COVID-19 Infection Spread Report

ISEN 613-603

November 30th, 2020

Hadrien Fleurat
Melissa Glass
Nicholas Sims
Yuting Zhou

Executive Summary

This project aimed to create and develop a model to predict the current speed of the spread of COVID-19, a highly infectious respiratory disease that led to a worldwide pandemic throughout 2020. This report details the process of developing three potential training models, comparing and selecting one of the models, testing and evaluation, and final development and improvement using test data.

Phase 1 involved developing three mathematical models to predict R_t values based on training data that was supplied from 152 days, using 66 predictors. The training data used was a standard data set in the statistical analysis program, R. The data set included 11 days of 6 sets of measurements for mobility of people in different locations, ranging from the workplace, retail, transportation, residential, grocery stores, and parks.

Model 1, which used the random forest method, had the lowest test mean squared error (MSE) by far, at 0.0149. This error was extremely attractive given that is showed a model that was highly accurate based on the trained data. Model 2, which used the lasso method, had the second highest test MSE of 0.0315. This MSE is also highly attractive because predictors that are deemed insignificant to that specific model are excluded, which is extremely useful in cases where p is extremely large, like in the COVID-19 study. Model 3, which used the boosting method, had the highest test MSE of the models presented, at 0.127. This MSE was still low, but not at all attractive when compared to Models 1 and 2. The best model that our team selected was Model 1: Random Forest. Based on the MSE, the random forest method provided the best prediction for R_t.

Phase 2 involved testing and further development of the best model, selected from Phase 1. Test data for 50 days was supplied and tested using the original random forest method. The model yielded a close prediction, with a test MSE of 0.0091. However, the variance was relatively quite high, as the R_t predictions seemed to oscillate.

The random forest model can be improved by using 10-fold cross validation on the training data to select the optimal mtry value of 16, and ntree value of 150. Combining this change with also using 70% of the training data to train, instead of 50%, the test MSE is decreased 40.1% to 0.005448. The improved random forest model shows workplaces and retail as the most important predictors, which makes sense since those are places where people are in close contact to potentially spread COVID-19.

Table of Contents

Technical Analysis of Model 1: Random Forest

This Random Forest model was generated as an effort to improve upon the decision tree analysis of the COVID data set. The decision tree approach segments the predictor space into several simple regions to make predictions. Its advantage is the interpretability of its results; however, the model was performing poorly. Thus, the Random Forest model presented below was generated in order to improve the result. This is a method based on averaging the results of individual trees. This technical analysis details how we performed this method.

The dataset used is the one given for the project and is composed of 152 data and 65 inputs. It has been randomly cut in half for the training and testing data. The randomForest library was imported in R in order to perform Bagging and Random Forest for this model.

