COVID-19 Variant Classification

Author: Tim Tuite

Date: 11.28.2021

Course: ISYE 7406 – Data Mining and Statistical Learning

Abstract

Coronavirus variants have proven difficult to contain and costly to genetically analyze. This prohibitive cost can limit the understanding of the spread and nature of these variants.

In the US, the CDC has collected demographic and medical information on 30+ million positive tests, and continues to collect this information as the pandemic continues. This data pipeline may be the best tool to determine a given variant's nature and spread in the aggregate and make policy decisions based on that knowledge.

Machine learning classification methods can be used to predict outcomes based on observed data, and infer relationships between those observed features and the outcome. We will build 7 classification models that predict the variant (for the purpose of this analysis, whether it's the Original variant or Delta variant) and infer the relationship between the variant and the observed demographic and medical data.

The 7 models perform similarly, but Extreme Gradient Boosting performs the best with a prediction accuracy of 64.19% and an Area Under Curve (AUC) of 62.43%. For inference, age group (those younger than 20 are especially more likely to contract the Delta variant, while those older than 40 skew towards the Original variant) and pre-existing medical conditions (those with pre-existing medical conditions were more likely to contract the Delta variant, while those without pre-existing medical conditions were more likely to contract the Original variant) have the strongest relationship with type of variant.

These machine learning classification models use observed medical and demographic data to predict the variant and infer relationships between the observed data and the variant. These predictions are better than random guessing and help understand who is more susceptible to certain variants. However, there are many false positives and false negatives that make it difficult to trust the results without more context. These analyses and results can be used as a tool for understanding variant spread in the aggregate and inform targeted public messaging, but would have to be one tool among many for informing macro policy decisions.

Introduction

The coronavirus has evolved since it originally appeared in late 2019. Certain variants are more dangerous, contagious, and prevalent than others at different times and in different regions. However, it is expensive and time consuming to determine which variant is present in a given community. What if we could predict based on other demographic and medical factors which type of variant a person had, and therefore the type of variant spread that exists in a community? And what if we knew which factors lead to certain variant contraction, thereby allowing for targeted testing on and messaging towards certain subsets of the population?

To answer these questions, we will utilize the CDC's COVID19 Case Surveillance Public Use dataset (found here: https://data.cdc.gov/Case-Surveillance/COVID-19-Case-Surveillance-Public-Use-Data/vbim-akqf). Since the beginning of the COVID-19 pandemic in the US, over 653 million tests have been conducted, returning over 49 million positive results. Among these 49 million positive results, the CDC had collected medical and demographic data from 32 million cases (as of 9/30/2021). This dataset has 12 variables. We can group these 12 variables into 3 types of variables:

- Testing information (date of the test, date of first sign of symptoms, whether the test is confirmed or probable).
- Demographic information (sex, age group, race).
- Medical information (whether the person was admitted into the hospital or ICU, whether they died, whether they had any underlying medical conditions).

Because the dataset contains 32 million records, we will do some preprocessing to keep the relevant records for our analysis (10 million records that correspond to the Original and Delta variant timeframe in the US) and then work with a random 1% sample from the preprocessed dataset (100k+ records). We will then split the data into a training dataset that's 75% of the dataset (75k+ records) and a testing dataset that's 25% of the dataset (25k+ records).

We will then train 7 classification models on the training dataset. We will tune those models' hyperparameters using 5-fold cross-validation to select the choice of hyperparameters that minimizes each model's cross-validated training error. We will then measure each model's performance on the testing dataset.

In real world applications, we would re-run the models on the full 100% dataset at the end to determine the true fits and measurements. However, this takes a significant amount of time and does not provide further learning, so the final results in this analysis will be based on that 1% dataset.

We will set the goal of an accuracy of 80%. If our best model can correctly classify 4 out of every 5 tests, we will consider our results successful. There is not a high cost of misclassification since this is meant to understand macro trends and broad policy decisions, so misclassifying 20% of the data as the wrong variant is not too harmful.

Exploratory Data Analysis

This CDC dataset consists of the below 12 variables.

Among the 4 date variables, we keep 1 (cdc_case_earliest_dt) for our initial exploratory analysis, as that is the CDC-recommended best date to use. However, we will see later that the date determines the variant. Therefore, we will remove the date variable during the model training.

We also remove the current_status, since all values are either "probable" or "laboratory-confirmed", but for the purpose of this analysis, we will treat either as a positive test.

Therefore, the variables in **bold** are the explanatory variables we will use in our model training and testing:

Variable	Description
cdc_report_dt	Date case was first reported to the CDC
cdc_case_earliest_dt	The earlier of the Clinical Date (date related to the illness or specimen collection) or the Date Received by CDC
pos_spec_dt	Date of first positive specimen collection
onset_dt	Date of symptom onset
current_status	What is the current status of this person?

sex	**Sex**
age_group	**Age group**
race_ethnicity_combined	**Race and Ethnicity (combined)**
hosp_yn	**Was the patient hospitalized?**
icu_yn	**Was the patient admitted to an intensive care unit (ICU)?**
death_yn	**Did the patient die as a result of this illness?**
medcond_yn	**Pre-existing medical conditions?**

