Universiteit Leiden
The Netherlands

Opleiding Informatica

Prediction of leprosy susceptibility
from an RNA-seq population study

Aart Rosmolen and Emma R. van Eerde

Supervisors:
Fons Verbeek, Annemieke Geluk & Matheus Rogério Almeida

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

24/12/2024

Abstract

Leprosy is still a prevalent disease within third-world countries, with the addition of the impact the disease can have on an individual. To make the disease less impactful, it is crucial to establish a diagnosis as soon as possible. With differential gene expression analysis and machine learning, it will be possible to identify genes that could be used to indicate the progression of the disease at an early stage. The data used in this study originates from a field trial, in which whole blood samples were collected from households of an individual that is diagnosed with leprosy.

This data will be used in this study to find genes that are differentially expressed, comparing progressors and non-progressors at the timepoint where neither showed any clinical symptoms. With these differentially expressed genes, a subgroup of genes will be identified with the use of machine learning models that can accurately predict the early development of leprosy.

Multiple differential gene expression (DGE) analysis methods (DESeq2, EdgeR, LimmaVoom) are used on RNA-Seq data between progressors without symptoms and household contacts. Subsequently, the predictive performance of four different machine learning models (Random Forest (RF), Support Vector Machine (SVM), Linear Logistic Regression (LLR), and K-Nearest-Neighbours (kNN)) along with the performance of two different feature selection methods (Chi-Squared and Recursive Feature Elimination) was compared to find the optimal predictor to identify genes leprosy susceptibility genes from the DGE results.

After performing DGE analysis, 108 genes were found that were persistent within each of the three libraries. Of these 108 genes, 62 are coding genes and 46 genes are noncoding genes.

With the use of different feature selection methods, subsets of 3-20 genes are created from each of the subgroups (all genes, coding, non-coding genes). These selector genes of 3-20 genes are then used to train the different machine learning models.

With the AUC score from these models, the conclusion was made that the Recursive Feature Elimination method is outperforming the Chi-Squared Feature Elimination. In addition, the AUC score informs us that the use of non-coding genes is improving the AUC score of the model compared to the AUC score from the coding genes. The Support Vector Machine and Random Forest appear to achieve better performance than the Linear Logistic Regression and k-Nearest-Neighbour models. Combining the results of the SVM and RF model with an AUC score greater than 96% gives a combined list of 20 genes of which 10 overlap with the 20 genes found by the evaluation of the DGE analysis using Random Forest. The SVM model with a selector of the 10 genes (TMEM238, SNHG8, C6orf48, HCG4P11, SNHG5, RN7SL3, RPL13AP5, TPGS1, RP11-533E19.7, DPM3) have an AUC score of 98,3%.

In future work the found genes could be analysed further with RT-qPCR and a machine learning model to confirm these findings.

Aart Rosmolen - s2548526
Emma van Eerde - s2800020

Contents

1 Introduction

In this section, we introduce leprosy and the importance of finding prospective biomarkers that can indicate future progression of the disease, after which key developments in the field of leprosy research will be addressed.
The aim of this thesis will be outlined, followed by an overview of the research methodology employed, and the research questions guiding this study will be presented and elaborated. Subsection 1.4 covers the definitions of some key concepts.
The final sections of the introduction will offer an overview of the thesis structure and the disclaimer of this thesis.

1.1 General background

Leprosy is an infectious disease caused by the pathogen *Mycobacterium leprae* and is the oldest infection specific to humans [HS14]. It affects skin and peripheral nerves, and when not diagnosed at an early stage, leprosy can cause irreversible disabilities and tissue damage such as limb deformities and blindness [MLZ24]. These clinical symptoms and their incubation period vary greatly between individuals and can range from asymptomatic infections to severe cases or even death.
The significance of this research lies in the fact that leprosy remains a serious health threat in certain regions, with a stable number of new cases annually. Although the disease has previously been declared to be eliminated, new cases of leprosy patients still pose a threat in low- to middle-income countries [ASTS20]. This continuation of transmission is mainly due to the reliance on clinical symptoms to detect leprosy, leading to delays in treatment. By identifying a prospective biomarker using transcriptomic analysis of whole blood samples from high-risk individuals, this study aims to facilitate early diagnosis, prevent disease progression and transmission, and reduce disability.

Leprosy originates as a human-specific disease, but after incidental transmission to armadillos and persistence of infection in the armadillo population, leprosy became a zoonotic disease [HS14]. Currently, the main known hosts are humans, armadillos, red squirrels, and chimpanzees. Transmission of leprosy between humans occurs through droplets from the mouth or nose during close and prolonged contact with an infected individual. Furthermore, transmission could occur through different routes, such as contact with blood or ingestion of breast milk [HSC24]. Exposure to *M. leprae* does not necessarily lead to infection, since only 5% of the exposed individuals can be infected of which only 20% will develop leprosy [MLZ24]. In addition, the *M. leprae* genome has maintained remarkable conservation for thousands of years. These facts make leprosy a predisposed disease and effectively eliminate the influence of pathogen variation on leprosy susceptibility, emphasizing the importance of host response and innate immunity to leprosy infection [MLZ24]. After infection, the incubation time can range from several months to cases where leprosy progresses only after twenty years [UdB+22] [Orga]. The high variability in incubation time between individuals and the fact that only 1% of exposed individuals develop leprosy complicates tracking the bacterium transmission route to ensure accurate and timely treatment.

The World Health Organization (WHO) has classified leprosy into two distinct categories - paucibacillary (PB) and multibacillary (MB) - to be able to provide accurate treatment based on the Bacilli Index (BI) and the number of skin lesions [Orgb]. These two different types of leprosy caused

by *Mycobacterium leprae* are based on the host's immune response to leprosy infection. Individuals with PB leprosy exhibit one to five skin lesions, with no detectable bacilli in a slit-skin smear. MB cases are characterized by more than five skin lesions or nerve involvement, or the detection of bacilli in a slit-skin smear.
A more detailed classification system [RJ62] separates leprosy cases into tuberculoid leprosy (TT) and lepromatous leprosy (LL) with three intermediate forms; borderline tuberculoid (BT), mid-borderline (BB), and borderline lepromatous (BL) leprosy. TT leprosy has a strong cell-mediated immune response to *M. leprae* and few lesions and a low number of bacilli [vHG21], while patients with LL leprosy have multiple skin lesions and foamy macrophages containing a high number of bacilli [MLZ24]. Intermediate forms of leprosy have an unstable immune reaction to the bacterial infection.
Leprosy treatment initially consisted of dapsone monotherapy, but this eventually led to dapsone-resistant *M. leprae* and the adverse reaction called Dapsone Hypersensitivity Syndrome (DHS) [MLZ24]. This caused the WHO to advise multidrug therapy (MDT) consisting of rifampicin, dapsone, and clofazimine. The WHO standardized this treatment worldwide in 2018 [HSC24].
In 2010, 22% of new leprosy patients were diagnosed with Grade 2 Disabilities (G2D) [HSC24], meaning that their symptoms had already reached advanced stages, highlighting the need for early detection and prophylactic treatment.

This study aims to look into an individuals likelihood or predisposition, also known as susceptibility, to develop leprosy. This predisposition is determined by a combination of genetic and immunological factors that influence how effectively the body can recognize, respond to, and control the infection. This thesis will focus on gene expression and regulatory mechanisms that can be derived from RNA expression data.

Differences in immune system function, such as weakened cell-mediated immunity, can affect the ability to control the bacterium after exposure. Van Hooij et. al. [vHG21] hypothesizes a constant battle between host and *M. leprae*, where individuals who do not progress to leprosy disease after prolonged, close contact win from the bacterium, while progressors do not. Identifying the immune response that explains this difference is important for identifying leprosy at an early stage.
Other research has shown that FLG gene mutations leading to loss-of-function could disrupt the skin's ability to protect against pathogens, potentially contributing to the onset of leprosy [WZZ+20]. In addition, studies have identified genes such as RAB32, LRRK2, and SLC29A3 to be associated with phagosome maturation [MLZ24] - a process essential for the destruction of bacteria and activation of innate immune responses. The recognition of pathogen-associated molecular patterns by pattern recognition receptors also plays an important role in the initiation of innate immunity in macrophages. NOD2, RIPK2, and LACC1 have been identified to be involved in this process of pattern recognition. Furthermore, in infected macrophages, S100A12 has been identified to be essential for the survival of *M. leprae* [vHTCV+20]. Additionally, during infection, a decrease in mitochondrial activity in Schwann cells, axons, and primary monocytes has been seen, which compromises the mitochondrial ability to immune modulation [OMM+21].
In short, based on previous research it can be concluded that inflammatory responses and recognition of the pathogen play an important role in leprosy progression, and genes associated with pathways connected to innate immunity, epithelial cells, mitochondrial activity, and peripheral nerves could be interesting findings of this thesis.

As mentioned before, early identification of individuals at risk of developing leprosy is crucial for preventing permanent physical damage, however, in many cases, leprosy diagnosis is established only after symptoms have occurred and irreversible damage has been done due to the lack of understanding of the transmission of *Mycobacterium leprae* and other factors. This lack of knowledge can be overcome by finding ways of establishing predictive biomarkers that indicate future development of the leprosy pathogen before any clinical symptoms arise by looking at differential gene expression levels of different groups. Predictive biomarkers will help to perform targeted, prophylactic treatment of leprosy disease and prevent further *Mycobacterium leprae* transmission.

Previous research found a 13-gene prospective signature capable of predicting leprosy progression with an area under the curve (AUC) of 95.2% and reduced this biomarker after validating the 13-gene signature to a 4-gene signature called RISK4LEP which showed an AUC of 86.4% [TC21]. Van Hooij et. al. [vvR+19] also aimed at identifying an early leprosy development biomarker, but used proteomics instead of transcriptomics. Other leprosy research found a signature of 16 genes with, however, a lower accuracy than the 4-gene signature found in the leprosy research of M. Tio-Coma [ZPNS+16].

This research extends the previous research of Tio Coma by using additional data containing noncoding RNA. In addition, multiple differential gene expression methods, machine learning models, and feature selectors are compared to find the best predictive model.
The gene expression methods DESeq2, EdgeR, and Limma+Voom (LimmaVoom) are implemented and used to identify important genes. In the previous study, only EdgeR was applied to identify these genes.
A Support Vector Machine (SVM), Random Forest (RF), Linear Logistic Regression (LLR), and k-Nearest-Neighbor (kNN) model are chosen as machine learning models to apply to the data assembled by the DGE analysis to find the best predictive model for leprosy susceptibility.
SVMs and Random Forest are complex models, while Linear Logistic Regression and kNN are more simplistic models. It is unknown if a more complex or simplistic algorithm is more suitable for the given dataset, which is why both types of algorithms are included.

DESeq2 [LHA14] is an R package that is created in order to identify differentially expressed genes from RNA sequencing data between conditions. This technique is widely used in research and performs well with a small sample size. Using a negative binomial distribution, it models the RNA count data to account for overdispersion (RNA counts that exceed the mean). Additionally, it normalizes the RNA count data with the median of ratios method to correct the difference in sequencing depth across samples. After these steps, it estimates gene-wise dispersion, followed by statistical testing to determine genes that show significant change in expression. Another R packages used to identify differentially expressed genes from RNA sequencing data is the R package EdgeR [RMS09]. This package accounts for the overdispersion in a way similar to that of the package DESeq2, thus using a negative binomial distribution. However, the normalization technique differs. The method Trimmed Mean of M is applied to the count data to correct the difference in sequencing depth across the samples. At last, the differentially expressed genes are determined using Empirical Bayes moderation and statistical tests. The final R package used in this research is the limma [RPW+15] package with the voom function. The LimmaVoom method uses the strength of linear modelling from microarray

data analysis to fit the RNA count so that it fits the assumptions of linear modelling. Thereafter, it uses the voom transformation to estimate the model parameters that allow LimmaVoom to perform statistical tests. With these statistical tests, it can determine which genes are differentially expressed.

Support Vector machines are powerful algorithms known for their ability to perform well on high-dimensional, small datasets [Por19] and are widely used in applications like image recognition and bioinformatics, where precision and high performance are critical. They work by finding the optimal hyperplane that best separates data into classes, maximizing the margin between data points of different categories. The kernel parameter of SVMs allows for handling both linear and non-linear data.
The Random Forest machine learning algorithm builds multiple decision trees during training and combines their outputs, which makes it an ensemble method, to improve accuracy and prevent overfitting [Rig17]. Random subsets of the data and features are used to train the trees to promote diversity in the forest and minimize bias. In case of a classification task predictions are made using majority voting.
Regression analyses investigate the relationship between the dependent and response variable, and in case of linear regression, it expects a linear relationship. Linear Logistic Regression estimates the probability of the dependent and response variable having a specific label. In contrast to linear regression, which requires continuous independent variables, logistic regression requires categorical labels [WFI07]. It maps the probability of the independent and dependent variables to have a specific label into a probability range between 0 and 1. It is a statistical method that can be used for classification tasks and is often used due to its simplicity and effectiveness in binary outcomes [Sto11].
k-Nearest-Neighbor is a non-parametric, simple, and supervised machine learning algorithm that does not explicitly learn a model, but makes predictions based on the labels of the closest data points in the multidimensional space [TDVS19]. The output is based on the majority class of the neighbours of a data point.

1.2 Methodology

This study propagates upon RNA-sequencing data obtained during an observational population study in Bangladesh where blood samples were taken from household contacts of individuals infected with leprosy [TC21]. Using the RNA-sequencing data, three different differentially gene expression analysis methods (DGE analysis methods) will be used in R to identify genes that are differently expressed. These R packages are DESeq2, EdgeR and LimmaVoom.
Subsequently, different machine learning models will be implemented and tested in Python to examine which model is best to predict leprosy progression with various combinations of genes and find out the relative contributions of the individual genes. The combinations of genes will be created from the differentially expressed genes discovered by the DGE analysis methods. The implemented machine learning models are a Support Vector Machine (SVM), Random Forest, Linear Logistic Regression, and k-Nearest-Neighbor (kNN).
To identify a predictive biomarker, a comparison will be made between the RNA-Seq results from household contacts of leprosy progressors with those from household contacts who eventually developed leprosy, but before any clinical symptoms were evident. This comparison will be conducted using the machine learning models outlined above. Since the data consists of myriad genes, several

variations of features will be made using feature selectors, and the efficacies of these different selectors will be compared to establish which feature selector is best to use considering the dataset. The best predictive model(s) will be used to determine which genes contribute most in predicting leprosy diagnoses and could potentially form a risk biomarker for leprosy progression.

1.3 Research questions

The study's objective is to research if a predictive biomarker for leprosy progression can be determined. Therefore, the research question is:

- Which genes can be considered biomarkers for leprosy?

This research question aims to identify transcriptomic factors that may contribute to the progression of leprosy in household contacts of leprosy progressors before the appearance of any symptoms.
To answer this research question, sub-parts have been formulated.
Due to the importance of the differentially expressed genes found by the DGE analysis methods, the use of the best suitable DGE analysis method is crucial for the outcome of this research. This prompts the next subquestion:

- Which differential gene expression analysis method performs best in predicting leprosy progression?

Given the dataset and its characteristics, such as size and dimensionality, it is important to determine which machine learning model is best applicable in finding the best predictions of leprosy progression and which technique is best to use to reduce the number of genes the ML models will use. This leads to the following subquestions:

- Which feature selector technique performs best in predicting leprosy progression from RNAseq data?
- Which machine learning model performs best in predicting leprosy progression from RNAseq data?

The following subquestions aim at comparing the findings of this thesis to previous research and ascertain the effect of including noncoding genes.

- Are differentially expressed genes found at preclinical timepoints similar to identified leprosy biomarkers in previous research?
- What is the effect of the inclusion of noncoding genes in the prediction of leprosy progression?

1.4 Definitions

This section clarifies and provides definitions of key terms that are abbreviated in some parts of this thesis. If applicable, the abbreviations are noted in between brackets.

1. **T1** is the timestamp at which blood samples of household contacts had been taken when they did not show any clinical symptoms of leprosy.

2. **T2** is the timestamp at which blood samples of household contacts that showed clinical symptoms of leprosy, a.k.a. progressors, had been taken.

3. **Household contacts (HHC)** are individuals who do not show any clinical symptoms of leprosy and live in the same or directly neighboring household as individuals who are positively diagnosed with leprosy [TC21]. Since HHC do not progress to leprosy disease, only blood samples at T1 were taken.

4. **The First group (First)** is defined as the group of individuals who progressed to leprosy disease between 4-61 months after T1. Blood samples of the First group were taken at T1 - when the individuals did not show any clinical symptoms yet.

5. **The Second group (Second)** consists of the same individuals as the First group and are thus defined as progressors. Their blood samples were taken at T2, at which leprosy had been diagnosed and did not yet receive treatment.

6. **All genes (ALL)** is defined as the total group of genes, i.e. including both coding and non-coding genes.

7. **Noncoding genes (NC)** is defined as the group of genes with one of the following biotypes: unitary pseudogene, unprocessed pseudogene, processed pseudogene, transcribed unprocessed pseudogene, antisense, transcribed unitary pseudogene, polymorphic pseudogene, lincRNA, sense intronic, transcribed processed pseudogene, sense overlapping, IG V pseudogene, pseudogene, 3prime overlapping ncRNA, bidirectional promoter lncRNA, snRNA, miRNA, misc RNA, snoRNA, rRNA, Mt tRNA, Mt rRNA, TR V pseudogene, TR J pseudogene, IG C pseudogene, IG J pseudogene, scRNA, scaRNA, vaultRNA, sRNA, macro lncRNA, non coding, IG pseudogene, processed transcript, ribozyme.

8. **Coding genes (C or CODING)** is defined as the group of genes with one of the following biotypes: IG D gene, protein coding, TR V gene, IG V gene, IG C gene, IG J gene, TR J gene, TR C gene, TR D gene, TEC.

9. **Path A** is the workflow that is concerned with the difference in DGE analysis. The subanalyses of this workflow are showcased in Table 2

10. **Path B** is the workflow that is concerned with the difference in DGE analysis. The subanalyses of this workflow are showcased in Table 3

1.5 Overview

The next chapters are structured as follows: Section 2 addresses the materials used, including software, programming packages, and machine learning models, and ends with outlining the experimental procedures. Section 3 covers the results of these experiments. These results are further discussed in section 4. At last, the research questions are answered in section 5, together with possible future work.

1.6 Disclaimer

This thesis was written collaboratively by Aart Rosmolen (AR) and Emma van Eerde (EE) as part of the requirements for obtaining the bachelor degree in Bioinformatics. AR focused on all aspects of Path A, which is mainly concerned with the theory and testing of different Differential Gene Expression analysis methods. EE focused on all aspects of Path B, i.e., testing and elaborating on different machine learning techniques that could form a good predictor for assessing the a for leprosy. The characteristics of both paths are discussed in detail in subsection 2.6.1.
As the subjects are so closely related, it was decided to combine the reporting in one thesis. In this manner the subject can be comprehensively discussed. The efforts in the writing are equally divided.

