Data Scientist Technical Skills Assessment
CSP Informatics Center, VA Boston Healthcare System

By: Victor Manuel Murcia Ruiz

Date: 9/23/2022

Abstract

In this project I was provided with a dataset containing simulated data for 190 patients. The aim of the project was to build and evaluate a predictive model of the one-year survival after diagnosis with NSCLC. As part of this project, I performed exploratory data analysis, data cleanup, data visualization, feature engineering, dimensionality reduction/feature selection through ANOVA, model building and model testing. A total of 10 different classifier models were tested and the 3 best performing models were determined to be the Perceptron, Random Forest, and SVC classifiers. The performance of these models was assessed based on their accuracy, sensitivity, and specificity scores.

The code written and used for this project can be found in the github repository located here: https://github.com/victormurcia/VA-Assignment/blob/main/VA%20Assignment.ipynb

Exploratory Data Analysis (EDA)

Two datasets were provided as part of this project: the clinical.csv dataset and the genomics.csv dataset. The clinical dataset contained 16 features and 190 unique entries associated with patients. Within these 16 features is the column labeled as 'Outcome' which states whether a patient is alive or dead after follow-up. The Outcome column will serve as the target/dependent variable for the model. Given that the target feature is a categorical variable (i.e., Alive or Dead), the problem that needs to be solved is a classification one.

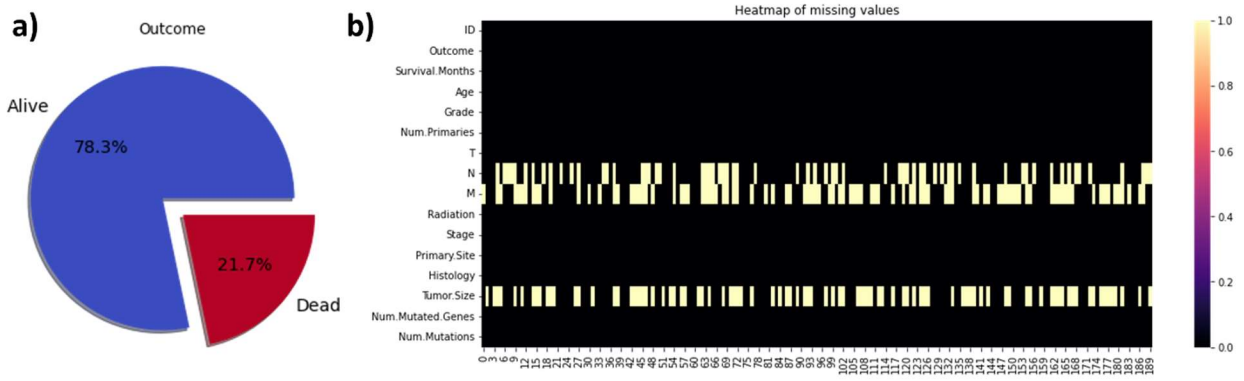

Figure 1. a) Outcome distribution shows that there is significant class imbalance. b) Null matrix showing missing values in the N,M and Tumor.Size columns

The distribution of Outcomes is highly unbalanced (Figure 1a) which means that this problem is an *imbalanced* classification problem. This is important because I will need to account for this during my modeling later. In addition to this, as can be seen from the null matrix (Figure

1b), the columns N, M, and Tumor.Size have a sizeable number of missing entries. Columns N and M are discrete categorical variables and have a cardinality of 9 and 2 respectively. Tumor.Size is a continuous numerical variable that is right-skewed. Given this, I decided to fill the missing entries in N and M with the mode of those columns, while the missing entries in Tumor.Size were filled with the median of the column.

Once these missing entries were dealt with, I looked into the genomics dataset and found that there was gene data for 184 unique patients. In other words, there were 6 patients without any genomic data from the clinical dataset. The missing patients were determined to be those with the following IDs : 19, 30, 40 , 89, 142, and 166. I decided to handle the 6 missing patients by dropping them from the dataset since the information loss from doing this Is outweighed by the information gain of introducing the genomic data. Then, I was able to merge both datasets through reshaping operations which left me with a dataset comprised of 184 patients with 66 features each.

Data Visualization

With the extended dataset I created, I started to explore the relationships between the different features/predictors to see how they were distributed with respect to the patient Outcome. Figure 2 shows how Grade, Num.Primaries, and T are related to the Outcome. The color of the bars represents whether the patient is alive (purple) or dead (orange). From the leftmost plot it can be seen that all kinds of tumor grades resulted in higher death rates than not. Interestingly, there were no tumor grades of 0 or 1 present in this dataset. The middle plot shows that having 0 as the number of primary tumors has a much higher death rate than having that value set to 1. The right most plot shows that Stage 3 results exclusively in patient death. It also shows that Stage 4, UNK, and 2b have disproportionately higher death rates than the other categories. The cancer grade gives an idea of the tumor growth speed, the likelihood a cancer will spread to other parts of the body, and the cell morphology. As such, this gave me the idea to generate/engineer a set of 9 new features: 'Slow_Growth', 'Medium Growth', 'Fast_Growth', 'LessLikely_Spread', 'Likely_Spread' , 'HighlyLikely_Spread', 'Normal_Cells', 'Abnormal_Cells' , and 'Most_Abnormal_Cells' based on the cancer grade.

