Twitter API
COVID Research

By: Katelyn Chen and Christina Li

Introduction

Our Case

- During this pandemic, there have been an increase usage of social media platforms such as Twitter, Facebook, Instagram, etc
- We wanted to focus on Twitter specifically COVID-19 tweets
- After taking Online Social Networking, there were a lot of similarities in our goals
    - Twitter API project

Goals

- Using data scraped from their API, we wanted to determine the biggest factor for our prediction

- Find the best model for our case

- Predicting the number of likes on the next tweet relating to COVID-19

Data Collection

Using Twitter API

Applied for an API Student Account

- Reasoning for access to Twitter API

- Generated Tokens and Keys

- Copy and paste 4 different tokens / secret keys in our code for security

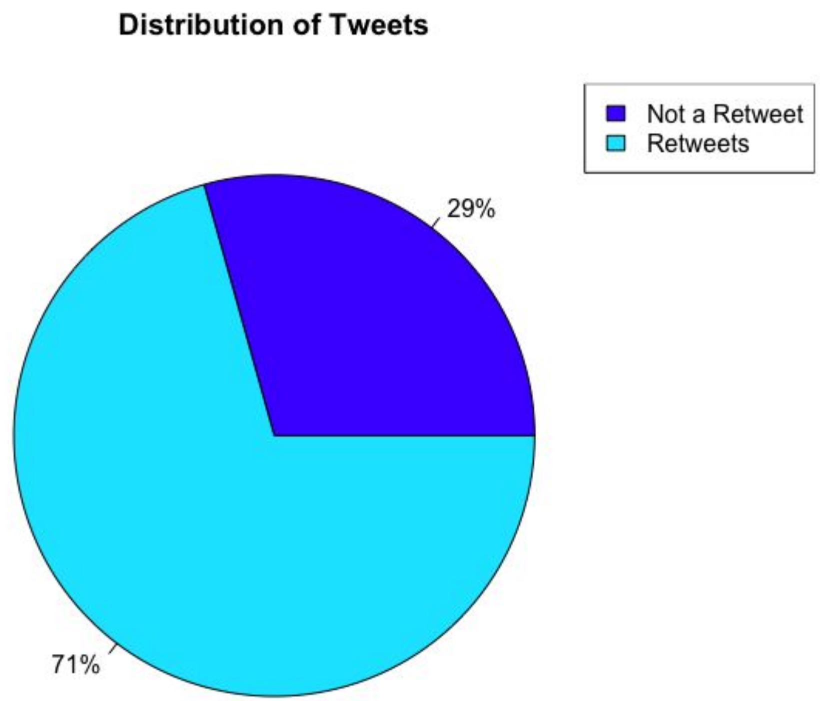
Using Python, generated a scraper file to scrape the necessary data and used Pandas Dataframe to convert data collected to CSV

Preprocessing Data

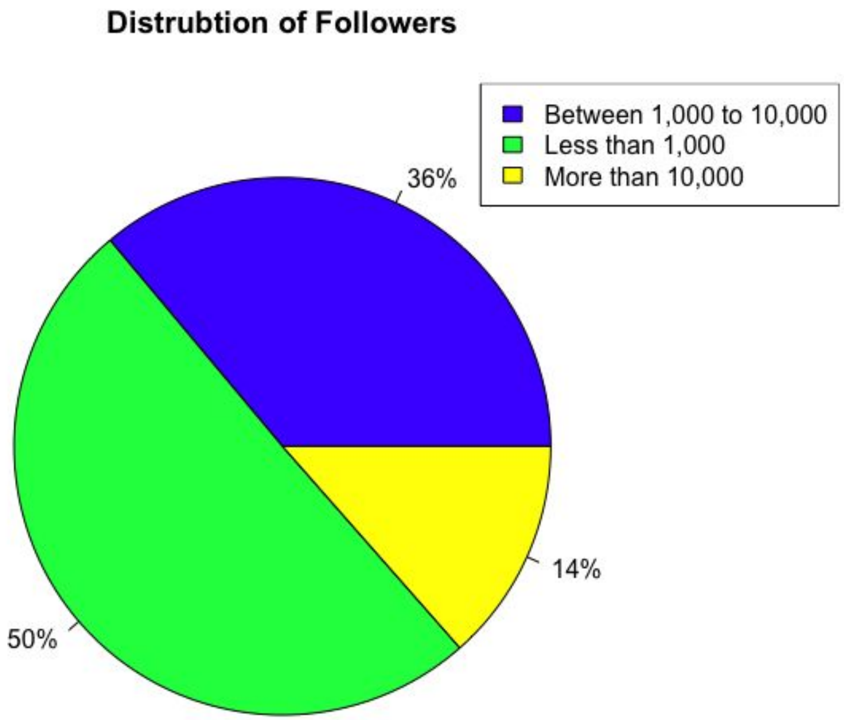
- Cleaned our dataset
  - Looked for null values
- Converted isRetweet to 1 or 0 instead of True and False
- Normalized the columns used for the modeling (evenly weighted)
- Split data into training and testing
  - 70% training
  - 30% testing