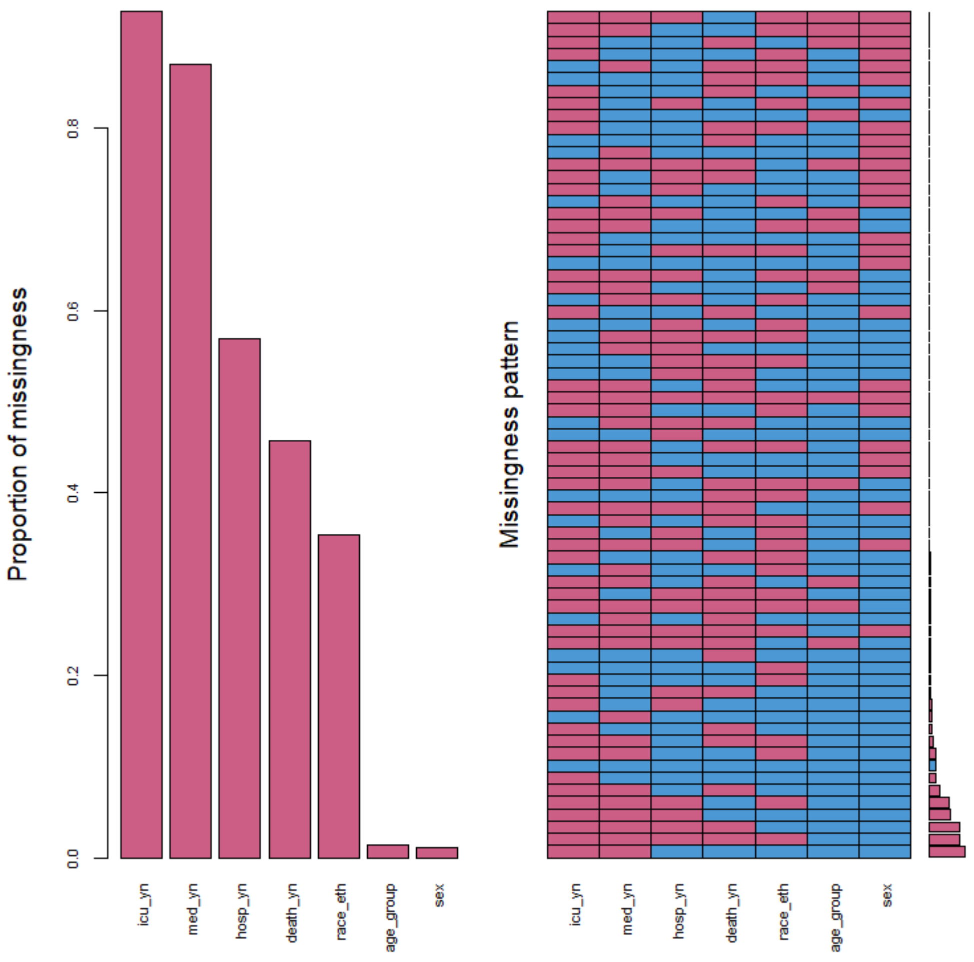
However, there are two main problems with this dataset:

- There are many missing values (96% of observations have missing values).
- There is no information on the COVID-19 variant.

We will address these two main problems as per below:

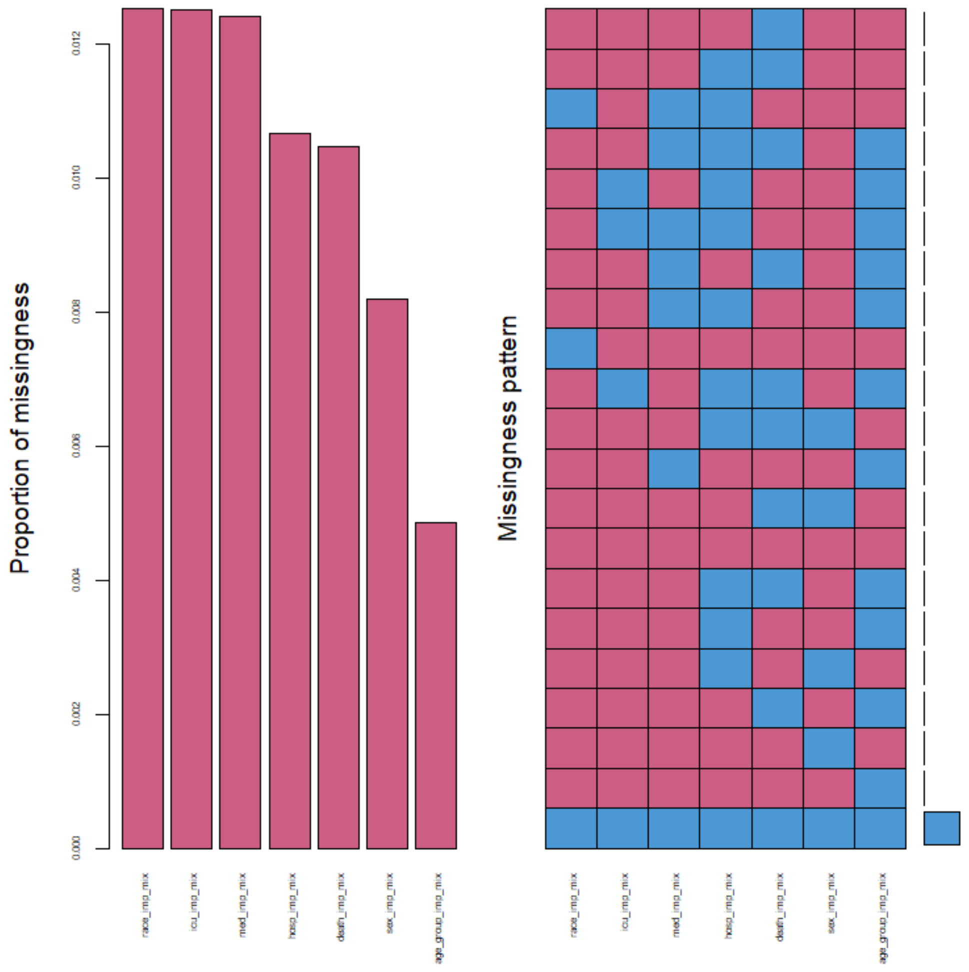
Data Imputation of Missing Variables

To address the missing values, we first must understand what data is missing. We see from the below chart the pattern of missing values pre-imputation. From the left chart, we see the proportion of missing values for each variable. ICU_YN, MED_YN, HOSP_YN, DEATH_YN, RACE_ETH, AGE_GROUP, and SEX all have missing values. We also see reading the right chart from the bottom, the most common arrangement is where an observation is missing both ICU_YN and MED_YN while all others variables are populated.



