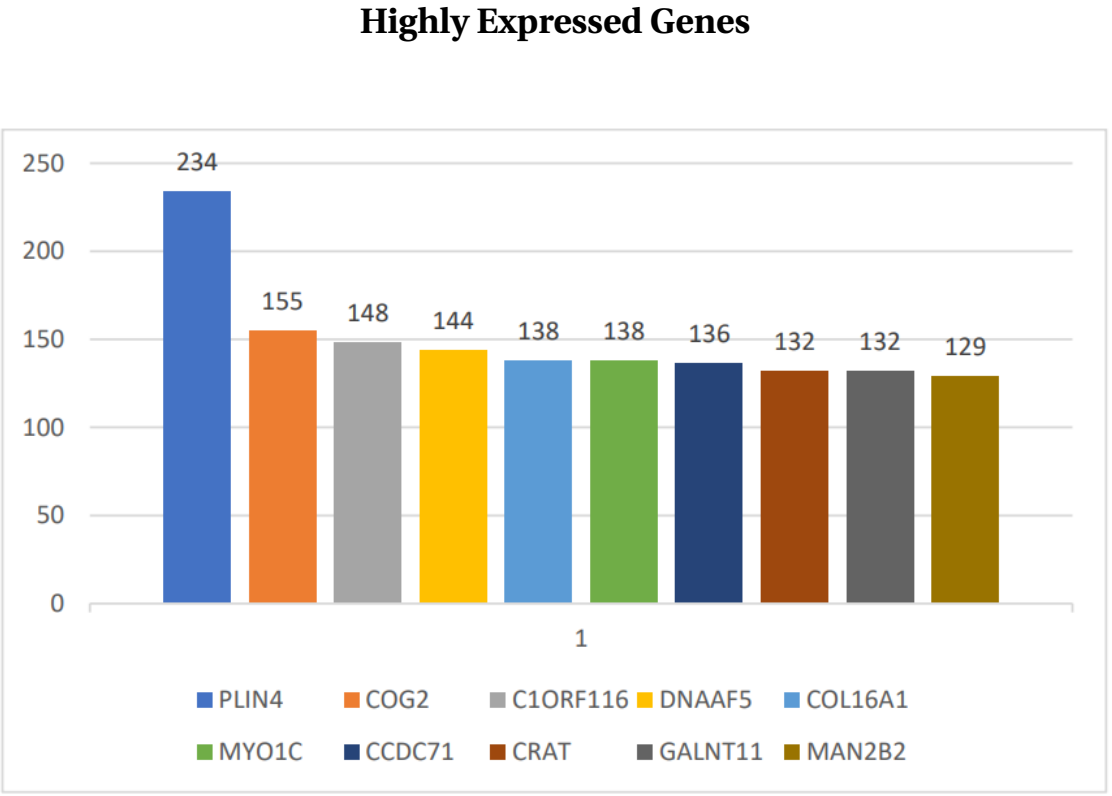


Figure 4.2: A plot of highly expressed genes with "PLIN4" being the most expressed one. This was the result of exploring summary statistics for Table 4.1 from a different point of view (i.e. by the most prelevant gene across the whole sample (n=20))

4.2 R-Square of Each Regression

Further analysis was conducted to actively evaluate the output of neoPredPipe with ML methods. This included a set of five regression models. The aim of these models was to improve the accuracy scores as much as possible. These scores can be depicted in Table 4.2.

All in all, the regression models were used with one dependent variable and one independent variable. This resulted in insufficient data and for that reason, more variables were introduced.

Different combinations of different parameters were tested in order to improve the R-Square scores. Regardless of the number of attempts made, none of them generated any substantial improvements. Apart from the KNeighbours regression, all the other regression methods had an average output of 0.651(%). The KNeighbours regression was tested with n=20. This produced an R-square of 0.597(%). This did not provide any significant results that would justify using an outlier detection method to remove any potential outliers.

R-Square Output

Description	R-Square (%)
Linear Regression	0.651
Ridge Regression	0.651
KNeighborsRegressor	0.597
DecisionTreeRegressor	0.651
VotingRegressor	0.652

Table 4.2: Values of the R-Square results next to the regression methods. First column represent the description of each result. Second column contains the R-square scores.

4.3 Differentially Expressed Genes

Due to the inconclusiveness of the previous Section (4.2), more tests were implemented. Conducting a differential analysis could help identify if a gene is differentially expressed. In addition, this type of analysis has never been combined before with neoantigen prediction. Thus, it was a great way to try a novel approach and to provide insights in the predicted neoantigens. On top of that, this could potentially assist in the process of defining high-scoring neoantigens.

Therefore, a differential analysis was conducted on the patients as described in Section 3.6.1. P-value threshold was set to 0.05% . After filtering and processing overlapping genes the resulted output was made out of seventeen genes as shown in Table 4.3.

Differential Analysis

Gene ID	Gene Name	Description	P-value Adj(%)	Log2 Fold Change
ENSG00000105369	CD79A	molecule	0.001	2.733
ENSG00000211895	IGHA1	Immunoglobulin	1.525E-07	1.423
ENSG00000211890	IGHA2	immunoglobulin	8.218E-06	2.156
ENSG00000211896	IGHG1	immunoglobulin	4.430E-13	1.564
ENSG00000211897	IGHG3	immunoglobulin	2.089E-12	2.152
ENSG00000211899	IGHM	immunoglobulin	0.001	0.480
ENSG00000211938	IGHV3-7	immunoglobulin	0.007	2.508
ENSG00000224373	IGHV4-59	immunoglobulin	0.004	3.928
ENSG00000243466	IGKV1-5	immunoglobulin	0.001	1.264
ENSG00000241351	IGKV3-11	immunoglobulin	0.002	2.042
ENSG00000239951	IGKV3-20	immunoglobulin	0.001	3.305
ENSG00000262902	IGLC3	immunoglobulin	1.096E-08	2.053
ENSG00000211679	MTCO1P40	MT-CO1 pseudogene 40	0.009	-0.211
ENSG00000228278	ORM2	orosomucoid	0.001	-7.168
ENSG00000095303	PTGS1	prostaglandin-endoperoxide	5.986E-10	-1.599
ENSG00000073756	PTGS2	prostaglandin-endoperoxide	0.032	0.234
ENSG00000124157	SEMG2	semenogelin	0.002	-2.464

Table 4.3: The first column contains the gene ID followed by the gene name and a sort description of the gene, mainly with what is associated. The next two columns contain the p-values and the log2 fold change values.

Having said that, only one gene was actually between the range of 0.05(%) and 0.01(%). The rest were equal or below the 0.01(%) threshold, making the rest sixteen genes highly significant. The log2 fold change was as high as 3.928 and contained lows of negative values as much as 7.168 (Figure 4.3) which is highly significant.

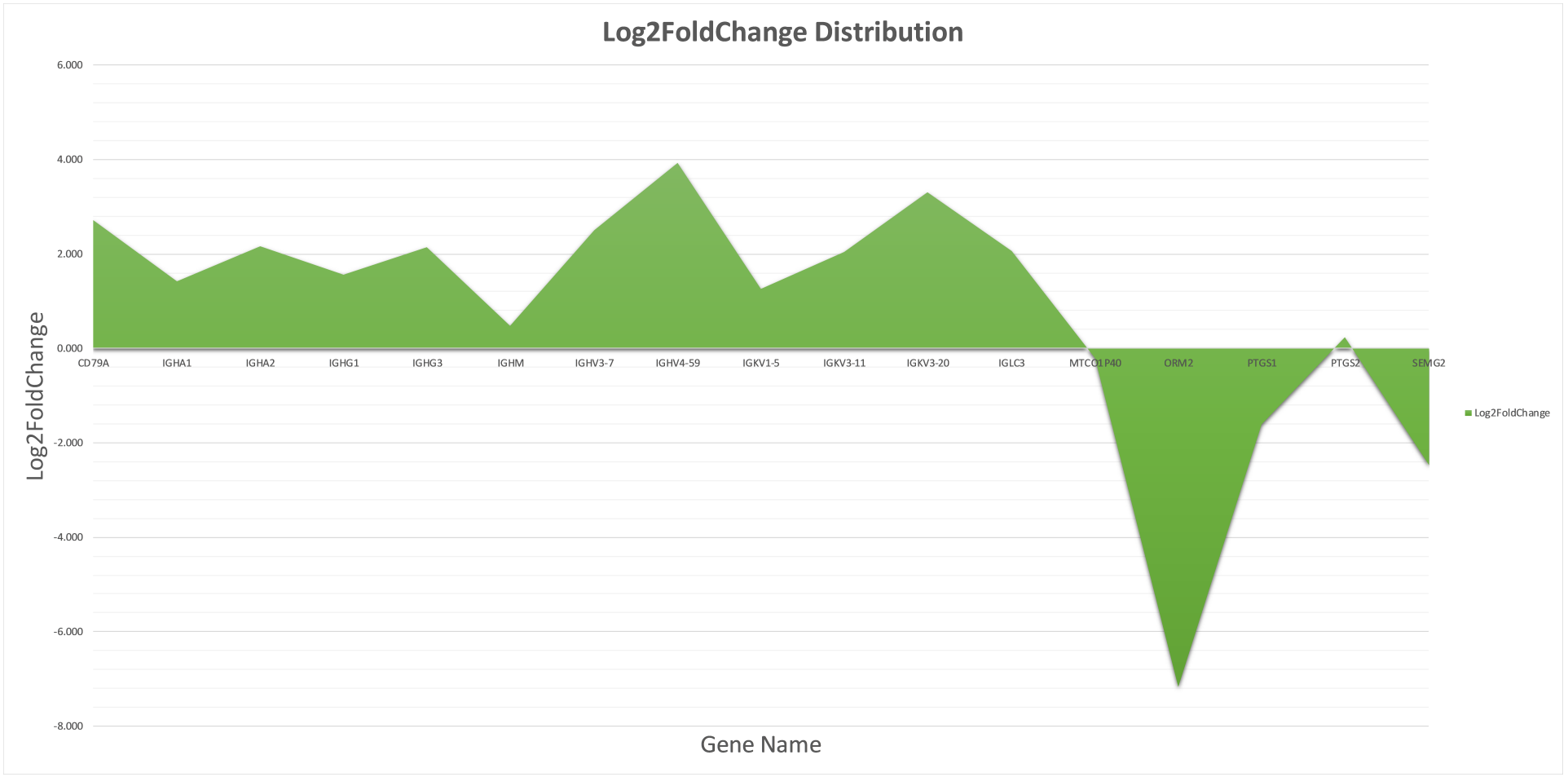


Figure 4.3: A plot of all the overlapping genes depicting the log2 fold change. Having both negative and positive values.

