METROBIKE AUSTIN TRIPS

STA380 Introduction to Machine Learning – Project

Chyavan Mysore Chandrashekar	CM65624
Pratik Gawli	PBG397
Rukh Agha	MSA3453
Spoorthi Anupuru	SA56643
Tanushree Devi Balaji	TB33857

Agenda

DATA DESCRIPTION

DATA CLEANING

REGRESSION

K – NEAREST NEIGHBORS

BAGGING RANDOM FOREST

BOOSTING

INSIGHTS AND RECOMMENDATION

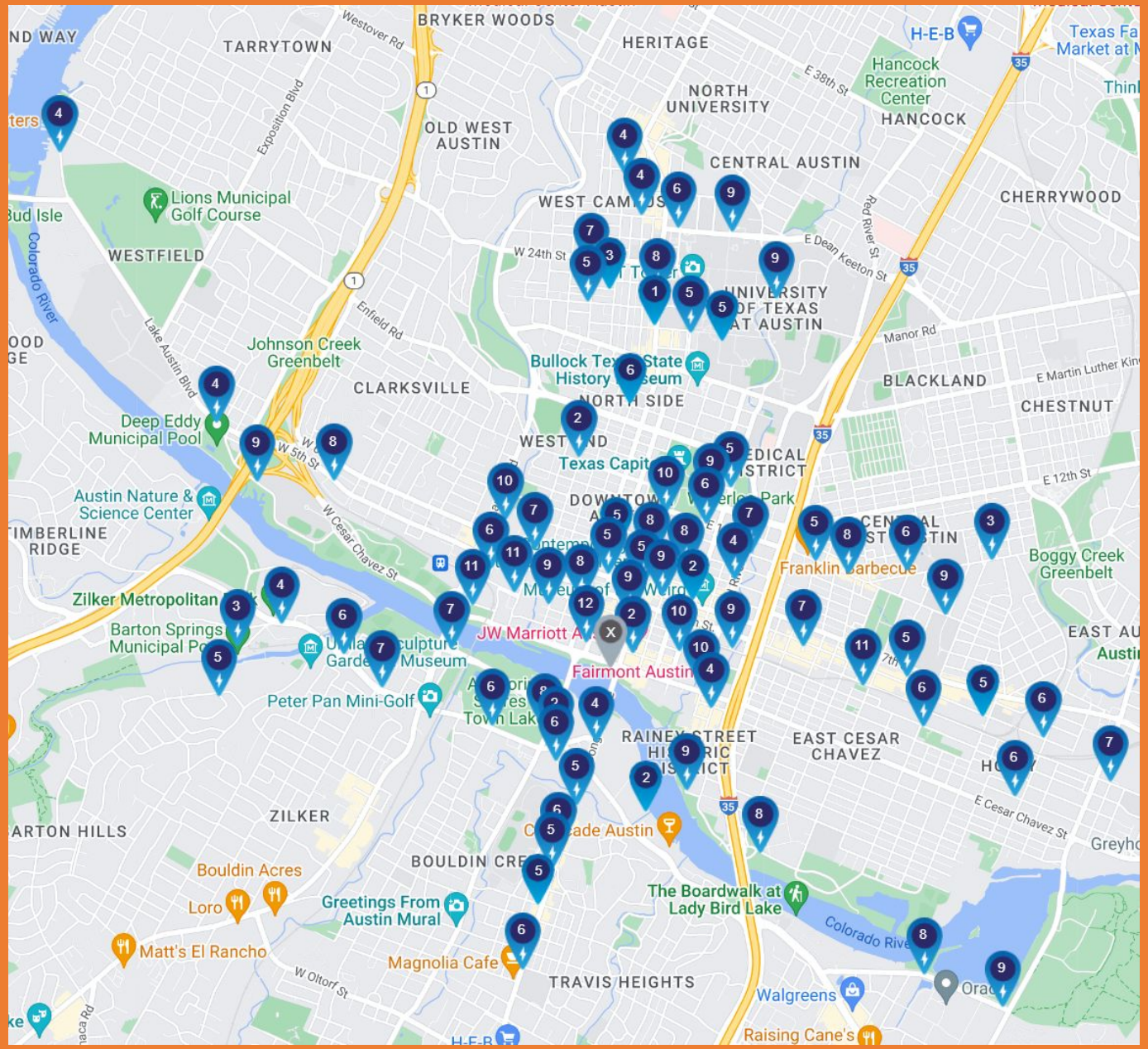
Data and Attributes

Data Source – Metrobike Austin: Trips

Time frame – 2013 to 2022

Size – 1.6M Rows

https://data.austintexas.gov/Transportation-and-Mobility/Austin-MetroBike-Trips/tyfh-5r8s



https://austin.bcycle.com/stations

Column Name	Description	Type
Trip ID	The MetroBike trip's unique ID.	Number #
Membership Type	The membership type of the MetroBike user. More info at :...	Plain Text T
Bicycle ID	The MetroBike's ID that was rented during the trip	Plain Text T
Bike Type	The type of bicycle used for the trip	Plain Text T
Checkout Date	The date the bike was checked out from the kiosk.	Date & Time
Checkout Time	The time the bike was checked out for the trip.	Plain Text T
Checkout Kiosk ID	The Kiosk ID of the Kiosk that was used to checkout the bic...	Plain Text T
Checkout Kiosk	The location of the checkout Kiosk based on its nearby cros...	Plain Text T
Return Kiosk ID	The ID of the Kiosk that was used to return the bicycle.	Plain Text T
Return Kiosk	The location of the Kiosk to return the bicycle. Based on th...	Plain Text T
Trip Duration Minutes	The total number of minutes that the bicycle was checked ...	Number #
Month	The calendar month on which the trip began	Plain Text T
Year	The year in which the trip began	Number #

What columns will we be considering?

Length of Trip

Location of the Checkout Kiosk

Average Monthly Temperature

Time of Year

Membership Type

OBJECTIVE

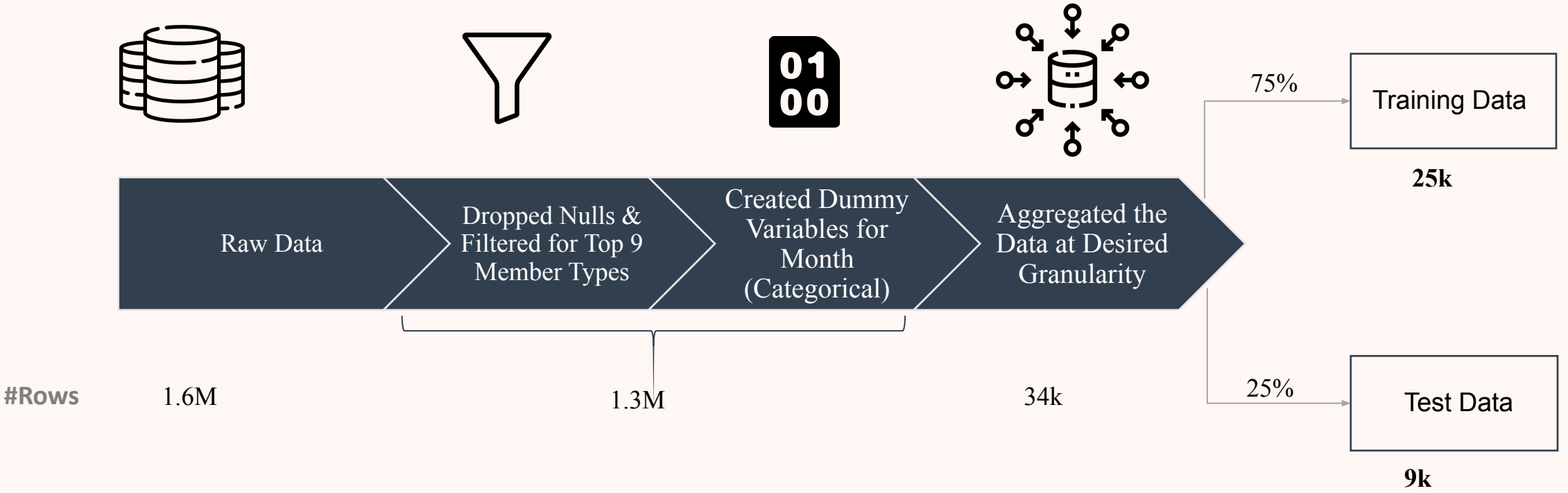
Predict volume of trips at a station in a given month based on:

- Station Location
- Avg Length of Trips at Station
- Customer Type
- Temperature
- The Month Itself!

