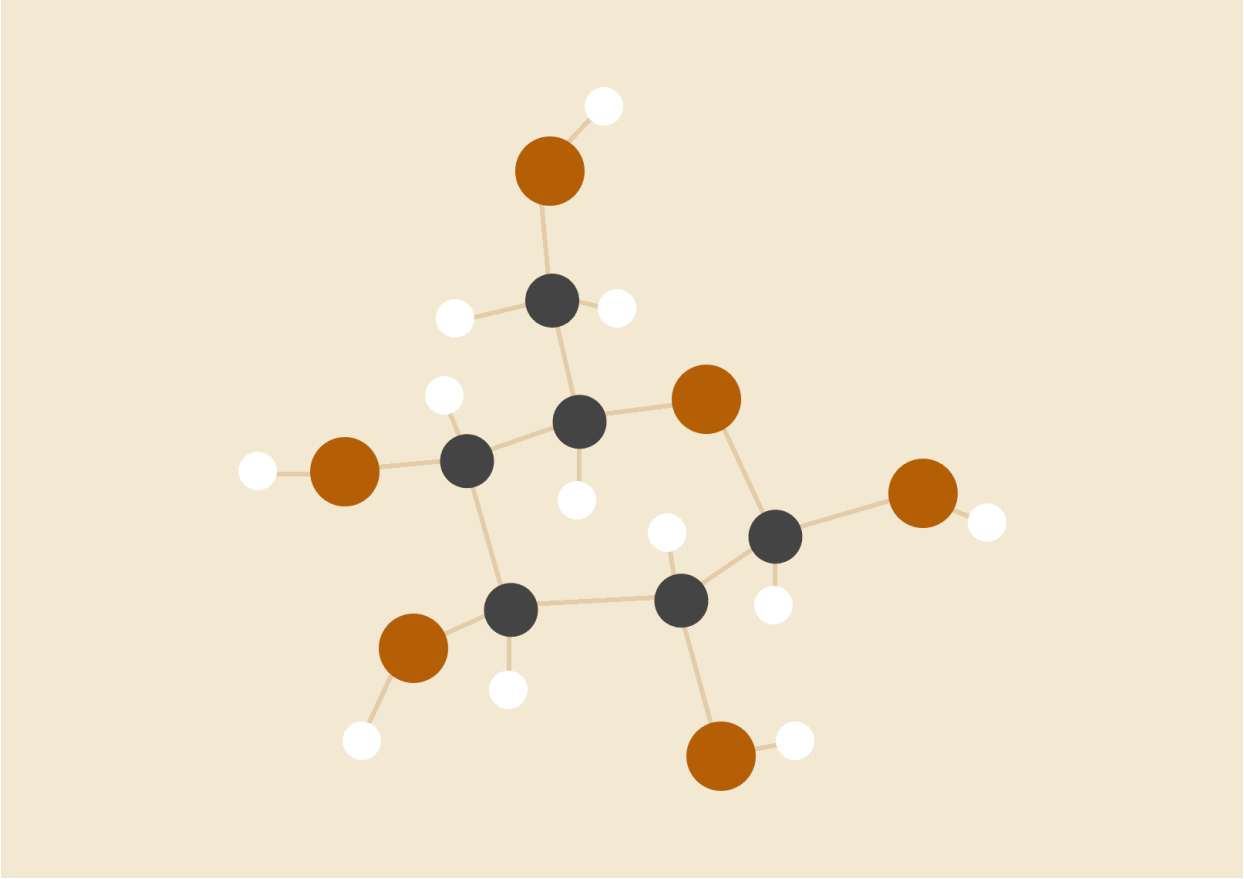
Classifying COVID-19 Data

Looking at classifying counties by their COVID-19 susceptibility



Ryan Capó, James Zhai, Will Clark

May 3, 2021

Data Mining

DATA PREPARATION 2
Features Description 3
Features Statistical Summary 5
Correlation Matrices 6
MODELING 9
Section 1.1: Identify training and testing data; find class. 9
Section 1.2: Classification model with rpart (CART) method 11
Section 1.3: Classification model with k-nearest neighbors method 15
Section 1.4: Classification model with rpart(CART) method with cost 17
Section 1.5: Classification model with Conditional Inference Tree 19
Section 1.6: Comparison between models 22
Section 2: Classification with spread rate and vaccination rate 23
EVALUATION 27
DEPLOYMENT 28
REFERENCES 29

DATA PREPARATION

Our code begins by reading in the csv file "COVID-19_cases_plus_census". It is read into CensusData and then we proceed to scale the data. The code then creates a dataframe called *TotalData* that takes various variables from the csv file as input. These variables include the following: county_name, state, confirmed_cases, deaths, cases_percent, population_density, male_pop_density, public_trans, old_percent, unemployed percent, Gini_index, Nohigh_degree, median_age, median_income, income_on_rent, and LongCommTime.

It is important to note that our code generates another data frame called *TotalData2*. This data frame utilizes variables from *TotalData* as well as the following: spread_rate, pvac_rate, tvac_rate, and people_vaccinated_per_hundred. It is also important to note that *TotalData2* utilizes states whereas *TotalData* utilizes solely counties.

In regards to the cleanliness of the data, duplicates were not necessarily an issue since a lot of the data is numerical, meaning duplicates does not affect the outcome of our prediction. Outliers were also not an issue. For the most part, there were no missing values. However, we did opt to remove some values. We decided we only wanted to use counties with confirmed cases, meaning if a county had 0 confirmed cases we removed it from the model. There are 3143 counties in the United States and only 4 of those had 0 confirmed cases. This takes the amount of counties in our model from 3143 to 3139.

The only area we had an issue with missing values was in the csv file titled "covid_confirmed_usfacts". This dataset provided us with information about confirmed cases between January and April 2021. In some cases, some of the dates had missing values between days. For example, let's say in Texas on January 1, 2021 there were 5000 confirmed cases and then January 3, 2021 had 10,000 confirmed cases. In the event that there is no data for January 2, 2021, we just took an average of the day before and day after and then inputted that number into the missing value. In this case, we would input 7,500 cases for January 2, 2021 to aid in our predictive model.

We created a class to determine the effects of *population_density* and *cases_percent* to ultimately determine two of the following things:

1. How dangerous a county is to being exposed to COVID-19 (*TotalData*)
2. How dangerous a state is to being exposed to COVID-19 (*TotalData2*)

We chose to define the class this way because we are interested in which counties and states in our data set are “COVID-19 hot spots” (Dangerous vs Not Dangerous). We use this class to ultimately predict a fourth wave of the COVID-19 pandemic throughout various counties and states in the United States. We also elected to choose arbitrary values of *population_density* and *cases_percent*. Our equation is (cases_percent > 200 & population_density <= 200) | (cases_percent > 8 & population density <= 200). If the value falls within these bounds, the area is determined dangerous. If the value falls outside these bounds, it is determined not dangerous.

We also created a class to determine the effects of *spread_rate* and *deaths_percent* to also determine the same above two criteria. Our equation to generate whether an area is dangerous or not dangerous is as follows: (spread_rate > 2000 | deaths_percent > 0.15). As previously mentioned, if the value falls within these bounds then the area is determined to be dangerous. If it falls outside, then the area is not dangerous. The values of the equation were chosen arbitrarily as they deemed sufficient criteria for our outcomes.

Features Description

County_name: This variable refers to the name of each county in the data. This variable was clean and had no missing values (county names).

State: This variable refers to the name of each state. This variable was clean and had no missing values (state names).

Confirmed_cases: This variable refers to the amount of confirmed cases per population. The data had no missing values.

Deaths: This variable refers to the amount of deaths per population. The data had no missing values.

Cases_percent: This variable refers to the percentage of confirmed cases with respect to the total population. It is calculated by taking confirmed_cases and dividing it by the total population * 100 to find the cases_percent.

Pop_density: This variable refers to population density, which is a measurement of the amount of people living in a unit of area. It is calculated by taking the total population and dividing it by the area. There were no missing values in the dataset.

Male_pop_density: This variable again refers to population density. However, it is just exclusive to the male population and it is also a percentage. It is calculated by dividing the male population by the total population * 100 to make it a percentage. There were no missing values in the dataset.

Public_trans: This variable refers to the amount of people who utilize public transportation per county.

It is calculated by pulling the commuters_by_public_transportation column data from the csv file. There were no missing values.

Old_percent: This variable refers to the amount of old people per county and it is also a percentage. It is calculated by taking data from male_60_61 all the way to male_85_and_over as well as female_60_61 all the way to female_85_and_over and then * 100 to make it a percentage. There were no missing values in the data.

Unemployed_percent: This variable refers to the unemployment rate. It is calculated by adding together the unemployed population and the population not in the labor force and subsequently divided by the summation of the unemployed population, population not in the labor force, and the employed population. The resulting percentage is the unemployed_percent variable. There were no missing values for the data.

Gini_index: This variable refers to the Gini index, or sometimes referred to as the Gini coefficient. It is a summary measure of income inequality, which summarizes the dispersion of income across the entire income distribution. The index ranges from 0 to 1. 0 indicates perfect equality, meaning everyone receives an equal share. 1 indicates perfect inequality, meaning only one recipient or group of recipients receives all the income. There were no missing values in the dataset [2].