The introduction (Chapter 1) was primarily written by EE, with some contributions and additions from AR. The chapter on materials and experimental procedures (Chapter 2) is divided into multiple subsections, of which AR wrote subsections 2.2, 2.3, 2.4.2, and 2.6.2. EE wrote the introduction on feature selectors in subsection 2.4, subsection 2.4.1, 2.5 and 2.6.3. The contributions of AR and EE to subsection 2.1 and 2.6.1 are equal.
Subsections 3.1 and 3.3 were written by AR, and subsection 3.2 by EE. Subsection 3.4 was written by both AR and EE, with AR taking the lead and contributing the most.
Subsection 4.1 and 4.3 were written by AR. Subsection 4.2 was written by EE. Subsection 4.4 and the conclusion 5 were written by both AR and EE, with an equal contribution between the two.

2 Materials and experimental procedures

This section will cover the dataset, software, differential gene expression analysis methods, feature selectors, machine learning models, and experimental procedures. Subsection 2.1 will cover the data description about a dataset gathered outside this thesis, which will be used for differential gene expression analysis. The subsection 2.2 will expand on the software used for this thesis. Subsections 2.3.2, 2.4, and 2.5 detail the materials and the theory behind it. To conclude, Subsection 2.6 outlines the workflow and procedures that combine the theory and materials discussed in subsections 2.1-2.5.

2.1 Dataset

The data used in this study originates from an observational population study in Bangladesh performed by Tio Coma et. al. [TC21]. Within this study, blood samples were collected from a total of 5352 household contacts. From these individuals, the first sample was collected between April 2013 and April 2018. This cohort consisted of 5033 individuals at T1 (definitions in 1.4). Of these individuals, 85 were diagnosed with leprosy and a second whole blood sample was taken at T2.
For the RNA analysis of these groups, the samples were divided into three groups, namely HHC, First, and Second (defined in 1.4). A total of 120 samples were divided in 40 samples in HHC, 40 samples in First, and 40 samples in Second. The blood samples of the First and Second group were from the same individuals, but at a different timepoint (T1 and T2, respectively). The individuals from HHC and the individuals of the First and Second group were coupled, meaning that they were from the same household.
Of each individual, the type of leprosy (MB or PB), year of birth, gender, and number of months between T1 and T2 is known. 66.25% of the samples are female and 33.75% are male.
RNA sequencing of the 120 samples was performed with the Open-Source BIOWDL RNA-Seq pipeline v2.0.0 [taLUMC19], developed at the Leiden Medical Centre. Additional steps to prepare the data for differential gene expression analysis was performed by Tió-Coma et. al. and described in section 2.5 of their paper [TC21]. This resulted in three datasets:

- The RNA counts dataset. This dataset contained a table with columns (120) indicating the samples and the rows (58032) indicating the genes. Additionally, a cell contains the number of occurrences that a RNA transcript of a gene was found in a certain blood sample.
- The genes dataset contained a total of 58032 genes and information on their Ensemble biotype, chromosome location and gene name.
- The sample dataset. This dataset contained a total of 120 household contacts. It describes the connection to other household contacts, their disease state and leprosy type, duration till disease diagnosis and some individual details such as year of birth and gender.

Since each sample is associated with 58032 features and there are only 120 samples, the dataset suffers from the curse of dimensionality [Ven18]. As the number of features increases, the feature space becomes larger and the occurrence of data points in this space becomes relatively sparser. This is why the dataset is called to be high-dimensional. The computational complications, that come with this, will be further discussed in section 2.6.3.

2.2 Software and tools

This section explains the setup used for both of the analyses. In Section 2.2.1, the different programming languages are listed. In addition, the packages used for each analyses are listed separately in order to allow reproduction of a single analysis. In Section 2.2.2, the details of the server is listed. This contains all the important parts of the system, such as the Operating System (OS), Central Processing Unit (CPU) and the amount of Random-Access Memory (RAM).

2.2.1 Software

To perform differential gene expression analysis, R.4.4.2 [R C24] has been used together with its R packages DESeq2 (v1.46.0) [LHA14], EdgeR (v4.4.1) [RMS09], and limma (v3.62.1) [RPW+15]. All of these packages are managed by the Bioconductor project, with its release version 3.20 [Bio], which is an organization that aims to create open software that is focussed on analysing biological data. In specific, the three packages mentioned before are created and managed in order to discover differentially expressed genes from RNA count data between one or multiple conditions. Additionally, the R package ggplot2 (3.5.1) [Wic16] has been used to showcase all of the different results generated from the differential gene expression analysis and partly from the results created by the machine learning methods.
The other analysis performed by this study is the analysis based on the results from different machine learning models. For this analysis, the programming language Python3.12 [VRD09] has been used together with the library Scikit-learn (v1.6) [PVG+11]. From the results, generated by this library ggplot2 and R, in addition to Matplotlib (v3.2.1) [Hun07] and Python, have been used to illustrate the results.

2.2.2 Hardware

Both of the different analyses, DGE analysis and machine learning model analysis, were run on a server with the following specifications:

1. **Operating System**: CentOS 7, Linux with the use of Elrepo 7
2. **CPU**: Intel Xeon E5-2630v3 16 cores @ 2.40GHz (32 threads)
3. **RAM**: 1.5 TB

Due to Scikit-learn's [PVG+11] inability to use a Graphics Processing Unit (GPU), it is not needed to GPU within the system to run the software. However, it is recommend to make use of a powerful CPU to speed up the process of machine learning, although any CPU should be able to run the software.
Additionally, it is recommended to at least use 8 GB of RAM in order to keep the code from stuttering. Finally, the disk space should at least have 5 GB of free disk space in order to save all the data, images and GIFs on the device.

2.3 Differential Gene Expression analysis

For the experiments, three different differential gene expression analysis methods have been used, i.e. DESeq2, EdgeR, and LimmaVoom. There are many differences between these three methods, such as the method used to normalize the RNA count data and the algorithms behind the analysis methods.
DGE analysis enables the prediction of differentially expressed genes from RNA-sequencing data or microarray data. These methods compare the differences between two or more different groups in order to establish a ratio of the difference in expression between conditions (Fold Change) and the p-value, indicating the statistical significance of the found genes. If the gene has a certain p-value below the maximum p-value, which is established by the researchers, it will be considered differentially expressed. In addition, it can be specified further by setting a certain value for the Fold Change. This will result in a further selection of differentially expressed genes to showcase genes that are differentially expressed only by a certain percentage between groups.

2.3.1 Normalization

Due to the differences between gene expression within individuals and even samples of the same individuals, it is important to apply normalization. Normalization is used to reduce these differences and allow for a more accurate analysis after this step. The method allows to minimize the variability, such as differences in library size or sequencing depth. This is usually performed with the use of adjusting the RNA counts to a representative sample. However, not all of these normalization methods are the same. Within this paper, two different normalization methods are applied to the RNA counts using these DGE analysis methods.
One of them is the normalization method called Median of Ratios or Relative Log Expression (RLE) used by DESeq2 [DRA+13a]. This normalization method makes use of a pseudo-reference sample to normalize individual samples with. The pseudo-reference sample, containing individual genes, is calculated by taking the mean of the RNA counts for all samples of one gene, doing so for every gene. With the use of pseudo-reference sample, the adjusted RNA counts per gene are calculated by dividing the RNA counts of a gene by the RNA counts of the pseudo-reference sample of a gene. This is performed for each gene and sample until every RNA count has been adjusted. From this, the normalization factors per sample are calculated by taking the median of the adjusted RNA counts of each gene for a sample. The final step is to perform the normalization of the counts by dividing the original RNA count of a gene by the normalization factor.
The second normalization method is known as Trimmed Mean of M-values (TMM) normalization [RO10] used by both EdgeR and LimmaVoom. Before deciding on a reference sample, this normalization method performs per-normalization of the RNA counts by dividing the RNA count per gene by the total number of reads of a gene of all samples. After the pre-normalization step, a reference sample is picked. By default, this reference sample consists of the gene of whom their RNA counts are closest to the upper quartile of the mean of the RNA count of the gene of all samples. Before being able to calculate the relative scaling factors for each gene based on the reference sample, the set of genes is trimmed. Hereby, genes that show gene-wise fold change and/or show high or low absolute expression levels are filtered out. The parameters for these changes are by default the top and bottom 30% and 5%, respectively, which were not adjusted for the experiments. This filtering reduces the chance that the normalization factor is effected by differentially expressed genes. For each gene in the subset, the relative expression is calculated by dividing the RNA counts of a

gene of that sample divided by the RNA counts of the reference sample. After this, the relative expression of genes is added together and divided by the number of genes within the subset in order to calculate the normalization factor. With this, the normalized RNA counts are calculated by dividing the RNA counts by the normalization factor while accounting for the relative size and library size.

2.3.2 Differential gene expression analysis methods

All DGE algorithms are created in order to improve for a specific use case that the other DGE method might lack or does not excel at. Therefore, an outline of the steps performed by each DGE method is given in the section below. In addition, the differences and overlap between the three methods during each step are displayed in Table 1.

Table 1: **This table summarizes the main steps of the three differential gene expression analysis methods (DESeq2, EdgeR and LimmaVoom** This table contains the overlap and differences between the three DGE analysis methods; DESeq2, EdgeR and LimmaVoom. These steps are further explained in 2.3.2.

Step	**DESeq2**	**EdgeR**	**LimmaVoom**
Normalization	Median of Ratio's or RLE	Trimmed Mean of M values	Trimmed Mean of M values
Distribution assumption	Negative binomial distribution	Negative binomial distribution	Log-normal distribution
Focus of dispersion estimation	Focus on individual dispersion estimate, followed by global variance adjustment	Focus on dispersion estimate of the mean of a gene of all samples	Focus on individual dispersion estimate, followed by global variance adjustment
Shrinkage	Shrinkage with Empirical Bayes	Shrinkage with Empirical Bayes	Shrinkage with voom transformation
Statistical test	Likelihood ratio test	Likelihood ratio test	t-test

After normalization of the data, performed with their respective method, the input for all the DGE analysis methods is expected to be conform to a standard. In specific, the input has to be a matrix, with rows corresponding to genomic features such as genes, transcripts, exons or tags. The columns should represent a specific biological sample. This sample is often a sample ID, which is coupled to another matrix containing all the known information of a sample, such as disease condition.
However, a lot of the individual data points (RNA counts) of a gene might differ more from the mean of a gene causing more variance within the individual RNA counts of that gene. This relationship between the mean and the variance is called dispersion. In an ideal situation, the mean and variance are equal, thus showcasing an idealized Poisson distribution. But, in real RNA-Seq count data, the variance is often larger than the mean. This allows for a Negative Binomial distribution to estimate the dispersion better with. Due to this, both DESeq2 and EdgeR make use of the assumption that the data is distributed like a Negative Binomial distribution. On the other hand, LimmaVoom makes use of a normal distribution scaled to a log-normal distribution allowing for linear modelling to be performed on the distribution. This distribution is commonly used when data is influenced by numerous smaller factors (in this case genes).
In order to account for the dispersion, dispersion estimations are done for all the different DGE methods. However, the focus of this estimation differs between two of the DGE methods. DESeq2 focuses on the maximum individual dispersion estimate, which is used to estimate the dispersion for each gene individually in order to more accurately capture the individual variability. Subsequently, the gene-wise estimations are scaled to a global variance estimation in order to stabilize the estimations and reduce the high and low estimations of individual genes. In addition, LimmaVoom is performing dispersion estimation similar to DESeq2. However, it uses its own limma R package to estimate the dispersions that cause some mathematical formulas to differ due to the linear

modelling. However, the main idea is the same. On the other hand, EdgeR estimates the dispersions with the focus on a common mean of gene expression from all samples of one gene. This allows for the reduction of noise, especially in genes with a low RNA count.
After the dispersion estimations have been calculated, each DGE method shrinks the individual RNA counts to stabilize and calculate the fold change estimations between conditions. DESeq2 and EdgeR make use of Empirical Bayes with the use of the dispersion estimations to shrink them. In addition, they both model or fit the data to the assumed distribution, in this case Negative Binomial distribution. Alternatively, LimmaVoom uses the voom function from the limma package in order to calculate precision weights and use them to shrink the RNA counts to the log-normal distribution trend showcasing the mean-variance relation.
After calculating these fold change estimations, a statistical test is performed in order to indicate the significance of this fold change estimation. Within this research, there were multiple conditions to identify a change between. Due to this, the DESeq2 and EdgeR methods use the same statistical test, namely the likelihood ratio test. Because of the linear modelling of LimmaVoom, these methods are not possible to use. Therefore, LimmaVoom makes use of the t-test in order to showcase the significance of the fold change estimations.

2.4 Feature selectors

The best algorithm for feature selection depends on the machine learning model used, the data, and context. Feature selection is a crucial step in the prediction process since we are dealing with a high-dimensional, small dataset. High-dimensionality can compromise the machine learning model's learning speed and capability of generalization on unseen data [Ven18], which is why uninformative features should be removed. It has been found that many features are often even irrelevant or redundant [BCSMAB13] and thus do not have to be included when training a model.
There are two categories in feature selection: individual evaluation and subset evaluation. Individual evaluation ranks the importance of the features individually, while subset evaluation assesses the importance of subsets of features. Feature selectors can also be divided into three different types; wrapper methods, filter methods, and embedded methods. Filter methods define the importance of a feature independent of the other features and the machine learning model, indicating that it is an individual evaluation method. It determines the relationship between the feature and the dependent variable using statistical metrics. Wrapper methods assess the importance of subgroups by evaluating the performance of the machine learning model, which means the feature selection is dependent on the model used. Embedded methods have feature selection integrated into the training of the model making it model-dependent.
Recursive Feature Elimination and Chi-Squared use subset evaluation and individual evaluation respectively. They frequently use feature selectors with a wide range of possible applications, such as in Support Vector Machines and Random Forest. Both will be tested to see which selection method is more suitable for the given dataset to find the optimal model-feature selector combination. The different theoretical backgrounds will be briefly discussed in subsection 2.4.1 and subsection 2.4.2.

2.4.1 Chi-Squared

Chi-Squared (χ^2) is a feature selector that uses a statistical approach in selecting the features with highest importance in making predictions about the given data. It calculates the Chi-Squared

statistic on a feature and its label to see if there is a significant relationship between the two. It applies the Chi-Squared test independently of all other features on the dependent variable and thus does not take into account any feature interactions that might exist.
If two features have a similar relation to the target, one of them is redundant even though both features may have a high Chi-Squared statistic. However, Chi-Squared cannot recognize this redundancy since it evaluates the features separately and will keep both features.

2.4.2 Recursive Feature Elimination

Another feature elimination method is the Recursive Feature Elimination [GWBV02]. The main idea of this elimination method is to minimize the negative effect on the cost function when removing a feature. This change in cost function based on removing a specific feature, created by LeCun et. al. [LDS89], which is shown in Formula 1.

$$\Delta J(i) = (1/2) * \frac{\delta^2 J}{\delta w_i^2} (\Delta w_i)^2 \tag{1}$$

The first step in the algorithm is the training of the classifier with the goal of optimizing for the weights of an individual feature with respect to the cost function J. Subsequently, the ranking criterion of all features will be calculated and sorted from largest to smallest. For the ranking criterion both $(\Delta J(i)$ or $(w_i)^2$ can be used. However, this algorithm makes use of only $(w_i)^2$ as the ranking criterion. The final step is to remove the feature with the smallest ranking criterion. These steps are repeated until the specified number of features are left, after which the method returns the features it has selected to be the best.
In contrast to Chi-Squared, RFE is capable of detecting redundancy and can capture feature interactions that individual evaluation methods could miss. However, this also makes the method computationally more demanding.

2.5 Machine learning models

This study concerns a binary classification problem that needs to be optimized. The aim is to classify the instances in the dataset into two classes; a positive or a negative leprosy diagnosis. Multiple machine learning models can be applied to binary classification problems, but some are more suitable than others based on the specific data characteristics, nature of the problem, computational feasibility, and desired interpretability. In this study, the dataset is small and high-dimensional, requiring a robust model to prevent overfitting. Another characteristic that should be considered is the desire for an interpretable model, since it concerns a healthcare problem that asks for transparency and interoperability of the decisions made to come to the results it finds. Considering these requirements, a few models are eligible to be included in the model comparison and selection based on their theoretical characteristics, strengths, and weaknesses.

A Deep Neural Network is not suitable due to the size of the dataset and the low interpretability since it is a black-box model. Simple models sometimes outperform complex models, so a k-Nearest-Neighbour and Linear Logistic Regression model will be tested. Random Forest is very versatile, has a strong performance in a wide range of classification problems, and is also robust because it is an ensemble method that combines multiple decision trees. Random Forest models are also

fit to handle high-dimensional data, preventing overfitting. kNN models have the advantage of simplicity and fast prediction because there is no training phase involved. In addition, Support Vector Machines are known for there ability to perform well on high-dimensional data. These considerations lead to the selection of a Support Vector Machine, Random Forest, Linear Logistic Regression, and k-Nearest-Neighbor model for further testing in finding the best model to apply to the dataset in question. The theoretical characteristics and important limitations of these models will be detailed in the following sections.

2.5.1 Support Vector Machine

A Support Vector Machine is a machine learning algorithm that aims to solve classification and regression problems by identifying patterns in the data. In classification problems, SVMs split the data into a specified number of classes based on patterns learned during the training phase. In addition, it has a powerful generalization capability and is therefore robust against noise in the data, preventing overfitting.
The SVM attempts to define the optimal hyperplane in the feature space that maximizes the margins between the two classes in binary classification. Two parallel hyperplanes are defined that signify the margins. The points closest to this margin are defined as the support vectors. The support vectors are used to find this optimal hyperplane and are often only a small part of the original dataset [GFLC24].
In cases where the data is not linearly separable, slack variables ξ_i are introduced [Ban19], which allow for softer margins and some misclassification. An approach to regulate this is with the use of a hyperparameter. The hyperparameter C regulates the balance between overfitting and regularization by balancing misclassification and the maximization of the margin[CGLRML20]. A lower value of C increases the margin and assigns a lower penalty to slack variables, allowing for more misclassification. This results in better generalization. Choosing a higher value of C penalizes slack variables more, decreases the margin, and is thus more prone to overfitting.
Hyperparameter *gamma* is especially influential when using non-linear kernels such as radial basis function (RBF) and polynomial kernels. It controls the influence of individual data points in determining the decision boundary. Lower *gamma* means a more spread-out influence over the feature space, while a higher *gamma* value ensures a more localized influence of data points. The former has a higher risk of underfitting and missing important nuances and patterns in the data. A higher *gamma* value tends to overfit and creates a narrower margin, but is better capable of catching important nuances.
The tolerance hyperparameter *tol* is concerned with the stopping criteria of the algorithm. A higher value of *tol* terminates the optimization process earlier than a lower value. The optimization process is terminated when the change in the objective function is smaller than the chosen threshold set by *tol*. With a higher value of *tol* the algorithm reaches convergence later than with a lower value of *tol* which also means shorter training time. A lower value of *tol* is preferred when having a small dataset and high accuracy is preferred over a short computation time.
The linear kernel often works well with high-dimensional data [MR16] and is computationally the most efficient choice between the possible kernels that can be used. However, complex data may need a more complex model to capture non-linear relationships. A polynomial kernel can capture non-linear relationships but can also become very complex and computationally expensive in a high-dimensional space. A high degree of the polynomial function and a large number of features

can result in more susceptibility to overfitting due to excessive model complexity. The RBF kernel might be more suitable when dealing with complex data but also numerous features. In contrast to the polynomial kernel the RBF kernel does not perform a transformation of the feature space which makes it less computationally expensive. It focuses on the similarities in the data instead of feature interactions, leading to less susceptibility to overfitting.

