Identification of a Novel Non-Small Cell Lung Cancer Transcriptomic Biomarker Using Supervised Learning Algorithms

Joseph C. Pettinelli

A Special Project

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

In

Data Science

Department of Mathematical Sciences

Central Connecticut State University

New Britain, Connecticut

February 2025

Special Project Advisor: Dr. Darius M. Dziuda

Identification of a Novel Non-Small Cell Lung Cancer Transcriptomic Biomarker Using Supervised Learning Algorithms

Joseph C. Pettinelli

An Abstract of a Special Project

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

In

Data Science

Department of Mathematical Sciences

Central Connecticut State University

New Britain, Connecticut

February 2025

Special Project Advisor: Dr. Darius M. Dziuda

Abstract

Non-small cell lung cancer [NSCLC] is a subtype of lung cancer that encompasses various carcinomas such as squamous cell carcinoma [SCC], adenocarcinoma [AC], and large-cell carcinoma [LCC]. NSCLC cases comprise most lung cancer cases, with the minority represented by small-cell lung cancer [SCLC]. Both subtypes are some of the deadliest cancers, partly due to early-stage detection being so difficult because of their asymptomatic nature early on and because of their tendency for recurrence. With a poor prognosis being so common, it is important to identify new therapeutic targets that may provide a starting point for the development of a cure. A crucial step in this process may be to identify a highly predictive biomarker at the transcriptomic level. This project builds upon the promising research from the *Integrated molecular portrait of non-small cell lung cancers* (Lazar et al., 2013) study that characterized the molecular distinctions between AC and SCC. The 15-gene classifier derived through L1-penalized logistic regression in that study only classified between the two NSCLC subtypes and did not consider the normal lung state. This project also used supervised multivariate feature selection with the goal of identifying a highly predictive transcriptomic biomarker, but with a couple key differences. This project classified between all the NSCLC subtypes as a single class and the normal lung class instead of the AC and SCC classification. The other difference was that two learning algorithms, random forests [RF] and support vector machines [SVM], were used with recursive feature elimination [RFE]. The 13 gene transcript biomarker identified with RF-RFE only culminated in an ROC-AUC of 0.624 on the test data set, while the 2 gene transcript biomarker identified with SVM-RFE was able to return a slightly better ROC-AUC of 0.785 on the test data set.

Acknowledgements

I would like to thank Professor Darius Dziuda for providing excellent guidance and a wealth of knowledge throughout this project. His expertise and insights were super valuable to the completion of this project and also for showing me many of the possibilities in the field of bioinformatics. I would also like to thank Professors Daniel Larose, Gurbakhshash Singh, and Frederic Latour for being on my committee and providing thoughtful feedback and advice.

Table of Contents

Research Question

This project aimed to answer the following question:

- Will combining all samples of NSCLC subtypes in the data (AC, SCC, and LCC) into a single NSCLC class, allow for a small and highly predictive biomarker of NSCLC to be identified using supervised multivariate feature selection in which the other class is normal lung?

This question was of interest because these three subtypes of cancer arise from different cell types. The AC samples arise from the alveoli, the SCC samples arise from the squamous cells that line the bronchi, and LCC can arise from any cell type in the lung. This has led to the development of many NSCLC biomarkers tailored specifically to certain subtypes such as the biomarkers in study 1 (Lazar et al., 2013) or study 2 (Šutić et al., 2021). The reason for treating all subtypes as a single heterogeneous class was to search for a more encompassing predictor of NSCLC. Such a multivariate biomarker would not only represent gene expression patterns distinguishing NSCLC from normal but would also identify patterns that are common among all the NSCLC subtypes and the cell types.

Literature Review

Study 1 - Integrated molecular portrait of non-small cell lung cancers (Lazar et al., 2013).

The goal of this study was to find characteristics of AC and SCC that distinguish between the two using various genomic data such as copy-number alteration, mRNA, and microRNA data. Lazar et al. (2013) were able to identify 15 classifier genes: S100A7, PKD2L1, TNNC2, CSTA, FCGBP, SLC1A7, TP63, WDR66, TFPI2, NR0B1, TESC, APOC1, AKR1C1, XAGE1, and SPTB. These genes were identified using L1-penalized logistic regression that achieved a cross-validated AUC of 96%. This learning algorithm performed feature selection by shrinking the coefficients of less important features to zero, effectively removing them. A 10-gene biomarker containing a completely different group of genes (except TP63) was also identified in the study using a univariate approach. The univariate approach identified overexpressed genes by setting a threshold of $\log_2$(gene expression in tumor tissue / gene expression in normal tissue) > 0 to determine overexpressed genes, and then selected the top 10 most overexpressed genes. SPP1, CTHRC1, and GREM1 secreted genes are biomarkers that identified NSCLC as well as possible blood-based biomarkers. Also, the SPINK1, and BMP7 secreted genes further provided the ability to distinguish SCC from AC because the over expression of these two genes compared to normal tissue varied greatly between the two NSCLC types (SCC and AC). Non-secreted genes: KRT6A, TP63, LGALS7, GCNT3, and SPRR2D distinguished between normal tissue and tumor tissue, and most were also useful for distinguishing SCC from AC (except for GCNT3).

Study 2 – Diagnostic, Predictive, and Prognostic Biomarkers in Non-Small Cell Lung Cancer (NSCLC) Management (Šutić et al., 2021).

The authors stated that when it comes to diagnosing lung cancer, medical imaging and scans usually do not provide a clear result. Collecting tissue samples is the preferred method because morphological features could be distinguished when stained with hematoxylin and eosin. Even this method does not always yield reliable results though because the tumor may not be differentiated, or it does not have key phenotypic features. Immunohistochemistry [IHC] can be used in these cases, and tissue can also be saved for biomarker testing. With IHC, the best biomarkers to classify AC and SCC are Thyroid Transcription Factor 1 [TTF-1] and p40 respectively. Aside from TTF-1, Napsin A is the best biomarker for AC. Finding biomarkers for IHC requires tumor samples which can be invasive, so there are also blood based biomarkers. Some blood-based biomarkers are Cytokeratin 19 fragment (CYFRA 21-1), carcinoembryonic antigen (CEA), squamous cell carcinoma antigen (SCCA), and carbohydrate antigen 125 (CA125) (Šutić et al., 2021). These biomarkers are advantageous because blood samples are much less invasive to extract from a patient. In this case, they can be used for early detection. Similar to the blood-based biomarkers, miRNAs can also be detected in bodily fluids such as blood, serum, and urine. The miRNAs that are already included in biomarkers are, "… miR-106a, miR-125a-5p, miR-129-3p, miR-205, miR-21, miR-29b, miR-375 and miR-7" (Šutić et al., 2021). This biomarker was highly accurate when it came to differentiating NSCLC and SCLC. This study shows that biomarker discovery is very important in the oncology field because these highly predictive biomarkers can be used for multiple purposes such as diagnosis, prediction, and prognosis.

Study 3 – The role of PRMT5 in Immuno-Oncology (Abe et al., 2023).

Many of the studies mentioned in this project involved the role of protein arginine methyltransferases (PRMTs), especially PRMT5, in lung cancer. This study claims that overexpression of PRMT5 in lung cancer results in a more deadly cancer from larger tumor size, and metastasis. This study identified the reasons for this enhanced disease state by explaining PRMT5 functions in the human immune system. PRMTs are a family of enzymes that methylate arginine residues which means a methyl group is added to the residue on the target substrate. Specifically, "PRMTs transfer S-adenosylmethionine (SAM) to the guanidino nitrogen atoms of arginine, resulting in the formation of methylarginine" (Abe et al., 2023). There are multiple types of PRMTs that are classified depending on the methylation mechanism. The PRMT mentioned multiple times in this study, PRMT5, belongs to the type 2 category which, "…catalyze symmetric di-methylation through monomethyl arginine" (Abe et al., 2023). PRMT5 is found on the inside of cells, specifically in the cytoplasm and nucleus. PRMT5 does not act alone, as there are other proteins that are known to be PRMT5 interacting partners. According to the study, "…methylosome protein 50 (MEP50) / WD repeat 77 (WDR77) is known as a major binding partner" (Abe et al., 2023). With the addition of MEP50, methylation is increased compared to the methylation rate of only PRMT5 alone. Arginine methylation by PRMT5 occurs on histones H2, H3, and H4 and reduces transcription of the genes in these regions.

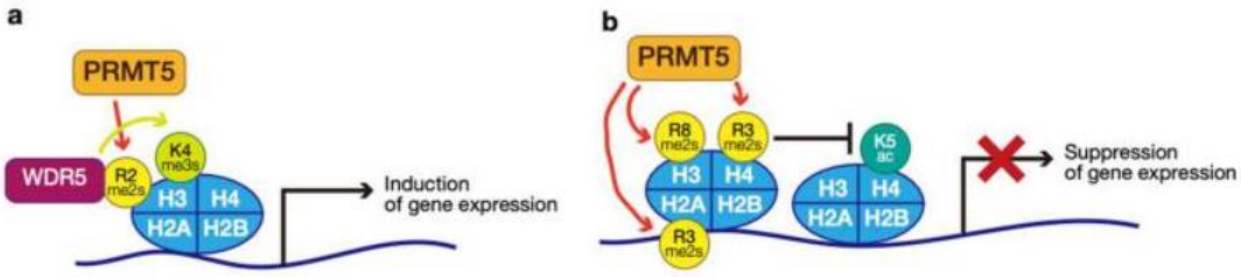

Figure 1 - PRMT mechanism (Abe et al., 2023).

In Figure 1a, it is shown that when PRMT5 methylates H3R2me2s, the WDR5 domain can recognize this methylation and then induces gene expression. Figure 1b shows a different scenario in which PRMT5 methylates H2AR3me2s, H3R8me2s, and H4R3me2s, and suppresses gene expression. Additionally, because mRNA splicing occurs in the cytoplasm of cells, the cytoplasmic PRMT5 plays a role in the regulation of this process. PRMT5 needs to methylate zinc finger 326 (ZNF326) to help with the accuracy of alternative splicing. These roles in gene expression and alternative splicing affect T cells and B cells in the immune system. When it comes to T cells, PRMT5 is needed for proper T cell proliferation. The need for PRMT5 is due to it promoting Interleukin 2 (IL-2) gene expression on CD4+ T cells and CD8+ T cells allowing these T cells to become activated. PRMT5 expression will also be upregulated when these same cells are stimulated by anti-CD3 and CD28 antibodies. The common cytokine receptor on IL-2, γc, and JAK3 are regulated by PRMT5, "… through precise mRNA splicing of those genes by PRMT5-mediated methylation of the Sm protein SmD3. Loss of PRMT5 can abolish T cell proliferation because of failed STAT5 activation" (Abe et al., 2023).

Study 4 – Genomic landscape of non-small-cell lung cancer with methylthioadenosine phosphorylate (MTAP) deficiency (Ashok Kumar et al., 2023).

The genomic loss of methylthioadenosine phosphorylate [MTAP] was found in 13% of NSCLC samples. MTAP deficient cells allow the substrate of MTAP, MTA, to accumulate, which is an inhibitor of protein arginine methyltransferase 5 [PRMT5]. Inhibiting PRMT5 is beneficial because overexpression of PRMT5 is known to aid in the progression of lung cancers because of its inhibitory effects on known tumor suppressors such as p53. This study mentioned that treating lung cancer with well-known drugs such as cisplatin and resveratrol in patients that are MTAP deficient, lead to more favorable outcomes in clinical trials. The studies *PRMT5 Selective Inhibitor Enhances Therapeutic Efficacy of Cisplatin in Lung Cancer Cells* by Bajbouj et al., (2021) and *PRMT5 Promotes Human Lung Cancer Cell Apoptosis via Akt/Gsk3 Signaling Induced by Resveratrol* by Li et al. (2019) claim that inhibiting PRMT5 can improve treatment of lung cancer with cisplatin and resveratrol respectively compared to the same treatment when PRMT5 is not also inhibited.

Study 5 - PRMT5 Promotes Human Lung Cancer Cell Apoptosis via Akt/Gsk3 Signaling Induced by Resveratrol (Li et al., 2019).

As prefaced in study 4, Li was able to prove that treating a PRMT5 knockdown cancer cell line with resveratrol significantly increased apoptosis as well as chemosensitivity. PRMT5 has previously been shown to be, "…involved in the regulation of chromatin remodeling, gene expression, cell cycle progression, cell proliferation, protein functions, and metabolism" (Li et al., 2019). It is also known to regulate tumor cell proliferation, so

when it is overexpressed, it is not surprising that it is linked to cancer. Cancer cell apoptosis was increased in resveratrol treated cells when PRMT5 was first inhibited by GSK591. This cell death was quantified with immunoblotting that identified apoptotic markers, cleaved caspase 3 and cleaved PARP. There was a large increase in these apoptotic markers when GSK591 was used.

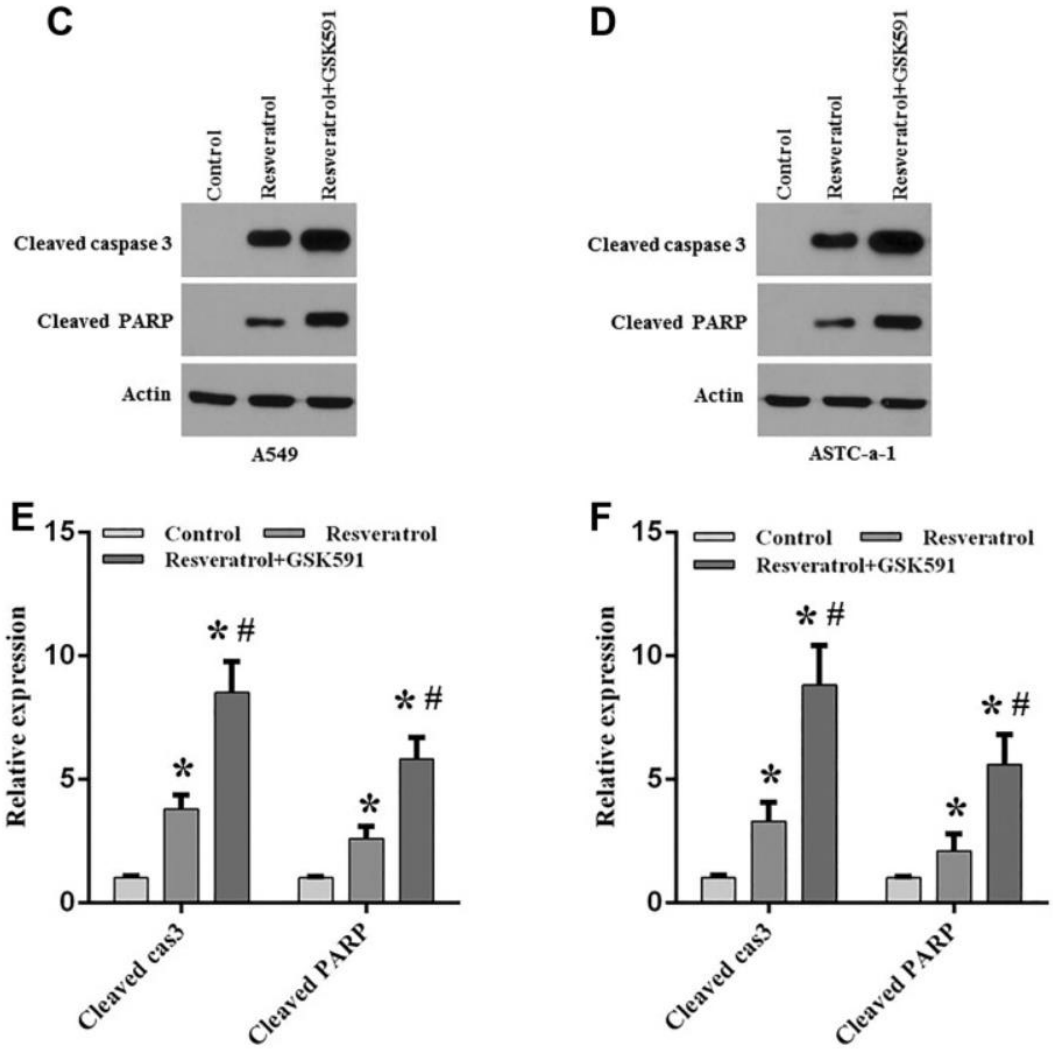

Figure 2 - Blocking PRMT5 promotes lung cancer cell apoptosis induced by resveratrol (Li et al., 2019).

Figure 2C and 2D show the immunoblotting results of two cell lines (A549 and ASTC-a-1) when treated with resveratrol and with resveratrol in conjunction with GSK591. Figure

2E and 5F show the quantified results of the immunoblotting. It is clear that apoptosis markers cleaved cas3 and cleaved PARP had their expression increased the most when cells were treated by a combination of resveratrol and GSK591, compared to treatment with resveratrol only or the control (no treatment). Similarly, when PRMT5 was inhibited with PRMT5-shRNA, lung cancer apoptosis was increased in the same two cell lines.

