Harvard Edx Data Science Capstone - Early COVID Growth Prediction

Billy Tomaszewski

2020-05-31

Contents

1 Overview

1.1 Disclaimer

I am not an epidemiologist or a health official. I am a Biomedical Engineer getting a PhD in Immunology from Duke, and an amateur Data Scientist. Conclusions made within the project, are not to be considered as fact.

1.2 Introduction

At the time of completion of this project the world is in the throws of the COVID-19 pandemic. Over 300,000 deaths have been attributed to COVID-19, with the United States being responsible for nearly a third, as of May 23rd. In order to slow the spread of the virus, many states elected to issue stay-at-home orders to reduce transmission. Stay-at-home orders, among other interventions, have resulted in a "bending" of the case and death curves in the US. At this time, many states are beginning lifting these orders with a wait and see approach to determine if transmission has slowed to a point that the healthcare system can handle.

While it is abundantly evident that COVID-19 has had a profound effect globally and in the US, it has also become clear that some areas were affected worse than others. What factors determine how quickly the virus grew in a given area remain to be totally understood.

The purpose of this project is to firstly complete the Capstone requirement for the Harvard Edx Data Science Specialization, and also to explore an interesting data set for a contemporary issue. These data that will be analyzed is publicly available information related to socio-economic and health data from counties and how they have been affected by the ongoing COVID-19 pandemic. The data was cleaned and aggregated by John Davis JDruns. Information detailing the source of the data, and how it was assembled can be found in John's kaggle notebook link. The data has information on 2,896 counties in the United States (US), and their County-level health and socioeconomic data. Some counties are not included in the data set because of the sparsity of information available for these counties.

Sources for Data Analyzed

1. New York Times - case and fatality data
2. 2016 CDC Social Vulnerability Data
3. 2020 Community Health Rankings Data *The bulk of the socio-economic and health data comes from here, and their methodology and definitions for these metrics can be found here

These data will be used to visualize how COVID-19 has affected counties across the US, define an outcome that will be predicted with machine learning, and validate the model we generate.

1.3 Success Criteria

Since there are no specific success criteria defined for this project beyond demonstrating learning, the following criteria will also be added. Broadly speaking, success will be defined as being able to stratify counties based off an important COVID-19 related outcome, and being able to successfully predict that outcome from the available socio-economic and health data.

1. Demonstrate understanding of Data Science for the purpose of the Capstone Project
2. Define outcome to be predicted by Machine Learning Algorithm
3. Build and optimize model for predicting outcome
4. Identify factors that were important in driving accuracy of the model
5. Assess Accuracy of the model

2 Analysis

2.1 Data

This section will primarily focus on getting a high level view of the data.

Counties	Days_of_Data	Earliest_Date	Latest_Date
2896	115	2020-01-21	2020-05-14

These data contain information for 2,896 counties over 115 days starting in January, and ending in mid-May. The first confirmed COVID-19 related death was widely considered to be on Feb.29th, but it was later

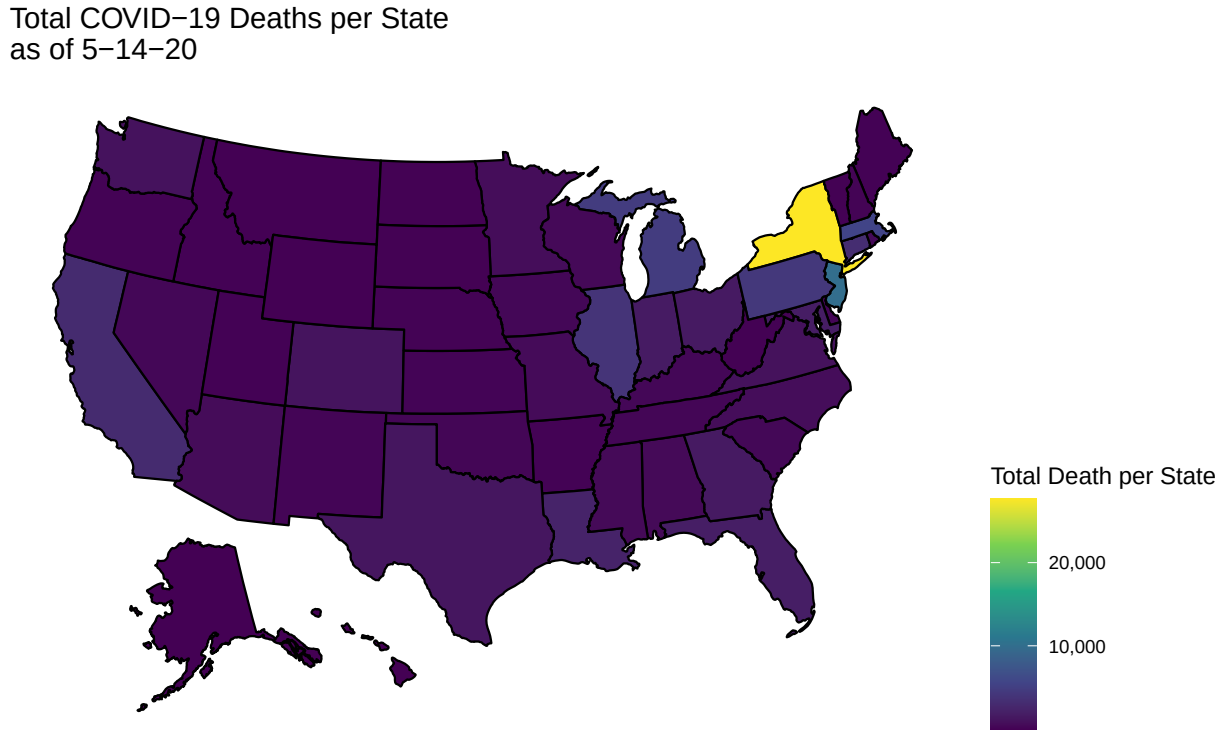
determined to have been as early as Feb. 6th.

Additionally, the data contains over 200 county level factors constituted by the socio-economic and health data previously mentioned.

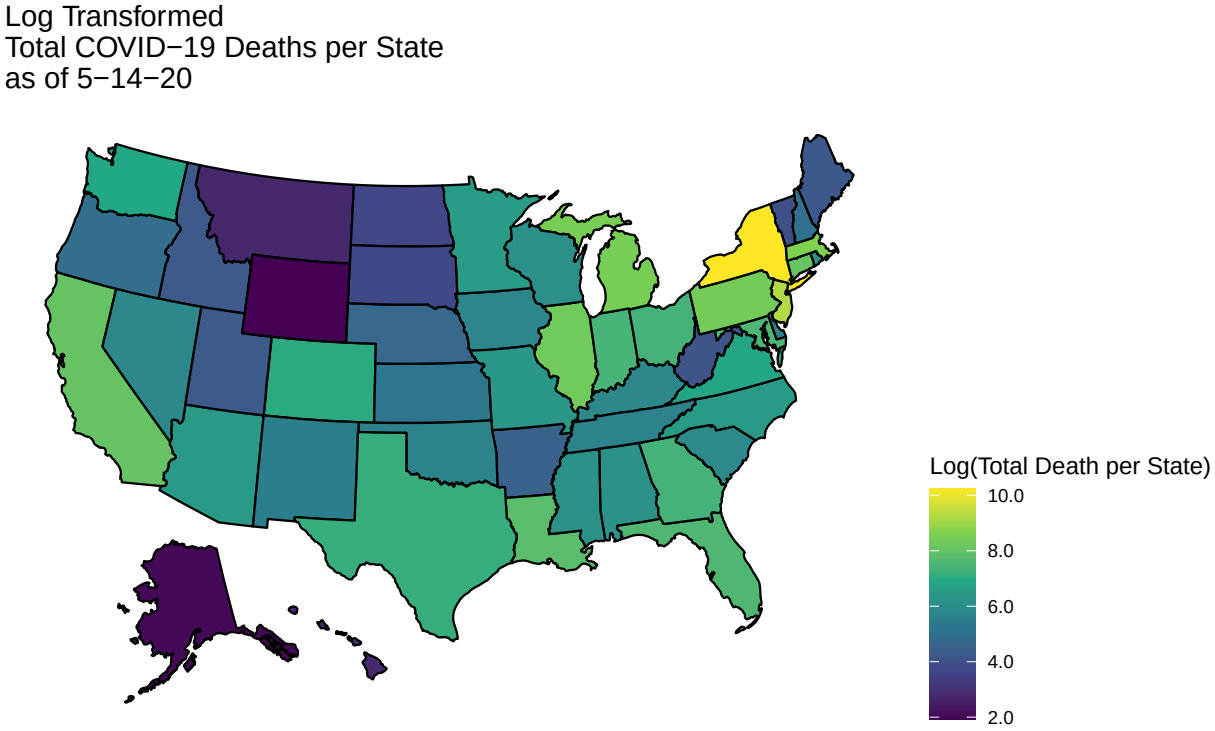
For this analysis, only COVID-19 fatality data will be considered, as the case data is confounded by a lack of uniform testing methodology as well as highly variable testing rates.

2.2 Data Visualization

The first visualization we will make from the data is the total COVID-19 fatalities per state.

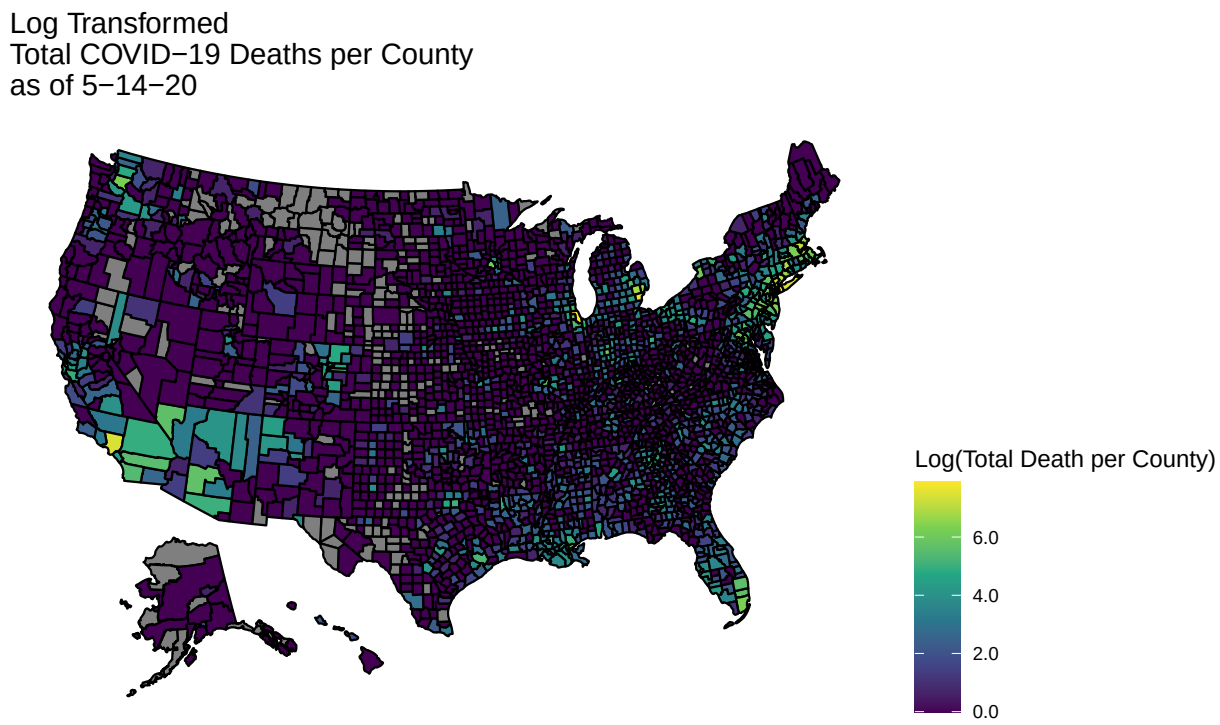


This visualization tells us that we will likely need to transform the fatality data to appreciate the differences. Below is the same visual, but the total deaths per state have been log transformed.



