Maxwell Llewellyn
Technology Fundamentals for Business Analytics

Final Project

This project is based on the Kaggle competition Bike Sharing Demand which can be found here https://www.kaggle.com/c/bike-sharing-demand.

Executive Summary

Citizens of most metro areas are growing accustomed to seeing rows of branded bikes appear in their city streets as bike sharing programs grow in popularity. A bike sharing program allows a user to check out a bike from a rack near them, ride the bike somewhere else in the city, and then return the bike to another bike share rack. This is a modern convenience that allows city dwellers to use a bike when they need it and not have to worry about maintaining or storing a bike. Bike sharing programs can be used by anyone simply by using their credit card however most bike sharing programs also offer monthly memberships that include a certain number of rides every day. The bike sharing program this data is from, Capital Bikeshare in Washington DC, offered service to both nonmembers (casual riders) and members. Bike sharing programs are not just convenient for their customers, they are also a huge source of data.

Bike sharing programs use highly connected digital infrastructure which means that during normal use they produce a large amount of data that can be valuable if analyzed correctly. Everytime a user rents or returns a bike a data point in generated by the bike sharing system. This data can be at the granular level, specifying exactly who rented which bike at what time from what station and when and where they returned it, however, the data from the Bike Sharing Demand Kaggle competition is much more zoomed out. In this Kaggle competition analysts are provided a tabular dataset where each row represents a single hor. In each row of data there is information about the weather, the type of day (for example a holiday or a weekend) and how many casual, registered and total bike rentals there were that hour.

The goal of the Bike Sharing Demand Kaggle competition is to predict the number of bike rentals there would be given information about the future date and time as well as weather conditions. The metric used to score this competition is Root Mean Squared Logarithmic Error (RMSLE) which is similar to Root Mean Squared Error however the error is the difference in the logs of predicted values vs actual values instead of simply the difference in values.
Many analysts tried their hand the this Kaggle competition and submitted kernels. While many analysts tried using many models the best performing technique was Random Forest. Gradient Boosting came in at a close second with nearly twice the error and linear, ridge and lasso regression all tied for a distant third with nine times the error.

Data description and initial processing

The data for this Kaggle competition is supplied at a CSV with X rows and Y columns. Each row in the dataset represents an hour with information about that hour and how many people used the capital bike share that day. In the data, there are 12 rows datetime, season, holiday, workingday, weather, temp, atemp, humidity, windspeed, casual, registered and count. Thankfully this is nicely prepared data and there are no NA values in any column.

The datetime column is self descriptive. The season column represents spring, summer, fall and winter in that order from 1 to 10. The holiday column represents if the day is considered a holiday however unfortunately Kaggle does not specify which holidays of the many observed are counted in this dataset. The workingday column specifies if the day is neither a weekend or a holiday. The weather column specifies how bad the weather is with 1 being nice and 4 being terrible. More information about specific descriptions of each weather condition is available on Kaggle and in the accompanying code. The temp column specifies the temperature in Celsius and the atemp column specifies the “feels like” temperature in Celsius. The humidity column specifies the relative humidity. The windpseed column specifies the wind speed that hour in unknown units. The casual column specifies how many nonmember bike rentals while the registered column specifies how many member bike rentals and the count column specifies the total number of bike rentals.

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
1	2011-01-01 00:00:00	1	0	0	1	9.84	14.395	81	0.0000	3	13	16
2	2011-01-01 01:00:00	1	0	0	1	9.02	13.635	80	0.0000	8	32	40
3	2011-01-01 02:00:00	1	0	0	1	9.02	13.635	80	0.0000	5	27	32
4	2011-01-01 03:00:00	1	0	0	1	9.84	14.395	75	0.0000	3	10	13
5	2011-01-01 04:00:00	1	0	0	1	9.84	14.395	75	0.0000	0	1	1
6	2011-01-01 05:00:00	1	0	0	2	9.84	12.880	75	6.0032	0	1	1
7	2011-01-01 06:00:00	1	0	0	1	9.02	13.635	80	0.0000	2	0	2
8	2011-01-01 07:00:00	1	0	0	1	8.20	12.880	86	0.0000	1	2	3
9	2011-01-01 08:00:00	1	0	0	1	9.84	14.395	75	0.0000	1	7	8
10	2011-01-01 09:00:00	1	0	0	1	13.12	17.425	76	0.0000	8	6	14

First 10 rows of Data Raw

It is important to know that merely reading in the CSV file will have unintended effects. There are several columns of ambiguous coded categorical data. These columns use integers to represent a categorical or logical variable so when reading in the data using

tools such as R or Pandas an analyst must be sure to specify how to interpret these columns. Once the categorical variable `season` and the logical variables `holiday` and `workingday` have been transformed appropriately the analysis can begin.