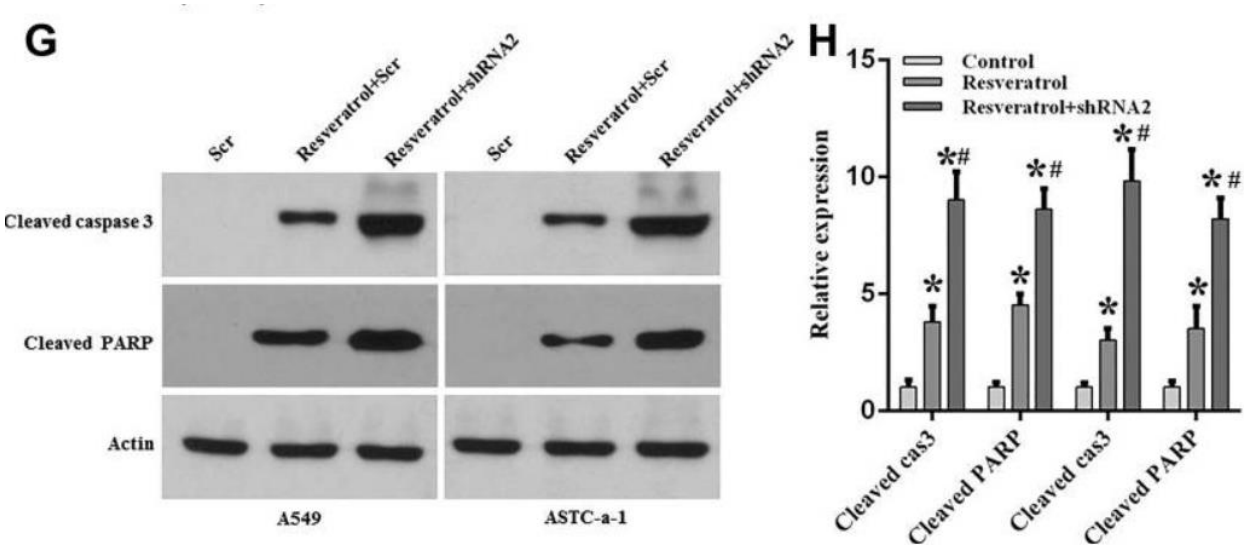

Figure 3 - Down-regulation of PRMT5 enhances lung cancer cell apoptosis induced by resveratrol (Li et al., 2019).

Treating A549 and ASTC-a-1 cells with resveratrol and shRNA increased cell apoptosis the most as shown by increased expression of the apoptotic markers (cleaved cas3 and cleaved PARP) in Figure 3G and 3H. This data helps to conclude that inhibiting PRMT5 can be further studied as a therapeutic for NSCLC because of its ability to increase apoptosis, cell death and chemosensitivity.

Study 6 - PRMT5 Selective Inhibitor Enhances Therapeutic Efficacy of Cisplatin in Lung Cancer Cells (Bajbouj et al., 2021).

Like the Li et al. (2019) study, this study aimed to inhibit PRMT5, but with arginine methyltransferase inhibitor 1 [AMI-1]. When this inhibitor and cisplatin treatment were combined, adenocarcinoma apoptosis was increased. The study was also able to show that this combination of AMI-1 and cisplatin did not affect the survival of normal epithelial cells, further validating AMI-1 as a potential therapeutic target to treat NSCLC. They performed this study in vitro with a lung cancer model compared to normal bronchial epithelial cells.

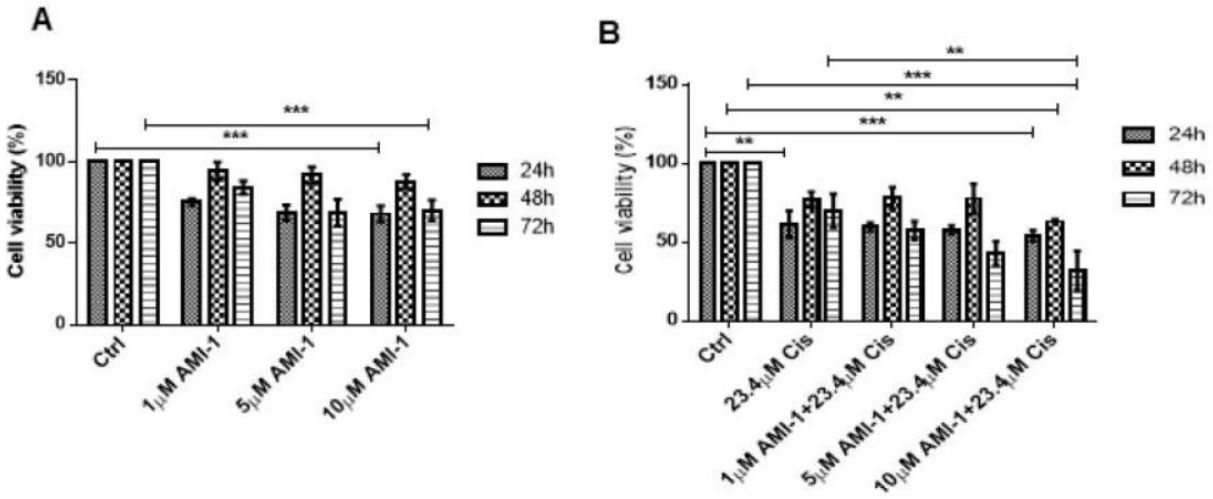

Figure 4 - AMI-1 alone and in combination with cisplatin reduces the viability of A549 cells (Bajbouj et al., 2021).

Figure 4A shows that treatment of A549 cells with the inhibitor AMI-1at concentrations of 1 µM, 5 µM, and 10 µM for 24 hours, 48 hours, and 72 hours, all decreased A549 cell viability compared the control. Moreover, Figure 4B shows that cisplatin alone decreased A549 cell viability at 24 hours, 48 hours, and 72 hours compared to the control. The most important finding of this study was the combined effect of cisplatin and AMI-1 on cell

viability that is also shown in Figure 4B. After 72 hours of treating A549 cells with cisplatin and AMI-1, cell viability was decreased the most compared to the control.

Statement of Purpose

The figures in this section had been updated as recently as June 2024, but the data used was only as recent as 2021. Figure 5 shows the 5-year survival rate for patients with each cancer type in the United States. Each bar on the plot represents patients with the same cancer type, while the x-axis of the plot is the percentage of these cancer patients that survive 5 years after diagnosis.

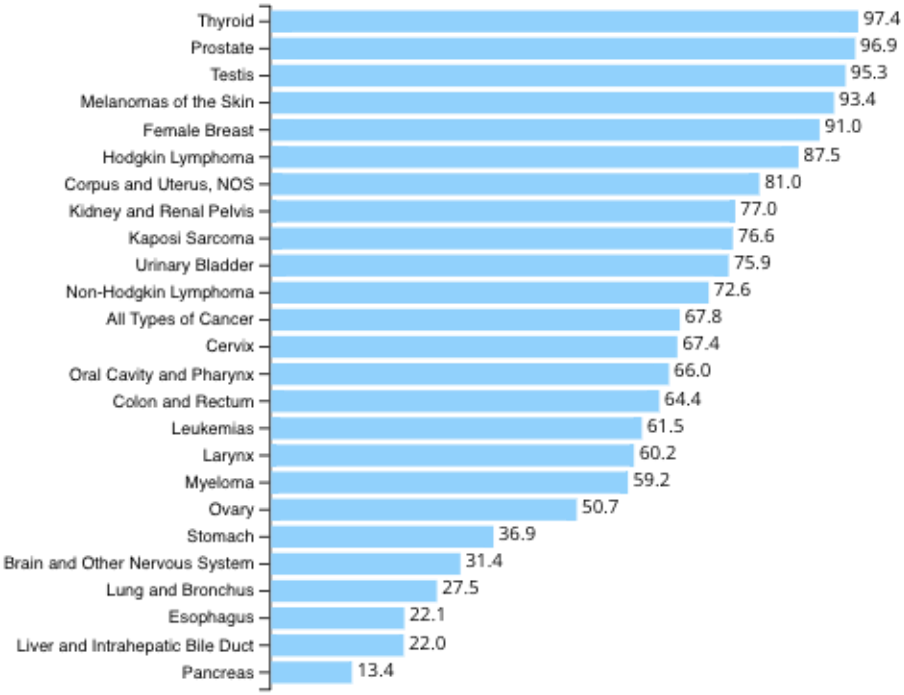

Figure 5 - U.S. Cancer Statistics Working Group. U.S. Cancer Statistics Data Visualizations Tool. U.S. Department of Health and Human Services, Centers for Disease Control and Prevention and National Cancer Institute; released in June 2024.

This shows that the 5-year survival rate for lung and bronchus type cancers is only 27.5% which places it in the top 5 most lethal cancer types for all patients. This rate also increases with older age demographics as shown in Figure 6 below. Each bar represents a population of patients with lung and bronchus cancer that are within a certain age range. The y-axis shows the survival rate of each age range.

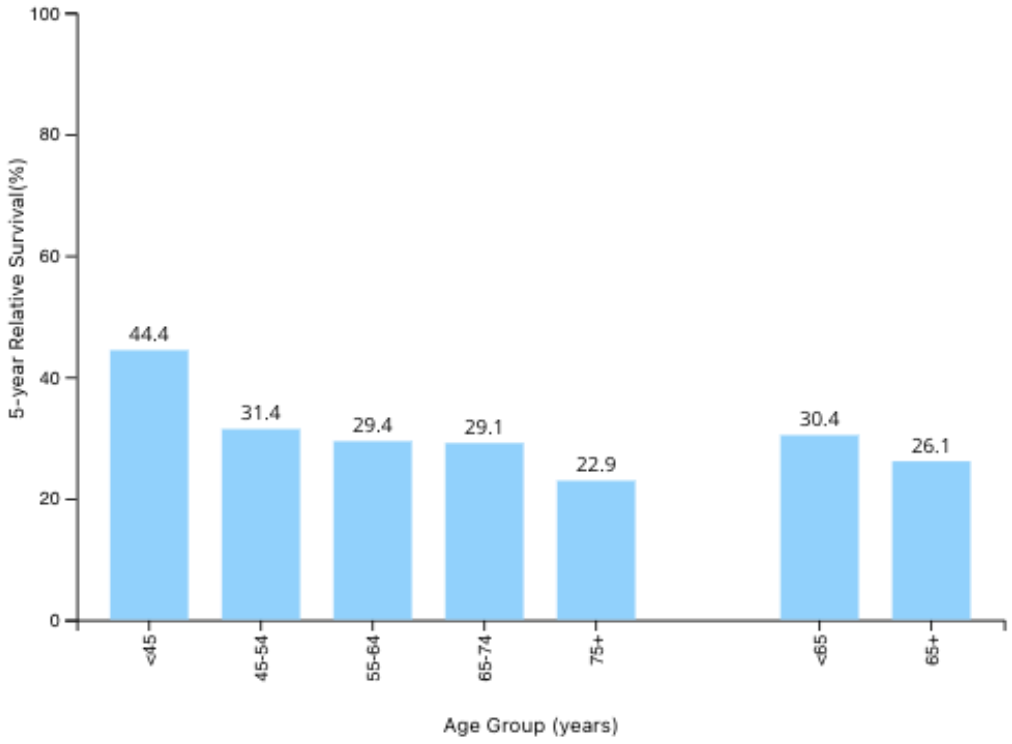

Figure 6 - U.S. Cancer Statistics Working Group. U.S. Cancer Statistics Data Visualizations Tool. U.S. Department of Health and Human Services, Centers for Disease Control and Prevention and National Cancer Institute; released in June 2024.

As age increases, the 5-year survival rate decreases. Also making the situation worse, the number of cases of lung and bronchus cancer types has been steadily increasing since 1999 possibly because of an aging population. This is shown by Figure 7 in which each bar represents the total number of lung or bronchus cancer diagnoses each year from

1999 to 2021. The numbers from 2020 and 2021 are affected by the COVID-19 pandemic because there were, "…delays and reductions in cancer screenings and diagnoses" (U.S. Cancer Statistics Working Group, 2024).

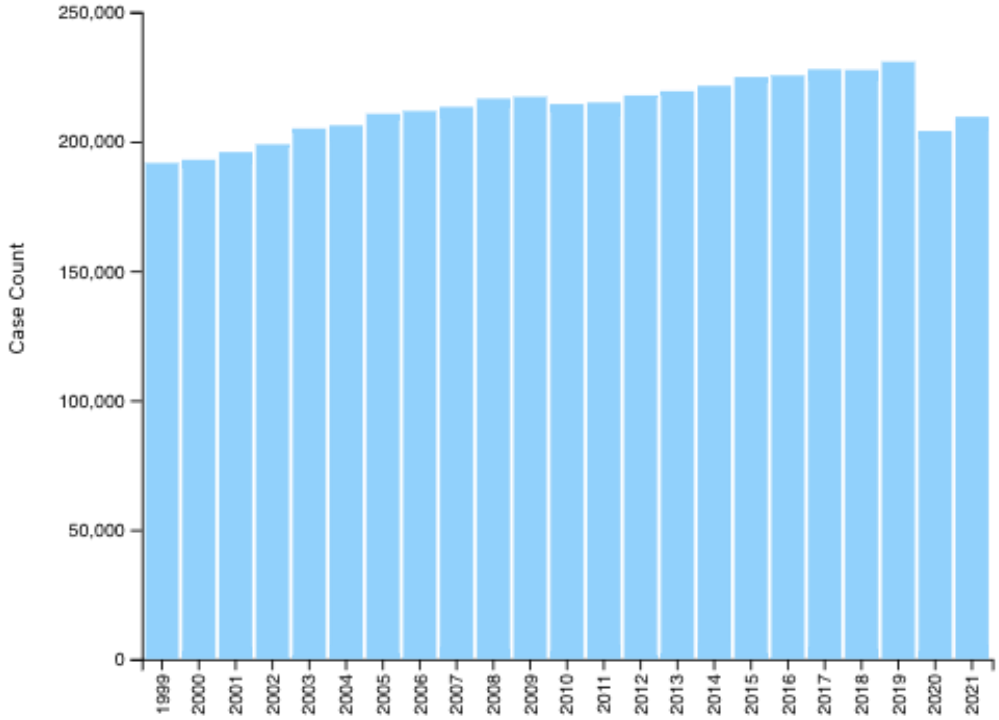

Figure 7 – U.S. Cancer Statistics Working Group. U.S. Cancer Statistics Data Visualizations Tool. U.S. Department of Health and Human Services, Centers for Disease Control and Prevention and National Cancer Institute; released in June 2024.

Based on the sheer number of cases, many of these diagnoses are bound to be NSCLC because it is the most common type of lung cancer. This reiterates the importance of identifying a more predictive biomarker that advances research on NSCLC to either learn how to prevent it or treat it.

Dataset and Data Preparation

Dataset Used

The dataset used in this project had been obtained from the ArrayExpress website under accession E-MTAB-1132 (Dessen, 2013; Lazar et al., 2013). It contained 123 paired human tumor and adjacent non-tumor samples (246 total) that were extracted and processed between 2002 and 2006. The population of samples included both males and females, with ages between 41 and 85 years, and stages of NSCLC ranging from 1 to 4. About half of the individuals were currently undergoing chemotherapy at the time of surgery. The class breakdown for the NSCLC tumor samples was 50 SCC, 57 AC, 13 LCC and 3 unclassified NSCLC, but this study only aimed to differentiate between the NSCLC class and normal lung class.

This experiment used a custom array design (Human Genome Exon 244K array) that incorporated the Whole Human Genome Microarray 4x44K from commercial design 014850 (Agilent Technologies, 2024). In addition to the gene transcript probes in the data, there were an additional 195,000 exon probes and 1,840 viral transcript probes. The exon-related and viral transcript-related variables were removed from this dataset, and only the overall transcriptional activity of genes was considered in this project. Each microarray contained 41,000 gene transcripts that corresponded to 20,007 unique genes that were listed in the Agilent gene list for design 014850 (Agilent Technologies, 2024). Each oligo probe used on the microarray was 60 nucleotides long. The fluorescence emitted from the labeled oligos was detected by a photomultiplier tube [PMT] that converted the light signal to an electrical signal. This signal was further processed into a digital measurement and written to a 20-bit Tagged Image File Format [TIFF] file. The

Agilent Feature Extraction software (version 10.5.1.1) was used with the default settings to process this raw data. For this project, the processed signal data was used.

According to the authors, the microarrays were made in both dual-color and with dyes swapped. This implies that for each patient, two microarrays were made. On the first microarray, the cDNA for the first condition was labeled with cy3, and the cDNA for the second condition was labeled with cy5. Then on the other microarray, cDNA for the first condition was labeled with cy5, and the cDNA for the second condition was labeled with cy3. The way that a single microarray was prepared can be visualized in Figure 8.