Dataset Analysis

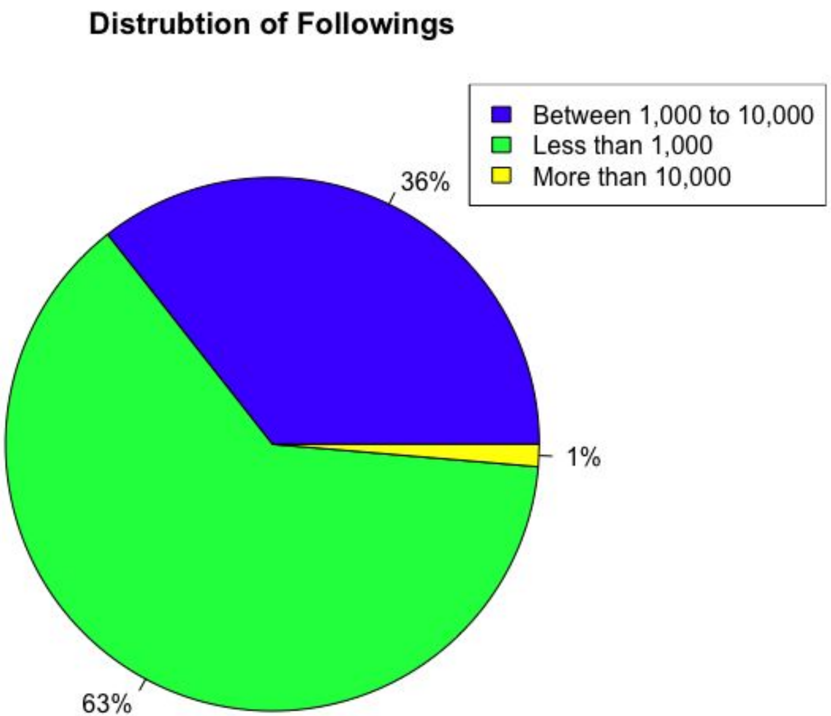
Distribution of Tweets



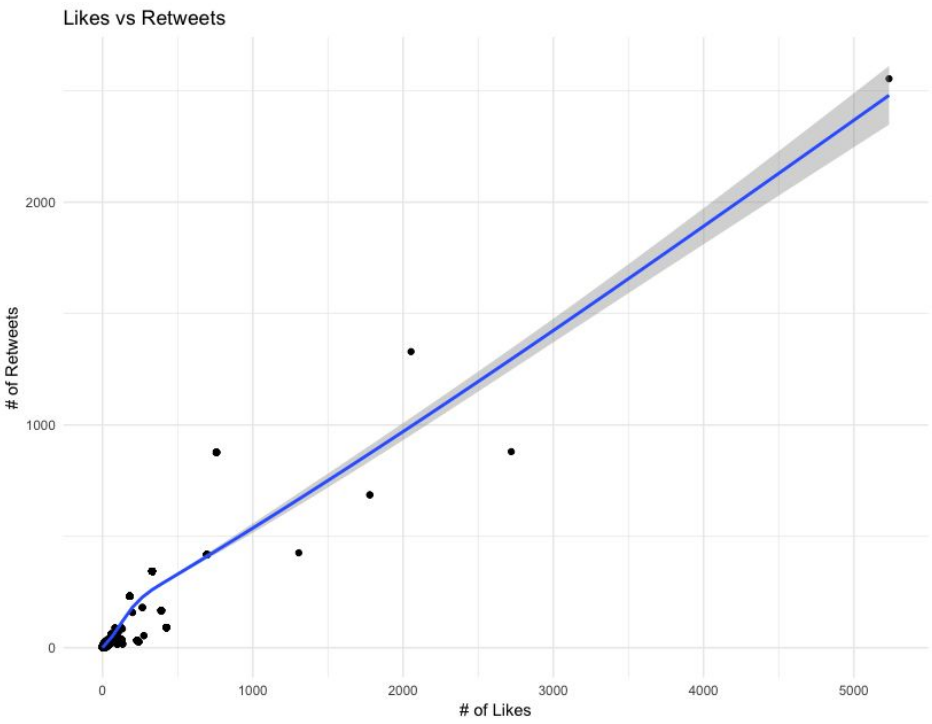
Distribution of Followers



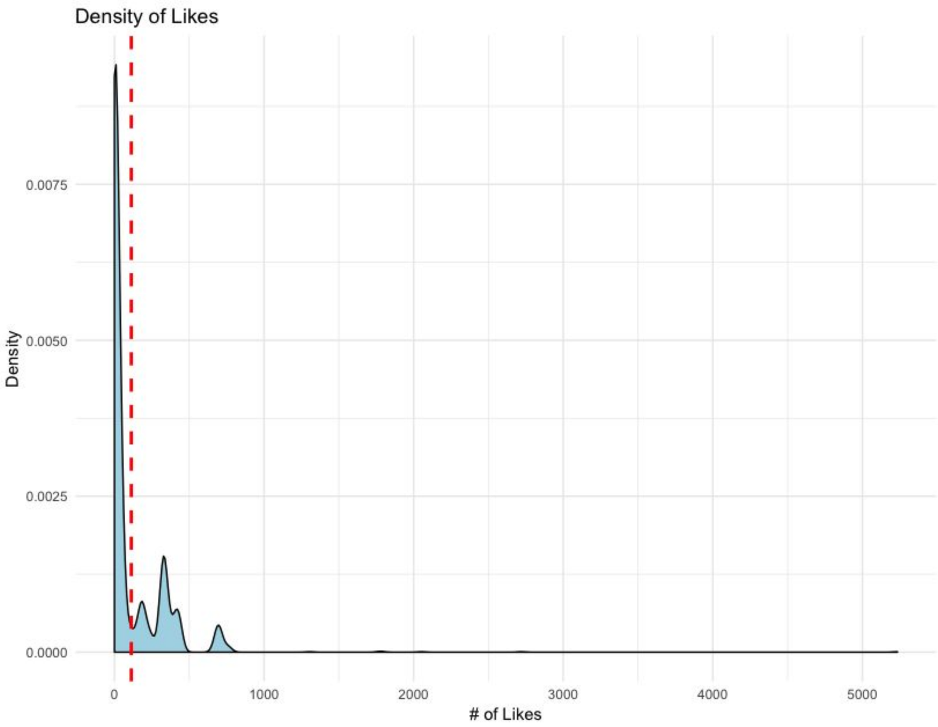
Distribution of Following