Next we have to decide how to handle this missing data. Multiple imputation is an often-used technique, but it works under the assumption data is Missing at Random (MAR). More description what this means, and why we believe it is a reasonable assumption, can be found in the Appendix.

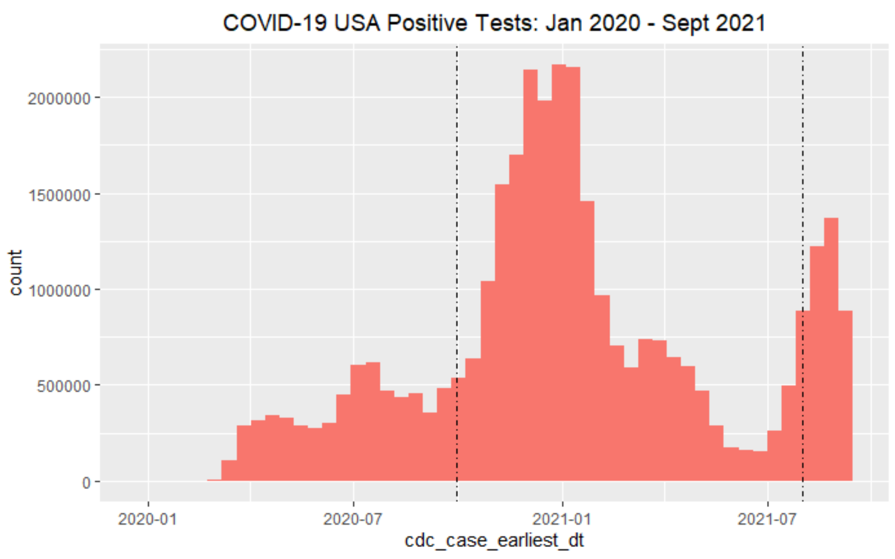
Following that, we will use multiple imputation to impute the missing variables. While imputation techniques are imperfect and constantly evolving, we will use a majority vote method to provide more robust predictions of missing variables. More details on the steps involved in this imputation can be found in the Appendix. Below is the data after imputation, with much fewer cases of missingness (about 96% of observations having missing data pre-imputation down to about 1% of observations having missing data post-imputation):



Creation of Delta Variable

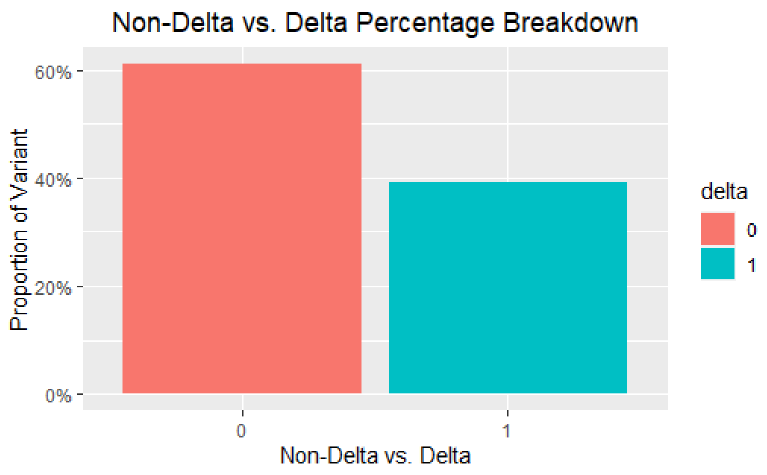
To address the missing variant, we leverage the dates in the dataset to decide which observations are likely to be the Original variant and which are likely to be the Delta variant. We recognize the temporal nature of these observations may affect our results. We will discuss this in our conclusions.

Based on external research, we included observations before 9/30/2020 as belonging to the Original variant, and observations after 8/1/2021 as belonging to the Delta variant. The below image shows the distribution over time:



To quicken analysis, we keep a random sample of 1% of observations. In practice, the full dataset would be used to draw final conclusions. But given system resource constraints and limited added learning of running the same models through the full dataset, we will work with the 1% sample.

To get a better sense of this Original vs. Delta breakdown, we plot a histogram of the two variants. We see overall, the Original variant makes up 61% of the observations, while Delta makes up 39% of observations.



Methodology

To create a model that can accurately predict which variant of coronavirus a person has, we will train 7 classification models on our training dataset, tuning different hyperparameters for the different models to determine which performs the best on the training dataset. We will then measure how well these models predict the correct variant in the test dataset to determine which model is best. At the same time, we will check which variables contribute most to predictions.

Measurements

To measure "how well" the classification is done, we will consider several measurements, since different measurements have different strengths and one measurement does not tell the full story:

- Accuracy: how often the model correctly classifies the variant on the test dataset.
- Sensitivity: how often the model correctly classifies positive cases (in this case, correctly classifying as the Delta variant) on the test dataset.
- Specificity: how often the model correctly classifies negative cases (in this case, correctly classifying as the Original variant) on the test dataset.
- Area Under Curve (AUC): the probability the model will correctly distinguish a positive and negative case (in this case, the chance of correctly classifying Delta and Original variant), where 0.5 is what a noninformative classifier would yield. An AUC above 0.5 indicates the model is better than random guessing.

Models

We will train our data on the below classification models. Logistic Regression will be our baseline model, which is a proven method for classification modeling. All other models will be trying to surpass this baseline model:

- Logistic Regression
- Naïve Bayes
- Random Forest
- Stochastic Gradient Boosting
- Extreme Gradient Boosting
- Single-Layer Neural Network
- Multilayer Perceptron Network

More details on these models can be found in the Appendix.

Prediction Probability Cutoff

In cases where there is a probability prediction, and thus requires a cutoff to classify the prediction as Original or Delta, we will choose 0.5 as the cutoff. Predictions predicting a probability of Delta > 0.5 are classified as Delta. Predictions predicting a probability of Delta < 0.5 are classified as Original. More description why this threshold was chosen can be found in the Appendix.