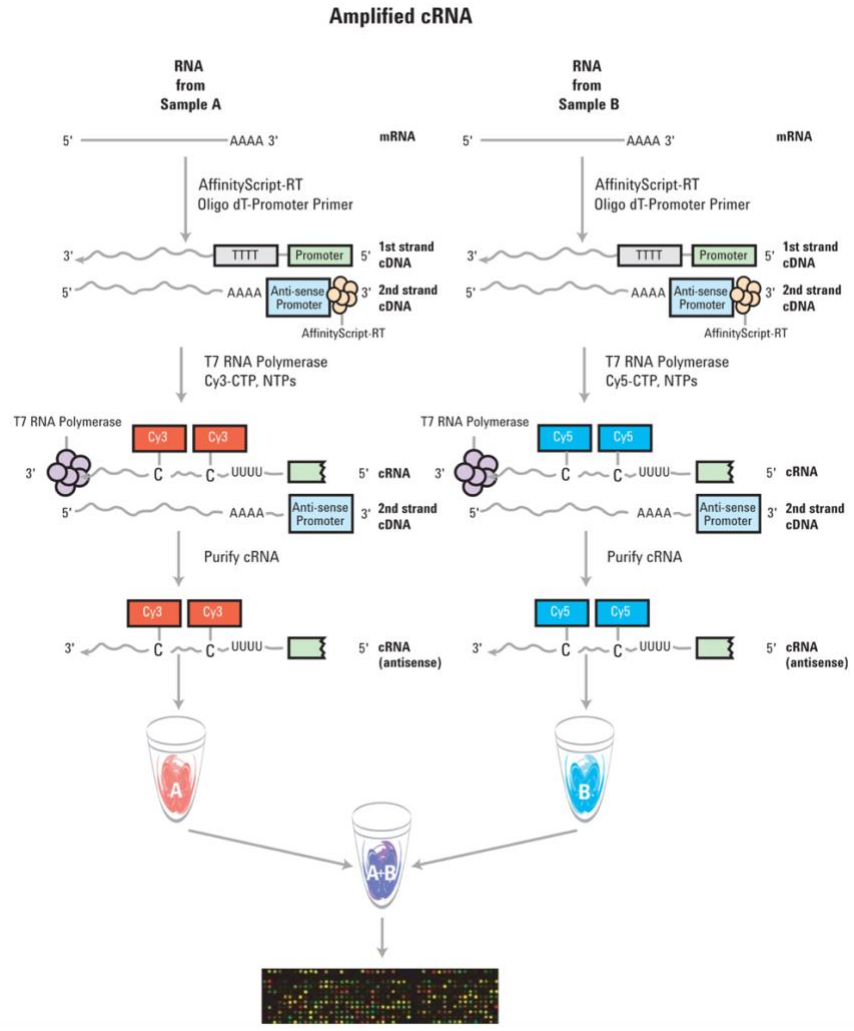

Figure 8 - cRNA extraction (Two-Color Microarray-Based Gene Expression Analysis, 2014).

Although this was most likely how the microarrays were made, it was unclear how the data was uploaded to ArrayExpress due to contradictions that were found in the metadata. Each downloadable file contained cy3 and cy5 signal data, but the associated metadata stated that the signal data for both channels were for the same condition. Confounding matters further, a unique microarray barcode was present in the metadata for each file. This implied that cDNA belonging to only one condition was hybridized on the same array, which is not what the authors stated. Because of the contradictions between the experiment procedure and the metadata, it could not be confirmed that Sample A and Sample B from the representation in Figure 8 were from different conditions. To guarantee that the data from both conditions was used correctly for a single patient, only the cy5 (green) signal data was used from both files for a single patient.

Preprocessing the Data

Agilent Feature Extraction Software

The downloaded data contained both the raw signal data and the corresponding processed signal data from Agilent Feature Extraction software of which the latter was used. Before explaining the preprocessing steps taken (or not taken) in this project, the steps taken by the Agilent Feature Extraction software to compute the processed signal data must first be explained.

Find Spots

Since the spots were arranged in a grid pattern, the find spots algorithm found the size and center of each spot (also known as the feature or probe set) on the microarray. Once

the spots were identified, the spot centroid was used as the feature, and the surrounding peripheral area was used as the local background area. This can be visualized in Figure 9:

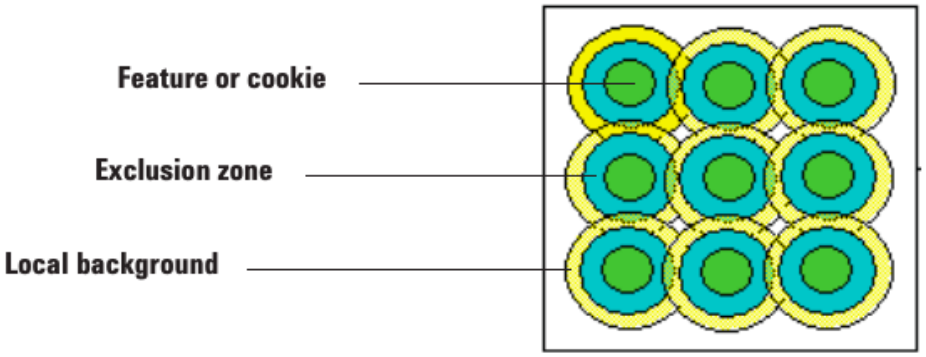

Figure 9 - Visual representation of spots on a microarray (Agilent Technologies, Inc., 2021).

The pixel outlier algorithm removed individual pixels from both the feature and background above the 75th percentile + (1.42 * IQR) and below the 25th percentile - (1.42 * IQR). They were removed so that they were not used in subsequent steps. The remaining inlier pixels in the feature and background area (~99% of original pixels) were used to calculate the median intensities which were the raw signal and raw background values. Additionally, the gIsSaturated and rIsSaturated columns were added in this step. These columns contained a Boolean flag (0 or 1) that represented whether the feature was saturated. The feature was considered saturated if more than half of the inlier pixels at this step were above the saturation threshold (~779,000 for green channel and ~778,000 for red channel).

Flag Outliers

This algorithm flagged individual features and local backgrounds (not individual pixels) in the form of a Boolean (0 or 1) in the output file. It only flagged them and did not

remove them. The columns gIsFeatureNonUnifOL, rIsFeatureNonUnifOL, gIsBGNonUnifOL and, rIsBGNonUnifOL were added to the output file during this step. A feature or local background that was flagged as a non-uniform outlier in this step had variance that was greater than the expected pixel variance plus the confidence interval multiplier (Agilent Technologies, Inc., 2021). The estimate of variance used in this equation depended on the microarray manufacture, wet lab chemistry, and scanner noise (Agilent Technologies, Inc., 2021). The local background population outliers, gIsBGPopOL and rIsBGPopOL were flagged as outliers if they were more than 1.42 * IQR away from the 25^{th} or 75^{th} percentile of local backgrounds (~1% of original local backgrounds in population).

Compute Background

This step was responsible for subtracting the background from each feature. This background was the local background identified in the Find Spots step and was subtracted from the corresponding feature. Additional adjustments such as spatial detrending and multiplicative detrending were also made to the background value to compensate for overestimates or underestimates of the local background level. gIsPosAndSignif and rIsPosAndSignif were added to the output file as a Boolean flag (0 or 1). A feature that was considered positive and significant, meant that the median feature signal was greater than background and deemed significant using a 2-sided t-test and p-value threshold of 0.01. More stringent metrics called gIsWellAboveBG and rIsWellAboveBG were also calculated at this step. These metrics had a Boolean value of

1 if IsPosAndSignif was true, and background subtracted signal was greater than 2.6 * the background standard deviation. Otherwise, the Boolean value was 0.

Correct Dye Biases

Lastly, since there was dye bias in both the green and red channels in a dual-color microarray, the algorithm applied a dye normalization factor to the features in each channel. A linear and global normalization of the data in each channel was performed before a LOWESS (locally weighted scatterplot smoothing) normalization was used. The dye normalized signal was calculated by multiplying the background subtracted signal * linear dye normalization factor * LOWESS dye normalization factor. The values resulting from this step were the gProcessedSignal and rProcessedSignal which were used for further processing in this project.

Preprocessing Steps

1. Each of the 246 downloaded files were parsed, and the columns gProcessedSignal, gIsWellAboveBG, gIsSaturated, and the four green channel outlier columns (mentioned in previous section) were extracted into their own file. The green channel data was chosen over the red channel data because there were only 149,301 positive outlier flags compared to the red channel's 182,825 positive outlier flags. Additionally, only the probes that corresponded to Agilent gene transcripts or the Agilent negative control probe (-3xSLv1) were retained. This left each file with 41,000 gene transcripts and ~600 negative control probe replicates. The negative control probe replicates were designed to not have any

specific binding, and if the microarray was perfectly processed by the Agilent Feature Extraction software, these negative control probes would have had signal equal to 0, which was not the case.

2. Before any further processing was done, if gIsSaturated or any outlier flags were positive, the corresponding gProcessedSignal and gIsWellAboveBG values were set to NA for that gene transcript. This is because any feature that was deemed to be saturated or was an outlier was not reliable enough to be included in any future calculations. These NA values were imputed in subsequent steps.

3. Since the gProcessedSignal of the negative control probes was greater than 0, it meant that some of the processed signal values still represented background noise. The remaining signal level of the negative control probes needed to be subtracted from all other gene transcript signal levels. To do this, the median gProcessedSignal of the ~600 negative control probes in each microarray were subtracted from the gProcessedSignal of the gene transcripts in the same microarray. The median signal for the negative control probes was minimal (only ~3.2 across all microarrays). Additionally, the ~600 negative control probes were removed in this step, so only the 41,000 gene transcript probes were left. Any gene transcript signal values that became negative were set to zero, as they represented no gene expression.

4. Until this step, all preprocessing had been done for single files separately and within single microarrays. In this step, normalization was done with all microarrays to make them comparable. The need for this was explained in a study done by Chain et al. (2010) on 77 Agilent data sets that found, "…interarray variability from replicate array measurements has a standard deviation (SD) of around 0.5 $\log_2$ units (6% of mean)". This could be due to factors in microarray preparation such as hybridization conditions, cRNA extraction, and fluorescent dye labeling. To reduce this variability in the data set used in this project, the signal data from each of the 246 microarrays was first loaded into a single gene expression matrix, and the NA values that were introduced during step 2 were handled. If more than 25% of all signal values for any gene transcript were set to NA, then the gene transcript was removed from the gene expression matrix. This resulted in 17 gene transcripts being removed. The other NA values in the gene expression matrix were imputed using the median of the other signal values belonging to the same gene transcript. This resulted in 147,986 NA values being imputed. The total number of NA values, 149,301, represented only 1.5% of the total gene expression matrix. The data was then $\log_2$ transformed.

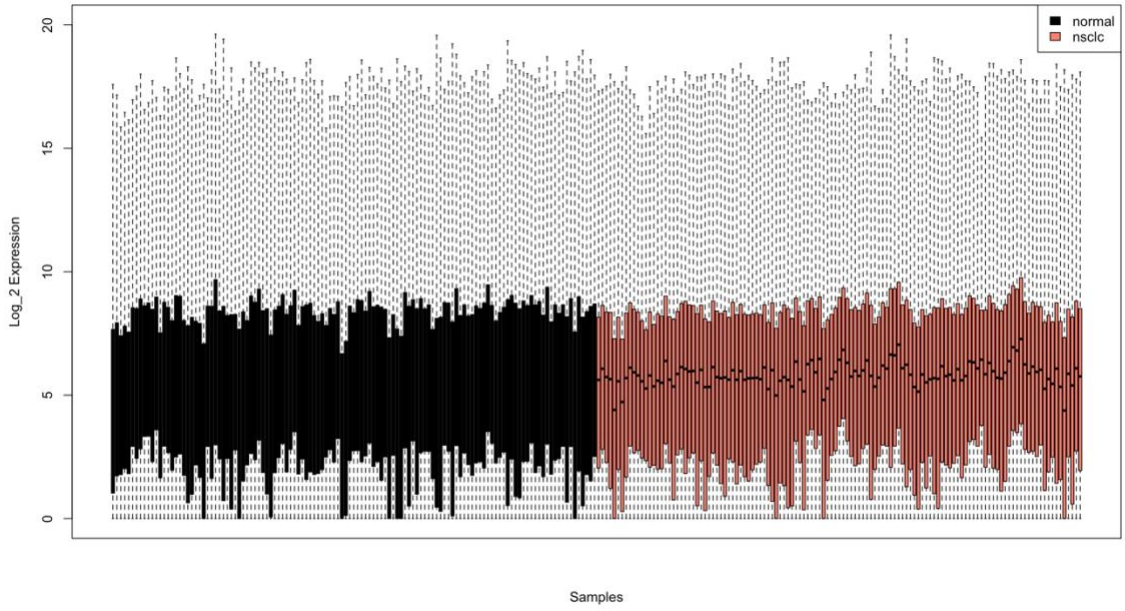

Figure 10 - Boxplot before between-array normalization.

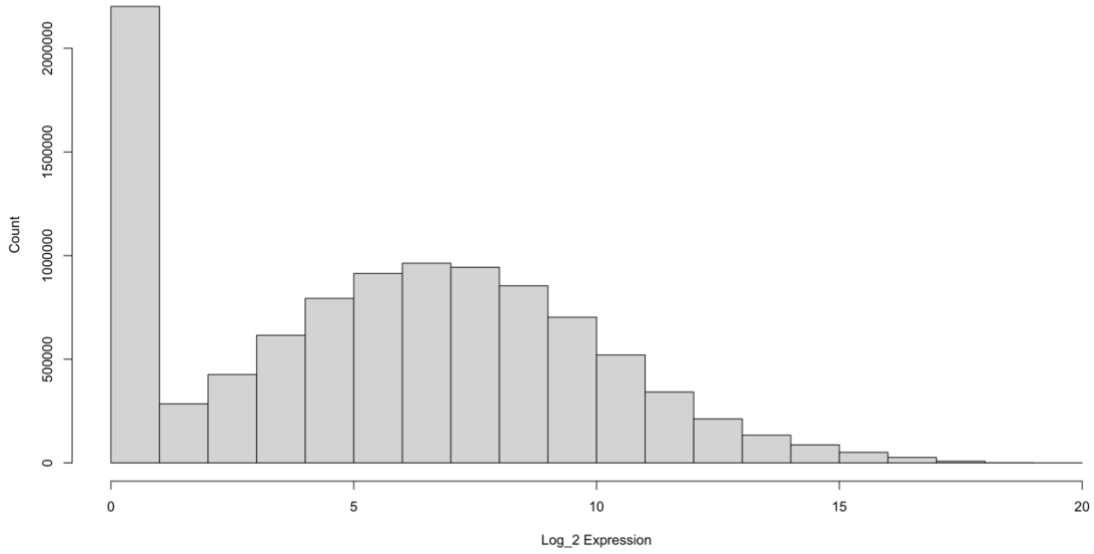

Figure 11 - Histogram before between-array normalization.

Table 1 - Statistics of data before between-array normalization.

Min	Max	Mean	SD	Q1	Median	Q3	IQR
0.0	19.93	5.57	4.0	2.10	5.79	8.47	6.37

This showed that a large proportion of the values were close to zero. The normalization method used was quantile normalization which is common in gene expression data. This normalization method prevented negative values that would have been introduced if other methods like cyclic loess normalization were used. The normalizeBetweenArrays function from the limma package (Ritchie et al., 2015) was used with the method parameter set to “quantile”. The boxplot of the normalized data is shown below.

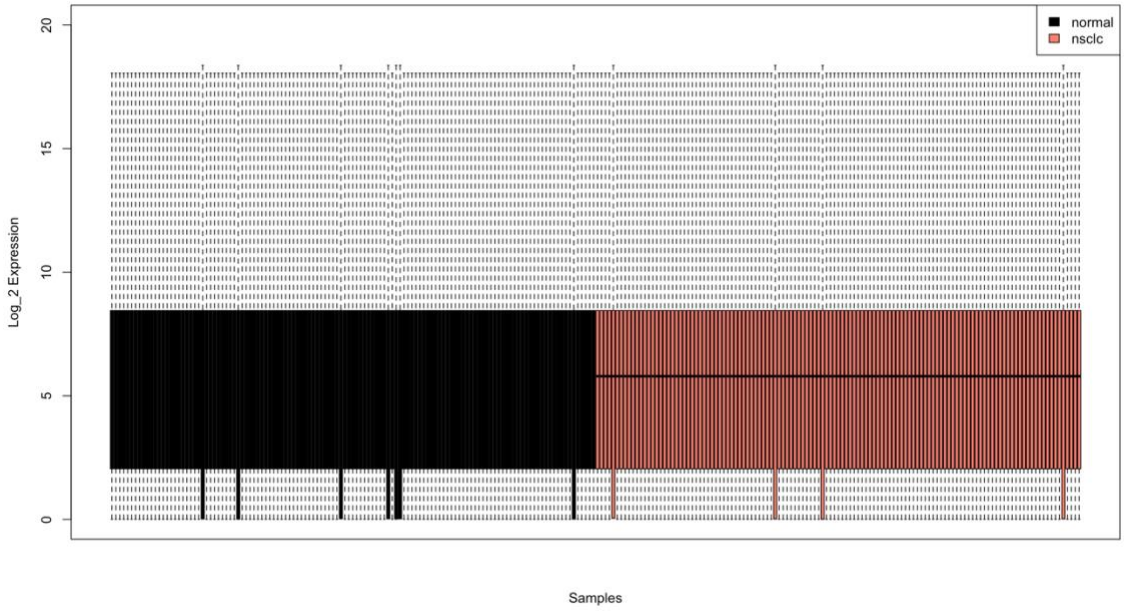

Figure 12 - Boxplot after between-array normalization.

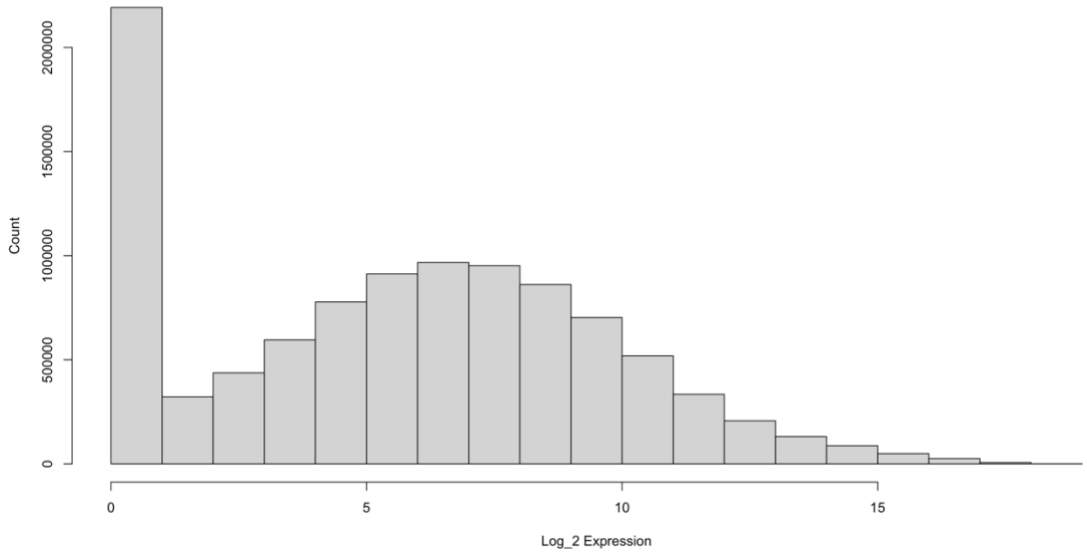

Figure 13 - Histogram after between-array normalization.