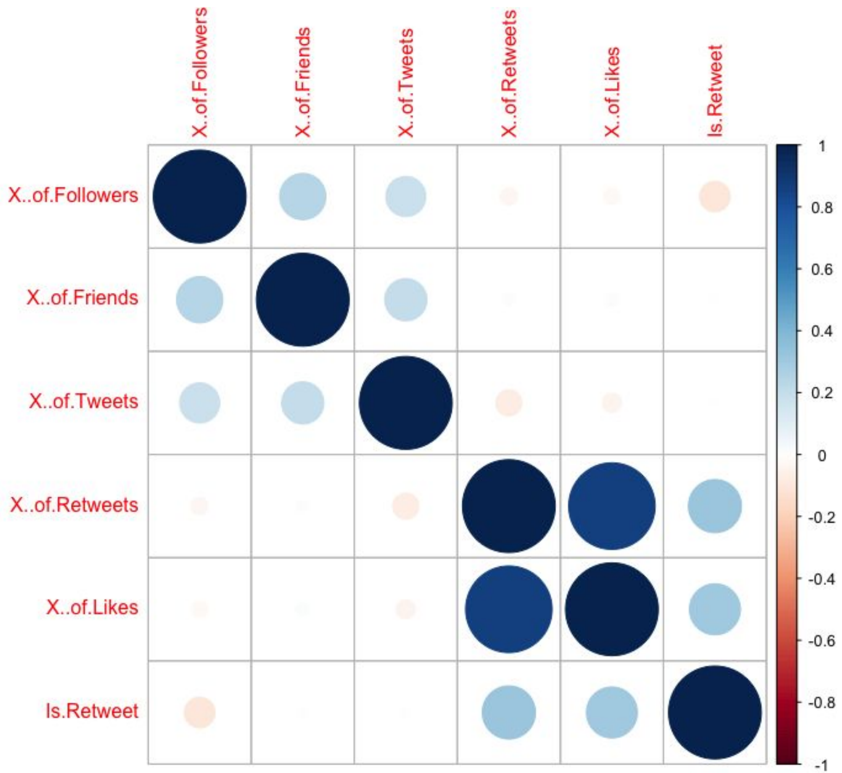
of Likes vs # of Retweets



Density of Likes



Correlation of Data



Modeling

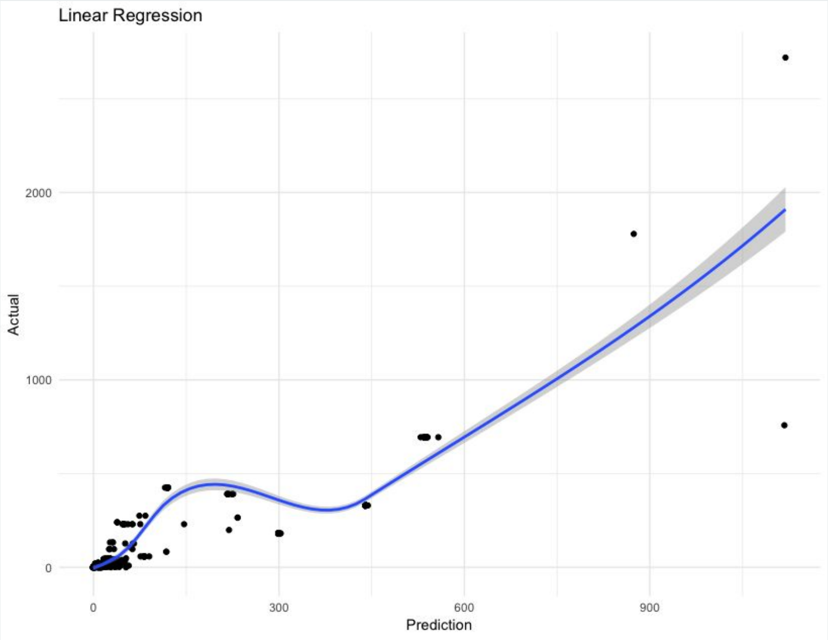
6 Models Used (Regressions)

1. Linear Regression
2. KNN
3. CART
4. Random Forest