4.4 Combining Deferentially Expressed Genes and Neoantigen Prediction

Overlapping Genes

Gene Name	SB	WB	Log2 Fold Change	pAdja value
ALDH3B2	1	92	2.603	0.001
CCN3	6	6	-2.803	0.037
GABRP	15	20	-1.758	0.001
MS4A4E	0	14	2.236	0.019
NCOA7	0	14	-2.282	0.031
NUDT8	0	2	2.384	0.019
OLFM4	0	2	-3.276	0.003
RADIL	0	1	1.847	0.039
RARRES	0	1	-0.788	0.167
SIX5	0	56	1.075	0.193
SOD2	26	26	-1.975	0.005
SPAG1	0	14	-2.033	0.001
UBD	31	12	-2.649	0.043
ZP3	30	60	2.223	0.001

Table 4.4: A list of overlapping genes was extracted after the combination of the neoantigen prediction and differential analysis,Values of gene show the sum of WB(weak binders), the sum of SB(Strong Binders), the adj p-value and a count of how many times the gene appeared in each analysis.

Figure 4.4 was created from the output of the neoPredPipe tool (Table 4.1 and Figure 4.1) and the highly significant output of the differentially expressed genes (Table 4.3 and Figure 4.3). Results listed in table 4.4 show the genes that are present in both outputs.

Overall, fourteen genes were found to be overlapping with twelve of them having a significant pAdja value. Out of these twelve genes, five have shown a lot of potential in regards to neoantigen burden as it can be depicted by their predicted strong binders. After further analysis, only the gene CCN3 (Perbal *et al.*, 2006) was related to prostate cancer with a significant P-value of 0.037%. A log2 fold change of -2.803 suggests three-fold down-regulation of the CCN3 gene.

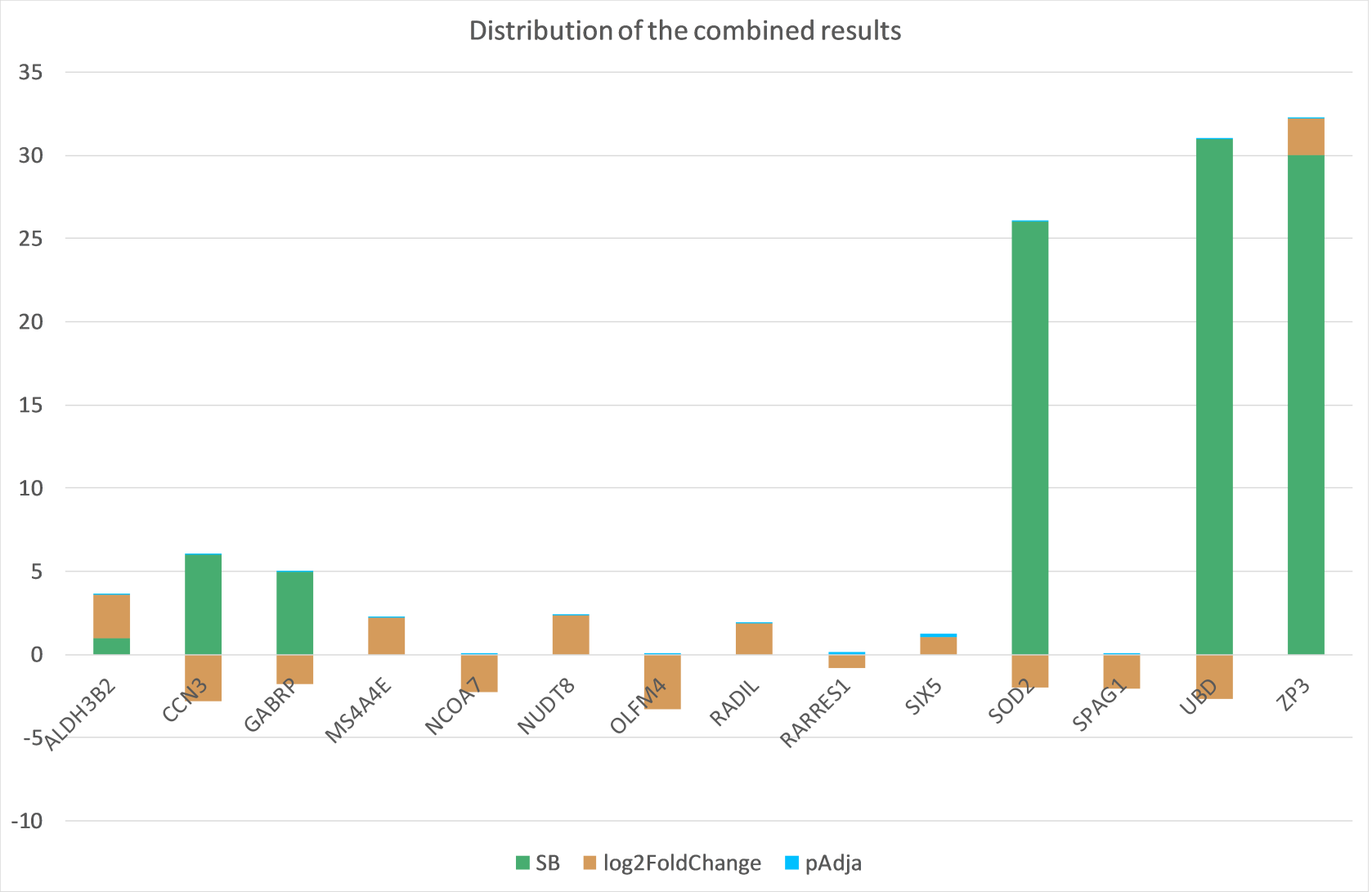


Figure 4.4: A plot of overlapping genes was extracted after the combination of the neoantigen prediction and differential analysis, Values show in this plot include; a count of SB(Strong Binders(green)), the adj p-value(light blue) and log2 Fold change values(orange).

4.5 GitHub

Many tools were used to develop this pipeline (4.5). One of these tools is GitHub. First of all, with version control, developers can easily record changes on a code, keep a back-up and have access to different versions of the development process, thus having the ability to recreate a particular version of the code (D., 2005).

Even though there are many simple ways to keep different versions of a project such as having a regular back-up of the code, these can cause a lot of errors.

GitHub Page

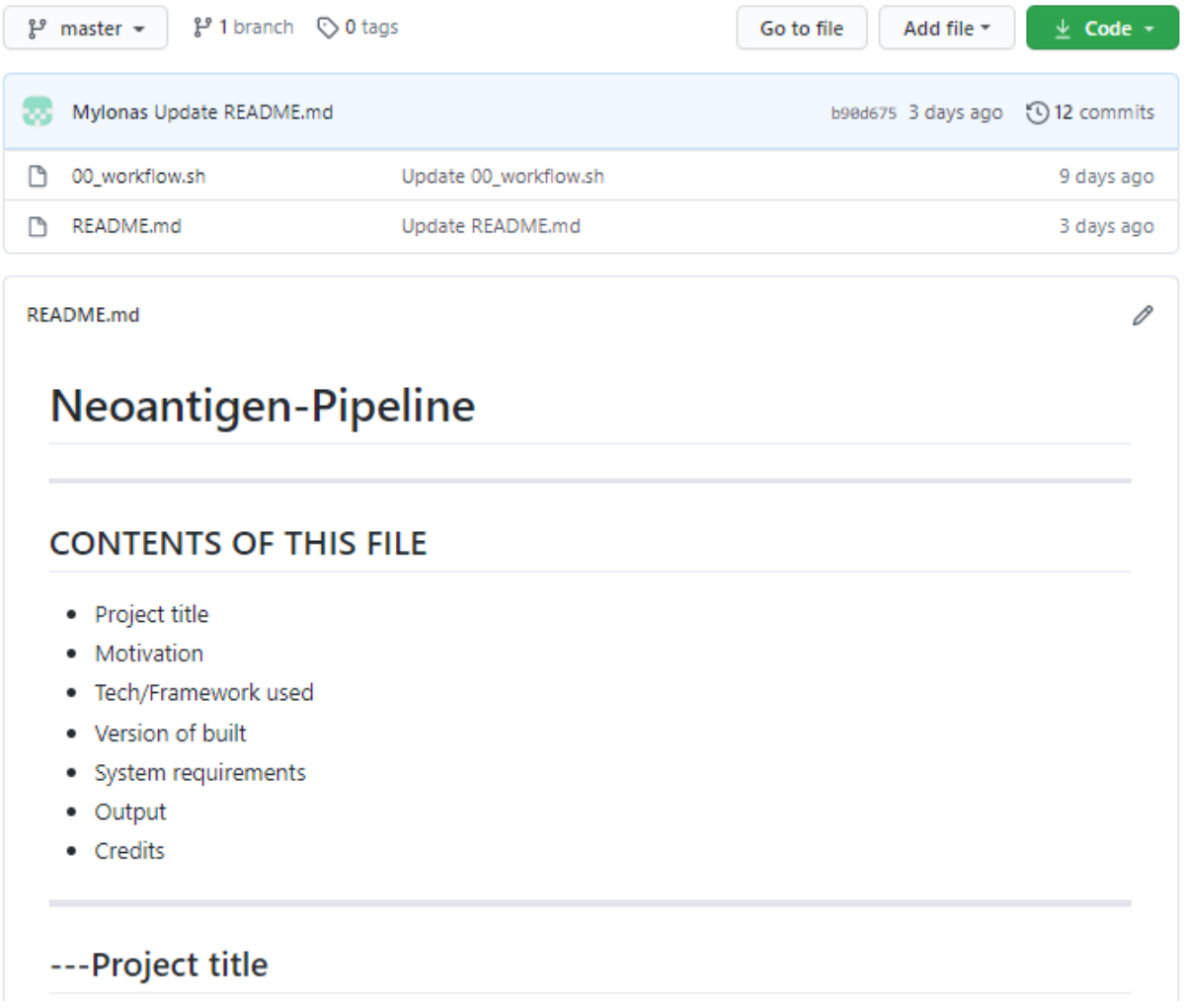


Figure 4.5: An example of the GitHub page set up for this project. The Figure displays parts of the readMe file where the different subsections are observable. In addition, at the top left corner the main script of the project is presented. The GitHub page can be found **here**.