Hyperparameter Tuning

In each of these cases, we will perform 5-fold cross-validation to determine the best hyperparameters for each model. At a high-level, this means each model has certain variables that lead to different results. We try out different values for these variables, and see which values lead to the best model performance. Those optimal variable values are then used by the final model that we run against the test dataset.

More details of what hyperparameter tuning means in the Appendix.

Variable Importance

It is valuable to understand which of the 7 explanatory variables contribute the most useful information for predicting a variant. For each model, we check which variables are most important, and look for common themes among the models to decide which variables are most related to variant prediction.

Results

Prediction Performance

Below is a summary of each final training model's prediction performance on the testing dataset, with **bold** values being the best model in each category:

Model	Accuracy	Sensitivity	Specificity	AUC
Logistic Regression	0.6411450	0.8985766	0.2339996	0.6215340
Naïve Bayes	0.6033707	0.7208145	**0.4176257**	0.5739785
Random Forest	0.6417706	**0.9043212**	0.2265294	0.6242671
Stochastic Gradient Boosting	0.6412623	0.8987043	0.2341005	0.6217691
Extreme Gradient Boosting	**0.6418879**	0.9040020	0.2273370	**0.6243733**
Single Layer Neural Network	0.6415360	0.9034914	0.2272360	0.6236036
Multilayer Perceptron Neural Network	0.6415360	0.9034914	0.2272360	0.6236036

A few comments on these results:

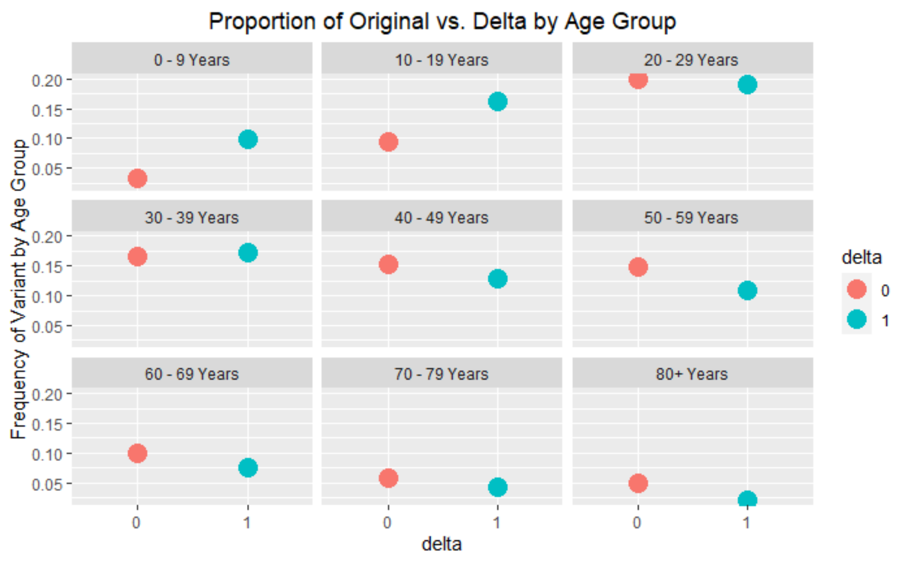
- All models perform similarly, possibly due to data imputation approach or the redundancy and lack of diversity of observations (only 752 unique observations in the training dataset of 76k+).
- When measuring performance based on Accuracy, Extreme Gradient Boosting performs the best (64.19%). In other words, the model correctly predicts the variant as either Original or Delta 64.19% of the time.
- When measuring performance based on Sensitivity, Random Forest performs the best (90.43%). In other words, if the variant is Delta, the model correctly predicts the variant as Delta 90.43% of the time.
- When measuring performance based on Specificity, Naïve Bayes performs the best (41.76%). In other words, if the variant is Original, the model correctly predicts the variant as Original 41.76% of the time.
- When measuring performance based on AUC, Extreme Gradient Boosting performs the best (62.44%). In other words, the model has a 62.44% chance to correctly distinguish a Delta and Original variant patient. The 95% confidence interval does not contain 50%, signifying this model is better than random guessing.

From these results, we will say that Extreme Gradient Boosting with its tuned hyperparameters is the best model for variant prediction.

Variable Importance

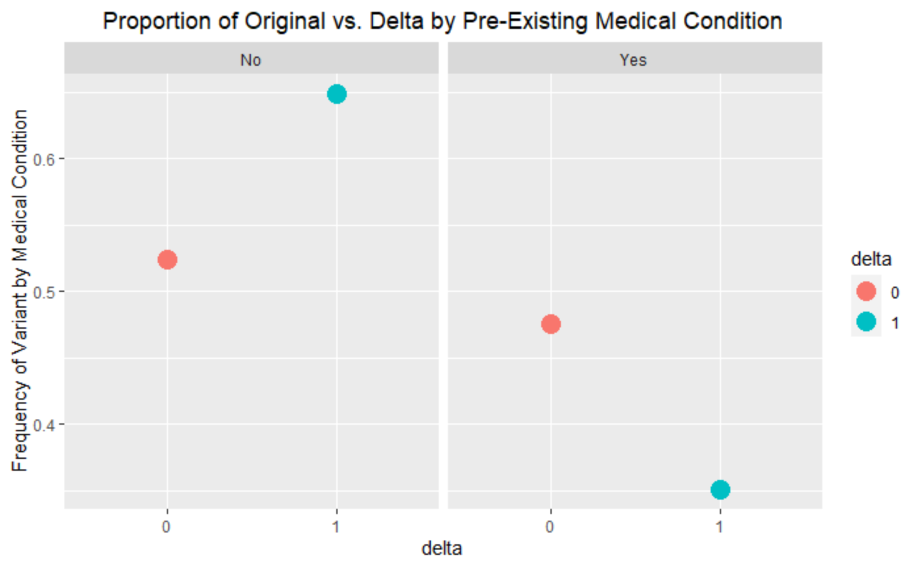
For Logistic Regression, Naïve Bayes, Random Forest, Stochastic Gradient Boosting, Single Layer Neural Network, Multilayer Perceptron Neural Network – **Age Group** is the strongest explanatory variable.