From this visual, we can conclude that Northeast has been hit the hardest in terms of total deaths.

Because these data include county-level data we can produce the same type of visual on a per-county basis.



This visual shows where the hot-spots are in a more granular view. Additionally it is clear that there are many counties that haven't experienced any deaths yet.

Deathless_Counties	Percent_Deathless
1284	0.4433702

Surprisingly, over 40 percent of the counties with data have yet to have a COVID related death. This is important for model building, because if we try to predict deaths, roughly 60 percent of the data will not have a death outcome to predict.

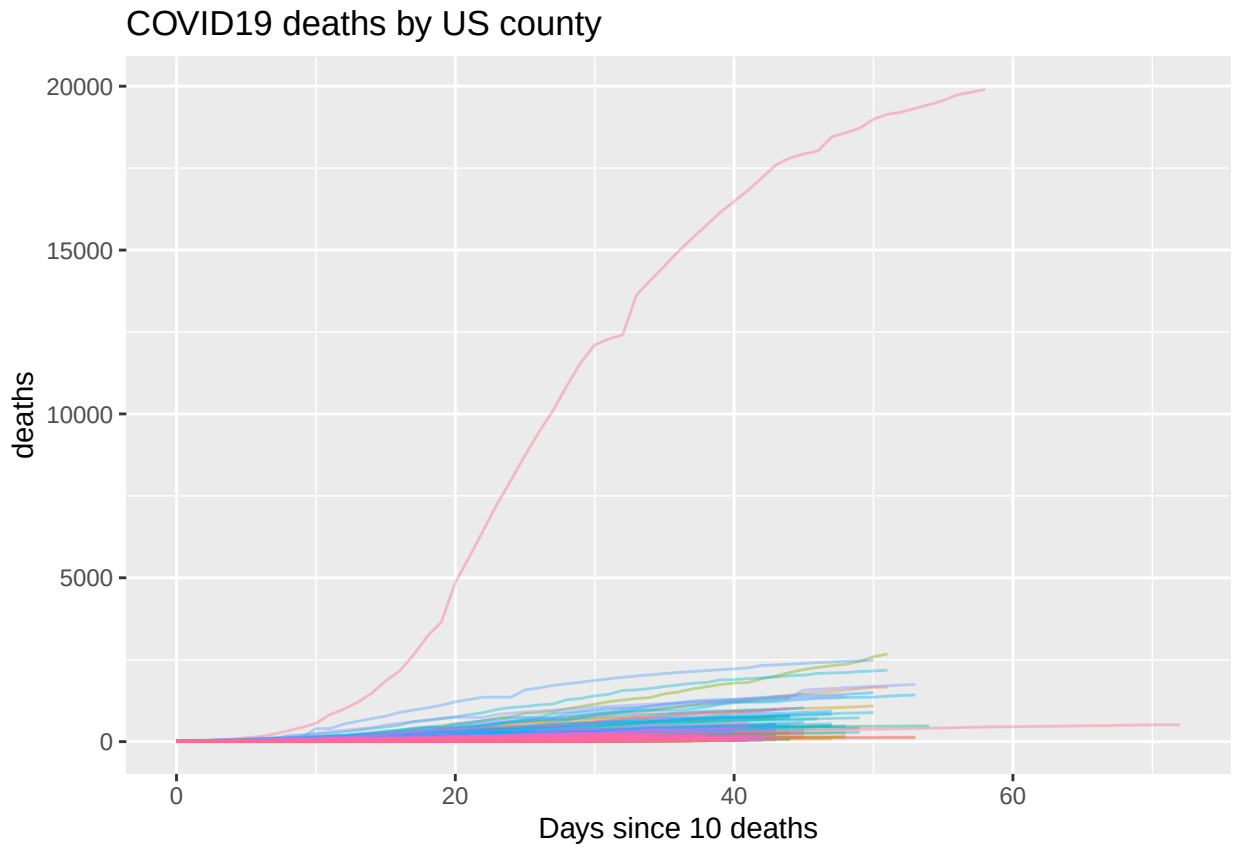
It is worth noting that this is not because the death data is missing, as there are zero missing values. Although there were several counties that weren't found in the dataset, likely due to missing data. The missing counties are displayed as grey in the above map.

```
#missing value check
sapply(red_covid,function(x)sum(is.na(x)))
```

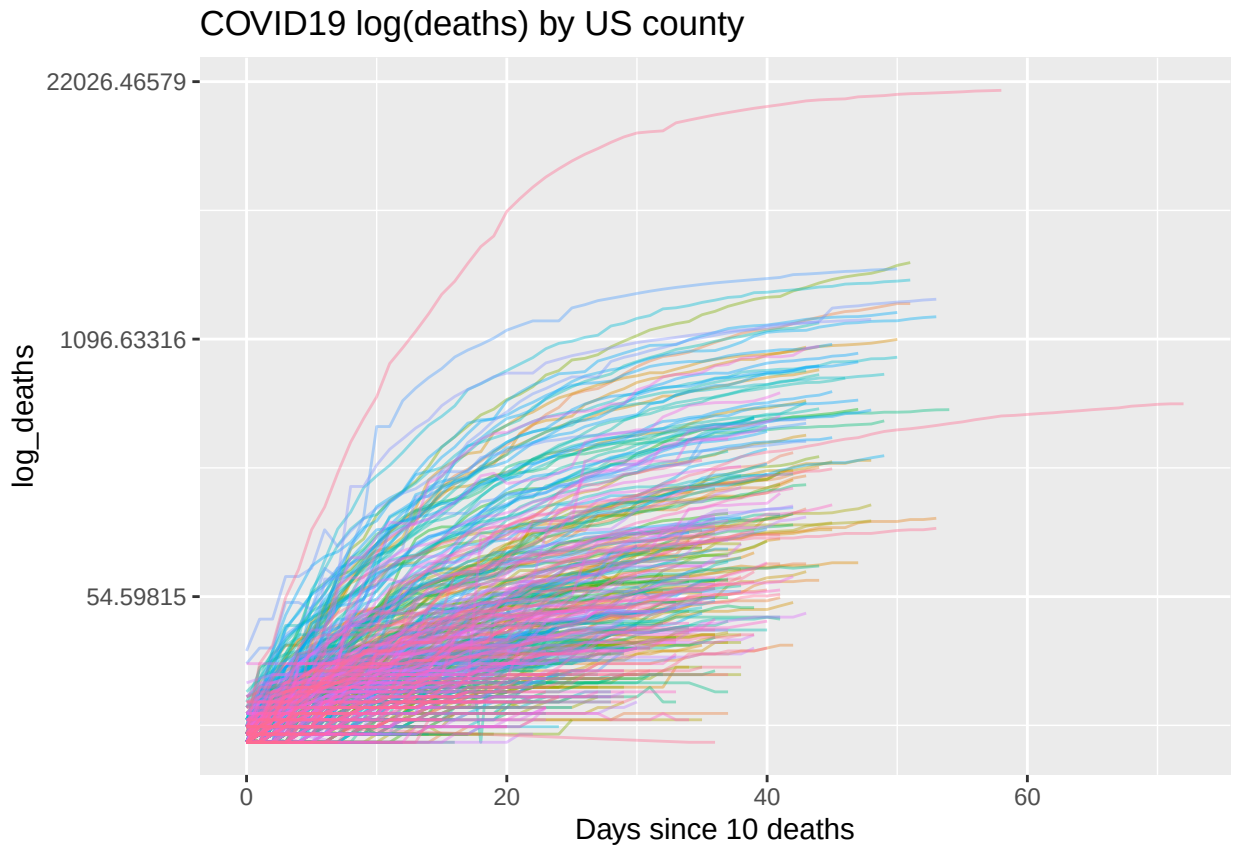
```
##                    date                  county                   state
##                       0                       0                       0
##                    fips                   cases                  deaths
##                       0                       0                       0
## stay_at_home_announced stay_at_home_effective                     lat
##                       0                       0                       0
##                     lon        total_population              area_sqmi
##                       0                       0                       0
```