The feature space of a support vector machine is an n-dimensional space representing all possible feature values of the dataset. In the feature space, the decision boundary of the SVM is defined as given by Equation 2.

$$w \cdot x + b = 0 \tag{2}$$

The equation consists of variables where w is the weight factor and signifies the importance of the feature, b stands for the bias, i.e., the offset of the hyperplane from the origin, and x denotes a vector with n components that represent the features of a data point:

$$x = [x_1, x_2, ..., x_n] \tag{3}$$

Each instance in the data used as input to the SVM can be represented as a tuple with an input vector x_i and corresponding label y_i.
The weight w can be used for interpretability as it denotes the feature importance, i.e., the contribution of a feature in defining the optimal hyperplane.

2.5.2 Random Forest

Random Forest is an ensemble method that combines the results from different generated decision trees into a single prediction using majority voting. The different decision trees should be as diverse as possible which is achieved by creating vectors that consist of different subsets of the samples and features to optimize the performance of the model [Bre01]. The vectors are selected independently from the previously generated vectors. Random Forest is used for both regression and classification tasks and is a supervised algorithm.

The selection of random subsets is done by sampling with replacement, also called 'bootstrap aggregation' or 'bagging' [Rig17]. The samples that are not included in the training data by sampling with replacement make up approximately $1/e$ of all samples and form the dataset that can be used to test the performance of a tree. This is repeated n times, specified by the *n_iter* hyperparameter. The *bootstrap* parameter defines if the random sampling with replacement technique is used to create subsets of the data (*bootstrap*=True) or if the whole dataset is used for training (*bootstrap*=False). The hyperparameter *n_estimators* specifies how many decision trees are generated and trained, and *max_depth* sets the maximum depth of the decision trees until a prediction is made. *min_samples_split* denotes the number of samples a node should have after which the node splits. The fewer splits a tree has, the better the generalization capability the decision tree has. *max_features* specifies how many features should be considered when splitting a node. The *min_samples_leaf* parameter sets the minimum number of samples a leaf node should have.

2.5.3 Linear Logistic Regression

Regression is a statistical method that makes predictions based on probabilities. It estimates the likelihood of a data point to have a certain label and is a widely used technique for making predictions on the relationship between explanatory variables and a response variable [HJLS13]. There are different kinds of regression models suitable for different purposes and datasets. Linear Regression, for example, is a type of regression where the relationship observed is linear. In our case, the response variable is the diagnosis feature, which can be equal to zero (HHC) or one (progressor or T1). Logistic Regression is, depending on the application, also called Linear Logistic Regression since it takes linear data as input to the regression function.

$$P(Y = 1|X) = \frac{1}{1 + e^{-z}} \tag{4}$$

The formula of equation 4 shows the mathematical formulation of the linear logistic regression that models the probability of the dependent and independent variables to have label 1 in case of binary classification. Here, z denotes the weights of the independent variables[Sla]:

$$z = \beta_0 + \beta_{1x} = w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_\mathrm{p} x_\mathrm{p} \tag{5}$$

Equation 5 represents the linear part of the Linear Logistic Regression model, which is used as input to the sigmoid (logistic) function shown in Equation 4. β_0 and β are learned parameters that should maximize the probability of the data points in the training dataset to fall into the correct category specified by the labels in the training set[FXMY14].

Just like with the Support Vector Machine, there are several hyperparameters that optimize the performance of the Regression model by avoiding overfitting and ensuring regularization of the model. Parameter C has the same functionality as with the SVM, and denotes the regularization strength of the model. *penalty* defines the type of regularization, which can be Ridge regularization (*L1*), Lasso regularization (*l2*), or a combination of both (*elasticnet*). Hyperparameter *solver* determines the algorithm used for the optimization. *l1_ratio* sets the ratio of *l1* and *l2* in case *elasticnet* is used as regularization method.

2.5.4 k-Nearest-Neighbour

The k-Nearest-Neighbour algorithm is a machine learning model that makes its predictions based on the label of its neighbouring data point(s), where k indicates the number of neighbours that are considered in this decision-making. It is a supervised learning technique and is non-parametric, meaning that it does not implicitly learn from the data, so there is no training phase included to make the predictions, and it only uses its k closest neighbours to determine the label of a certain data point.
The number of neighbours that should be considered is set using the hyperparameter *n_neighbours*, where an odd number of k prevents a fifty-fifty result of the majority voting.
The distance between data points can be determined by different methods. The default metric for this is *minkowski*, which is a variation on the Euclidean distance. Other possible metric values are the standard Euclidean distance, Manhattan distance, and cosine similarity.

The *algorithm* parameter of kNN specifies the method used to efficiently find the closest neighbouring data points. The choice of the best *algorithm* value depends on the size of the dataset and number of features. The efficiency is optimized by a tree-based structure that is used to organize the dataset. The leaf size of the tree is set by the *leaf_size* parameter which specifies the number of data points are stored in a leaf. A smaller leaf size creates more splits in the tree and thus a deeper tree which requires more memory space. A larger leaf size results in fewer splits but also more data points in a leaf which is computationally more demanding.
It is also possible to assign variable weights to different data points with the hyperparameter *weights*, where the weight depends on the distance, and closer neighbours have a stronger influence on the determination of the category.

2.6 Experimental procedures

This section covers the experiments conducted to answer the research questions of this thesis. We begin with the experimental setup, detailing the data sources and preprocessing steps. Next the methods of the differential gene expression are explored. The subsequent steps include hyperparameter optimization and feature selection, aimed at enhancing model performance. The section will be closed by naming the methods used to ensure the models' generalization capabilities and robustness. Each section contributes to .. a comprehensive approach for deriving meaningful insights and reliable predictions.

2.6.1 Experimental setup

The setup shown in Figure 1 displays the workflows used to come to a final list of genes that could potentially form a biomarker. The whole process consists of two parts that are represented as two pathways in the flowchart to indicate the different analysis methods. The two analysis methods are separated in time but also have some overlapping procedures, where Path A is the start of the whole workflow and is concerned with the Differential Gene Expression analysis. Path B comes in after the differentially expressed genes are identified and focuses on applying different machine learning techniques to the results of the DGE methods. Where the procedures of the different paths are different from each other, the distinction is made using the colors green (Path A) and pink (path B). At the end of the process, the results of the different analysis paths join to come to a combined list of genes.

The workflow starts with the differential gene expression analysis where three different methods are applied: DESeq2, EdgeR, and LimmaVoom. Each will result in a different number of differentially expressed genes based on their specific methods. After the DGE analysis there are five different datasets created based on the DGE analysis results that will be used to make predictions on a risk of leprosy susceptibility. The five datasets are DESeq2, EdgeR, LimmaVoom, Union, and Intersection. The Union dataset consists of the union of the genes identified by at least one of the methods; DESeq2, EdgeR, and LimmaVoom datasets, and the Intersection dataset consists of the intersection of genes identified by all of the three methods; the DESeq2, EdgeR, and LimmaVoom datasets. The exclusion of samples due to being outliers occurs after this with the use of MDS plots.
All five datasets are divided into three datasets: ALL, NC, and CODING, which totals in fifteen datasets. Before the data is subjected to machine learning models, two steps must be taken, namely

hyperparameter optimization and feature selection. The hyperparameter optimization process will be further discussed in subsection 2.6.3.
After hyperparameter optimization, size 3-20 feature selections are made using χ^2 or RFE (based on the experiment). The aim of using subsets of genes varying from 3 to 20 is to test model performance for different group sizes of genes and see if there is a point where adding extra features does not yield better predictions. Due to practicality a biomarker of fewer genes is preferable, so the gene group size starts at 3 and builds up to 20. The model predicts the labels of the test set for each of these 18 selectors using leave-one-out cross-validation (LOOCV) to help ensure robustness in the models' predictions. The predictions are evaluated using accuracy, precision, recall, F1, FPR, TPR, AUC score, and specificity as performance metrics. These values are averaged from the 78 predictions of the LOOCV. The leave-one-out cross-validation is performed for five iterations and again averaged to create additional robustness. The predictions of the support vector machine are not iterated five times, since SVMs are deterministic models, thus repeating the LOOCV would result in the same outcomes.

Path A then uses the fifteen datasets to assess the performance using Random Forest. Path B only uses the Intersection dataset and applies four different machine learning models; a Support Vector Machine, Random Forest, Linear Logistic Regression, and k-Nearest-Neighbour model.
In case of Path A there are fifteen datasets on which a Random Forest model using Chi-Squared and RFE is applied, resulting to $15 \cdot 1 \cdot 2 = 30$ analyses which are listed in Table 2. In case of Path B, $3 \cdot 4 \cdot 2 - 3 = 21$ analyses are performed which are shown in Table 3. Each analysis consists of 18 subanalyses due to the different subsets of genes used as features to the models. This leads to $30 \cdot 18 = 540$ results for Path A and $21 \cdot 18 = 378$ results for Path B.
To narrow the results down, all subanalyses with an AUC score lower than 96.0% will be left out of the further analyses. Both Paths then determine which genes occur in most subanalyses and make a selection of the 20 most occurring genes. The intersection of those two results will be taken to select the genes that are found in both analyses.

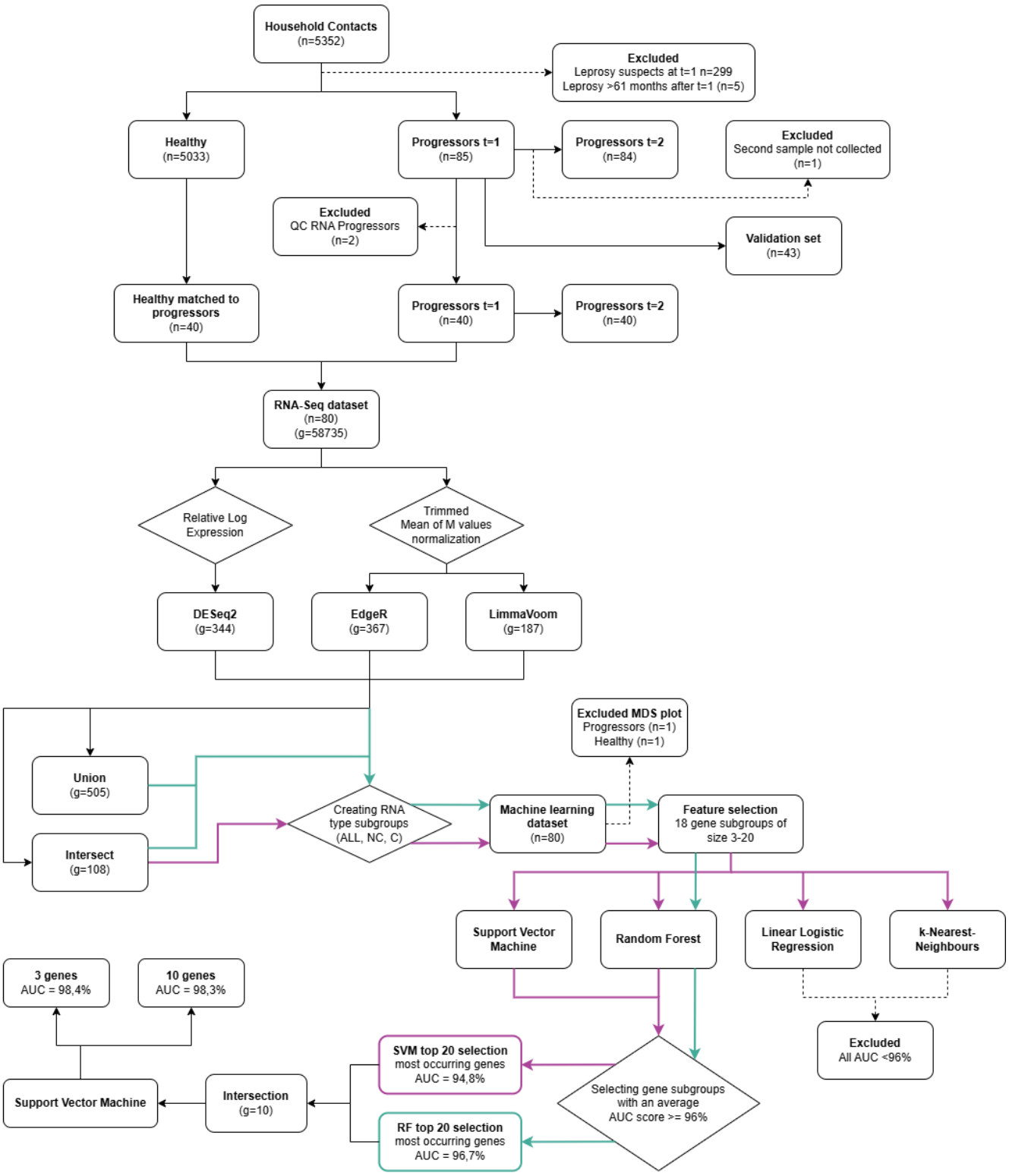


Figure 1: **Flowchart showcasing the different experiments** In this figure, the workflow is shown which is divided into three parts contains 3 parts: the RNA-Seq dataset, DGE analysis and Machine learning. The first part spans from the start until the RNA-Seq dataset. The second part covers the process from the RNA-Seq dataset to the machine learning dataset. Finally, the third part includes all steps involving the machine learning dataset. The difference in subanalyses is depicted with green (Path A) and pink (Path B) and these are shown in Table 2 & 3.

Table 2: **Sub-analyses performed by Path A** The thirty analyses are all the possible combinations of type RNA and DGE method. Each experiment is given an analysis ID by which the experiments can be referenced. The column "Type of RNA" indicates which type of genes were used for the prediction, All the analyses are trained and tested with the Random Forest model. See 1.4 for the definitions of the abbreviations.

Analysis ID	**Type of RNA**	**DGE Method**	**Feature Selector**
1a	ALL	Intersection	RFE
2a	ALL	Intersection	χ^2
3a	ALL	Union	RFE
4a	ALL	Union	χ^2
5a	ALL	DESeq2	RFE
6a	ALL	DESeq2	χ^2
7a	ALL	EdgeR	RFE
8a	ALL	EdgeR	χ^2
9a	ALL	LimmaVoom	RFE
10a	ALL	LimmaVoom	χ^2
11a	NC	Intersection	RFE
12a	NC	Intersection	χ^2
13a	NC	Union	RFE
14a	NC	Union	χ^2
15a	NC	DESeq2	RFE
16a	NC	DESeq2	χ^2
17a	NC	EdgeR	RFE
18a	NC	EdgeR	χ^2
19a	NC	LimmaVoom	RFE
20a	NC	LimmaVoom	χ^2
21a	C	Intersection	RFE
22a	C	Intersection	χ^2
23a	C	Union	RFE
24a	C	Union	χ^2
25a	C	DESeq2	RFE
26a	C	DESeq2	χ^2
27a	C	EdgeR	RFE
28a	C	EdgeR	χ^2
29a	C	LimmaVoom	RFE
30a	C	LimmaVoom	χ^2

Table 3: **Sub-analyses performed by Path B** The twenty-one analyses are all the possible combinations of type of RNA, model, and feature selector. Each analysis is given an analysis ID by which the experiments can be referenced. The column "type of RNA" indicates which type of genes were used for the prediction, "Model" is one of the four machine learning models, and "Feature selector" is one of the two feature selectors. See 1.4 for the definitions of the abbreviations.

Analysis ID	Type of RNA	Model	Feature selector
1b	ALL	SVM	χ^2
2b	ALL	SVM	RFE
3b	NC	SVM	χ^2
4b	NC	SVM	RFE
5b	C	SVM	χ^2
6b	C	SVM	RFE
7b	ALL	RF	χ^2
8b	ALL	RF	RFE
9b	NC	RF	χ^2
10b	NC	RF	RFE
11b	C	RF	χ^2
12b	C	RF	RFE
13b	ALL	LLR	χ^2
14b	ALL	LLR	RFE
15b	NC	LLR	χ^2
16b	NC	LLR	RFE
17b	C	LLR	χ^2
18b	C	LLR	RFE
19b	ALL	kNN	χ^2
20b	NC	kNN	χ^2
21b	C	kNN	χ^2

RFE is not used as a feature selector for the kNN model, since RFE recursively selects features based on their importance to the model's performance, and kNN does not assign any importance measure or coefficients to the features. For each other model, both χ^2 and RFE are tested in one of the experiments. Each of these combinations is tested on all genes, non-coding genes, and coding genes.

2.6.2 Differential gene expression parameters

In order to correct the calculated p-value to the adjusted p-value, which accounts for performing multiple tests, the Benjamini & Hochberg procedure [BH95] was applied. This method is used by default for the package DESeq2. Therefore, the adjustment parameter within EdgeR and LimmaVoom was adjusted to make use of this procedure. The method allows to correct for genes that are calculated to be significant as end result, but are not differentially expressed. With this adjusted p-value, a threshold is set to only label the genes, that have a adjusted p-value below or equal to 0.05, as significant genes. In addition, the threshold of 1.5 ($\log_2(1.5) \approx 0.585$) has been set for the fold change of a gene for the condition First - HHC. This threshold allows the genes that differ 50% in gene expression between conditions to be labelled as differentially expressed. By combining these two thresholds, it only selects the significant differentially expressed genes with a fold change of 50%.

2.6.3 Hyperparameter optimization

Hyperparameter optimization (HPO) has been performed on all four machine learnin models to improve their performance. Given the high dimensionality of the dataset and the small number of

samples, an exhaustive search of the hyperparameter space of all hyperparameters is computationally expensive. Exhaustive search is especially unsuitable with complex models like a Support Vector Machine and Random Forest. Moreover, the optimal hyperparameters depend on variables such as the model, the dataset size, the number of features, and the feature values. These variables differ among all 378 analysis settings and therefore, in theory, these analyses require separate hyperparameter optimization. Therefore the hyperparameter optimization methodology of each model is carefully thought out based on the models' characteristics and theory and will be detailed in this subsection.

Due to the complexity of the Support Vector Machine [Slb] and the problem to be solved, hyperparameter optimization of the SVM is computationally very demanding. Setting *max_iter* to 1000 or even 10000 does not result in convergence to an optimal solution, and thus *max_iter* is set to -1 to let the optimization process run until convergence is attained. Due to this computationally expensive process, a combination of theory and selective use of randomized search will be used to determine the optimal hyperparameter settings of the analyses with the SVM.
The complexity of the SVM makes it infeasible to apply a grid search approach for the optimization, so a randomized search approach is chosen for efficient search space exploration. To make sure the SVM generalizes well to unseen instances the hyperparameters *C*, *gamma*, and *tol* are included in the randomized search with as search spaces a uniform distribution with the following intervals: *C*: (0.1, 10), *gamma*: (0.001, 0.1), *tol*: (0.001, 0.0005). The randomized search performs 10-fold cross-validation to validate the performance of each randomly sampled set of hyperparameters and takes 100 samples of hyperparameter combinations. The random state parameter is set to 42.
The data is completely balanced, which means the default setting of the *class_weight* parameter will be used. Shrinking is set to *True* to speed up the optimization process by ignoring some training samples that are unlikely to be support vectors. The probability parameter is set to *False* since its additional value of giving insight into the class membership probabilities does not outweigh its increased computation and training time when comparing the models' performances.
After hyperparameter optimization of *C*, *gamma*, and *tol*, the parameters of the SVM are set to the following values: *max_iter*=-1, *shrinking*=True, *C*=3.8, *gamma*=0.2, *break_ties*=True, *probability*=False, *tol*=0.0004.