Nohigh_degree: This variable refers to the population of people in a county that do not have a "high degree". The criteria for a high degree is people who have a masters degree, a graduate degree, a bachelor's degree, or an associates degree. It is calculated by subtracting 1 from the summation of masters_degree, graduate_professional_degree, bachelors_degree, and associates_degree. This number is then divided by the total population to output the number of "high degrees".

Median_age: This variable refers to the median age of a population in a county. There were no missing values in the data.

Median_income: This variable refers to the median income of a population in a county. There were no missing values in the data.

Income_on_rent: This variable refers to the percentage of income that is spent on rent for a population in a county.

LongCommTime: This variable refers to the percentage of people who have a "long commute time" to work. The formula for this variable is as follows:

100 - (commute_less_10 mins + commute_10_14_mins + commute_15_19_mins + commute_20_24_mins + commute_25_29_mins + commute_30_34_mins + commute_35_44_mins) / (commute_less_10_mins + commute_10_14_mins + commute_15_19_mins + commute_25_29_mins +

commute_30_34_mins + commute_35_44_mins + commute_45_59_mins + commute_60_more_mins) *100 to find the percentage. There were no missing values in the data set.

Spread_rate: This variable refers to the rate at which COVID-19 spreads. Essentially, it refers to daily new confirmed cases between January and April 2021.

Pvac_rate: This variable refers to the daily percentage of people who received a COVID-19 vaccination between January and April 2021.

Tvac_rate: This variable refers to the daily number of new vaccines supplied to the state between January and April 2021.

People_vaccinated_per_hundred: This variable refers to the percentage of people who received a COVID-19 vaccination before April 28, 2021.

Features Statistical Summary

Variable	Scale	Range	Median	Mean	Variance	Std.	Max	Min

						Dev.		
Cases_%	Nominal	3.2	7.5	7	8.1	4.81	3.2	0
Pop_density	Ratio	48,643	42	224	1.64e6	1,283	48,643	0
Public_trans	Nominal	7.35e5	33	2423	5.84e8	2.4e4	7.35e5	0
Old_%	Nominal	5.8	24	24	3.1	5	64	5
Unemployed_%	Nominal	7.1e-1	4.4e-1	4.5e-1	7.01e-3	8.37e-2	8.9e-1	1.17e-1
Gini_index	Ratio	2.7e-1	4.4e-1	4.4e-1	1.2e-3	3.44e-2	5.9e-1	3.2e-1
Median_age	Ordinal	4.4e1	4.1e1	4.1e1	2.8e1	5.37	66	21
Median_income	Ratio	1.1e5	4.8e4	4.9e4	1.7e8	1.3e4	1.2e5	1.9e4
Income_on_rent	Ratio	40	28	27	19	44	50	10
LongCommTime	Nominal	51	86	85	61	7.8	99	48
Confirmed_cases	Nominal	1.0e6	1.9e3	1.2e2	7.9e8	2.8e4	1.0e6	1
Male_pop_density	Ratio	38	49	50	58	2.4	80	41
Spread_rate	Nominal	8.2e3	1.02e3	1.68e3	4.26e6	2.06e3	8.3e3	74
Pvac_rate	Ratio	.27	.37	.37	0	.06	.55	.27
Tvac_rate	Nominal	2.66e5	2.67e4	4.08e4	2.29e9	4.7e4	2.69e5	3.1e3
People_vac_100	Ratio	29.08	42.16	42.75	46.4	6.81	59.82	30.74

Table 1: Statistical Summary for all Features

The table highlights some of the most important statistics utilized in our models. The correlation matrices on the next page give an in depth look at these features as they determine the dependability of our variables with respect to our models.

Correlation Matrices

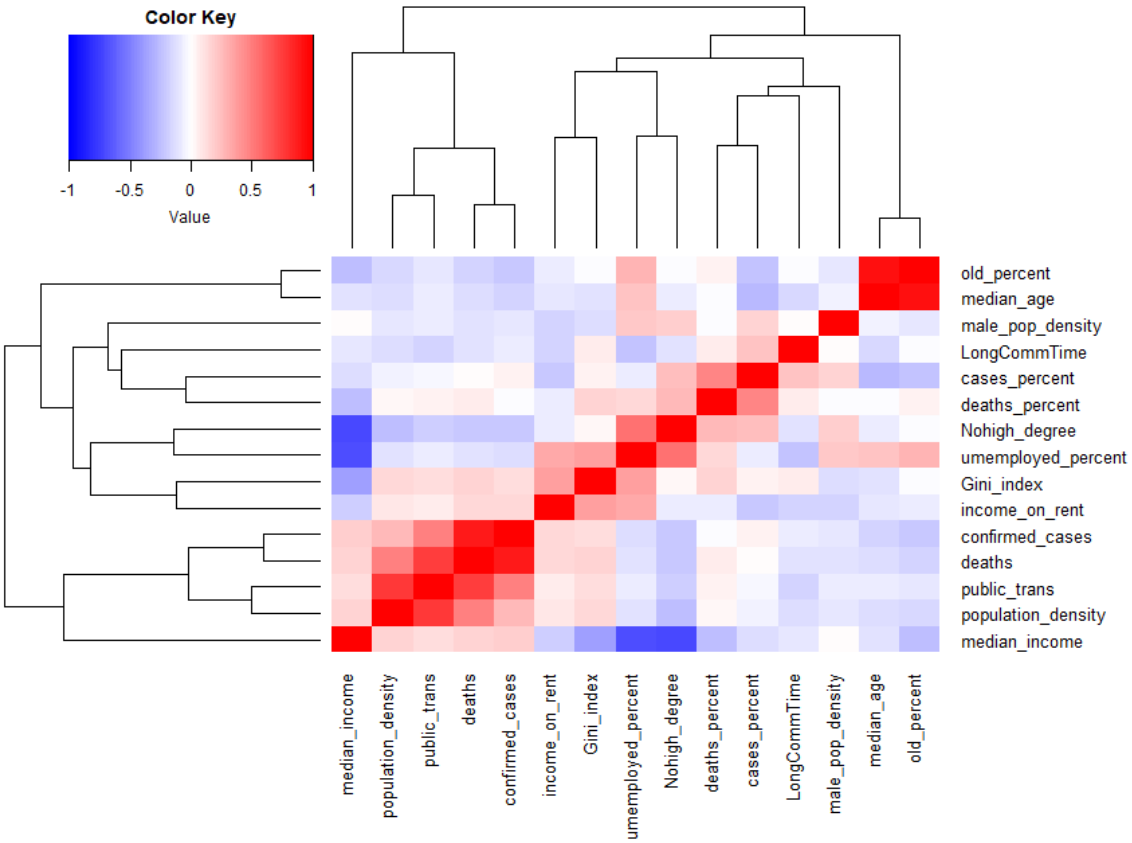


Figure 1: Correlation Matrix for TotalData

The first correlation matrix portrays features from the data frame *TotalData*. It portrays how dependent variables are on each other. A red cell indicates a strong dependence whereas a blue cell indicates a weak dependence. One can observe a linear path of red and this is for the variables that depend on itself. For example in the top right, *old percent* is all red indicating that old percent is in fact dependent itself, which seems obvious. However, we are really more interested in the variables outside of this linear path. This matrix allows us to evaluate which variables are strongly dependent on each other. For example, we can see a deep red color in the cell for *population density* and *public_trans* indicating a strong correlation between the two. This becomes an area of interest for our objective and something we can explore deeper in our models.