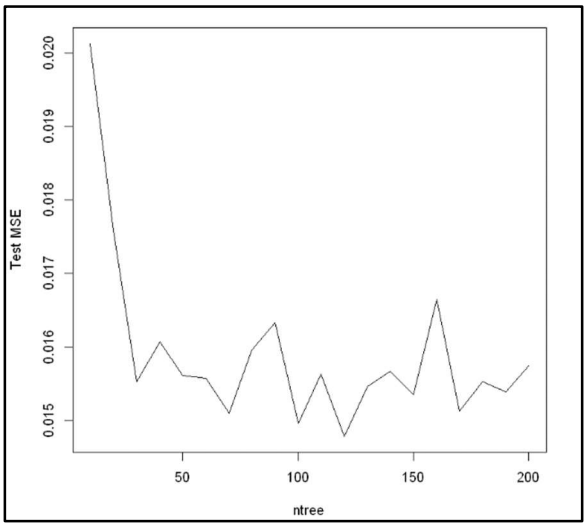


Figure 1. mtry vs. Test MSE

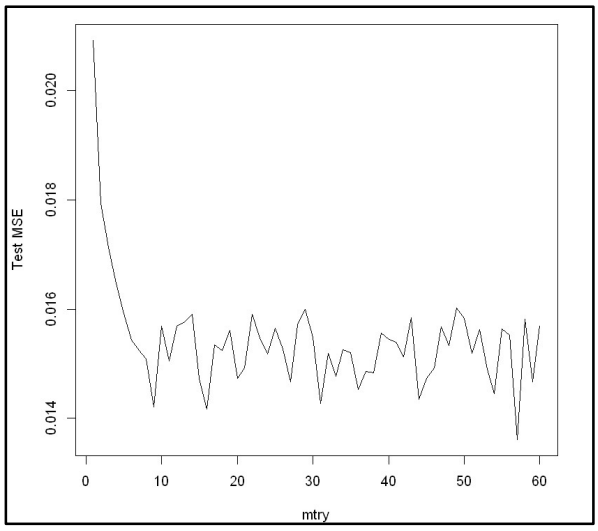


Figure 2. ntree vs Test MSE

To make sure Random forest was the best choice and not bagging, we plotted the test MSE given different mtry. One of the best mtry is $\sqrt{65} \approx 8$. Then we are going to use random forest. In order to select the best number of trees to grow by the random forest, we plotted the Test MSE given different ntree for a selected mtry=8. Any value after 60 give some good result. We decided to take ntree=100

Using mtry= 8 and ntree=100 we can explain 87.72% of the training data with a training MSE of 0.0102 and the test MSE is 0.01495 according to Equation 1.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Equation 1. MSE calculation

Random forest is an average of multiple decision trees, so it is not easy to visualize the most usefull parameter. However, it is still possible view the importance of each variable. Figure 3 shows what are the most important input and how pure the node is. This model shows the most important variable is workplaces_9. The 10 first input explain 32.23% of the total train MSE. The other figure shows the importance of the 20 most important input of this model.

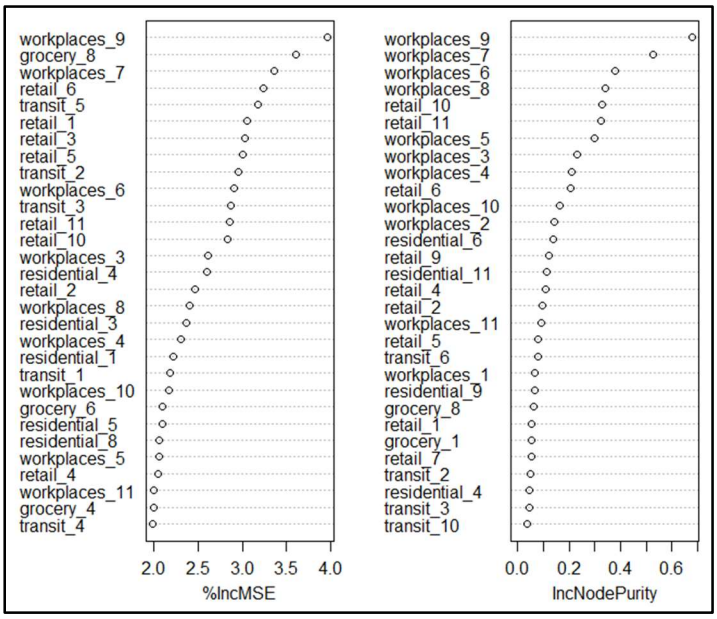


Figure 3. Tuning parameter vs MSE

Table 1. Most important predictors and their coefficients for random forest

Workplaces 9	3.95
Grocery 8	3.60
Workplaces 7	3.35
Retail 6	3.23
Transit 5	3.17
Retail 1	3.05
Retail 3	3.03
Retail 5	3.00
Transit 2	2.95
Workplaces 6	2.90
Transit 3	2.86
Retail 11	2.85
Retail 10	2.83
Workplaces 3	2.61
Residential 4	2.59

As we can see, the most important inputs are essentially all workplaces, retail & grocery, and transit. This intuitively makes sense since those are the areas that COVID infections are most likely to take place. Research has already shown that workplaces and public transport are the biggest “hotspots” for COVID infections.

Technical Analysis of Model 2: Lasso

The Lasso approach is a shrinkage method used to simply models by constraining its coefficients. This is helpful in reducing the model's variance over subset selection models. The goal of the Lasso method is to minimize the tuning parameter λ. Lasso uses a penalty term L1, which is depicted below in Equation 1. The penalty term reduces some coefficients to zero, fully eliminating them from the model. The Lasso method is therefore able to select the important variables for a model and remove other unimportant variables. These leads to an easier to interpret model.

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j| \;=\mathrm{RSS}+\lambda\sum_{j=1}^{p}|\beta_j|.$$

Equation 2. Lasso method minimization equation

For this model, the data set was split in half into a training set and a test set. Next, a matrix was created using the model.matrix() function, which expands factors to sets of dummy variables. Finally, k-fold cross validation function cv.glmnet() is used with an alpha value of 1. This uses the glmnet package in R. This essentially performs the Lasso equation, Equation 2 above, and provides the minimized lambda value which determines the model. The minimum lambda value obtained is used to predict the lasso model. This cross validation is shown in Figure 5 below and gives the best lambda value is 0.011.