However, since the pipeline was meant to be shared after completion for use by other bioinformaticians, it was very important to use a tool just for that. That's where GitHub can become a useful asset. Distribute version control systems such as GitHub exist solely for that purpose and are of great benefit to large organisations and corporations. In this particular scenario, having the code in a public directory such as GitHub can result in a lot of benefits, not only for the user but also for potential collaborators (S. and B., 2014).

GitHub can be broken down into the Git, and the Hub. The program provides access control, as well as a range of teamwork applications such as simple task management tools for other projects that you perform. GitHub has the ability to store the source code of a project in a variety of programming languages and can keep track of the various improvements that are made to each specific version.

It can never be deemed infallible to use an online archive, but it is a convenient and easy way to make the code and version history accessible online, thus providing an extra layer of security in case of damage to the local computer. Although this is enough for some people, it is recommended to have another contingency plan.

GitHub makes it much easier to maintain excellent documentation. Besides guides and articles there is also a lot of support from the GitHub community.

Additionally, with GitHub, it is much easier to showcase the work you have done. It allows developers to demonstrate their work and at the same time to share it with other members of the community.

An online GitHub is a simple solution that does not require new users to install additional software. On top of that, there is no need to login to the company's virtual private network (VPN), it might be better to upload files to a private GitHub repository. Having such an advantage makes it easier for those collaborating together on a project that is not part of a formal community – specifically open source projects.

Most developers are now comfortable with using GitHub and if they wish to contribute, it makes that task very simple and quick. Thus, this GitHub page (Website can be access **here**) was developed in order to easily share the pipeline.

Chapter 5

Discussion

Overall, very little is known about the structure and complexity of neoantigen-specific T-Cells (Magrini *et al.*, 2015b). To address this question, neoantigens were identified from sequencing data (Kannan *et al.*, 2011) in order to gain in-depth knowledge about this subject. This research concluded that analysis with high-affinity, patient-specific, tumour-derived neoantigens improves the predicted accuracy of the model.

Data-sets of thirty patients available from Baylor Prostate Specialised Research-Excellence Tissue Core (SPORE) (O. *et al.*, 1992) have been studied in the current research paper to predict neoantigens being presented by the autologous HLA molecules of each individual.

The average mean number of predicted neoantigens in the samples was 1283.833. Out of these, 25.232% were strong binders (SB) and out of these 26.546% had an affinity score of larger or equal to 1000, suggesting a extremely good prediction of neoantigen presentation, antigenicity and immunogenicity. With that said, the number of predicted neoantigens was not showing a pattern of a significant correlation with binding affinity regardless of their predicted value. This is similar to what has been recorded previously in (Fritsch *et al.*, 2014) and (Ghorani *et al.*, 2018).

The affinity score was crucial on defining strong and weak binders. Since this is a patient-specific vaccine, there were a number of tools used to predict which neoantigens will be the best fit for each individual. Therefore, the affinity score for each neoantigen it is certainly a way to differentiate "better" neoantigens. The lack of knowledge though in the SPORE (O. *et al.*, 1992) Tissue Core dataset did not allow samples to be stratified or their effect assessed for both tumour mutational burden and neoantigen load. Thus, the affinity score by itself is not enough to define the very best neoantigens. Other parameters need to be taken into account too.

The function of the HLA class I molecules has resulted in findings conveying no association with the number of predicted neoantigens related with the three main alleles. This indicates that such a factor will not provide a predictor for identifying high-scoring neoantigens.

Gene PLIN4 was identified as being highly potential for neoantigen selection. This showed a high recurrence number compared to other genes. Having a count of 234 among the most highly expressed genes. Further research showed PLIN4 to be over-expressed as well (Blanc *et al.*, 2019). This has also suggested PLIN4 to be a novel lipid that it is chemo resistant. Therefore, cross-validating the results of this study which have identified PLIN4 only in tumour tissue.

Consequently, further analysis was undertaken to determine any trends that may reveal a pattern which will support neoantigen prediction. Thus, the output from the differential analysis and neoPredPipe tool were combined. These resulted in the identification of gene CCN3. As mentioned, this gene was differentiated from other genes as it is aberrantly expressed in several other malignancies and is linked with the advancement of prostate cancer (Cadot *et al.*, 2002), Ewing's sarcoma (Perbal *et al.*, 2009) and renal cell carcinoma (Liu *et al.*, 2012). Particularly for prostate cancer, Studies have shown this to be up-regulated (Chen *et al.*, 2017) but also down-regulated (Dankner *et al.*, 2019) in prostate cancer. Indeed CCN3 has very conflicting roles that have resulted in poor patient prognosis (Dankner *et al.*, 2019). Further research will potentially lead the way in better understanding this gene. It could as well be the case of differentiation between early-stage and metastatic cancer as the same study (Dankner *et al.*, 2019) has also suggested CCN3 to be a mediator of prostate cancer metastasis.

Having said that, Figure 4.1 depicts a trend between samples. Half of the patients with cancer tissue have significantly fewer neoantigens compared to the other half suggesting some sort of different criteria that subdivide this group. This may seem to be fairly insignificant. However, taking that into account along with the conflicting role of CCN3, it may as well lead to some conclusions regarding CCN3, how it is expressed with prostate cancer and how all this may be related to cancer progression. This is also supported by a very recent study (Fang *et al.*, 2020) published a week ago by the time this was written. This study concluded that a higher success rate was achievable in patients with advanced solid tumours, This supports further the results of this study that indeed the stage of each individual's cancer may play a significant role in identifying the best neoantigen. If this is proven valid, it will lead the way in predicting better neoantigens as it will help to better understand success rates on neoantigen vaccines.

5.1 Limitations

Routinely maintenance on the university's supercomputer lead in some occasions in file corruption. The unstable system cause many major issues on different packages causing widespread problems and obstacles.

On top of that, NetChop has been developed for C-terminal proteasomal cleavage prediction. Besides that NetChop was published in 2005, its limited training set could also be an important limitation. Finally, most tests (Calis *et al.*, 2014) find that these techniques, when used in combination with MHC I binding predictors, do not increase precision.

Numerous possible complications could prevent adequate immune responses. Sometimes the expression of MHC complexes which are essential for peptide processing may be limited to cells (Wieczorek *et al.*, 2017).

The subject was so interesting and it was indeed really unfortunate to have such a limited time to undertake this study. It would be great to have more time to expand the research and there are numerous ways in which this can be accomplished.

5.2 Future Work

As with all tools, there are always areas for expansion and improvement. Additionally, due to new framework editions and new software as well as updates of existing software, there will be the need for the constant support of this tool. At the same time, as infrastructure improves there will be the possibility of introducing new features and the ability to improve the efficiency of the pipeline.

Improving GitHub can result in better adaptability of the tool. This may allow work to be done on other frameworks too. As a result, this will simplify this procedure even more.

Another area of expansion will be to improve the user interface. Tools such as NextFlow (Federico *et al.*, 2019), SnakeMake (Köster *et al.*, 2012) and Luigi (Rieger *et al.*, 2017) are well known for pipeline management (Leipzig, 2017). This can aid the simplification even more and will also make it more accessible, a tool that will be easier to use.

Additionally, recent data showed the TCRs can be diversified in patients; this research paper (Magrini *et al.*, 2015a) suggests a therapeutic strategy of testing checkpoint inhibitors. Similar classifications of genomic alterations(i.e. deletions, insertions, and frame-shift mutations (Liu *et al.*, 2017a)) could as well be a target for potential neoantigens (Mardis, 2019). Extraction of these neoantigens could be especially relevant in malignancies which were shown to have a low mutational burden, thus, making a perfect field in which this research can contribute to explore possibilities of improving the neoantigen accuracy prediction.

5.3 Conclusion

Targeting peptides are an appealing alternative as vaccines because of their ability to work directly as key T-Cell peptides. On top of that, peptides include their low toxicity profiles, specific targeting, low relative cost and simple and direct chemical synthesis. In addition, numerous peptide epitopes may be incorporated in a single vaccine, raising the odds of activating multiple T-cells and avoiding loss or changes in peptides during tumour progression, which in turn prevents tumour escape.

Finally, it would be really beneficial to validate the viability of the neoantigen technique for clinical studies. This will require a lot of detailed sorting of individual data before the vaccinations are ready for wider use. Recognising the neoantigens formed by the tumour of a person and then identifying if any of those proteins could be recognised by the patient's immune system is still incredibly challenging.

Bibliography

A., F. *et al.* 2009. Estructura genética de tres poblaciones mexicanas en base al sistema hla-a. *Revista de Salud Pública y Nutrición* 10.