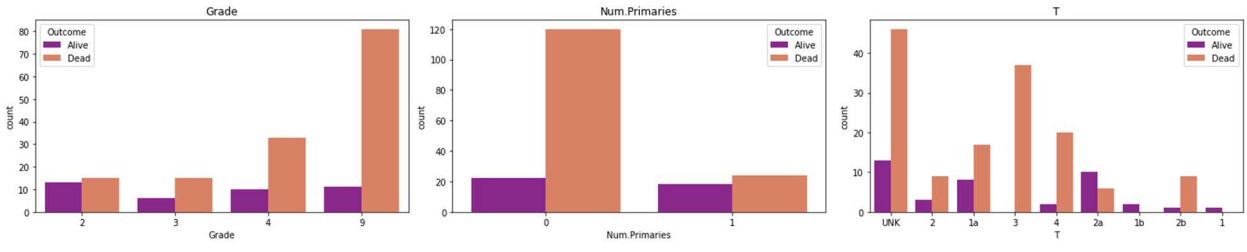

Figure 2. Visualization of features and distribution to patient outcome left) Cancer grade. middle) Num.Primaries. right) T value

Figure 3 shows the visualization results for N, M, Radiation and Stage. The main observation for N is that a value of 2 results in a much higher death rate relative to the other N values. It can also be seen that having 0 distant metastases has a much higher death rate than not. No exposure to radiation appears to have much higher death rates than not. Finally, all cancer stages with the exception of IB and 1B had higher survival rates than not. My main takeaway from

these observations is that there is an error in the labeling of stages IB and 1B as well as stage IV and IVB. Seeing these graphs made me realize that IB and 1B are likely referring to the same stage and should be combined into a single class. In addition to this, I assumed that Stage IV likely refers to Stage IVA. I incorporated these adjustments into the dataframe which also allowed me to engineer a few new features. N stands for node in the TNM staging scheme and describes whether cancer has spread to lymph nodes. There are 3 categories that indicate cancer being present N1-N3. N0 means that there are no cancer cells. NX means that lymph nodes can't be assessed (does not necessarily mean that cancer is not present). Hence, I could make a new feature called 'Spread_To_Lymph_Nodes' based on whether the N value is greater than 0. In addition to this, I could use the Stage feature to generate two new features called ‘Stage_A’ and ‘Stage_B’ based on whether the cancer stage has the A or B qualifier.