2.3 Outcome Definition

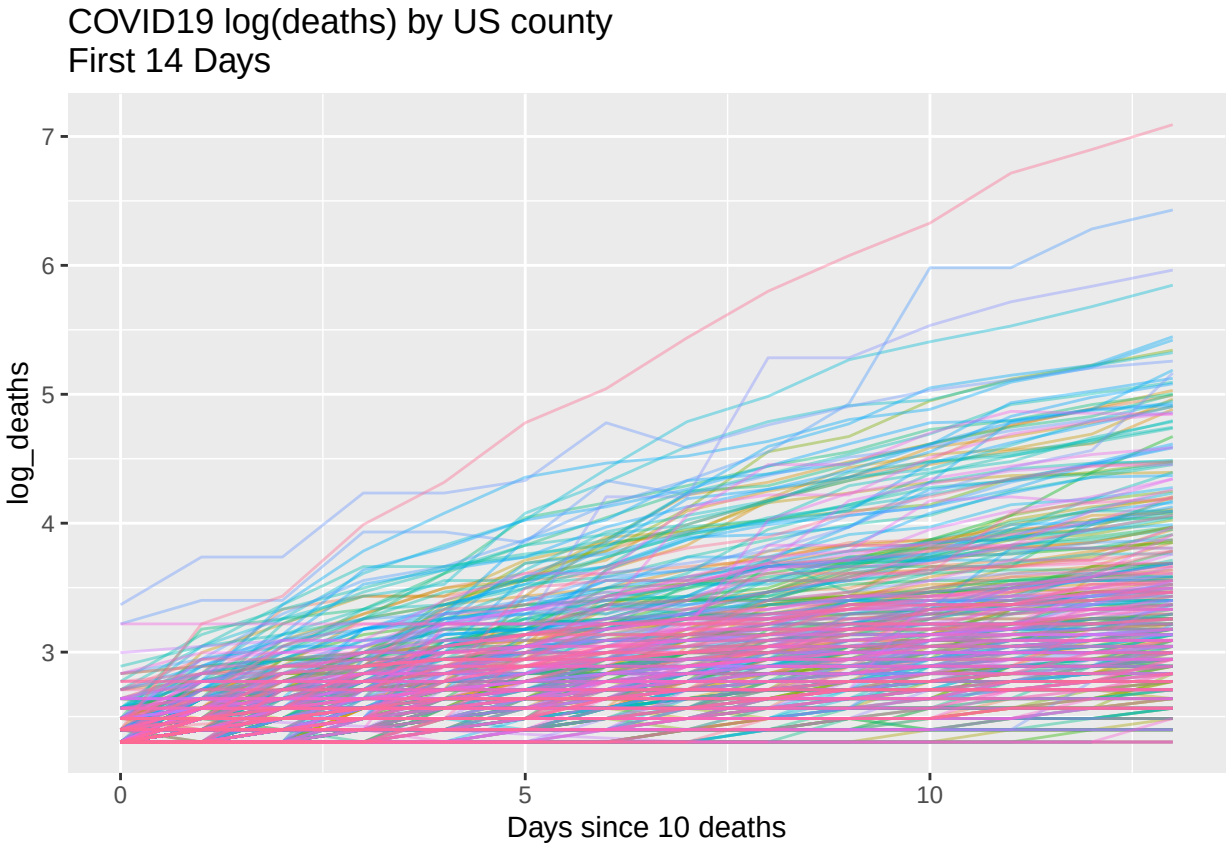
Because the county data has more observations, as there are more counties than states, we will define our outcome using this variation of the data set. Below is a visualization of how COVID-19 deaths have grown over time in counties that have documented more than 10 COVID-19 deaths. For the purpose of this project, less than 10 deaths in a county over 115 days observed will be considered negligible growth.



This visual tells us that the death counts over time will also need to be log transformed to be meaningfully visualized. The log transformed graph is below.



From this visual we can conclude that around 20 days after a counties 10th death, many of the curves leveled off. Interestingly, many of the log(death) curves are mostly linear for the first 14 days.



This suggests that we can perform linear regression, and the slope coefficient of the fitted line will describe the growth rate for each county, with higher slopes meaning faster COVID-19 fatality growth.

In order to determine if our assumption of linearity was correct we can also calculate the R^2 for the linear fits for each county.

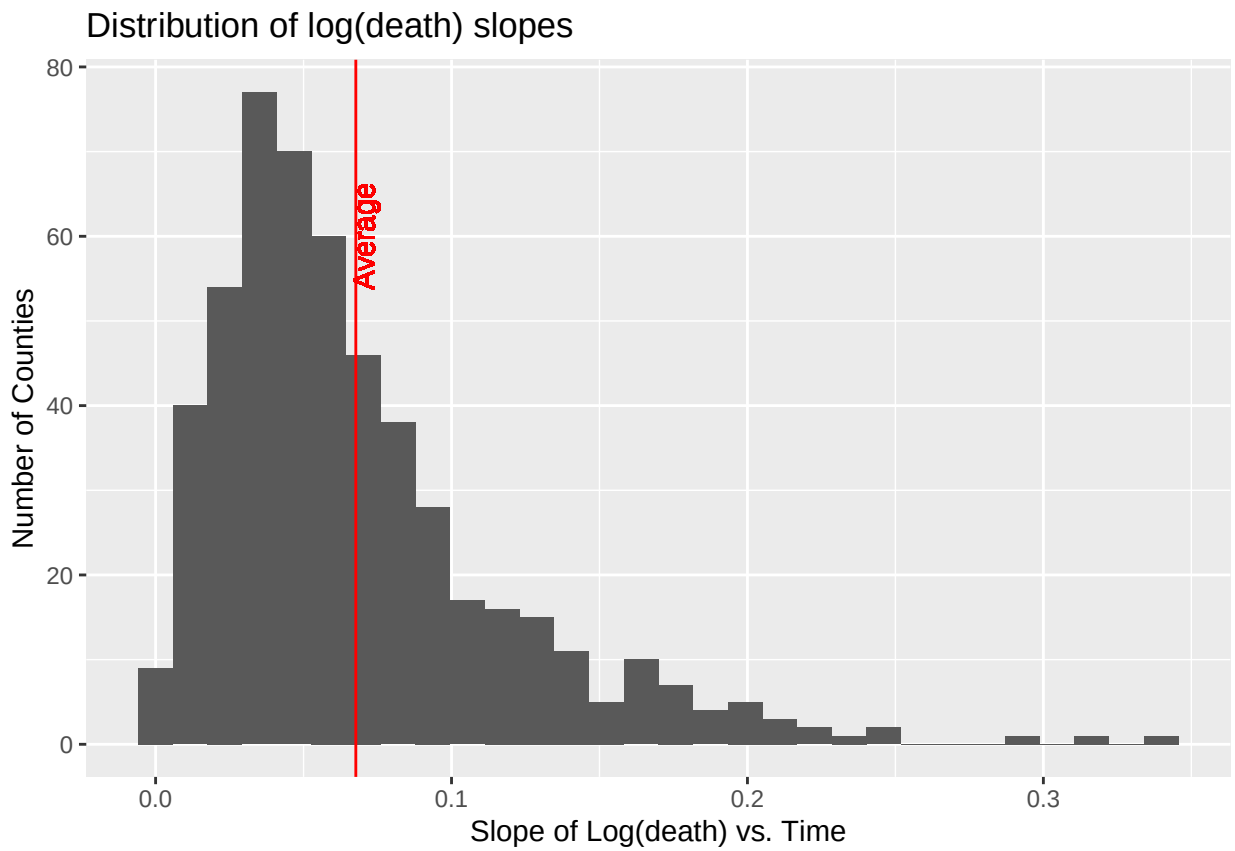
The average R^2 for the linear fits is 0.863265, which is suitable for the purpose of this project.

R^2 is calculated by the following equation.

$$R^2 = 1 - \frac{\sum y_i - \hat{y}_i}{\sum y_i - \overline{y}_i}$$

Where y_i is the actual value, $\hat{y}_i$ is the predicted value, and $\overline{y}_i$ is the mean of y values.

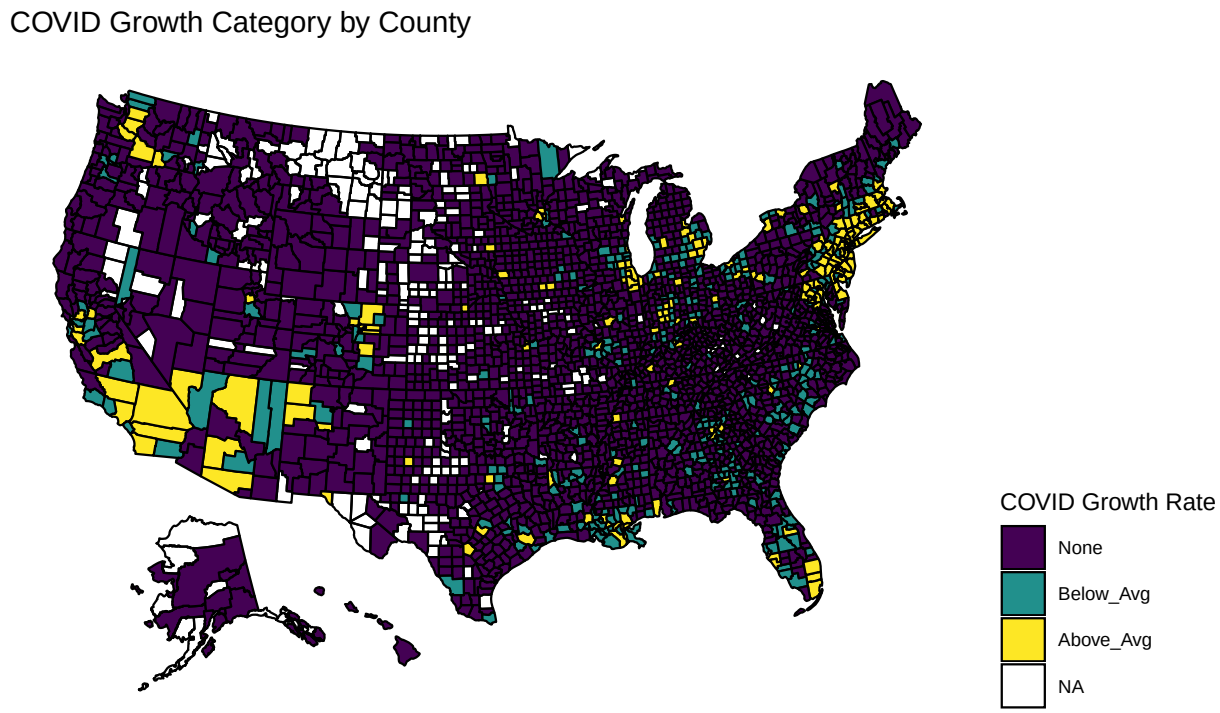
By looking at the distribution of the slopes, it appears that the distribution is relatively normal, although right skewed.



This distribution also shows that most counties have a relatively slow growth, with some outliers that are quite fast.