5. Naive Bayes
6. SVM

Used R^2 Score to determine how close the data is fitted to the regression line

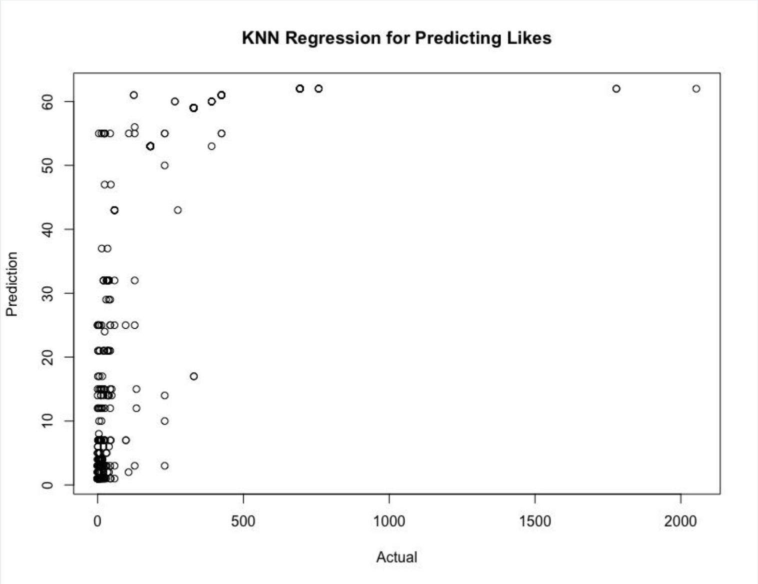
Used Standard Error to determine the distance that observed values fall from the regression line