Table 2 - Statistics of data after between-array normalization.

Min	Max	Mean	SD	Q1	Median	Q3	IQR
0.0	18.36	5.55	3.98	2.02	5.79	8.45	6.43

5. Then gene transcripts that represented biological noise needed to be filtered out. The IsWellAboveBG data for each microarray was loaded into a separate data frame with class labels (NSCLC or normal). Like the gene matrix, this data also contained the NA values that were added in step 2, and they were handled using the same logic. Gene transcripts that were not IsWellAboveBG in at least 25% of samples of at least one class were removed. The IsWellAboveBG percentage was calculated as the sum of all IsWellAboveBG values for any given gene transcript across all samples in a single class divided by the number of samples in that class

(then multiplied by 100). This filtering method identified 10,212 gene transcripts to filter out, but they were not removed yet.

6. The low amplitude gene transcripts needed to be removed as well. The $\log_2$ transformed and normalized gene expression matrix was first un-logged for this. The 4% trimmed mean was calculated for each of the 246 samples separately.

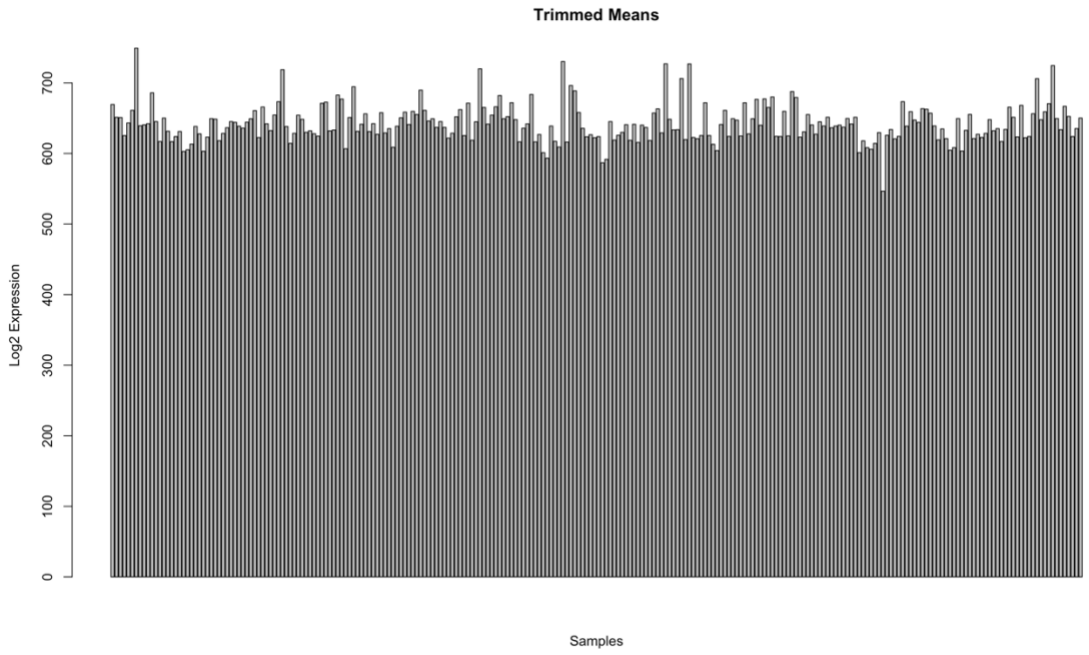

Figure 14 - Bar plot of 4% trimmed means.

Then the mean of all trimmed means (642) was rounded to 700 and used as the threshold noise level for filtering. This noise level was small relative to the range of signal values in the data (337,439). Any gene transcript with amplitude (max expression level – min expression level) less than 700 was considered low amplitude, and this included 14,543 gene transcripts. All filtering methods found a

total of 24,772 gene transcripts to remove, which left 16,228 gene transcripts. The gene expression matrix was then $\log_2$ transformed again.

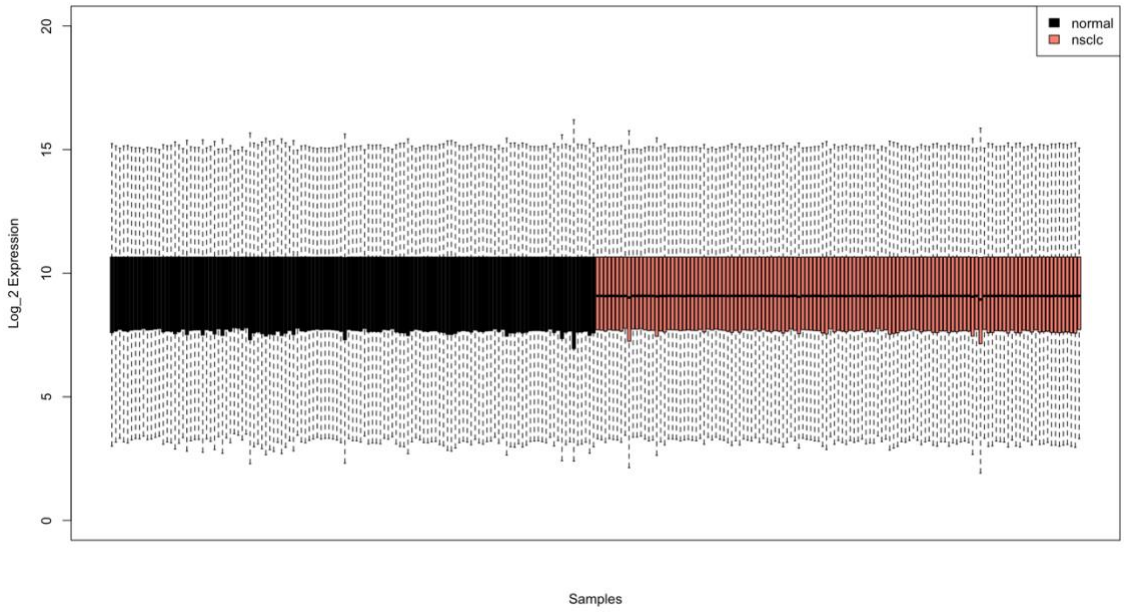

Figure 15 - Boxplot after gene transcript filtering.

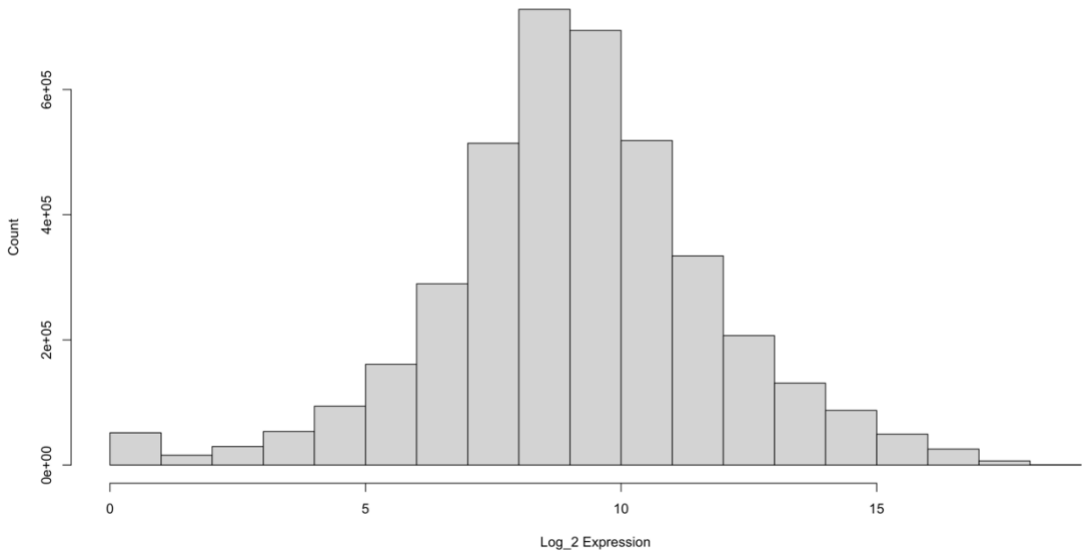

Figure 16 - Histogram after gene transcript filtering.

Table 3 - Statistics of data after probe filtering.

Min	Max	Mean	SD	Q1	Median	Q3	IQR
0.0	18.36	9.10	2.73	7.65	9.08	10.65	3.01

These preprocessing steps had the effect of decreasing both the intra-array variability and the inter-array variability. The SD decreased by 32%, and the IQR decreased by 53%. Additionally, the median increased by 57% because so many gene transcripts with low expression were removed. A more comprehensive description of the filtered data per class is in the Exploratory Data Analysis section.

Methods

Random Forests

The random forests learning algorithm is a supervised method that uses bagging (bootstrap aggregating) to create an ensemble of many decision trees, in order to achieve a generalizable classification model. This explanation will apply to the creation of only one of the many trees of the ensemble. For a single tree, random sampling with replacement, known as bootstrapping, is used to select a random subset of samples from the training data that is of the same size as the original training data. Because the sampling is done with replacement, some samples can be chosen multiple times, and some may not be chosen at all, for a particular bootstrap resample. Then a decision tree is created using the samples in that bootstrap. At each decision node in the tree, a random subset of variables ($\sqrt{number\ of\ variables}$ is a common size for classification) is

selected without replacement to be considered for the split. This extra layer of randomness added by selecting only some of the variables at each node helps to prevent overfitting by decreasing the correlation between the trees in the ensemble. The tree will continue to grow and split until all the nodes are leaf nodes or a predefined maximum depth is reached (ex: 30 nodes). The performance of this tree is estimated by predicting the samples that were not selected into the bootstrapped subset of samples (known as the out-of-bag [OOB] samples). This process of generating a tree is repeated hundreds or even thousands of times to generate the ensemble / random forest classifier. The OOB estimates of performance from all trees in the forest are aggregated to provide performance measures for the forest. Although such aggregated estimates can be considered reliable estimates of the performance of the random forest model, it is still best practice to confirm the performance with a separate test data set (Dziuda, 2010). New observations are classified by a plurality vote of all the decision trees in the random forest.

Support Vector Machines

The SVM learning algorithm is a supervised learning algorithm well known for its ability in binary classification tasks. The fundamental principle behind SVM is to find the optimal hyperplane that maximizes the margin from itself to the nearest training data while also minimizing misclassification of the training data.

In instances where the training data has N samples and p variables and the data can be linearly separable by the classes of the response variable, hard-margin SVM are used. With hard-margin SVM, this p-dimensional data can be perfectly separated by a (p-

1)-dimensional hyperplane in the input space. Although there is an infinite number of such hyperplanes, the optimal separating hyperplane, or maximum margin classifier, will provide the best classification of unseen data because it will have the largest margin for error. A separating hyperplane is represented by the following equation (Dziuda, 2010):

$$\boldsymbol{w}^T\boldsymbol{x} + b = 0,$$

in which $\boldsymbol{w}$ and $\boldsymbol{x}$ are p-dimensional vectors. $\boldsymbol{w}$ is a weight vector that determines the orientation of the hyperplane, and $\boldsymbol{x}$ is a p-dimensional feature vector representing a single sample. $\boldsymbol{w}^T\boldsymbol{x}$ is the dot product between $\boldsymbol{w}$ and $\boldsymbol{x}$ (calculated by summing each product of corresponding elements). b is the offset of the hyperplane from the origin of the input space. Two parallel and equidistant support hyperplanes to the separating hyperplane are defined at the minimal Euclidean distance from the separating hyperplane and the training data of either class. The support hyperplanes form the boundaries of the margin, and the data points that lie on this margin boundary are known as the support vectors. There will be no data points within the margin boundaries given by the equations: $\boldsymbol{w}^T\boldsymbol{x} + b = +1$ and $\boldsymbol{w}^T\boldsymbol{x} + b = -1$ (Dziuda, 2010) because the separating hyperplane is the solution to the optimization problem (Dziuda, 2010):

$$\underset{\boldsymbol{w}, b}{minimize} \; \|\boldsymbol{w}\|^2$$

that is subject to the constraint:

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1,$$

in which y_i is the class label (+1 for samples in the positive class, or -1 for samples in the negative class). With $p > N$ data, the optimization problem should be changed to use a vector of non-negative Lagrange multipliers ($\boldsymbol{a}$) that are associated with samples instead of variables. Each element of this vector is a multiplier a_i $(i = 1, \ldots, N)$ that is associated

with one training sample. This can significantly reduce the number of variables in the optimization problem. Only the samples on the support hyperplanes have non-zero Lagrange multipliers, and they are the support vectors. These will be the only samples to affect the orientation of the optimal separating hyperplane given by the equation (Dziuda, 2010):

$$\boldsymbol{w} = \sum_{i=1}^{N} a_i y_i \boldsymbol{x}_i.$$

Equation 1- Weight vector in a linear SVM

Therefore, these support vectors will be the only samples used to make classification decisions. The offset must still be calculated to completely define the optimal separating hyperplane. This can be done by plugging any of the support vectors into the following equation (Dziuda, 2010):

$$b = y_i - \boldsymbol{w}^T \boldsymbol{x}_i,$$

and solving for *b*. The function to then classify a new sample $\boldsymbol{s}$, is (Dziuda, 2010):

$$f(\boldsymbol{s}) = \sum_{i=1}^{N} a_i y_i \boldsymbol{x}_i{}^T \boldsymbol{s} + b,$$

so that if $f(\boldsymbol{s}) > 0$, the sample is classified to the positive class, or if $f(\boldsymbol{s}) < 0$, the sample is classified to the negative class. The description of SVM provided thus far has only applied to linearly separable hard-margin SVM.

In more complex data, a hard-margin SVM cannot separate the data without any misclassification, so a soft-margin SVM will be used. The soft-margin SVM provides more flexibility by allowing some misclassifications. The optimal separating hyperplane will now be a compromise between maximizing the margin and minimizing the cost of misclassification (Dziuda, 2010). The optimization problem is updated to add additional parameters that allow for the needed flexibility. The problem is now (Dziuda, 2010):

$$\underset{\boldsymbol{w},b,v}{minimize}\ \|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} v_i,$$

subject to:

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 - v_i,$$

for all $\boldsymbol{x}_i$ where $i = 1, \ldots, N$. The variable *v* is a slack variable that is used to add a penalty cost to training data that violate the margin. The larger the violation of the margin, the larger the penalty cost and vice versa. There will be a penalty ($0 \leq v < 1$) if the sample is classified correctly. The penalty will only be 0 (no penalty) if the sample does not violate the margin. Samples that are classified incorrectly, meaning they are on the wrong side of the separating hyperplane, will have a penalty $v > 1$. Additionally, the penalty $v = 1$ occurs when the sample lies on the separating hyperplane and thus cannot be classified. The scalar *C* is a regularization parameter that controls the degree of penalization. Increasing *C* will decrease the size of the margin because there will be larger penalties for margin violations. The opposite is true when *C* is decreased. *C* also adds an upper-bound on the Lagrange multipliers a_i. When $0 < a_i < C$ the sample *i* lies on one of the support hyperplanes and is considered an unbounded support vector. Samples with $a_i = C$ are known as bounded support vectors and affect the solution too, which differs from that of the hard-margin SVM problem (Dziuda, 2010).

When the boundary between classes is nonlinear, neither the hard-margin or soft-margin SVM described previously will provide a boundary that separates the classes in the input space. The data in the input space needs to be mapped to a higher dimensional feature space. This allows the classes to be linearly separable in that feature space, although they are still not linearly separable in the original input space. The mapping of data to the feature space can be done implicitly through the use of the kernel trick.

Solving the dual optimization problem in the feature space then leads to the following classification function (Dziuda, 2010):

$$f(\boldsymbol{x}) = sign(\sum_{i=1}^{N} a_i y_i K(\boldsymbol{x}_\mathrm{i}, \boldsymbol{x}) + b) ,$$

in which the function *K* is the kernel function, and the arguments to this function are vectors in the input space. One of the most common kernels is the radial basis function kernel:

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}\right),$$

Equation 2 - Radial Basis Function kernel

where $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$ is the Euclidean distance between the vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Additionally, σ is a hyperparameter that can be tuned and controls the spread of the kernel. Smaller values of σ increase the chances of overfitting, while larger values of σ increase the chances of underfitting.