Auton, A. *et al.* 2015. A global reference for human genetic variation. *Nature* 526(7571), pp. 68–74. Available at: https://doi.org/10.1038/nature15393.

Blanc, E. *et al.* 2019. Identification and ranking of recurrent neo-epitopes in cancer. *BMC medical genomics* 12(1), pp. 171–171. Available at: https://pubmed.ncbi.nlm.nih.gov/31775766. 31775766[pmid].

Blankenstein, T. *et al.* 2015. Targeting cancer-specific mutations by t cell receptor gene therapy. *Current opinion in immunology* 33, pp. 112–119. Available at: https://pubmed.ncbi.nlm.nih.gov/25728991. 25728991[pmid].

Bolotin, D. A. *et al.* 2015. Mixcr: software for comprehensive adaptive immunity profiling. *Nature Methods* 12(5), pp. 380–381. Available at: https://doi.org/10.1038/nmeth.3364.

Cadot, B. *et al.* 2002. Differential expression of the ccn3 (nov) proto-oncogene in human prostate cell lines and tissues. *Molecular Pathology* 54(4), pp. 275–280. Available at: https://mp.bmj.com/content/54/4/275.

Calis, J. J. A. *et al.* 2014. Role of peptide processing predictions in t cell epitope identification: contribution of different prediction programs. *Immunogenetics* 67(2), pp. 85–93. Available at: https://doi.org/10.1007/s00251-014-0815-0.

Chen, F. *et al.* 2019. Neoantigen identification strategies enable personalized immunotherapy in refractory solid tumors. *The Journal of Clinical Investigation* 129(5), pp. 2056–2070. Available at: https://www.jci.org/articles/view/99538.

Chen, P.-C. *et al.* 2017. Ccn3 promotes epithelial-mesenchymal transition in prostate cancer via fak/akt/hif-1α-induced twist expression. *Oncotarget* 8(43), pp. 74506–74518. Available at: https://pubmed.ncbi.nlm.nih.gov/29088803. 29088803[pmid].

Chu, Y. *et al.* 2018. Personalized cancer neoantigen vaccines come of age. *Theranostics* 8(15), pp. 4238–4246. Available at: https://pubmed.ncbi.nlm.nih.gov/30128050. 30128050[pmid].

Cingolani, P. *et al.* 2012. A program for annotating and predicting the effects of single nucleotide polymorphisms, snpeff: Snps in the genome of drosophila melanogaster strain w1118; iso-2; iso-3. *Fly* 6(2), pp. 80–92. Available at: https://pubmed.ncbi.nlm.nih.gov/22728672. 22728672[pmid].

D., S. 2005. Version control systems. *IEEE Software* 22(5), pp. 108–109.

Daehwan, K. *et al.* 2013. Tophat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome Biology* 14(4), p. R36. Available at: https://doi.org/10.1186/gb-2013-14-4-r36.

Damoiseaux, J. G. M. C. *et al.* 1999. A dominant role for the thymus and mhc genes in determining the peripheral cd4/cd8 t cell ratio in the rat. *The Journal of Immunology* 163(6), pp. 2983–2989. Available at: https://www.jimmunol.org/content/163/6/2983.

Dankner, M. *et al.* 2019. CCN3/nephroblastoma overexpressed is a functional mediator of prostate cancer bone metastasis that is associated with poor patient prognosis. *The American Journal of Pathology* 189(7), pp. 1451–1461. Available at: https://doi.org/10.1016/j.ajpath.2019.04.006.

Derhovanessian, E. *et al.* 2017. Personalized rna mutanome vaccines mobilize poly-specific therapeutic immunity against cancer. *Nature* 547(7662), pp. 222–226. Available at: https://doi.org/10.1038/nature23003.

Diamandis, E. P. *et al.* 2016. Cancer immunotherapy: the beginning of the end of cancer? *BMC Medicine* 14(1), p. 73. Available at: https://doi.org/10.1186/s12916-016-0623-5.

Fang, Y. *et al.* 2020. A pan-cancer clinical study of personalized neoantigen vaccine monotherapy in treating patients with various types of advanced solid tumors. *Clinical Cancer Research* 26(17), pp. 4511–4520. Available at: https://clincancerres.aacrjournals.org/content/26/17/4511.

Federico, A., *et al.* 2019. Pipeliner: A nextflow-based framework for the definition of sequencing data processing pipelines. *Frontiers in Genetics* 10, p. 614. Available at: https://www.frontiersin.org/article/10.3389/fgene.2019.00614.

Fritsch, E. *et al.* 2016. *Shared neoantigens*, Available at:. Available at: https://patents.google.com/patent/CN108025048A/en.

Fritsch, E. F. *et al.* 2014. Hla-binding properties of tumor neoepitopes in humans. *Cancer immunology research* 2(6), pp. 522–529. Available at: https://pubmed.ncbi.nlm.nih.gov/24894089. 24894089[pmid].

Ghorani, E. *et al.* 2018. Differential binding affinity of mutated peptides for mhc class i is a predictor of survival in advanced lung cancer and melanoma. *Annals of oncology : official journal of the*

European Society for Medical Oncology 29(1), pp. 271–279. Available at: https://pubmed.ncbi.nlm.nih.gov/29361136. 29361136[pmid].

Hollingsworth, R. E. and Jansen, K. 2019. Turning the corner on therapeutic cancer vaccines. *npj Vaccines* 4(1), p. 7. Available at: https://doi.org/10.1038/s41541-019-0103-y.

Hoof, I. *et al.* 2009. NetMHCpan, a method for MHC class I binding prediction beyond humans. *Immunogenetics* 61(1), pp. 1–13.

Hu, Z. *et al.* 2017. An immunogenic personal neoantigen vaccine for patients with melanoma. *Nature* 547(7662), pp. 217–221. Available at: https://doi.org/10.1038/nature22991.

Hutchinson, J. A. *et al.* 2015. HLA typing. *Transplantation* 99(1), pp. 6–7. Available at: https://doi.org/10.1097/tp.0000000000000531.

I., M. *et al.* 2014. Moderated estimation of fold change and dispersion for rna-seq data with deseq2. *J. Mach. Learn. Res.* 12(null), p. 2825–2830.

Kannan, K. *et al.* 2011. Recurrent chimeric RNAs enriched in human prostate cancer identified by deep sequencing. *Proc. Natl. Acad. Sci. U.S.A.* 108(22), pp. 9172–9177.

Kersten, S. 2001. Mechanisms of nutritional and hormonal regulation of lipogenesis. *EMBO reports* 2(4), pp. 282–286. Available at: https://pubmed.ncbi.nlm.nih.gov/11306547. 11306547[pmid].

Khailaie, S. *et al.* 2013. A mathematical model of immune activation with a unified self-nonself concept. *Frontiers in immunology* 4, pp. 474–474. Available at: https://pubmed.ncbi.nlm.nih.gov/24409179. 24409179[pmid].

Köster, J. *et al.* 2012. Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics* 28(19), pp. 2520–2522. Available at: https://doi.org/10.1093/bioinformatics/bts480.

Lee, M. *et al.* 2018. Cancer immuotherapy for hepatocellular carcinoma. *Hepatoma Research* 4, p. 51.

Leipzig, J. 2017. A review of bioinformatic pipeline frameworks. *Briefings in bioinformatics* 18(3), pp. 530–536. Available at: https://pubmed.ncbi.nlm.nih.gov/27013646. 27013646[pmid].

Li, H. *et al.* 2009. The Sequence Alignment/Map format and SAMtools. *Bioinformatics* 25(16), pp. 2078–2079.

Li, M. *et al.* 2010. ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. *Nucleic Acids Research* 38(16), pp. e164–e164. Available at: https://doi.org/10.1093/nar/gkq603.

Liao, Y. *et al.* 2013. featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics* 30(7), pp. 923–930. Available at: https://doi.org/10.1093/bioinformatics/btt656.

Liu, G. *et al.* 2017a. Genomics alterations of metastatic and primary tissues across 15 cancer types. *Scientific Reports* 7(1), p. 13262. Available at: https://doi.org/10.1038/s41598-017-13650-3.

Liu, S. *et al.* 2012. Ccn3 (nov) regulates proliferation, adhesion, migration and invasion in clear cell renal cell carcinoma. *Oncology letters* 3(5), pp. 1099–1104. Available at: https://pubmed.ncbi.nlm.nih.gov/22783399. 22783399[pmid].

Liu, X. S. *et al.* 2017b. Applications of immunogenomics to cancer. *Cell* 168(4), pp. 600–612. Available at: https://pubmed.ncbi.nlm.nih.gov/28187283. 28187283[pmid].

Luckheeram, R. V. *et al.* 2012. Cd4⁺t cells: Differentiation and functions. *Clinical and Developmental Immunology* 2012, p. 925135. Available at: https://doi.org/10.1155/2012/925135.

Magrini, V. *et al.* 2015a. Cancer immunotherapy. a dendritic cell vaccine increases the breadth and diversity of melanoma neoantigen-specific t cells. *Science (New York, N.Y.)* 348(6236), pp. 803–808. Available at: https://pubmed.ncbi.nlm.nih.gov/25837513. 25837513[pmid].

Magrini, V. *et al.* 2015b. A dendritic cell vaccine increases the breadth and diversity of melanoma neoantigen-specific t cells. *Science* 348(6236), pp. 803–808. Available at: https://science.sciencemag.org/content/348/6236/803.

Mardis, E. R. 2019. Neoantigens and genome instability: impact on immunogenomic phenotypes and immunotherapy response. *Genome Medicine* 11(1), p. 71. Available at: https://doi.org/10.1186/s13073-019-0684-0.

Nakane, T. *et al.* 2019. Identification of mammalian glycoproteins with type-i lacdinac structures synthesized by the glycosyltransferase b3galnt2. *The Journal of biological chemistry* 294(18), pp. 7433–7444. Available at: https://pubmed.ncbi.nlm.nih.gov/30898876. 30898876[pmid].