And to offer potential recommendations for:

- Segment Marketing
- Supply Logistics

Data Cleaning

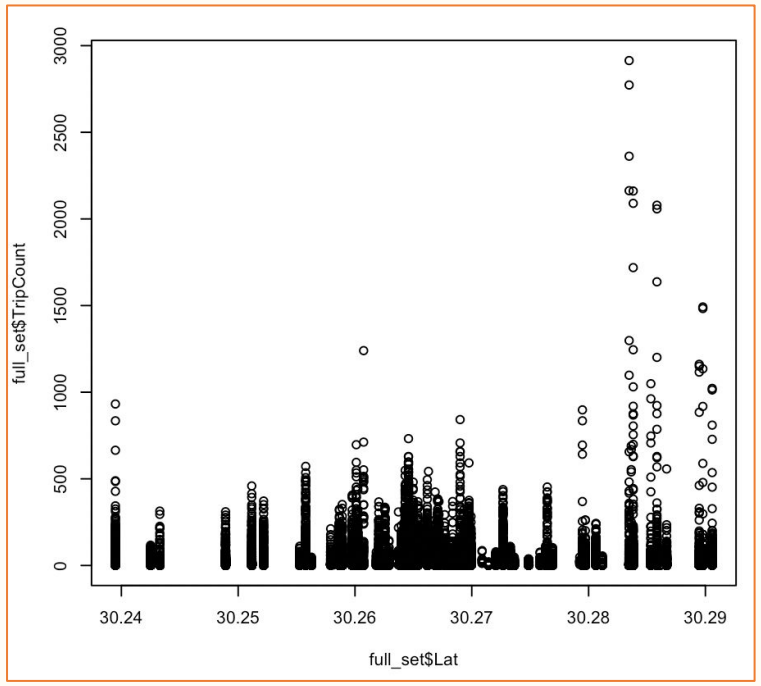


Regression

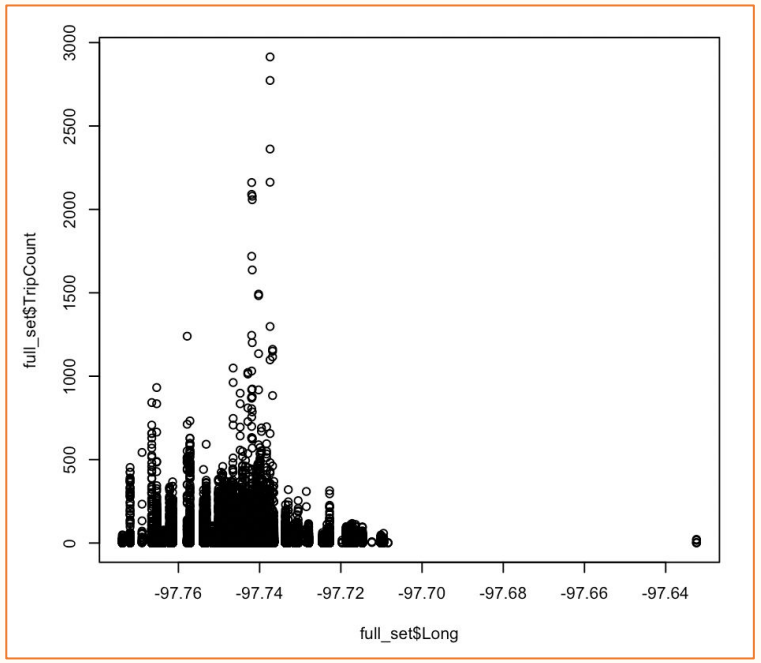
Predictor Trends

The first step was to understand the relationship between trip volume and the predictors. Based on the plots below, fair to say that there are no clear-cut linear relationships

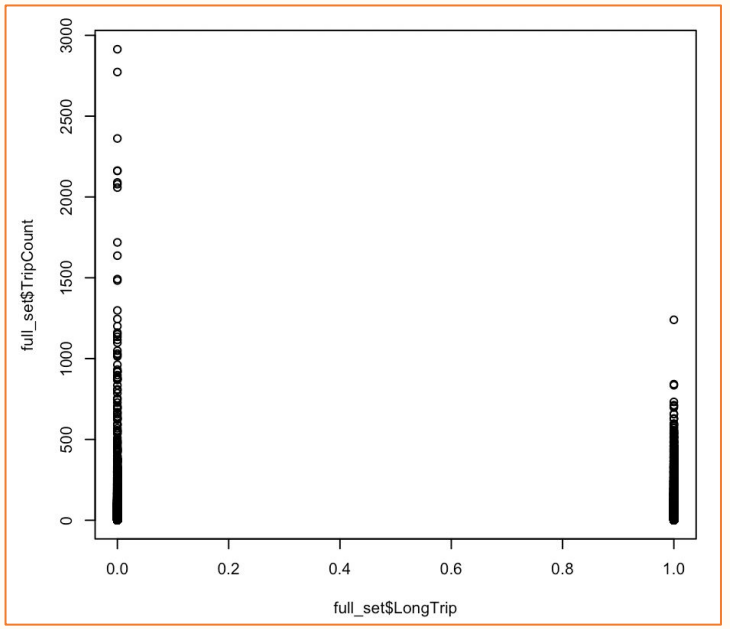
Latitude



Longitude

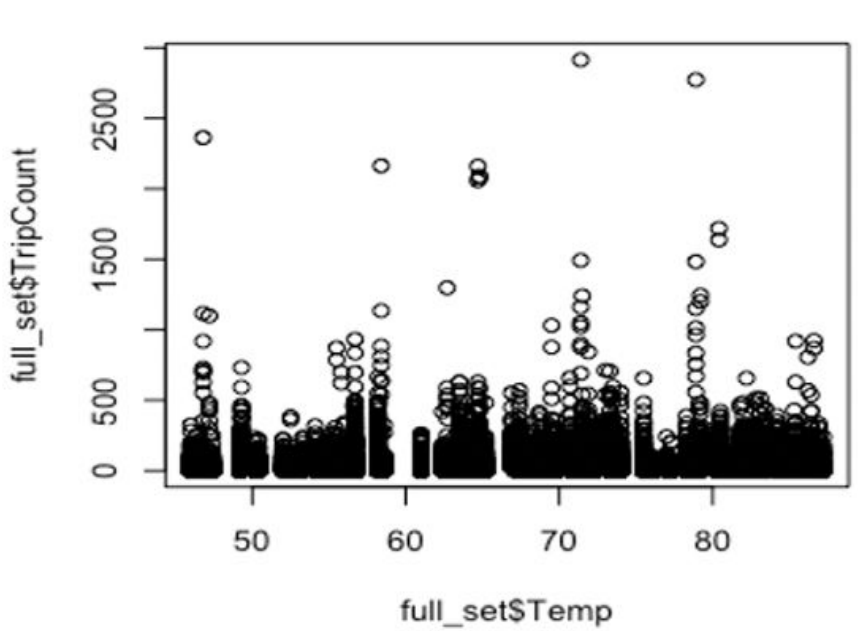


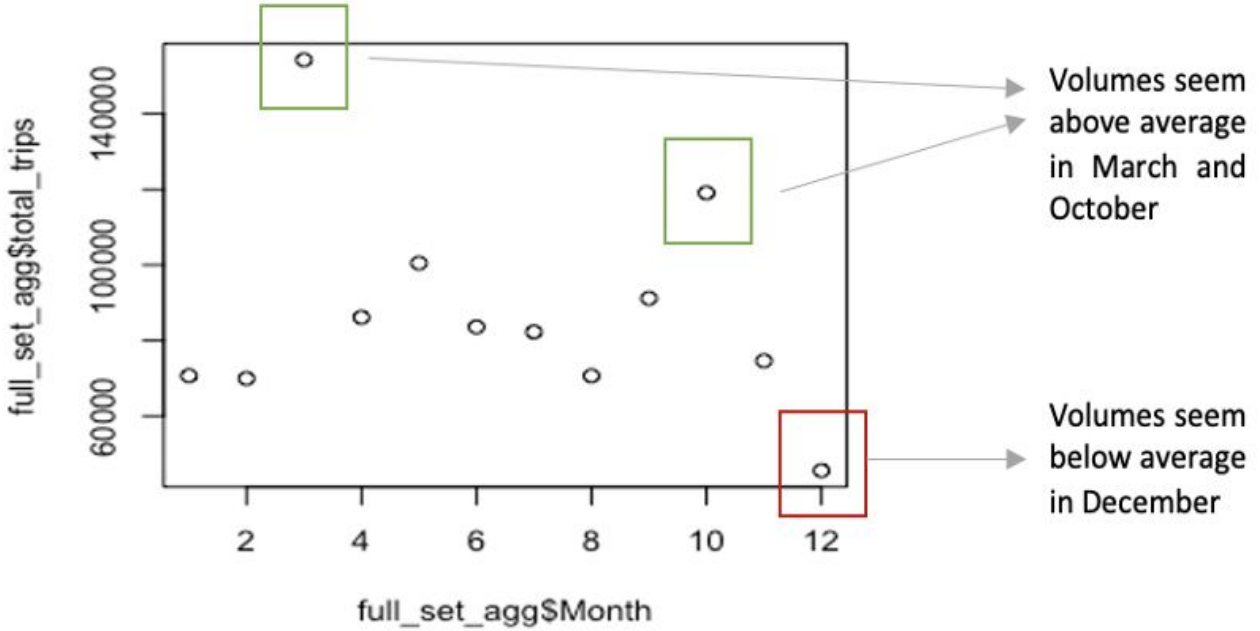
is_Long_Trip



Predictor Trends (Contd.)