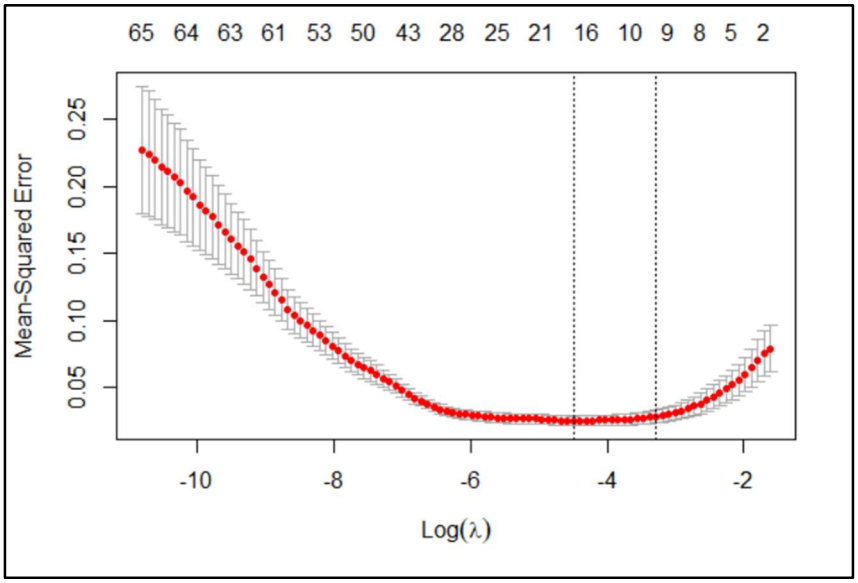


Figure 5. Tuning parameter vs MSE

The prediction of the Lasso model is then calculated using the predict() function. The coefficients of the Lasso model are determined by this function. Since Lasso is able to fully reduce coefficients to zero, only the non-zero variables remain significant. These predictors and their corresponding coefficients are shown below in Figure 6. It is seen that trips to the grocery store, workplace and parks are the main travel occurrences having an effect on the COVID-19 rates. All instances of transit and retail have been fully removed from the model. This lasso model is much more interpretable than others which still contain a large number of predictors, such as the Boosting model presented in the next section.

(Intercept)	grocery_11
1.7585928339	-0.0010583740
parks_11	grocery_10
-0.0001782448	-0.0017258585
work_10	work_9
0.0037602639	0.0020189944
grocery_8	work_8
-0.0016658007	0.0027027310
res_8	work_6
-0.0016871531	0.0005441839
work_5	res_5
0.0025257367	-0.0021083175
grocery_4	park_3
-0.0014273744	-0.0005728256
work_3	work_2
0.0006105094	0.0010440414
park_1	work_1
-0.0002713189	0.0018295922

