Data Mining, Analysis and Machine learning for Bike sharing

Colin Allen x15540607*, Alexander Millea x10349189†, Rafal Konarzewski x15019535‡,
School of Computing, National College of Ireland, Dublin

Abstract—**This report details the data mining and machine learning techniques used to explore data and predict how many people would avail of a bike sharing service. Throughout the process multiple algorithms were used for exploratory analysis, such as Decision Trees, Clustering, Multiple Linear Regression, to make predictions, ultimately conducting a statistical test for serial correlation and concluding with ARIMA and ARIMAX for time series analysis.**

I. Introduction

This report initially began as a solution to a Kaggle competition on bike sharing demand. [1] The analysis of the data set began and halfway into the analysis it was discovered that testing any predictions made on the data set would not be possible, as Kaggle restricts portions of the data set for the use of their competitions. After this, it was decided to switch over to the full data set as it was readily available online, and work continued. The report is split into three main sections, Data cleaning and pre-processing, Clustering and Algorithms. This is to show the work being performed over time, and how new information was gathered through the usage of multiple machine learning techniques.

II. Data Cleaning

Referred to as pre-processing or cleansing, data cleaning is a highly important step of any analysis, once the data has been collected. A vital part of the process, it can be time-consuming and may develop into an ongoing task throughout. The reason this is of high importance is due to its ability to help identify, correct and remove any errors found within the dataset, enhancing the data consistency. Furthermore, it helps to obtain a more accurate data mining model and overall statistical result.

For this study the bike sharing demand dataset was used which was retrieved from Kaggle [1], consisting of 12 columns and 10,900 rows of attribute-rich ordinal and nominal variables. The first steps carried out consisted of identifying the number of rows and columns, the structure of the data, the variable classes and the identification of null/na values. While it is unrealistic to expect that data will be perfect [2] the data didnt contain missing values or a great deal of error. This was helpful when carrying out the process although cleaning techniques still needed to be implemented. The next step was to extract the time (in hours), from the data column which was in the format of date and hour. Firstly, a time column was created. The time was then extracted from the date column using the strptime() function and saved to the new column.

The benefit of doing this is that it enhances the analysis and machine learning process as it gives further insight into data and can be used as an independent variable. Next, the date column was then reformatted to year-month-day. Class correction was then completed with season, month, weekday and weather all re-classed as factors containing their corresponding levels. The gsub function was then used to replace the weekday values from binary to characters which were most suitable for the exploratory analysis stage. Once completed, that concluded the pre-processing stages for the project. However, during the analysis stage, a roadblock was identified which prevented the completion of the given task of identifying the total count for bike rentals within each hour. Due to Kaggles terms, when taking part in one of their competitions a portion of the data is not supplied. In this instance the data was missing the days which were to be predicted, preventing the outcome to be compared to support the overall model accuracy. Because of this, a new bike-sharing demand dataset was sourced from the UCI machine learning repository [3] which contained the missing data. The cleaning process was again completed with minor changes to the script. This time, the first column instant was removed as it was meaningless, the hour did not need to be extracted as it already contained hour within a separate column, and the date column was re-formatted using as.POSIXct for GMT formatting.

Any columns which were named using abbreviations were changed to their correct spelling to help avoid any confusion it may have caused. Once this was completed, the exploratory analysis was conducted.

A. Descriptive and Exploratory Analysis

Descriptive statistics are used to help gain a more in-depth understanding of the data. It allows the analyst to further their analysis and to identify any underlying patterns or trends within the data. The aim is to unearth patterns so that assumptions and suggestions can be made regarding the data for both present and future workings. It also allows the analyst to identify whether the data is normally distributed and if any outliers are present which contributes to skewed data. The first task of conducting the descriptive analysis was to obtain the datas descriptive statistics. Using both the Hmisc and psych libraries, part of R Studio, two different approaches were taken. Doing so, not only were the statistics given as an overall summary but were segmented by each variable, giving a more in-depth outcome helping to gain a better understanding of the data; this has been included within the appendices. This helped to identify whether the data were normally distributed which was determined by evaluating the mean and median, outlining whether skewness and kurtosis were evident. Focusing on both the temp and atemp variables it is evident that both the mean and median values are the same, suggesting that the data is normal. However, to fully confirm this suggesting the mode value is needed. There is also little-to-none skewness or kurtosis present contributing to a normal distribution. The same observations were made between the variables casual and registered. Analysing the same statistics, both have major differences in both mean and median values. As well as this the skewness levels are strong, especially for casual with 2.50, giving the data a positive skew. Again, kurtosis levels were high within both but stronger within the casual data with a figure of 7.57. This would suggest the data is non-normal data, has a positive skewness and a large level of kurtosis present. To confirm these assumptions histograms were created which can be seen in figure.1.