The first step was to understand the relationship between trip volume and the predictors. Based on the plots below, fair to say that there are no clear-cut linear relationships





Multiple Linear Regression

Despite the lack of a visible linear relationship, multiple linear regression felt like a good starting point to get a sense of the data

```
Call:
lm(formula = numtrips ~ ., data = XXmulreg)

Residuals:
   Min     1Q Median     3Q    Max
-853.0 -217.2  -84.0  112.3 3839.4

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  366.026      8.688  42.132  < 2e-16 ***
LongTrip      -3.024      8.690  -0.348 0.727882
Lat          -20.948      8.793  -2.382 0.017289 *
Long         -90.082      8.800 -10.237  < 2e-16 ***
Month_Jan     29.021     11.941   2.430 0.015164 *
Month_Feb     29.733     12.024   2.473 0.013483 *
Month_Mar    127.796     11.878  10.759  < 2e-16 ***
Month_Apr      7.583     13.973   0.543 0.587391
Month_May     60.551     15.005   4.035 5.64e-05 ***
Month_Jun     41.635     15.034   2.769 0.005664 **
Month_Jul     44.297     14.919   2.969 0.003019 **
Month_Aug     29.544     14.947   1.977 0.048225 *
Month_Sep     52.093     14.976   3.479 0.000514 ***
Month_Oct     23.923     15.007   1.594 0.111072
Month_Nov     35.096     11.815   2.970 0.003007 **
Month_Dec         NA         NA      NA       NA
is_hot_day     7.324     17.443   0.420 0.674622
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 402.4 on 2129 degrees of freedom
Multiple R-squared:  0.1196,    Adjusted R-squared:  0.1134
F-statistic: 19.28 on 15 and 2129 DF,  p-value: < 2.2e-16
```

At first look, you notice:

1. Low R square
2. Most of these variables are insignificant and have low t-values

Multiple Linear Regression (Contd.)

Fine tuning this model, the results are as shown below:

```
Call:
lm(formula = numtrips ~ Long + Month_Mar + Month_May + Month_Dec,
    data = XXmulreg)

Residuals:
   Min     1Q Median     3Q    Max
-828.4 -229.6  -88.3  123.4 3904.0

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  366.026      8.742  41.871  < 2e-16 ***
Long         -86.674      8.744  -9.913  < 2e-16 ***
Month_Mar     96.361      8.806  10.943  < 2e-16 ***
Month_May     33.097      8.806   3.759 0.000175 ***
Month_Dec    -30.985      8.805  -3.519 0.000442 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 404.9 on 2140 degrees of freedom
Multiple R-squared:  0.104,     Adjusted R-squared:  0.1023
F-statistic: 62.07 on 4 and 2140 DF,  p-value: < 2.2e-16
```

With this model, you notice:

1. Trade off between complexity and R Square
2. Variables dictating trip duration and temperature are gone

Maybe, these variables are not meaningful as stand-alone predictors

Forward Regression

While individual variables may not be able to explain the variability of trip volume on their own, combinations of them might. Below are the results of the forward step regression model:

```
Call:
lm(formula = numtrips ~ Month_Mar + Long + LongTrip.Lat + Lat.Month_Mar +
    Lat.Long.Month_Mar + Lat.Long + is_hot_day + LongTrip.Long +
    LongTrip.Lat.Long + Month_Apr + Lat.Month_Sep + Month_Dec +
    Month_Oct + Lat + Lat.is_hot_day + Month_May, data = dfdata)

Residuals:
    Min      1Q  Median      3Q     Max
-1420.5  -218.0   -70.4   109.0  3543.5

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       373.990      8.360  44.734  < 2e-16 ***
Month_Mar         105.485      9.049  11.656  < 2e-16 ***
Long              -64.997      9.450  -6.878 7.96e-12 ***
LongTrip.Lat      -69.950      8.391  -8.336  < 2e-16 ***
Lat.Month_Mar     -64.014      8.678  -7.376 2.32e-13 ***
Lat.Long.Month_Mar  51.316      7.926   6.475 1.18e-10 ***
Lat.Long            48.586      8.791   5.527 3.66e-08 ***
is_hot_day          22.127      9.452   2.341 0.019327 *
LongTrip.Long      -56.476      9.417  -5.997 2.35e-09 ***
LongTrip.Lat.Long  -37.848      8.674  -4.363 1.34e-05 ***
Month_Apr          -37.428      8.651  -4.326 1.59e-05 ***
Lat.Month_Sep       34.829      8.607   4.047 5.38e-05 ***
Month_Dec          -33.211      8.931  -3.719 0.000206 ***
Month_Oct          -23.905      8.615  -2.775 0.005571 **
Lat                -25.675      8.395  -3.058 0.002253 **
Lat.is_hot_day     -24.175      8.975  -2.694 0.007121 **
Month_May           19.923      8.760   2.274 0.023046 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 382.3 on 2128 degrees of freedom
Multiple R-squared:  0.2055,    Adjusted R-squared:  0.1995
F-statistic: 34.41 on 16 and 2128 DF,  p-value: < 2.2e-16
```

Information about length of the trip and average temperature become more meaningful estimators of trip volume when combined with geographical information

There is also almost 100% improvement in R square

Forward Regression (Contd.)

Validating this same model using test data, the R square takes a considerable hit.

```
Call:
lm(formula = numtrips_test ~ Month_Mar + Long + LongTrip.Lat +
    Lat.Month_Mar + Lat.Long.Month_Mar + Lat.Long + is_hot_day +
    LongTrip.Long + LongTrip.Lat.Long + Month_Apr + Lat.Month_Sep +
    Month_Dec + Month_Oct + Lat + Lat.is_hot_day + Month_May,
    data = dfdata_test)

Residuals:
    Min      1Q  Median      3Q     Max
-551.54  -96.28  -39.27   35.11 2143.53

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)        137.329      4.233  32.441  < 2e-16 ***
Month_Mar           46.999      4.598  10.221  < 2e-16 ***
Long               -24.562      4.810  -5.106 3.61e-07 ***
LongTrip.Lat       -25.495      4.254  -5.993 2.45e-09 ***
Lat.Month_Mar      -14.488      4.430  -3.271 0.001092 **
Lat.Long.Month_Mar  22.157      3.940   5.624 2.14e-08 ***
Lat.Long            14.976      4.442   3.371 0.000763 ***
is_hot_day           9.672      4.838   1.999 0.045717 *
LongTrip.Long      -19.647      4.796  -4.097 4.36e-05 ***
LongTrip.Lat.Long  -12.436      4.389  -2.833 0.004654 **
Month_Apr           -7.306      4.403  -1.659 0.097228 .
Lat.Month_Sep        7.460      4.355   1.713 0.086867 .
Month_Dec          -10.018      4.539  -2.207 0.027422 *
Month_Oct            1.336      4.362   0.306 0.759358
Lat                -10.937      4.257  -2.569 0.010275 *
Lat.is_hot_day      -5.939      4.560  -1.302 0.192919
Month_May           10.918      4.439   2.459 0.014006 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 184.8 on 1935 degrees of freedom
Multiple R-squared:  0.1306,    Adjusted R-squared:  0.1234
F-statistic: 18.16 on 16 and 1935 DF,  p-value: < 2.2e-16
```

A lot of the previously significant variable are not significant with the test set, indicating possible overfitting on the train set

***Note:** that backward and step-wise regressions yielded similar results*

Log Regression

It’s possible that the relationship maybe more exponential than linear. Below are the results of log regression:

```
Call:
lm(formula = numtrips_log ~ Long + Month_Mar + LongTrip.Lat +
    Month_Dec + Lat + Month_Apr + LongTrip.Lat.Long + LongTrip.Long +
    Lat.Long + Long.Month_Feb + is_hot_day + Month_Oct + Lat.Month_Mar +
    Lat.Long.Month_Mar + Lat.Month_May + Lat.Long.is_hot_day,
    data = dfdata_log)

Residuals:
    Min      1Q  Median      3Q     Max
-5.4457 -0.5843  0.2261  0.7901  2.9129

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         5.27687    0.02632 200.519  < 2e-16 ***
Long               -0.25431    0.03083  -8.248 2.78e-16 ***
Month_Mar           0.19465    0.02841   6.851 9.58e-12 ***
LongTrip.Lat       -0.17890    0.02634  -6.791 1.44e-11 ***
Month_Dec          -0.12199    0.02803  -4.352 1.41e-05 ***
Lat                -0.11635    0.02637  -4.412 1.07e-05 ***
Month_Apr          -0.11411    0.02698  -4.229 2.44e-05 ***
LongTrip.Lat.Long  -0.14847    0.02724  -5.451 5.58e-08 ***
LongTrip.Long      -0.14355    0.02957  -4.855 1.29e-06 ***
Lat.Long            0.12471    0.02845   4.384 1.22e-05 ***
Long.Month_Feb     -0.07254    0.02088  -3.473 0.000525 ***
is_hot_day          0.07683    0.02901   2.648 0.008155 **
Month_Oct          -0.06319    0.02675  -2.362 0.018269 *
Lat.Month_Mar      -0.06971    0.02632  -2.649 0.008140 **
Lat.Long.Month_Mar  0.07005    0.02576   2.719 0.006608 **
Lat.Month_May      -0.06038    0.02608  -2.315 0.020698 *
Lat.Long.is_hot_day 0.05281    0.02516   2.099 0.035926 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.2 on 2128 degrees of freedom
Multiple R-squared:  0.1593,    Adjusted R-squared:  0.1529
F-statistic: 25.19 on 16 and 2128 DF,  p-value: < 2.2e-16
```

R square takes a hit when estimating log of trip volume, without much change in the predictors

What's Next?