The randomized search is applied to the same dataset but with two different numbers of features and compared to the results and performance of using the same hyperparameter settings. It can be concluded that there are no differences in results and performance between using the same hyperparameter setting or specific settings for different group sizes. Therefore all analyses and all different group sizes of the SVM will be trained with the same hyperparameter settings.

Similar to the HPO of the Support Vector Machine, a randomized search approach is used to optimize the hyperparameters of the Random Forest model. Like with the SVM, the main challenge is to prevent overfitting and make sure the model generalizes well on unseen data. Taking this into account, the focus of the HPO will lie on the *n_estimators*, *max_features*, *max_depth*, *min_samples_split*, *min_samples_leaf*, and *bootstrap* hyperparameters. The search spaces of these hyperparameters are set to the following values: *n_estimators*: a list of ten equally spaced numbers between 50 and 200, *max_features*: [auto, sqrt, log2], *max_depth*: None, and a list of eleven values between 10 and 110, *min_samples_split*: [2, 5, 10], *min_samples_leaf*: [1, 2, 4], *bootstrap*: [True, False]. The number of

iterations of the randomized search is set to 100 with a 5-fold cross-validation and random state 42. The optimal hyperparameters of each analysis found after hyperparameter optimization can be found in [RvE].
The number of decision trees is set to a common default that balances model performance and computational cost. The depth of the tree is set to be relatively deep, which means it can capture complex patterns in the data but might also overfit. The number of samples required to split a node is set to the default value of 2, which is relatively low and might cause overfitting. The number of samples in a leaf is set to 2, which lowers the tendency of the model to overfit by adding regularization to the tree. Setting *max_features* to 'sqrt' means as many features as the square root of the total number of features are considered when deciding on the best split. This allows for randomness and thus diversity between the trees improving the robustness of the model while also preventing the model from overfitting. Bootstrap also adds to the diversity of the trees and therefore increases the robustness of the model.

Linear Logistic Regression is a less complex model that allows for an exhaustive search of the best parameter settings such as a grid search approach. This is a brute-force method that systematically evaluates all possible parameter combinations based on the defined grid with possible combinations. The grid is defined as follows: *C*: [0.01, 0.1, 1, 10, 100], *penalty*: [l1, l2, elasticnet, none], *solver*: [liblinear, saga], *l1_ratio*: [0.3, 0.5, 0.7]. *C*, *penalty*, and *solver* play important roles in the regularization of the model and handling of high dimensionality and are thus included in the HPO. Since the data is completely balanced, the hyperparameter *class_weight* is left at the default value. The final parameter settings of the Linear Logistic Regression model after optimization are stated below.
k-Nearest-Neighbour is a non-parametric model and thus the focus of the HPO of this model will lie on the prevention of overfitting by the right selection of the number of neighbours and finding the optimal distance metric. A small number of neighbours results in a model that is more prone to overfitting but can better capture patterns and finer details. In addition, the right selection of the distance metric and finding the best algorithm is crucial to handling high-dimensional data, since some kNN algorithms perform worse when the number of features increases. The parameter search spaces are defined as follows: *n_neighbours*: [3, 5, 7, 9, 11], *weights*: [uniform, distance], *algorithm*: [auto, ball_tree, kd_tree, brute], *leaf_size*: [20, 30, 40], *p*: [1, 2], *metric*: [Euclidean, Manhattan, Minkowski, cosine]. A grid search approach with 5-fold cross-validation is applied since the models' simplicity allows for an extensive search of the search space. Accuracy is used as the performance metric in the optimization process. The optimal hyperparameters of each analysis found after hyperparameter optimization can be found in [RvE].

3 Results

This chapter is organized to outline the results of the different analyses performed in sequential order. The chapter starts by listing the comparison and findings of the differential gene expression analysis methods in section 3.1, followed by section 3.2 that covers the results of the comparison of the four different machine learning models and Chi-Squared and RFE feature selectors. Section 3.3 looks into the effect of using noncoding genes in predicting a risk for leprosy compared to using only coding genes or combining the two. Finally, section 3.4 closes the chapter by listing the genes discovered based on the results of section 3.1, 3.2, and 3.3.

3.1 Differential gene expression analysis methods

With the results from the DGE analysis of the three DGE methods; DESeq2, EdgeR and LimmaVoom, a summary of the results was generated and shown in Table 4. This table showcases the differences between the number of genes identified by each of three DGE methods. It divides the regulatory state into 3 states; upregulated genes (UP), downregulated genes (DOWN) and genes that are not differentially expressed (STABLE STATE). Another division shown in this table is the division of type of RNA in three subgroups, namely coding genes, noncoding genes and the combination of the coding and noncoding genes (ALL).

Table 4: **Summary of the results obtained from the differential gene expression analyses** This table showcases the regulatory state (up- or downregulated genes or not differentially expressed genes and their type of RNA (CODING, NONCODING, or ALL types) per DGE analysis method (DESeq2, EdgeR, and LimmaVoom).

Regulatory State	Type of RNA	DESeq2	EdgeR	LimmaVoom
UP	CODING	79	94	54
UP	NONCODING	63	89	47
DOWN	CODING	101	90	48
DOWN	NONCODING	101	94	38
UP & DOWN	CODING	180	184	102
UP & DOWN	NONCODING	164	183	85
STABLE STATE	CODING	21002	13415	13497
STABLE STATE	NONCODING	36101	3679	3777
ALL	ALL	57447	17461	17461

Another figure used to showcase the results of the DGE analysis of the three methods is Figure 2. This figure showcases the overlap of the genes identified as differentially expressed (UP & DOWN in Table 4 with all types of RNA. One observation from this venn diagram is that the intersection of genes identified by the three DGE methods, also known as the intersection of the results of the three DGE methods, resulted in 108 genes.

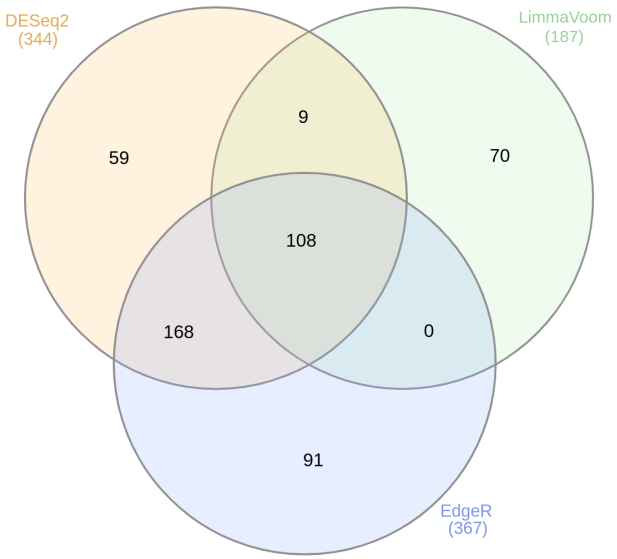


Figure 2: **Venn Diagram of the number of genes discovered by the three differential gene expression analysis methods** This figure showcases the overlap between the genes discovered by the three DGE methods; DESeq2, EdgeR, and LimmaVoom. Therefore, the number in the center indicates the number of genes discovered by all methods (108 genes) and the outer numbers indicate the number of genes discovered by each method individually but not by the other methods (DESeq2: 59, LimmaVoom: 70, EdgeR: 91). The numbers between two methods indicate the genes that they were discovered by both of the methods (DESeq2 & EdgeR: 168, DESeq2 & LimmaVoom: 9 and EdgeR & LimmaVoom: 0). The total number of genes discovered by each DGE method is shown in Table 4, where the sum of UP & DOWN CODING and UP & DOWN NONCODING is taken.

Figure 3 presents the difference in performance of Random Forest models trained with different features. These features are based on 5 different subgroups of the differentially expressed genes, namely:

- **Union**, differentially expressed genes of all the unique genes discovered by the methods DESeq2, EdgeR and LimmaVoom, which totals to 540 genes
- **DESeq2**, differentially expressed genes discovered only by DESeq2, which totals to 344 genes
- **EdgeR**, differentially expressed genes discovered only by EdgeR, which totals to 367 genes
- **LimmaVoom**, differentially expressed genes discovered only by LimmaVoom, which totals to 187 genes
- **Intersection**, differentially expressed genes of the intersection of the genes discovered by the methods DESeq2, EdgeR, and LimmaVoom, which totals to 108 genes

From the Figure 4, it is shown that the Random Forest model trained with the genes from LimmaVoom is performing significantly better (p-value lower than 0.05, namely 0.0034 and 0.0043 respectively) than the Random Forest models trained with the genes from the other two DGE methods; DESeq2 and EdgeR (95.1% versus 93.0% and 93.2% respectively).
In addition to this, the Random Forest models trained with the genes from the Intersection are not

significantly being outperformed by LimmaVoom, while the Random Forest models trained with the genes from the Union are outperformed significantly (95.1% LimmaVoom, 94.3% Intersection and 93.3% Union, with p-values of 0.098 LimmaVoom versus Intersection and 0.014 LimmaVoom versus Union).
A final observation, that is displayed in Figure 4, is that DESeq2 and EdgeR have a similar performance. In order to visualise the similarity better between the two, Table 5 was created. This table showcases the amount of overlap between the features identified and used to train each model of two DGE methods. As shown in the table, there is substantially more overlap between EdgeR and DESeq2 than LimmaVoom compared to the other methods.

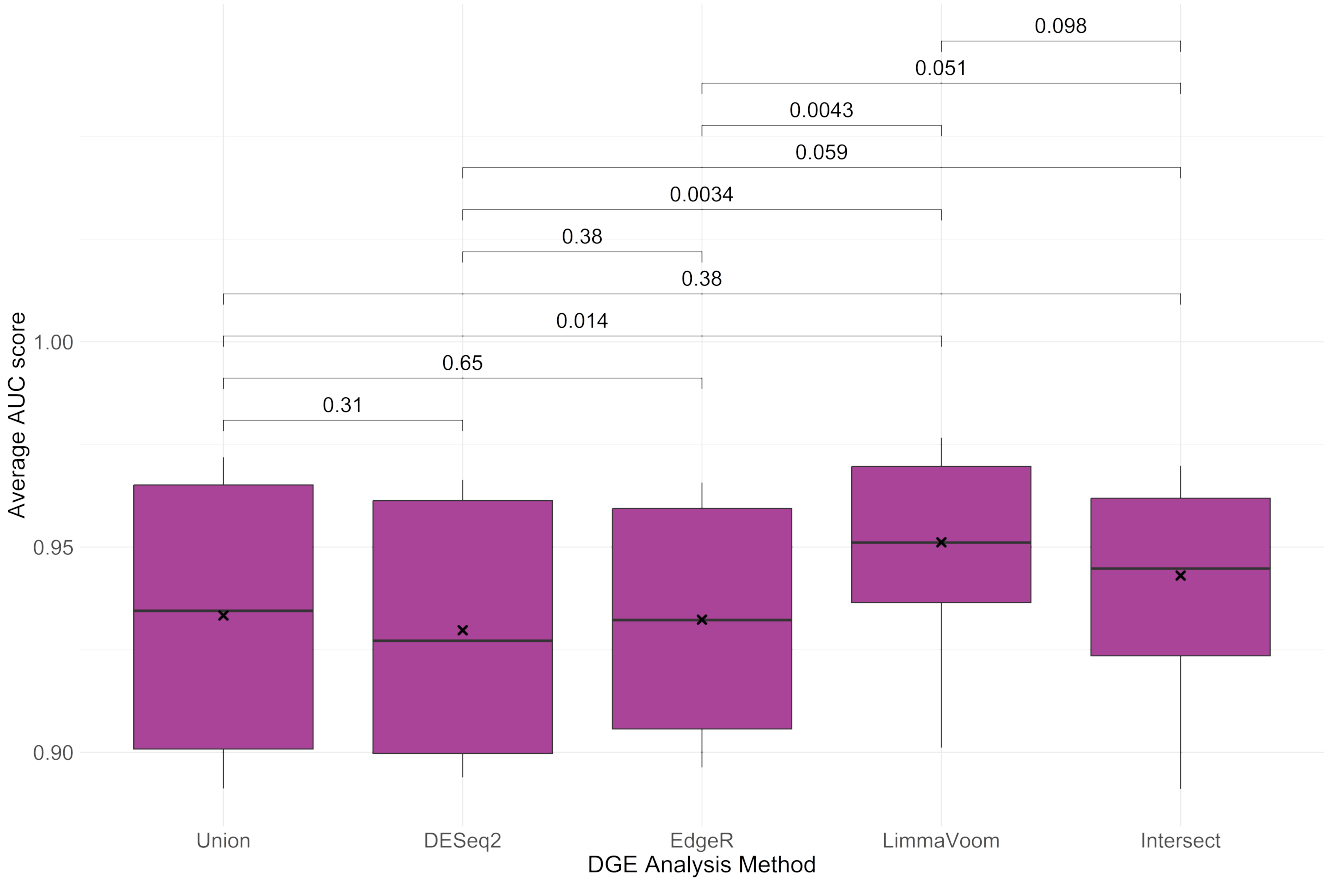


Figure 3: **Difference in performance of a Random Forest model with features from different DGE methods** In this figure the performance of the Random Forest models trained with the features generated from the differentially expressed genes discovered by DESeq2, EdgeR, LimmaVoom, the intersection of the genes of the three DGE methods and the union of the genes of the three DGE methods are displayed. The subanalyses run in this figure are subanalyses 1a-30a from Table 2. The mean performance of each DGE method from left to right is 93.3%, 93.0%, 93.2%, 95.1%, 94.3%. Additionally, the p-values calculated with the Mann-Whitney U test [MW47] are shown above of the boxplots, with a line indicating which DGE methods are compared. The y-axis starts at an AUC score of 0.85 due to scale considerations.

Table 5: **Overlap in features between features generated from different DGE methods (Path A)** In this table, the overlap of features (identified by the DGE methods) of subanalyses 1a-30a is displayed.

DGE Method 1	DGE Method 2	Average overlap
DESeq2	EdgeR	87.1%
DESeq2	LimmaVoom	33.9%
EdgeR	LimmaVoom	32.6%

3.2 Machine learning and feature selectors

A couple of statements can be made based on the results shown in Figure 4, which shows the performance of the different model-feature selector combinations. There is an obvious distinction between the performance of the χ^2 feature selector compared to RFE. When looking at the average performance of the feature selectors over the three datasets (ALL, NC, and CODING), RFE performs significantly better than χ^2 using SVM, RF, and LLR. However, when considering the performance of the feature selectors on the individual datasets (see Figure 11), the Linear Logistic Regression model In addition, when looking at Figure 11a, 11b, and 11c, RFE overall has a higher AUC score in all three datasets. Furthermore, if the analysis is only performed on Random Forest with a dataset of each DGE method, the intersection and union of the DGE methods, as done in Figure 13, a similar result is shown.

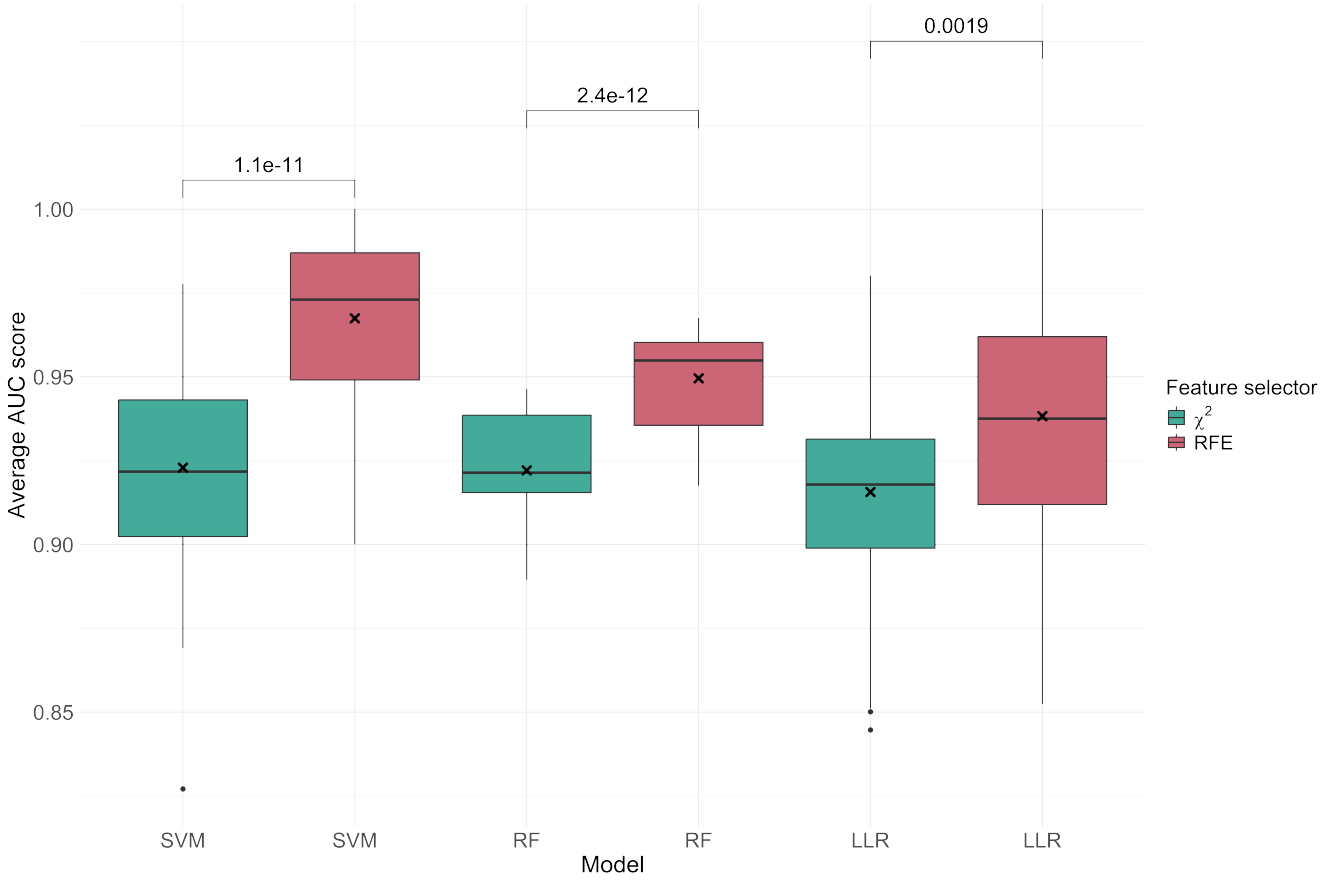


Figure 4: **Feature selector performance per model** The average performance of the three datasets (ALL, NC, CODING) of the Chi-Squared and RFE feature selectors per model is shown in this figure. From left to right, the average AUCs are 92.3%, 96.7%, 92.2%, 95.0%, 91.6%, and 93.8%. The associated p-values of the difference in AUC per model are given at the top of the boxplots.