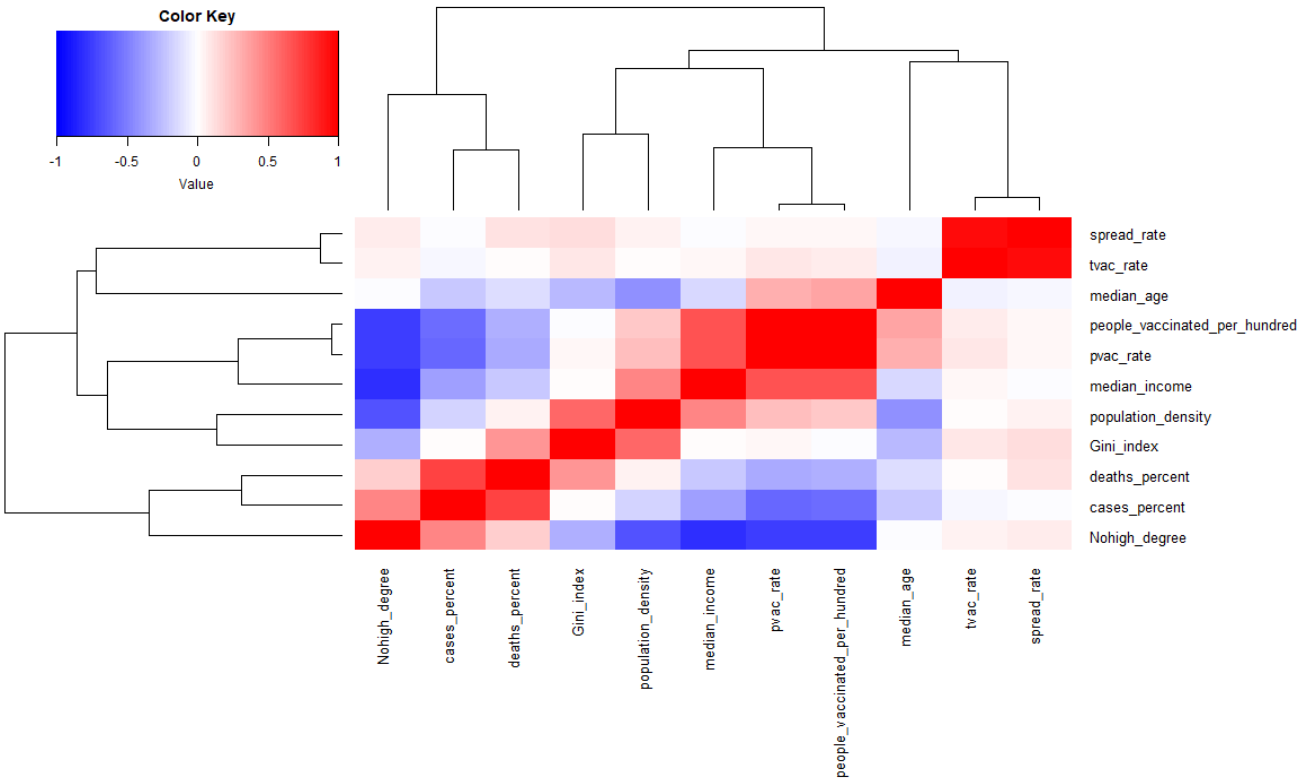


Figure 2: Correlation Matrix for TotalData2

In this matrix we utilize the data frame *TotalData2*. *TotalData2* contains some of the same features of *TotalData* as well as data for spread rate and vaccination rates. Like the previous correlation matrix, this aids in determining strong and weak correlations between variables. An example of a weak correlation between variables would be *Nohigh_degree* and *pvac_rate*. In this case, this would be a correlation that we would not pursue. We aim to pursue strong correlations between variables as it will produce the best results for our prediction.

MODELING

Section 1.1: Identify training and testing data; find class.

After creating the dataset with all the elements above, the project will split the TotalData into two parts, which is the training data and the testing data. (does not use hyper parameters in this project). For the whole data set which represents 3139 counties in 51 states, The project chooses counties in Maryland, California, Mississippi, Delaware, Texas, New York, Iowa, Kansas, Nevada, and North Dakota as the training dataset. The project chose specifically those ten states because it has states with metropolitan counties like New York and California, as well as states with rural counties like Kansas and Mississippi. Besides, the project wants to use at least 20 percent of the total counties to make the training data big enough to analyze. For the training dataset and the testing dataset, it has a relatively balanced number of dangerous/non-dangerous counties.

	Dangerous counties (TRUE)	Non-Dangerous counties (FALSE)
Training Data	305	452
Testing Data	1201	1181

Table 2: counties in training and testing data

Then, the project will visualize the training data by building up the map and put all the train counties into the map. It uses the USA-Counties map with the map_data function, and it links with the TotalData with the county name. In the map, the red counties are counties that are marked as dangerous counties. On the other hand, the blue counties are “non-dangerous” counties that do not fulfill the TRUE class requirement.

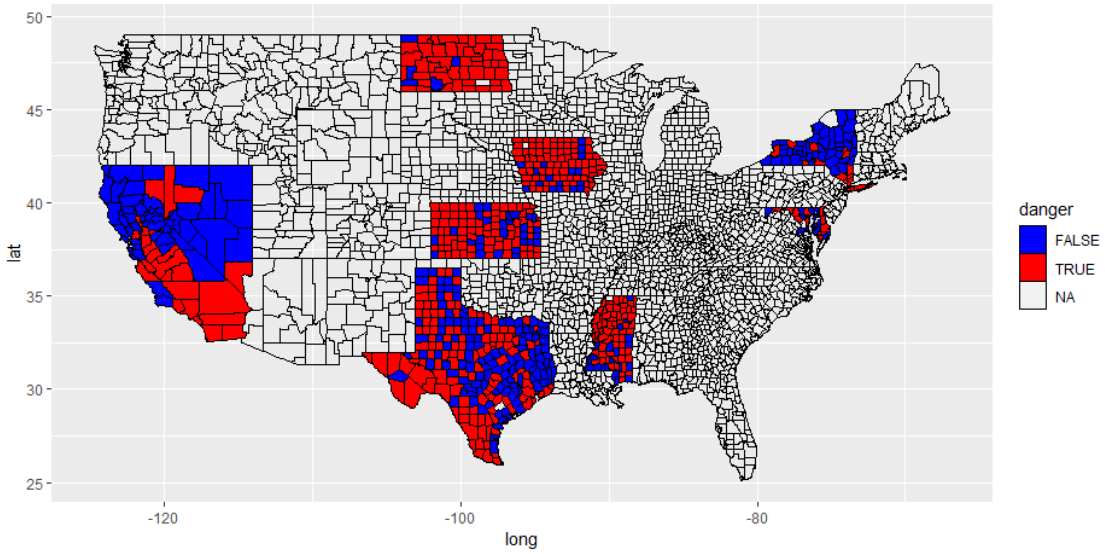


Figure 3: Training counties on the map

In the training map, it shows clearly that the dangerous counties have two types. For counties like Dallas and Harris, they are marked as red because they have a huge population density and significant number of case percent. On the other hand, many counties in North Dakota and Mississippi have also been marked in red even though their population density is small; that is because those counties have a relatively higher case percent, so they are also “counties in danger.”

To identify how the training data works in the classification, the project will show a list of attributes’ importance to the classification. Here we should remove the attributes that have either no-relationship with the class (e.g. county name) or have direct connection with the class (e.g. case percent and death percent).

	Before remove		After remove	

1	County name	0.9101669	Unemployment	0.2872829
2	Case percent	0.8790394	Median Age	0.2621499
3	Death percent	0.3414783	Long commute	0.2529029
4	Unemployment	0.2872829	Old percent	0.2401546
5	Confirmed case	0.2669150	Pop density	0.1572103

Table 3: List of importance for TotalData to the class

The table above shows that the unemployment rate and the age data may have some connection with the dangerous situation of that county, and it may be because a higher unemployment rate will force people to go outside to find jobs; for the age data, old people may be more likely to get infected.

Section 1.2: Classification model with rpart (CART) method

The project will first build the classification model with the rpart method. It will set the minsplit to 2 and tunelength to 10 so that it will form a more practical decision tree. Then, the project will build the decision tree for the training data. Besides, the project will also create a plot for the model which shows the connection between layer and accuracy.

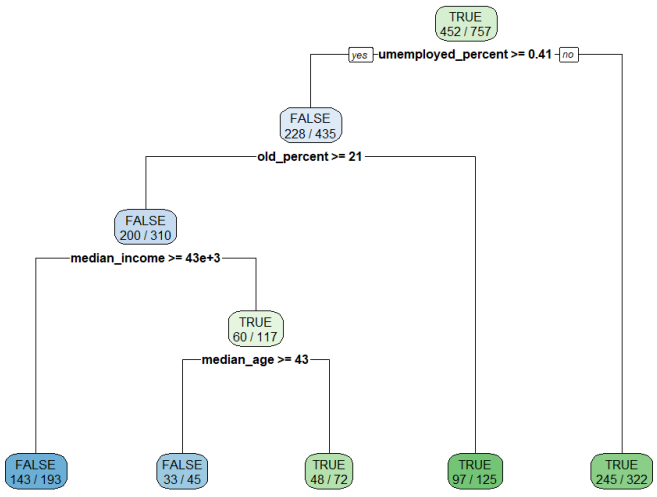


Figure 4: Decision tree for rpart method classification model

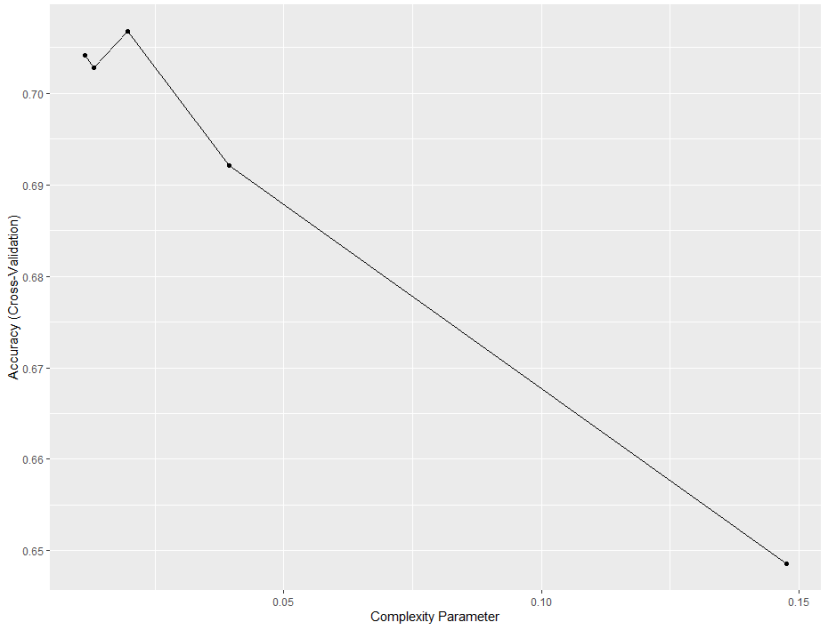


Figure 5: rpart-Complexity Parameter and Accuracy

The rpart classification model has dependencies with the training data attributes. The report shows a part of the list in the table below.