Looking at the training data, Delta is more prevalent in younger age groups, and Original is more prevalent in older age groups. We also see the highest prevalence for both variants is in the 20-29 age group, with about 20% of observations for each variant coming from that group:



For Extreme Gradient Boosting – **Existing Medical Conditions** is the strongest explanatory variable.

Looking at the training data, people with pre-existing medical conditions were more likely to contract the Original variant, while people without pre-existing medical conditions were more likely to contract the Delta variant.



A few comments on the above findings:

- Early in the pandemic, testing was scarce so more tests were done on the older (more vulnerable) population. This can make it look like the Original variant impacted older, less healthy people more, while really those were the people who were more likely to get tested. That being said, we can see the frequency of Delta is low in the older age groups and those with pre-existing conditions, showing Delta is more common in younger, healthier people (especially those < 20 years old) at a time when testing is widely available for all. Therefore, we are comfortable concluding that younger, healthier people are more likely to contract the Delta variant.
- Age group and pre-existing medical conditions are highly correlated, so both results tell a similar story. Generally, younger people have fewer pre-existing medical conditions, so the young, healthy population contracts the Delta variant more often. Meanwhile, older people have more pre-existing medical conditions, so the older, less healthy population contracts the Original variant more often (or at least, as per the above bullet, tests positive for the Original variant more often).

Conclusions

We set out to predict the type of COVID-19 variant a person would contract, and infer relationships between observed variables and the type of variant a person contracts. To do this, we ran the CDC's observed medical, demographic, and testing data through several machine learning classification models, measuring their performance to determine which model performs the best on a previously-unseen dataset.

Ultimately, the 7 models perform similarly, but the Extreme Gradient Boosting classification model performs the best with a prediction accuracy of 64.19% and an Area Under Curve (AUC) of 62.43%. Originally we had set a goal of accurately classifying 80% of the results. Unfortunately, no model came close to that level. While this Extreme Gradient Boosting model is better than a random guess, it is simply one tool in the toolbox for understanding variant spread and the populations at risk.

When reviewing the models for which observed variables contribute most to predicting the variant, age group (especially those younger than 20 being more likely to contract the Delta variant) and pre-existing medical conditions (those without pre-existing medical conditions were more likely to contract the Delta variant) consistently have the strongest relationship with type of variant.

It's possible these findings could shape public messaging, where in regions where it's believed the Delta variant is spreading, governments can tailor public messages towards younger people, encouraging them to adopt spread mitigation strategies like wearing masks and avoiding indoor public gatherings since they are more prone to Delta contraction relative to older people. As new variants like the Omicron variant take shape, testing data can be collected and new models can be designed to understand who is most affected by these variants.

There are a few caveats with these results.

- **Similar prediction performance across models** - Predictions on highly overlapping data (with only 0.98% of observations being unique when considering our key explanatory variables, i.e. there are

752 unique observations in a dataset of 75k+, meaning a given observation will repeat itself on average 100 times in the dataset) yields very similar prediction performance across models. The reason for this overlapping is all the 7 explanatory variables are multinomial, so there's a limit to how many combinations of distinct explanatory variable observations are possible.

- **Similar prediction performance across hyperparameters** - Hyperparameter tuning to choose the optimal hyperparameter for each model does improve the prediction accuracy. But limited variation in the data leads to small differences in prediction accuracy between different hyperparameters. And the size of the dataset leads to slow training times, which limits the grid over which we search for hyperparameters. Therefore the “optimal” hyperparameters are locally optimal over the grid we searched but may not be globally optimal. A broader range of hyperparameters would have to be tuned, preferably on a more powerful machine.
- **Significant data imputation** - Only 3.4% of the original dataset had all explanatory variables populated. The imputation method was thorough but may have introduced bias into the model, as the predictions were based largely on imputed data.
- **Time-based response variable** - While the classification models are trained on 7 explanatory variables to predict 1 response variable, there are likely underlying time-based influences that make the explanatory variables heterogenous, potentially skewing results. This limits the practicality and application of this specific analysis in a few ways:
    - There was very little overlap between circulation of the Original variant and the Delta variant. The Original variant was the dominant strain in the US until Fall 2020, while the Delta variant didn't become the dominant strain until Summer 2021. In reality, if one wanted to know which variant was circulating, the date of the positive test is as good as predictor as any of our other variables. But for the sake of this analysis, we remove the dates when performing our modeling, to simulate the type of analysis medical professionals could do if this time-based relationship is not known beforehand. This methodology may work better where rates of disease prevalence are relatively uniform over time (like cancer or diabetes or heart disease, though obviously would require other datasets, but similar data mining methods could be used).
    - The Original variant circulated through the US at a time when treatments and knowledge of the virus was limited. Therefore, variables like ICU_YN and DEATH_YN, which can be prevented with better and earlier treatment intervention, were influenced by those medical advances that were around when the Delta variant was circulating. Meaning it's difficult to isolate these observed variables from the time in which they were observed. Therefore, the distribution of the explanatory variables would change over time, which limits its usefulness for predicting future variants that may again have different distributions of observed variables. There are time-series classification models, but given the response (Original vs. Delta) is generated based on the date, the date would be too strong a predictor of the variant, so we excluded it to make the analysis more interesting.

In the future, below are areas of improvement or further investigation:

- **Tune more hyperparameters for each model -** Due to performance challenges and limited additional learnings from doing broader grid search of hyperparameter values, most models searched over a grid of 2 or 3 hyperparameters. A more powerful or virtual machine could help with this. Though as stated above, given the similarity of data, there wasn’t much difference in performance across different hyperparameters.
- **Handle the null values differently** – possibly could use different combinations of input variables to predict the missing variables. Or use the same handling, but compare performance on the

imputed model vs. a non-imputed model which drops all unobserved variables. Given only 3.4% of the observations have all data populated, dropping 96.6% of observations would make the non-imputed model very weak, so this may not work. Further research would be needed to understand available options.