Recursive Feature Elimination

RFE is a multivariate feature selection wrapper that uses sequential backward selection. This means that the search process starts with all the original variables, then recursively eliminates the least important ones based on their multivariate importance. Because this algorithm acts as a wrapper, it can work with many learning algorithms including RF or SVM. Within each iteration of this algorithm, a model is trained by a learning algorithm using the remaining important variables, the model's performance is calculated, and then the multivariate importance of each variable is calculated again. The variables that are determined to be the least important are iteratively removed until a set number of variables is reached, and the recursion continues. Because datasets usually contain many

variables at the start of feature selection, many variables will be removed in the first few iterations of this algorithm, then fewer and fewer variables will be eliminated per iteration toward the end of the experiment. This algorithm continues until a stop criterion such as a predefined number of variables is left, or there are no more variables. This algorithm will be a good choice for a binary response variable because neither class will have a large degree of separation from the training data set. This can be a problem with a response variable containing more than two classes and some classes are more difficult to predict than others (Kuhn et al., 2013).

Parallel Feature Selection

Because the data set used in this project is p >> N, the curse of dimensionality must be handled. Overfitting will occur if the recursive feature elimination algorithm mentioned above is only performed once. Performing feature selection only one time will result in the learning algorithm, regardless of the type, "memorizing" the training data, and causing the model to be overfit and not perform well on unseen data. Therefore, an external bootstrap resampling can be used to select a subset of starting samples each time recursive feature elimination is performed to remove bias. Many of these bootstrap resamples and feature selection experiments will drastically increase the chances that the optimal subset of variables is generalizable to future data. Therefore, the RFE algorithm described above will be done in parallel with a different bootstrap resample each time. The optimal number of variables in the biomarker and the optimal variables for that number of variables will now be based on an aggregated measure of performance across

all parallel recursive feature elimination experiments. The pseudo code for this algorithm is described in Figure 17.

```
for each resampling iteration do
    Partition data into training and test/hold-back set via resampling
    Tune/train the model on the training set using all P predictors
    Calculate model performance
    Calculate variable importance or rankings
    for Each subset size S_i, i = 1 ... S do
        Keep the S_i most important variables
        [Optional] Pre-process the data
        Tune/train the model on the training set using S_i predictors
        Calculate model performance using the held-back samples
        [Optional] Recalculate the rankings for each predictor
    end
end
Calculate the performance profile over the S_i using the held-back samples
Determine the appropriate number of predictors
Determine the final ranks of each predictor
Fit the final model based on the optimal S_i using the original training set
```

Figure 17 - Backward Recursive Feature Selection with resampling (Kuhn et al., 2013).

Evaluation

When using classification algorithms, a variety of evaluation methods should be used to assess algorithm performance. In this project, accuracy, sensitivity, specificity, and Receiver Operating Characteristic – Area Under the Curve (ROC-AUC) will be used. Each classification made by a classification algorithm falls into one of the following four categories: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The counts of the classifications in these four categories can be used as input to

calculate the previously mentioned metrics. Accuracy is defined as the proportion of correct classifications and can be calculated as $\frac{\mathrm{TN}+\mathrm{TP}}{N}$ (Larose et al., 2015) where *N* is the total number of classifications made by the algorithm. Sensitivity can be calculated as $\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$ and represents the ability of the algorithm to classify a positive record correctly (Larose et al., 2015). Sensitivity can be especially meaningful when making as many correct positive classifications as possible is the goal, and it does not matter as much if true negative classifications are missed. On the contrary, specificity can be calculated as $\frac{\mathrm{TN}}{\mathrm{FP}+\mathrm{TN}}$ and represents the ability of the algorithm to classify a negative record correctly (Larose et al., 2015). Specificity can be especially meaningful when making as many correct negative classifications as possible is the goal, and it does not matter as much if true positive classifications are missed. A good algorithm will have acceptable levels of both sensitivity and specificity. For that reason, ROC-AUC will be used because it evaluates the tradeoff between the sensitivity and specificity at different confidence thresholds. The ROC curve is plotted in a 2D space in which "…the resulting true-positive rate (i.e., the sensitivity) and the false-positive rate (1 - specificity) are plotted against each other" (Kuhn et al., 2013). The x-axis represents the false positive rate (spanning from 0 to 1), and the y-axis represents the true positive rate (also spanning from 0 to 1). An AUC equal to 1 would indicate a perfect set of classifications that completely separate the two classes. An AUC of 0.50 would indicate that the algorithm is completely ineffective and performs only as well as making a random classification. These evaluation methods were used to assess both the RF and SVM models developed in this project to help determine the biomarker.

Exploratory Data Analysis

After Preprocessing the Data, the exploratory data analysis [EDA] step was done to find properties of the $\log_2$ transformed gene expression values of 246 samples and 16,228 gene transcripts. It was found that the minimum value was 0.0, the maximum value was 18.36, and the average value was about 9.10 for all samples. The table below shows the more detailed sample statistics split by class, and the boxplot can be seen in Figure 15 - Boxplot after gene transcript filtering.Figure 15 - Boxplot after gene transcript filtering.

Table 4 - Statistics of gene expression levels of all samples per class in log2 scale.

Class	Min	Max	Mean	SD	Q1	Median	Q3	IQR
Normal	0.00	18.36	9.10	2.74	7.64	9.08	10.65	3.01
NSCLC	0.00	18.36	9.11	2.73	7.65	9.08	10.65	3.00

The maximum values in both classes were slightly further from the median than the minimum values in both classes. The maximum value of 18.36 for both classes was 9.28 units away from the median of 9.08. The minimum value of 0.0 for both classes was only 9.08 units away from the median for both classes. The Q3 and Q1 distances to the median were more similar in both classes. The Q3 value of 10.65 for both classes was 1.57 units away from the median. This was about 0.57 standard deviations from the median. The Q1 values of 7.64 and 7.65 for the normal and NSCLC classes respectively, had distances to the median of about 1.44 units, which was only about 0.52 standard deviations. The mean value for both classes was about 0.02 units larger than the median value. The IQR for the normal and NSCLC classes were 3.01 and 3.0 respectively, which meant that the middle half of the values fell within ~16% of the total range of values for each class. Using only

the previous statistical measures, the data would appear to have a slight positive skew, but this was not the case. There were similar numbers of high outliers and low outliers in both classes, although there were significantly more low outliers. High outliers were defined as being 1.5 * IQR more than Q3, and there were 35,543 high outliers in the normal class and 36,161 high outliers in the NSCLC class. Using the same logic to determine low outliers (1.5 * IQR less than Q1) resulted in 51,522 low outliers in the normal class and 51,011 low outliers in the NSCLC class. This resulted in a slight negative skew for both classes of about -0.25, which is almost symmetrical. The inlier and outlier values for each sample were very similar to each other making the data comparable across samples. Additionally, the data was visualized using principal component analysis [PCA] and colored by diagnosis.

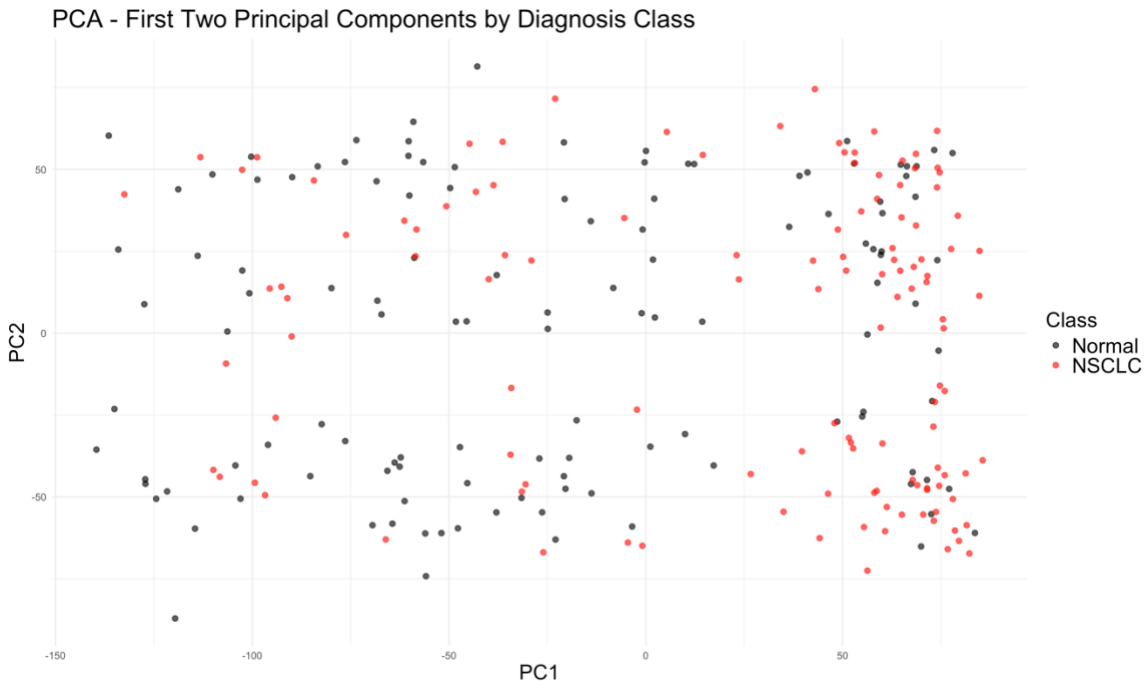

Figure 18 - Plot of data in the space of first two principal components colored by diagnosis.

This plot showed that the data from both classes were overlapping, although there is a higher density of normal samples on the left side of the plot and a higher density of NSCLC samples on the right side of the plot. This indicated that the unsupervised grouping of samples was not perfectly aligned with diagnosis. Because of the lack of separation in the space of the first two (and, later, three) principal components, it could not be guaranteed that the supervised learning algorithms planned for this project would achieve a highly accurate classification. Since the first two principal components were not sufficient to explain the variance in the data, PCA was also done in three dimensions with the first three principal components.

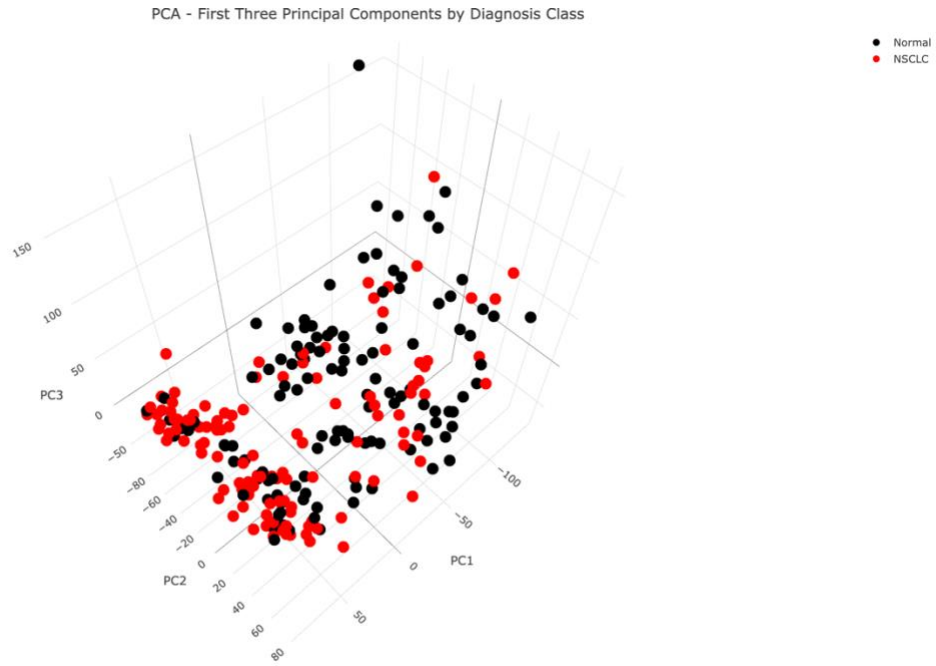

Figure 19 - Plot of data in the space of first three principal components colored by diagnosis.

This did not reveal much better separation between diagnosis. The statistics describing the first two principal components are shown below.

Table 5 - Summary of first three principal components.

	PC1	**PC2**	**PC3**
Standard Deviation	68.12	43.52	32.84
Proportion of Variance	0.24	0.10	0.06
Cumulative Variance	0.24	0.34	0.40
Most influential gene transcripts	A_23_P150316 A_23_P166360 A_23_P340698 A_23_P36901 A_23_P23783	A_24_P631948 A_24_P178154 A_24_P229807 A_24_P92267 A_24_P282597	A_24_P465799 A_24_P16004 A_23_P259763 A_24_P608268 A_23_P361654

As expected, even the first principal component did not explain that much variance in the data (only 24%). The second principal component explained significantly less variance (only 10%). It required 5 principal components to reach 50% cumulative variance, 32 principal components to reach 75% cumulative variance, and 96 principal components to reach 90% cumulative variance. For the sole purpose of exploring the data, additional PCA was done and colored by each diagnosis subtype (normal, NSCLC, SCC, AC, and LCC). The NSCLC subtypes were not well separated, as most subtypes overlapped in some way. This PCA was not shown because this project was not concerned with the individual NSCLC subtypes for supervised analysis.

Partition Data

The data was split into a training data set that contained 80% of the original data and a test data set that contained 20% of the original data. This split was used so that the training data set would contain a significant number of samples, while still leaving enough samples for testing. Since the original data samples were pairs of normal lung and NSCLC, the pairs were kept together when the data was split. The following chart shows the partition of the original data into the training and test set.

Table 6 - Partition of samples into the training and test data set.

	Normal Lung	**NSCLC**	**Total**
Training	98	98	196
Test	25	25	50
Total	123	123	246

Keeping the pairs together allowed each data set to contain the same exact number of normal lung samples and NSCLC samples. Then the training set and test set were further explored with the same basic statistics that were used when previously comparing the normal and NSCL classes.

Table 7 - Statistics describing all samples in the training and test data sets.

Class	**Min**	**Max**	**Mean**	**SD**	**Q1**	**Median**	**Q3**	**IQR**
Training	0.00	18.36	9.10	2.74	7.65	9.08	10.65	3.01
Test	0.00	18.36	9.11	2.72	7.66	9.08	10.65	3.00

The results for the training and test sets were very similar. Splitting the data into a training and test data set did not result in either data set having a drastically different

composition than the original full data set. The training data was used for all analysis, while the test data was used exclusively for evaluating the final biomarkers. A more detailed look at which samples were split into each class can be seen in the following tables.

Table 8 - Sample ID's partitioned into both classes from the training data.

NSCLC	Normal Lung
004, 005, 006, 007, 010, 012, 017, 026, 030, 035, 036, 040, 043, 048, 051, 053, 055, 056, 063, 065, 071, 072, 075, 077, 078, 081, 083, 086, 088, 091, 094, 095, 097, 099, 105, 107, 109, 112, 114, 115, 117, 120, 121, 123, 126, 129, 131, 144, 149, 151, 154, 156, 158, 159, 164, 167, 170, 176, 177, 180, 182, 183, 186, 188, 190, 194, 196, 225, 227, 231, 234, 236, 238, 239, 242, 243, 247, 249, 251, 255, 257, 261, 263, 266, 267, 269, 273, 274, 276, 278, 280, 282, 283, 288, 291, 293, 309, 313	003, 009, 013, 016, 024, 025, 027, 031, 034, 037, 038, 039, 041, 042, 044, 049, 050, 052, 054, 062, 064, 067, 070, 073, 074, 076, 079, 080, 082, 087, 092, 093, 096, 098, 100, 106, 108, 110, 111, 113, 116, 118, 119, 122, 124, 125, 132, 145, 150, 155, 157, 160, 161, 163, 165, 166, 169, 175, 178, 179, 181, 185, 189, 191, 195, 197, 221, 224, 226, 230, 232, 233, 235, 237, 240, 241, 246, 248, 250, 253, 254, 260, 262, 264, 265, 268, 270, 272, 275, 277, 279, 281, 284, 287, 289, 290, 292, 312

Table 9 - Sample ID's partitioned into both classes from the test data.

NSCLC	Normal Lung
002, 015, 018, 028, 032, 059, 061, 066, 069, 090, 101, 104, 147, 148, 162, 174, 193, 198, 222, 229, 245, 256, 286, 297, 311	001, 008, 014, 019, 029, 033, 057, 058, 060, 068, 089, 102, 103, 130, 146, 173, 184, 192, 223, 228, 244, 252, 285, 296, 310

Biomarker Discovery

Feature Selection

Feature selection is the most important step in biomarker discovery. The goal of feature selection is to identify a parsimonious subset of genes that maximizes the chances for generalization to the target population. To do this, supervised multivariate feature selection was performed on the training data by utilizing an RFE wrapper around random forests and SVM learning algorithms. The choice to use an RFE wrapper with many parallel experiments was due to the training data set containing only 196 samples and 16,228 gene transcripts that are mostly not important and noninformative for predicting NSCLC. Removing such genes was essential for maximizing the chance of identifying a parsimonious biomarker that would generalize well to unseen data. The rfe function from the caret package was used to implement the algorithm outlined in Figure 17 - Backward Recursive Feature Selection with resampling (Kuhn et al., 2013). Details of the steps that are common among all parallel RFE experiments, regardless of learning algorithm, are explained further.

Resampling

This project uses 100 parallel recursive feature elimination experiments, so 100 bootstrap resamples (with replacement) that have the same number of samples as the original training data set are created and used at each iteration.

Subset sizes

The number of variables to retain at each iteration of the internal recursive feature elimination were 16228, 8114, 4057, 2028, 1014, 507, 253, 126, 63, 31, 30, ..., 2,

and this was generated by the following algorithm:

1. Initialize vector $\boldsymbol{x}$ where the last element $\boldsymbol{x}_k$ is equal to P (the total number of variables in original training data set).
2. While $\boldsymbol{x}_k > 2$:
    a. If $\boldsymbol{x}_k > 32$:
        i. Append floor($\boldsymbol{x}_k$ / 2) to $\boldsymbol{x}$.
    b. Else:
        i. Append $\boldsymbol{x}_k$ - 1 to $\boldsymbol{x}$.
3. Return $\boldsymbol{x}$ with $\boldsymbol{x}_k = 2$.