	datetime	season	holiday	workingday	weather	temp	atemp	humidity	windspeed	casual	registered	count
1	2011-01-01 00:00:00	spring	FALSE	FALSE	Good	9.84	14.395	81	0.0000	3	13	16
2	2011-01-01 01:00:00	spring	FALSE	FALSE	Good	9.02	13.635	80	0.0000	8	32	40
3	2011-01-01 02:00:00	spring	FALSE	FALSE	Good	9.02	13.635	80	0.0000	5	27	32
4	2011-01-01 03:00:00	spring	FALSE	FALSE	Good	9.84	14.395	75	0.0000	3	10	13
5	2011-01-01 04:00:00	spring	FALSE	FALSE	Good	9.84	14.395	75	0.0000	0	1	1
6	2011-01-01 05:00:00	spring	FALSE	FALSE	Fair	9.84	12.880	75	6.0032	0	1	1
7	2011-01-01 06:00:00	spring	FALSE	FALSE	Good	9.02	13.635	80	0.0000	2	0	2
8	2011-01-01 07:00:00	spring	FALSE	FALSE	Good	8.20	12.880	86	0.0000	1	2	3
9	2011-01-01 08:00:00	spring	FALSE	FALSE	Good	9.84	14.395	75	0.0000	1	7	8
10	2011-01-01 09:00:00	spring	FALSE	FALSE	Good	13.12	17.425	76	0.0000	8	6	14

First 10 rows of data Transformed

A good first step in analyzing any dataset is to look at 6 number summaries and box plots of numerical data and counts of categorical and logical data.

```
      temp            atemp           humidity       windspeed     
 Min.   : 0.82   Min.   : 0.76   Min.   :  0.00   Min.   : 0.000  
 1st Qu.:13.94   1st Qu.:16.66   1st Qu.: 47.00   1st Qu.: 7.002  
 Median :20.50   Median :24.24   Median : 62.00   Median :12.998  
 Mean   :20.23   Mean   :23.66   Mean   : 61.89   Mean   :12.799  
 3rd Qu.:26.24   3rd Qu.:31.06   3rd Qu.: 77.00   3rd Qu.:16.998  
 Max.   :41.00   Max.   :45.45   Max.   :100.00   Max.   :56.997  

     casual        registered       count      
 Min.   :  0.00   Min.   :  0.0   Min.   :  1.0  
 1st Qu.:  4.00   1st Qu.: 36.0   1st Qu.: 42.0  
 Median : 17.00   Median :118.0   Median :145.0  
 Mean   : 36.02   Mean   :155.6   Mean   :191.6  
 3rd Qu.: 49.00   3rd Qu.:222.0   3rd Qu.:284.0  
 Max.   :367.00   Max.   :886.0   Max.   :977.0  
```

6 Number Summaries of Numerical Data

```
   season      holiday         workingday      weather     
 spring:2686   Mode :logical   Mode :logical   Good:7192  
 summer:2733   FALSE:10575     FALSE:3474      Fair:2834  
 fall  :2733   TRUE :311       TRUE :7412      Poor: 859  
```

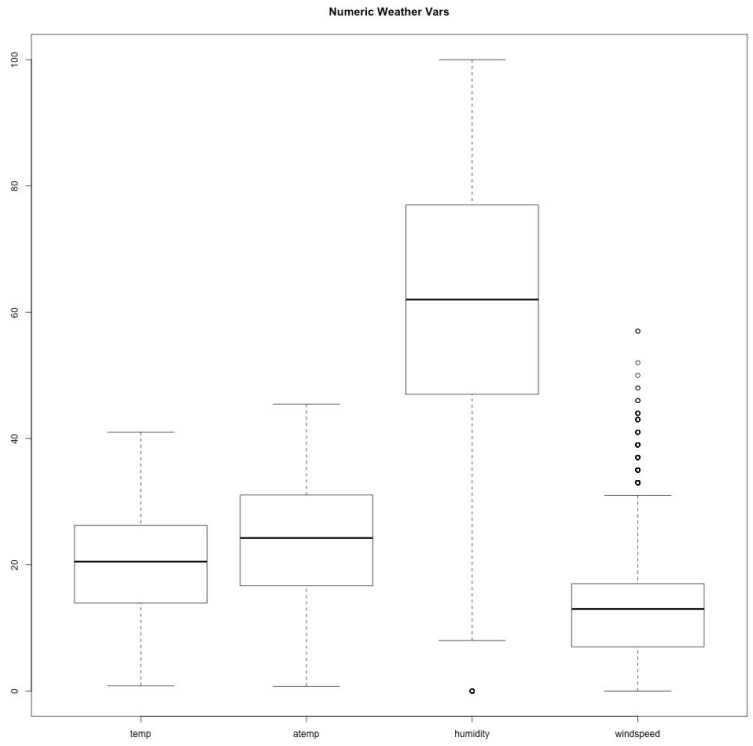
```
 winter:2734   NA's :0          NA's :0          Bad :   1
```