While linear models maybe easy to interpret, they don't explain the variability in this dataset quite that well.

Next steps could be to:

1. Test out other linear methods like Ridge and Lasso Regression
2. Work out non-parametric models to modify the shape of the curve.

This is what we'll do next and save lasso/ridge for future iterations

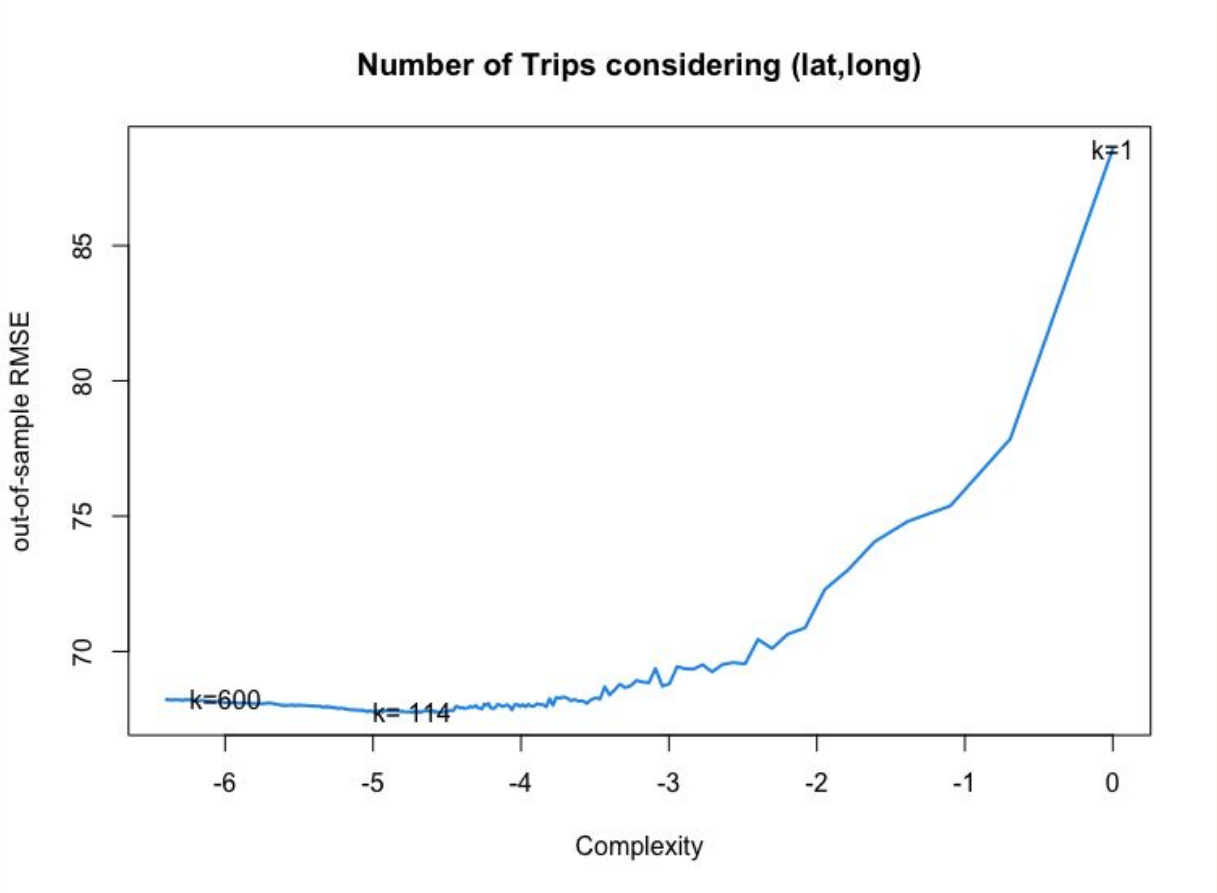
K Nearest Neighbors

KNN Models

Non parametric model to predict the output variable, requires numeric input variables

Model 1:

- Predicting the volume of trips based on Latitude and Longitude:
  - K fold cross validation used to calculate out of Sample RMSE(K=5); helps to average out the RMSE by testing on 5 different test sets coming from the whole data set.



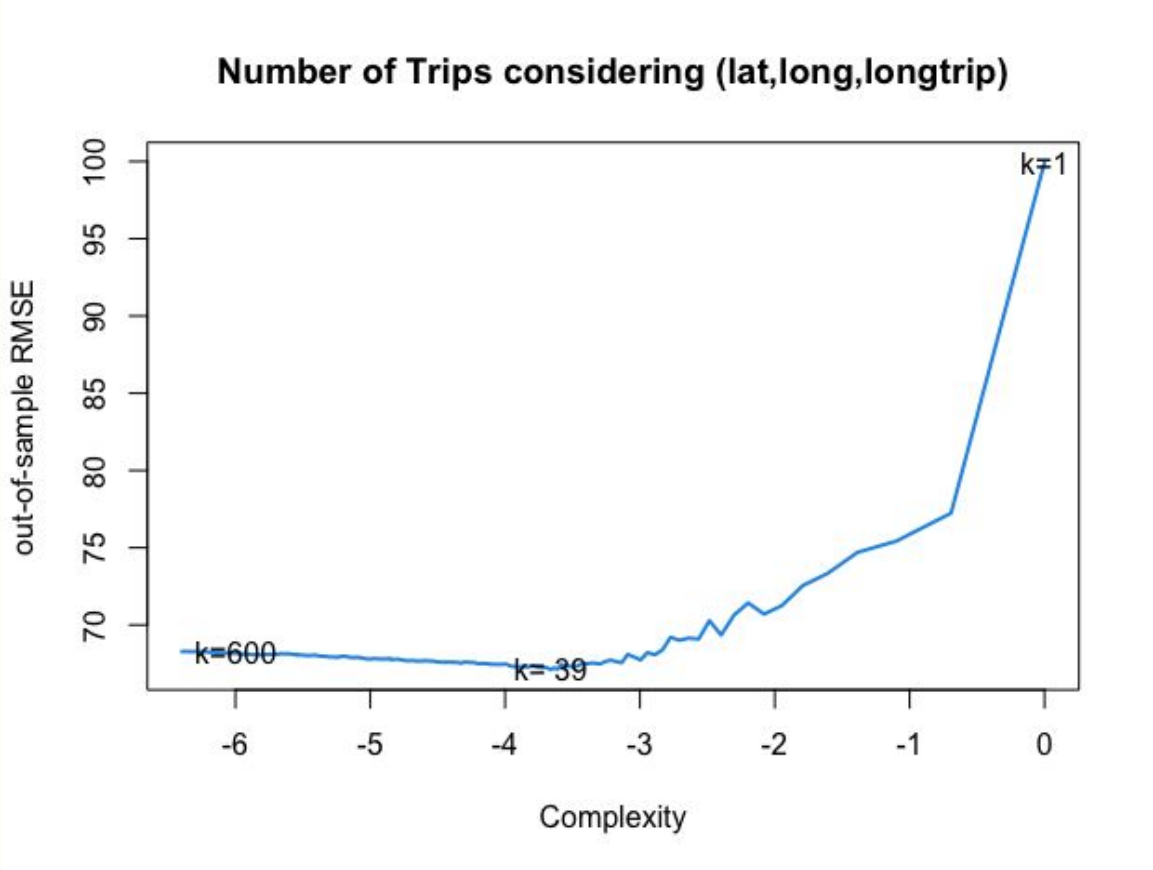
Minimum RMSE observed(Out of Sample)=67.7 for K= 114

KNN Models contd.

Model 2:

- Predicting the volume of trips based on Latitude, Longitude and LongTrip.
- Scaling of variables to a common scale required to calculate distances appropriately

```
AUData$LongTrip<-scale(AUData$LongTrip,sd(AUData$Long))
```



Minimum RMSE observed(Out of Sample)=**67.1** for K= 39

Improvement from providing only Latitude and Longitude!