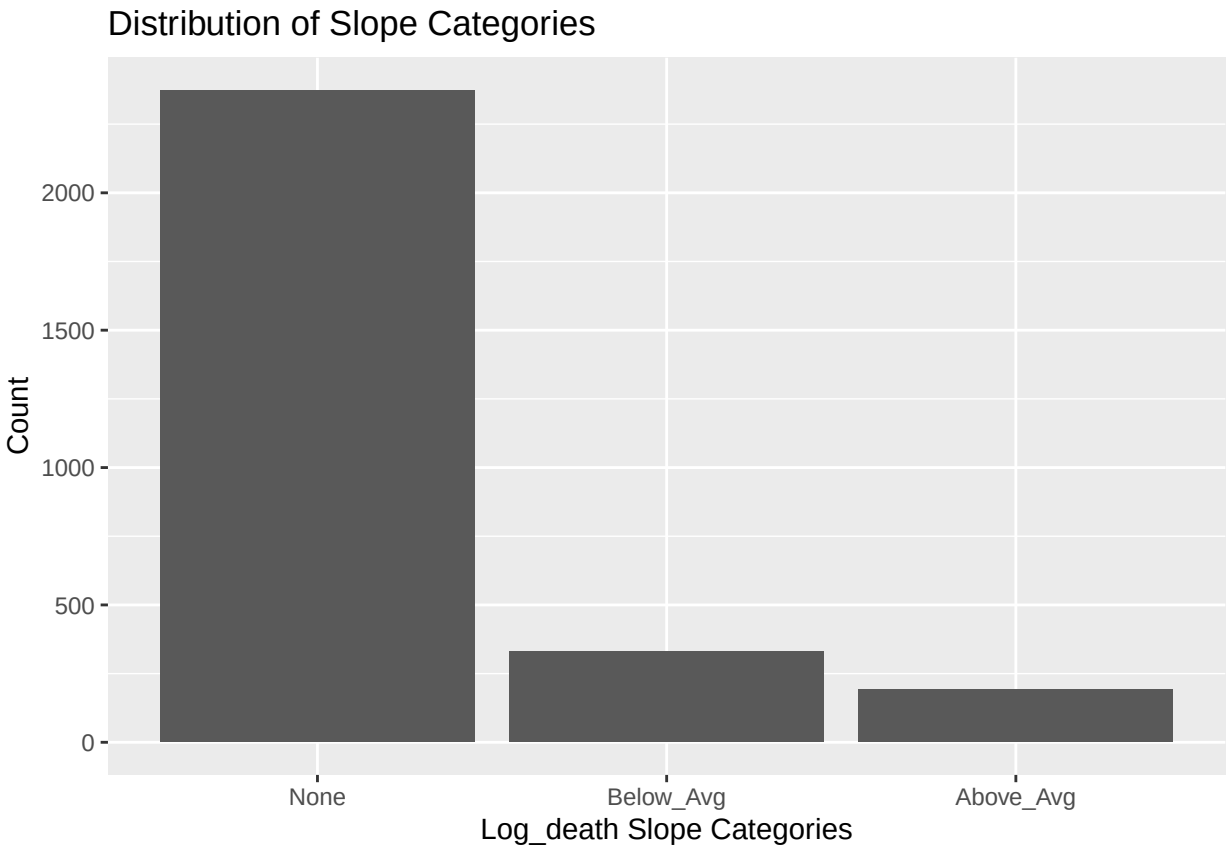
We can categorize the slopes as having no growth, below average growth, and above average growth. With no growth being defined as having fewer than 10 deaths, or a negative slope.

This is what the those categories look like plotted on the county map.



The counties with the top 10 growth rates were primarily the ones people heard on the news. `fct_slp` is the variable name being used to store the growth category data.

state	county	slope	fct_slp
New York	New York City	0.3397697	Above_Avg
New York	Westchester	0.3184947	Above_Avg
New York	Nassau	0.2894984	Above_Avg
New Jersey	Essex	0.2448402	Above_Avg
Michigan	Wayne	0.2442922	Above_Avg
Illinois	Cook	0.2344475	Above_Avg
New Jersey	Bergen	0.2199243	Above_Avg
New Jersey	Union	0.2181669	Above_Avg
Pennsylvania	Philadelphia	0.2141607	Above_Avg
Michigan	Oakland	0.2140561	Above_Avg



Again, from the above graph it becomes evident that the majority of counties experienced negligible or no growth.

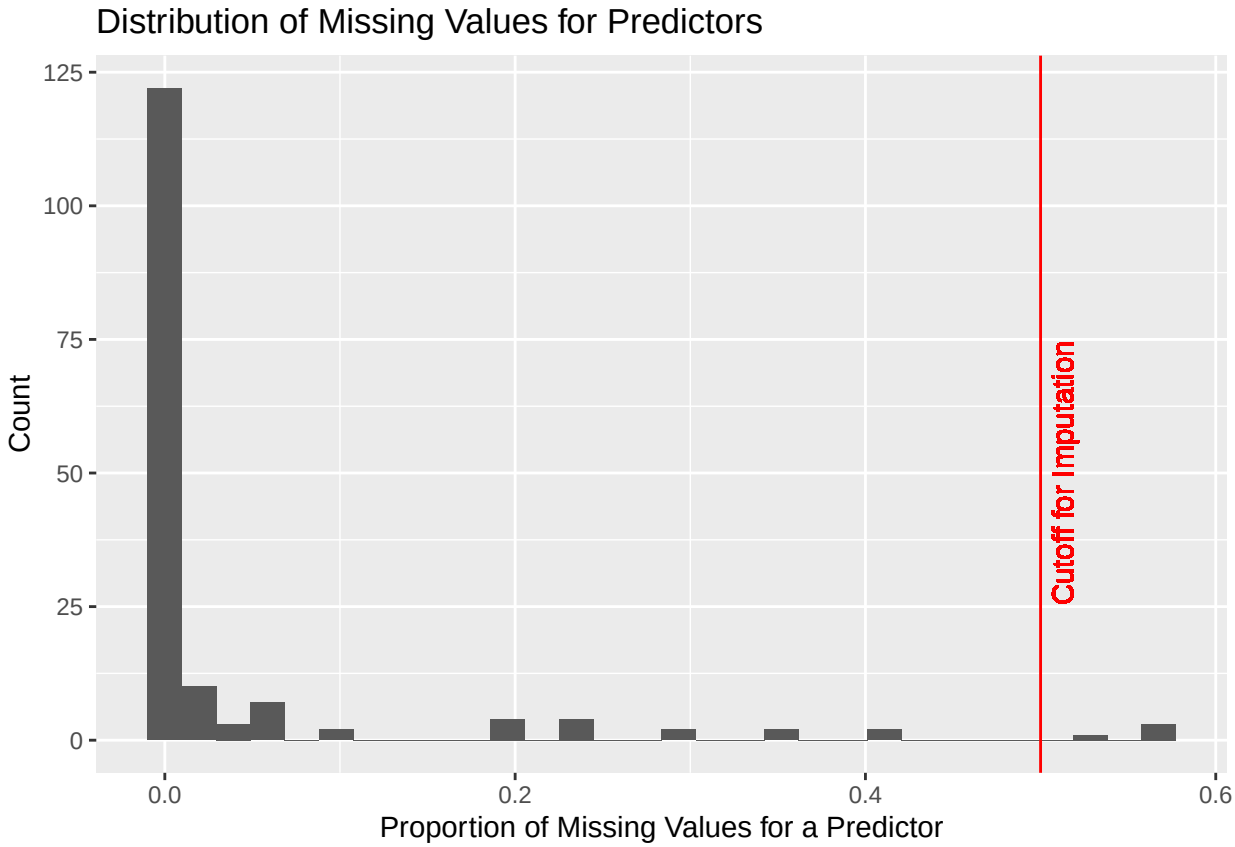
fct_slope	n	frequency
None	2373	0.8194061
Below_Avg	331	0.1142956
Above_Avg	192	0.0662983

Approximately 80% of the data has little to no growth, and 20% experienced growth.

These categories will serve as the outcomes that we will try to predict with the features that are available in the data.

2.4 Data Cleaning

While there was no missing data in the cases and fatality data there are missing values in the predictor/feature



data.

For this data, anything with more than 50 percent missing data, will be considered too sparse to impute.

2.5 Feature Engineering

2.5.1 kNN Imputation

kNN imputation is a method for restoring missing data. The maintainer of the `caret` package describes it this way, "k-nearest neighbor imputation is carried out by finding the k closest samples (Euclidean distance) in the training set. Imputation via bagging fits a bagged tree model for each predictor (as a function of all the others)". More can be learned about kNN imputation in this peer-reviewed article.

Euclidean distance is defined as:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

More information on Euclidean Distance can be found here

Part of the kNN imputation includes normalizing the predictors. Here Normalization means, for each predictor a transformation is applied so the resulting mean is 0 and the standard deviation is 1.

2.5.2 Near Zero Variance

Feature with near-zero variance will be uninformative, and will bog down the machine learning algorithm.

```
#checking for features with near zero variance
nzv <- nearZeroVar(impute_dat_clean, saveMetrics = T)
```

```r
#Finding how many features hav near zero variance
sum(nzv$nzv)
```

```
## [1] 0
```

Fortunately, none of the predictors have nearly zero variance, so they all will be maintained.

2.5.3 Multicolinearity - Removing Highly Correlated Features

A simple definition of Correlation is when two or more features vary together. The formula is:

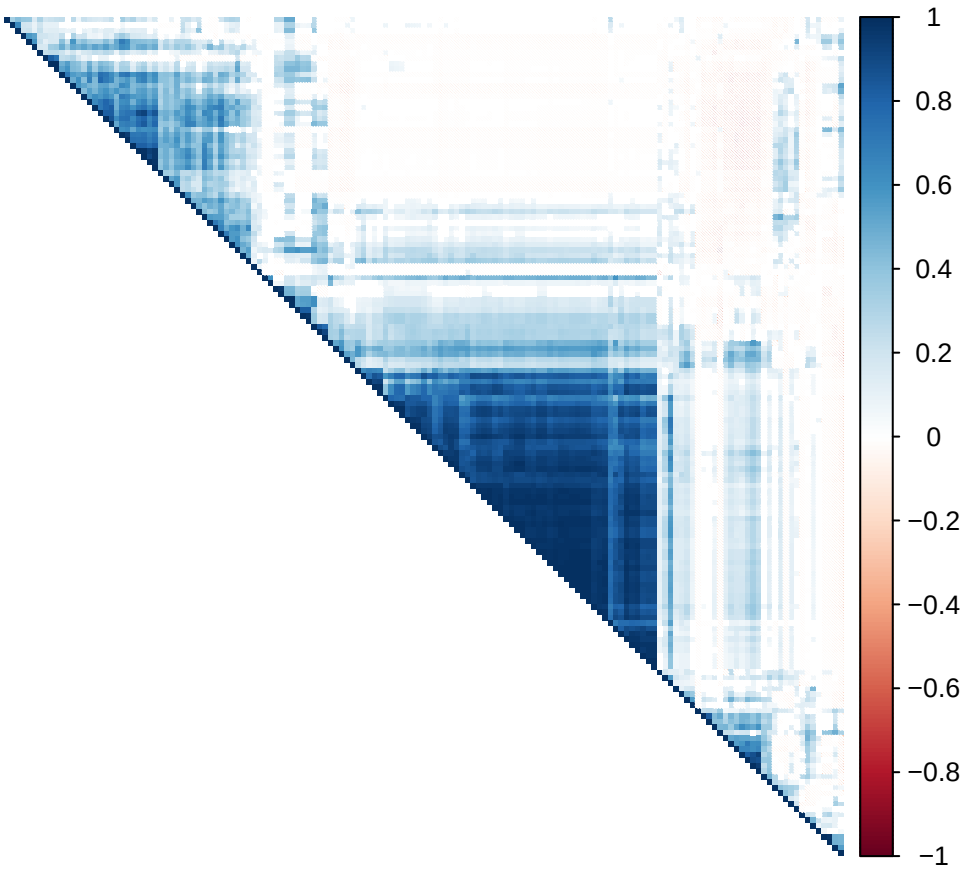
$$cor_{x,y} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2 \sum_{i=1}^{n}(y_i - \overline{y})^2}}$$

Removing one of the two features that correlate together, reduces the importance of that pair of features. For the purpose of identifying features that were most important for generating an accurate prediction, removing the highly correlated features will be important.