Figure 20 - Pseudocode to generate feature subset sizes.

Prediction

The model was used to make predictions on the OOB samples and first predicted the diagnosis class (normal or NSCLC), and then also predicted the class probability (ranging from 0-1) for each class. The class probabilities were needed to calculate the ROC-AUC performance metric.

Summary

The summary function calculated ROC-AUC, accuracy, sensitivity, and specificity using the predicted classes and predicted probabilities. ROC-AUC was the metric used to assess the predictive performance of the model and was calculated with the following:

1. For each threshold T_i, where $i = 0, 1, \ldots, 1000$ and $T_i = i / 1000$:
    a. Calculate true-positive rate (TPR) using:
        i. True positive (TP) defined as NSCLC class probability >= T_i and the actual class is NSCLC.
        ii. ii. TPR = number of TP across all predictions / number of actual NSCLC.
    b. Calculate false-positive rate (FPR) using:
        i. False positive (FP) defined as NSCLC class probability >= T_i and the actual class is normal.
        ii. ii. FPR = number of FP across all predictions / number of actual normal.
2. Return all thresholds with their corresponding FPR, TPR in which these coordinates define the curve.

Figure 21 - Pseudocode to generate points of the ROC curve.

To calculate the area under the ROC curve (the integral) defined by these FPR, TPR points, the trapezoidal rule was used. This calculation is shown below:

1. $\boldsymbol{x}_i$ is the vector of FPRs such that: $\boldsymbol{x}_i = [FPR_1, FPR_2, \ldots, FPR_n]$.
2. $\boldsymbol{y}_i$ is the vector of TPRs such that: $\boldsymbol{y}_i = [TPR_1, TPR_2, \ldots, TPR_n]$.
3. $\boldsymbol{dx}_i$ is the vector of differences of consecutive elements of $\boldsymbol{x}_i$ and each $\boldsymbol{dx}_i$ represents the width of a trapezoid: $\boldsymbol{dx}_i = [x_2 - x_1, x_3 - x_2, \ldots, x_n - x_{n-1}]$.
4. $\boldsymbol{sy}_i$ is the vector of sums of consecutive elements of $\boldsymbol{y}_i$ and each $\boldsymbol{sy}_i$ represents the height of a trapezoid: $\boldsymbol{sy}_i = [y_1 + y_2, y_2 + y_3, \ldots, y_{n-1} + y_n]$.
5. $\mathbf{a}_i$ is the area of the i-th trapezoid and is calculated as: $\mathbf{a}_i = \boldsymbol{dx}_i * (\boldsymbol{sy}_i / 2)$.
6. The AUC is the sum of all n-1 trapezoids: $AUC = \mathbf{a}_1 + \mathbf{a}_2 + \ldots + \mathbf{a}_{n-1}$.
7. Return AUC.

Figure 22 - Calculation of AUC using trapezoid rule.

Select Size

The One Standard Error method was used to select the smallest subset size that performs within one standard error of best-performing subset size. The best-performing subset size was defined as the subset size that resulted in the largest average ROC-AUC across all 100 resamples of the same subset size. The standard error of this best-performing subset size was calculated as the standard deviation of the 100 ROC-AUC values divided by

$\sqrt{100}$. The returned subset size chosen by the one standard error method was the smallest one that had an average ROC-AUC within the one standard error of the maximum ROC-AUC (Dziuda, 2010). In this project, if the best-performing subset size was smaller than 20, the One Standard Error method was skipped, and the best-performing subset size was returned. Using that returned subset size as a starting point, the other metrics (accuracy, sensitivity, and specificity) were considered along with subset size to identify a subset size that performed relatively well in all metrics for its size. This was done to find a subset size that provides an optimal compromise between minimizing the size and maximizing the performance.

Select Variables

The number of variables returned matched the size, *S*, determined by the one standard error method. The individual variables comprising the *S* variables was determined by aggregating variable importance across all resamples. For each of the 100 resampling iterations, only the subset of *S* variables was retained. Then the frequency of each variable across all subsets of size *S* was calculated. These variables were sorted in descending order by their frequency, and then the top *S* variables were retained.

Random Forests – RFE

While the algorithms explained above were used for both versions of recursive feature elimination, the random forest algorithm had its own model fitting and variable ranking algorithms. The randomForest package (Liaw & Weiner, 2002) was used to fit the model. The number of variables tried at each decision node, mtry, was always set to the square root of the number of variables that still remained at the current iteration of recursive

feature elimination. The metric used to calculate variable importance was the mean decrease in accuracy. This was calculated as follows:

1. For each tree t_i where $i = 1, 2, ..., N$ where N is the number of trees in the forest (2000):
    a. $error_i$ = classification error rate on OOB data.
    b. For each predictor variable k, $k = 1, ..., p$:
        i. $error_k$ = classification error rate on OOB data with k permuted.
        ii. $decrease_k = error_i - error_k$.
2. MDA_k = (sum $decrease_k$ over all t_i) / N.
3. Sort all k in descending order by corresponding MDA_k.

Figure 23 - Calculation of variable mean decrease in accuracy.

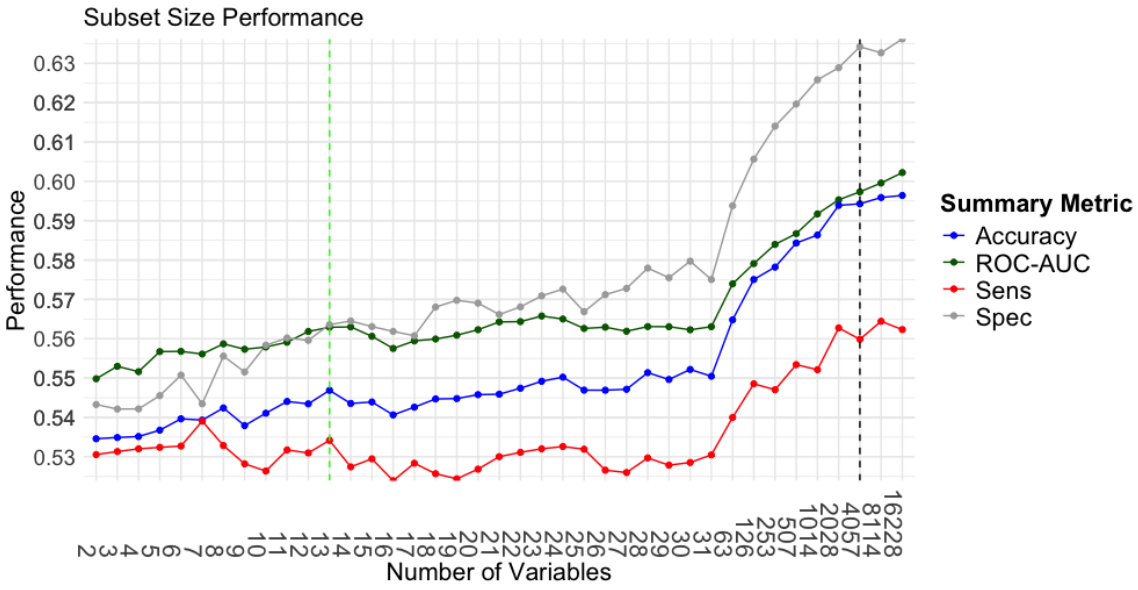

Figure 24 - RF-RFE performance summary. The black and vertical line is at the automatically chosen size, and the green and vertical line is at the manually chosen size.

The subset size selected automatically was 4057, which is far from parsimonious. The ROC-AUC at this subset size was only 0.597, while the maximum ROC-AUC when all features were included was only 0.602. At 4057 features, the accuracy was 0.594, the sensitivity was 0.560, and the specificity was 0.634. Because the subset size was so large, a smaller subset size still had to be chosen that would still locally maximize the

performance metrics. Based on the plot above, there were two local peaks in performance at subset sizes 24 and 13. Although the performance was slightly better with 24 features (0.37% increase in ROC-AUC, 0.60% increase in accuracy, 0.30% increase in sensitivity, and 1.57% decrease in specificity), 13 was chosen as the subset size because it is nearly half the size, making it much more parsimonious. The performance values with this subset size: ROC-AUC was 0.563, accuracy was 0.547, sensitivity was 0.534, and specificity was 0.564. At all subset sizes, the sensitivity was smaller than the specificity which meant the model was biased toward selecting the negative class (normal by default), even though there were equal numbers of both classes in the training data.

Table 10 - Description of gene transcripts selected for RF-RFE.

Probe ID	Gene Symbol	Gene Name
A_32_P122715	MICAL2	microtubule associated monooxygenase, calponin and LIM domain containing 2
A_24_P302374	CLCN6	chloride voltage-gated channel 6
A_23_P82370	HOXA6	homeobox A6
A_23_P108294	PLPP2	phospholipid phosphatase 2
A_23_P74042	ADGRL2	adhesion G protein-coupled receptor L2
A_24_P330366	RCOR1	REST corepressor 1
A_24_P228637	SETD4	SET domain containing 4
A_24_P85317	CHD2	chromodomain helicase DNA binding protein 2
A_23_P156497	ELOVL5	ELOVL fatty acid elongase 5
A_23_P322845	PLPP5	phospholipid phosphatase 5
A_23_P2661	RAP1B	RAP1B, member of RAS oncogene family
A_23_P205007	IPO5	importin 5
A_23_P203841	BAZ2A	bromodomain adjacent to zinc finger domain 2A

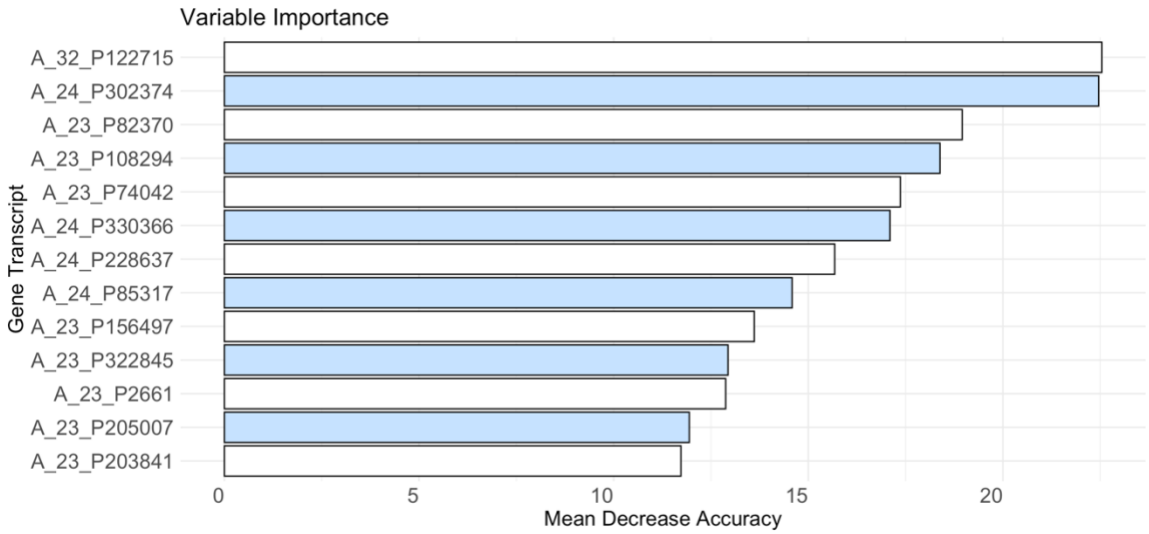

Figure 25 - Final random forests biomarker variable importance.

The most important gene transcript was A_32_P122715 followed by A_24_P302374 in close second. These two variables contributed to 22.54% and 22.46% of the biomarker's classification accuracy on the training OOB samples. The third most important gene transcript had an MDA of 18.95%, and the importance of the rest of the gene transcripts decreased at a pretty constant rate until A_23_P203841with an MDA of 11.73.

SVM – RFEs

Linear Kernel

This algorithm utilized the ksvm function from the kernlab package (Karatzoglou et al., 2004) to fit the model. The type parameter was set to 'C-svc' for classification, and the kernel parameter was set to "vanilladot" for linear. The metric used to calculate variable importance with the linear vanilladot kernel was the square of variable weights, in which a larger squared weight indicated a more important variable (Guyon et al., 2002). These weights were calculated as follows:

1. Calculate weight vector $\boldsymbol{w}$ with the equation - *Weight vector in a linear SVM*
2. The squared weight vector $\mathbf{w_{sq}}$ is calculated with $\boldsymbol{w}^2$.
3. Each element in $\mathbf{w_{sq}}$ represents the importance of a variable.
4. The variables are sorted in descending order by their corresponding importance.

Figure 26 - Calculation of variable squared weights.

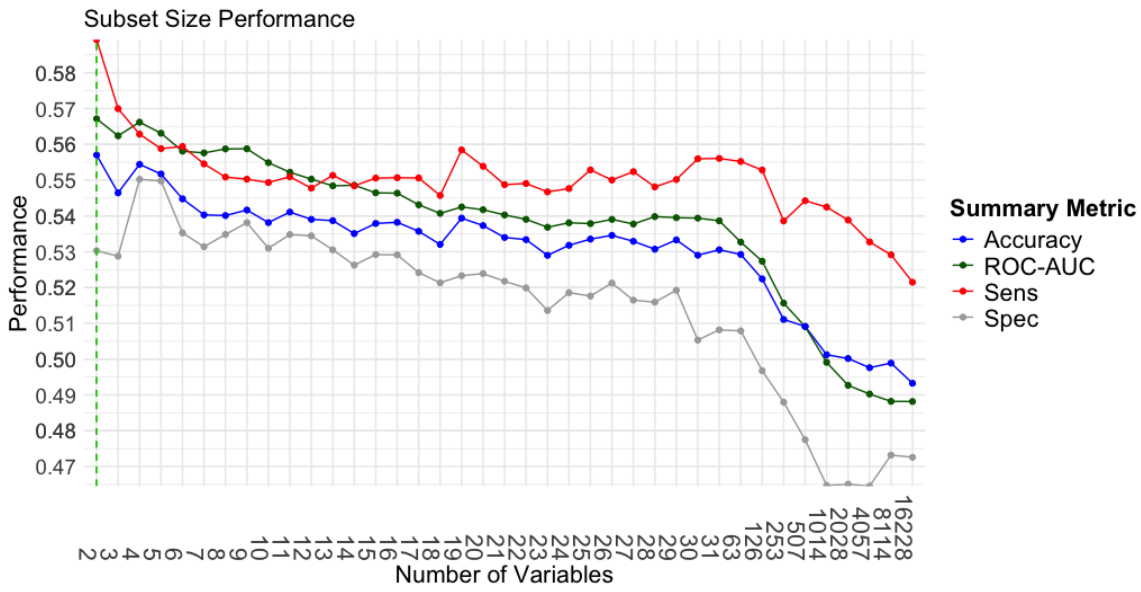

Figure 27 - SVM-RFE with linear kernel performance summary. The green and vertical line is at the automatically chosen size and the manually chosen size.

This feature selection experiment was able to select a very parsimonious biomarker. This biomarker only had two features, and this was without applying the One Standard Error method. However, the biomarker being parsimonious did not mean much since the ROC-AUC was only 0.567. The accuracy was 0.557, the sensitivity was 0.589, and the specificity was 0.530. Contrary to the RF-RFE experiment, as the subset size increased, the performance actually decreased. This indicates that the models could have been overfit on the training samples and were therefore not generalizable for the OOB samples. Because the subset size was already two, and the performance metrics were already relatively good compared to the other subset sizes, there was no need to select a

different subset size. The selected probe IDs are shown below, although Agilent did not have the complete annotations for the gene transcript associated with the A_23_P149798 probe ID. Neither of the probe IDs selected in this experiment were also selected in the RF-RFE experiment, which was not a surprise since different learning algorithms were expected to perform differently.

Table 11 - Description of gene transcripts selected by linear SVM-RFE.

Probe ID	Gene Symbol	Gene Name
A_23_P111583	CD36	CD36 molecule
A_23_P149798	N/A	N/A

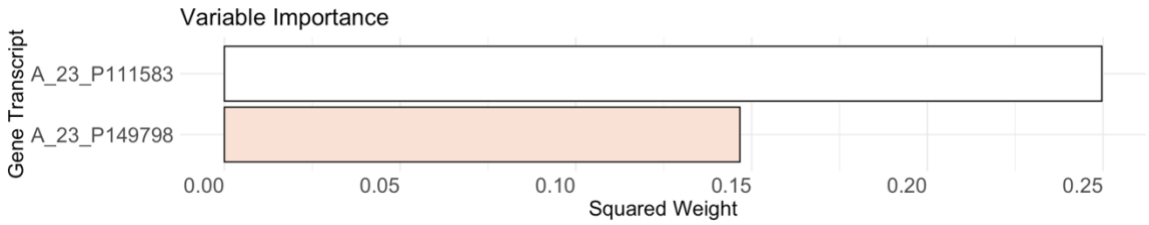

Figure 28 - Final linear SVM biomarker variable importance.

The first gene transcript, A_23_P111583, with a squared weight of 0.250 was much more important than A_23_P149798 with a squared weight of only 0.147.