Counts of Categorical and Logical Data

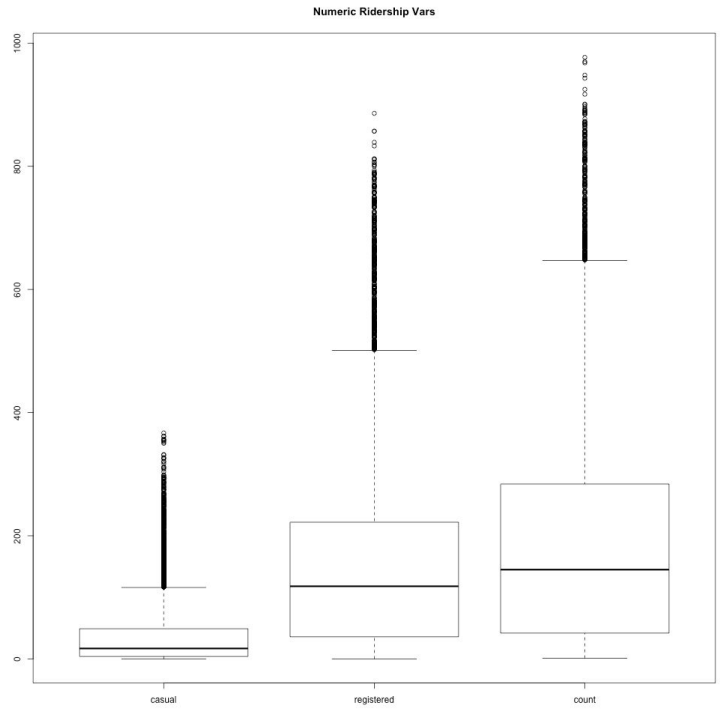
An interesting attribute of the temp, atemp, humidity and windspeed variables is that the mean is close to the median and the 1st quartile is a similar distance from the mean as the 3rd quartile. This suggests that these columns are not skewed, although this will be confirmed or denied by looking at the box plots. The count column however looks significantly skewed because the mean and median are far apart.

Looking at the season variable we can see that we have a relatively even number of each value. In the holiday variable we can see that most of the data points were not from a holiday day. Contrastingly most of the data points are from a workingday. The weather overall was good with many fair days, a few poor days and one bad day.

Box plots are a good tool to quickly analyze the skewness of data and detect outliers.



Temp, atemp, humidity, windspeed box plots



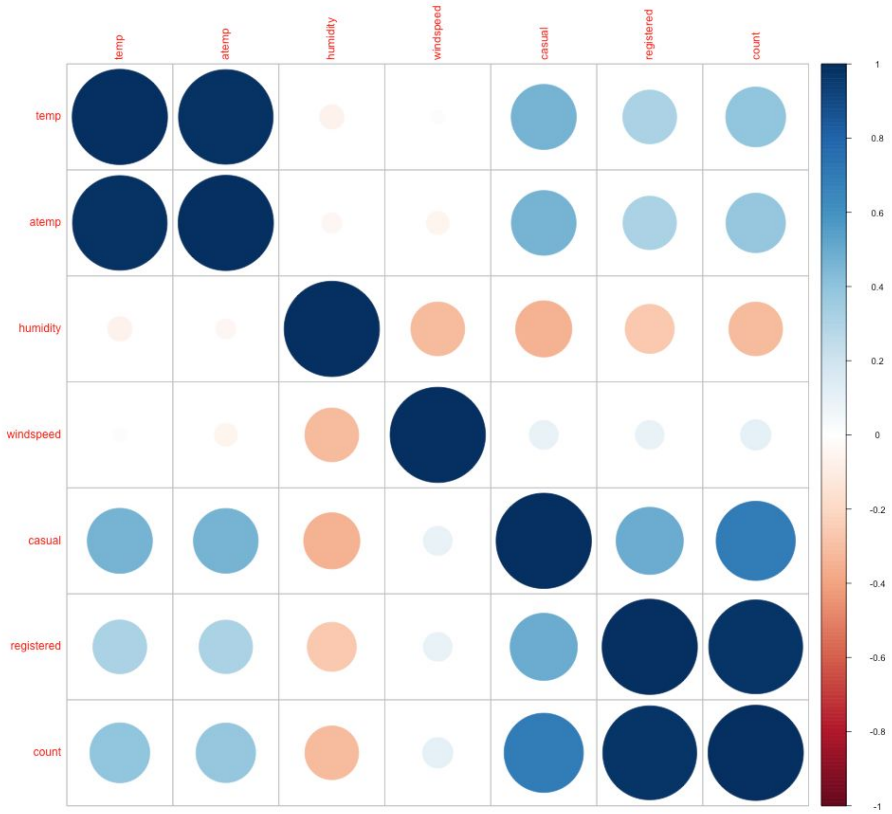
Ridership bot plots

Looking at these box plots we can see. That temp, atemp and humidity are all fairly symmetrical distributions. However windspeed, casual, registered and count are all skewed left. There are many outliers, especially in the ridership variables. It is hard to determine from the plot just how many outliers there are which is typical of a large dataset where event plotting 1% of points looks like a solid line. Based on this information outlier removal should be considered before modeling.