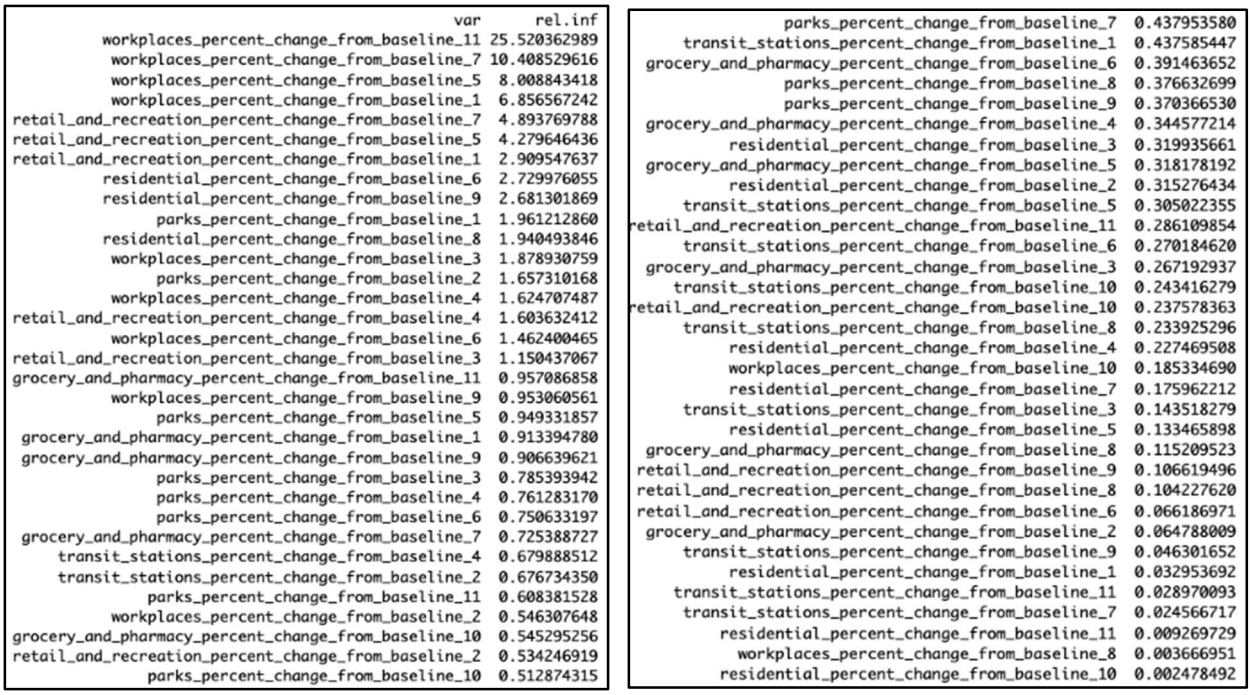
Figure 6. Summary of coefficients from lasso method

Finally, the test mean square error is obtained using Equation 1. The Lasso model produced a training MSE of 0.031499, which is greater than the Random Forest model presented earlier. However, the Lasso model does improve interpretability by reducing the number of predictors in the model.

Technical Analysis of Model 3: Boosting

The boosting approach is a variance reduction method, which can improve our decision trees. Boosting is a sequential process in which each next model which is generated is added so as to improve a bit from the previous model. It does not involve bootstrap sampling, which means each tree is fit on a modified version of the original dataset.

The third model was created utilizing the boosting method. Here we use the gbm package in R, and within the gbm() function, to fit boosted regression trees to our dataset which is from the Google COVID-19 Community Mobility Reports. For this project, we run gbm() with the option distribution = "gaussian" since this is a regression problem.