Linear Regression



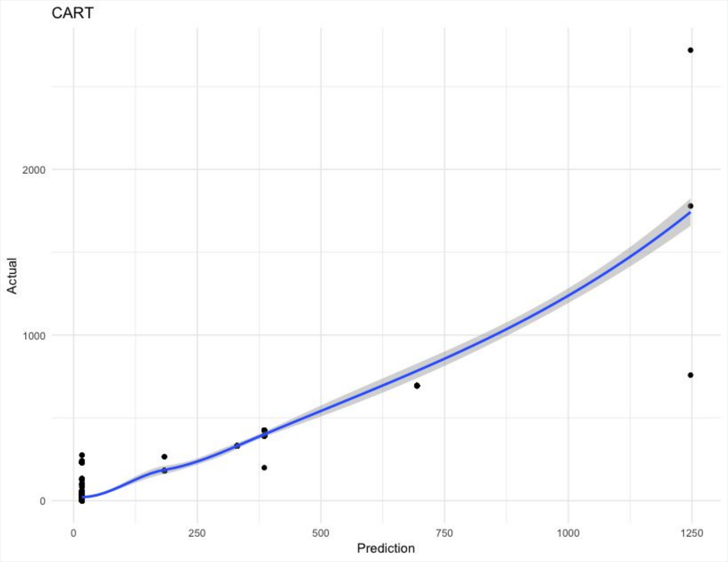

```
# Linear Regression
linearRegression <- lm(y_train~., data = x_train)
linearRegressionPredict <- predict(linearRegression,
x_test, type="response")
ggplot(NULL,aes(x=linearRegressionPredict,
y=y_test)) + geom_point() +
  geom_smooth() +
  theme_minimal() +
  labs(title="Linear Regression", x="Prediction",
y='Actual')
linearRegressionSummary <- summary(linearRegression)
linearRegressionR2 <-
linearRegressionSummary$r.squared
linearRegressionStandardError <-
linearRegressionSummary$sigma
```

KNN



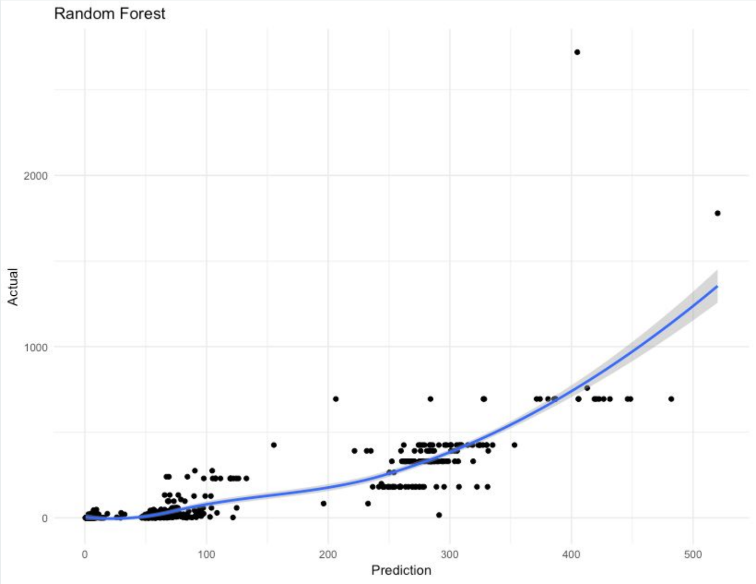

```
# KNN
likesPredictionKNN <- knn(train=x_train,
test=x_test, cl=y_train, k=15)
plot(y_test, likesPredictionKNN, xlab='Actual',
ylab='Prediction', main='KNN Regression for
Predicting Likes')
knnSummary <- summary(lm(y_test~likesPredictionKNN))
knnR2 <- knnSummary$r.squared
knnStandardError <- knnSummary$sigma
```