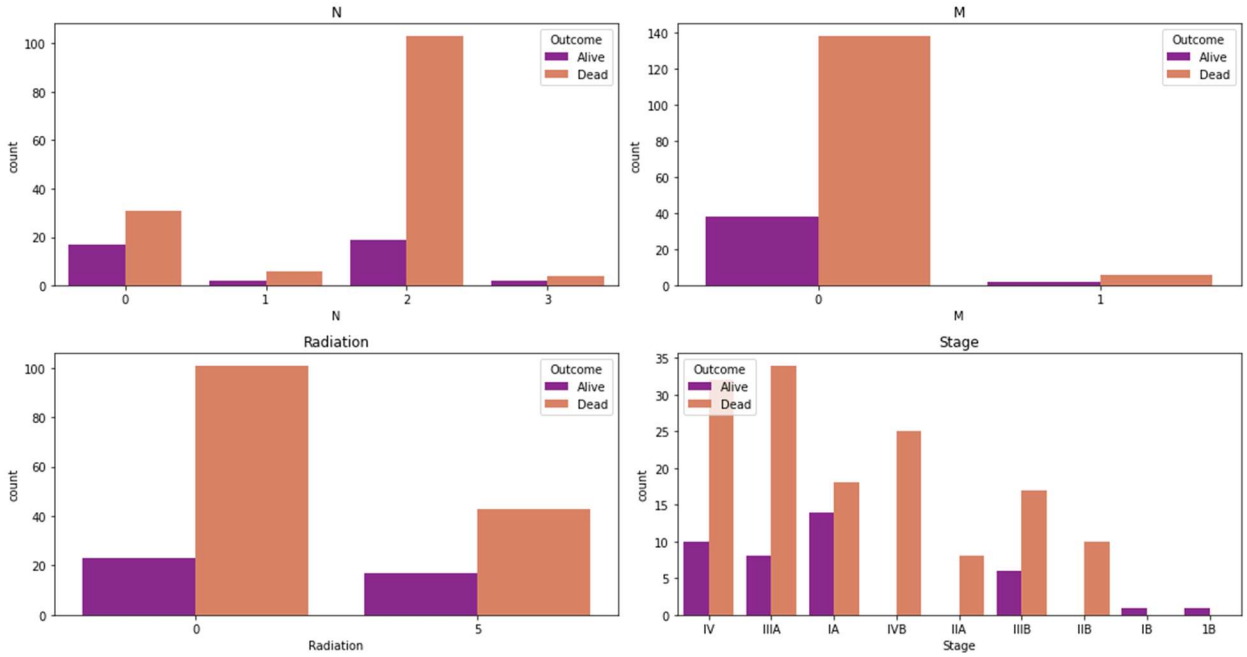

Figure 3. Relationship of N, M, Radiation, and Stage to patient outcome.

From Figure 4 the relationship of Primary.Site was explored. From this plot I observed that if primary tumor site is left hilar, right hilar, or both lungs then there is no survival. If primary tumor site is right upper lobe, left upper lobe, right upper lobe or right middle lobe then there is a higher death rate than not. The remaining categories have higher survival rates From these observation I decided that I would make a category called ‘Hilar’ that would assess if the Primary.Site was in the Left or Right Hilar. I also decided to make two categories called Left_PS and Right_PS that would be populated based on whether the primary tumor site is situated on the right or the left side. I also noticed what I’m almost certain is a typo in one of the categories ‘Righ Upper Lobe’. This should almost certainly be labeled as 'Right Upper lobe'.

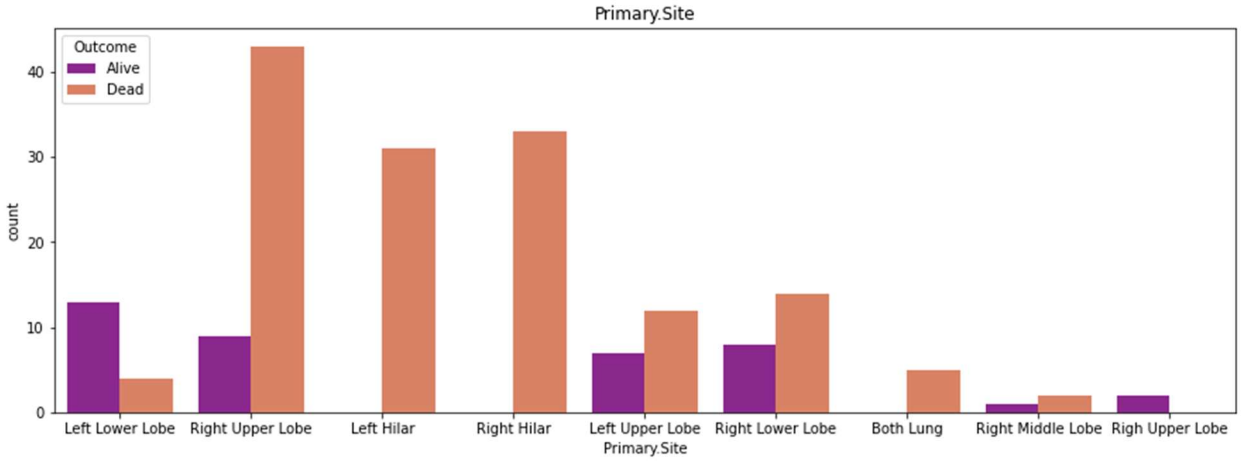

Figure 4. Relationship between tumor primary site and patient outcome

From Figure 5 I determined that large-cell carcinoma and adenocarcinoma have much higher death rates than squamous cell carcinoma. I also noticed that patients with 8 mutated genes in the tumor had higher survival rates. All other numbers of mutated genes had higher death rates. The biggest difference in survival and not seem to be for 1,2,and 3 mutated genes. The number of genes with mutations between 1-6 have higher death rates. The ones higher than 6 have higher survival rates.

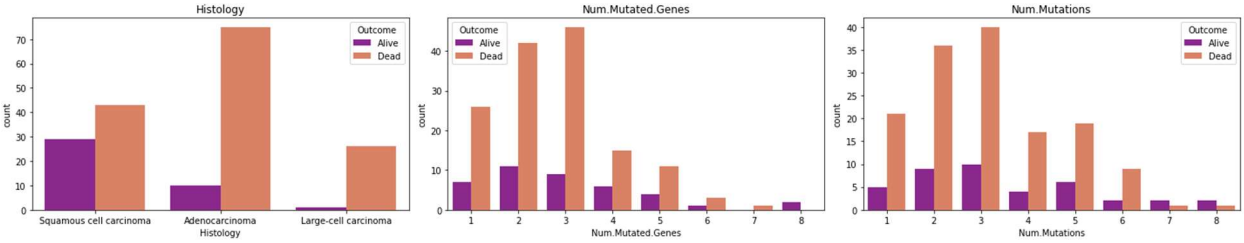

Figure 5. Relationship between Histology, Number of Mutated Genes, and Number of mutations to patient outcome.