Correlation matrices are a good tool to find multicollinearity in the data.

	temp	atemp	humidity	windspeed	casual	registered	count
temp	1.00000000	0.98494811	-0.06494877	-0.01785201	0.46709706	0.31857128	0.3944536
atemp	0.98494811	1.00000000	-0.04353571	-0.05747300	0.46206654	0.31463539	0.3897844
humidity	-0.06494877	-0.04353571	1.00000000	-0.31860699	-0.34818690	-0.26545787	-0.3173715
windspeed	-0.01785201	-0.05747300	-0.31860699	1.00000000	0.09227619	0.09105166	0.1013695
casual	0.46709706	0.46206654	-0.34818690	0.09227619	1.00000000	0.49724969	0.6904136
registered	0.31857128	0.31463539	-0.26545787	0.09105166	0.49724969	1.00000000	0.9709481
count	0.39445364	0.38978444	-0.31737148	0.10136947	0.69041357	0.97094811	1.0000000

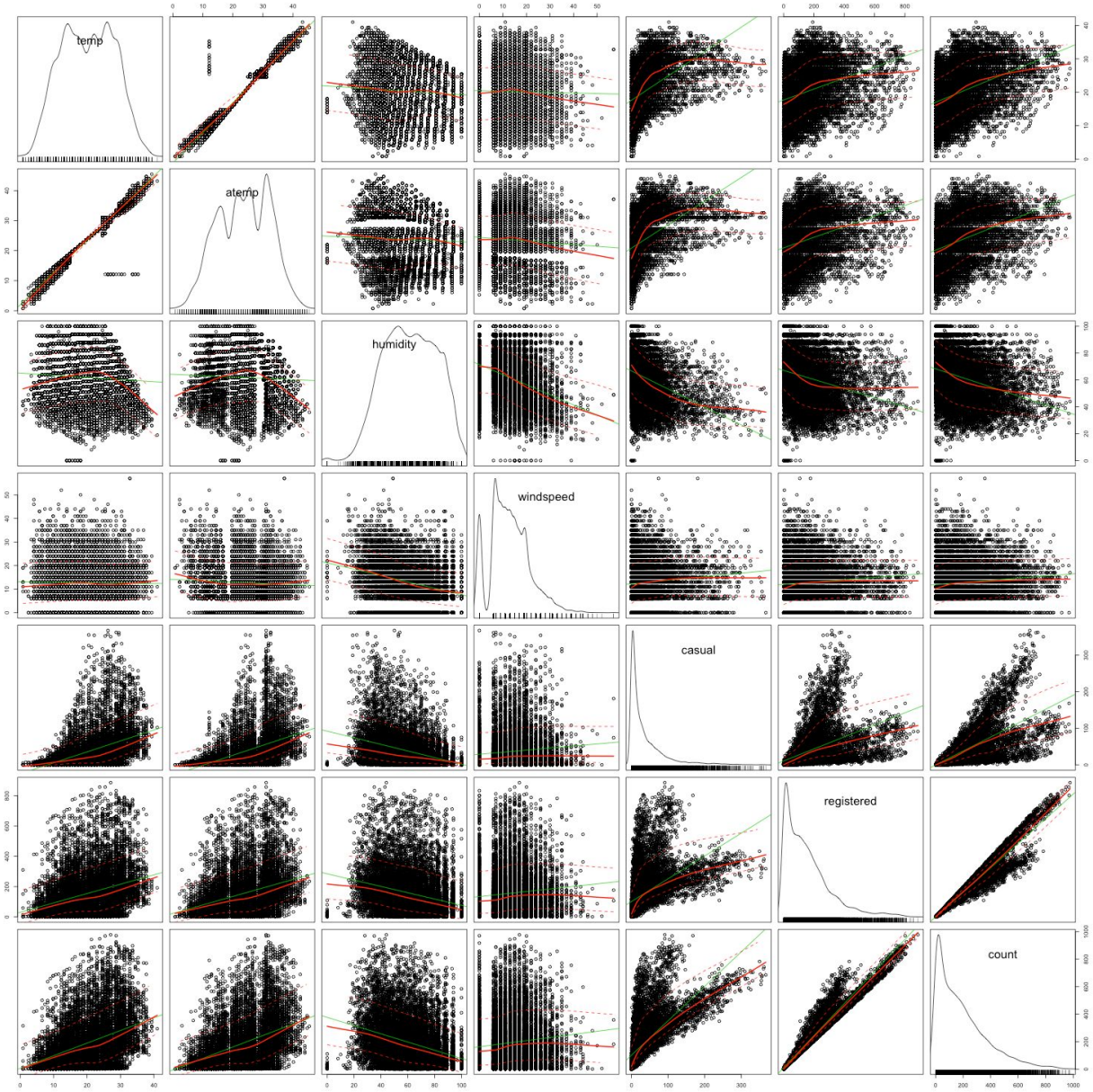
Correlation Matrix



Correlation Matrix Visualization

The correlation matrix shows us the obvious result that temp and atemp are extremely correlated and that casual, registered and count are fairly correlated. It is reasonable to see that humidity is negatively correlated with ridership and that temperature is positively correlated with ridership. Interestingly we see that humidity is negatively correlated with windspeed.