CART



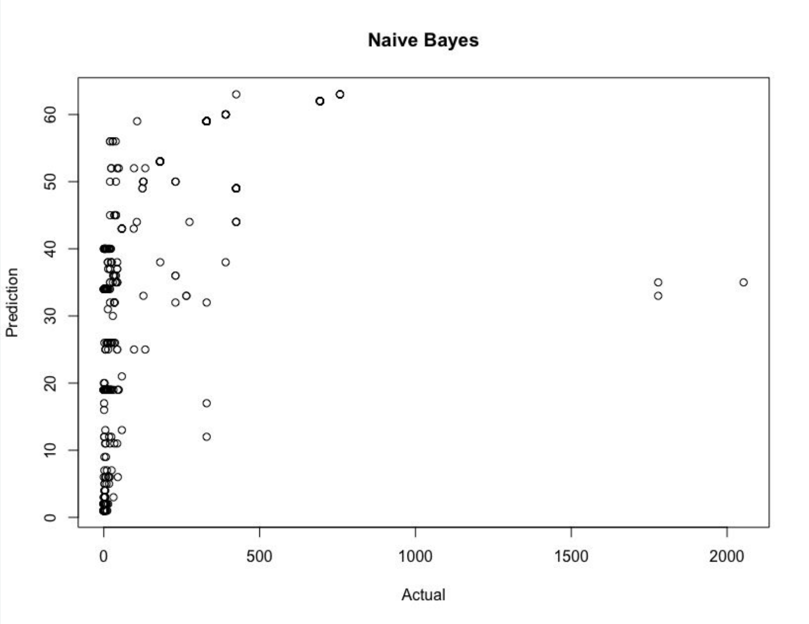

```
# CART
cartModel <- rpart(y_train~., data=x_train,
method="anova")
rpart.plot(cartModel)
cartPredict <- predict(cartModel, x_test)
cartSummary <- summary(lm(y_test~cartPredict))
cartR2 <- cartSummary$r.squared
cartStandardError <- cartSummary$sigma
```

Random Forest



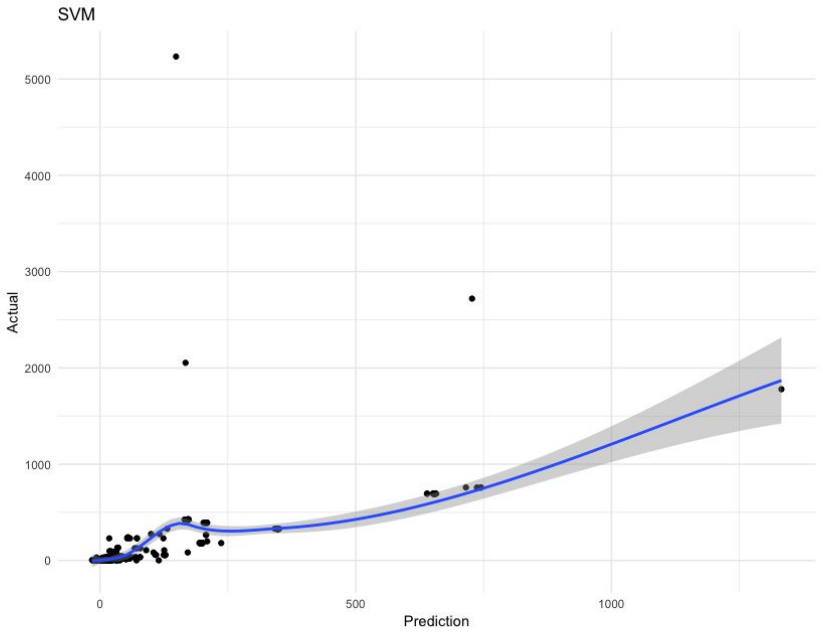

```
# Random Forest
randomForestFit <- randomForest(y_train~.,
data=data.frame(x_train))
importance(randomForestFit)
varImpPlot(randomForestFit)
randomForestPrediction <- predict(randomForestFit,
x_test)
ggplot(NULL,aes(x=randomForestPrediction, y=y_test)) +
geom_point() +
 geom_smooth() +
 theme_minimal() +
 labs(title="Random Forest", x="Prediction", y='Actual')
randomForestSummary <-
summary(lm(y_test~randomForestPrediction))
randomForestR2 <- randomForestSummary$r.squared
randomForestStandardError <- randomForestSummary$sigma
```