- **Apply these methods to the full dataset, not just the 1% sample we used -** Again, a more powerful or virtual machine would help with this. It's unlikely the results would be very different, given the limited variation in data explained before.
- **Explore time-based classification models** - Though as stated above, the date was an overwhelmingly strong predictor of variant since it was derived from it. So more thought would be needed to determine how this could work.
- **Include more variants (especially if new ones emerge, like the Omicron variant)** – this would make it a multinomial classification rather than a binary classification. More time would be needed to gather testing data on these new variants.

While this is based on the coronavirus, if we had similar data for other diseases, potentially these models and others could be leveraged to perform similar predictions on whether or not a person had e.g. cancer or diabetes without expensive testing. However, the prediction accuracy would have to be much better to rely on such a model, especially if it were to be used for individual diagnoses.

Lessons Learned

There were several skills I wanted to explore, and thus why I chose this dataset. This project tested several key data mining and statistical learning skills, including:

- I wanted to analyze an interesting healthcare dataset. This COVID-19 dataset was not my first choice, but was readily available and met other criteria that made it a useful dataset for data mining practice.
- I wanted to explore some of R's performance tools (e.g. vroom for uploading data quickly, data.table for manipulating datasets faster than dplyr/dataframes, miceFast for imputing data quickly). This dataset (32M observations originally, reduced to 75k+ for analysis) forced me to consider performance impacts when making data loading and data manipulation decisions.
- I wanted to practice creating clear and useful visualizations. Several simple yet powerful visuals helped quickly analyze the data and make decisions based on it.
- I wanted to improve my data cleansing and data wrangling skills. Imputing the explanatory variables on a large dataset was the toughest and most time-consuming part of the implementation. Given the size of the dataset, the standard mice package didn't work, which gave me the chance to explore the miceFast package. Likewise the automatic imputation was slow, so I simulated the multiple imputation majority vote method to perform the imputations essentially from scratch. This helped me understand the data imputation process. This also let me practice my usage of data.table syntax to quickly impute many values. While I'm not certain the veracity of the results given the large amount of imputation, it offered a lot of useful learning and practice in an area that was outside the scope of this course.
- I wanted to apply models learned in class (Logistic Regression, Naïve Bayes, Random Forest, Stochastic Gradient Boosting, Single-Layer Neural Network), and explore new models (Extreme Gradient Boosting, Multilayer Perceptron Network).
- I wanted to perform hyperparameter tuning on many different models, to understand the diversity of hyperparameter options across models.

- I wanted to practice my written communication skills, presenting technical data mining results to a non-technical audience.

Appendix

Data Imputation

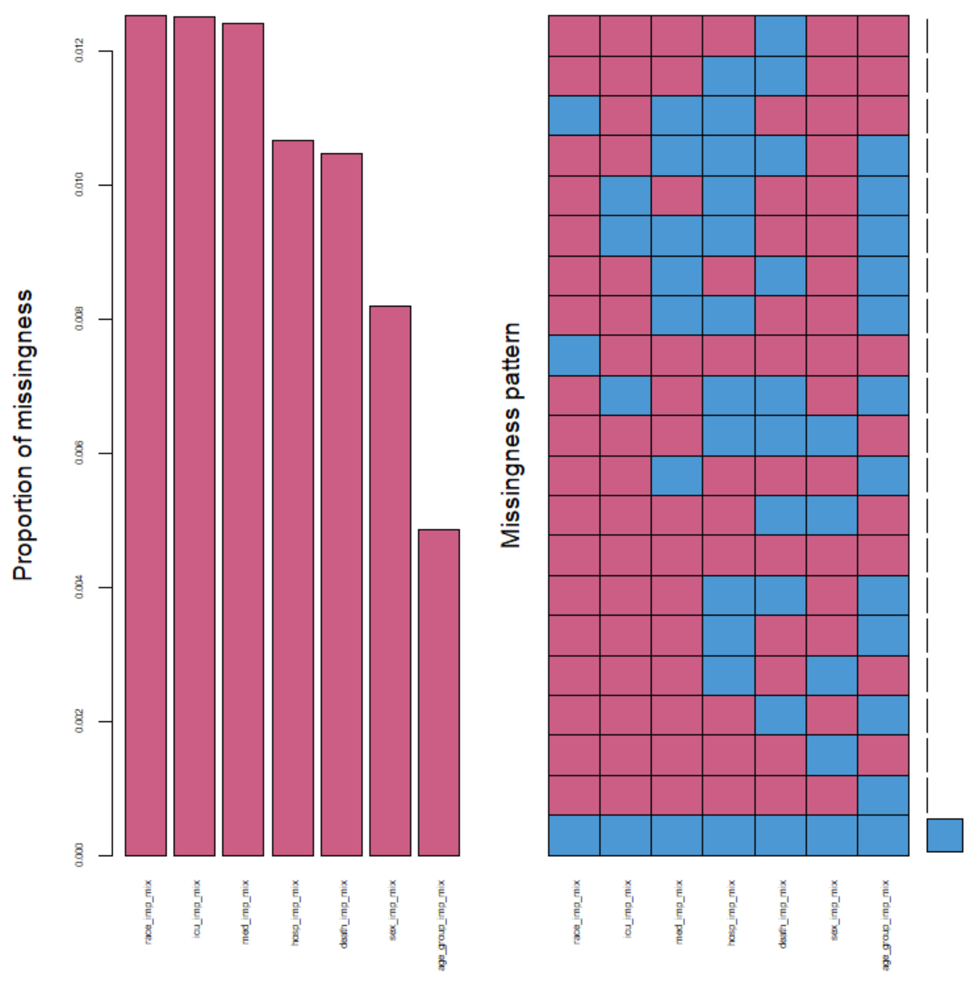
Missing at Random assumption:

Missing at Random (MAR) means any systemic pattern in missing data can be explained by the observed data. In our example, ICU_YN is most commonly missing. For it to be considered MAR, we can posit the missingness of ICU_YN data can be explained by other observed variables like age group and preexisting medical conditions. For example, a person in their 20's with no preexisting medical conditions make that person less likely to be admitted into the ICU. Therefore, a person who is observed to be in their 20's with no preexisting medical conditions could reasonably predict ICU_YN = "No" via this imputation method. While it is difficult to prove that data truly is Missing at Random, we will make that assumption for ICU_YN and the other variables we will impute, as we believe the variables in this dataset are interrelated, and observed values for a given observation can offer clues as to what the corresponding missing variable would be.