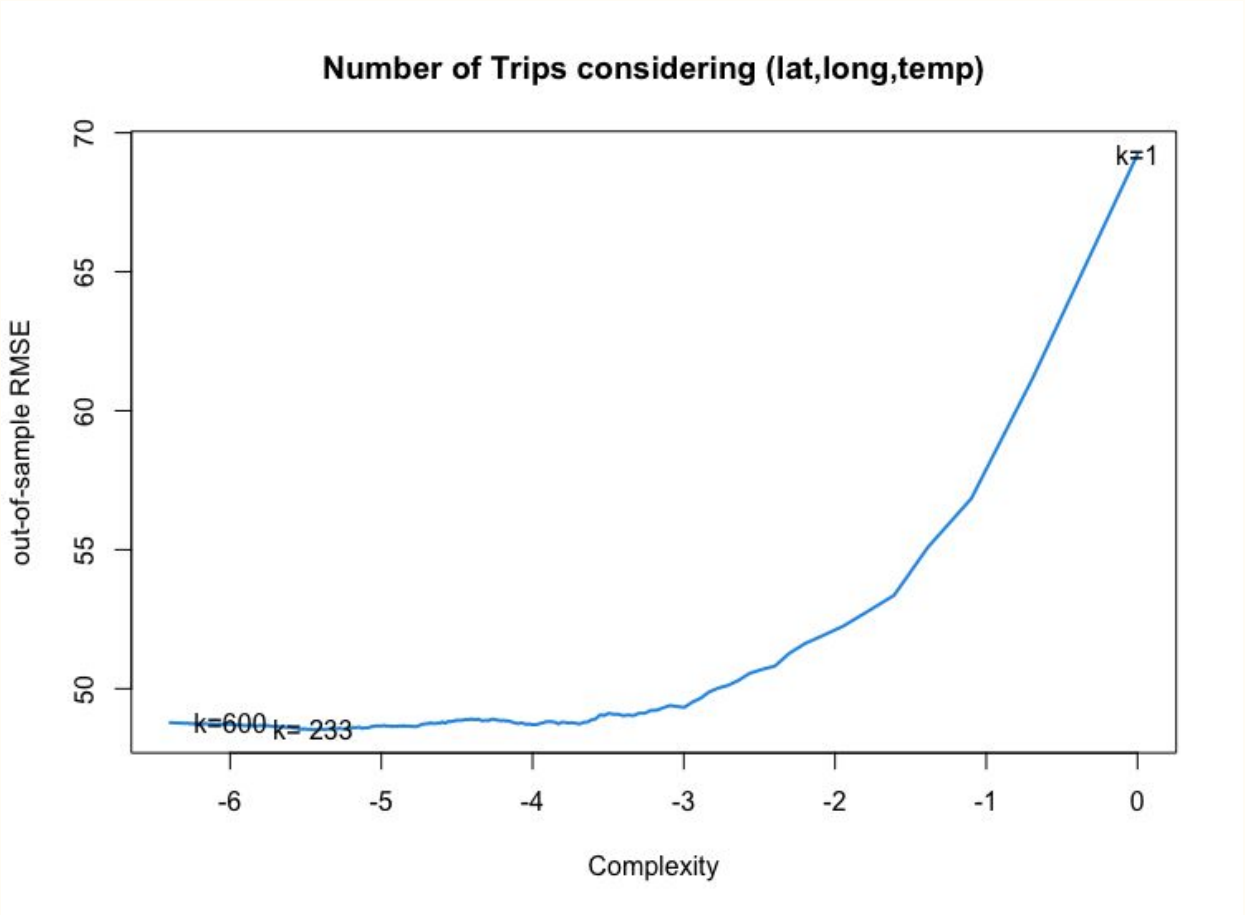
for eg:
Short trips from University to Apartment versus Kiosks only used for long trips

KNN Models contd.

Model 3:

- Predicting the volume of trips based on Latitude, Longitude and Temperature.
- Scaling of variables to a common scale required to calculate distances appropriately

```
AUData$Temp<-scale(AUData$Temp,sd(AUData$Long))
```



Minimum RMSE observed(Out of Sample)=**48.5** for K= 233

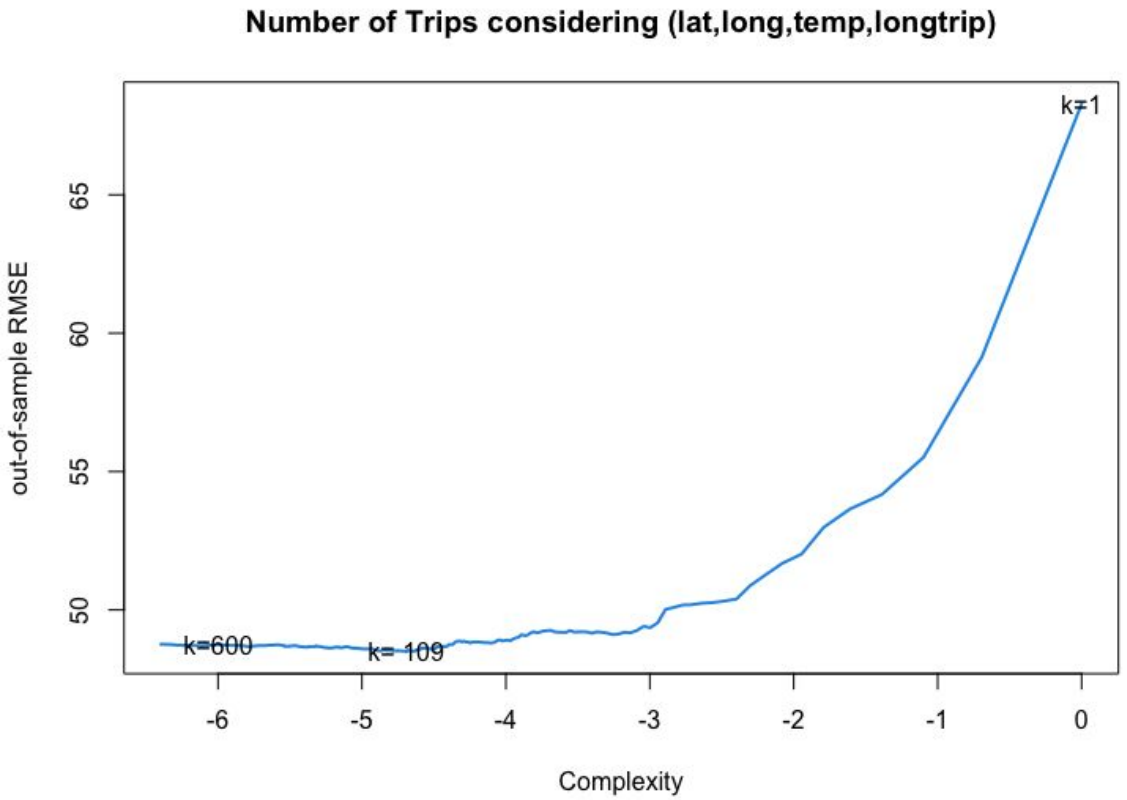
Improvement by providing Temperature as a predictor variable instead of LongTrip. - Makes sense as it provides more numerical information which helps predict the number of trips better, reducing the RMSE.

KNN Models contd.

Model 4:

- Predicting the volume of trips based on Latitude, Longitude, Temperature and LongTrip.
- Scaling of variables to a common scale required to calculate distances appropriately

```
AUData$Temp<-scale(AUData$Temp,sd(AUData$Long))
AUData$LongTrip<-scale(AUData$LongTrip,sd(AUData$Long))
```



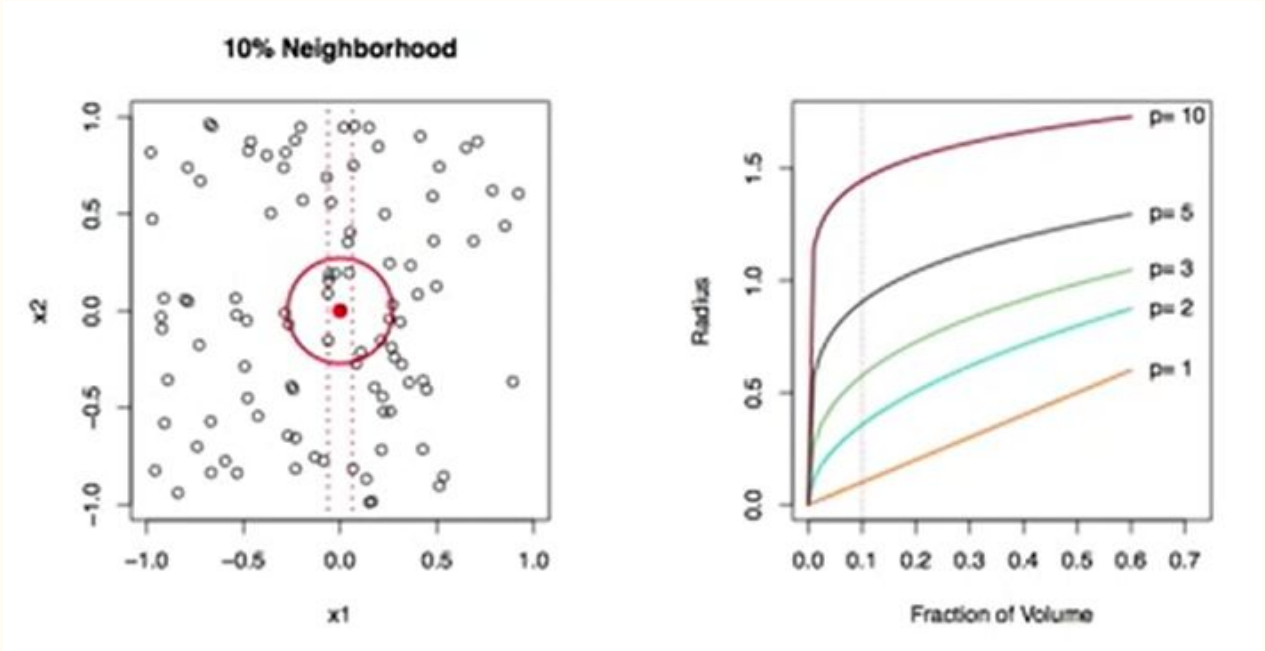
Minimum RMSE observed(Out of Sample)=**48.5** for K= 109

Performance less similar as to including only temperature along with Latitude and Longitude, just number of nearest neighbors required corresponding to minimum RMSE has decreased.