var	rel.inf
workplaces_percent_change_from_baseline_11	25.520362989
workplaces_percent_change_from_baseline_7	10.408529616
workplaces_percent_change_from_baseline_5	8.008843418
workplaces_percent_change_from_baseline_1	6.856567242
retail_and_recreation_percent_change_from_baseline_7	4.893769788
retail_and_recreation_percent_change_from_baseline_5	4.279646436
retail_and_recreation_percent_change_from_baseline_1	2.909547637
residential_percent_change_from_baseline_6	2.729976055
residential_percent_change_from_baseline_9	2.681301869
parks_percent_change_from_baseline_1	1.961212860
residential_percent_change_from_baseline_8	1.940493846
workplaces_percent_change_from_baseline_3	1.878930759
parks_percent_change_from_baseline_2	1.657310168
workplaces_percent_change_from_baseline_4	1.624707487
retail_and_recreation_percent_change_from_baseline_4	1.603632412
workplaces_percent_change_from_baseline_6	1.462400465
retail_and_recreation_percent_change_from_baseline_3	1.150437067
grocery_and_pharmacy_percent_change_from_baseline_11	0.957086858
workplaces_percent_change_from_baseline_9	0.953060561
parks_percent_change_from_baseline_5	0.949331857
grocery_and_pharmacy_percent_change_from_baseline_1	0.913394780
grocery_and_pharmacy_percent_change_from_baseline_9	0.906639621
parks_percent_change_from_baseline_3	0.785393942
parks_percent_change_from_baseline_4	0.761283170
parks_percent_change_from_baseline_6	0.750633197
grocery_and_pharmacy_percent_change_from_baseline_7	0.725388727
transit_stations_percent_change_from_baseline_4	0.679888512
transit_stations_percent_change_from_baseline_2	0.676734350
parks_percent_change_from_baseline_11	0.608381528
workplaces_percent_change_from_baseline_2	0.546307648
grocery_and_pharmacy_percent_change_from_baseline_10	0.545295256
retail_and_recreation_percent_change_from_baseline_2	0.534246919
parks_percent_change_from_baseline_10	0.512874315
parks_percent_change_from_baseline_7	0.437953580
transit_stations_percent_change_from_baseline_1	0.437585447
grocery_and_pharmacy_percent_change_from_baseline_6	0.391463652
parks_percent_change_from_baseline_8	0.376632699
parks_percent_change_from_baseline_9	0.370366530
grocery_and_pharmacy_percent_change_from_baseline_4	0.344577214
residential_percent_change_from_baseline_3	0.319935661
grocery_and_pharmacy_percent_change_from_baseline_5	0.318178192
residential_percent_change_from_baseline_2	0.315276434
transit_stations_percent_change_from_baseline_5	0.305022355
retail_and_recreation_percent_change_from_baseline_11	0.286109854
transit_stations_percent_change_from_baseline_6	0.270184620
grocery_and_pharmacy_percent_change_from_baseline_3	0.267192937
transit_stations_percent_change_from_baseline_10	0.243416279
retail_and_recreation_percent_change_from_baseline_10	0.237578363
transit_stations_percent_change_from_baseline_8	0.233925296
residential_percent_change_from_baseline_4	0.227469508
workplaces_percent_change_from_baseline_10	0.185334690
residential_percent_change_from_baseline_7	0.175962212
transit_stations_percent_change_from_baseline_3	0.143518279
residential_percent_change_from_baseline_5	0.133465898
grocery_and_pharmacy_percent_change_from_baseline_8	0.115209523
retail_and_recreation_percent_change_from_baseline_9	0.106619496
retail_and_recreation_percent_change_from_baseline_8	0.104227620
retail_and_recreation_percent_change_from_baseline_6	0.066186971
grocery_and_pharmacy_percent_change_from_baseline_2	0.064788009
transit_stations_percent_change_from_baseline_9	0.046301652
residential_percent_change_from_baseline_1	0.032953692
transit_stations_percent_change_from_baseline_11	0.028970093
transit_stations_percent_change_from_baseline_7	0.024566717
residential_percent_change_from_baseline_11	0.009269729
workplaces_percent_change_from_baseline_8	0.003666951
residential_percent_change_from_baseline_10	0.002478492

Figure 7. Summary of coefficients from boosting method

The above Boosted Model is a Gradient Boosted Model which generates 300 trees and the shrinkage parameter (\lambda= 0.01\) which is also a sort of learning Rate. Next parameter is the interaction depth which is the limit of the depth of each tree we want to do. So here each tree is a small tree with only 4 splits. The summary of the Model gives a feature importance plot, which is shown below in Figure 8. The variables in Figure 7 are listed from the most important variable at the top to the least important variable at the end of the list.

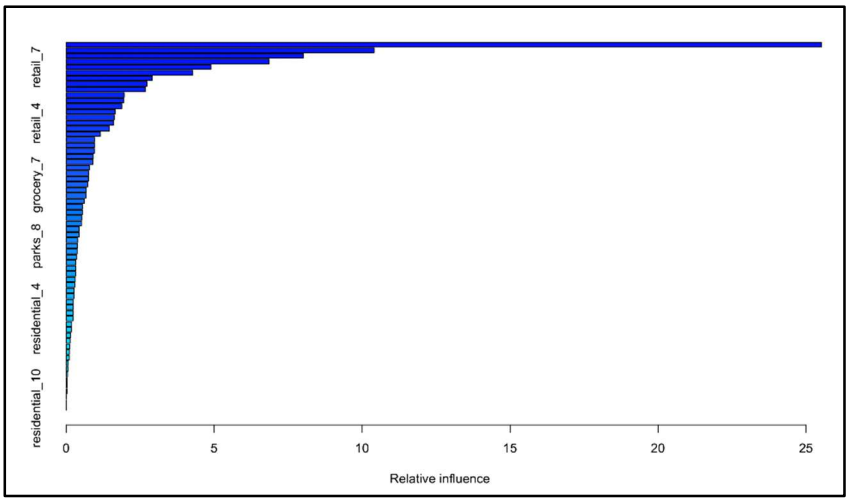


Figure 8. Summary of coefficients from boosting method

Based on the above result, we see that "workplace_percent_change_from_baseline_11" is by far the most importance variable. We can also produce partial dependence plots for this variable. The plot is shown in Figure 9. This plot illustrates marginal effect of the selected variable on the response after integrating out the other variables. From the above plot, in this case, as we might expect, Rt value is increasing with the variable "workplace_percent_change_from_baseline_11".