From Figure 6, the KDE curves show that Survival.Months category could be split into 3 classes, namely, 0-40, 40-50 and 50+ months. Similarly, the Age and Tumor.Size features could be partitioned into two categories. Age for instance could be split into a group of patients with ages between50-65 years old and another group of patients with ages above 65 years old. Tumor.Size could be split into a group where sizes are below 6cm and another group where the tumor sizes are above 6cm.

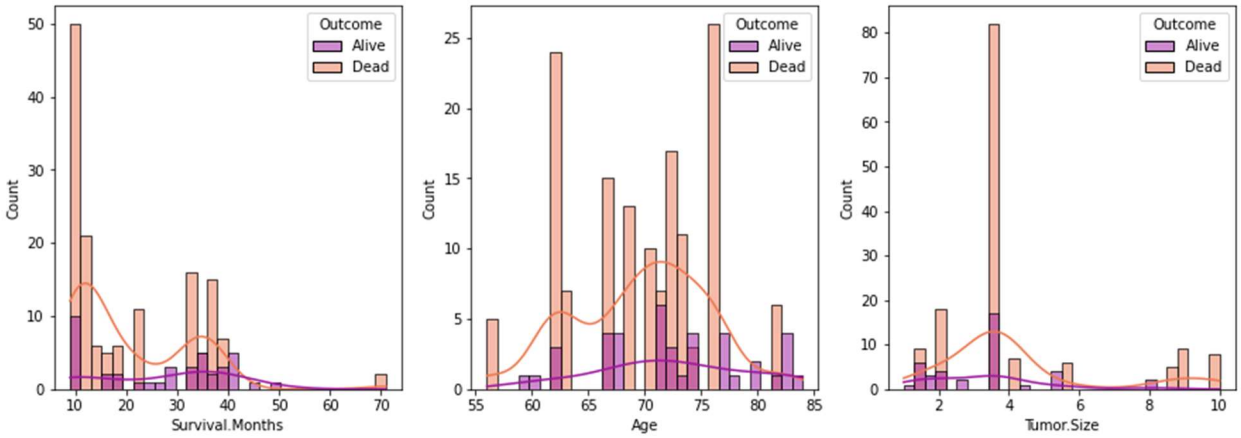

Figure 6. Relationship between the Survival months, age and tumor size to patient Outcome.

In addition to looking at all these features, I looked into the relationship the presence of all the genes and the patient outcome. However, I did not notice anything particularly interesting. If I had more time, I would like to delve a bit deeper into these gene expressions and see if there is some pattern hidden in there that could help engineer features from them. The plots showing this exploration can be found in the accompanying Jupyter notebook.

Feature Engineering

I generated a total of 22 new features through feature engineering. I will show the most interesting/useful ones in this report, but if you'd like to see them all, you may refer to the Jupyter notebook. From Figure 7 it can be seen that if the primary site was in the Hilar then the patient definitely died. Otherwise, they still have a pretty high chance of dying. It can also be seen that there seems to be a higher chance of death if the primary site of the tumor is on the left. It can also be seen that Stage A has a higher death rate than Stage B. The last bit of feature engineering that was carried out involved one-hot encoding of the categorical/discrete features present in the data. The result of this process increased the dimensionality of my feature space from 88 to 161.