Below are the steps involved in multiple imputation:

1. Impute a given variable's missing values using 3 methods that can be used for imputing multilevel data:
    1. Linear Discriminant Analysis
    2. Predictive Mean Matching (using the majority vote of the 5 closest points)
    3. Bayesian Linear Regression (using the majority vote of 5 imputations)

2. Each of these 3 methods will predict a certain value for a given observation's variable. Among these 3 predictions, we take a majority vote. Whichever imputed value is most common among the 3 predictions, we will use that as the final imputed value.
3. Once completed, we see the number of missing variables drops significantly. Now about 99% of the observations have all variables populated. The largest missing variable is race, with just 1.1% of its observations missing. The below chart shows this.
4. Finally, we will remove the incomplete rows (i.e. the 1%+ of observations where there's still a missing value), so that our models can perform predictions without having to deal with missing observations.



<u>General comments on imputation</u>:

For data imputation, it is recognized that it is best practice to split the dataset into train/test first, and then impute the training set, and then train the models, and then impute the test set. Because of the size of the dataset, the fact that these variables are multinomial (so only a defined number of options to predict), and the multiple iterations of imputation being done, we will assume the imputation done on the full dataset upfront provides similar results as imputing the training and test set separately, without creating additional untoward bias.

For variables with little missingness, we'll use both original and imputed value. But for variables with large missingness, we'll use just the imputed value as the predictor, otherwise mice cannot make many predictions with many NA's in the original value. While this may lead to compounding bias (a previous "wrong" imputation will lead to futher "wrong" imputations downstream), other attempts imputed very few values, so we assumed this method of imputation would yield similar prediction and inference robustness as excluding most observations in the dataset.

Models

High-level descriptions of models we are training:

- Logistic Regression - A model that predicts the probability of a certain binary outcome based on certain inputs (in this case, the probability that an observation is the Delta variant based on the demographic and medical information of the patient). This probability is then converted to a binary yes/no (Delta/Original variant) by choosing a cutoff and setting all probabilities above the cutoff to yes/Delta and all probabilities below the cutoff to no/Original. This is our baseline model. Our goal will to beat its performance with the other models.
- Naïve Bayes - A model that assumes all observed variables are independent of each other, and that each of those variables provide clues as to which outcome is most likely. Based on the combination of these independent input variables, the model makes a determination which outcome a given observation represents. In this case, we're assuming all the observed demographic and medical data are independent of each other, and the model predicts whether the variant is Original or Delta.
- Random Forest - A model that builds many decision tree flowcharts, which each predict the outcome based on a couple of the original variables. It then takes the average (for continuous data) or most common (for multinomial data) of those many predictions to make a final prediction. In this case, we create hundreds of these decision trees, each with a subset of 3 of the original demographic and medical variables (different for each tree), each of which predict whether the variant is Original or Delta. Then the final prediction for each observation is based on which variant is most frequently chosen.
- Gradient Boosting (Stochastic and Extreme) - A model that sequentially builds many decision tree flowcharts, where the learnings from an earlier tree are incorporated into future trees. By the end, we have a strong model that's learned from all the previous weak models. In this case, the trees are built sequentially to predict the variant for each observation.
- Neural Networks (Single Layer and Multilayer Perceptron) - An artificial neural network model that makes predictions in stages, with each stage communicating with the previous stage to improve the final prediction. In this case, the neural networks are passing through stages to classify the variant based on the observed data.

There are other common models we didn't use. We didn't use KNN because there were unstable results, so KNN couldn't distinguish between them (many warnings about too many ties in KNN due to very similar data, therefore can't make predictions). We didn't use SVM because it is a very slow trainer. We tried for an even smaller dataset, and its performance was worse than most of the other models. So we didn't use it on the full (1%) dataset we used for training other models.

We also created autoML models using the H2O package, whose best model was a stacked ensemble model composed of 3 deep learning, 2 distributed random forest, 10 GBM, and 1 GLM model. This produced a training data 5-fold CV AUC of 0.6468741, but a testing AUC of 0.609 and an accuracy of 0.5226. This is more balanced, but not as high of accuracy or AUC. We only started using autoML after all other analysis was completed as a last attempt, so limited analysis was done why this performed worse than other models, and its results excluded from the final results.

Prediction Probability Cutoff

We consider a 0.5 cutoff reasonable because of two competing factors that balance out:

- There is a higher cost of a false negative (predicting it's the Original variant when it is Delta) than a false positive (predicting it's Delta when it's original), since the Delta variant can be more contagious and lead to more severe disease, so we would want to quarantine and monitor a person with Delta more strictly (though in either case, quarantining and monitoring would be necessary). This is an argument to make the threshold lower, so that more observations are classified as Delta.
- There is a slight imbalance in the dataset, where there are more original observations than Delta observations (60% vs. 40%). This is an argument to make the threshold higher, so that more observations are classified as Original.

We will say these two factors offset, and that 0.5 is an appropriate threshold.

We also did some empirical analysis, where when fitting logistic regression model, when lowering the threshold to 0.4 to make it more aligned with the distribution of Delta observations in the dataset, it does make the false positives and false negatives more balanced, but reduces the accuracy. Increasing the threshold to 0.6 creates significant imbalance, because then the model too often predicts Original, which increases the sensitivity, but reduces the specificity, and also reduced the accuracy. Therefore, we will use 0.5 as a threshold for all models which predict probabilities rather than directly predicting class values.