This is a summary of the correlation data:

Correlation
Min. :-0.99774
1st Qu.:-0.08495
Median : 0.06181
Mean : 0.14288
3rd Qu.: 0.29219
Max. : 0.99979

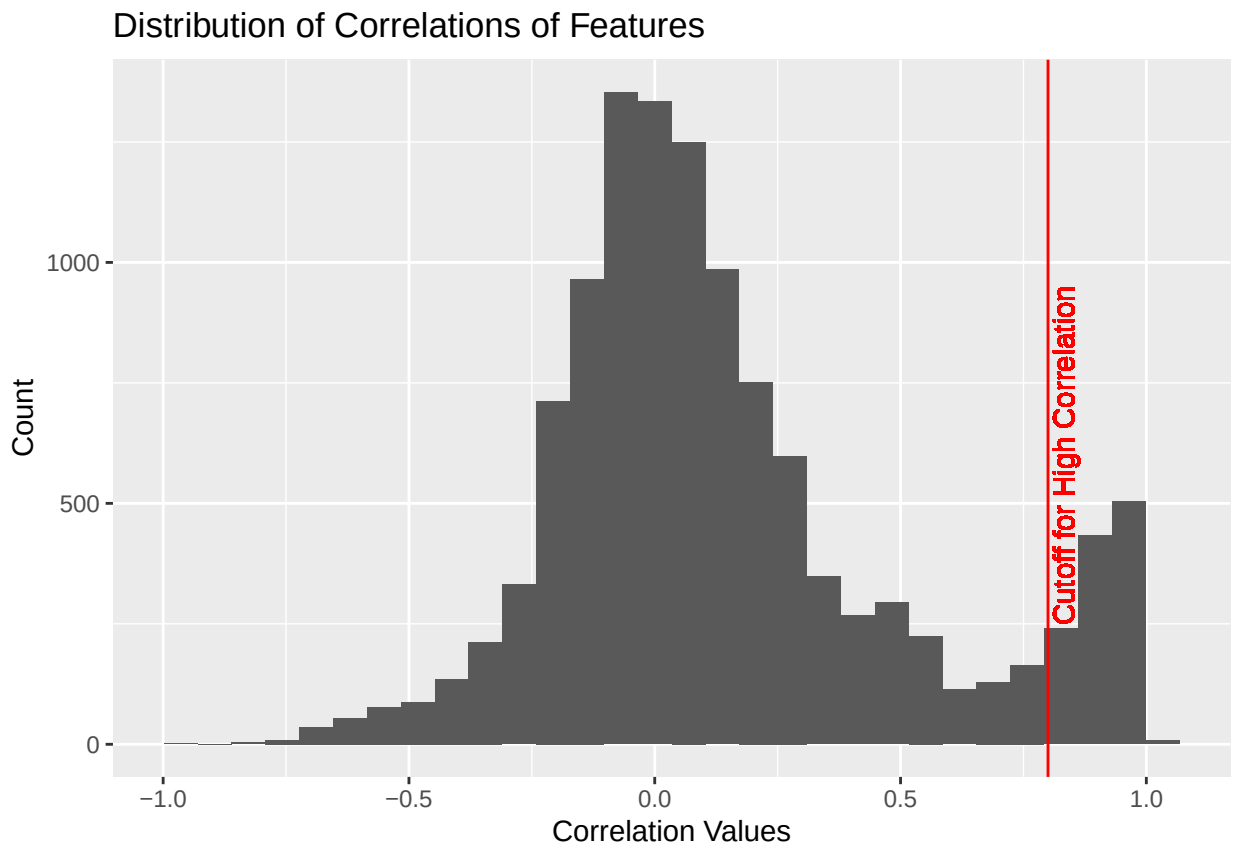
This can also be visualized in a heat-map.



Here is a table of the features that are most strongly correlated:

row	column	cor	p
total_population	num_deaths	0.9757081	0
percent_fair_or_poor_health	average_number_of_physically_unhealthy_days	0.8901026	0
percent_fair_or_poor_health	average_number_of_mentally_unhealthy_days	0.7785234	0
average_number_of_physically_unhealthy_days	average_number_of_mentally_unhealthy_days	0.9338153	0
average_number_of_physically_unhealthy_days	percent_smokers	0.8115085	0
average_number_of_mentally_unhealthy_days	percent_smokers	0.7620586	0
total_population	num_alcohol_impaired_driving_deaths	0.8493999	0
num_deaths	num_alcohol_impaired_driving_deaths	0.8807518	0
total_population	num_driving_deaths	0.8967743	0
num_deaths	num_driving_deaths	0.9215324	0

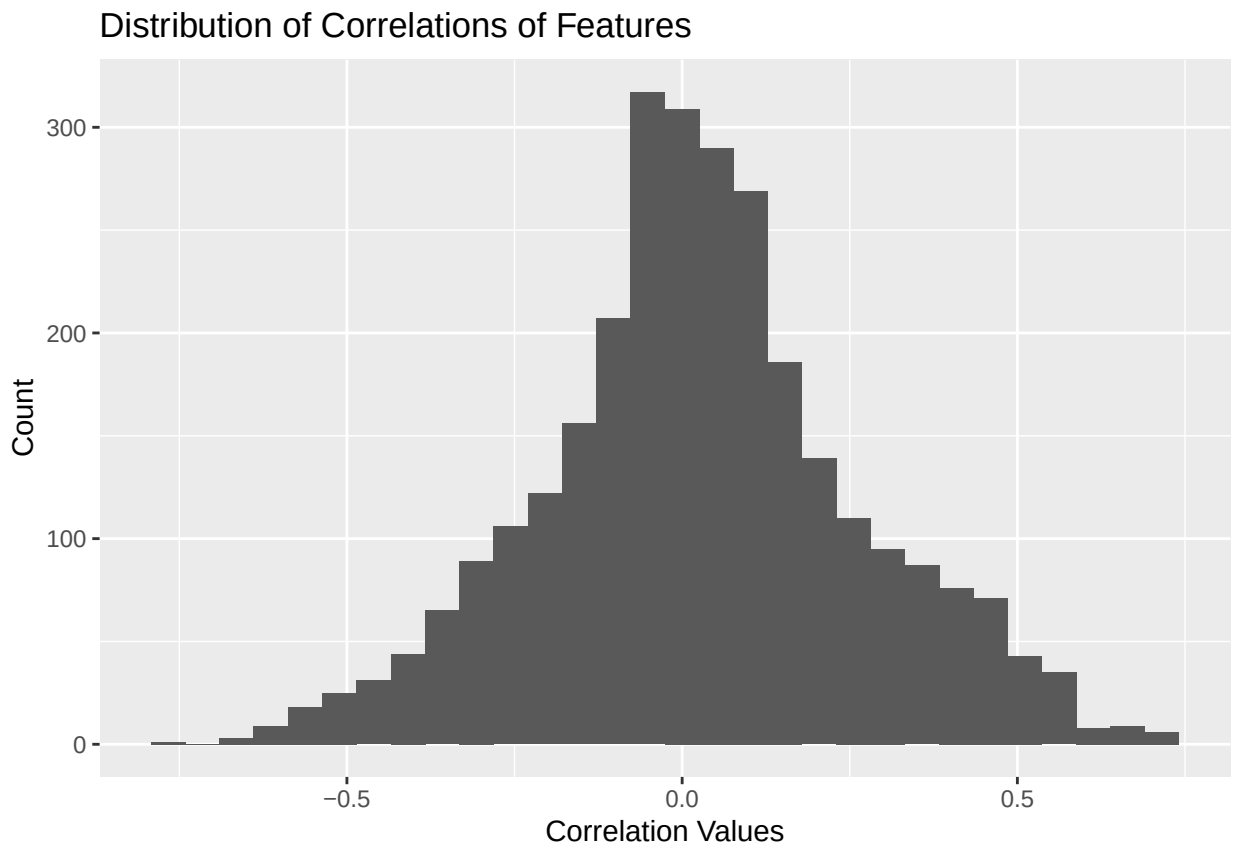
Below is a distribution plot showing the degree of correlation across features.



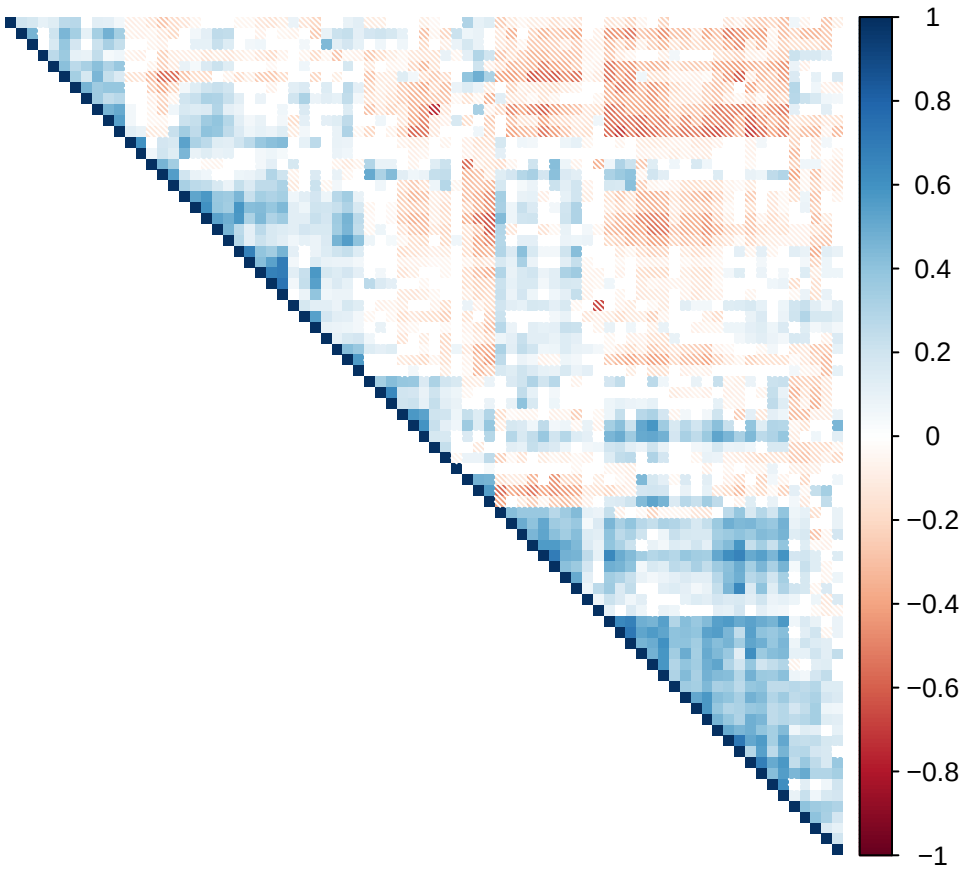
Considering this, a cutoff for being considered highly correlated will be set at an absolute value of .8.