The highest scoring model-feature selector combination is the SVM using RFE with an AUC of 96.7%. The combination of Random Forest using RFE shows the lowest variance when looking at the three individual datasets in Figure 11. The most substantial difference in performance can be seen within the SVM model, where the difference in AUC score is equal to 4.4%.

The difference in performance is statistically significant for all three models. However, this cannot be said for the difference between Chi-Squared and RFE when looking at the LLR model applied to the 'ALL' and 'NC' datasets in Figure 11a, 11b, and 11c, which have a p-value of 0.24 and 0.12 respectively.
In Figure 11 it can be seen that RF has the lowest variability in performance compared to SVM and LLR which show more of a dispersion of the data.

Figure 5 shows the performance of the SVM, Random Forest, Linear Logistic Regression, and k-Nearest-Neighbour machine learning models. The averages are based on the performance of the models in all 21 analyses of Path B.
The SVM has the highest AUC score of 94.5% which is significantly better than all other three models, followed by Random Forest (AUC = 93.6%) and LLR (92.7%). The kNN model performs

significantly worse than all other three models with an AUC of 84.2% with a significance of p < 2.22e-16 compared to the SVM and RF models, and p-value of 3.8e-13 compared to Linear Logistic Regression. The kNN model also shows a dispersion of the data points compared to SVM, RF, and LLR which have a lower variability.

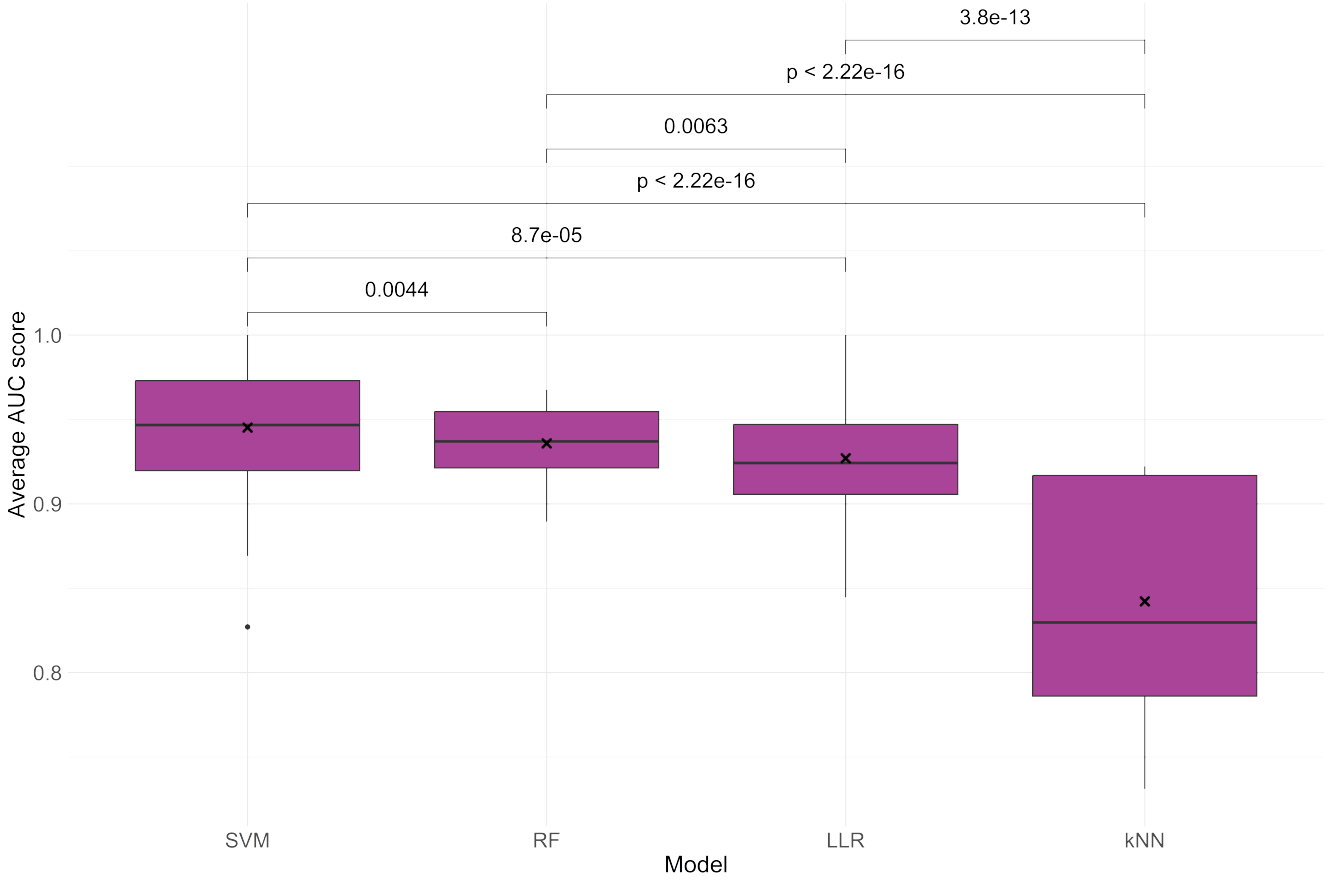


Figure 5: **Machine learning model performance** This figure shows the average performance of the machine learning models. The performances are based on all 21 analyses (SVM = 6 analyses, RF = 6 analyses, LLR = 6 analyses, kNN = 3 analyses). From left to right, the average AUCs are 94.5%, 93.6%, 92.7%, and 84.2%. The associated p-values of the difference in AUC per model calculated with the Mann-Whitney U test are given at the top of the boxplots. The y-axis starts at an AUC score of 0.7 due to scale considerations.

3.3 Type of RNA comparison

When looking at the performance of the individual models on the different datasets, it is noticeable that the SVM and LLR models show no significant difference in performance when applied to ALL, NC, and CODING genes separately. Random Forest does show a significantly better AUC score when using the noncoding genes dataset compared to using the coding genes dataset. Combining the two also results in a significantly better AUC score than using only the coding genes dataset. The performance of the k-Nearest-Neighbour model is significantly different between the 'ALL', 'NC', and 'CODING' datasets, where the noncoding dataset has the highest performance with an AUC of 91.7%. However, the performances of the kNN model deviate from the other three models, especially when applied to the 'ALL' and 'CODING' datasets, where the model achieves an AUC

of 83.9% and 77.1% respectively. Therefore, the kNN model results will be left out of the further comparisons.

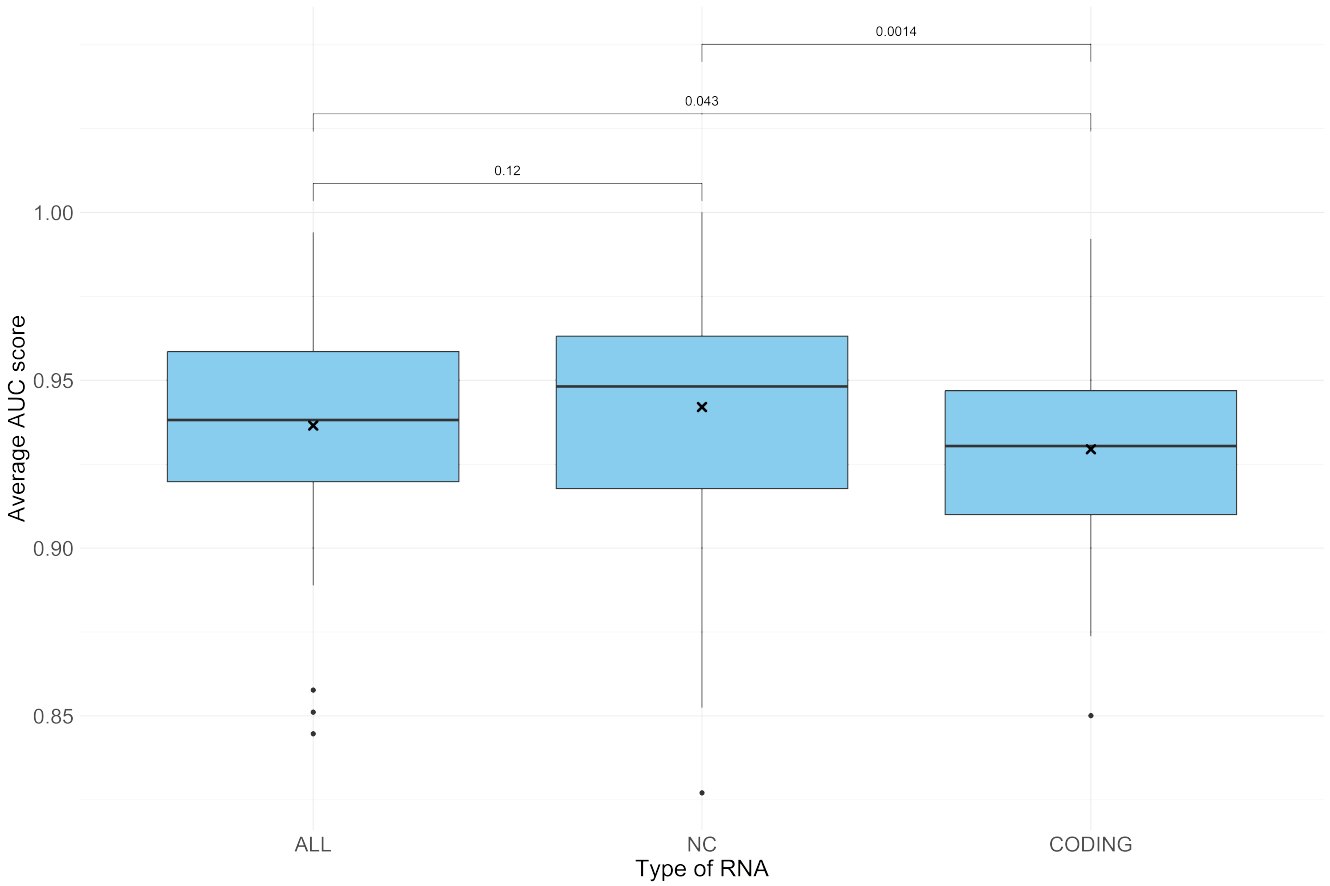


Figure 6: **Performance per type of RNA (Path B)** This figure shows the average performance of the SVM and RF models on the different datasets (ALL, NC, CODING). From left to right, the average AUCs are 93.7%, 94.2%, and 92.9%. The associated p-values of the difference in AUC per type of RNA are given at the top of the boxplots. The y-axis starts at an AUC score of 0.82 due to scale considerations.

From Figure 6 and Figure 14 it can be observed that the performance of predicting leprosy progression based on only noncoding genes is significantly better than making predictions based on only coding genes (p-value = 0.0014). Figure 6 shows that combining coding- and noncoding genes results in a higher average AUC score than performing machine learning only on coding genes. With an AUC of 94.2% noncoding has a higher average AUC score than the performance of the ML models applied to the combination of noncoding- and coding genes (ALL), which has an AUC of 93.7%. However, this is not a significant difference (p-value = 0.12). This contrasts with the results from Figure 14, which sees a significant difference between all genes versus noncoding genes (p-value = 0.0039).

3.4 Finding biomarkers

As shown by Table 6, thirty differentially expressed genes were identified by the machine learning methods; Random Forest and Support Vector Machine. This table is the combination of two tables that contain the top twenty most occurring genes of the sub-analyses performed by path A and path B, of which the corresponding ROC curves are shown in Figure 15a and Figure 15b respectively.
The top twenty table of Path A consists of the bolted gene names and the genes that have a different DGE method than "Intersection". These genes were identified to most occurring genes from the sub-analyses of Path A with an AUC score above or equal to 96.0%. If these twenty genes were used as features inside the Random Forest model, it resulted in a model with an accuracy of 87.7% and an AUC score of 96.8%.
Alternatively, the twenty genes with the DGE method "Intersection" result into the top twenty table discovered by Path B. Similarly to the top twenty table of Path A, these genes were also identified as the most occurring genes from the sub-analyses with an AUC score above or equal to 96.0%. However, the difference is that these analyses stems from Path B. With these twenty genes, a SVM model was trained and tested, resulting in an accuracy of 87.2% and an AUC score of 94.8%.
From the thirty genes in Table 6, ten genes (underlined) were classified with a biotype indicating they are coding genes, while twenty genes were classified with a biotype indicating they are non-coding genes. Within the noncoding genes, the following Ensembl biotypes [Ens24] were identified; transcribed unprocessed pseudogene, processed pseudogene, antisense, processed transcript, unprocessed pseudogene, lincRNA, and misc_RNA.
Additionally, this table contains ten genes, which are not the same ten as the ten coding genes, that were discovered by both the results from Path A as the results from Path B (indicated by the gene names being bold). When using these 10 genes as features for a Support Vector machine, the model had an accuracy of 91.0% and an AUC score of 98.3%. In addition, to identify whether smaller subsets have a similar or improved performace, RFE feature selection was used to create subgroups with features of 3-10 genes. These models identified a subgroup of three genes, namely RN7SL3 or Metazoa_SRP, RP11-533E19.7, and TMEM238, with an accuracy of 91.0% and an AUC score of 98.4%. The corresponding ROC curves of the combination of 10 and 2

Table 6: **Gene information on most occurring differentially expressed genes** In this figure, 30 differentially expressed genes are shown. These 30 genes were the combination of the top 20 most occurring genes of the sub-analyses of Path A and B with an AUC score above or equal to 96.0%. The genes that are shown in bold are the intersection between the top 20 genes of each Path. Underlining the genes showcases whether a gene is a coding (underlined) or a noncoding genes (not underlined). The column "Type of RNA" indicates the biotypes from the ensemble biotypes [Ens24]. The column "DGE analysis method" showcases which DGE analysis method identified the gene to be differentially expressed. In this column Intersection indicates that all the DGE method did identify the gene as differentially expressed

Gene name	Ensembl ID	Type of RNA	DGE analysis method
CROCCP2	ENSG00000215908	transcribed unprocessed pseudogene	DESeq2, EdgeR
CTB-79E8.3	ENSG00000253683	processed pseudogene	DESeq2, EdgeR
DPM3	ENSG00000179085	protein coding	Intersection
FOXN3-AS1	ENSG00000258920	antisense	LimmaVoom
GAS5	ENSG00000234741	processed transcript	DESeq2, EdgeR
HCG4P11	ENSG00000225864	unprocessed pseudogene	Intersection
HDAC4-AS1 or AC062017.1	ENSG00000222020	antisense	Intersection
HES4	ENSG00000188290	protein coding	Intersection
HNRNPD-DT or RP11-127B20.3	ENSG00000272677	antisense	Intersection
LINC00865	ENSG00000232229	lincRNA	Intersection
MFSD3	ENSG00000167700	protein coding	Intersection
PSMA6P1	ENSG00000215414	processed pseudogene	DESeq2, LimmaVoom
PYURF	ENSG00000145337	protein coding	DESeq2, EdgeR
RN7SL2	ENSG00000274012	misc RNA	LimmaVoom
RN7SL3 or Metazoa_SRP	ENSG00000278771	misc RNA	Intersection
RP11-533E19.7	ENSG00000272906	lincRNA	Intersection
RP11-59C5.3	ENSG00000273599	antisense	Intersection
RPL13AP5	ENSG00000236552	processed pseudogene	Intersection
RPS3P3 or RP11-778D9.4	ENSG00000228205	processed pseudogene	Intersection
RPS9	ENSG00000170889	protein coding	DESeq2, EdgeR
SNHG32 or C6orf48	ENSG00000204387	protein coding	Intersection
SNHG5	ENSG00000203875	processed transcript	Intersection
SNHG8	ENSG00000269893	lincRNA	Intersection
SNX32	ENSG00000172803	protein coding	Intersection
TMEM121	ENSG00000184986	protein coding	Intersection
TMEM191A	ENSG00000226287	transcribed unprocessed pseudogene	LimmaVoom
TMEM238	ENSG00000233493	protein coding	Intersection
TPGS1	ENSG00000141933	protein coding	Intersection
UQCRFS1P1	ENSG00000226085	processed pseudogene	Intersection
ZNF252P-AS1	ENSG00000255559	antisense	LimmaVoom

In Figure 7, the statistical significance as well as the fold change between conditions First and HHC of the genes from Table 6 are displayed. From this figure, it is visible that all thirty genes are identified to be upregulated, meaning that gene expression in the condition First is at least 50% higher than the gene expression in the condition HHC.

However, some genes (GAS5, CTB-79E8.3, PYURF, CROCCP2) are not within the box enclosed by the dashed lines (p-value is 0.05 and fold change = 1.5). The reason behind this is that these genes are not identified by the LimmaVoom method. This occurs because the data behind this plot originates from the DGE analysis results from LimmaVoom rather than the DGE method they were discovered with. Even so, in Figure 10 it is visible that genes are still upregulated when comparing the RNA counts (in CPM adjusted to $\log_2$) between the condition First and HHC.