Noris, M. and Remuzzi, G. 2013. Overview of complement activation and regulation. *Seminars in nephrology* 33(6), pp. 479–492. Available at: https://pubmed.ncbi.nlm.nih.gov/24161035. 24161035[pmid].

O., K. *et al.* 1992. Specialized programs of research excellence (spores). *Immunogenetics* Available at: https://www.bcm.edu/academic-centers/lester-and-sue-smith-breast-center/research/spore.

Oikkonen, L. and Lise, S. 2017. Making the most of rna-seq: Pre-processing sequencing data with opossum for reliable snp variant detection. *Wellcome open research* 2, pp. 6–6. Available at: https://pubmed.ncbi.nlm.nih.gov/28239666. 28239666[pmid].

Orenbuch, R. *et al.* 2019. arcasHLA: high-resolution HLA typing from RNAseq. *Bioinformatics* 36(1), pp. 33–40. Available at: https://doi.org/10.1093/bioinformatics/btz474.

Ovid. 1946. Ovid medline database 18. Available at: https://www.ovid.com/product-details.901.html.

Patente, T. A. *et al.* 2019. Human dendritic cells: Their heterogeneity and clinical application potential in cancer immunotherapy. *Frontiers in Immunology* 9, p. 3176. Available at: https://www.frontiersin.org/article/10.3389/fimmu.2018.03176.

Pedregosa, F. *et al.* 2011. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.* 12(null), p. 2825–2830.

Peng, M. *et al.* 2019. Neoantigen vaccine: an emerging tumor immunotherapy. *Molecular Cancer* 18(1), p. 128. Available at: https://doi.org/10.1186/s12943-019-1055-6.

Perbal, B. *et al.* 2006. Nov story: the way to ccn3. *Cell Communication and Signaling* 4(1), p. 3. Available at: https://doi.org/10.1186/1478-811X-4-3.

Perbal, B. *et al.* 2009. Prognostic relevance of ccn3 in ewing sarcoma. *Human Pathology* 40(10), pp. 1479 – 1486. Available at: http://www.sciencedirect.com/science/article/pii/S0046817709001932.

Rajasagi, M. *et al.* 2015. Tumor Neoantigens Are Abundant Across Cancers. *Blood* 122(21), pp. 3265–3265. Available at: https://doi.org/10.1182/blood.V122.21.3265.3265.

Richardson, K. *et al.* 2011. The plin4 variant rs8887 modulates obesity related phenotypes in humans through creation of a novel mir-522 seed site. *PLOS ONE* 6(4), pp. 1–11. Available at: https://doi.org/10.1371/journal.pone.0017944.

Richters *et al.* 2019. Best practices for bioinformatic characterization of neoantigens for clinical utility. *Genome Medicine* 11(1), p. 56. Available at: https://doi.org/10.1186/s13073-019-0666-2.

Rieger, M. *et al.* 2017. Design and execution of make-like, distributed analyses based on spotify's pipelining package luigi. *Journal of Physics: Conference Series* 898.

Rosenberg, S. A. *et al.* 2015. Adoptive cell transfer as personalized immunotherapy for human cancer. *Science (New York, N.Y.)* 348(6230), pp. 62–68. Available at: https://pubmed.ncbi.nlm.nih.gov/25838374. 25838374[pmid].

S., C. and B., S. 2014. PRO GIT Everything you need to know, Available at:. https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control [Accessed: July 2020].

Saxová, P. *et al.* 2003. Predicting proteasomal cleavage sites: a comparison of available methods. *International Immunology* 15(7), pp. 781–787. Available at: https://doi.org/10.1093/intimm/dxg084.

Schenck, R. O. *et al.* 2019. Neopredpipe: high-throughput neoantigen prediction and recognition potential pipeline. *BMC Bioinformatics* 20(1), p. 264. Available at: https://doi.org/10.1186/s12859-019-2876-4.

Schumacher, T. N. and Schreiber, R. D. 2015. Neoantigens in cancer immunotherapy. *Science* 348(6230), pp. 69–74. Available at: https://doi.org/10.1126/science.aaa4971.

Topalian, S. L. *et al.* 2011. Cancer immunotherapy comes of age. *Journal of clinical oncology : official journal of the American Society of Clinical Oncology* 29(36), pp. 4828–4836. Available at: https://pubmed.ncbi.nlm.nih.gov/22042955. 22042955[pmid].

Tsukumo, S.-i. *et al.* 2018. Regulation of cd8+ t cells and antitumor immunity by notch signaling. *Frontiers in Immunology* 9, p. 101. Available at: https://www.frontiersin.org/article/10.3389/fimmu.2018.00101.

Wagner, S. *et al.* 2018. Colorectal cancer vaccines: Tumor-associated antigens vs neoantigens. *World journal of gastroenterology* 24(48), pp. 5418–5432. Available at: https://pubmed.ncbi.nlm.nih.gov/30622371. 30622371[pmid].

Wieczorek, M. *et al.* 2017. Major histocompatibility complex (mhc) class i and mhc class ii proteins: Conformational plasticity in antigen presentation. *Frontiers in immunology* 8, pp. 292–292. Available at: https://pubmed.ncbi.nlm.nih.gov/28367149. 28367149[pmid].

Wirth, T. C. *et al.* 2017. Neoantigen targeting-dawn of a new era in cancer immunotherapy? *Frontiers in immunology* 8, pp. 1848–1848. Available at: https://pubmed.ncbi.nlm.nih.gov/29312332. 29312332[pmid].

Yin, Q. *et al.* 2019. Next-generation sequencing technologies accelerate advances in t-cell therapy for cancer. *Briefings in functional genomics* 18(2), p. 119—128. Available at: https://europepmc.org/articles/PMC6488971.

Zhang, J.-Y. *et al.* 2009. Identification of tumor-associated antigens as diagnostic and predictive biomarkers in cancer. *Methods in molecular biology (Clifton, N.J.)* 520, pp. 1–10. Available at: https://pubmed.ncbi.nlm.nih.gov/19381943. 19381943[pmid].

Chapter 6

Appendix

6.1 Appendix A: Introduction

6.1.1 Validating Data sets

Validating Data sets **Date: 10/06/2020**

Neoantigen Database Comparison

Database Name	NeoPeptide	TSNAdb	dbPepNeo
Main Information for each entry	• Gene • Mutation • Position in Peptide • HLA allele • Mut Affinity • Mut Rank • Peptide • Verification • Predicted binding level	• Gene • Mutation • Position in Peptide • HLA allele • Peptide • Predicted binding level • IEDB(Immune epitope database and analysis resource) result • IEDB assay ID	• Cancer • Gene • HLA allele • Mut peptide • Mut Affinity • Mut Rank • Bind Level • Wt Peptide • Peptide Length • Verification • Mutation • Reference • Confidence
Data Resources	No information was provided	• TCGA(Cancer Genome Atlas) • HLA(human leukocyte antigen) allele data from TCIA	• Immune Epitope Database • Cancer Immunity Peptide Database
Neoantigen Prediction Tools	No information was provided	• NetMHCpan, v2.8 • NetMHCpan v4.0	• ProGeo-neo NetMHCpan (v.4.0) MaxQuant • INeo-Epp random forest algorithm
Database size	181137 epitopes derived from more than 36000 neoantigens	• 7748 samples of 16 tumors • 3707562 and 1146961 neoantigens predicted of whom 716876 were found in both predictions	• 295 high confidence(hc) • 247 medium confidence(mc) • 407794 low condifence(lc)
First Build	December 2019	August 2018	June 2019

Table 6.1: A comprehensive comparison between potential databases containing test data. First row contains the name of each database. The second row contains all the information that can be found for each sample listed in the database. Subsequent rows provide validation criteria.

Description:

A comprehensive analysis to compare different data sets. This analysis was accomplished in order to gain a better understanding of neoantigen databases and to also define certain data sets that may be used as test data.