A data set is created that has the highly correlated features removed.

After removing the highly correlated features the distribution of correlated features looks more normal.



Additionally the heat-map of correlations looks more symmetrical.



The imputed, normalized, and un-correlated data is now ready for the next step.

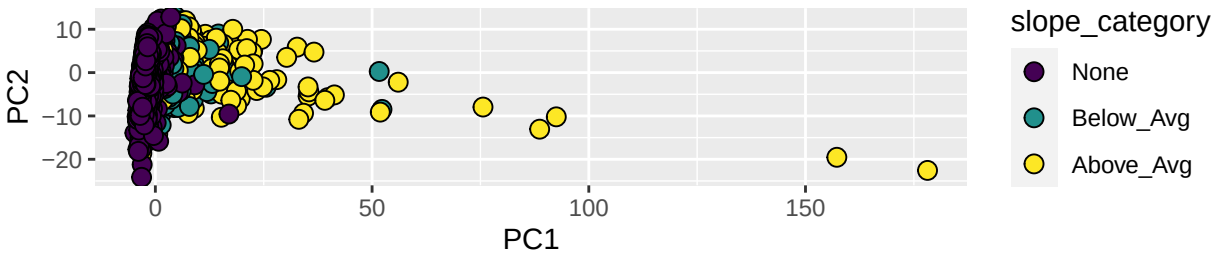
2.6 Clustering

Clustering techniques seek to visualize how categories of data might group together in a dimensionally reduced space. This enables us to see if our categories would naturally group together in an unsupervised way based on their features.

2.6.1 Principle Components Analysis - PCA

PCA seeks to reduce dimensionality of data by ensuring that the new features - called principle components (PCs) - are orthogonal to each other and thus should be independent and uncorrelated. PCA is a linear algorithm.

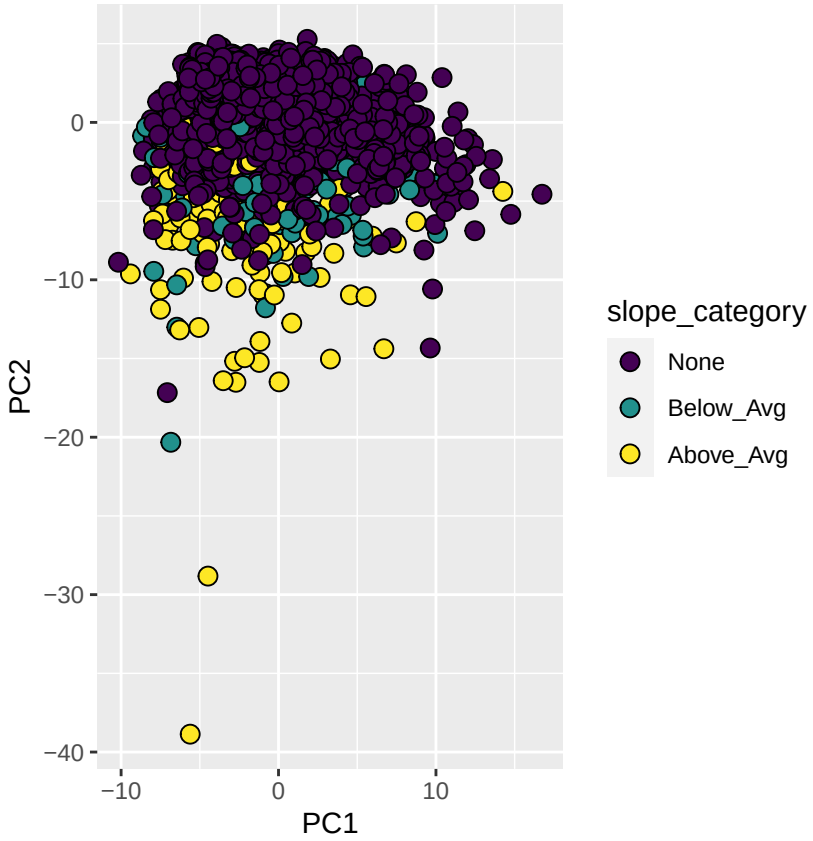
The plot of the first two principle components (PC) of the data without highly correlated variables removed, is below.



	PC1	PC2
Standard deviation	7.118901	5.031861
Proportion of Variance	0.339390	0.169560
Cumulative Proportion	0.339390	0.508950

It appears that the first two PCs do not adequately separate the groups. Although it appears the high growth counties have little overlap with the counties that experienced little to no growth. We can see that the first two principle components only explain about 50 percent of the total variance in the data.

If the highly correlated variables are removed, this may improve the visualization of the data. Below is the PC analysis plot of that data.



	PC1	PC2
Standard deviation	3.788798	3.079165
Proportion of Variance	0.191480	0.126470
Cumulative Proportion	0.191480	0.317940

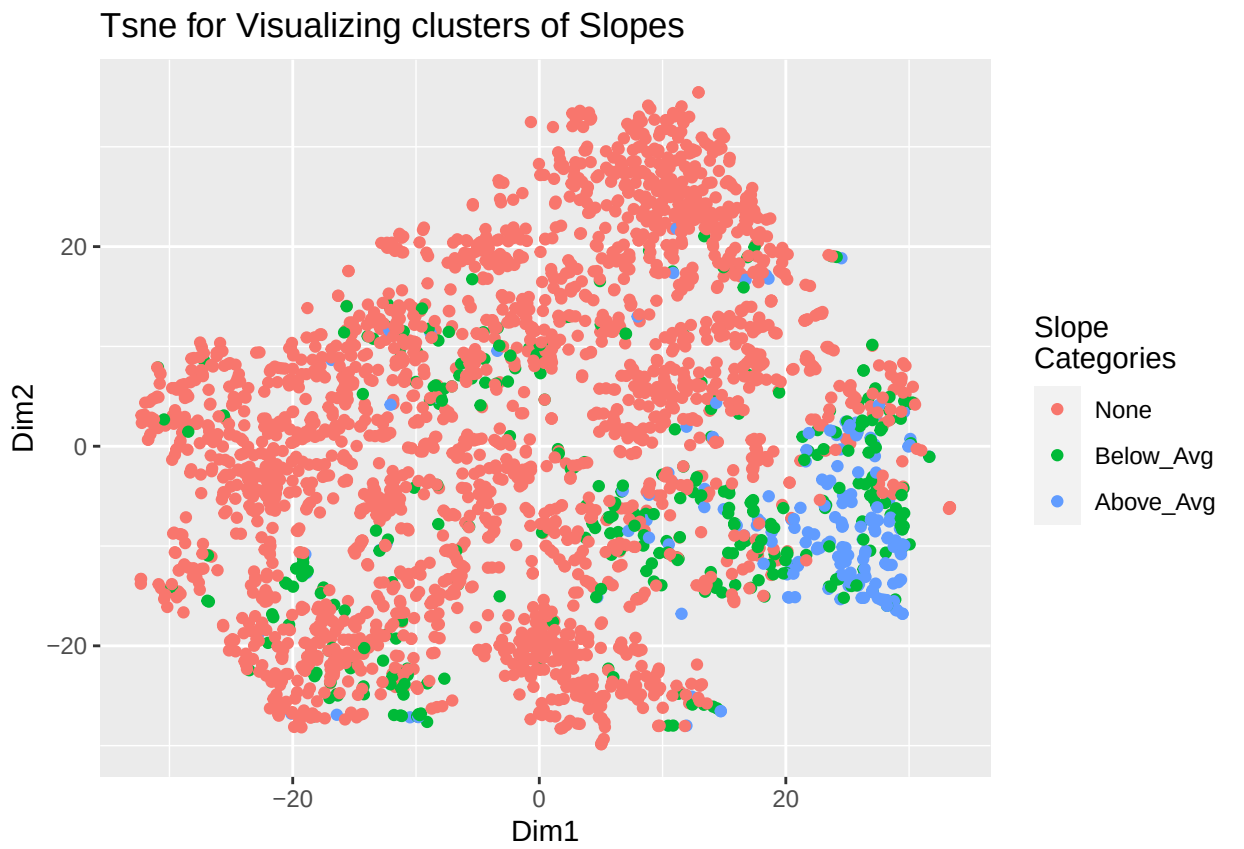
Here we see more overlap with all the growth categories, this suggests the data without the correlated features will have worse ability to resolve differences in groups. The first two PCs of this data set explain about 30 percent. Further demonstrating that removing correlated variables might reduce the ability of the data to resolve the growth categories.

2.6.2 t-SNE

T-distributed Stochastic Neighbor Embedding (t-SNE) is a non-linear dimensionality reduction method. t-SNE, like kNN, uses Euclidean distance but instead converts that distance to a conditional probability. This probability is the likelihood that a given value would select another value to be close to it. The formula for this conditional probability is below:

$$p_{j|i} = \frac{exp(-||x_i - x_j||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x_j||^2/2\sigma_i^2)}$$

Below is the t-SNE plot for the data that includes the correlated features, as this was the data set that clustered better with PCA.



This visualization shows us that non-linear distance based methods, like kNN, may successfully identify the differences between the negligible growth counties and the above average growth counties.

2.7 Modelling

This section will focus on using machine learning algorithms to identify socio-economic and health predictors that are important for classifying the the degree of growth experienced in the early days of COVID-19.

2.7.1 Data Partitioning

The first step in training the models, will be partitioning data into training and validation sets.

```
#splitting data into validation and training for applying machine learning algos
set.seed(1993)
edxIndex <- createDataPartition(y,p = .8,list = F,times = 1)
cor <- x1 %>% add_column(fct_slope = y)
nocor <- x1_cor_out %>% add_column(fct_slope = y)
#getting training and validation sets for the data with correlated features
edx_cor <- cor[edxIndex,]
validation_cor <-cor[-edxIndex,]
#subsetting data that has highly correlated features removed
edx_no_cor <- nocor[edxIndex,]
validation_no_cor <- nocor[-edxIndex,]
```

2.7.2 Naive kNN

Before parameter tuning, an unmodified kNN will be performed on the data with correlates and without to determine which data set produces the most accurate predictions. Prediction accuracy will be evaluated

using repeated 10-fold cross validation. Cross-validation here means the training data will be split into 10 subsets, one subset will be reserved as the validation subset, and this is repeated until each subset has been used as the validation subset.