Radial Basis Function (RBF) Kernel

Because the linear kernel did not separate the two classes well, a nonlinear kernel was also tried with the RFE experiments to determine if it would be worth pursuing. The difference in this implementation was that the e1071 package (Meyer et al., 2024) was used with the kernel parameter set to “radial” for Equation 2 - Radial Basis Function kernel (RBF). This package was used because during initial attempts with the kernlab

package, warnings were raised stating that the algorithm was not able to numerically solve the SVM optimization problem in the hardcoded number of iterations (100). Changing the kernel also meant that the metric used to calculate variable importance with the linear kernel implementation could not be used with the RBF implementation. Instead, the variable importance was estimated with the change in the kernel function (Dziuda, 2024):

1. Use Equation 2 - Radial Basis Function kernel to calculate the value of the kernel function with all variables.
2. For each predictor variable k, $k = 1, ..., p^*$:
    a. Calculate the value of the kernel function with k removed so that the kernel function becomes $K\left(\boldsymbol{x}_{i,(p^*-k)}, \boldsymbol{x}_{j,(p^*-k)}\right)$.
    b. The importance of the removed variable, I_k, is: $\left|K\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) - K\left(\boldsymbol{x}_{i,(p^*-k)}, \boldsymbol{x}_{j,(p^*-k)}\right)\right|$.
3. Sort all k in descending order by corresponding I_k.

Figure 29 - Calculation for change in RBF kernel function.

The computational time increased significantly while using the RBF kernel and new importance algorithm, but 100 parallel feature selection experiments were still performed to maintain consistency and comparability across learning algorithms.

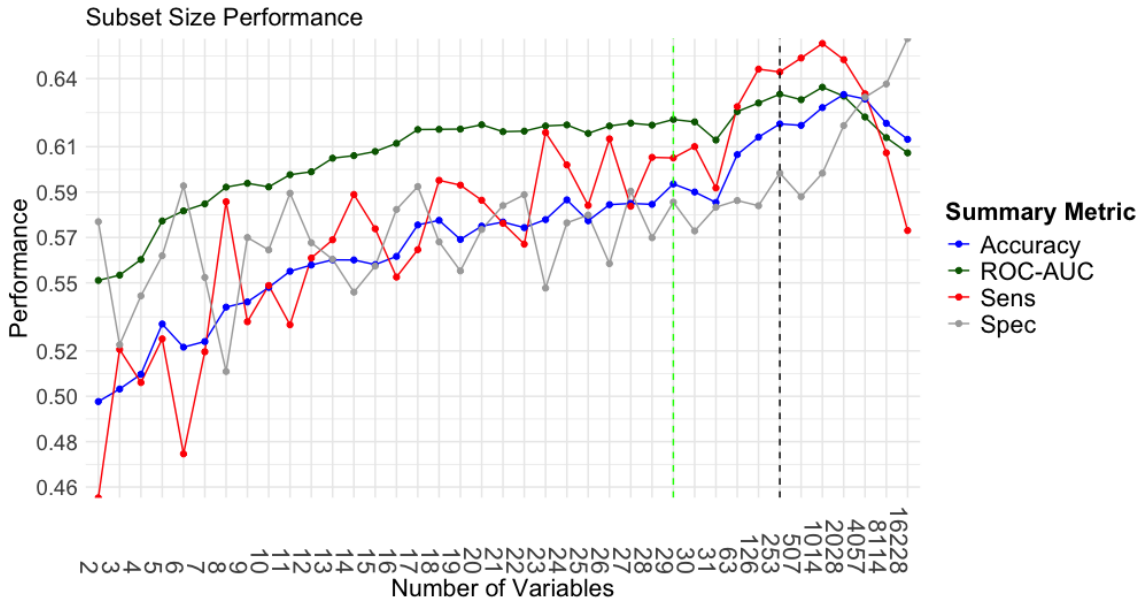

Figure 30 - SVM-RFE with RBF kernel performance summary. The black and vertical line is at the automatically chosen size, and the green and vertical line is at the manually chosen size.

The ROC-AUC (0.622), accuracy (0.594), sensitivity (0.605), and specificity (0.586) were the best out of the three feature selection experiments, but these metrics were recorded at a much larger subset size of 29. Any of the smaller subset sizes resulted in a worse performance than the subset size of 29 in most of the metrics (although either sensitivity or specificity could have been larger at a given subset size). When the subset size was very small, the performance was more similar to the performance of the linear SVM biomarker. The larger subset size and non-linear kernel made this model the most susceptible to being overfit. None of the probe IDs selected in this feature selection experiment were selected in the previous two experiments. It seems that none of the probe IDs that are consistently important predictors of NSLC, which makes the different ranking algorithms less likely to select the same probe IDs as being important across different experiments and learning algorithms.

Table 12 - Description of top four gene transcripts selected by RBF SVM-RFE.

Probe ID	Gene Symbol	Gene Name
A_24_P688133	PPM1A	protein phosphatase, Mg2+/Mn2+ dependent 1A
A_24_P403244	PILRB	paired immunoglobin like type 2 receptor beta
A_24_P117803	N/A	peptidylprolyl isomerase A pseudogene 35
A_23_P131308	CYP27A1	cytochrome P450 family 27 subfamily A member 1
…	…	…

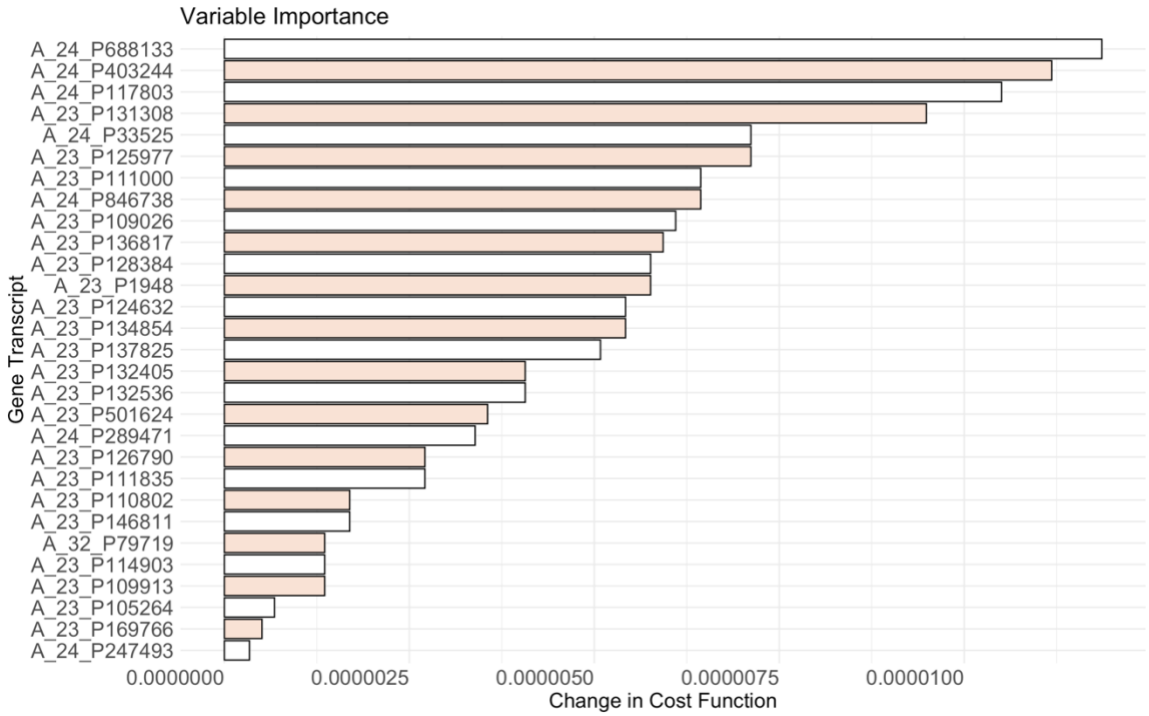

Figure 31 - Final RBF SVM biomarker variable importance.

The most important gene transcript was A_24_P688133 with a change in the cost function of 1.19e-5, the second gene transcript was A_24_P2403244 with a change in the cost function of 1.12e-5, the third gene transcript was A_24_P117803, and the fourth gene transcript was A_23_P131308. The importance of the remaining gene transcripts dropped quite a bit with the fifth gene transcript, A_24_P33525, having a change in the

cost function of 7.12e-6. The least important gene transcript had a change in cost function of 3.39e-7.

Parameter Tuning

This section provides an explanation of the parameter tuning that was incorporated into the feature selection experiments in the previous section. In the algorithm shown in Figure 17 - Backward Recursive Feature Selection with resampling (Kuhn et al., 2013)., step 9 which was responsible for model fitting was modified to tune the hyperparameters. This was done within the original feature selection experiments because tuning parameters after feature selection and also using the same training data (with only the selected features), would have been an internal validation. The model would have been trained and also validated on the same training data. In the modified implementation of step 9, an additional and separate bootstrap resampling experiment was performed with five resamples. Only five inner bootstrap resamples were used at each step 9, as there were already 100 parallel RFE experiments performed in the outer loop. For each of the five inner resamples (all with the same features), a different model was fit for each hyperparameter value or combination of hyperparameter values. The hyperparameter values that were chosen at each step 9 were the values that resulted in the largest average ROC-AUC on the OOB samples of the five inner resamples. Then after all outer resamples had finished and the optimal subset size was chosen using the calculation described in the Select Size section from Feature Selection, the final hyperparameter values were extracted. At this step, only the hyperparameter values that were the best performing during each inner resampling experiment and at the optimal subset size remained. The hyperparameter values at the chosen subset size were the only values

considered because hyperparameters at the different subsets of variables would not be reliable or relevant. The average of the performance metrics for each combination of hyperparameter values across all 100 external resamples were used to determine the values that would be used in the final models. The caret package selected the hyperparameter values solely based on ROC-AUC, but the other metrics (accuracy, sensitivity, and specificity) were also considered to determine the final values.

Linear Kernel SVM Tuning

The only hyperparameter to tune for the SVM algorithm using the linear kernel was *C* which controls the degree of penalization for margin violations. As mentioned in the **Error! Reference source not found.** section, the size of *C* was inversely correlated with the size of the margin. These values of *C* are relatively small compared to the ksvm default size of 1.

Table 13 - Tune grid for SVM-RFE with linear kernel.
The best performing value is highlighted in green.

C
0.016
0.031
0.063
0.125
0.250

RBF Kernel SVM Tuning

The SVM algorithm using the RBF kernel required additional tuning of the sigma value. The potential values were calculated using the sigest function from kernlab. They were the 10% quantile, median, and 90% quantile of the pairwise distances between all

samples in the input space. According to Caputo et al., 2002, any value between the 10% quantile and 90% quantile would provide good results for the radial basis function kernel.

Table 14 - Tune grid for SVM-RFE with RBF kernel.
The best combination of values is highlighted in green.

C	sigma
0.063	2.002e-05
0.063	3.682e-05
0.063	7.239e-05
0.125	2.002e-05
0.125	3.682e-05
0.125	7.239e-05
0.25	2.002e-05
0.25	3.682e-05
0.25	7.239e-05
0.5	2.002e-05
0.5	3.682e-05
0.5	7.239e-05

Biomarker Comparison

The performance metrics of the three biomarkers at the chosen subset sizes were plotted against each other to better visualize the OOB performance. In order to avoid an internal validation, these OOB performance measures are based on the original RFE experiments that can be seen in Figures 24, 27, and 30.

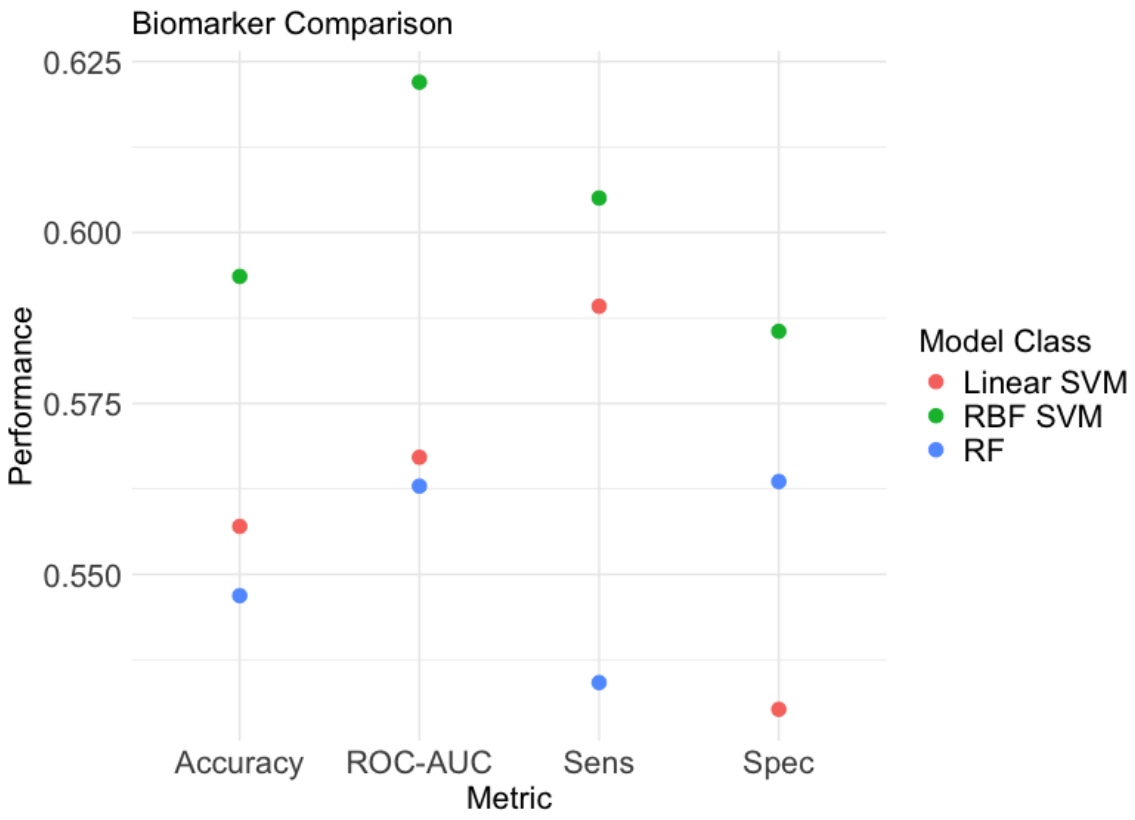

Figure 32- Biomarker OOB metric comparisons.

The RBF SVM biomarker outperformed the other two biomarkers in all performance metrics, although it was the largest with a subset size of 29 compared to 13 for the RF biomarker and 2 for the linear SVM biomarker. The most notable improvement shown by the RBF SVM biomarker was the ROC-AUC being 0.622 which was 9.7% larger than the linear SVM ROC-AUC of 0.567 and 10.4% larger than the RF ROC-AUC of 0.563. The RF sensitivity was also quite separated from the sensitivity values of RBF SVM and linear SVM. The RF sensitivity of 0.534 was 10.25% smaller than the linear SVM sensitivity of 0.589 and 13.28% smaller than the RBF SVM sensitivity. The performance of the biomarkers in regard to accuracy and specificity were more similar. The linear SVM biomarker outperformed the RF biomarker in accuracy, but not specificity. The

RBF SVM biomarker performed the best, the linear SVM biomarker was the most parsimonious, and the RF biomarker performed the worst, but was moderately parsimonious compared to the other biomarkers. All biomarkers were tested on the test data set.

Biomarker Evaluation

All three biomarkers were tested on the 50 test set samples (that had been held aside until now), and their performance was evaluated. During classification, the probability (between 0-1) of a sample belonging to the positive class was first assigned. Whether this probability was greater than a threshold (0.50 was default for the packages used in R), determined the predicted class for the sample. Samples with an assigned probability greater than or equal to 0.50 were classified as the positive class, and samples with an assigned probability less than 0.50 were classified as the negative class. This probability was needed for calculating the ROC curves shown in Figure 33-35. The point on the curve corresponding to the 0.50 threshold indicates the TPR and FPR from the classification performance of the test samples.

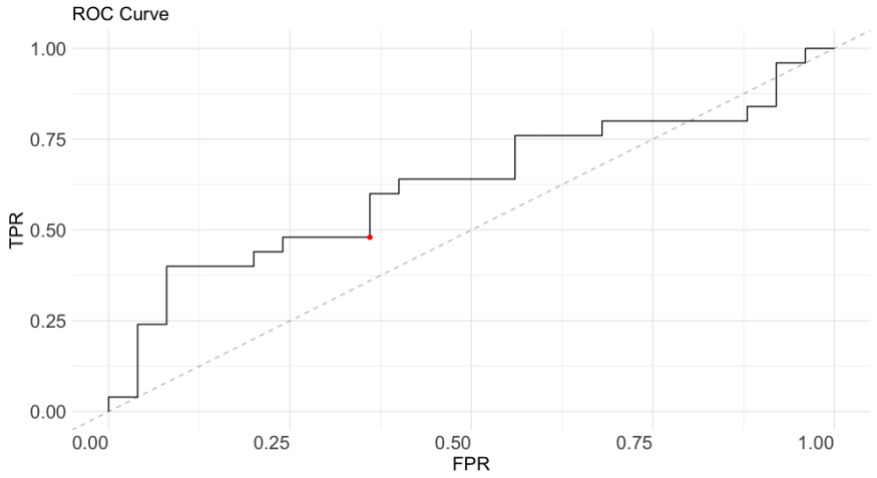

Figure 33 - RF ROC curve. The red dot is where the threshold is 50%.