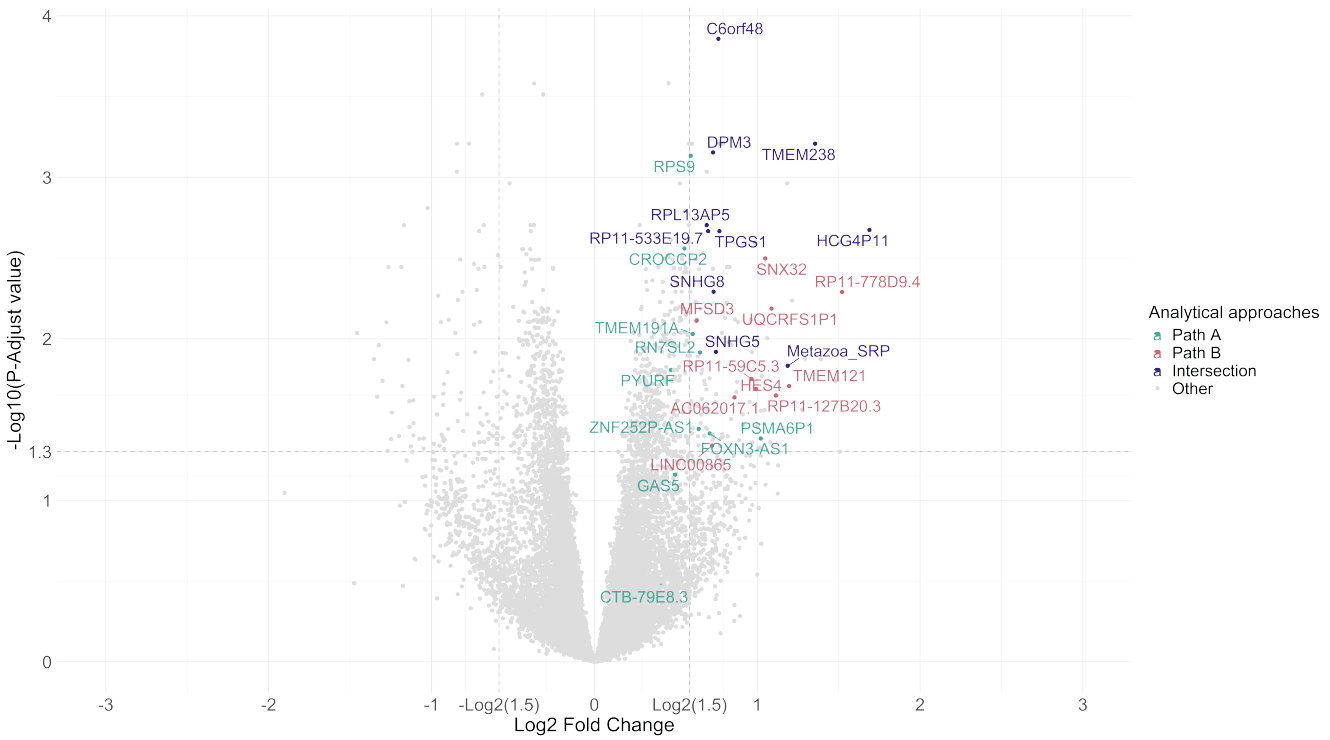
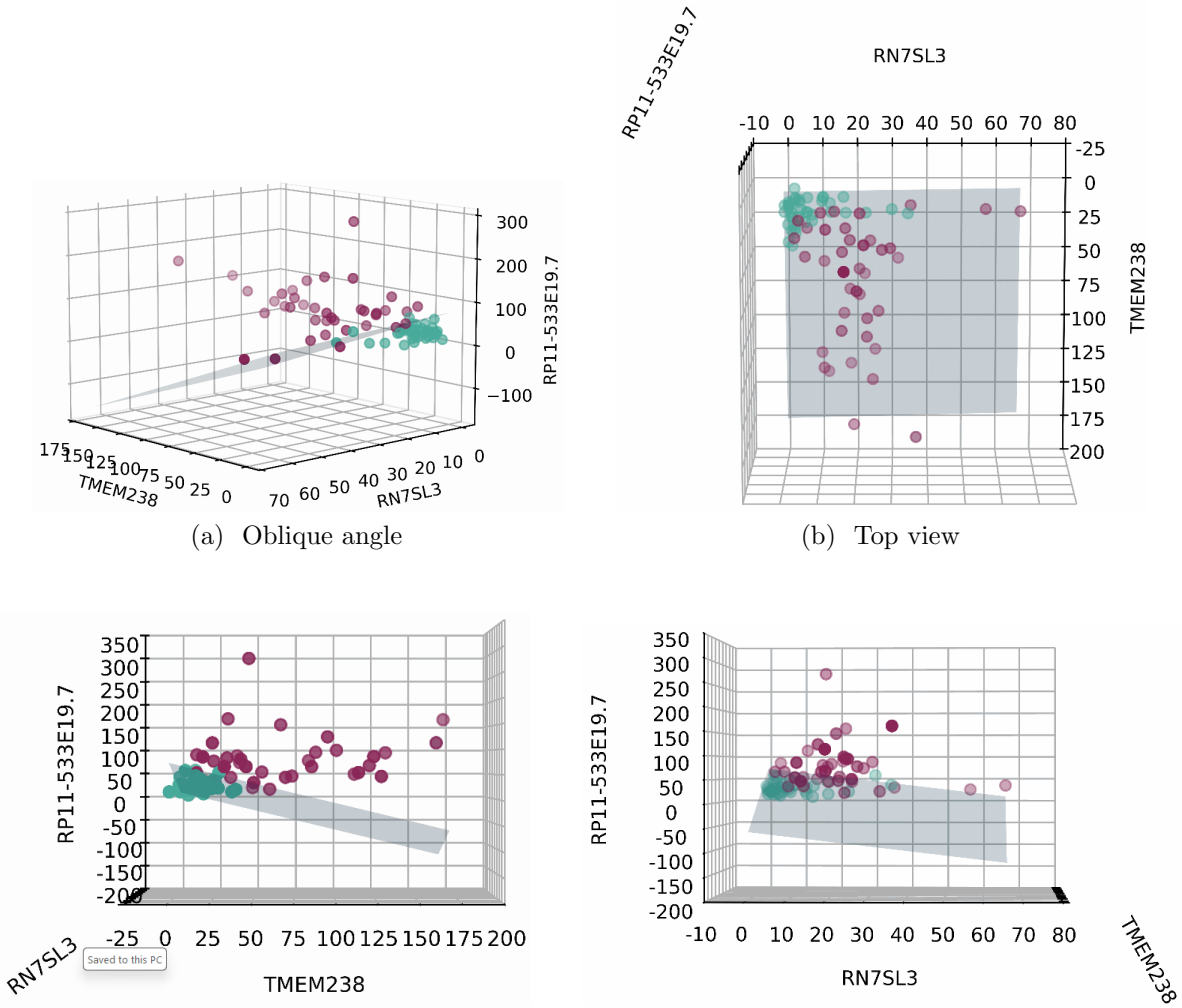


Figure 7: **Volcanoplot showcasing the difference in fold change and statistical significance between the 30 differentially expressed genes** This Volcanoplot showcases the 30 genes from Table 6 with the genes from Path A in green, the genes from Path B in pink and the genes from the intersection of Path A and B in blue. The x-axis indicates the Fold change in $\log_2$ and the y-axis indicates the statistical significance with the p-value in $-\log_{10}$. For a gene to be identified as differentially expressed, the gene should have a difference of 50% in gene expression ($\pm \log_2(1.5)$) between the two conditions and a statistical significance in p-value equal or greater than 0.05 ($-\log_{10}(0.05) \approx 1.3$). The data behind this Volcanaplot is from the differentially gene expression analysis performed by LimmaVoom

In Figure 8 the hyperplane defined by the Support Vector Machine that separates the positively and negatively diagnosed samples from each other based on the expression levels of TMEM238, RN7SL3, and RP11-533E19.7 is visualized. TMEM238, RN7SL3 and RP11-533E19 are the three genes from the 3-gene combination that results in the highest performance (dit klopt niet echt denk ik?). A clear separation of the two classes can be seen, especially from the angle shown in Figure 8a. When looking at Figure 8c it can be seen that overall, leprosy progressors show an increase in expression of the TMEM238 gene, as well as the RN7SL3 and Rp11-533E19.7 genes compared to household contacts.



(a) Oblique angle

(b) Top view

(c) Front view

(d) Side view

Figure 8: **3D hyperplane visualization of the best 3-gene combination** The green colored data points represent the positively diagnosed samples (First) and the pink the negatively diagnosed samples (HHC). The axes show the expression levels of the three genes in TEA3LEP: TMEM238, RN7SL3, and RP11-533E19.7. The predictions are made using a Support Vector Machine.

4 Discussion

In this section we interpret the findings from our investigation into differential gene expression analysis methods, machine learning models and feature selectors, and biomarker discovery. Subsection 4.1 discusses the findings on the comparison of the performance of the DGE methods, followed by the interpretation of the performances of the different machine learning models and feature selectors covered in section 4.2. In addition, the results of using different datasets are deliberated in subsection 4.3. To conclude, the theoretical biomarkers found will be contextualized in subsection 4.4.

4.1 Differential gene expression analysis methods

As shown in Figure 3, LimmaVoom is significantly outperforming the other two DGE methods; DESeq2 and EdgeR with a difference in AUC score of 2.1% ($p = 0.0034$) and 1.9% ($p = 0.0043$) respectively. This finding is similar to previous research applied to the comparison of different DEG methods [LGP+22, QQW+22, LGC+16, CRAL+20]. The specific reason behind this increase in accuracy is difficult to tell. However one of the reason could be that the negative binomial distribution, which is assumed by both DESeq2 and EdgeR, is not suited for the data used in this DGE analysis. Alternatively, it could be that the voom transformation in order to shrink the RNA counts is performing better due to the larger amount of samples (n=80).
In addition, to this finding, it is shown in Figure 3 that the DGE methods DESeq2 and EdgeR result in Random Forest models that have a similar performance with an AUC score of 93.0% and 93.2%. As Table 5 showcase, the differentially expressed genes, which are used as features, between these two methods have on average 87.1% overlap between these two DGE methods. In all likelihood, this overlap in genes is the reason behind the similarity in performance of the model. Unfortunately, the impact of the difference in normalization technique between the two DGE methods is not distinguishable from these results. However previous research [DRA+13b, CRAL+20] indicates that although TMM and Median of ratios are different, they are both considered to be performing similarly well on smaller sample sizes used in this research. Additionally, research performed by Luis A. Corchete et. al. [DRA+13b] claims that the normalization technique itself has a small or negligible impact on the discovered differentially expressed genes by a DGE method. Therefore, it is likely that the small difference in discovered genes and model performance is due to the normalization technique.
Lastly, as shown in Figure 3, it is visible that the performance of the Random Forest models trained with the features (genes) from the intersection of the differentially expressed genes of each DGE method is significantly the Random Forest models trained with the features from the union. This is because the true positive rate (TPR) of the genes discovered by each DGE method has more certainty than the genes discovered by the DGE method individually. In addition to this, the sub-analyses from the intersection with the RFE feature selection will be higher than those from the union, because RFE performs better, when it selects features from a smaller dataset.

4.2 Machine learning and feature selectors

In subsection 3.2 it was shown that the RFE feature selector performed better than Chi-Squared. Specifically in Figure 13, RFE is outperforming Chi-squared feature selection with an average of

3.1% (p-value ¡ 0.002). A possible explanation could be that RFE bases the selection on the performance of subsets of genes instead of single genes, and thus also looks at gene combinations. The stability and low variance of the Random Forest model could possibly be attributed to the resilient character of Random Forest and its ability to adapt to varying distributions and relationships between features. Due to its ensemble nature and use of bagging, the model does not overly rely on certain features since it uses random subsets of features for each decision tree. Furthermore, Random Forest is very robust and can handle outliers.
When analysing the results, the SVM model appeared to be the best performing machine learning model compared to Random Forest, Linear Logistic Regression, and k-Nearest-Neighbour. Beforehand it had already been acknowledged that Support Vector Machines are known for their ability to perform well on high-dimensional, small datasets, while the other models might suffer from the curse of dimensionality, which is why this result could be expected. The Random Forest model could be outperforming the Logistic Regression and kNN model since it is very robust and resilient when dealing with outliers and noise, while Logistic Regression and especially kNN are not.

4.3 Type of RNA comparison

From both Figures 6 & 14, the conclusion can be made that the models trained with features containing only noncoding genes or noncoding and coding genes are significantly outperforming the models trained with features only containing coding genes. As discussed by Aleksandra E. Kornienko et al. [KDG+16], there is an increase in variation of gene expression of long noncoding RNAs (lncRNAs) between individuals. Therefore, the variation between conditions might also vary more, which leads to an increase of performance due to a better distinction of conditions.
However, not all noncoding genes are classified as long noncoding RNAs. From the noncoding genes, other types are noncoding RNAs and pseudogenes. It is expected that the noncoding RNAs showcase similar variability as long noncoding RNAs, because of their similarity in function. But for the type pseudogene, it is harder to predict. Although more than 65% of the noncoding RNAs are not pseudogenes, their variability should be determined in future research. Other research [MGB23] performed on the difference of performance based on the type of RNA supports the claim that including noncoding genes into the features improves the prediction significantly.

4.4 Finding biomarkers

Tio Coma et al found a biomarker of 4 genes, RISK4LEP, which achieved an accuracy of 79.0% and AUC of 86.4%. When we compare this to our best 3-gene combination, TEA3LEP which has an accuracy of 91.0% and AUC of 98.4%, it can be concluded that TEA3LEP in theory is a better predictor. However, this is not validated using RT-qPCR in contrast to RISK4LEP.

It is also noticeable that TPGS1 was found by Tío-Coma et. al. as well as in our top 10. TPGS1 is a protein-coding gene located in the centrosome that is inferred to be associated with borderline leprosy and intermediate leprosy [gena]. TPGS1 stands for Tubulin Polyglutamylase Complex Subunit 1 and is part of the TPGC complex that forms the Tubulin Polyglutamylase Complex [WPK+22]. Polyglutamylation is a post-transcriptional modification where glutamines are added to the C-terminal ends of tubulin [BJM21]. TPGS1 knockout mice have shown that TPGS1 affects tubulin

polyglutamylation and the binding of certain MAPs in neurons. Together with the TTLL1 enzyme TPGS1 is needed for synaptic function, and the development of cilia and sperm flagella [WPK+22]. Additionally, TPGS1's related pathway is the metabolism of proteins.

TMEM238 is one of the three genes in the best performing 3-gene combination found in this thesis, TEA3LEP. TMEM238 is a protein coding gene which encodes for a transmembrane protein [genb]. It is associated with Bardet-Biedl syndrome, which is characterized by rod-cone dystrophy, polydactyly, brachydactyly, and ataxia among other symptoms [SI16]. Genes that are part of the Bardet-Biedl Syndrome (BBS) gene family have shared roles in the functioning and formation of cilia [VRY+12]. While symptoms and some associated genes between leprosy and the Bardet-Biedl syndrome might have some similarity, there is no real connection between the two.

Two other genes within TEA3LEP is are RP11-533E19.7 and RN7SL3. Unfortunately, there is currently little known about the two genes. RP11-533E19.7 is identified to be a novel transcript, which is an antisense to TOR1AIP1. Therefore, it is expected that the gene is regulating this gene, that is required for the nuclear membrane integrity. In specific, TOR1AIP1 might have a role in membrane attachment and assembly of the nuclear lamina [MLBS23].
The other gene, RN7SL3 is a 7SL RNA molecule [EFJB04]. This molecule together with six polypeptides combine to the signal recognition particle (SRP). Since this has not been researched or analysed yet, it is hard to discuss the relationship with leprosy.
Additionally, the gene family SNHG, Small Nucleolar RNA Host Gene, appears to be promising a promissing biomarker. Of this gene family, three genes (SNHG5, SNHG8 and SNHG32) were identified as differentially expressed by the intersection of the three DGE methods. In addition, one gene (GAS5, previously known as SNHG2) was identified only by DESeq2 and EdgeR. When looking deeper into the SNHG family, 8 genes (SNHG3, SNHG6, SNHG7, SNHG10, SNHG15, SNHG17, SNHG19 and SNHG25) were upregulated, but did not fit the criteria of fold change and / or p-adjust value. These genes appear to are involved in five molecular mechanism that are currently known. According to Alina-Andreea Zimta et. al. [ZTB+20], these 5 molecular functions are:

1. Influencing DNA methylation through modulation of methylation enzymes
2. Interaction with transcription factors and repression of gene transcription.
3. MiRNA sponging and the releasing of miRNA targets
4. Direct binding to the mRNA and repression of translation
5. Prevention of protein ubiquitination through single protein interaction or multi-protein complex.

5 Conclusions and Further Research

In conclusion, this research provides significant insights in understanding and predicting leprosy susceptibility through differential gene expression analysis methods, the application of feature selection techniques and machine learning models, and the identification of a key biomarker. These findings highlight several contributions to the field.
With this research we sought to answer the following research questions;

1. Which genes can be considered biomarkers for leprosy?
2. Which differential gene expression analysis method performs best in predicting leprosy progression?
3. Which feature selector technique performs best in predicting leprosy progression from RNAseq data?
4. Which machine learning model performs best in predicting leprosy progression from RNAseq data?
5. Are differentially expressed genes found at preclinical timepoints similar to identified leprosy biomarkers in previous research?
6. What is the effect of the inclusion of noncoding genes in the prediction of leprosy progression?

The answers to these questions and possible explanations for these findings will be discussed in the following section.

5.1 Conclusion

First, a promising biomarker, TEA3LEP, was identified as a promising biomarker containing the genes TMEM238, RP11-533E19.7, and RN7SL3. This biomarker demonstrated theoretically improved performance compared to similar research, with an accuracy of 91.0% and an AUC score of 98.4%, indicating its high potential as a plausible predictor for leprosy susceptibility.
When evaluating differential gene expression (DGE) analysis methods, it is shown that LimmaVoom outperforms the other two methods—DESeq2 and EdgeR by a significant margin. The intersection approach of using all three DGE methods yielded better results compared to the union approach. This showcases the possible false positive rate of differentially expressed genes discovered by individual methods.
In terms of feature selection, the Recursive Feature Elimination (RFE) technique outperformed the Chi-Squared feature selector across nearly all tested models, except for k-Nearest neighbours, which does not support the use of RFE. This highlights the importance of looking at subsets of genes rather than individual genes when identifying a transcriptomic biomarker.
Among the machine learning models tested, the Support Vector Machines proved to be the most effective in predicting leprosy susceptibility, outperforming other models such as Logistic Regression and k-Nearest neighbours, which demonstrated less reliable performance. However, Random Forest showed a more stable performance across subanalyses, suggesting that while SVM is highly accurate, Random Forest may be preferred in situations where model stability is crucial.
An important additional key finding from this research is the importance of including noncoding

genes in predictive models. It was observed that the inclusion of noncoding genes not only improved the performance of models containing both coding and noncoding genes, but also showed that noncoding genes alone were capable of outperforming coding genes in certain cases. This suggests that noncoding genes play a significant role in the identification of a disease and should be included when identifying a biomarker.
Finally, TPSG1 was identified in this research as well as the previous study performed by Tío-Coma et. al. [TC21]. This indicates that the additional methods support the previous claim of this gene.

5.2 Future work

Possible future research on this experiment should focus on validating the TEA3LEP biomarker identified. Although the TEA3LEP showcases promising results, it should be testing with a RT-qPCR experiment in order to confirm the results.
In addition, the current study only identified the differentially expressed genes between HHC and First. However it would be an interesting angle to analyse both the comparison HHC and First to the comparison HHC and Second. With these analyses it would be possible to see whether a better biomarker could be found that can identify both the progressor before and after showcasing symptoms.

6 Acknowledgements

While performing this research multiple people contributed tremendously to the outcome of this paper.

The first person that we want to thank is our supervisor Fons Verbeek. During our research, he always made sure we had what we needed to finish this research with success.

Additionally, we want to show our gratitude to Annemieke Geluk. She made significant contributions to this research by providing valuable insights during our discussions of preliminary data and suggesting compelling directions for future studies.

Last but not least, we wish to thank Matheus Rogério Almeida. He was of great importance while understanding the previous research [TC21] and supporting our findings from a biological standpoint.

References

[ASTS20] Charlotte Avanzi, Pushpendra Singh, Richard W. Truman, and Philip N. Suffys. Molecular epidemiology of leprosy: An update. *Infection, Genetics and Evolution*, 86:104581, 2020.

[Ban19] Michael Banf. Learning theory and support vector machines-a primer. *arXiv preprint arXiv:1902.04622*, 2019.

[BCSMAB13] Verónica Bolón-Canedo, Noelia Sánchez-Maroño, and Amparo Alonso-Betanzos. A review of feature selection methods on synthetic data. *Knowledge and information systems*, 34:483–519, 2013.

[BH95] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.

[Bio] Bioconductor. Open source software for bioinformatics.