Method	Data	Accuracy
Untuned kNN	Complete Training	0.8631171
Untuned kNN	Training w/o Corr Features	0.8537585

This data suggests that the data that includes the correlated features produces more accurate predictions. We will tune the kNN algorithm on this data set as a result.

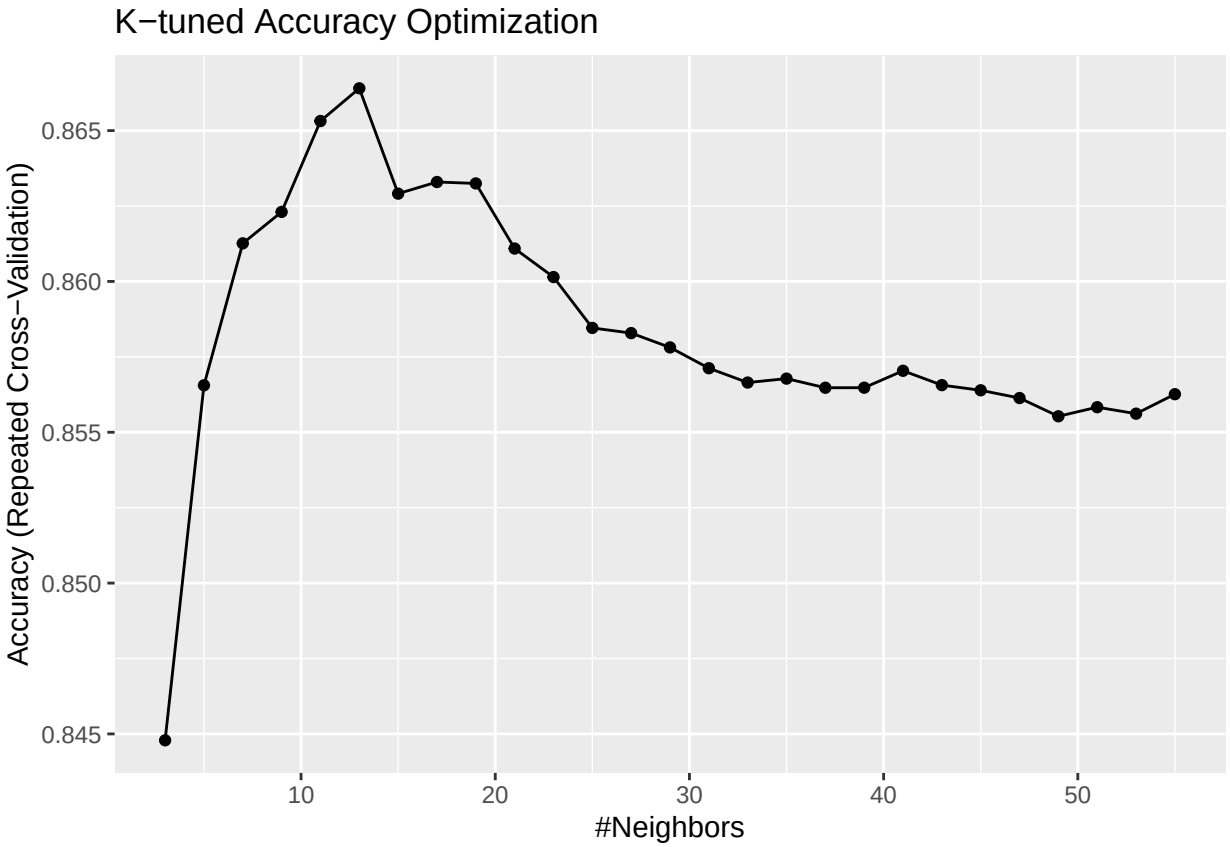
2.7.3 KNN Optimization

Optimization will begin with determining if any transformations improve the accuracy of predictions. Because the imputation process normalizes the data it is already transformed such that the the average of each feature is 0 and the standard deviation is 1. Two more common transformations are to scale the data such that it falls between 0 and 1, and a Yeo-Johnson transformation. Yeo-Johnson is a power transformation that further stabilizes variance, and makes the data more normal.

Method	Data	Accuracy
Untuned kNN	Complete Training	0.8631171
Untuned kNN	Training w/o Corr Features	0.8537585
Scaled kNN	Complete Training	0.8586363
Yeo-Johnson kNN	Complete Training	0.8630384

This shows that for these transformations, neither the Yeo-Johnson or the scaling improves the accuracy of the predictions.

As a result just the normalized data will be used moving forward for the rest of model training. Using this transformed data, the k-nearest neighbor will be tuned, to maximize accuracy.

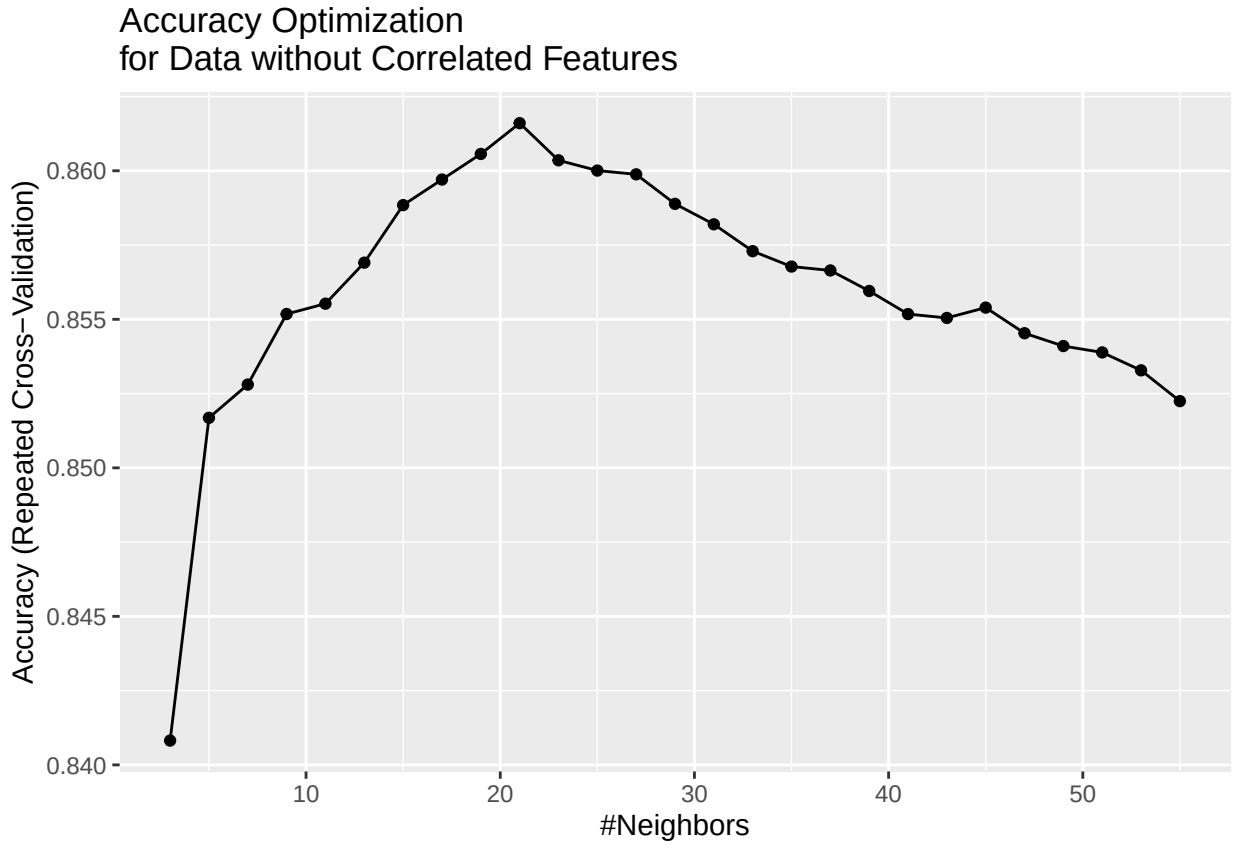


Method	Data	Accuracy
Untuned kNN	Complete Training	0.8631171
Untuned kNN	Training w/o Corr Features	0.8537585
Scaled kNN	Complete Training	0.8586363
Yeo-Johnson kNN	Complete Training	0.8630384
Tuned kNN	Complete Training	0.8664013

The K-tuned accuracy plot makes it evident that the accuracy peaks at 11 and then declines. The default k-values are 5,7, and 9 so it is not surprising that tuning only minorly improves the accuracy.

This will be the the model that we will use for the prediction on the validation set. Before determining how the trained model performs on the validation set, the features that were important for the accuracy of the prediction will be analyzed.

2.7.4 Variable Importance - kNN



The k-tuning on the data without highly correlated features, seems to converge instead of peak. The performance of the kNN algorithm on the data without highly correlated features performs worse than that the complete training data.

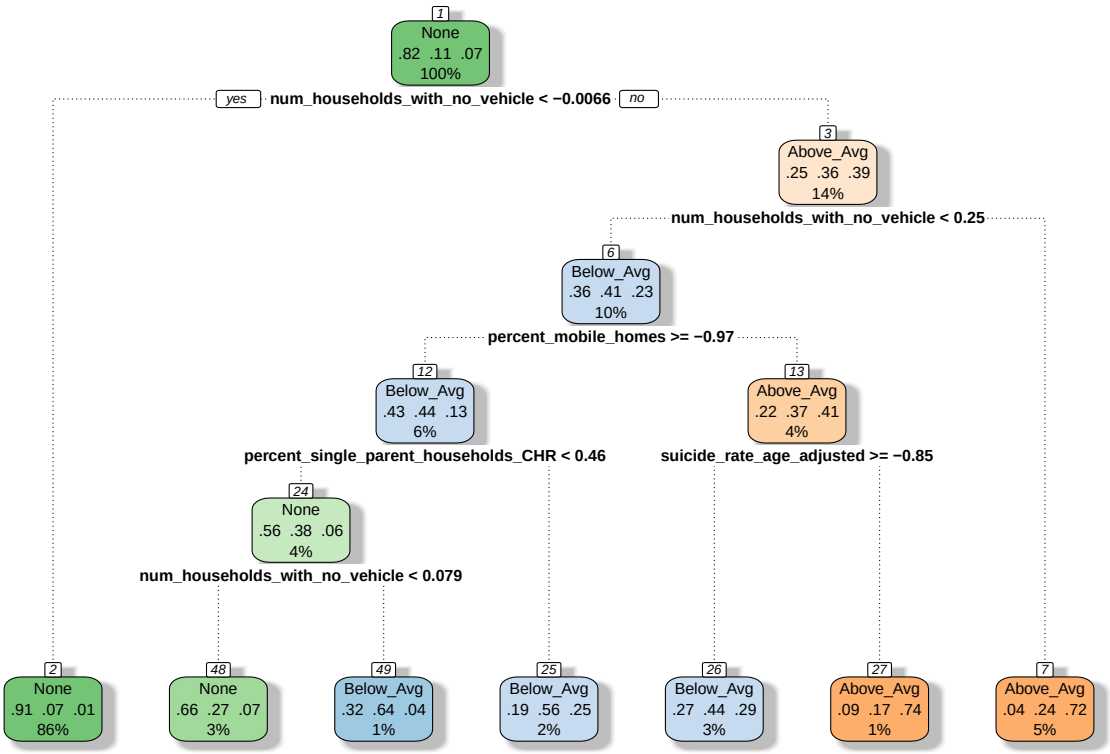
Important_Features
num_households_with_no_vehicle
num_not_proficient_in_english
percent_rural
population_density_per_sqmi
num_native_hawaiian_other_pacific_islander
num_firearm_fatalities
average_traffic_volume_per_meter_of_major_roadways
percent_asian
num_american_indian_alaska_native
percent_multi_unit_housing

The top 10 important features, roughly describe urban areas. From our previous COVID-19 growth plots displayed on maps, we know that this is often where the growth was the highest. It is interesting to see these features identified by a machine learning algorithm, even if some of them may have been intuitive.