Hyperparameter Tuning Concept + Optimal Hyperparameters

Hyperparameter tuning is done by, for each model:

- We will fit a subset of 80% of the training data with a series of hyperparameters
- We will measure the performance of that model on 20% of the training data.
- This will be done 5 times on a rotating subset of the data, and the accuracy of the 5 runs will be averaged. This accuracy will represent the accuracy of the model with the first set of hyperparameters.
- We will then repeat this process for the the next set of hyperparameters to determine the accuracy of the model when using those hyperparameters.
- We will then select the set of hyperparameters that produces the best accuracy on the cross-validated error, and use that as our final training model.
- Later we will apply that final training model on the testing data, and measure the testing accuracy based on that final training model.

When training, below are the optimal hyperparameters for each model. Note the Extreme Gradient Boosting DART version has many hyperparameters, so it extends across 3 rows:

Model	Hyperparameter 1	Hyperparameter 2	Hyperparameter 3	Hyperparameter 4	Testing Accuracy
Logistic Regression					0.6411450
Naïve Bayes	Laplace Correction = 0	Distribution Type = False	Bandwidth Adjustment = 1		0.6033707
Random Forest	# of Variables per Tree = 2	# of Trees = 1000			0.6417706

Stochastic Gradient Boosting	# of Trees = 150	Max Tree Depth = 2	Shrinkage = 0.1	Minimum Terminal Node Size = 10	0.6412623
Extreme Gradient Boosting (row 1 - hyperparameters continue below)	# of Boosting Iterations = 100	Max Tree Depth = 2	Shrinkage = 0.4	Minimum Loss Reduction = 0	0.6418879
Extreme Gradient Boosting (row 2 - continued from above)	Subsample Percentage = 1	Subsample Ratio of Columns = 0.8	Fraction of Trees Dropped = 0.01	Prob. of Skipping Drop-out = 0.95	
Extreme Gradient Boosting (row 3 - continued from above)	Minimum Sum of Instance Weight = 1				
Single Layer Neural Network	# of Hidden Units = 5	Weight Decay = 0			0.6415360
Multilayer Perceptron Neural Network	# of Hidden Units- Layer 1 = 7	# of Hidden Units- Layer 2 = 0	# of Hidden Units- Layer 3 = 0		0.6415360

Bibliography and Credits

Below are several links which I found useful, along with their citations:

https://data.cdc.gov/Case-Surveillance/COVID-19-Case-Surveillance-Public-Use-Data/vbim-akqf

COVID-19 Case Surveillance Public Use Data | Data | Centers for Disease Control and Prevention. https://data.cdc.gov/Case-Surveillance/COVID-19-Case-Surveillance-Public-Use-Data/vbim-akqf. Accessed 26 Nov. 2021.

https://www.cdc.gov/coronavirus/2019-ncov/cdcresponse/by-the-numbers.html

CDC. "Coronavirus Disease 2019 (COVID-19)." *Centers for Disease Control and Prevention*, 11 Feb. 2020, https://www.cdc.gov/coronavirus/2019-ncov/cdcresponse/by-the-numbers.html.

https://www.bmj.com/content/338/bmj.b2393

Sterne, Jonathan A. C., et al. "Multiple Imputation for Missing Data in Epidemiological and Clinical Research: Potential and Pitfalls." *BMJ*, vol. 338, June 2009, p. b2393. *www.bmj.com*, https://doi.org/10.1136/bmj.b2393.

https://covid.cdc.gov/covid-data-tracker/#variant-proportions

CDC. "COVID Data Tracker." *Centers for Disease Control and Prevention*, 28 Mar. 2020, https://covid.cdc.gov/covid-data-tracker.

https://en.wikipedia.org/wiki/SARS-CoV-2_Alpha_variant#:~:text=to%20community%20transmission.-,Spread%20in%20North%20America,in%20Ontario%20late%20December%202020

"SARS-CoV-2 Alpha Variant." *Wikipedia*, 15 Nov. 2021. *Wikipedia*, https://en.wikipedia.org/w/index.php?title=SARS-CoV-2_Alpha_variant&oldid=1055338402.

https://stefvanbuuren.name/fimd/sec-modelform.html

Https://Stefvanbuuren.Name/Fimd/Sec-Modelform.Html. *stefvanbuuren.name*, https://stefvanbuuren.name/fimd/sec-modelform.html. Accessed 26 Nov. 2021.

https://rdrr.io/cran/miceFast/man/fill_NA.html

Fill_NA: "fill_NA" Function for the Imputations Purpose. in MiceFast: Fast Imputations Using "Rcpp" and "Armadillo." https://rdrr.io/cran/miceFast/man/fill_NA.html. Accessed 26 Nov. 2021.

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4082461/

Siddique, Juned, et al. "Binary Variable Multiple-Model Multiple Imputation to Address Missing Data Mechanism Uncertainty: Application to a Smoking Cessation Trial." *Statistics in Medicine*, vol. 33, no. 17, July 2014, pp. 3013–28. *PubMed Central*, https://doi.org/10.1002/sim.6137.

https://www.machinelearningplus.com/machine-learning/caret-package/

"Caret Package - A Complete Guide to Build Machine Learning in R." *Machine Learning Plus*, 11 Mar. 2018, https://www.machinelearningplus.com/machine-learning/caret-package/.

https://machinelearningmastery.com/hyperparameters-for-classification-machine-learning-algorithms/

Brownlee, Jason. "Tune Hyperparameters for Classification Machine Learning Algorithms." *Machine Learning Mastery*, 12 Dec. 2019, https://machinelearningmastery.com/hyperparameters-for-classification-machine-learning-algorithms/.

https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/beginners-tutorial-on-xgboost-parameter-tuning-r/tutorial/

"Beginners Tutorial on XGBoost and Parameter Tuning in R Tutorials & Notes | Machine Learning." *HackerEarth*, https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/beginners-tutorial-on-xgboost-parameter-tuning-r/tutorial/. Accessed 26 Nov. 2021.