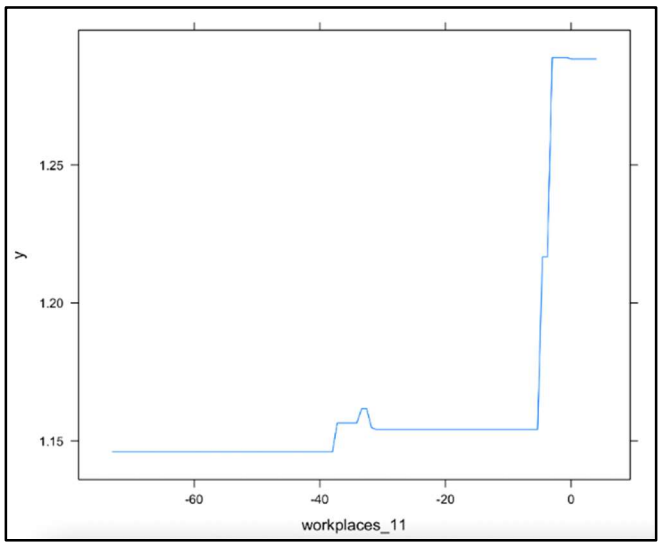


Figure 9. Dependence plot for Workplace_11

Finally, we use the boosted model to predict Rt on the test set and calculate the test MSE using Equation 1, with the code shown below in Figure 10. The test MSE obtained is 0.127; larger to the test MSE for random forests and Lasso.

```
> gbm.test<-predict(gbm.rt,newdata = test,n.trees = 300)
> rmse(gbm.test, test$Mean.R.)
[1] 0.1265238
```

Figure 10. Test MSE calculation for boosting method

Model Comparison & Selection

In order to best predict future R_t values for COVID-19, Models 1, 2, and 3 were compared primarily using the training test error. However, there is more to be considered such as interpretability, performance, and sensitivity to variance and bias.

Model 1, which uses the random forest method, has the lowest test MSE by far, at 0.0149. This MSE is extremely attractive given that is shows a model that is highly accurate based on the trained data. Random forest models achieve low training MSEs by decorrelating the trees, that is, when the decision tree is built, a random sample of the predictors are chosen as split candidates, of which only one predictor can be used. At each split, a new sample of predictors is taken, and thus prevents the model from only considering the "strong predictors". Random forests are a major improvement over normal and bagged decision trees because of this decorrelation, which reduces the variance in the resulting trees, and therefore produces a more reliable model.

Model 2, which uses the lasso method, has the second highest test MSE of 0.0315. This MSE is still highly attractive because it is so low, however the lasso method must be analyzed to see its pros and cons. Lasso models use the tuning parameter λ to select predictors to include in the final model. This is a very similar approach to the ridge regression method; however, the lasso approach allows for the coefficient of a predictor to be lowered to exactly zero. This means that predictors that are deemed insignificant to that specific model, which is extremely useful in cases where p is extremely large, like in the COVID-19 study which is better for interpretability. This also prevents overfitting of the data. However, the lasso approach yields features that will be highly biased to the training data, and final predictors that be different for each case of bootstrapped data.

Model 3, which uses the boosting method, has the highest test MSE of the models presented, at 0.127. This MSE is low, but not at all attractive when compared to Models 1 and 2. Boosted models are easy to read, since the boosting method is similar to random forests, and is highly preventative of overfitting the data. However, boosting is sensitive to outliers, but also that since the trees are grown sequentially, the interpretability is higher, but at the cost of a lower precision in prediction, as seen in the training MSE.

The best model that our team has selected is Model 1: Random Forest. Based on the MSE, the random forest method provided the best prediction for R_t. Additionally, the method reduces variance, and allows for analysis on which predictors are most and least important. Although the random forest model may be limited based on the range of the training data, our team believes it gives the best chance of accurately and precisely predicting future R_t values based on the test data that will be provided. Once the test data is acquired and run through the random forest model, an analysis will take place to see if any improvements can be made to the model and increase its precision.

Best Model Evaluation and Testing

In order to evaluate our model, we tested our best model over the new dataset. We obtain a **test MSE of 0.0091** which is even better than our previous training test-MSE of 0.01495 that made us choose this model. To better understand how our model perform we draw a graph to see the performance of our model compared to the real output.