A scatterplot matrix produced by the car R library is a useful tool because it also generates histograms and performs a linear regression for each pair of variables.



Scatter Plot Matrix of numeric variables

In this scatter plot matrix we can see a similar story to the correlation matrix. Most of the data is relatively uncorrelated with the exceptions of temp with atemp and casual, registered and count alltogether. We can see from this histograms that temp, atemp and humidity are very roughly normal distributions and that windspeed, casual, registered and count are all skewed left.

Modeling and evaluation of 3 other solutions

Solution Overview

Kernel	Features	Modeling Approach	Performance (RSMLE)
EDA & Ensemble Model (Top 10 Percentile)	-New cols from Datetime -Coercing of columns to "categorical" data type	Linear Regression Ridge Regression Lasso Regression Random Forest Gradient Boost	0.977996 0.977996 0.978133 0.102804 0.189973
Comprehensive EDA with XGBoost (Top 10 percentile)	-Transforming count to log(count) -Creation of dummy vars for categorical data	XGBoost Random Forest	Did not put in notebook :-(
bikes	- Creation of dayofweek, hour, month and year columns from the datetime column	Random Forest	0.106703

Solution Commentary

EDA & Ensemble Model (Top 10 Percentile)

This kernel was made by Vivek Srinivasan and can be found here https://www.kaggle.com/viveksrinivasan/eda-ensemble-model-top-10-percentile.

Vivek has the clearest notebook of the three and his exploratory data analysis phase is very easy to follow. First, they read in the data and ascertains the shape, looks at the first 2 rows, and looks at the automatically chosen data types of each row. After that they create several features, hour, weekday, and month from the datatime column and then drops the datetime column for each row and casts the categorical data rows to categorical data. They then check for missing values and finds none. Vivek then makes box plots to find outliers (points above 3 standard deviations) in the count column and decides to remove those rows from the data. After that, they perform a correlation analysis and

removes highly correlated variables from the dataset. Then they look at histograms and a probability plot of the count column with and without outliers to visualize the distribution of the data. After creating additional count visualizations they build a random forest model to predict when the wind speed is 0 and then never uses this model. It is unclear why they included this model in the notebook. Vivek then builds models to test linear, ride and lasso regression, random forest and gradient boost models and chooses the one with the best fitting to the training data. They do not use any validation data which raises concerns about overfitting the training data however despite this the model performs reasonably on the test data.

Comprehensive EDA with XGBoost (Top 10 percentile)

This kernel was made by Mitesh Yadav and can be found here https://www.kaggle.com/miteshyadav/comprehensive-eda-with-xgboost-top-10-percentile

This notebook contains the most extensive exploratory data analysis of all the notebooks. Mitesh begins the analysis by viewing the first few rows of data. They then create a bot plot and remove rows where the value of count is higher than 3 standard deviations above the mean. They then continue the exploratory data analysis with a histogram of count with the outliers removed, bar charts to compare the frequency of the categorical and logical variables, and box plots of the continuous variables. After this Mitesh creates a graph of a rolling average of count and shows that over the span of the dataset ridership is cyclic and increasing. After that, they make join plots for every combination of continuous variables which show a few artifacts in the data but mostly show loosely correlated or uncorrelated variables. Mitesh then creates a correlation matrix visualization with a terrible color scheme and begins modeling. Mitesh creates a random forest model from scikit-learn and a gradient boosting model from xgboost however never displays the mean error, merely prints out a list of errors which makes this model difficult to compare to other models.

bikes

This kernel was made by meena and can be found here https://www.kaggle.com/meenaj/bikes
This notebook is the least descriptive of the three so it is fortunate the code is easy to follow. meena first creates new columns for dayofweek, hour, month and year from the datetime column. After this, they make pair plots between count and the other non date columns. meena does not drop or correctly interpret and categorical or logical variables but proceeds to make a correlation matrix anyway. After dropping the casual, registered and datetime columns meena separates the training data into a X and Y dataframe. Despite importing a GradientBoostingRegressor from sklearn.ensembles meena only uses a RandomForestRegressor and despite not correctly interpreting all of the data columns archives

a RSMLE almost identical to the other best. This good result suggests just how powerful the random forest model is.

Modeling

Training and Test Set

This Kaggle competition only provides a labeled training set. To avoid errors from overfitting during model evaluation this “training” set was split into a training and test set with 70% of the data in the training set and 30% of the data in the test set.

“Null” Model

Before journeying into complicated modeling techniques is it best to first consider the null model. In this dataset, the average value of the count variable in the training set is 191.5542, which produces an error of 1.549051.

Linear Regression Model

A linear regression model is interesting because it has both predictive and explanatory properties. An unexpected hiccup from using a linear model was that the linear model can erroneously produce negative results, which do not make sense in the context of bike sharing. Logically, the fewest bikes that can be ridden is zero. To solve this problem, and the undefined log of a negative value it produces in the error metric, all predictions less than zero were replaced with zero. It is also important to know, that R automatically creates dummy variables for the factor variables season and weather. The “default” state of these is weather=Good and season=spring.