Attribute	Importance
Median age	100.00
Old percent	85.70
Unemployed percent	49.74
Long commute time	40.88

Table 4: rpart-model importance

After building up the model, the project will do the prediction; it uses the model to determine whether the testing counties are dangerous(TRUE) or not dangerous(FALSE), and it will store the result into the data frame with the testing data. Since the testing data also has a list of counties' names, the project then can put the prediction result into the map of the United States. Here there will be two maps, the first one shows the ideal result, which is determined by the class variable as the training data; the second one represents the prediction result, using the classification model and the attributes in the testing data.

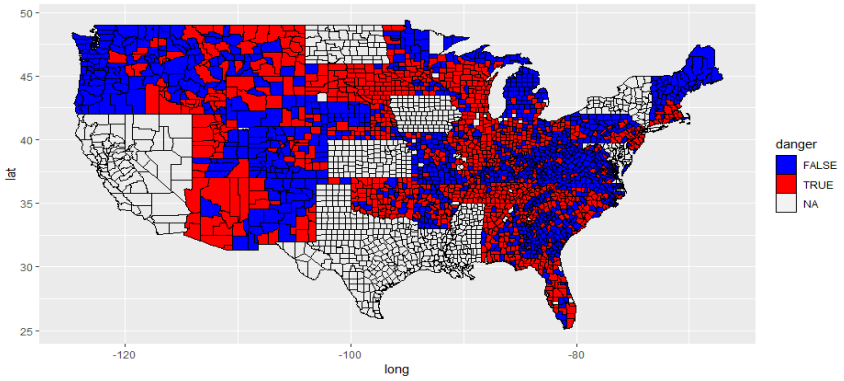


Figure 6: Actual counties' situation (rpart)

In the figure above, the red counties are "dangerous counties" because their either population density or case percent are high. The blue counties are the places that are not dangerous.

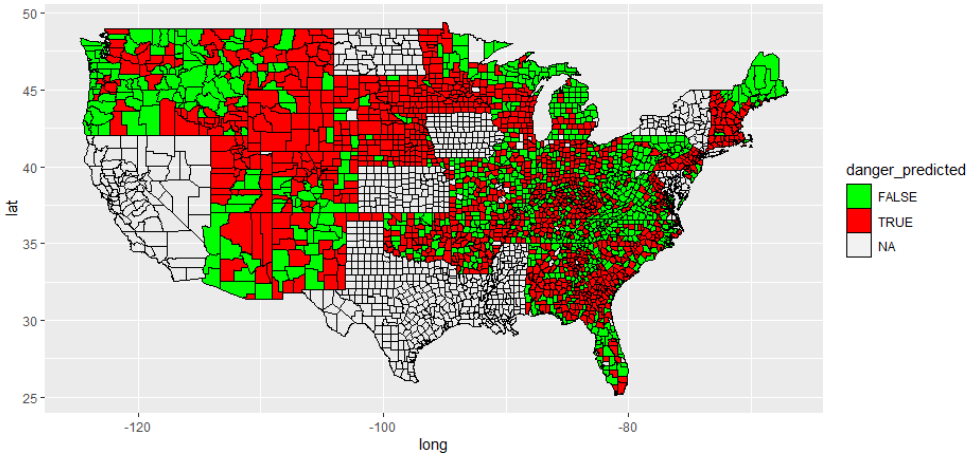


Figure 7: Predicted counties' situation (rpart)

In the figure above, the red counties are the dangerous counties determined by the predication, and the

blues are the non-dangerous counties.

To visualize how good the rpart classification model is, the project uses a two by two confusion matrix to represent the result. The rows are the actual result, and the columns are the predicted result. Therefore in the figure below, the blue blocks are the ones which made the good prediction, and the orange blocks refers to the bad predictions. (1)

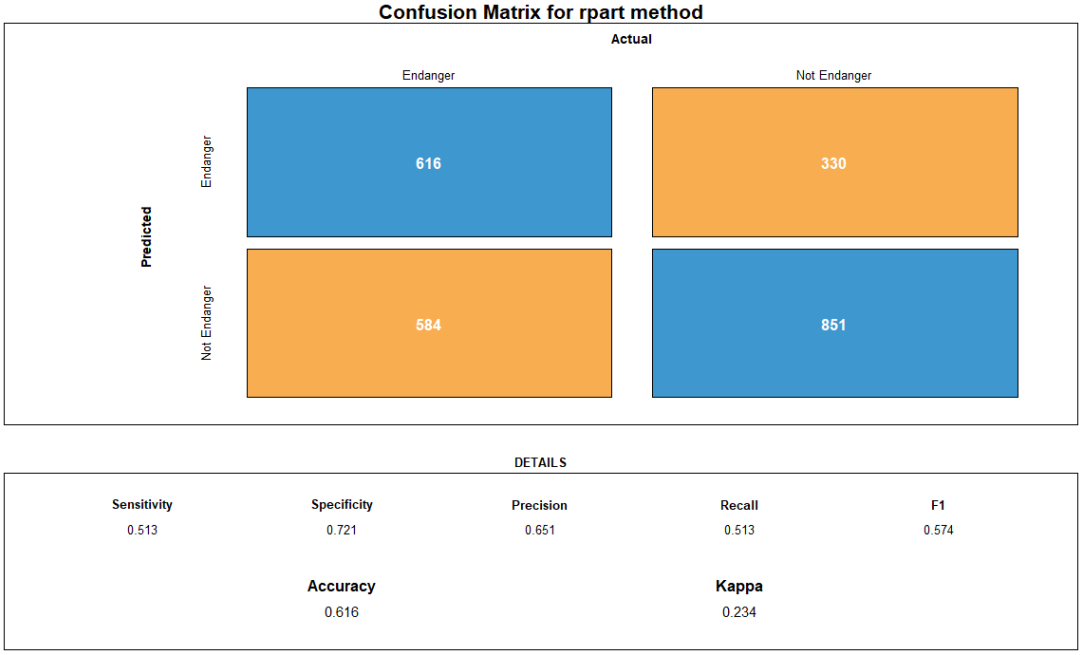


Figure 8: Confusion matrix for rpart method

Accuracy = (TN + TP)/(TN+TP+FN+FP) = (Number of correct assessments)/Number of all assessments)

As we can see, the accuracy of this model is .616. This model is relatively accurate, and does a good job of predicting the danger correctly a majority of the time. However, this model only uses population density and case percentage. As we saw previously in evaluating this data set, population density is not the best identifier of how a county will be impacted by COVID.

Sensitivity = TP/(TP + FN) = (Number of true positive assessment)/(Number of all positive assessment)

Specificity = TN/(TN + FP) = (Number of true negative assessment)/(Number of all negative assessment)

As suggested by the above equations, sensitivity is the proportion of true positives that are correctly identified; it shows how good the test is at detecting a county is in danger. Specificity is the proportion of the true negatives correctly identified. Therefore, in the confusion matrix above, the specificity is relatively higher than the sensitivity, so it means that the classifier is more good at identifying safe counties, and its functionality on identifying dangerous counties is also great.

Section 1.3: Classification model with k-nearest neighbors method

The next step is to model the classification using the k-nearest neighbors classifier. The project uses the same modeling function, same tunelength, but different classification method. Then, it will do the prediction as what it did in the previous section.

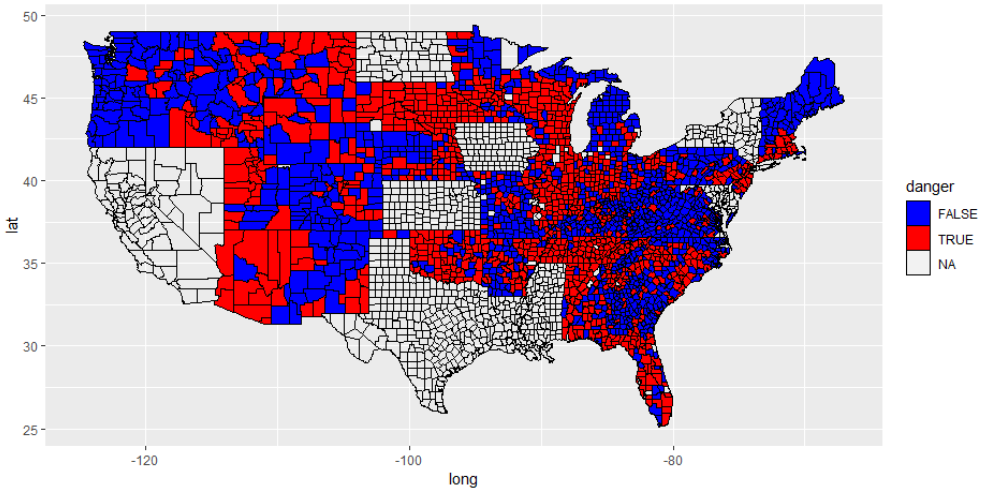


Figure 9: Actual counties' situation (KNN)