We will confirm these features with a second machine learning model to determine that there is consensus on what factors are important between algorithms.

2.7.5 Variable Importance - CART

Classification and Regression Trees (CART), is a machine learning algorithm where a series of yes or no questions are asked and the answer generates a branch, which eventually leads to a classification. The decision trees these models produce are easy to interpret, and visually show which factors are important.



Method	Data	Accuracy
Untuned kNN	Complete Training	0.8631171
Untuned kNN	Training w/o Corr Features	0.8537585
Scaled kNN	Complete Training	0.8586363
Yeo-Johnson kNN	Complete Training	0.8630384
Tuned kNN	Complete Training	0.8664013
Tuned kNN	Training w/o Corr Features	0.8616032
Tuned CART	Training w/o Corr Features	0.8518171

The CART model does not perform as as well as the kNN model, but the decision tree makes it easy to recognize the important features.

Method	Important_Features
kNN	num_households_with_no_vehicle
kNN	num_not_proficient_in_english
kNN	percent_rural
kNN	population_density_per_sqmi
kNN	num_native_hawaiian_other_pacific_islander
kNN	num_firearm_fatalities
kNN	average_traffic_volume_per_meter_of_major_roadways
kNN	percent_asian
kNN	num_american_indian_alaska_native
kNN	percent_multi_unit_housing
CART	num_households_with_no_vehicle
CART	population_density_per_sqmi
CART	num_not_proficient_in_english
CART	num_firearm_fatalities
CART	percent_rural
CART	percent_mobile_homes
CART	percent_single_parent_households_CHR
CART	food_environment_index
CART	average_number_of_mentally_unhealthy_days
CART	percent_age_17_and_younger

The important variables in both models are largely overlapping, and the interpretation is still that cities tend to see higher growth than more rural areas.

2.7.6 Model Validation

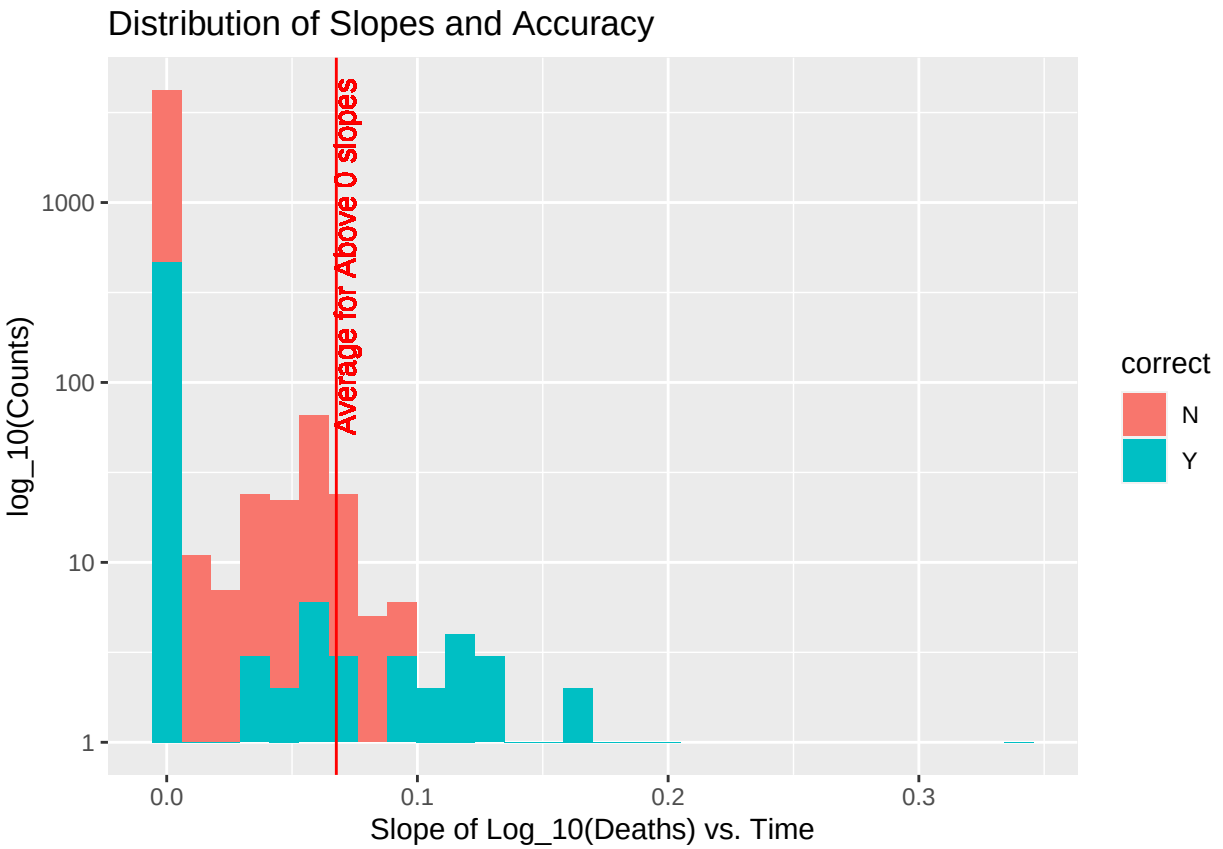
The normalized, tuned, kNN model with correlates will be used to predict the growth category for each county in the validation data set, considering this had the highest accuracy.

3 Results

	F1	Balanced Accuracy
Class: None	0.9396378	0.7378083
Class: Below_Avg	0.2765957	0.5838364
Class: Above_Avg	0.6764706	0.7961501

This data shows that the above average growth counties and the negligible growth counties were classified with reasonable accuracy, although the below average counties were harder to predict. This was what the clustering data would have made us assume.

How the correct predictions line up with different slopes, will inform if the border cases were the hardest to define.



Generally speaking the slopes at zero and far away from zero were easier to predict, with the close to zero slope counties being the most difficult.

Below is a summary of the accuracies:

Method	Data	Accuracy
Untuned kNN	Complete Training	0.8631171
Untuned kNN	Training w/o Corr Features	0.8537585
Scaled kNN	Complete Training	0.8586363
Yeo-Johnson kNN	Complete Training	0.8630384
Tuned kNN	Complete Training	0.8664013
Tuned kNN	Training w/o Corr Features	0.8616032
Tuned CART	Training w/o Corr Features	0.8518171
Tuned kNN	Complete Validation	0.8702422

The normalized data, with tuned parameters, in the kNN model successfully predicted the growth category for each county 87.0242215 percent of the time.

4 Conclusions

4.1 Summary and Future Directions

The COVID-19 pandemic has led to a tremendous loss of life, and totally transformed society in a matter of a few months. However this catastrophe has not had an equal effect in all regions of the world, and within

the US. Leaving many to wonder, what determines whether the virus will creep along in a given area or if it will see explosive growth.This project focused on utilizing county-level COVID-19, socio-economic and health data to determine what factors were important in the early days of the pandemic. The rate of growth in COVID-19 related fatalities in a given county was selected as the outcome to be predicted as opposed to cases because case numbers are largely a function of amount and type of testing being conducted, which varies widely across counties. However, the fatality data is more standardized, although surely contains some reporting error still. The growth of COVID-19 related fatalities in the first 14 days after the 10th death in the county was examined, because it was largely before any interventions were in place. Examining the growth rates beyond that point would have been confounded by how early stay-at-home orders were put in place and how well they were adhered to. The growth rate in this early time window is a better reflection of that counties natural vulnerability.

After preliminary data visualization, cleaning, and feature engineering machine learning algorithms were applied to determine if the degree of growth could be predicted accurately based off socio-economic and health data, and what factors were most important in producing an accurate prediction. After training and tuning the model it was able to accurately predict what degree of growth, in COVID-19 related deaths, a county would experience 87.0242215 percent of the time.

The factors that were identified as important for accurate predictions were largely reflective of the type of socio-economic factors you might see in a city. Some of these included: population density per square mile, number of households with no vehicle, percent rural (with a low percentage meaning an urban environment like a city), percent multi-unit housing, and average traffic volume per meter of highway. Perhaps the least intuitive factor that the algorithm picked up on is how Native Americans are being disproportionately affected by COVID-19. While this has not received the same media attention as areas like New York and New Jersey, this Harvard Gazette article states that the Navajo nation has the third highest per capita rate of COVID-19 in the country, as of April 30th. You can see this reflected in the map plots as the above average growth areas in seen in New Mexico and Arizona. While some of these factors may have been intuitive, hopefully this serves as a data driven approach into what made certain areas more vulnerable.

Some alternative approaches to this question, which may have yielded different results, include: using more robust machine learning like boosted random forrest, predicting the growth as continuous instead of categorical outcome, and considering more factors for each county. Google has made mobility reports available to see how COVID-19 has changed the way people move around. Joining this data with the data set analyzed for this project would enable assessing how effective stay-at-home orders were in different areas, how tightly they were complied with, and if socio-economic or health factors were predictive for their success or adherence. Lastly, this dataset is considerably unbalanced, with low/no growth counties greatly outnumbering counties with appreciable growth. This makes this dataset a great candidate for resampling. Tools like SMOTE and Tomek links could be used to undersample the low/no growth cases and upsample the growth cases. This would further improve the sensitivity and recall of the model.

5 Appendix

```
##                _
## platform       x86_64-w64-mingw32
## arch           x86_64
## os             mingw32
## system         x86_64, mingw32
## status
## major          4
## minor          0.0
## year           2020
## month          04
## day            24
## svn rev        78286
## language       R
```

```
## version.string R version 4.0.0 (2020-04-24)
## nickname       Arbor Day
```