6.2 Appendix B: Literature Review & Aims

6.2.1 Mindmap for literature review

MindMap of Literature Review **Date: 02/07/2020**

Literature Review Search Strategy

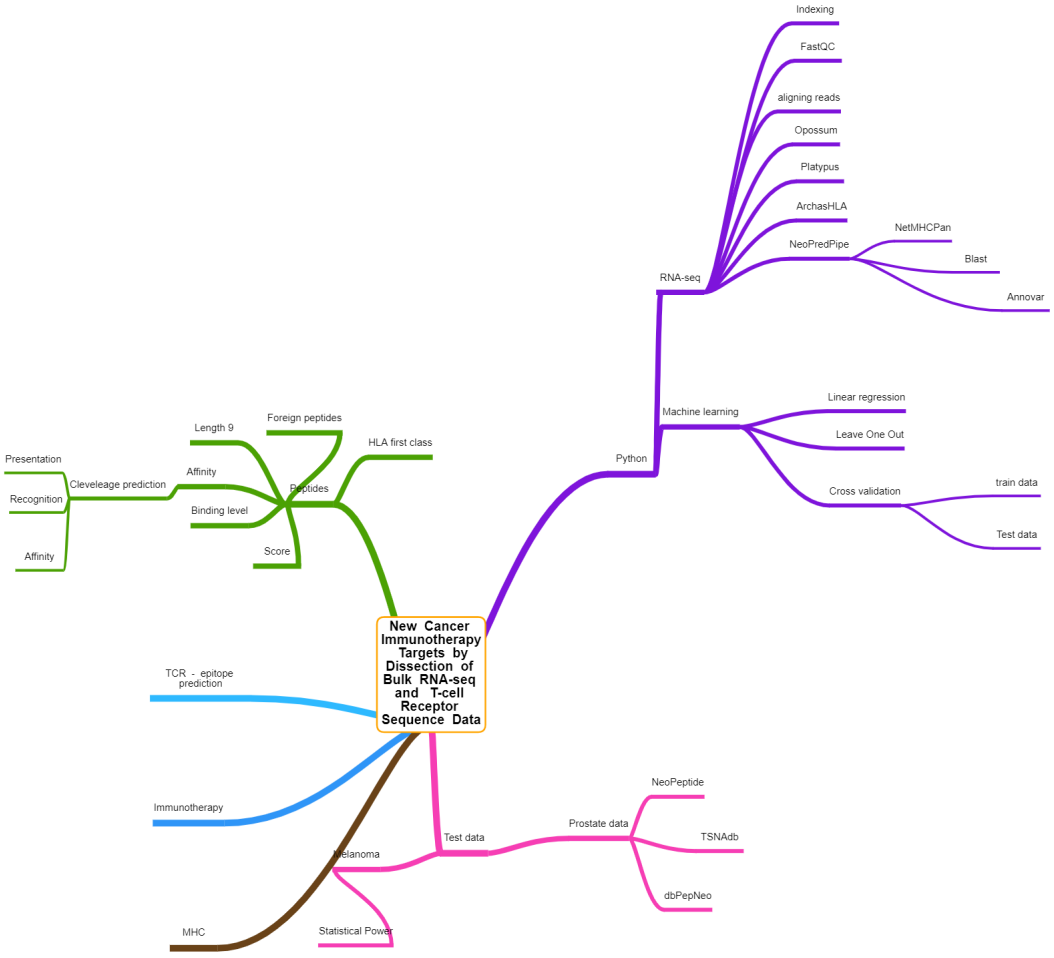


Figure 6.1: A simplistic literature review diagram. This diagram spans in four different areas of the project and address all related parts from machine learning and computational methods to the biology behind neoantigens.

Description:

The literature review was performed with the Ovid medline database. The purpose of it was to gain a better understating of the literature related to this project and gain an in-depth understanding of different methods and tools. On top of that, to provide the reader of this document all the relevant information over key findings. Thus, this mind-map conveys the different aspects of this project. In addition, it addresses multiple disciplines related to it.

6.3 Appendix C: Methods

6.3.1 Whole pipeline listing

Complete pipeline listing **Date: 12/08/2020**

```
#!/bin/bash

sampleIDs=("SRR057629" "SRR057630" "SRR057631" "SRR057632" "SRR057633" "
    SRR057634" "SRR057635" "SRR057636" "SRR057637" "SRR057638" "SRR057639" "
    SRR057640" "SRR057641" "SRR057642" "SRR057643" "SRR057644" "SRR057645" "
    SRR057646" "SRR057647" "SRR057648" "SRR057649" "SRR057650" "SRR057651" "
    SRR057652" "SRR057653" "SRR057654" "SRR057655" "SRR057656" "SRR057657" "
    SRR057658")

genomeIndex="Homo_sapiens.GRCh37.dna_sm.primary_assembly"
gtfIndex="Homo_sapiens.GRCh37.87"
  echo "module load tophat/2.0.11" >> ${scriptName}
  echo "module load FastQC/0.11.8" >> ${scriptName}

  #echo "fastqc -o ${myDir}/../output/${sampleID}/fastqc/raw/ ${myDir}/../input/${
      sampleID}_1.fastq.gz ${myDir}/../input/${sampleID}_2.fastq.gz" >> ${scriptName
      }

  ## run the tophat mapping command
  #echo "tophat2 --transcriptome-index ${myDir}/../resources/${gtfIndex} -o ${
      myDir}/../output/${sampleID}/tophat ${myDir}/../resources/${genomeIndex} ${
      myDir}/../input/${sampleID}_1.fastq.gz ${myDir}/../input/${sampleID}_2.fastq.
      gz" >> ${scriptName}

  ## Data pre processing with opossum and variant calling with platypus
  echo "module purge" >> ${scriptName}
  echo "module load python-h5py/2.8.0" >> ${scriptName}
  echo "pip install --user pysam==0.10.0 more-itertools argparse os-win runner" >>
      ${scriptName}

  echo "mkdir -p ${myDir}/../output/${sampleID}/opossum/" >> ${scriptName}
  echo "mkdir -p ${myDir}/../output/${sampleID}/platypus/" >> ${scriptName}

  ##run fastaAlternator
  #echo "gatk IndexFeatureFile -F ${myDir}/../output/${sampleID}/platypus/${
      sampleID}.vcf" >> ${scriptName}
  #echo "gatk FastaAlternateReferenceMaker -R ${myDir}/../resources/${
```

```
        genomeIndex}.fa -O ${myDir}/../output/${sampleID}/gatk/${sampleID}.fasta -V
        ${myDir}/../output/${sampleID}/platypus/${sampleID}.vcf" >> ${scriptName}
26
27  ## run getfasta to create a chr delited fasta file
28  #echo "mkdir -p ${myDir}/../output/${sampleID}/getfasta/" >> ${scriptName}
29  #echo "module purge" >> ${scriptName}
30  #echo "module load compiler/gnu/7" >> ${scriptName}
31  #echo "module load mpi/intel/2018/2" >> ${scriptName}
32  #echo "module load hdf5" >> ${scriptName}
33  #echo "module load bedtools" >> ${scriptName}
34  #echo "bedtools getfasta -fi ${myDir}/../output/${sampleID}/gatk/${sampleID}.
        fasta -bed ${myDir}/../resources/${gtfIndex}.gtf -fo ${myDir}/../output/${
        sampleID}/getfasta/${sampleID}.fasta" >> ${scriptName}
35
36  ## run netchop
37  echo "mkdir -p ${myDir}/../output/${sampleID}/netchop/" >> ${scriptName}
38  echo "netchop-3.1/netchop ${myDir}/../output/${sampleID}/getfasta/${sampleID}.
        fasta > ${myDir}/../output/${sampleID}/netchop/${sampleID}.out" >> ${
        scriptName}
39  #echo "netchop-3.1/netchop  ${myDir}/../output/${sampleID}/gatk/${sampleID}.
        fasta >  ${myDir}/../output/${sampleID}/netchop/${sampleID}.out" >> ${
        scriptName}
40
41  chmod u+x ${scriptName}
42
43  sbatch ${scriptName}
44
45 done
```

Code Listing 6.1: Complete pipeline of the project

Description:

Neoantigens are newly developed peptides produced from genetic variations capable of inducing the identification of tumour-specific T cells. Scientists and clinicians have lately leveraged next-generation sequencing tools to classify neoantigens and deliver customised immunotherapies for treating cancer. Neoantigens need to be computationally predicted from matched tumour–normal sequencing data to produce a unique cancer vaccine, and then classified according to their predicted ability to stimulate a T-cell response. This neoantigen prediction pipeline includes several steps such as the detection of somatic mutations, HLA typing, peptide processing, and binding prediction of peptide-MHC as it can be depicted in figure 2.1.

6.3.2 MixCR complete code listing

MixCR complete code listing **Date: 02/08/2020**

```
1  ## run align on the raw fastq
2   ./mixcr align  \
3   ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
        fasta # input
4   ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
        vdjca # output
5
6   ## run export on the clones
7  ./mixcr exportClones \
8  ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
        clns \
9  ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}_
        aligned.txt
10
11 ## Optionally run export on the clns with a csv file format output
12 ./mixcr exportClonesPretty \
13 ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}.
        clns \
14 ${myDir}/../../Prostate_Adenocarcinoma/output/${sampleID}/mixcr/${sampleID}_
        aligned.csv
```

Code Listing 6.2: MixCR on the the FASTQ data for reading alignments

Description: This is another tool for next generation sequencing analysis. In particular, mixCR was used for the alighment of the immune sequencing data. The entire code for the mixCR pipeline can be split in the following five listing.

The first listing(6.2) is just for aligning the raw fasta files. The output of this step is a non human readable format. However, is the format required for the next step of the analysis.

Then, listing 6.2 is the step for assembling clones.The command mixcr assemble will create a report file from the out of the alignment file of the previous step. This step is computationally expensive but it is necessary to create a binary file.

The final step is for producing a file in a txt format that contains all the clones from the alignment files. An an optional step, the keyword pretty can be added as show in listing 6.2 line 7 to obtain a better looking report in which is also comma separated.

6.3.3 Complete listing of python code

Python code listing **Date: 20/08/2020**

```
from collections import defaultdict

from scipy.stats import stats
import random
import operator
import math
import seaborn as seabornInstance
import seaborn as sns  # Plotting library

#def f(x):
 #   return np.int(x)


#def rmse(predictions, targets):  # root mean squared error
 #   predictions = predictions.astype(int)
  #  targets = targets.astype(int)
   # f3 = np.vectorize(f)
    #return (np.sqrt(np.mean((float((predictions)) - ((targets))) ** 2)))


def agglomerativeClustering():

    dataset = read_df()
    dataset = dataset[['BindLevel', 'Gl', 'Gp', 'Ip']].values
    data = dataset

def read_df():
    # Convert to DB.. style names
    df = pd.read_csv("./out.txt", sep="\t")

    print(df.tail())


    # df = pd.read_csv("./matching.txt", sep="\t")

    #test_data_length_9 = pd.read_csv("./Prostate_test_data_length_9.txt", sep="\t")
    # df = test_data_length_9 # for test data