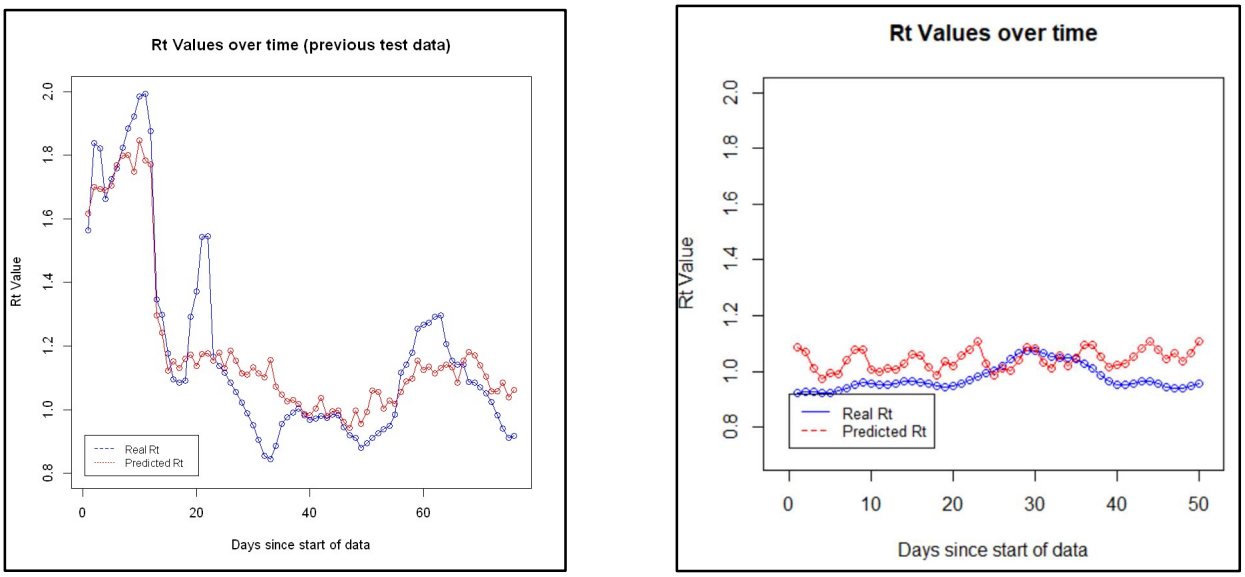


Figure 11. Rt Values over time (Training data vs. Test data)

The right-hand figure illustrates our model performance on the test data. The test data shows a much more constant real R_t, and a relatively close predicted R_t that seem to oscillate. The graphs show that the average of the predicted fit is similar to the average of the real fit. The model is able to predict some change of trend, however, the predicted fit doesn't really predict the real fit. In the training data, the big spike in missed. When drawing the predicted fit and the real fit of our previous data (the test dataset), the predicted values are following more closely the real one. However, the spikes were missed as well.

A few elements might explain why this model is not performing as well as expected:

- The spike might be hard to predict because they may be part of the variance of the data
- Our model does not take into account the current number of cases. For example, a same number of people going to a store while there is no virus or while the transmission is high will have the same answer from our model. However, we know that in the latter more contamination will occur and Rt will be more important.

Model Development & Improvement

After receiving the test data, further improvement to the Random Forest model was implemented. This involved only several minor changes, but produced a dramatically improved final test mean standard error over the original model submitted in Phase I.

The first change implemented was how the data was split. In the Phase I model, the data was simply split in half to form training and test data sets. Since more data is available, the Phase II model was created using 70% of the data points to train the model. The remaining 30% served as the test data set. The other changes in the Phase II model included changing the input parameters for the Random Forest model. For example, mtry is the number of variables chosen as candidates for each split. These were randomly selected for Phase 1. An mtry value of 8 was used based on the equation mtry = sqrt (total number predictors). This value was altered from 8 to 16 to improve the specificity of the model. This change is further described next.

A 10-fold cross-validation was performed, and the best mtry was found to be mtry = 16. This typically limits the interpretability of the model, but in this case primarily served to reduce the test MSE. Additionally, the ntree parameter, which determines the number of trees grown, was increased from 100 to 150. By a large enough number for ntree, this ensures a sufficient number of trees grown so that each input is predicted at least several times. Increasing the ntree value past 150 produced diminishing returns, therefore it was determined that this value was sufficient to improve the model. The analysis discussed in the Best Model Evaluation and Testing portion of this paper discuss why these changes did not cause an overfit of the data. Below in Figure 13, an improved plot of the test vs real data is shown to be very close. The MSE for this improved model is calculated to be **0.005448**, an improvement of 40%. Apart from the oscillations caused by variance, the model is very accurate based on this test data.