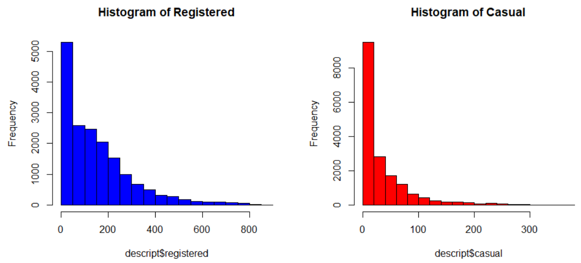


Fig. 1: Histograms

The next steps were to identify any outliers present. Using the histograms as an indication that there were outliers present, this was important to explore. Not only were the variables mentioned above explored, using the count variable which is a sum of both casual and registered as well as acting as the predictor, it was also compared with independent variables such as season, weather and weekday. Relating the descriptive analysis back to the overall task of predicting the number of bike rentals for each hour, these factors are highly influential in increasing/decreasing this figure.

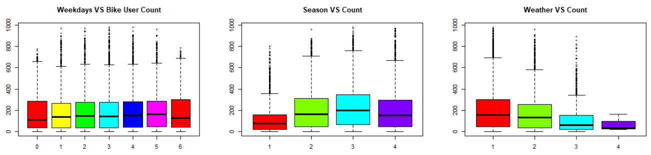


Fig. 2: Boxplots

The boxplots produced identify that there were outliers within the data, as well as showing the maximum, minimum and median values. It is evident within the count versus weekdays plot that the weekdays tend to have a larger amount of usage compared to the weekend days. It is also evident that each day contain a large number of outliers which is creating skewed data. In the season versus count boxplot, there is a significant drop in usage during the spring months while summer, autumn and winter all have a high, consistent amount of usage. It also details there is a difference in medians and would suggest there are differences present between spring and the other three seasons. The final boxplot compared weather and count. Most notable from this is the lack of usage when the weather consists of heavy rain and thunderstorms. As assumed, the highest usage is when the weather has clear-to-partial cloudiness.

Scatter plots were then created to visualise the user variables (casual, registered and count). This method is highly beneficial and clearly outlines the highest and lowest amount of usage based on the hour. Focusing on the registered output, it is evident that high levels of bike usage occur between the hours 7-9am and again between 5-7pm. The lowest level of usage occurs during the hours of 12-5am and again between 10-11pm. Between the times of 10am-4pm can be considered middle-level usage. Comparing this with the casual usage there is a significant difference in usage patterns. The highest level of bike usage for casual users is between 11am-7pm. Between the hours of 11 pm 8 am is the lowest levels of usage. From 7-10am and again at 7-10pm can be classed as mid-level usage. These differences can be down to several factors such as registered users having full-time employment, using bikes to travel and from work. Casual users are most likely made-up of students, tourists, and part-time employees as the usage is shown to be very high during normal working day hours, based on the 9-to-5 working day. Evaluating the count variable both assumptions are evident. To gain even further insight into the data, a subset of the date column was created which contained only the year (2011 and 2012). The reason to do this was to identify which year had the largest number of users overall. From the boxplot available in the appendices, it clearly shows that the number of users increased dramatically in 2012 compared with 2011. Once the data had been explored and investigated to identify any notable trends, the final steps carried out to conclude the exploratory analysis was to conduct a test for normality. A Shapiro-Wilk test was conducted to test for normality within each of the given variables. From conducting this, the statistical figures indicated that the data being used is non-normal data. To further this, a statistical analysis will be conducted to test to determine whether there is a significant difference in the median ranks between the casual bike users and registered bike users. The outcome of this test is detailed further within the report.