Creating a linear regression model using all of these variables produces these coefficients. After replacing 81 predictions less than 0 with 0 the error is 1.443519.

```
Residuals:
    Min      1Q  Median      3Q     Max
-331.01 -100.28  -30.25   64.19  686.83

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    125.1618    10.7415  11.652  < 2e-16 ***
temp             9.8059     1.4369   6.824 9.51e-12 ***
atemp            1.2565     1.2584   0.999 0.318058
humidity        -2.8533     0.1116 -25.571  < 2e-16 ***
windspeed        0.4299     0.2353   1.827 0.067737 .
seasonsummer    -3.0801     6.3907  -0.482 0.629845
```

```
seasonfall     -39.9629     8.1574  -4.899 9.83e-07 ***
seasonwinter    66.4533     5.3846  12.341  < 2e-16 ***
holidayTRUE     -5.0374    10.8340  -0.465 0.641972
workingdayTRUE  -1.0518     3.8843  -0.271 0.786561
weatherFair     15.6127     4.2584   3.666 0.000248 ***
weatherPoor    -12.4425     7.1944  -1.729 0.083763 .
weatherBad     188.0083   152.7801   1.231 0.218518
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 152.7 on 7607 degrees of freedom
Multiple R-squared:  0.2815,    Adjusted R-squared:  0.2804
F-statistic: 248.4 on 12 and 7607 DF,  p-value: < 2.2e-16
```

Summary of Linear Regression using all variables

Interestingly despite this being a model using all of the variables in the dataset it only performs marginally better than the null model. Looking at the model summary we can see that not all of the variables have a statistically significant linear correlation with the count variable in this regression. Interestingly there is a strong relationship between temp and count but not between atemp and count. While part of this is due to multicollinearity between atemp and the other weather related variables in the dataset, it also suggests people make riding decisions more based on the reported temperature than the temperature they feel. Interestingly enough the holiday and working day variables have virtually no impact on the linear regression.

The two factor variables, season and weather are both interesting to analyze because some of their corresponding dummy variables are significant and some are not. For example, seasonfall and seasonwinter have a significant impact, however, seasonsummer does not. This should not be interpreted as bike riders ignoring when it is summer but rather that they treat summer and the default, spring, the same. In the weather factor variable, we can see that there is a significant difference between the weather being Good and Fair, however, a barely significant difference between being Good and Poor and no significant difference between the weather being Good and Bad. This could be due to the small number of data points that have weather as Poor or Bad.

Rerunning the regression with the least significant variables removed produces similar results to the complete regression. There are still 81 predictions less than 0 that needed to be replaced with 0 and the error was slightly worse at 1.443519.

```
Residuals:
    Min      1Q  Median      3Q     Max
-336.80 -100.39  -29.79   64.01  691.78

Coefficients:
```

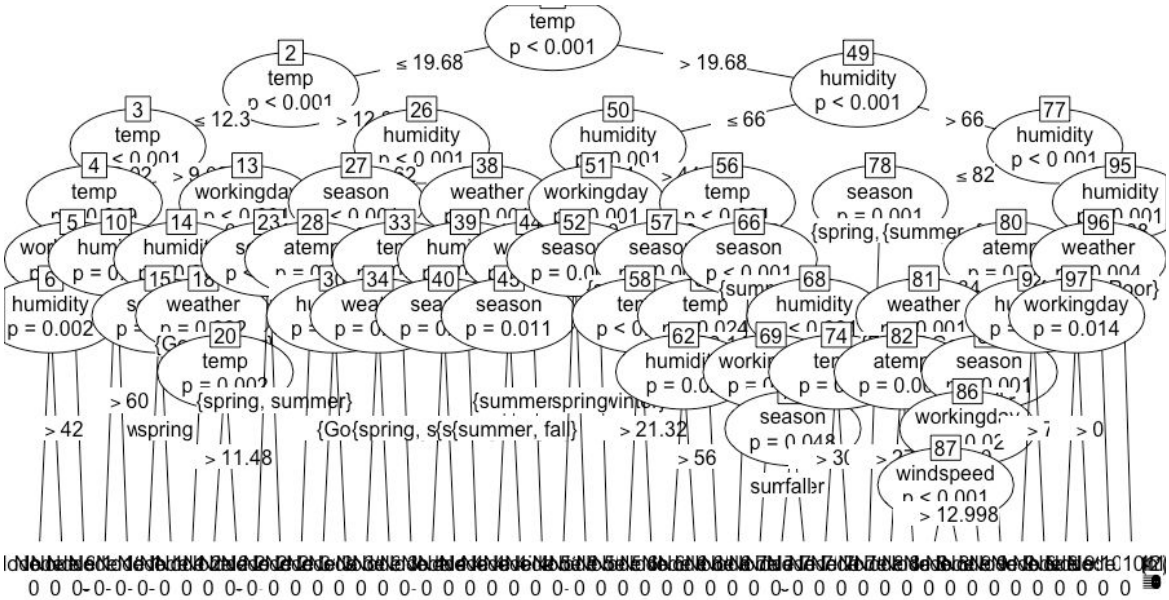
```
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  134.9522     8.8251  15.292  < 2e-16 ***
temp          11.1999     0.3772  29.695  < 2e-16 ***
humidity      -2.9048     0.1052 -27.624  < 2e-16 ***
seasonsummer  -3.3926     6.3850  -0.531 0.595200
seasonfall   -41.7641     8.0802  -5.169 2.42e-07 ***
seasonwinter  65.9393     5.3673  12.285  < 2e-16 ***
weatherFair   16.0537     4.2467   3.780 0.000158 ***
weatherPoor  -11.0063     7.0957  -1.551 0.120912
weatherBad   187.0206   152.7713   1.224 0.220920
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 152.7 on 7611 degrees of freedom
Multiple R-squared:  0.2812,    Adjusted R-squared:  0.2804
F-statistic: 372.1 on 8 and 7611 DF,  p-value: < 2.2e-16
```