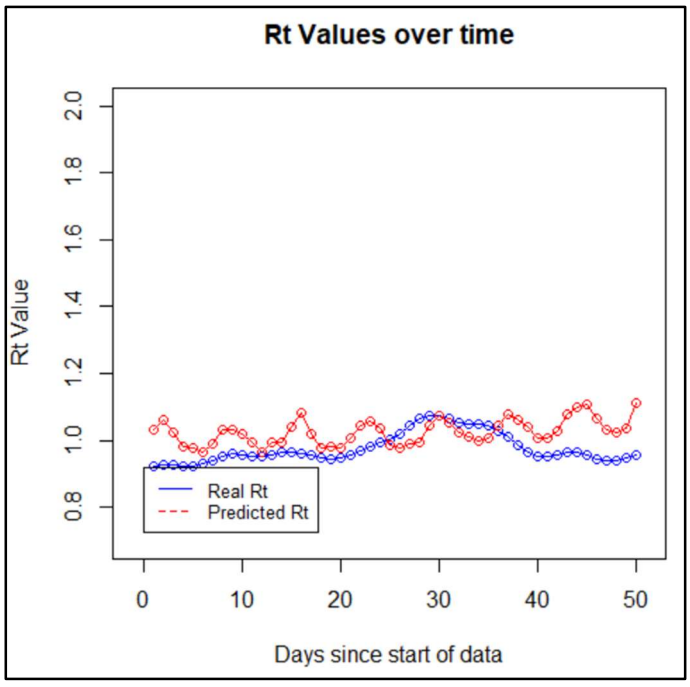


Figure 12. Improved model plot

Based on the above improvements we made, the Figure 13 and Table 2 are shown our new result of what are the most important input and how pure the node is.

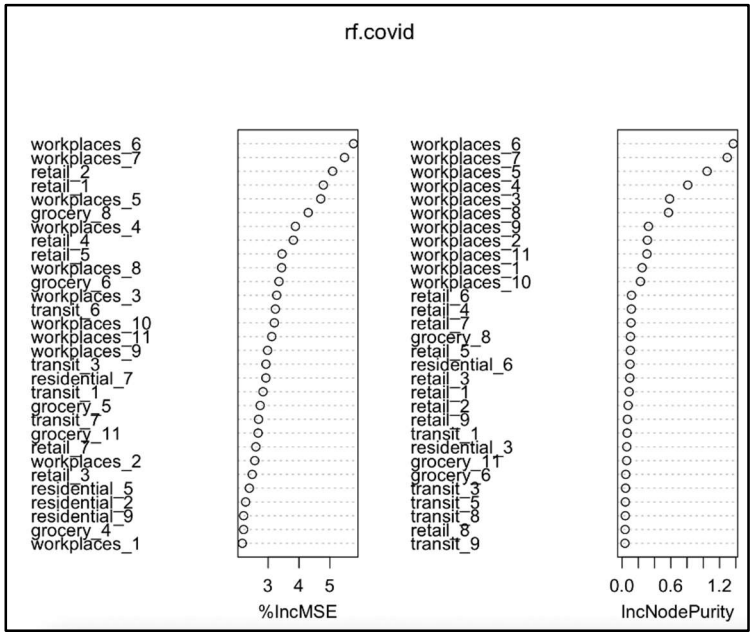


Figure 13. New tuning parameter vs. MSE

Table 2. New most important predictors and their coefficients for random forest

Workplaces 6	5.77
Workplaces 7	5.48
Retail 2	5.09
Retail 1	4.79
Workplaces 5	4.70
Grocery 8	4.30
Workplace 4	3.88
Retail 4	3.82
Retail 5	3.45
Workplaces 8	3.44
Grocery 6	3.35
Workplace 3	3.28
Transit 6	3.23
Workplaces 10	3.20
Workplaces 11	3.12

As per Table 2, the workplaces_6 is the most significant factor. It's different from the original Random Forest model. Similarly, the important inputs are essentially all workplaces, retail & grocery, and transit. These make sense, since the workplace and retail stores are where people are in the most random contact with the largest group of individuals that may increase the spread of COVID-19.