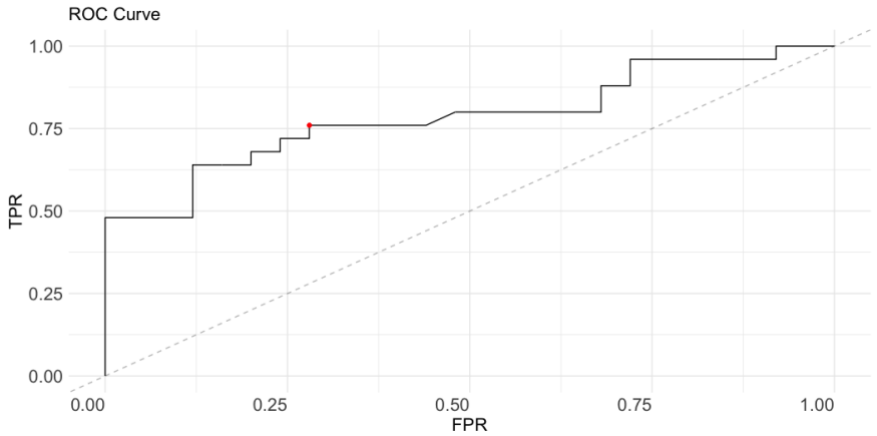

Figure 34 – Linear SVM ROC curve. The red dot is where the threshold is 50%.

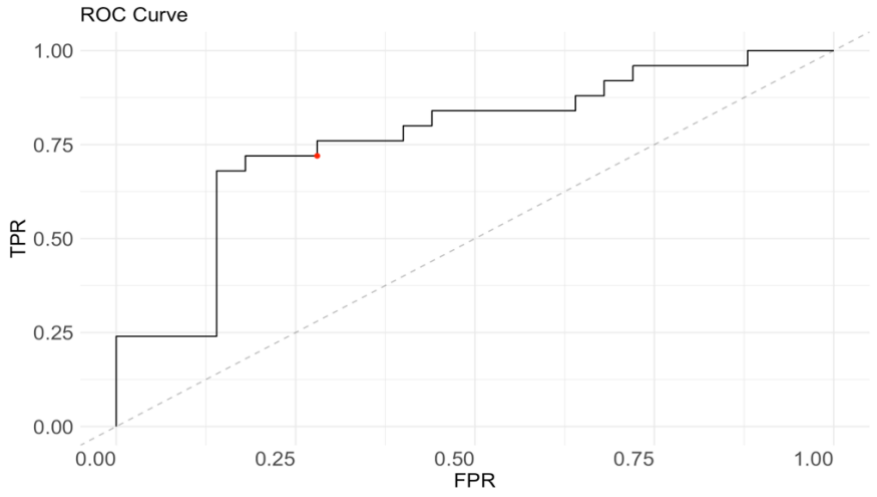

Figure 35 - RBF SVM ROC curve. The red dot is where the threshold is 50%.

Table 15 - Final model performance on test data set.

Algorithm	ROC-AUC	Accuracy	Sensitivity	Specificity
RF	0.624	0.560	0.480	0.640
Linear SVM	0.785	0.740	0.760	0.720
RBF SVM	0.770	0.700	0.680	0.700

As expected, none of the biomarkers were highly predictive, although they all performed better on the test data set then they did on the OOB samples during training. The linear SVM biomarker was the best biomarker as all the performance metrics were better, and it was a smaller biomarker with only two gene transcripts. The performance of these biomarkers on the test data set is a testament to the importance of a parsimonious biomarker. Although the performance on the OOB data was not good, the small size of the biomarkers allowed them to generalize better to the unseen test data set. Using many feature selection experiments, the distribution of data being nearly identical in both the training and the test set, and the ratio of samples belonging to each class in both the training and test data were 1:1 (as shown in the Partition Data section) lead to a predictable performance on the test data set. The number of misclassifications at each decision threshold are shown in Figures 36-38.

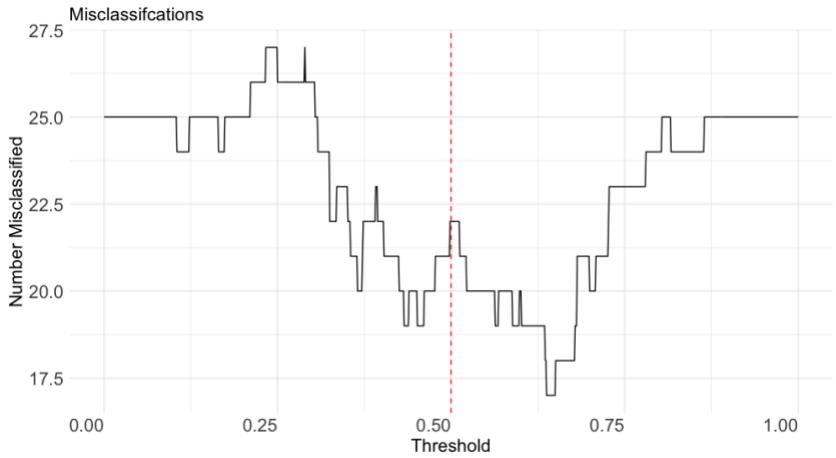

Figure 36 – Random forests number of misclassifications at different thresholds. The red line at the 50% probability threshold that was used.

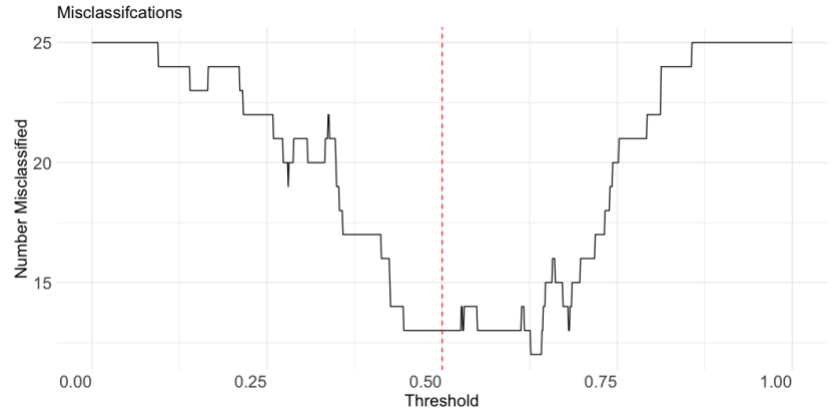

Figure 37 – Linear SVM number of misclassifications at different thresholds. The red line is at the 50% probability threshold that was used.

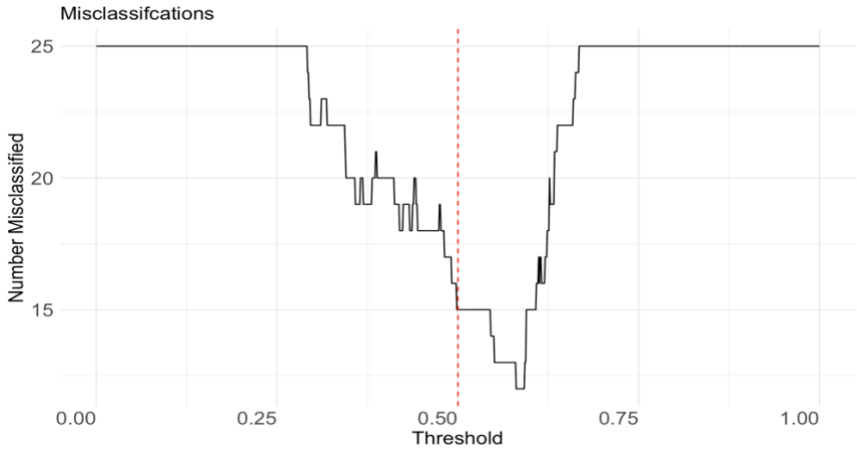

Figure 38 - RBF SVM number of misclassifications at different thresholds. The red line is at the 50% probability threshold that was used.

Figures 36-38 show that the optimal threshold on the test data set would have been different for all of the models and would not have led to a highly predictive model at any threshold. The amount of fluctuation in the plots showed that optimizing this value would not have been practical with such a small test data set that only contained 50 samples as this value may change on a different dataset. The confusion matrices below show the summary of classifications made by the models.

Table 16 - RF confusion matrix on test set.

		Actual	
		Normal	NSCLC
Prediction	Normal	16	13
	NSCLC	9	12

Table 17 - Linear SVM confusion matrix on test set.

		Actual	
		Normal	NSCLC
Prediction	Normal	18	6
	NSCLC	7	19

Table 18 - RBF SVM confusion matrix on test set.

		Actual	
		Normal	NSCLC
Prediction	Normal	18	7
	NSCLC	7	18

Discussion

This project aimed to answer the Research Question. The answer to the question was that combining all samples of NSCLC subtypes in the data (AC, SCC, and LCC) into a single NSCLC class, **did not** allow for a small and highly predictive biomarker of NSCLC to be identified using supervised multivariate feature selection in which the other class was normal lung. The identified biomarkers were subpar when used to classify the test data set with ROC-AUC values of 0.624, 0.785, and 0.770. While the biomarkers were parsimonious - with the RF model using 13 gene transcripts, the linear SVM model using only 2 transcripts, and the RBF SVM model using 29 gene transcripts - this was nothing to brag about. Their lack of predictive power made them unsuitable to reliably classify the two classes. In contrast, the 15-gene classifier found by Lazar et al. (2013) achieved a cross-validated AUC of 96% when distinguishing between the two NSCLC subclasses (AC and SCC).

One reason for the disparity in results between the two projects may be due to the composition and variability of the data within each class. The samples in both classes of this project were heterogeneous as the NSCLC subtypes arise in different cell types within the lungs, and they were grouped into a single class. Therefore the samples in the normal class would be from adjacent healthy tissue that would also be from different cell types. In the Lazar et al. (2013) experiment, the AC samples arise from the alveoli, and the SCC samples arise from the squamous cells that line the bronchi which made those classes more homogenous. This heterogeneity could have obscured the discriminatory gene expression patterns associated specifically with NSCLC making it challenging for the learning algorithms to identify predictive biomarkers. To improve the chances of

success, a larger data set may have been needed that allowed the learning algorithms to uncover the underlying gene expression patterns of NSCLC.

Another reason for the poor performance of the biomarkers identified in this project could have been due to differences in the preprocessing of the data used for feature selection and model training. Lazar et al. (2013) used the raw Agilent data, and it was not clear how they preprocessed the data. In this project, the processed Agilent data was used in which the Agilent Feature Extraction software did much of the low-level processing such as removing most of the dye bias and subtracting background signal. Additional processing was done prior to feature selection in the Preprocessing Steps section. Lazar et al. (2013) also used the exon probes and viral transcript probes, while this project only included the 41,000 gene transcripts.

Limitations

One of the limitations to this project were the small number of samples in the dataset (123 samples in each class). This was small relative to even the number of variables that remained for biomarker discovery (16,228). To deal with this curse of dimensionality, feature selection was performed many times and results aggregated. The one standard error rule was also implemented to balance biomarker performance and simplicity. Even then, there was still a risk of overfitting the training data, and it is of course possible that these biomarkers would not perform the same on a different dataset or a larger data set.

Additionally, the processed data was not processed perfectly by the Agilent Feature Extraction software. The benefits of the processing software such as accounting for variability of background noise among different regions of the microarray and

correcting dye bias certainly outweighed the imperfections. However, imperfections in the processing were exposed by the negative control probes (-3xSLv1) in each microarray still having positive signal values. Since the negative control probes were designed to not bind any cRNA on the microarray, the signal value for these probes should have been zero after processing if it had been perfect. The addition of the negative control probes to the microarrays allowed for the remaining background level to be removed from the other probes.

Another limitation was that only the cy5 data was used, and the cy3 data was not used. The reason for this was explained in the Dataset Used section. Using only cy5 data meant that only data from a single channel from each microarray was used. This left the data more vulnerable to non-biological variation arising from dye labeling, hybridization to the microarray, scanning, or dye degradation. The use of the processed data helped mitigate the variation that would have much more extreme had the raw data been used instead.

Another limitation was that this gene expression data was derived from patient biopsies, which are typically only obtained after a tumor has been diagnosed in the patient. Consequently, any biomarker identified in this project would not have been able to be used for early diagnosis since a biopsy would not be performed preemptively. Any biomarker identified could only be a contribution towards understanding the underlying mechanism of NSCLC, and not diagnosis.

References

Abe, Y., Sano, T., & Tanaka, N. (2023). The Role of PRMT5 in Immuno-Oncology. *Genes*, *14*(3), 678. https://doi.org/10.3390/genes14030678

Agilent Technologies, Inc. (2021). *Agilent feature extraction* (12.2 ed.).

Agilent Technologies. (2024). Earray.chem.agilent.com. Retrieved April 2, 2024, from https://earray.chem.agilent.com/earray/catalogGeneLists.do?action=displaylist#

Ashok Kumar, P., Graziano, S. L., Danziger, N., Pavlick, D., Severson, E. A., Ramkissoon, S. H., Huang, R. S. P., Decker, B., & Ross, J. S. (2023). Genomic landscape of non-small-cell lung cancer with methylthioadenosine phosphorylase (MTAP) deficiency. *Cancer medicine*, *12*(2), 1157–1166. https://doi.org/10.1002/cam4.4971

Bajbouj, K., Ramakrishnan, R. K., Saber-Ayad, M., Omar, H. A., Saheb Sharif-Askari, N., Shafarin, J., Elmoselhi, A. B., Ihmaid, A., AlHaj Ali, S., Alalool, A., Abdullah, R., & Hamid, Q. (2021). PRMT5 Selective Inhibitor Enhances Therapeutic Efficacy of Cisplatin in Lung Cancer Cells. *International journal of molecular sciences*, *22*(11), 6131. https://doi.org/10.3390/ijms22116131

Caputo, B., Sim, K., Furesjo, F., & Smola, A. (2002). Appearance-based object recognition using SVMs: Which kernel should I use? *Proceedings of the NIPS Workshop on Statistical Methods for Computational Experiments in Visual Processing and Computer Vision*, Whistler, Canada.

Chain, B. (2023). *Agilp Manual Vignette*. Retrieved April 1, 2024, from https://bioconductor.org/packages/release/bioc/vignettes/agilp/inst/doc/agilp_manual.pdf

Chain, B., Bowen, H., Hammond, J., Larkin, E., McDade, S., & Wilson, T. (2010). Error, reproducibility and sensitivity: A pipeline for data processing of Agilent oligonucleotide expression arrays. *BMC Bioinformatics, 11*, 344. https://doi.org/10.1186/1471-2105-11-344

Dessen, Philippe (2013). Transcription profiling by array on 123 paired tumor and non-tumor tissue samples from patients with non-small cell lung carcinoma to gain a systems biology insight into the current clinical classification. *BioStudies*, E-MTAB-1132. Retrieved from https://www.ebi.ac.uk/biostudies/arrayexpress/studies/E-MTAB-1132

Dziuda, D. M. (2010). Data mining for genomics and proteomics: analysis of gene and protein expression data (Vol. 1). John Wiley & Sons

Dziuda, D. M. (2024). 11.5 Variable importance. In *Multivariate biomarker discovery*. Cambridge University Press.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning, 46*(1-3), 389–422. https://doi.org/10.1023/A:1012487302797

Karatzoglou, A., Smola, A., Hornik, K., & Zeileis, A. (2004). kernlab - An S4 Package for Kernel Methods in R. *Journal of Statistical Software*, *11*(9), 1–20.

Kuhn, M. (2008). Building Predictive Models in R Using the caret Package. *Journal of Statistical Software*, *28*(5), 1–26. https://doi.org/10.18637/jss.v028.i05

Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling*. Springer.

Larose, D. T., & Larose, C. D. (2015). Data mining and predictive analytics (2nd ed.). John Wiley & Sons

Lazar, V., Suo, C., Orear, C. *et al.* Integrated molecular portrait of non-small cell lung cancers. *BMC Med Genomics* **6**, 53 (2013). https://doi.org/10.1186/1755-8794-6-53

Li, Y., Yang, Y., Liu, X., Long, Y., & Zheng, Y. (2019). PRMT5 Promotes Human Lung Cancer Cell Apoptosis via Akt/Gsk3β Signaling Induced by Resveratrol. *Cell transplantation*, *28*(12), 1664–1673. https://doi.org/10.1177/0963689719885083

Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. *R News*, *2*(3), 18–22.

Meyer, D., Evgenia, E., Hornik, K., Weingessel, A., Leisch, F., Chang, C., & Lin, C. (2024). E1071 – Misc Functions of the Department of Statistics, Probability Theory Group, 1.7-16.

Ritchie, M.E., Phipson, B., Wu, D., Hu, Y., Law, C.W., Shi, W., and Smyth, G.K. (2015). limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Research 43(7), e47.

Šutić, M., Vukić, A., Baranašić, J., Försti, A., Džubur, F., Samaržija, M., Jakopović, M., Brčić, L., & Knežević, J. (2021). Diagnostic, Predictive, and Prognostic Biomarkers in Non-Small Cell Lung Cancer (NSCLC) Management. *Journal of personalized medicine*, *11*(11), 1102. https://doi.org/10.3390/jpm11111102

Two-Color Microarray-Based Gene Expression Analysis. (2014). Agilent Technologies. https://www.agilent.com/cs/library/usermanuals/Public/G4140-90050_GeneExpression_TwoColor_6.7.pdf

U.S. Cancer Statistics Working Group. U.S. Cancer Statistics Data Visualizations Tool. U.S. Department of Health and Human Services, Centers for Disease Control and

Prevention and National Cancer Institute; released in June 2024.

https://www.cdc.gov/cancer/dataviz