Expected behavior as temperature being highly influential in determining the trip length

KNN Models - Summary

- Best KNN out of Sample RMSE was with the value of 57.8 for K equals 106.
- KNN requires numeric type predictor values to calculate distances effectively.
- If Membership type and Month,Year need to be included in the predictor set, they need to be encoded as dummy variables.
- But performance degrades with increasing predictor variables - Curse of Dimensionality for KNN.



- Next up: Testing other non-parametric models to leverage the information of MembershipType and (Month, year) to improve prediction results by reducing RMSE.

Boosting

Boosting

In this model, we have the control over

1. Eta (Lambda) - The shrinkage factor
2. Maximum depth of the tree
3. Number of trees

To regress the **Y** (Volume of Trips) against **Xs** (Latitude, Longitude, Temperature, Trip Length*, Membership Type*, Month of the year*) we are using a package called **`XGBoost`**, which stands for Extreme Gradient Boosting - a gradient-boosted decision tree (GBDT) library.

```
model = xgboost(data= as.matrix(Xs_training),
                label= y_training,
                params= params,
                eta = row$Eta,
                max_depth = row$Max_d,
                nrounds= row$Num_trees,
                verbose= 0)
```

Boosting

In this model, we have the control over

1. Eta (Lambda) - The shrinkage factor
2. Maximum depth of the tree
3. Number of trees

Eta	Max_d	Num_trees	Training_RMSE	Testing_RMSE
0.01	2	10	79.443	76.124
0.01	4	200	61.077	57.968
0.01	10	1000	14.541	40.132
0.05	2	10	74.231	70.420
0.05	4	1000	36.269	40.790
0.05	10	1000	4.832	39.377
0.2	2	10	68.010	63.992
0.2	2	1000	54.663	53.945
0.2	6	10	49.783	51.283
0.2	6	200	21.187	33.216
0.2	6	1000	10.807	33.357
0.2	8	1000	3.798	36.905
0.2	10	1000	1.513	40.738
0.3	10	1000	1.424	40.803
0.4	4	1000	22.382	38.510
0.4	10	1000	1.413	38.708

Boosting

We obtained the minimum RMSE for the Validation data set for the values highlighted in green.

As we can see, the minimum training error can be minimized as we make the model more complex, but the minimum RMSE for the Validation data set for the values highlighted in green, which necessarily are not associated with the most complex model.

Eta	Max_d	Num_trees	Training_RMSE	Testing_RMSE
0.01	2	10	79.443	76.124
0.01	4	200	61.077	57.968
0.01	10	1000	14.541	40.132
0.05	2	10	74.231	70.420
0.05	4	1000	36.269	40.790
0.05	10	1000	4.832	39.377
0.2	2	10	68.010	63.992
0.2	2	1000	54.663	53.945
0.2	6	10	49.783	51.283
0.2	6	200	21.187	33.216
0.2	6	1000	10.807	33.357
0.2	8	1000	3.798	36.905
0.2	10	1000	1.513	40.738
0.3	10	1000	1.424	40.803
0.4	4	1000	22.382	38.510
0.4	10	1000	1.413	38.708

Boosting

Let’s go one more step further to get the best validation RMSE!

We considered minute changes about the “knobs” that gave us the best Validation RMSE in the first iteration.

Eta	Max_d	Num_trees	Training_RMSE	Testing_RMSE
0.12	4	120	45.278	46.215
0.12	5	120	37.146	41.676
0.12	6	120	30.081	37.415
0.12	7	200	19.681	32.244
0.12	7	240	18.015	32.095
0.12	7	280	16.992	32.252
0.12	8	240	13.279	36.043
0.16	7	280	15.295	35.654
0.2	6	200	21.187	33.216
0.2	8	280	9.173	36.714
0.24	4	120	40.407	43.865
0.24	8	280	7.897	33.071
0.28	8	280	7.382	37.799

We can keep refining the settings, but this is a significant minimum for the test RMSE.

Boosting

Now, let's see what we can interfere from the model with the best RMSE

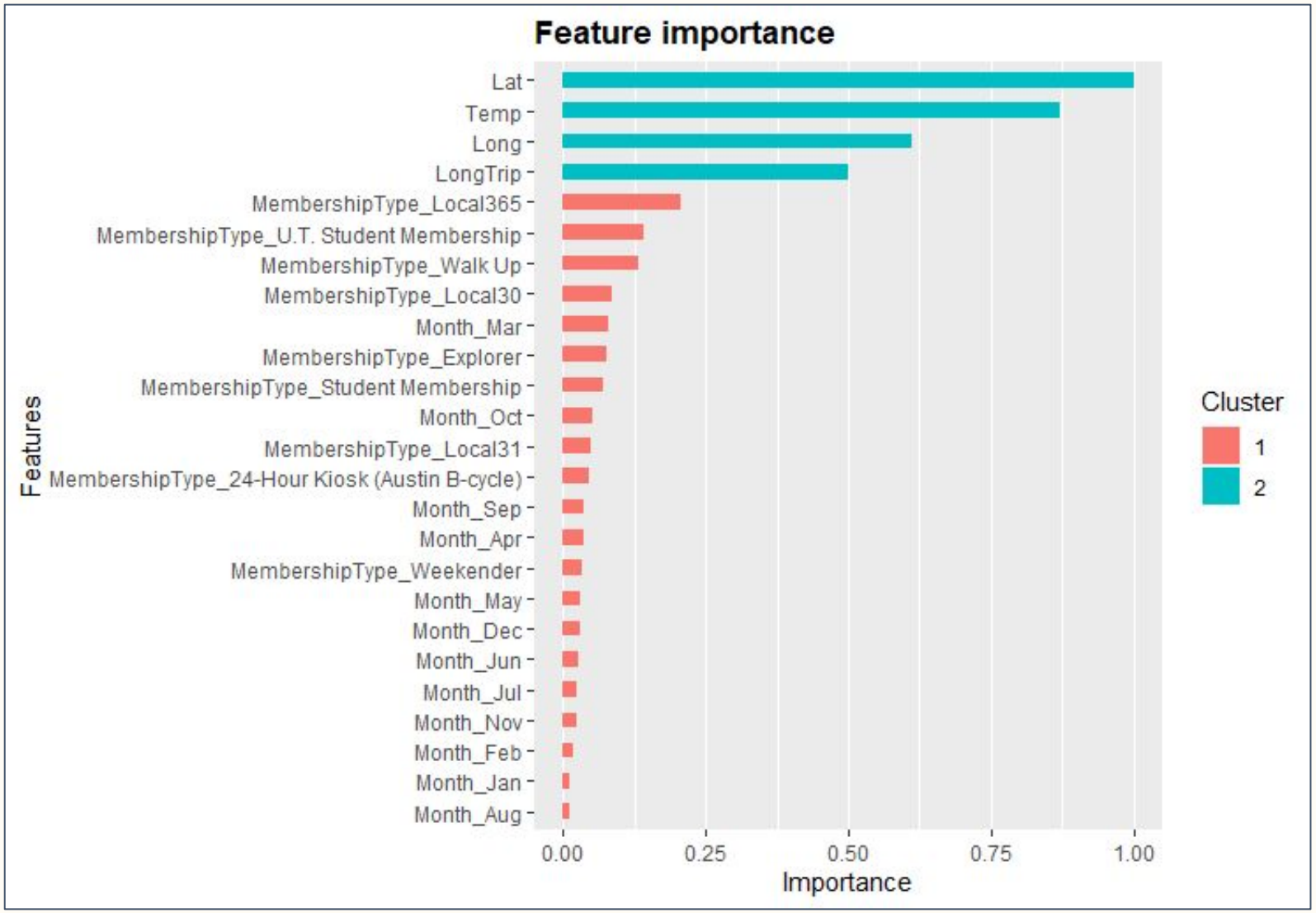
```
model = xgboost(data= as.matrix(Xs_training),
                label= y_training,
                params= params,
                eta = best_Eta,
                max_depth = best_Max_d,
                nrounds= best_Num_trees,
                verbose= 0)
```

We'll calculate the feature importance for each feature (input)

```
# Calculate feature importance of each of the X
feature_importance = xgb.importance(colnames(as.matrix(Xs_training)), model = model)

# Plot feature importance
(gg <- xgb.ggplot.importance(feature_importance, measure = "Frequency", rel_to_first = TRUE))
```

Boosting

What can we interpret from this Variable Importance Graph?