Summary of Linear Regression using most significant variables

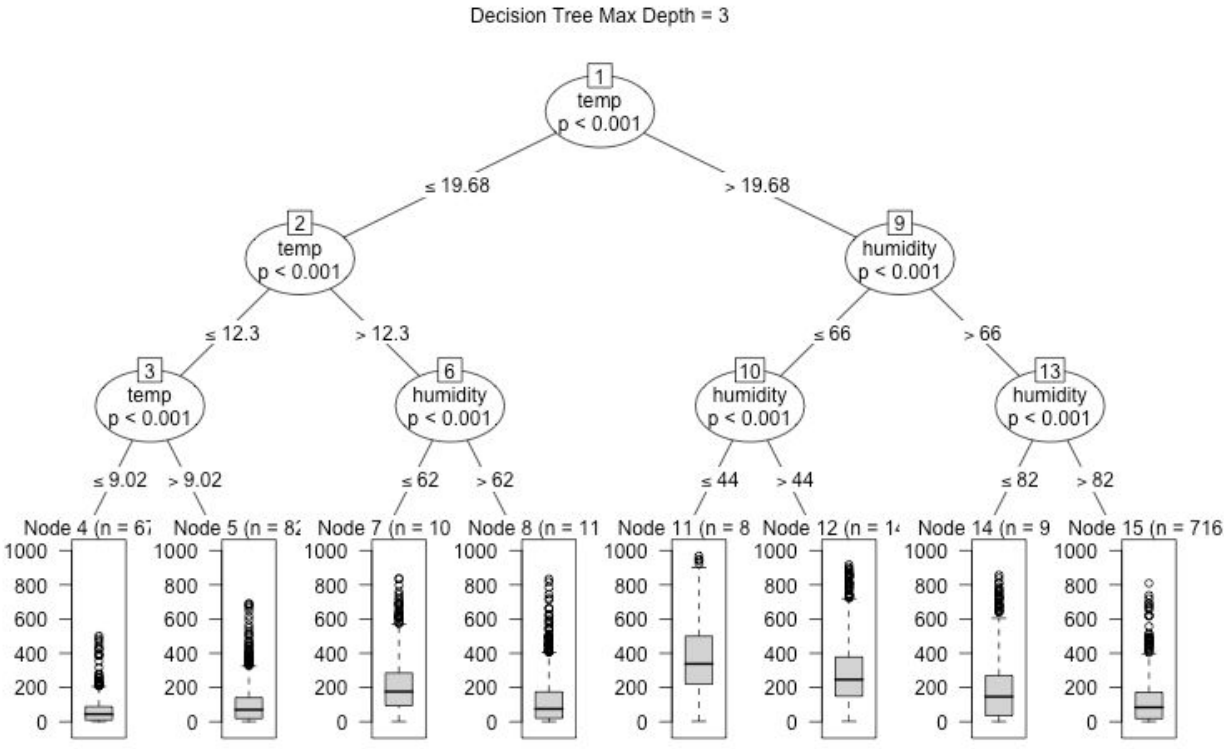
Decision Tree Model

A decision tree, like a linear regression, is an interesting model because it has both explanatory and predictive power. However without controls a decision tree can become unwieldy and overfit data. Simply using the default parameters of the ctree model in the party library produces a model with a low error of 1.339608 however the model is very difficult to interpret.