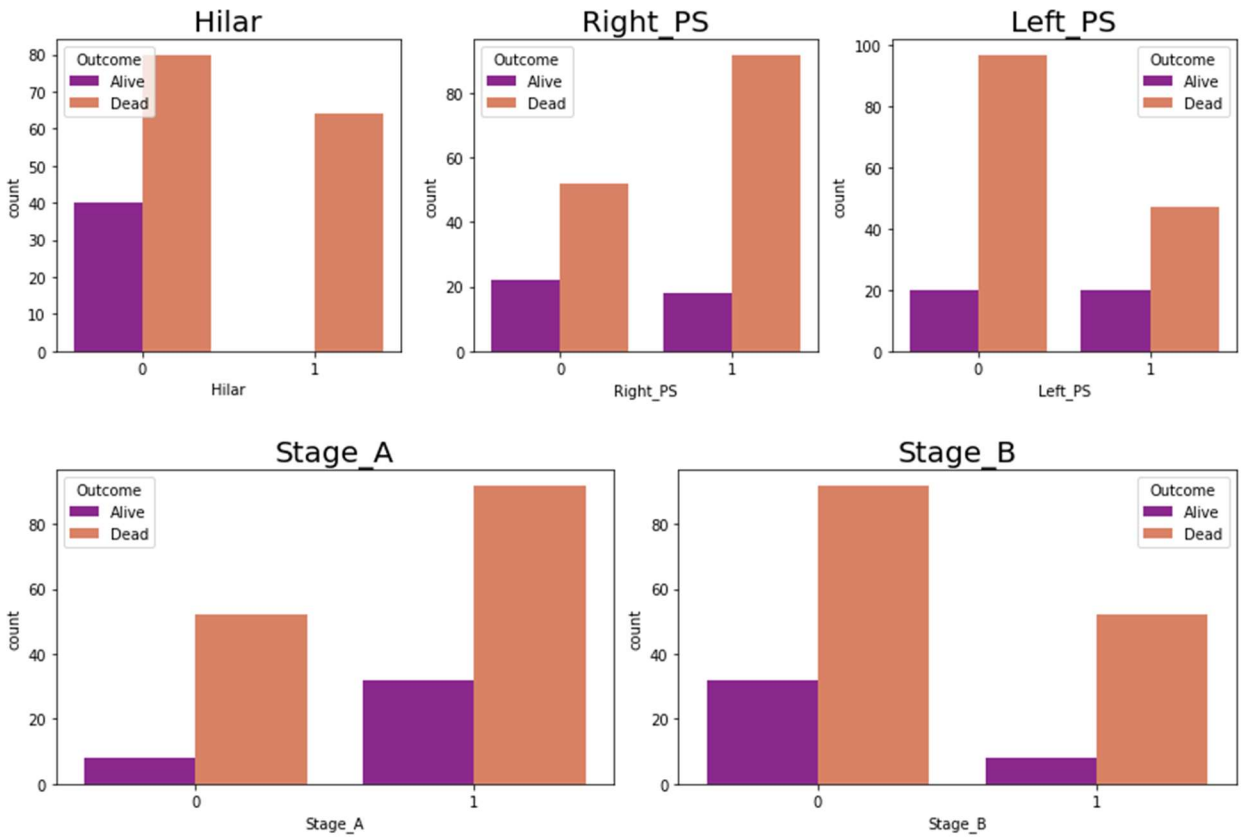

Figure 7. Relationship between the engineered features (Hilar, Right/Left primary site, and Cancer Stage) to the patient outcome

Feature Selection and Dimensionality Reduction

More features don't necessarily equate to a better model. Rather choosing the right features generate better models. Having too many features can also lead to overfitting problems which make models less generalizable and flexible. As such, I carried out an ANOVA test to determine the features with the highest predictive power that I have at my disposal based on the F-Score. The result of the ANOVA test is shown below were the F-Score distribution is plotted. From this plot it can be seen that nearly 100 of the features have F-Scores below 5. These features have the lowest predictive power and as such I will not use them for my model. I can always return and add more features later if necessary. The result of choosing features with F-Scores above 5 reduced my feature dimensionality to 62 attributes. The 3 strongest attributes based on the ANOVA test were determined to be Primary.Site_Left Lower Lobe, Hilar, and Histology_Squamous cell carcinoma. Now, I have cleaned, visualized, described, engineered, selected and understood my data fairly well and am ready to start training and building machine learning models.