III. Clustering

Clustering is a very useful and beneficial technique to implement when working with any type of data and is particularly suited for the task which this study is focused on. Clustering, an unsupervised method, allows for the identification of similar groups within data. Unearthing these patterns assumptions can be made regarding the contained data however predictions cant

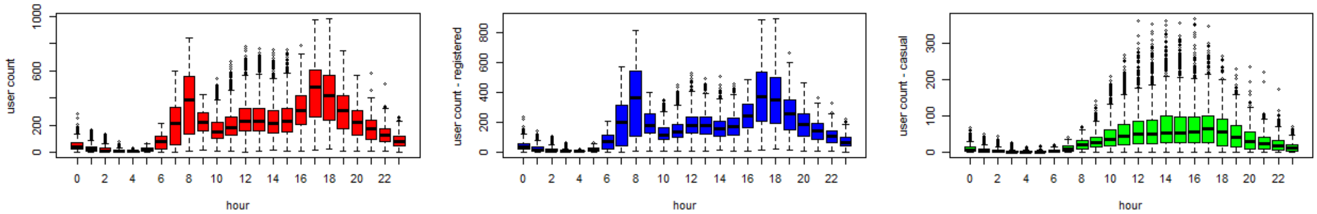


Fig. 3: Scatterplots

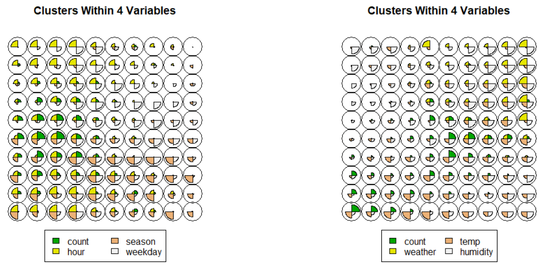


Fig. 4: Self-Organising Maps

be. This study implemented two types of clustering methods; k-means clustering and self-organising maps (SOM).

A. Self-Organising Maps

Self-organising maps (also known as Kohonen Maps) is a clustering technique that is widely used within deep learning. The unsupervised approach identified detects clusters within the data reducing dimensionality [3]. Using this technique helps identify clusters within the data through set variables. For this process clusters were determined using the count variable. The count variable was tested twice; firstly, compared with the variables count, weather, temp and humidity. Creating clusters using these variables, it is evident there is clusters within count, temp, and hour as well as weather, temp and humidity. The method indicated a very small amount of similarities between all four groups. The second test compared count with season, hour, and weekday. These clusters identified strong similarities within hour and season as well as season and weekday. However, like the previous cluster output, it only identified a small amount of similarities between all four variables.

Using this method helped visually display the data and helped further the understanding of the overall data being used. While the clusters have not shown strong groupings between all the tested variable, it is important that clustering methods are tested to expand on the data exploration.

B. K-means

K-means clustering was used to gain greater insight into the data and help identify key clusters within the data. The method uses a centroid (a centre point) which data points are assigned to, based on their closeness. To ensure that the optimal number of clusters was obtained, a k-value was determined [4]. To determine this value an elbow curve was used which is based on the use of Euclidean distance measurement. The elbow curve indicated the optimal k value based on where the line flattens out. In this instance, the elbow curve diagram (figure number) did not clearly outline the optimal k-value. As well as this, using test values such as two three and four, the clustering technique did not clearly distinguish clear, separate clusters. This is mainly due to the data containing a large volume of outliers, which classes the data as noisy data. Another issue using this technique is that the majority of the data is discrete data. To help solve this issue all factors were removed during the second testing alas, this did not improve the overall results. The k-means clustering technique output is included in the appendices.