[BJM21] Satish Bodakuntla, Carsten Janke, and Maria M Magiera. Tubulin polyglutamylation, a regulator of microtubule functions, can cause neurodegeneration. *Neuroscience Letters*, 746:135656, 2021.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[CGLRML20] Jair Cervantes, Farid Garcia-Lamont, Lisbeth Rodríguez-Mazahua, and Asdrubal Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189–215, 2020.

[CRAL+20] Luis A Corchete, Elizabeta A Rojas, Diego Alonso-López, Javier De Las Rivas, Norma C Gutiérrez, and Francisco J Burguillo. Systematic comparison and assessment of rna-seq procedures for gene expression quantitative analysis. *Scientific reports*, 10(1):19737, 2020.

[DRA+13a] Marie-Agnès Dillies, Andrea Rau, Julie Aubert, Christelle Hennequet-Antier, Marine Jeanmougin, Nicolas Servant, Céline Keime, Guillemette Marot, David Castel, Jordi Estelle, et al. A comprehensive evaluation of normalization methods for illumina high-throughput rna sequencing data analysis. *Briefings in bioinformatics*, 14(6):671–683, 2013.

[DRA+13b] Marie-Agnès Dillies, Andrea Rau, Julie Aubert, Christelle Hennequet-Antier, Marine Jeanmougin, Nicolas Servant, Céline Keime, Guillemette Marot, David Castel, Jordi Estelle, et al. A comprehensive evaluation of normalization methods for illumina high-throughput rna sequencing data analysis. *Briefings in bioinformatics*, 14(6):671–683, 2013.

[EFJB04] Markus Englert, Martha Felis, Volker Junker, and Hildburg Beier. Novel upstream and intragenic control elements for the rna polymerase iii-dependent transcription of human 7sl rna genes. *Biochimie*, 86(12):867–874, 2004.

[Ens24] Ensemble. Biotypes, Oct 2024.

[FXMY14] Jiashi Feng, Huan Xu, Shie Mannor, and Shuicheng Yan. Robust logistic regression and classification. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[gena] genecard. https://www.genecards.org/cgi-bin/carddisp.pl?gene=TPGS1. accessed on 18/12/2024.

[genb] genecard. https://www.genecards.org/cgi-bin/carddisp.pl?gene=TMEM238. accessed on 18/12/2024.

[GFLC24] Rosita Guido, Stefania Ferrisi, Danilo Lofaro, and Domenico Conforti. An overview on the advancements of support vector machine models in healthcare applications: A review. *Information*, 15(4), 2024.

[GWBV02] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46:389–422, 2002.

[HJLS13] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*. John Wiley & Sons, 2013.

[HS14] Xiang Han and Francisco Silva. On the age of leprosy. *PLoS neglected tropical diseases*, 8:e2544, 02 2014.

[HSC24] Chien-Yuan Huang, Shih-Bin Su, and Kow-Tong Chen. An update of the diagnosis, treatment, and prevention of leprosy: A narrative review. *Medicine*, 103, 2024.

[Hun07] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[KDG+16] Aleksandra E Kornienko, Christoph P Dotter, Philipp M Guenzl, Heinz Gisslinger, Bettina Gisslinger, Ciara Cleary, Robert Kralovics, Florian M Pauler, and Denise P Barlow. Long non-coding rnas display higher natural expression variation than protein-coding genes in healthy humans. *Genome biology*, 17:1–23, 2016.

[LDS89] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

[LGC+16] Yanzhu Lin, Kseniya Golovnina, Zhen-Xia Chen, Hang Noh Lee, Yazmin L Serrano Negron, Hina Sultana, Brian Oliver, and Susan T Harbison. Comparison of normalization and differential expression analyses using rna-seq data from 726 individual drosophila melanogaster. *BMC genomics*, 17:1–20, 2016.

[LGP+22] Yumei Li, Xinzhou Ge, Fanglue Peng, Wei Li, and Jingyi Jessica Li. Exaggerated false positives by popular differential expression methods when analyzing human population samples. *Genome biology*, 23(1):79, 2022.

[LHA14] Michael I Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for rna-seq data with deseq2. *Genome biology*, 15:1–21, 2014.

[MGB23] Deisy Morselli Gysi and Albert-Laszlo Barabasi. Noncoding rnas improve the predictive power of network medicine. *Proceedings of the National Academy of Sciences*, 120(45):e2301342120, 2023.

[MLBS23] Laurane Mackels, Xincheng Liu, Gisèle Bonne, and Laurent Servais. Tor1aip1-associated nuclear envelopathies. *International Journal of Molecular Sciences*, 24(8), 2023.

[MLZ24] Zihao Mi, Hong Liu, and Furen Zhang. Advances in the pathogenic, genetic and immunological studies of leprosy. *hLife*, 2(1):6–17, 2024.

[MR16] M. N. Murty and Rashmi Raghava. *Kernel-Based SVM*, pages 57–67. Springer International Publishing, Cham, 2016.

[MW47] Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947.

[OMM+21] Marcus Fernandes de Oliveira, Rychelle Clayde Affonso Medeiros, Bruno de Siqueira Mietto, Thyago Leal Calvo, Ana Paula Miranda Mendonça, Thabatta Leal Silveira Andrezo Rosa, Débora Santos da Silva, Karina Girardi do Carmo de Vasconcelos, Antonio Marcos Rodrigues Pereira, Cristiana Santos de Macedo, Geraldo Moura Batista Pereira, Marcia de Berrêdo Pinho Moreira, Maria Cristina Vidal Pessolani, Milton Ozório Moraes, and Flávio Alves Lara. Reduction of host cell mitochondrial activity as mycobacterium leprae's strategy to evade host innate immunity. *Immunological Reviews*, 301:193–208, 2021.

[Orga] World Health Organization. https://www.who.int/westernpacific/health-topics/leprosy#tab=tab_1. accessed on 15/12/2024.

[Orgb] World Health Organization. https://www.who.int/news-room/fact-sheets/detail/leprosy. accessed on 15/11/2024.

[Por19] John C. Porcello. Designing and implementing svms for high-dimensional knowledge discovery using fpgas. In *2019 IEEE Aerospace Conference*, pages 1–8, 2019.

[PVG+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[QQW+22] Han Qu, Meng Qu, Shibo Wang, Lei Yu, Qiong Jia, Xuesong Wang, and Zhenyu Jia. Differential expression analysis: Simple pair, interaction, time-series. *Bio-101*, page e4455, Jul 2022. Bio-101 2022;:e4455.

[R C24] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2024.

[Rig17] Steven J. Rigatti. Random forest. *Journal of Insurance Medicine*, 47(1):31–39, 01 2017.

[RJ62] DS Ridley and WH Jopling. A classification of leprosy for research purposes., 1962.

[RMS09] Mark D. Robinson, Davis J. McCarthy, and Gordon K. Smyth. edger: a bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1):139–140, 11 2009.

[RO10] Mark D Robinson and Alicia Oshlack. A scaling normalization method for differential expression analysis of rna-seq data. *Genome biology*, 11:1–9, 2010.

[RPW+15] Matthew E Ritchie, Belinda Phipson, DI Wu, Yifang Hu, Charity W Law, Wei Shi, and Gordon K Smyth. limma powers differential expression analyses for rna-sequencing and microarray studies. *Nucleic acids research*, 43(7):e47–e47, 2015.

[RvE] Aart Rosmolen and Emma Rosa van Eerde. Leprosy RNA analysis.

[SI16] Evgeny N. Suspitsin and Evgeny N. Imyanitov. Bardet-biedl syndrome. *Molecular Syndromology*, 7(2):62–71, 04 2016.

[Sla] Scikit-learn. https://scikit-learn.org/1.5/modules/linear_model.html#binary-case. accessed on 23/12/2024.

[Slb] Scikit-learn. https://scikit-learn.org/1.5/modules/svm.html. accessed on 15/11/2024.

[Sto11] Jill C. Stoltzfus. Logistic regression: A brief primer. *Academic Emergency Medicine*, 18(10):1099–1104, 2011.

[taLUMC19] SASC team at Leiden University Medical Center. Biowdl rna-seq pipeline, Dec 2019.

[TC21] M. Tio-Coma. Blood rna signature risk4lep predicts leprosy years before clinical onset. *Elsevier*, 2021.

[TDVS19] Kashvi Taunk, Sanjukta De, Srishti Verma, and Aleena Swetapadma. A brief review of nearest neighbor algorithm for learning and classification. In *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pages 1255–1260, 2019.

[UdB+22] Kedir Urgesa, Naomi D. de Bruijne, Kidist Bobosha, Berhanu Seyoum, Adane Mihret, Biftu Geda, Anne Schoenmakers, Liesbeth Mieras, Robin van Wijk, Christa Kasang, Mirgissa Kaba, and Abraham Aseffa. Prolonged delays in leprosy case detection in a leprosy hot spot setting in eastern ethiopia. *PLoS Neglected Tropical Diseases*, 16(9), September 2022. Funding Information: This project is part of the EDCTP2 programme supported by the European Union awarded to NLR/LM (grant

number RIA2017NIM-1839-PEP4LEP), and the Leprosy Research Initiative (LRI; www.leprosyresearch.org) awarded to NLR/LM (grant number 707.19.58.). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript. Publisher Copyright: © 2022 Urgesa et al.

[Ven18] Naveen Venkat. The curse of dimensionality: Inside out. *Pilani (IN): Birla Institute of Technology and Science, Pilani, Department of Computer Science and Information Systems*, 10, 2018.

[vHG21] Anouk van Hooij and Annemieke Geluk. In search of biomarkers for leprosy by unraveling the host immune response to mycobacterium leprae. *Immunological Reviews*, 301(1):175–192, 2021.

[vHTCV+20] A. (Anouk) van Hooij, M. (Maria) Tió-Coma, E.M. (Els M.) Verhard, M. (Marufa) Khatun, K. (Khorshed) Alam, E. (Elisa) Tjon Kon Fat, D. (Danielle) de Jong, A. (Abu) Sufian Chowdhury, P. (Paul) Corstjens, Jan Hendrik Richardus, and A. (Annemieke) Geluk. Household contacts of leprosy patients in endemic areas display a specific innate immunity profile. *Frontiers in Immunology*, 11, August 2020.

[VRD09] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.

[VRY+12] Megan Smith Valentine, Anbazhagan Rajendran, Junji Yano, S Dilhan Weeraratne, Janine Beisson, Jean Cohen, France Koll, and Judith Van Houten. Paramecium bbs genes are key to presence of channels in cilia. *Cilia*, 1:1–16, 2012.

[vvR+19] Anouk van Hooij, Susan van den Eeden, Renate Richardus, Elisa Tjon Kon Fat, Louis Wilson, Kees L.M.C. Franken, Roel Faber, Merufa Khatun, Khorshed Alam, Abu Sufian Chowdhury, Jan Hendrik Richardus, Paul Corstjens, and Annemieke Geluk. Application of new host biomarker profiles in quantitative point-of-care tests facilitates leprosy diagnosis in the field. *EBioMedicine*, 47:301–308, 2019.

[WFI07] Andrew Worster, Jerome Fan, and Afisi Ismaila. Understanding linear and logistic regression analyses. *Canadian Journal of Emergency Medicine*, 9(2):111–116, 2007.

[Wic16] Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016.

[WPK+22] Lei Wang, Sharad C Paudyal, Yuchen Kang, Mikito Owa, Feng-Xia Liang, Alexander Spektor, Holger Knaut, Irma Sánchez, and Brian D Dynlacht. Regulators of tubulin polyglutamylation control nuclear shape and cilium disassembly by balancing microtubule and actin assembly. *Cell Research*, 32(2):190–209, 2022.

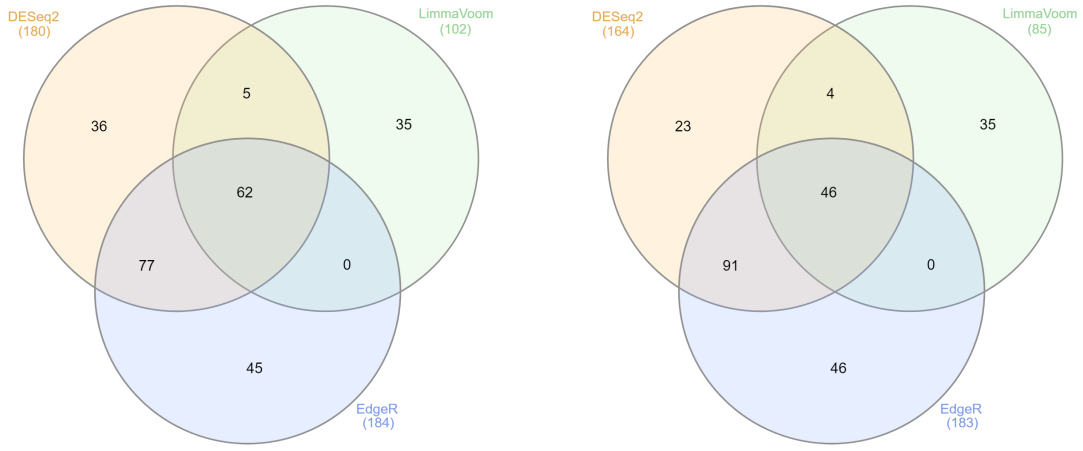
[WZZ+20] Shi W, Mi Z, Wang Z, Zhang H, Wang N, Wang Z, Zhang B, Xia Q, Yu Y, Yu G, Sun L, Fu X, Wang C, Liu H, and Zhang F. Massively parallel sequencing of the filaggrin gene reveals an association between flg loss-of-function mutations and leprosy. *Acta dermato-venereologica*, 01/2020 2020.

[ZPNS+16] Daniel E Zak, Adam Penn-Nicholson, Thomas J Scriba, Ethan Thompson, Sara Suliman, Lynn M Amon, Hassan Mahomed, Mzwandile Erasmus, Wendy Whatney, Gregory D Hussey, Deborah Abrahams, Fazlin Kafaar, Tony Hawkridge, Suzanne Verver, E Jane Hughes, Martin Ota, Jayne Sutherland, Rawleigh Howe, Hazel M Dockrell, W Henry Boom, Bonnie Thiel, Tom H M Ottenhoff, Harriet Mayanja-Kizza, Amelia C Crampin, Katrina Downing, Mark Hatherill, Joe Valvo, Smitha Shankar, Shreemanta K Parida, Stefan H E Kaufmann, Gerhard Walzl, Alan Aderem, and Willem A Hanekom. A blood rna signature for tuberculosis disease risk: a prospective cohort study. *The Lancet*, 387(10035):2312–2322, 2016.

[ZTB+20] Alina-Andreea Zimta, Adrian Bogdan Tigu, Cornelia Braicu, Cristina Stefan, Calin Ionescu, and Ioana Berindan-Neagoe. An emerging class of long non-coding rna with oncogenic role arises from the snorna host genes. *Frontiers in oncology*, 10:389, 2020.

Appendix

The following appendix contains additional information that supports the main content of this report. It includes detailed that are referenced throughout the document but are too extensive or technical to be included in the main body. This section is intended to provide further context and clarify specific points made in the report, enabling readers to delve deeper into the subject matter if they choose. Additionally, the data used and generated from this study can be found in the Github repository [RvE]



(a) Venn Diagram of coding DE genes

(b) Venn Diagram of noncoding DE genes

Figure 9: **Venn Diagram of the number of genes discovered by the three differentialy gene expression analysis methods; DESeq2, EdgeR and LimmaVoom for CODING genes (a) and NONCODING genes (b)** This figure showcases the overlap between the genes discovered by the three DGE methods; DESeq2, EdgeR, and LimmaVoom. The total number of genes discovered by one DGE method can be seen in Table 4

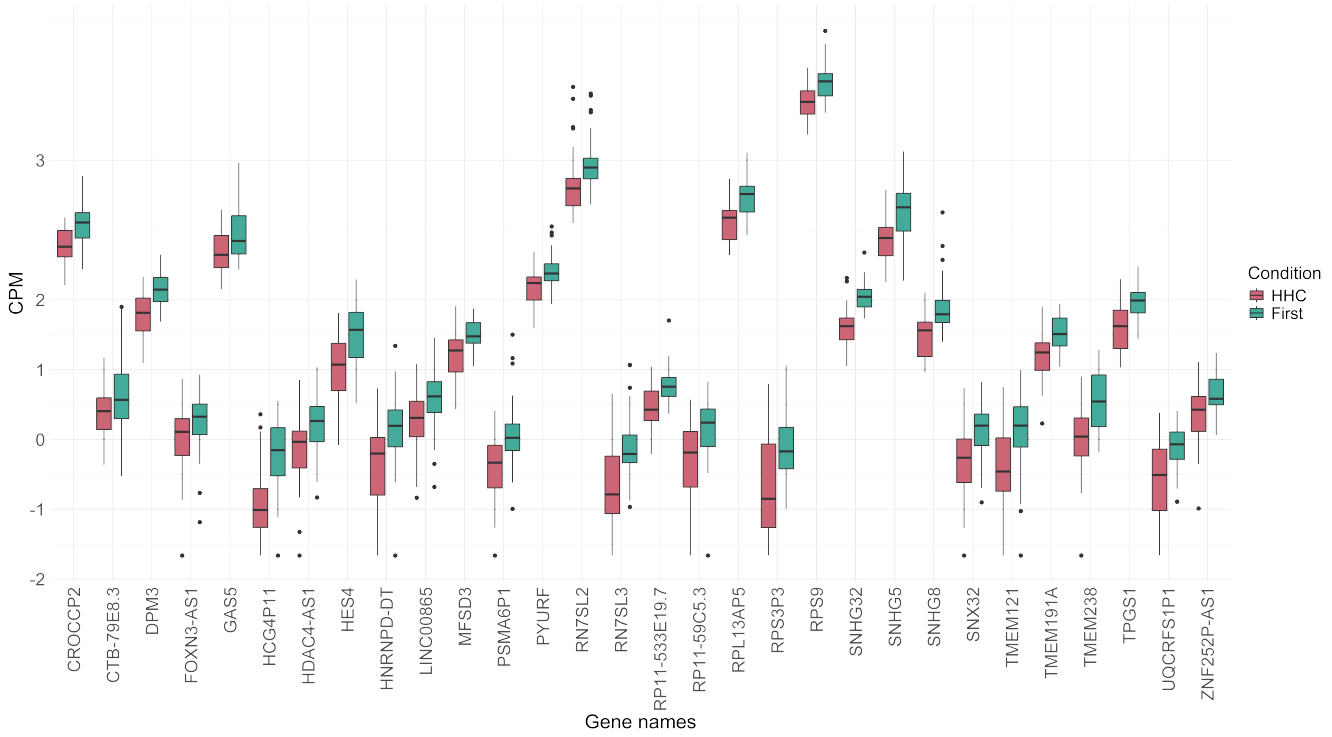
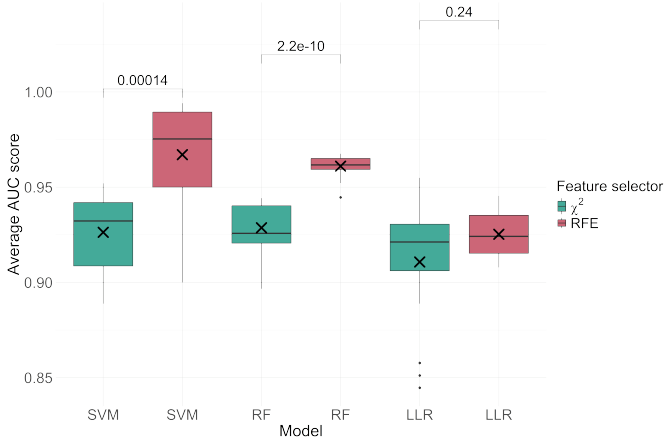
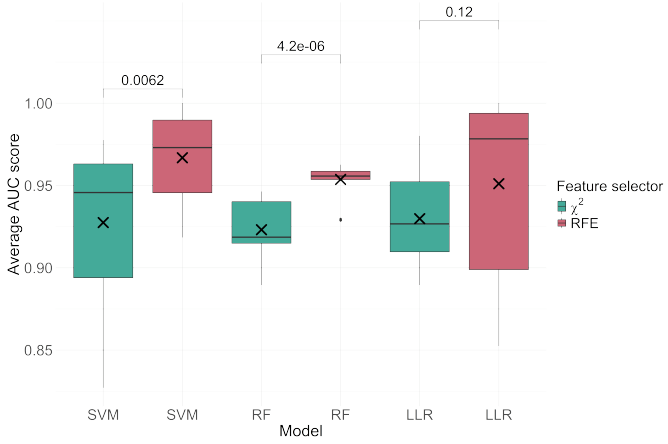