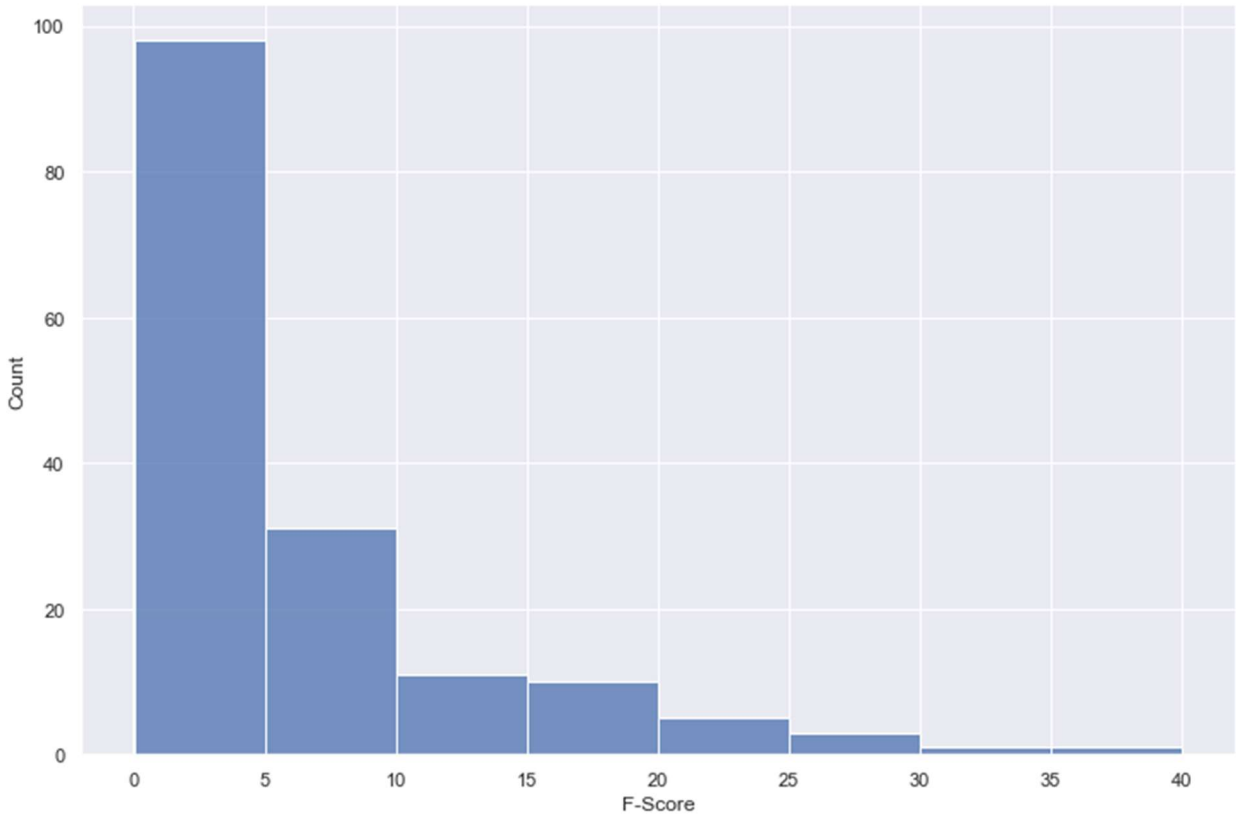

Figure 8. F-Score distribution of data attributes.

Preparing and Testing Classifier Models

Now that I have my data nice and prepared and have a pretty good understanding of what is going on, I'm finally ready to start testing some classifier models. I'll setup a wrapper function that will allow me to easily train and compare multiple classifiers. It will do this while doing grid search and cross validation on the models so the results that I get will be reliable. The results of this function will allow me to determine which models are performing the best which I can then optimize further. For this project, I'll use cross-validation with k=10 to try to ameliorate the effect of class imbalance. This performs a stratified K-Fold which is better suited to handle imbalanced data. I'll also use class weights wherever possible to handle the class imbalance. The data will use an 80/20 split per common practice.

This wrapper function will take the model dataframe, a model definition list, and the dependent variable as inputs. The model definition list is composed of:

```
The model initializer,
The model name
A dictionary containing the values to use for the grid search
```

The output of this function will be dataframes containing the results of the models that I trained.

The results dataframe will contain the model:

```
Accuracy (int)
Specificity (int)
Sensitivity (int)
Training Time (int)
Best parameters from Grid Search (dict)
```

The 10 models I tested are: Perceptron, Gaussian Naïve Bayes, Random Forest, XGBoost, K Neighbors, SVC, MLP, Gradient Boosting, Bagging, and Decision Tree. The results of the training process are shown below.

	Method	Accuracy Score	Sensitivity	Specificity	Training Time	Best Parameters
1	GaussianNB	0.97297	0.96875	1.0	0.0	{'var_smoothing': 1e-12}
0	Perceptron	0.91892	0.911765	1.0	0.0	{'alpha': 0.001, 'penalty': 'l1'}
2	Random Forest	0.91892	1.0	0.666667	0.1	{'max_depth': 7, 'n_estimators': 250}
4	XGBoost	0.91892	0.966667	0.714286	0.02	{'learning_rate': 1, 'max_depth': 4}
5	K Neighbors	0.91892	0.911765	1.0	0.0	{'n_neighbors': 6, 'p': 1}
8	SVC	0.91892	1.0	0.666667	0.0	{'C': 1, 'gamma': 'scale', 'kernel': 'linear'}
9	MLP	0.91892	0.966667	0.714286	0.05	{'hidden_layer_sizes': 10, 'learning_rate_init...
7	Gradient Boosting	0.89189	0.909091	0.75	0.05	{'max_depth': 2, 'n_estimators': 400}
3	Bagging	0.86486	0.933333	0.571429	0.04	{'n_estimators': 400}
6	DecisionTree	0.75676	1.0	0.4	0.0	{'max_depth': 6}

The results in this table are sorted by the accuracy score. The model performance is visualized in the plots below for each of the metrics.