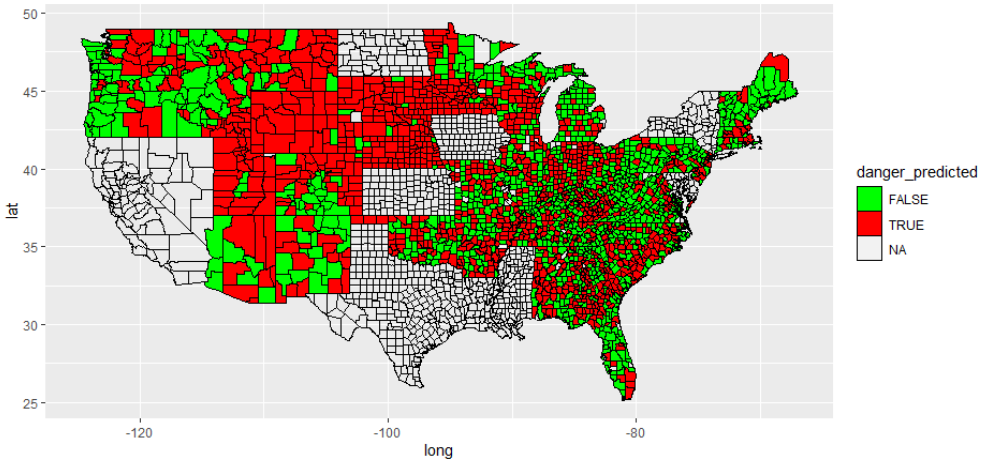


Figure 10: Predicted counties' situation (KNN)

Then, it will build the confusion matrix again to measure the accuracy of k-nearest neighbors classifier.

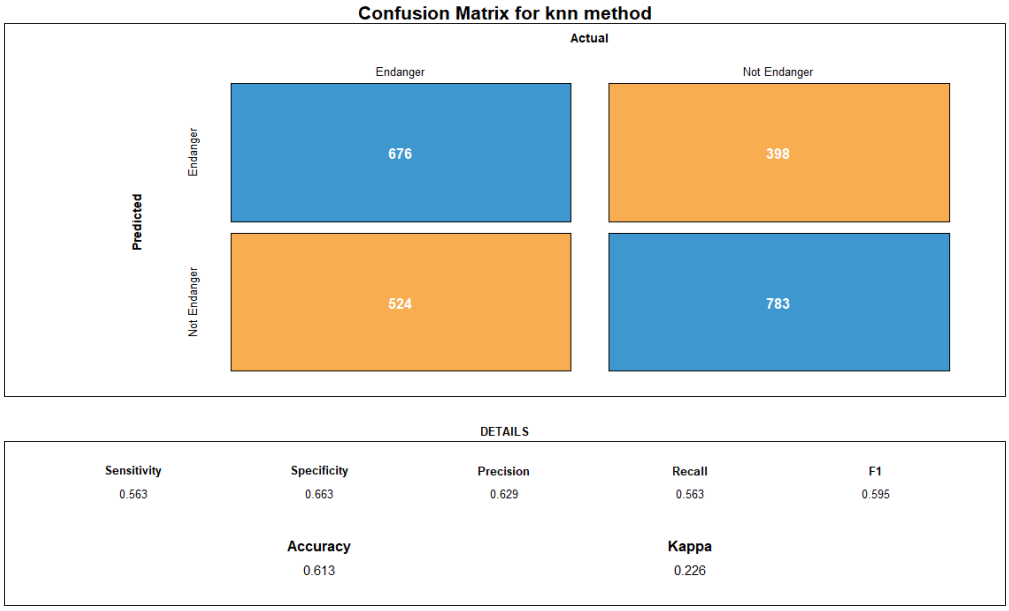


Figure 11: Confusion matrix for knn classifier

The result from the confusion matrix above shows a similar accuracy as the CART classifier at the

previous section. However, with slightly a higher sensitivity and a lower specificity, the knn classifier could be better at identifying dangerous counties.

Section 1.4: Classification model with rpart(CART) method with cost

For the third classification model, the project will go back to the CART classifier but use a cost matrix to prevent potential dangerous prediction. In this section, the cost matrix is like the matrix below; a dangerous county which is mispredicted as a non-dangerous county will get a high penalty, whereas a right prediction will not have any penalty.

0	5
1	0

Figure 12: Cost Matrix for rpart-c

It will then create its corresponding decision tree, and the result shows that it is different compared to a normal decision tree in the standard CART classifier.

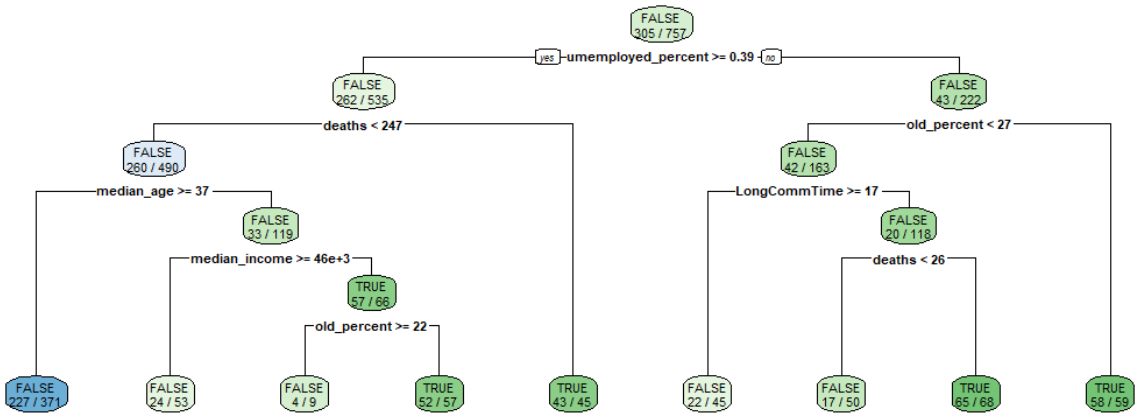


Figure 13: Decision tree for rpart method classification model with cost

The project will then do the prediction and print out the result on the map.

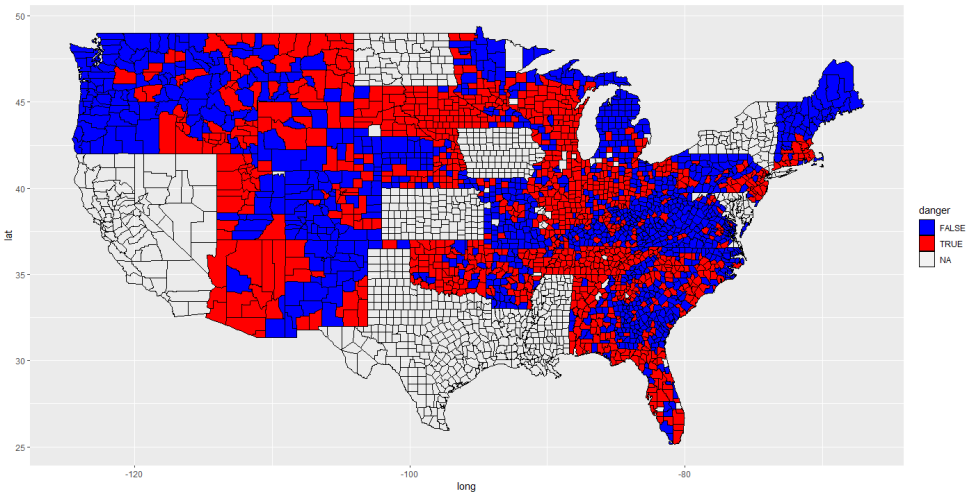


Figure 14: Actual counties' condition (rpart with cost)

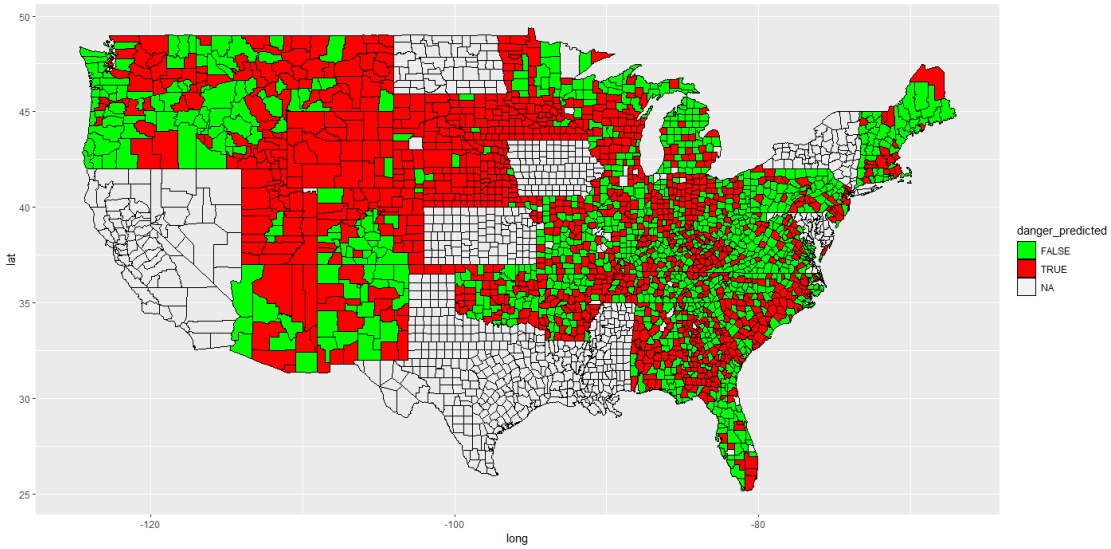


Figure 15: Predicted counties' condition (rpart with cost)