Naive Bayes




```
# Naive Bayes

naiveBayesModel <- naiveBayes(y_train~., x_train)

naiveBayesPrediction <- predict(naiveBayesModel,
x_test)

plot(y_test, naiveBayesPrediction, xlab='Actual',
ylab='Prediction', main='Naive Bayes')

naiveBayesSummary <-
summary(lm(y_test~naiveBayesPrediction))

naiveBayesR2 <- naiveBayesSummary$r.squared

naiveBayesStandardError <- naiveBayesSummary$sigma
```

SVM



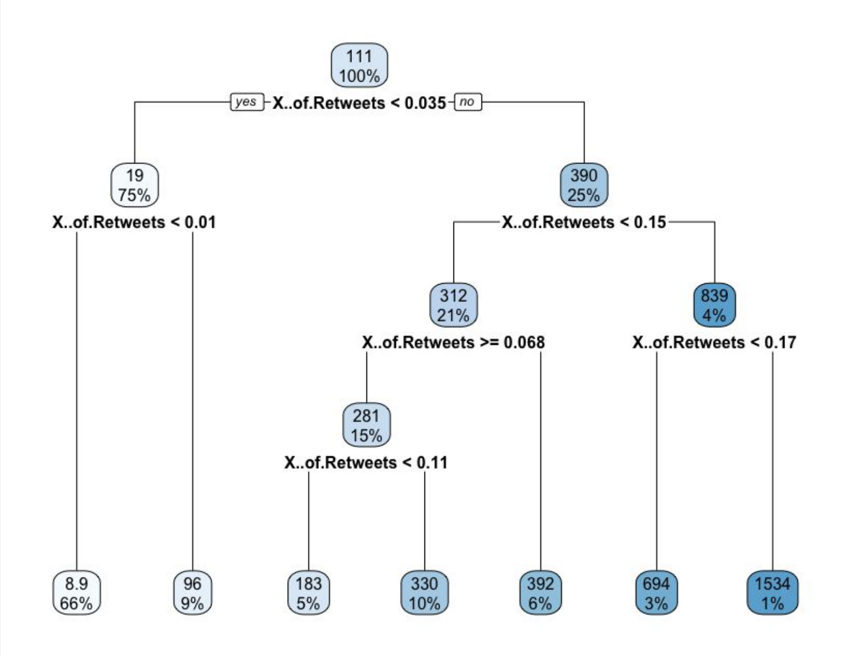

```
# SVM

svmModel <- svm(y_train~., data=x_train)

svmPrediction <- predict(svmModel,x_test)

ggplot(NULL,aes(x=svmPrediction, y=y_test)) + geom_point() +

  geom_smooth() +

  theme_minimal() +

  labs(title="SVM", x="Prediction", y='Actual')

svmSummary <- summary(lm(y_test~svmPrediction))

svmR2 <- svmSummary$r.squared

svmStandardError <- svmSummary$sigma
```

Comparison of the Models

Model	R^2 Score	Standard Error
Linear Regression	0.7829257	108.81694
KNN	0.8409609	89.38537
CART	0.8714464	78.11426
Random Forest	0.6850138	122.27393
Naive Bayes	0.7582547	110.89211
SVM	0.7269860	113.83624

Conclusion

Model



- CART was the best model
- Prediction: retweets was going to be the biggest factor in our model
- Conclusion: confirmed that the number of retweets was the best factor to determine the number of likes on a COVID post

Next Steps

- Using Twitter's Enterprise API
    - impression count (# views)
    - pull more data at a time (limit 3000)
    - filter tweets / retweets when scraping rather than sorting through in RStudio
    - gather information from start of covid
- Gather more than 2000 rows of data, having a bigger dataset, we could have better model to predict more accurate results

Thank You! Any Questions?