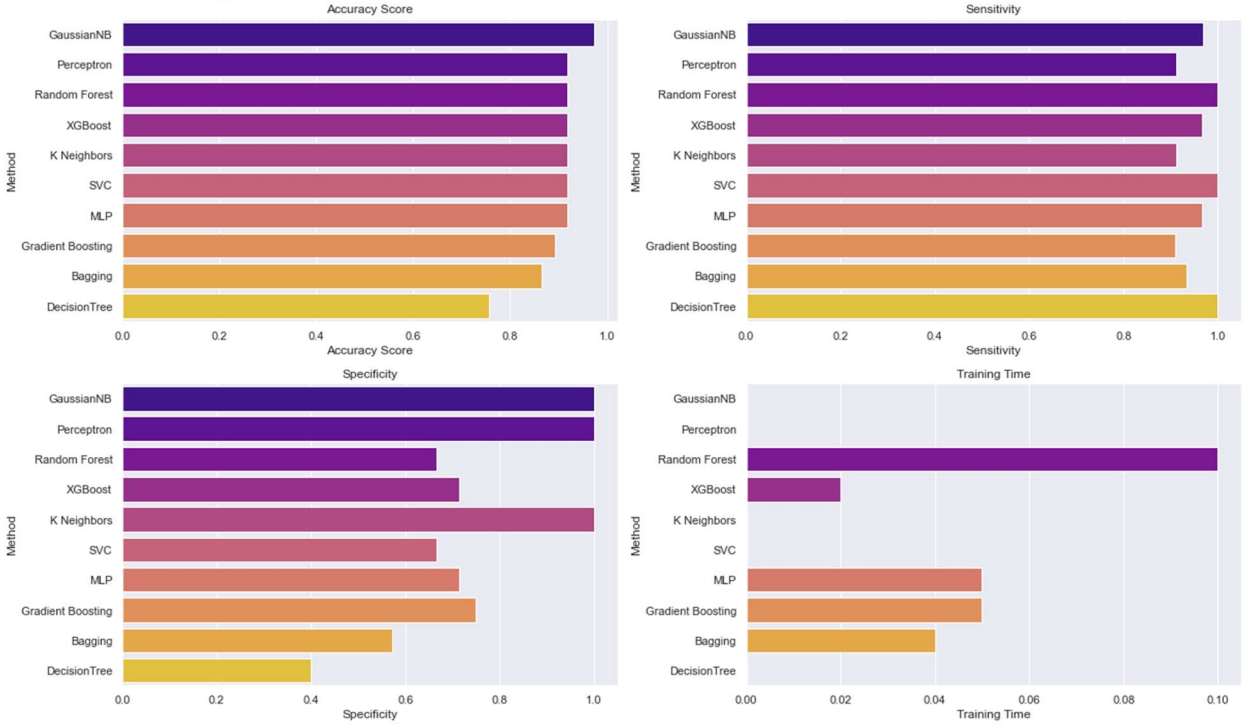

Figure 9. Performance of tested models according to various performance metrics.

From these results, it was determined that the Perceptron, Random Forest, and SVC models were the most reliable and best performing. I ultimately did not use the Gaussian Naïve Bayes, XGBoost, or K Neighbors model due to issues implementing class weights that I did not have time to fix. A closer look into the Perceptron model is shown via the confusion matrix (Figure 10) and the ROC-AUC curve (Figure 11). The confusion matrix shown below was obtained via evaluation of the test set. The confusion matrix shows that the model classified 31 entries as True Positives and 3 entries as True Negatives. There were also no False Positives and there were 3 False Negatives.

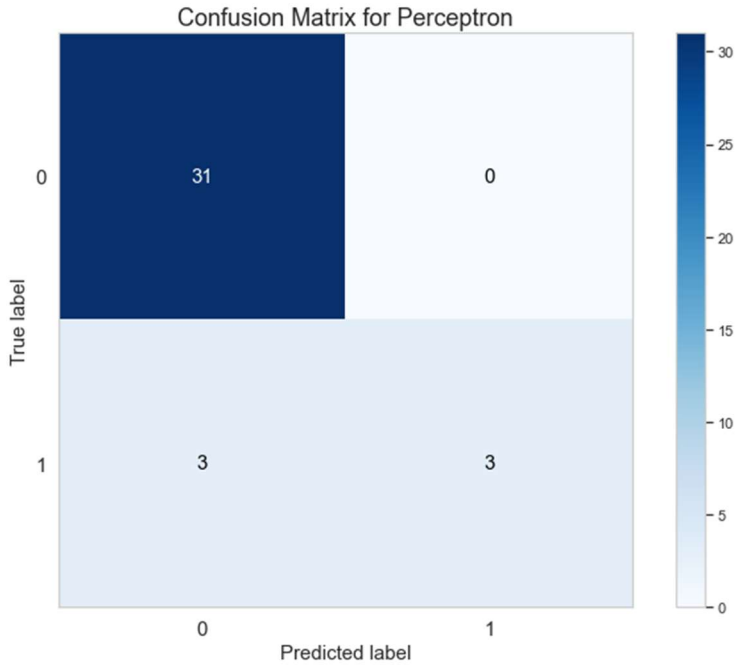

Figure 10. Confusion matrix derived from Perceptron model on test set.