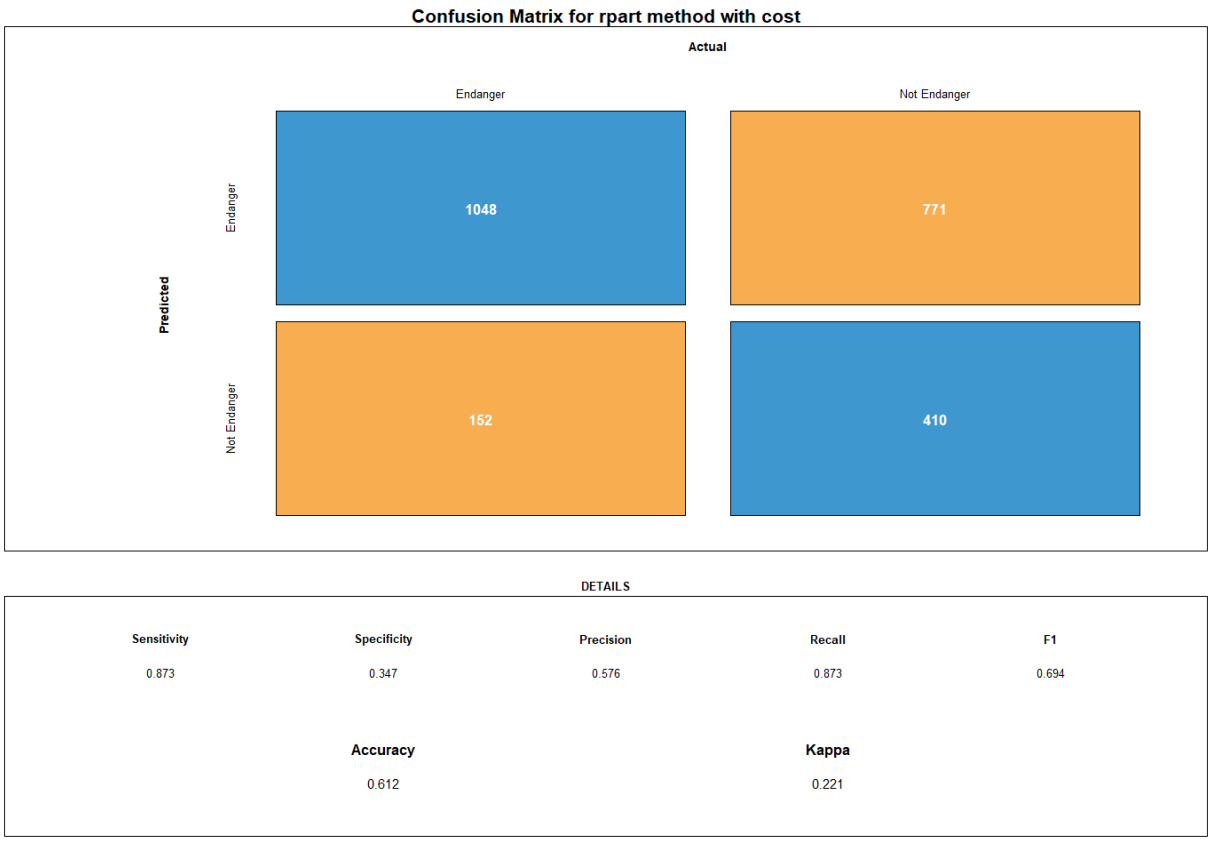


Figure 16: Confusion matrix for rpart method with cost

Even though this model is more complex than previous models, its accuracy is still hovering in the .61 range. However, this model is easily tweaked by using the cost matrix. This matrix allows the user to stress the importance of different missed classifications by changing the value of that classification. This allows for importance to be placed more easily than in other models. However, we are still limiting the number of variables used which means we are not necessarily using all the information that could prove useful.

From the high sensitivity, it could tell that the classifier is more likely to identity a county as "dangerous;" that is because the project has set a penalty for the classifier, so to minimize the penalty it got, it will try to predict more toward "dangerous" rather than "non-dangerous."

Section 1.5: Classification model with Conditional Inference Tree

In this section, the project will do the similar classification with the Conditional Inference Tree (Decision Tree).

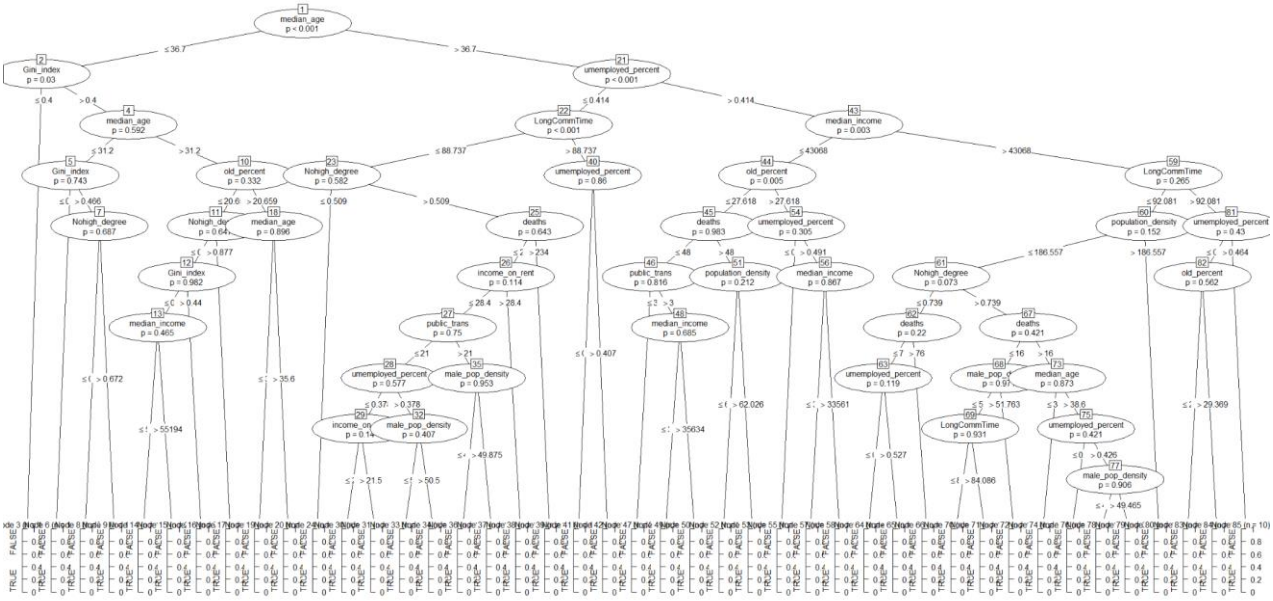


Figure 17: Conditional Inference Tree

The model will create a huge decision tree with all the attributes in the dataset. Later when the project does the prediction, it will determine the class based on the result from the tree.

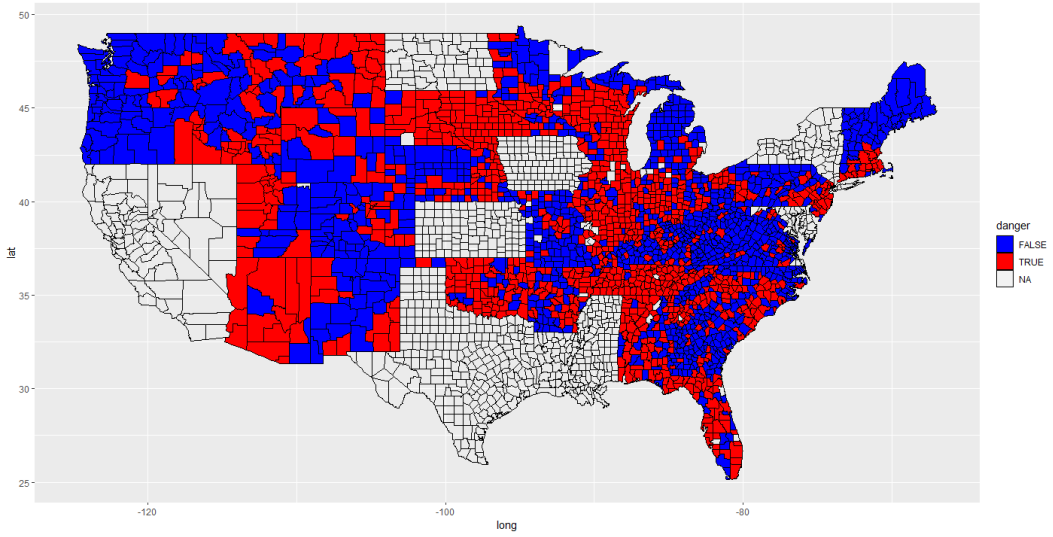


Figure 18: Actual counties' condition (Ctree)