```

```
39      return df
40
41  def regression(regressor, x_name, y_name,df):
42
43      # Make plot of distribution
44      df.plot(x=x_name, y=y_name, style='o')
45      plt.title(x_name + " vs " + y_name)
46      plt.xlabel(x_name)
47      plt.ylabel(y_name)
48      plt.show()
49
50      plt.figure(figsize=(15, 10))
51      plt.tight_layout()
52      seabornInstance.distplot(df[y_name])
53
54      # Define train and test data
55      X = df[x_name].values.reshape(-1, 1)
56      y = df[y_name].values.reshape(-1, 1)
57
58      # Split data in sets
59      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
60
61      regressor.fit(X_train, y_train)  # training the algorithm
62
63      # Print separating line
64      print("." * 80)
65      print("Predictions -----" + " " + regressor.__class__.__name__)
66
67      # For retrieving score
68      print("Score: ", regressor.score(X, y, sample_weight=None))
69      print("RMSE: ", mean_squared_error(X_train, y_train))
70      #print("Correlation Coefficient: ", stats.pearsonr(X_train, y_train))
71
72
73  def main():
74      df = read_df()
75      #df = pd.DataFrame(df)
76
77      x_name = "BindLevel"
78      y_name = "Rank"
79
```

```
80  X = df[x_name].values
81  y = df[y_name].values
82
83  regression(LinearRegression(), x_name, y_name, df)
84  #regression(Ridge(alpha=.5), x_name, y_name, df)
85  #regression(neighbors.KNeighborsRegressor(), x_name, y_name, df)
86  #regression(DecisionTreeRegressor(random_state=0), x_name, y_name, df)
87  #regression(RANSACRegressor(random_state=0), x_name, y_name, df)
88  #regression(VotingRegressor([('lr', LinearRegression()), ('rf', RandomForestRegressor
        (n_estimators=10, random_state=1))]), x_name, y_name, df)
89
90  # get samples column and make it a list
91  samples = list(df["Sample"].values)
92
93  # count how many neoantigens for each sample
94  for patient in set(samples):
95      print(patient + " - has " + str(samples.count(patient)) + " neoantigens")
96
97
98  #patient_unique =[]
99  mySamples = list(dict.fromkeys(samples))
100 #print(mySamples)
101
102 # count weak snd strong binders, get average ranking
103 for patient in mySamples:
104     dataset = df[df['Sample'].str.contains(patient)]
105     print(patient + " - has " + str(list(dataset["BindLevel"].values).count("0")) + "
        Weak Binders and " + str(list(dataset["BindLevel"].values).count("1")) + " Strong
        Binders"
106         + "average Ranking value is " + str(dataset["Rank"].mean()))
107     #print(patient + " - has " + str(list(dataset["BindLevel"].values).count("1")) + "
        Strong Binders")
108
109 df['Score'] = df['Score'].apply((np.ceil))
110 df['Rank'] = df['Rank'].apply((lambda x:x*1000))
111 df['Rank'] = df['Rank'].apply((np.ceil))
112
113 Gl = (list(map(int, df["Gl"].values)))
114 Of = (list(map(int, df["Of"].values)))
115 Gp = (list(map(int, df["Gp"].values)))
116 Score = (list(map(int, df["Score"].values)))
```

```
117     Rank = (list(map(int, df["Rank"].values)))
118     BindLevel = list(map(int, df["BindLevel"].values))
119 
120     list_of_tuples = list(zip(Score, BindLevel, Gl, Of, Gp, Rank))
121     df2 = pd.DataFrame(list_of_tuples, columns=['Score', 'BindLevel', 'Gl', 'Of', 'Gp', '
            Rank'])
122     print(df2)
123     print(stats.pearsonr(BindLevel, Rank))
124 
125     g = sns.lmplot(x="Score", y="Rank",hue="BindLevel", data=df2, palette="Set1")
126     plt.show()
127 
128     # covariance matrix
129     #covMatrix = df.cov()
130     #sn.heatmap(covMatrix, annot=True, fmt='g')
131     #plt.show()
132 
133     #print(dataset)
134     # Selecting columns
135     #dataset = df[['BindLevel', 'Gl', 'Gp', 'Ip', 'Mixcr']]
136 
137     #k_neihgbours(dataset)
138 
139     #agglomerativeClustering()
140 
141 
142 
143 if __name__ == '__main__':
144     import sys
145     import warnings
146 
147     if not sys.warnoptions:
148         warnings.simplefilter("ignore")
149     main()
```

Code Listing 6.3: Complete code of Python

Description: Here is a complete listing of the code implemented in python. This code can also be found on GitHub under neoantigen-pipeline URL.

6.3.4 DESeq2 listing

DESeq2 listing **Date: 03/09/2020**

```
1
2  Install packages from cran
3
4  ```{r}
5
6  #install.packages("dplyr")
7  #install.packages("ggplot2")
8
9  ```
10
11 Load packages
12
13 DESeq2 = functions for differential analysis
14 biomaRt = gene annotation from BIOMART (geneID -> genename)
15 rtracklayer = gene annotation from GTF
16 GenomicFeature = data objects for GTF information
17
18 ggplot2 = plotting (PCA)
19 dplyr = data wraggling
20 stringr = string manipulations
21
22 ```{r}
23
24 library(DESeq2)
25 #library(biomaRt)
26 library("biomaRt")
27
28 library(ggplot2)
29 library(dplyr)
30
31 ```
32
33 Open the first of the featureCount files and read-in gene lengths for the FPKM
       calculation
34
35 "Column Length always contains one single value which is the total number of non-
       overlapping bases included in a meta-feature (or a feature), regardless of
       counting at meta-feature level or feature level. When counting RNA-seq reads to
```

```
    genes, the Length column typically contains the total number of non-
    overlapping bases in exons belonging to the same gene for each gene."
36
37 ```{r}
38
39 ## read in the file
40
41 firstFileData <- read.table(paste("../input/", targets$sampleID[1], ".markdup.
    featurecount", sep=""), sep="\t", header=T, comment.char="#")
42
43 ## select the first (Geneid) and second-to-last (Length) columns
44
45 geneLengths <- dplyr::select(firstFileData, Geneid, Length)
46
47 write.table(geneLengths$Geneid, file = "../resources/mygenes.txt",quote = F, row.
    names = F, col.names = F)
48
49 ```
50
51 Get gene annotation for all the genes in the featureCounts files
52
53 ```{r}
54 colnames(annotationData)
55
56 ```
57
58 Create a data structure called rawData containing the read in featureCounts for all
    samples
59
60 ```{r}
61
62 ## create an empty data.frame with a single columns of Gene IDs
63
64 rawData <- data.frame(Geneid=geneLengths$Geneid)
65
66 # loop over each filename in the first column in targets
67
68 for (sampleID in levels(targets[,1])) {
69
70  print (sampleID)
71
```

```
  # read the file

  fileContents <- read.table(paste("../input/", sampleID, ".markdup.featurecount",
      sep=""), sep="\t", header=TRUE, comment.char="#")

  # rename the last (7th column)
  colnames(fileContents)[7] <- "rawCounts"


  # gene the columns Geneid and rawCounts

  geneRawCounts <- dplyr::select(fileContents, Geneid, rawCounts)


  # add this to the rawData data.frame

  rawData <- merge(rawData, geneRawCounts, by="Geneid")

  # rename the rawCounts column to reflect the name of the sample
  # this means renaming the last column

  colnames(rawData)[length(colnames(rawData))] <- sampleID
}

```

Add row names to the rawData data.frame, and remove the Geneid column
to leave a data.frame of numbers only. This is the required input format for DESeq2

```{r}

rownames(rawData) <- rawData$Geneid
rawData$Geneid <- NULL

```

Create a DeSEQ2 design matrix

```{r}

exptDesign = data.frame(
```

```
  row.names = colnames(rawData),
  condition = targets$treatment
)

```

Create a DeSEQ2 experimental object

```{r}

exptObject <- DESeqDataSetFromMatrix(
  countData = rawData,
  colData = exptDesign,
  design = ~ condition
)

```

Run the analysis

```{r}

analysisObject = DESeq(exptObject)