The top-4 variables are non-dummy variables,

1. Latitude
2. Temperature
3. Longitude
4. Long/Short Trip

Latitude has the most importance among all of them! This is clear if we consider the map of Austin!

Random Forests

Random Forests at (B = 50, B = 150, B = 300) and (M=6) yield different results

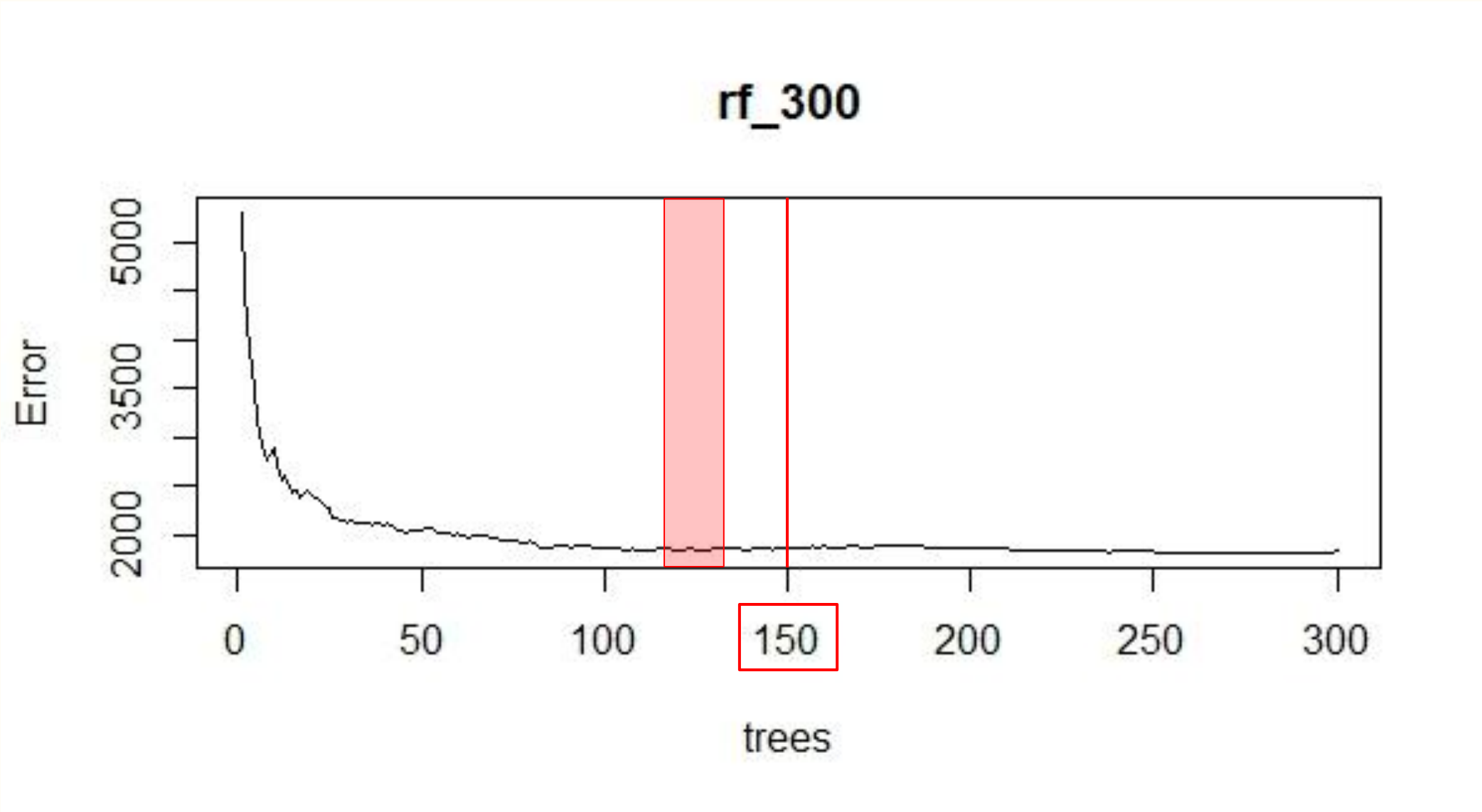
```
Call:
 randomForest(formula = TripCount ~ ., data = train, ntree = 50)
               Type of random forest: regression
                     Number of trees: 50
No. of variables tried at each split: 6

          Mean of squared residuals: 2128.242
                    % Var explained: 62.78
```

```
Call:
 randomForest(formula = TripCount ~ ., data = train, ntree = 150)
               Type of random forest: regression
                     Number of trees: 150
No. of variables tried at each split: 6

          Mean of squared residuals: 1982.937
                    % Var explained: 65.32
```

```
Call:
 randomForest(formula = TripCount ~ ., data = train, ntree = 300)
               Type of random forest: regression
                     Number of trees: 300
No. of variables tried at each split: 6

          Mean of squared residuals: 1827.339
                    % Var explained: 68.04
```

m = p/3 = 6	ntrees	RMSE	% Var Explained
6	**50**	**46.13**	**62.78%**
6	**150**	**44.53**	**65.32%**
6	**300**	**41.06**	**68.04%**

Random Forests begin to diminish returns a little over 100 trees



Cross-validating Random Forest (ntrees = 150) has reasonable results.

```
Call:
 randomForest(formula = TripCount ~ ., data = train, xtest = pred,
 ytest = predY, ntree = 150)
               Type of random forest: regression
                     Number of trees: 150
No. of variables tried at each split: 6

          Mean of squared residuals: 1845.093
                    % Var explained: 67.73
                       Test set MSE: 1685.52
                    % Var explained: 67.61
```

Validation RMSE = 41.06

Though the model is not optimal, the MSE in the test set is lower than the training set.
So OVERFITTING to noise is NOT LIKELY.
The Random Forests have some predictive value.

Bagging

Bagging at (B = 20, B = 50, B = 200) yields different results

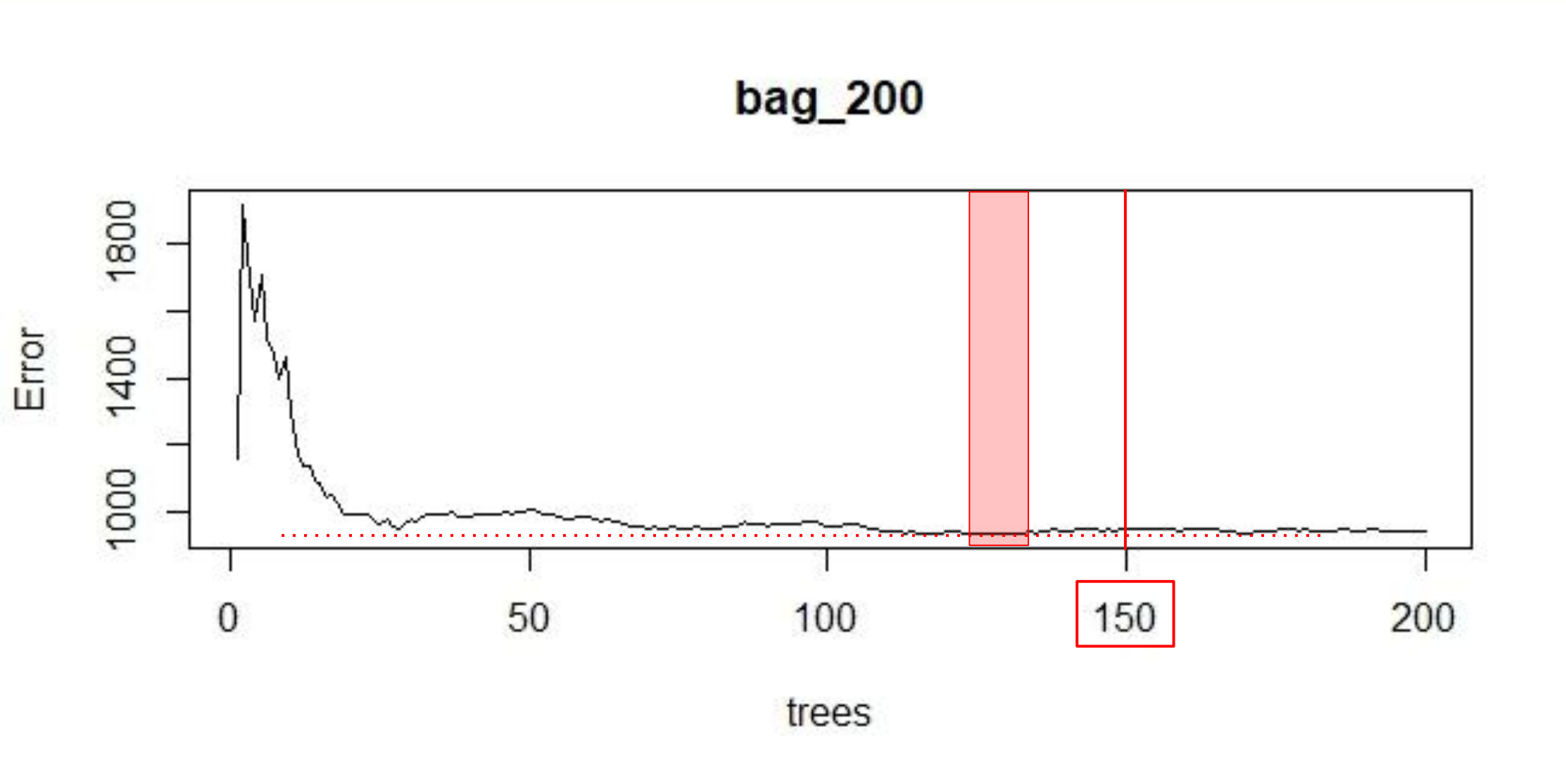
```
Call:
 randomForest(formula = TripCount ~ ., data = train, ntree = 20,
 mtry = 19)
               Type of random forest: regression
                     Number of trees: 20
No. of variables tried at each split: 19

          Mean of squared residuals: 1129.233
                    % Var explained: 80.25
```

```
Call:
 randomForest(formula = TripCount ~ ., data = train, ntree = 50,
 mtry = 19)
               Type of random forest: regression
                     Number of trees: 50
No. of variables tried at each split: 19

          Mean of squared residuals: 1030.613
                    % Var explained: 81.98
```

```
Call:
 randomForest(formula = TripCount ~ ., data = train, ntree = 200,
 mtry = 19)
               Type of random forest: regression
                     Number of trees: 200
No. of variables tried at each split: 19

          Mean of squared residuals: 942.7443
                    % Var explained: 83.51
```