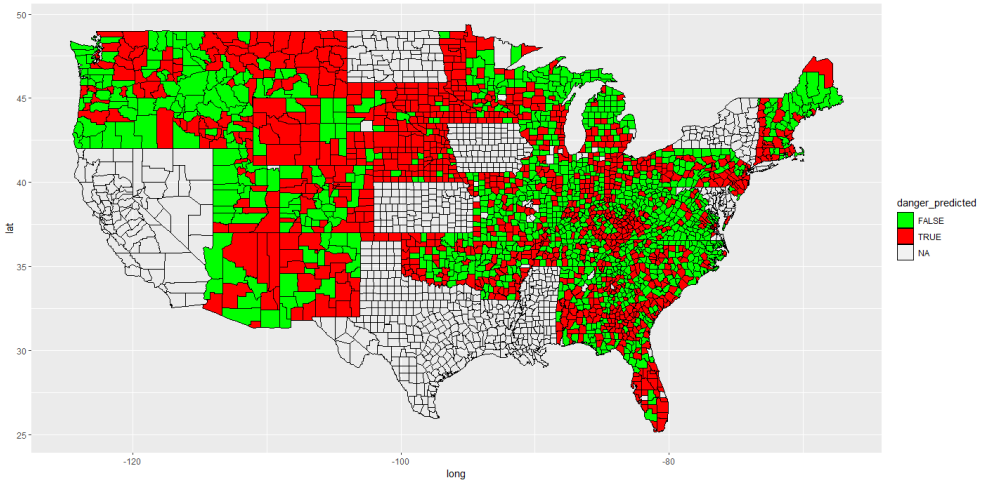


Figure 19: Predicted counties' condition (Ctree)

Figure 20: Confusion matrix for Conditional Inference Tree classifier

This model has a slightly higher accuracy than the models thus far, but it is still in the .61 range. This

shows that using all of the available variables may not make our model more accurate. This means that some of the variables we have are not good predictors of COVID-19's impact.

Compared to models in the previous sections, the model with conditional inference tree has reflected high numbers on both the sensitivity and specificity, which may refer to that the model has overall less misprediction, and it may explain why the accuracy it got is higher than the previous models.

Section 1.6: Comparison between models

After building up four different classifiers, the project puts them together and compares their performance. The code uses resamples to visualize the comparison result, and the result as what is showing in the table below.

Accuracy	Min	1st Qu.	Median	Mean	3rd Qu.	Max	NA's
CART	0.6266667	0.6721206	0.7039474	0.7054097	0.7359649	0.7763158	0
KNN	0.6933333	0.7426316	0.8039132	0.7805131	0.8169697	0.8289474	0
CART_C	0.5263158	0.5900000	0.6159649	0.6180237	0.6546053	0.6973684	0
CTREE	0.7532468	0.7900000	0.8278947	0.8203368	0.8415789	0.8947368	0
KAPPA	Min	1st Qu.	Median	Mean	3rd Qu.	Max	NA's
CART	0.2134831	0.2839709	0.3728974	0.3670913	0.4299044	0.5142857	0
KNN	0.3575419	0.4523095	0.5824829	0.5320617	0.6023188	0.6448087	0
CART_C	0.1325301	0.2398399	0.2885542	0.2848572	0.3425970	0.4280105	0
CTREE	0.5150812	0.5642077	0.6387998	0.6291669	0.6690200	0.7753141	0

Table 5: Models' Comparison Result

From the result above, it can be found out that the model with condition inference tree has a higher accuracy and KAPPA compared to others, which means that it does better on doing the prediction, and the result distribution is also consistent. Knn classifier and CART classifier have similar performance, but from the results from previous sections, it can knn classifier has a higher specificity number. CART-C is identified as the worst classifier overall; that is because the model will try its best to avoid a wrong prediction with penalty, and since the attributes the project used for prediction do not have strong connections on determine whether a county is dangerous or not, it will do more prediction toward the choice that has no penalty. However, identifying more counties as dangerous will create more wrong

predictions, which let some safe counties become dangerous counties.

As a result, the CTREE classifier would be more suitable on identifying dangerous overall; KNN classifier and CART classifier are similar on prediction performance, with minor difference on specificity and sensitivity. The CART classifier model is derived from the standard CART classifier, but it may depend more on the connection between the attributes and the classes; if the dependencies are poor, the prediction may become an arbitrary selection or a blind selection that tries to get a lower penalty.

Section 2: Classification with spread rate and vaccination rate

In this section, the project will not only use the data from the previous section but also add it with the spread rate of COVID-19 and vaccination rate in each state; the first half of the "TotalData2," those attributes are selected because they have shown greater importance to the classification in the previous section.

The project will split the "TotalData2" in two parts, training and testing data, with the method in the first section, but the data will be in states instead of counties. Therefore, the training data would be the information for states: Maryland, California, Mississippi, Delaware, Texas, New York, Iowa, Kansas, Nevada, and North Dakota; the rest states will be used for testing. In order to compare the result between those two sections, the project tries to make the implementation as close as possible.

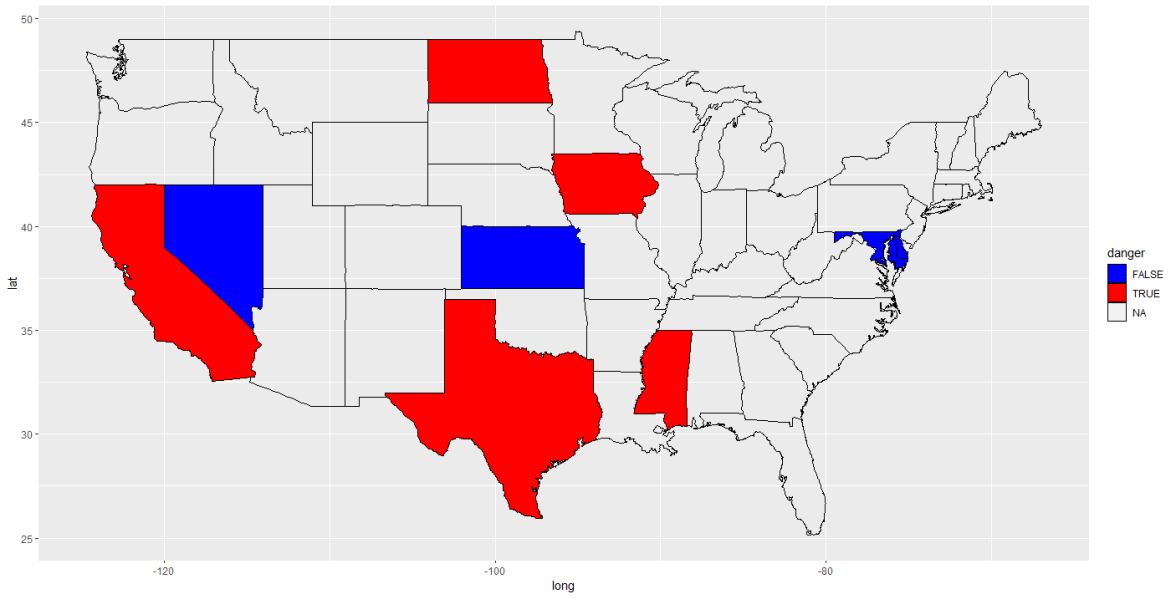


Figure 21: Training states on the map.

The training result shown on the map has two colors: the red states are dangerous states under the pandemic, and the blue states are the relatively safer states. This result shows some similarity with the training result in the first section. For example, California and North Dakota are marked as dangerous states in both sections. On the contrary, this result also has some differences, such as the blue Kansas.

When it built the decision tree for the CART classifier, the figure represents that the vaccination data does help the classification; a state with a higher number of vaccines is usually not a dangerous state.

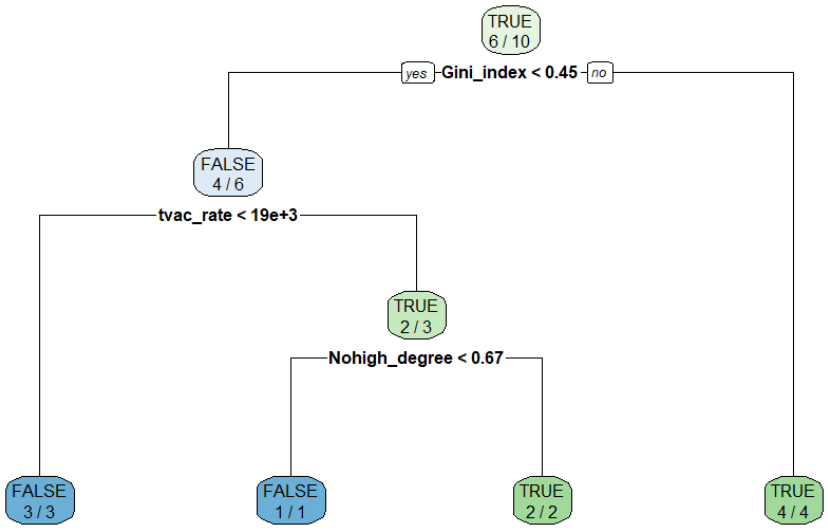


Figure 22: Decision tree for rpart (Section2)

Then, the project will do prediction using the rpart model, the first graph is the actual result determined by the class variable, and the second figure is the one predicted by the model.