```

Get the raw and normalised data back out of the object

```{r}

rawCounts <- as.data.frame(counts(analysisObject, normalized=FALSE))

# add the name "raw" to each of the column names

colnames(rawCounts) <- paste("raw", colnames(rawCounts), sep=".")

normalisedCounts <- as.data.frame(counts(analysisObject, normalized=TRUE))

# add the name "norm" to each of the column names

colnames(normalisedCounts) <- paste("norm", colnames(normalisedCounts), sep=".
```

```
      ")
153
154 ```
155
156 ```{r}
157
158 # define a function to calculate FPKM values from (i) vector of raw counts; (ii) vector of
      gene lengths
159
160 # apply this function to each column of the rawCounts data.frame
161
162 orderedGeneLengths <- geneLengths[match(rownames(rawCounts), geneLengths$
      Geneid),]$Length
163 fpkmNormalisedCounts <- sapply(rawCounts, function(x) countToFpkm(x,
      orderedGeneLengths))
164 rownames(fpkmNormalisedCounts) <- rownames(rawCounts)
165
166 # add the name "fpkm.norm" to each of the column names
167
168 colnames(fpkmNormalisedCounts) <- paste("fpkm.norm", colnames(
      fpkmNormalisedCounts), sep=".")
169
170 ```
171
172
173 # apply this function to each column of the rawCounts data.frame
174
175 orderedGeneLengths <- geneLengths[match(rownames(rawCounts), geneLengths$
      Geneid),]$Length
176 tpmNormalisedCounts <- sapply(rawCounts, function(x) countToTpm(x,
      orderedGeneLengths))
177 rownames(tpmNormalisedCounts) <- rownames(rawCounts)
178
179 # add the name "tpm.norm" to each of the column names
180
181 colnames(tpmNormalisedCounts) <- paste("tpm.norm", colnames(
      tpmNormalisedCounts), sep=".")
182
183 ```
184
185 Tumour versus normal comparison analysis
```

```
186
187 ```{r}
188
189 conditionOne <- "Tumour5"
190 conditionTwo <- "Normal1"
191
192 deData <- as.data.frame(results(analysisObject, contrast=c("condition",
        conditionOne, conditionTwo), pAdjustMethod="BH"))
193
194 ## sort the data by the pvalue column (don't sort by adjusted pvalue, because this
        contains NAs)
195
196 sortedFinalData <- finalData[order(finalData$pvalue), ]
197
198 ## get a vector significant genes for the heatmap
199
200 significantGenes <- as.vector(unlist(subset(sortedFinalData, padj <= 0.01, select=c("
        ensembl_gene_id"))))
201
202 ```
203
204 Tumour versus normal comparison analysis
205
206 ```{r}
207
208 conditionOne <- "Tumour5"
209 conditionTwo <- "Normal2"
210
211 deData <- as.data.frame(results(analysisObject, contrast=c("condition",
        conditionOne, conditionTwo), pAdjustMethod="BH"))
212
213 ## sort the data by the pvalue column (don't sort by adjusted pvalue, because this
        contains NAs)
214
215 sortedFinalData <- finalData[order(finalData$pvalue), ]
216
217 ## get a vector significant genes for the heatmap
218
219 significantGenes <- as.vector(unlist(subset(sortedFinalData, padj <= 0.01, select=c("
        ensembl_gene_id"))))
220
```

```
221 ```
222
223 Tumour versus normal comparison analysis
224
225 ```{r}
226
227 conditionOne <- "Tumour5"
228 conditionTwo <- "Normal3"
229
230 deData <- as.data.frame(results(analysisObject, contrast=c("condition",
        conditionOne, conditionTwo), pAdjustMethod="BH"))
231
232
233 ## sort the data by the pvalue column (don't sort by adjusted pvalue, because this
        contains NAs)
234
235 ## get a vector significant genes for the heatmap
236
237 significantGenes <- as.vector(unlist(subset(sortedFinalData, padj <= 0.01, select=c("
        ensembl_gene_id"))))
238
239 ```
240
241 Tumour versus normal comparison analysis
242
243 ```{r}
244
245 conditionOne <- "Tumour5"
246 conditionTwo <- "Normal4"
247
248 deData <- as.data.frame(results(analysisObject, contrast=c("condition",
        conditionOne, conditionTwo), pAdjustMethod="BH"))
249
250 ## sort the data by the pvalue column (don't sort by adjusted pvalue, because this
        contains NAs)
251
252 sortedFinalData <- finalData[order(finalData$pvalue), ]
253
254 ## get a vector significant genes for the heatmap
255
256 significantGenes <- as.vector(unlist(subset(sortedFinalData, padj <= 0.01, select=c("
```

```
    ensembl_gene_id"))))
257
258 ```
259
260 Tumour versus normal comparison analysis
261
262 ```{r}
263
264 conditionOne <- "Tumour5"
265 conditionTwo <- "Normal5"
266
267 deData <- as.data.frame(results(analysisObject, contrast=c("condition",
    conditionOne, conditionTwo), pAdjustMethod="BH"))
268
269 ## sort the data by the pvalue column (don't sort by adjusted pvalue, because this
    contains NAs)
270
271 sortedFinalData <- finalData[order(finalData$pvalue), ]
272
273 ## get a vector significant genes for the heatmap
274
275 significantGenes <- as.vector(unlist(subset(sortedFinalData, padj <= 0.01, select=c("
    ensembl_gene_id"))))
276
277 ```
```

Code Listing 6.4: Complete Deseq2 code of the project.
Including all file input and output.

Description:

During the last part of this analysis a differential analysis was performed on 5 patients with cancer and tissue and five patients with tumour tissue. During this analysis, p-values and fold2 change values were extracted. A filter was applied for all p-values under 0.05(%). Thus, setting a threshold of 0.05. In addition, there was a threshold of 1.75 absolute log2 fold change but this did not generate any output and for that reason it was not included in the analysis.

Leave One Out Cross-Validation

Leave One Out Cross-Validation Implementation **Date: 15/08/2020**

```
# A function to calculate leaveOneOut cross validation


def leaveOneOut(X, y, model):

    loo = LeaveOneOut()

    predicted = []
    actual = []

    for train_index, test_index in loo.split(X):
        predicted.append(rf.predict(X_test)[0])  # returns 1D array
        actual.append(y_test[0])

    predicted = np.array(predicted)
    actual = np.array(actual)

    # Print separating ine
    print("." * 80)

    print("Predictions leaveOneOut -----")

    print({"RMSE": rmse(predicted, actual)})

    print({"Rho": stats.pearsonr(predicted, actual)[0]})

    print({"R^2": stats.linregress(predicted, actual)[2] ** 2}, "\n")

    print({"R^2": metrics.r2_score(actual, predicted)}, "\n")
```

Code Listing 6.5: This part of the code describes all workings in Python to calculate leave one out Cross-Validation

Description:

This method has the attributes and methods to train the model using the leave one out cross validation method. There are not a lot of annotations as the code can be self explanatory given the variable values.

K-Fold Cross-Validation

K-Fold Cross-Validation Implementation **Date: 16/08/2020**

```
# A function to fit a model for k-fold cross validation

def k_fold(X, y, model):

    # lm = linear_model.LinearRegression()
    lm = model
    model = lm.fit(X, y)

    predictions = cross_val_predict(model, X, y, cv=20)

    # r2 = metrics.r2_score(y, predictions)

    # Print separating line
    print("." * 80)

    print("Predictions K-Fold -----")

    print("RMSE on test data: ", rmse(predictions, y))

    print("Pearsons rho: ", stats.pearsonr(predictions, y))
    # slope, intercept, r_value, p_value, std_err = stats.linregress(predictions, y)
    # print("R^2: ", r_value ** 2, "\n")

    print({"R^2": metrics.r2_score(y, predictions)}, "\n")
```

Code Listing 6.6: This part of the code describes all workings in Python to calculate K-Fold Cross-Validation

Description:

This method contains parameters such as the regression model on which we will train our model. The specified fold are set to 20. For the output there is information on the RMSE, spearman's rank correlation coefficient and the r square.

6.4 Appendix D: Results

6.4.1 Neoantigen Repertoire

Neoantigen repertoire in prostate cancer **Date: 12/09/2020**

Formation of Neoantigens

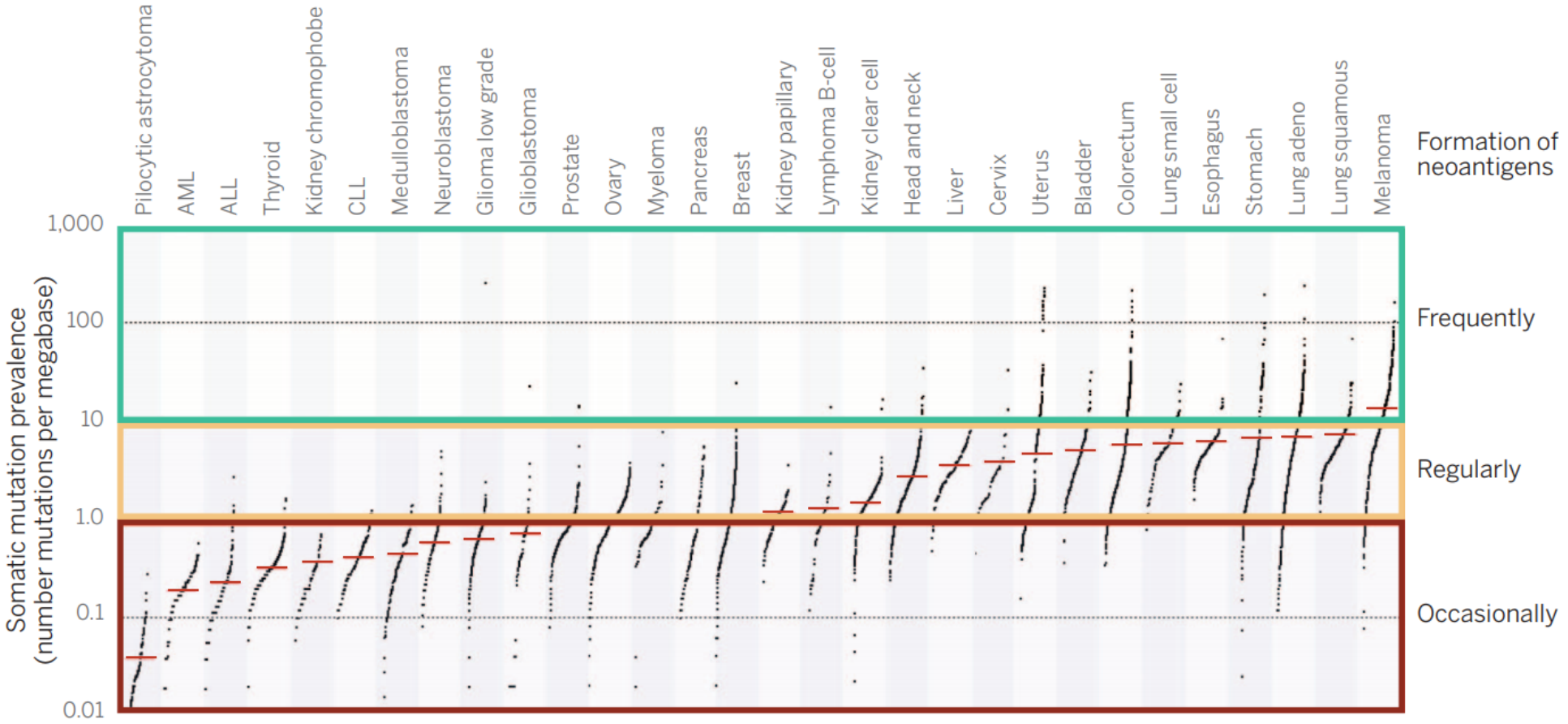


Figure 6.2: A plot of Formation of Neoantigens being the most expressed in melanoma. Source: https://science.sciencemag.org/content/sci/348/6230/69.full.pdf

Description:

A Figure (6.2) depicting the neoantigen repertoire of different cancer types. Neoantigen repertoire in prostate cancer in specific shows that the likelihood of neoantigen formation is roughly one mutation per megabase which means these mutations occur regularly.