The Receiver operating characteristic curve (ROC curve) and the area under the curve (AUC) for this model are shown below. ROC is a probability curve and AUC represents the degree or measure of separability. The Higher the AUC, the better the model is at distinguishing between patients with the disease and no disease. An excellent model has AUC near to the 1 which means it has a good measure of separability. A poor model has an AUC near 0 which means it has the worst measure of separability. For this model, the AUC was calculated to be 0.75 which is indicative of moderate separability. This could certainly be improved but is a fairly good start.

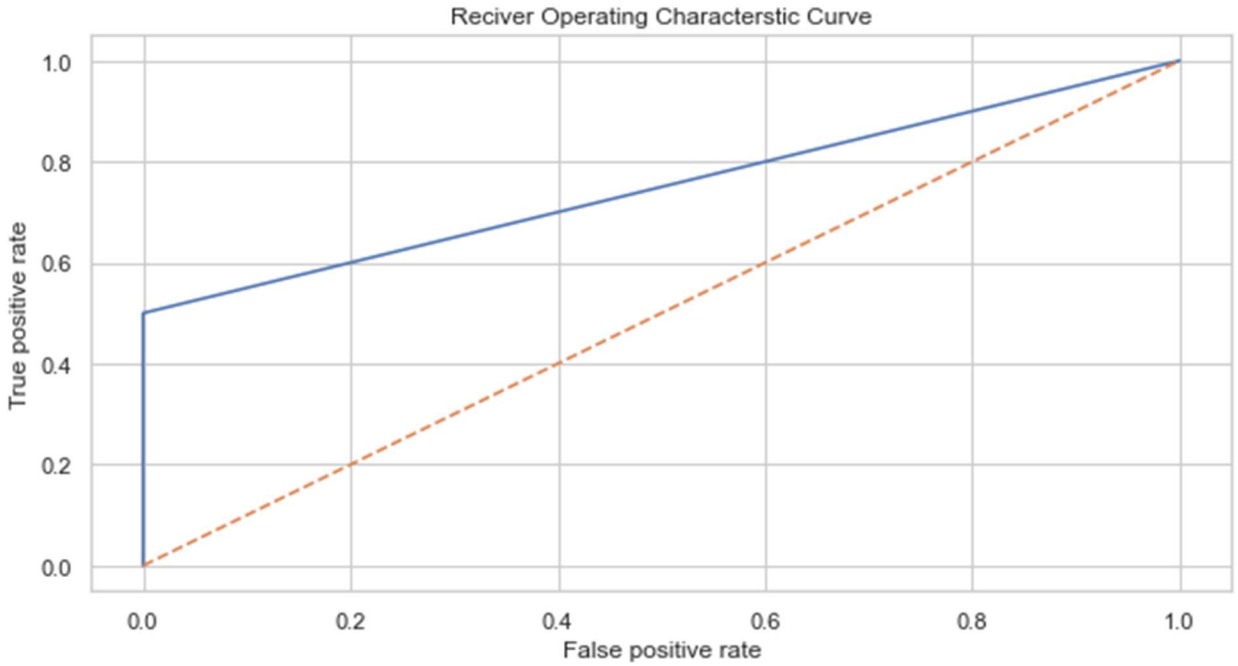

Figure 11. ROC-AUC curve for Perceptron model. The AUC for this model is 0.75.

In addition to testing these models, I constructed my own ensemble model from the top 3 classifiers. However, I didn't have time to refine it nor optimize it further. The code for it can be found in the Jupyter notebook. I also wanted to look into the model explainability, but I unfortunately do not have the time at the moment to explore that further due to a bug in my environment. I wanted to look into the weights of the most important features, to generate Partial Dependence Plots for said features, and look at the SHAP values to better understand how the model is making the decision/prediction.

Conclusion and Future Directions

I performed exploratory data analysis, data cleanup, data visualization, feature engineering, dimensionality reduction/feature selection through ANOVA, model building and model testing. A total of 10 different classifier models were tested and the 3 best performing models were determined to be the Perceptron, Random Forest, and SVC classifiers. The performance of these models was assessed based on their accuracy, sensitivity, and specificity scores. The classification accuracy of these models was determined to be 91.9%.

If I had more time/expertise I would like to first explore the gene data in greater depth/detail. I'd be interested in seeing if there is some clustering analysis I could carry out to identify similarities or perhaps some biology/biochemistry principles I could use to identify relationships between them. I would also like to explore more methods to handle class imbalance more appropriately. I would also like to take the time to optimize these models more and see if I can enhance their separability, recall, sensitivity and so on. I would also be interested in trying to find a way to properly reincorporate the 5 patients I discarded due to them having missing genomic data. I'd also like to try a larger set of models like CatBoost and LBGM.