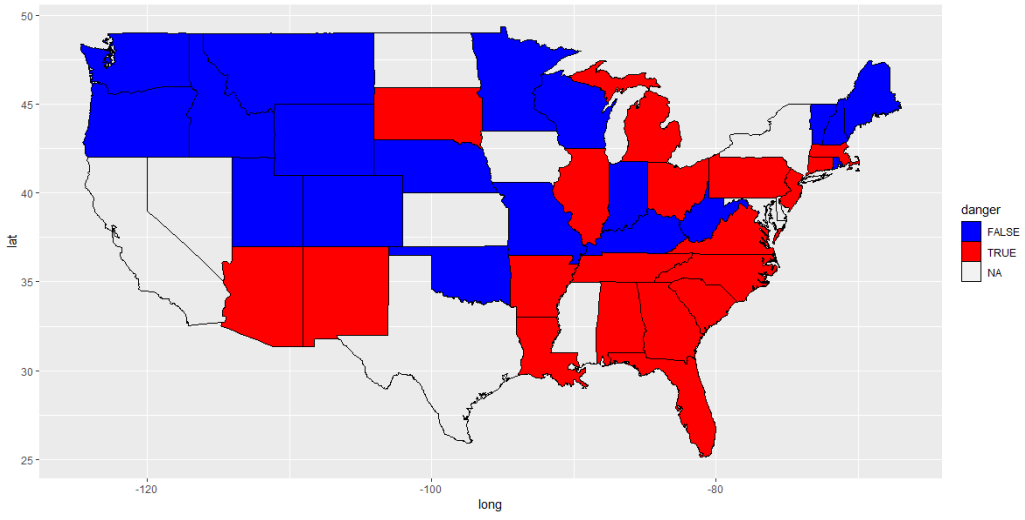


Figure 23: Actual states' condition (rpart-section2)

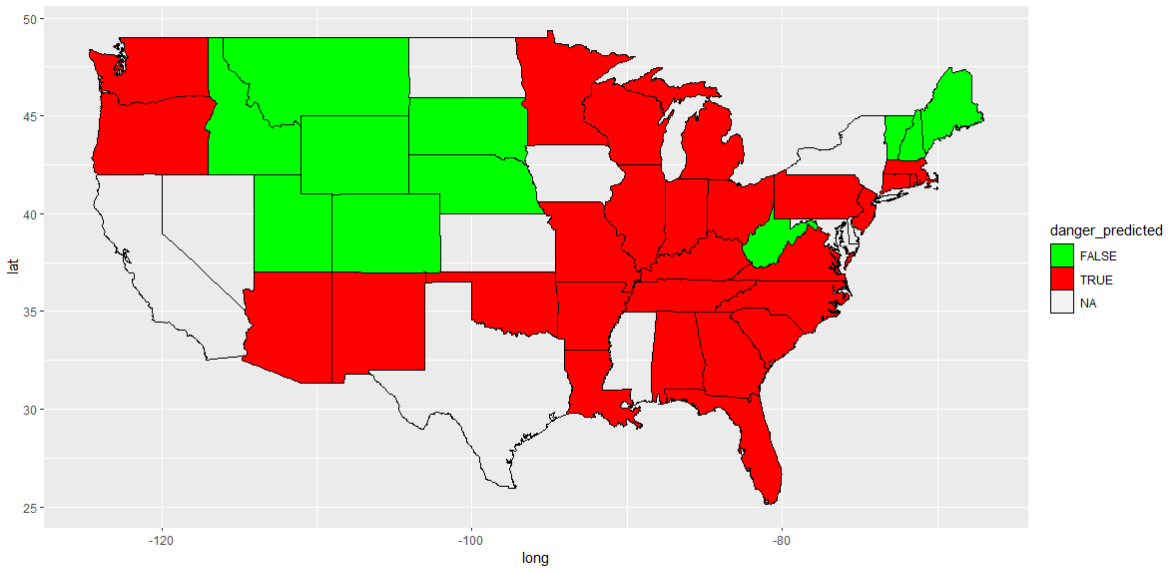


Figure 24: Predicted states' condition (rpart-section2)

The red states are the states that are dangerous, and the blue or green states are the safe states. The two figures do show some similarity, and the result is also proved in the confusion matrix with a high accuracy.

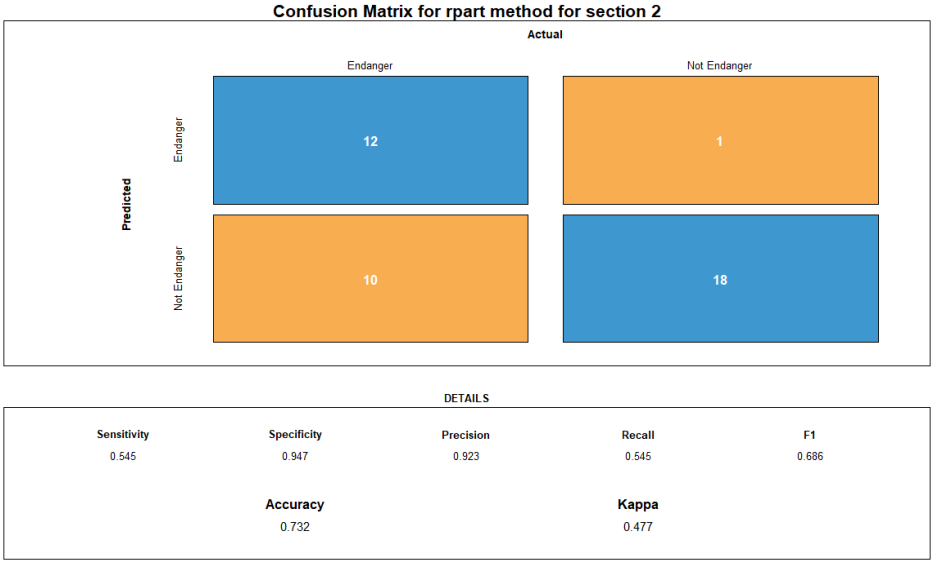


Figure 25: Confusion matrix for rpart method (section 2)

For this last model, we shift our focus from counties to states as we were unable to find data on

individual counties' vaccination rates. However, we see that by including vaccination rates, the accuracy of our model increases to .732, and the Kappa value of .477 shows that the variables are doing a good job of agreeing. This all helps create a more accurate prediction. We can see that our model was more accurate in predicting accurately endangered and safe states, however it still struggled at times.

By including the data on vaccination rates, we are relying more heavily on things we know have an impact of COVID rather than just things that show a correlation. Also, by decreasing the number of classifications, we have more data for each individual state. This means our model is able to be slightly more accurate overall, but it loses a significant amount of its specificity.

EVALUATION

In order to determine how useful our model would be for our stakeholder, in this case, the CDC, we must look at the predictions our model made and compare those predictions to the real information. Using the confusion matrices, we see that the average accuracy of our models is .638. Although ideally our models would have a 1.00 accuracy, the accuracy we have is impressive.

For example, when looking at Figure 16, we have a cost matrix associated with the confusion matrix. This cost matrix, as previously described, allows for the evaluation of the model's accuracy. When the model correctly predicts a county's susceptibility, there is no cost involved since the correct prediction was made. When it predicts that a county is safe but it actually is not safe, there is a large punishment since this could put the population in danger. However, when the model predicts that a county is dangerous when it isn't actually endangered, there is a cost. This cost is much smaller, since when dealing with a dangerous virus, it is better to be overcautious than cautious. There still must be a cost involved however, since the prediction was wrong. This is due to the limited resources available to fight the COVID-19 virus.

We are able to change the costs associated with our predictions, and in an effort to limit the cost, the model will relax the constraints a bit to avoid predicting a county is safe when in reality it is not.

When the model predicts that a county is endangered incorrectly, it could lead the CDC to send limited resources to a county that is not in need of those resources. This limits the amount of resources available to counties that are in need of extra resources. This could be a critical mistake that could cost people their lives, so it is extremely important to make our model as accurate as possible.

DEPLOYMENT

Our model could be used to roughly predict the impact COVID-19 would have in any given county based on some simple census data. This would be important to the CDC for determining where it would be important to distribute resources in the case of a new strand of the COVID-19 virus.

This could be essential in predicting COVID-19 effects early and preparing to fight the virus preemptively. As we saw in our first assignment, the time to stop COVID-19 is before it starts, as any action taken after the virus begins spreading has limited impact. By using our model, it would be possible to predict the future effects of the virus before it is too late to put mandates and orders in place to prevent its spread.

In order to retain the most accuracy and produce the most effective model, it would be necessary to update the model as often as new data is received, likely daily. Since COVID-19 has been constantly evolving, it is hard to know whether the information we have from 12 months ago is still relevant to the strand of COVID we are dealing with today. We have likely seen 3 different strands of the virus over the past year, and all of them are slightly different than the first. Although they are all COVID, and former versions can be useful to predict future versions, it will always be more accurate to use data pertaining to what strand we are dealing with today. This means that having a month to two months of data used in our classification should be more relevant to what we are dealing with than using all the data we have. This gets tricky, however, since the more data we have, the more accurate our prediction will be. However, since we are not dealing with a static virus, keeping our information up to date is extremely necessary in fighting COVID-19 and its potential new strands most effectively.

REFERENCES

1. https://stackoverflow.com/questions/23891140/r-how-to-visualize-confusion-matrix-using-the-caret-package/42940553
2. https://www.census.gov/topics/income-poverty/income-inequality/about/metrics/gini-index.html