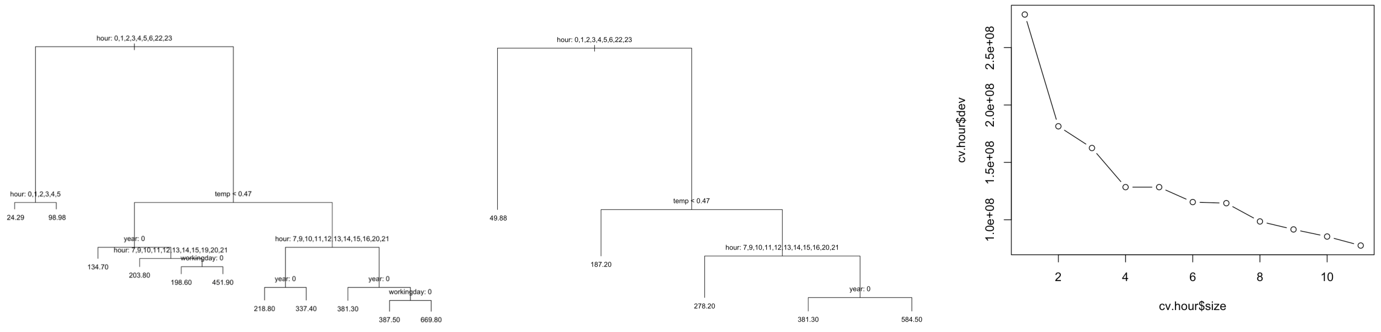
IV. Algorithms

A significant part of data mining is the use of algorithms to make informed future data predictions. Depending on the type of prediction that is requested several algorithms can be applied. As this study focuses on predicting the total bike count within each hour, suitable algorithms were chosen to determine this. This section will discuss, in detail, the performance of the applied algorithms and will detail whether their outcome was beneficial to the overall prediction. The algorithms used in this study included decision tree, boosted decision trees, random forest, multiple linear regression, arima, and arimax.

A. Decision Tree

One of the first algorthims used to understand the dataset and the underlying factors of the dataset was a Decision Tree. A decision tree was built to find out what are the most important predictors in the dataset and to see if any predictions could be made. The dataset that was cleaned above was read into R and restructured to fit to a decision tree. Date, Casual and Registered were removed. The Date wasnt relevant to the decision tree, and Casual and Registered were just a combination of our target variable Count. The dataset was split into Training and Test data, and then a tree was formed. The most important variables were Hour, Temp, Year and Workingday.

The structure of our tree shows the emphasis on those variables, specifically Hour and Temp being strong predictors



(a) First Tree

(b) Pruned Tree

(c) Cross validation: Size vs Deviance

Fig. 5: Decision Tree Graphs

for the earlier leaf nodes. From here the tree needed to be pruned for simplification. To prune the decision tree, an optimal level of pruning needs to be found, to find this cv.tree() function is used. This function performs cross-validation in order to determine the optimal level of tree complexity; cost complexity pruning is used in order to select a sequence of trees for consideration. [5]

After plotting the result of the cv.tree() function this graph is produced. This is a cross validation of the trees size by its deviance. The graph shows which value should be selected for pruning, a value of 5 is chosen as it is generally the middle of the graph between deviance and size.

The tree is pruned and our decision tree is simplified, and again shows the strength of the predictors Hour, Temp and Year. After pruning the tree, predictions can be made on the test data and see how accurate our decision tree is.

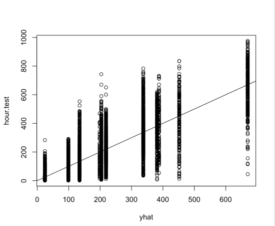


Fig. 6: Decision Tree: Predicted vs Actual

A plot of the test values vs predicted values, it can be seen that the values dont line up. The line is the prediction with the circles being the actual values. This leads to a Mean Squared Error of 10505.52, definitely nowhere near accurate for predictions.

B. Boosted Decision Trees

The predictive ability of A single pruned decision tree wasnt strong, so then boosted decision trees might give better results. The cleaned data was used and was restructured again to fit into a boosted decision trees model, Date, Casual and Registered values removed as before, for the same reasoning. The dataset was again randomly split into Training and Test datasets, and then a boosted decision tree model was built. A Gaussian Distribution, with 5000 trees and a depth of 4 was used as an initial model. A summary of the model shows the most relevant predictors on the model.