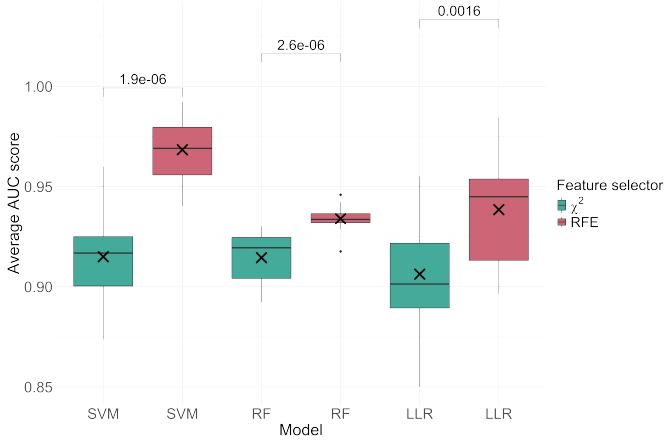
Figure 10: **Difference in gene expression of the top thirty most occurring genes between the two conditions; HHC and First** In this figure, the normalized RNA counts in log_2 per condition (HHC and First) are displayed. The genes shown in this plot originate from table 6. The normalization of RNA counts ([RvE]) is done with Trimmed Mean of M method (the standard normalization technique of EdgeR and LimmaVoom), after which the genes are refactored to the log_2



(a) **ALL genes** The average AUC from left to right are: 92.6%, 96.7%, 92.9%, 96.1%, 91.1%, 92.5%.



(b) **NC genes** The average AUC from left to right are: 92.7%, 96.7%, 92.3%, 95.4%, 93.0%, 95.1%.



(c) **CODING genes** The average AUC from left to right are: 91.5%, 96.8%, 91.5%, 93.4%, 90.6%, 93.8%.

Figure 11: **Performance per model and feature selector** Each figure shows the average AUC of each combination of model and feature selector, except kNN, which does not support RFE and thus is not included in the comparison of feature selector performance.

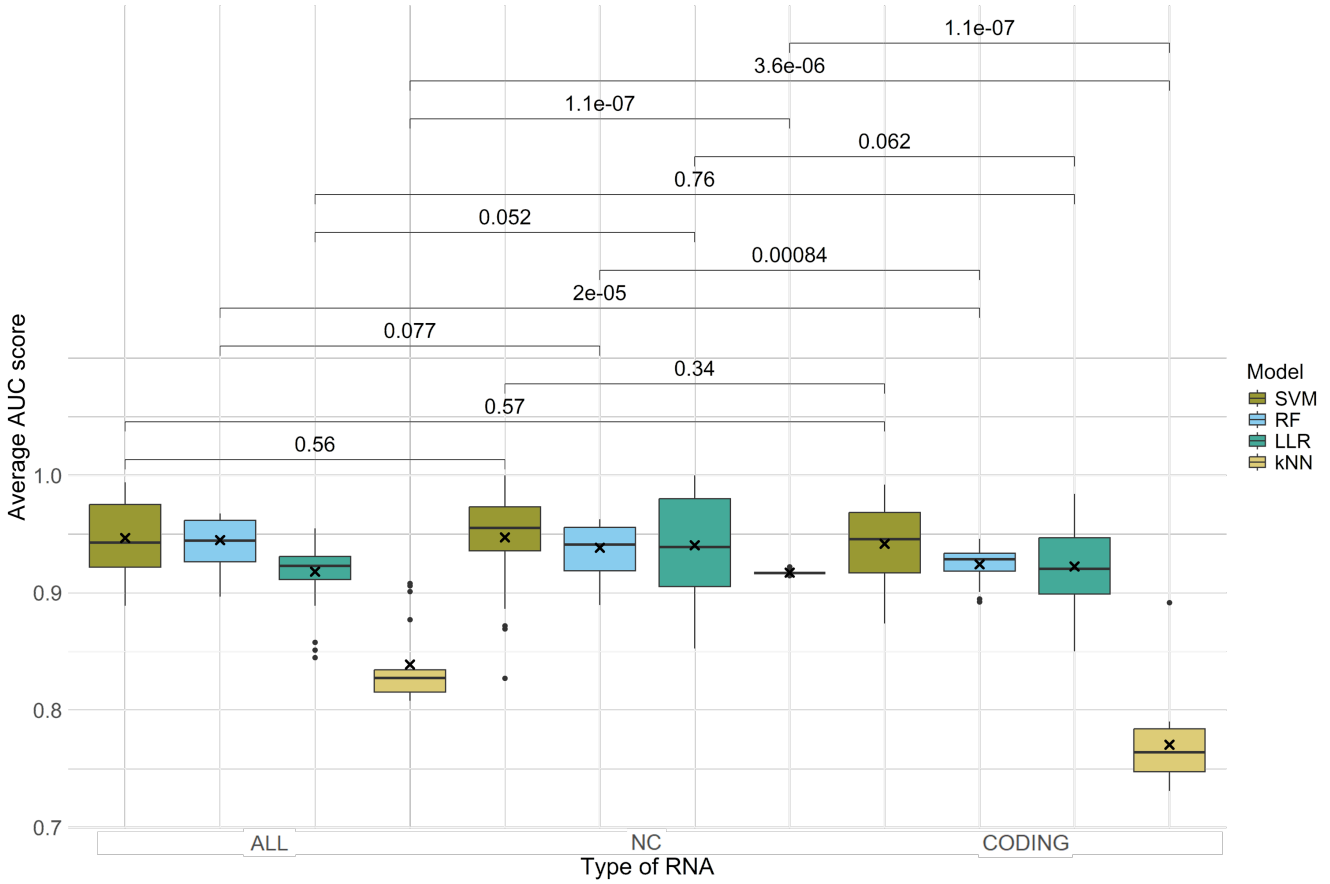


Figure 12: **Performance per type of RNA per model** This figure shows the average performance of the machine learning models on the different datasets (ALL, NC, CODING). From left to right, the average AUC scores are 94.7%, 94.5%, 91.8%, 83.9%, 94.7%, 93.8%, 94.0%, 91.7%, 94.2%, 92.4%, 92.2%, and 77.1%. The associated p-values of the difference in AUC per model are given at the top of the boxplots. The y-axis starts at an AUC score of 0.7 due to scale considerations.

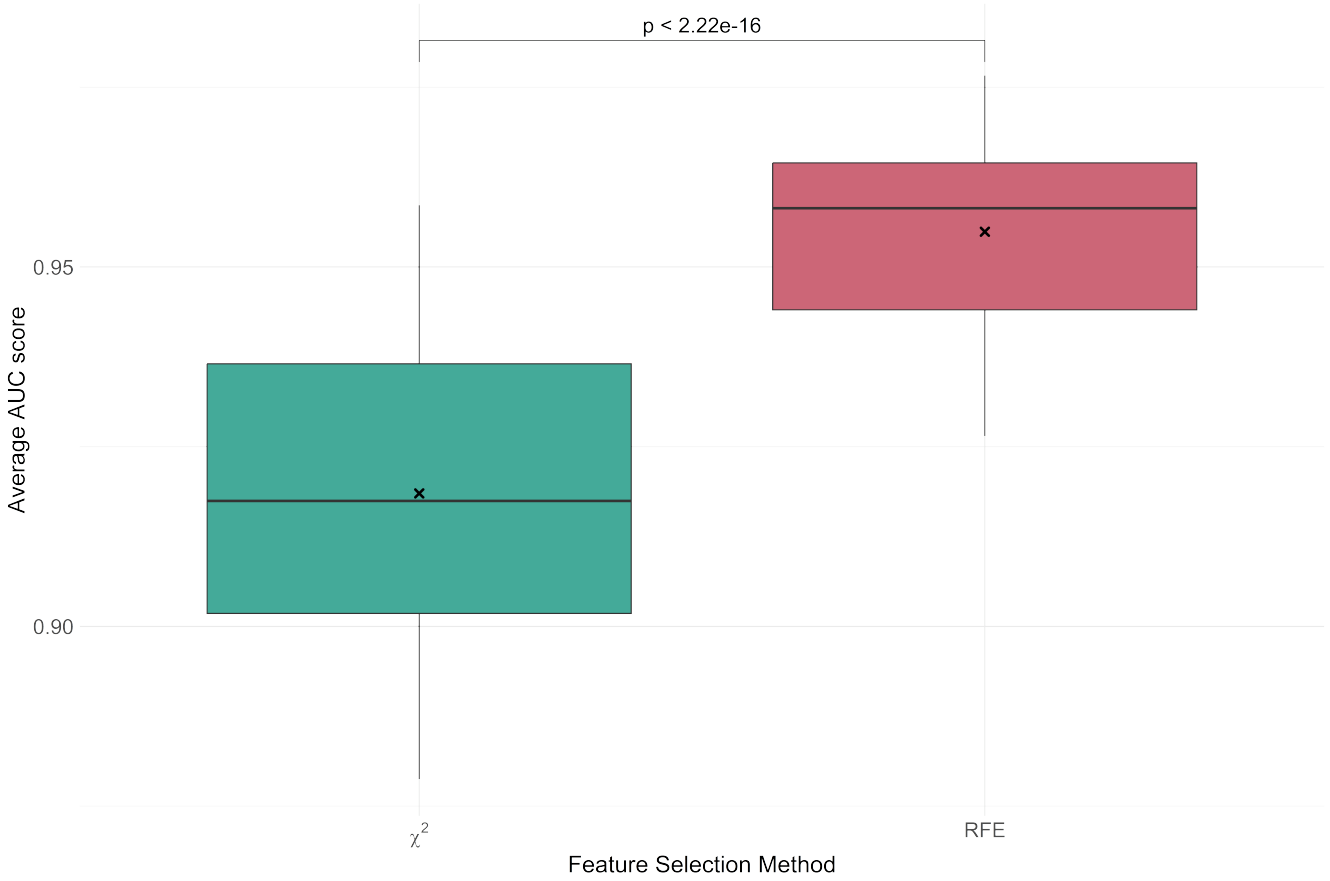


Figure 13: **Feature selector performance Random Forest (path A)** The average performance of the three datasets (ALL, NC, CODING) of the Chi-Squared and RFE feature selectors for Random Forest with each DGE method, the intersection and union of the DGE method is shown in this figure. From left to right, the average AUCs are 91.9% and 95.5%. The associated p-values of the difference in AUC per model are given at the top of the boxplots.

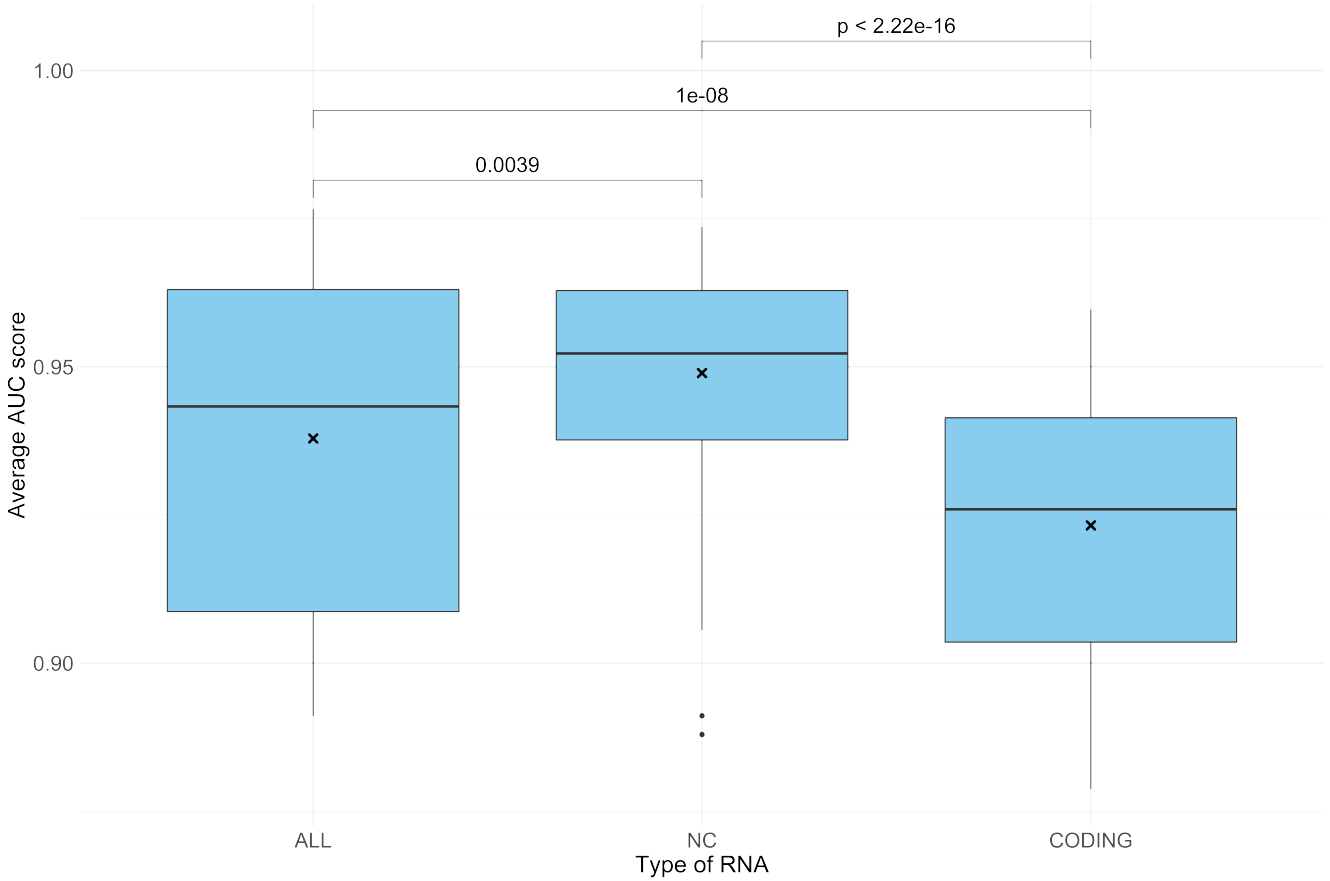
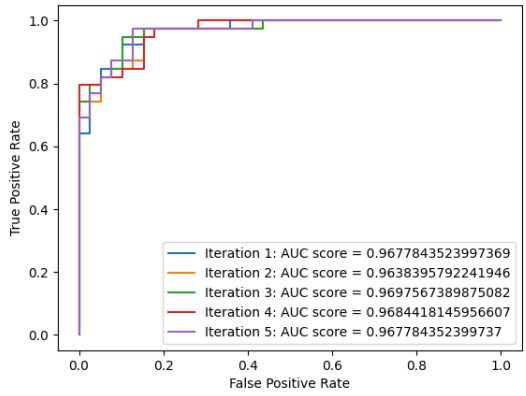
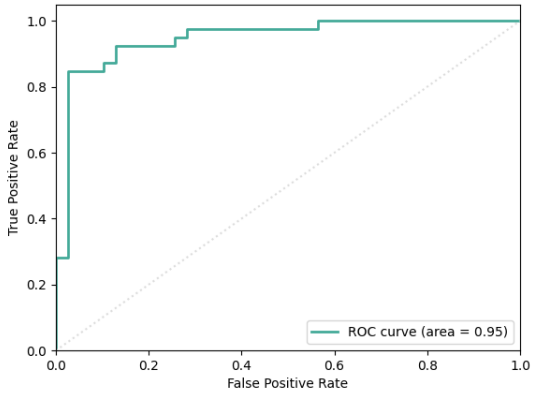


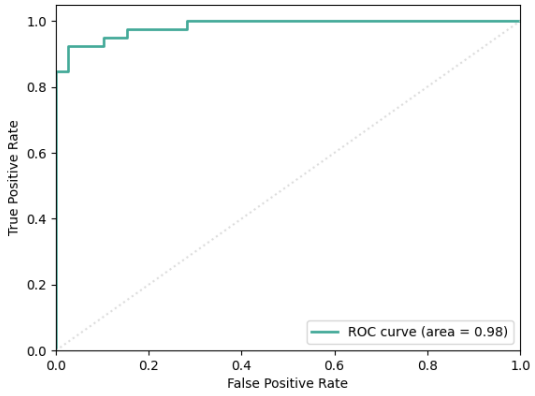
Figure 14: **Performance per type of RNA (Path A)** This figure shows the average performance of the Random Forest models trained on data from each DGE method, the intersection and union of the DGE methods on the different datasets (ALL, NC, CODING). From left to right, the average AUCs are 93.8%, 92.3% and 94.9%. The associated p-values of the difference in AUC per type of RNA are given at the top of the boxplots. The y-axis starts at an AUC score of 0.82 due to scale considerations.



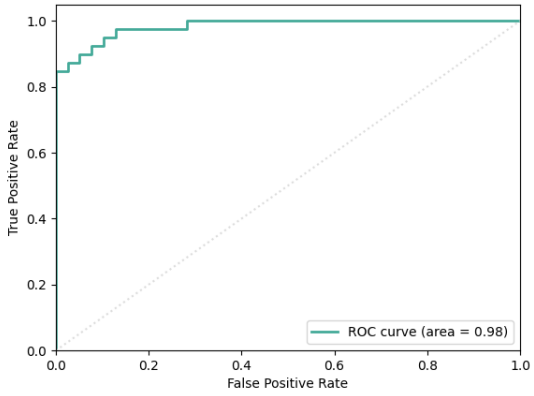
(a) TEA20LEPA



(b) TEA20LEPB



(c) TEA3LEP



(d) TEA10LEP

Figure 15: **ROC curve of the best combinations found by Path A and B** This figure shows the Receiver Operating Characteristic curves of the top 20 genes found by Path A and Path B, from which the top 3 and top 10 are derived.