Unrestricted ctree model

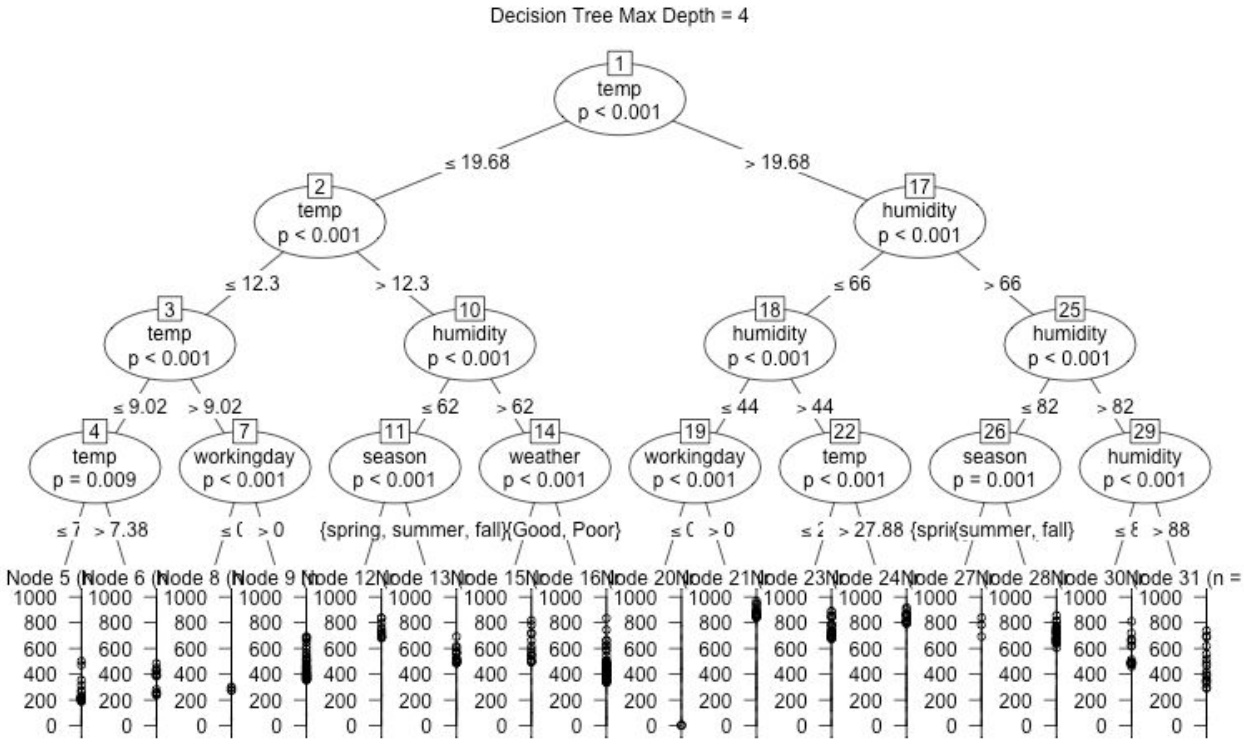
It is clear that the above model is useless without taking a lot of time to parse through the many many nodes and laters. Restricting the depth of a tree will make it more interpretable. Restricting the depth of the tree to 3 layers increases the error to 1.392339 however the resulting model is much easier to understand.



ctree with max depth 3

The error barely took a hit when we simplified the model and this simple decision tree still outperformed linear regression. This suggests there is a lot of nonlinearity in the dataset that the linear regression was not able to model well but the decision tree could accommodate. It is very informative that the only two variables this decision tree uses are temp and humidity. A decision tree will split using the variables that separate the data the most so we can see that temp and humidity are the two most impactful variables in the data.

Loosening the depth restriction of the tree to 4 layers decreases the error to 1.372008 without a significant loss in interpretability.



ctree with max depth 4

When the decision tree is expanded to include a fourth layer many more variables are brought in. The decision tree uses workingday and season twice and weather once. Nodes 7 and 19 show that on a working day (when working day = 1) people are much more likely to ride bikes.

Random Forest Model

Random forest models are popular on Kaggle because of their generally high performance. Using the cforrest model from the party library without any tuning parameters was able to produce an error of 1.299507. One technique to avoid overfitting in a random forest is when training each tree to only use a random subset of the input variables. This means that each tree is ignoring some variables which after all the trees are combined creates a more robust tree. Restricting the random forest to a random sample of three input variables yielded an error of 1.360319 and restricting the random forest to a random sample of six input variables yielded an error of 1.323216. Since the restricted tree error is higher on the test set the restricted model should not be chosen.

Summary

Understanding the usage of bike sharing networks is an important part of their continued growth. This data is not very granular, providing details only down to the day and not the individual ride, however, it still provides reasonable and informative insights into broad historical usage trends and allows rough prediction of future riders. Exploratory data analysis and modeling show that the most important variables in the dataset are temp and humidity. Exploring other Kaggle users Kernels shows that most modeling techniques were not able to significantly outperform the null model, only reducing the error by approximately one third. Random forests and gradient boosting were the only models that reduced the error by more than 80% with random forests performing the best. It was difficult to replicate these results, although part of that could be that the other Kaggle user's kernels calculated error on the training set while the analysis was done in this paper used a separate train and test set. With its data collection resources and the combined insight of the Kaggle community, the Capital Bikeshare is going to have a prosperous future.

Appendix

All code, data, images and other materials can be found here https://github.com/mllewellyn/TechFundamentalsFinalProject .