m = p = 19	ntrees	RMSE	% Var Explained
19	**20**	**33.60**	**80.25%**
19	**50**	**32.10**	**81.98%**
19	**200**	**30.70**	**83.51%**

Bagging begins to diminish returns close to (B = 150)



Cross-validating Bagging (B = 150) is strong!

```
Call:
 randomForest(formula = TripCount ~ ., data = train, xtest = pred,
  ytest = predY, ntree = 150, mtry = 22)
               Type of random forest: regression
                     Number of trees: 150
No. of variables tried at each split: 19

          Mean of squared residuals: 903.5386
                    % Var explained: 84.2
                       Test set MSE: 867.44
                    % Var explained: 83.33
```

Validation RMSE = 29.45!

Low RMSE with bagging signals that a large part of the variables should be important if not all.

Insights from Random Forests and Bagging

- Nonparametric Models → Predictive Value for this data set
- Bagging > Random Forests! → Most to all included variables are important
- This makes sense:

  More than 50% of variables are indicators for **month**, and **all 12 can be important for deciding to ride a bike** regardless of temperature (vacations, festivals, school calendar, and other calendar-related variables approximated by month affect Austin's population and behavior)

Recommendations

- Create a new Bagging model with the following predictors:
  - Months
  - Membership type
  - Location

Use the model to:
-- Establish monthly supply standards at high volume stations, especially for valuable customer types.
-- Identify the right locations for promotions/ad campaigns for low volume months

THANK YOU!

Appendix

Cross-validating Bagging (B = 150) without Year as a Predictor

```
Call:
 randomForest(formula = TripCount ~ ., data = train, xtest = pred,
 ytest = predY, ntree = 150, mtry = 18)
               Type of random forest: regression
                     Number of trees: 150
No. of variables tried at each split: 18

          Mean of squared residuals: 1607.479
                    % Var explained: 71.89
                       Test set MSE: 1713.37
                    % Var explained: 67.07
```

Validation RMSE = 41.4

A higher RMSE, but more relevant predictive power for future years.

Removes cuts where pathway will be impossible (eg. year < 2019)

Cross-validating Bagging (B = 150) with McCombs

```
mlat = 30.2840788
mlong = -97.7376486

train = mutate(train, distanceMcCombs = sqrt((Lat-mlat)^2+(Long-mlong)^2))
test = mutate(test, distanceMcCombs = sqrt((Lat-mlat)^2+(Long-mlong)^2))
```

```
Call:
 randomForest(formula = TripCount ~ ., data = train, xtest = pred,
 ytest = predY, ntree = 150, mtry = 20)
               Type of random forest: regression
                     Number of trees: 150
No. of variables tried at each split: 20

          Mean of squared residuals: 971.3079
                    % Var explained: 83.01
                       Test set MSE: 833.57
                    % Var explained: 83.98
```

Validation RMSE = 28.87

Correlation Matrix

The correlations between Trip volume and rest of the variables remains quite low

	LongTrip	Lat	Long	Year	Month	Temp	TripCount	Month_Jan	Month_Feb	Month_Mar	Month_Apr	Month_May	Month_Jun	Month_Jul	Month_Aug	Month_Sep	Month_Oct	Month_Nov	Month_Dec	is_hot_day
LongTrip	100%	0%	0%	2%	0%	0%	-1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Lat	0%	100%	-11%	9%	1%	0%	3%	0%	0%	-1%	0%	-1%	0%	0%	0%	1%	0%	0%	0%	0%
Long	0%	-11%	100%	1%	0%	0%	-8%	0%	1%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%	-1%
Year	2%	9%	1%	100%	-10%	-9%	-14%	7%	8%	-1%	4%	-1%	-2%	-2%	-2%	-2%	-3%	-3%	-2%	-10%
Month	0%	1%	0%	-10%	100%	29%	-2%	-49%	-39%	-32%	-20%	-13%	-4%	5%	14%	23%	33%	40%	44%	22%
Temp	0%	0%	0%	-9%	29%	100%	0%	-48%	-37%	-16%	-5%	15%	32%	39%	38%	26%	3%	-21%	-33%	86%
TripCount	-1%	3%	-8%	-14%	-2%	0%	100%	-3%	-2%	8%	1%	1%	-1%	-1%	-3%	0%	4%	-1%	-4%	1%
Month_Jan	0%	0%	0%	7%	-49%	-48%	-3%	100%	-9%	-10%	-9%	-10%	-9%	-9%	-9%	-9%	-9%	-9%	-8%	-30%
Month_Feb	0%	0%	1%	8%	-39%	-37%	-2%	-9%	100%	-9%	-8%	-9%	-9%	-9%	-9%	-9%	-9%	-9%	-8%	-29%
Month_Mar	0%	-1%	1%	-1%	-32%	-16%	8%	-10%	-9%	100%	-9%	-10%	-10%	-10%	-10%	-10%	-10%	-9%	-8%	-31%
Month_Apr	0%	0%	0%	4%	-20%	-5%	1%	-9%	-8%	-9%	100%	-9%	-9%	-9%	-9%	-9%	-9%	-8%	-8%	-17%
Month_May	0%	-1%	0%	-1%	-13%	15%	1%	-10%	-9%	-10%	-9%	100%	-10%	-10%	-10%	-10%	-10%	-9%	-8%	32%
Month_Jun	0%	0%	0%	-2%	-4%	32%	-1%	-9%	-9%	-10%	-9%	-10%	100%	-10%	-9%	-9%	-10%	-9%	-8%	31%
Month_Jul	0%	0%	0%	-2%	5%	39%	-1%	-9%	-9%	-10%	-9%	-10%	-10%	100%	-9%	-10%	-10%	-9%	-8%	31%
Month_Aug	0%	0%	0%	-2%	14%	38%	-3%	-9%	-9%	-10%	-9%	-10%	-9%	-9%	100%	-9%	-9%	-9%	-8%	31%
Month_Sep	0%	1%	0%	-2%	23%	26%	0%	-9%	-9%	-10%	-9%	-10%	-9%	-10%	-9%	100%	-10%	-9%	-8%	31%
Month_Oct	0%	0%	0%	-3%	33%	3%	4%	-9%	-9%	-10%	-9%	-10%	-10%	-10%	-9%	-10%	100%	-9%	-8%	0%
Month_Nov	0%	0%	0%	-3%	40%	-21%	-1%	-9%	-9%	-9%	-8%	-9%	-9%	-9%	-9%	-9%	-9%	100%	-8%	-29%
Month_Dec	0%	0%	0%	-2%	44%	-33%	-4%	-8%	-8%	-8%	-8%	-8%	-8%	-8%	-8%	-8%	-8%	-8%	100%	-27%
is_hot_day	0%	0%	-1%	-10%	22%	86%	1%	-30%	-29%	-31%	-17%	32%	31%	31%	31%	31%	0%	-29%	-27%	100%