Hour was by far the strongest deciding factor, with a relative influence of 56.21. The next was Temp with a relevance of 8.03, and year with 8.00. A single decision tree shows which are the most important predictors, but the relative influence chart output with boosted decision trees shows just how strong each predictor is acting on the model. This model had an MSE of 1914.354, which is significantly better than the single pruned decision tree, but isnt as strong as it could be, so further analysis was performed. A model with the same amount of trees, just a reduced depth value of 2 was performed. Understandably, this gave even greater relative influence to Hour 59.07, as the trees were not as deep the other values had even smaller influence. Year with 6.97 and Temp with 6.74. The MSE for this tree was higher at 2705.432. This indicated that maybe deeper trees could reveal better results. A model with an increased depth of 6 was performed, which produced very similar results to the first model with the depth of 4. The relative influence of each predictor was nearly identical, and the MSE was 1947.051, again very similar to the first model. The one constant between all of the models was the strength of the hour variable, maybe removing the hour variable would allow the rest of the data have a stronger representation and lead to a more accurate model.

The removal of the Hour predictor allowed for the relative influence of the other predictors to increase and become more balanced. Humidity became the new strongest predictor, with a relative influence of 22.08, weekday had an influence of 16.45, with Atemp having an influence of 14.84. More importantly than the strongest influencers, the influence was further spread out amongst the data, this spread of relative influence across the data gave hope of a more accurate model. However, with an MSE of 20798.38, this model was the worst of all the tree based models, even the single pruned tree outperformed this model.

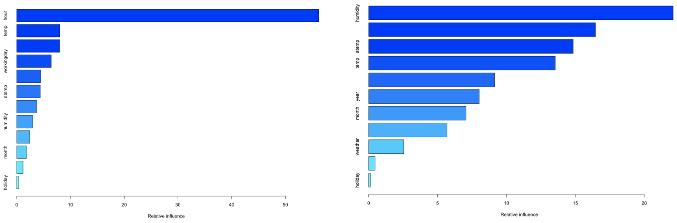


Fig. 7: Boosted Trees Relative Influence

C. Random Forest

Random Forest is an extension of the decision tree model which aggregates trees for classification and regression purposes. Considered a strong learner, it has an ability to deal with large amounts of data and at the same time avoids overfitting [6]. This is highly important and beneficial when applying the model to the bike sharing data.

To implement the random forest algorithm the data first needed to be subsetted, a requirement when using this technique. Partitioning the data into train data and test data, the implementation process could then begin. The ratio split between the data was set to 70:30. Using the R package randomForest, the random forest function was used to deal with an array of variables from the dataset. These variables included season, year, month, holiday, weekday, workingday, temp, weather, atemp, humidity, and windspeed. The next step was to determine the actual prediction. This was implemented using the R package caret. Caret is a set of functions that attempt to streamline the process for creating predictive models. [7] Using this, the prediction was then made which is outlined below.

The root-mean-square error (RMSE) (also known as the root mean squared error) were used to measure the differences between values predicted by a model and the values observed. The RMSE identifies the absolute fit of the model to the datahow close the observed data points are to the models predicted values [8]. Aiming to achieve an accurate model, the lower the RMSE value obtain the better the fit. The obtained RMSE value is 331.9368, indicating that the model is not highly accurate. Aiming to achieve a more optimized model, the tuneRF function was used. Based on these two obtained values the prediction can be optimized with the help of tuneRF function:

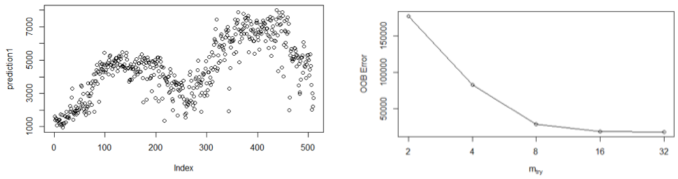


Fig. 8: Random Forest

D. Multiple Linear Regression

Multiple linear regression is a regression technique used to predict an outcome using two independent variables. Each independent variable must be unique to contribute to the overall prediction if they are not multicollinearity will occur. For this method, the aim was to predict the count of bike users based on a number of independent variables. Using the dependent variable count, the model was used to compare this with independent variables including season, weather and hour. Doing so, the first step was to create a correlation matrix to identify any highly correlated variables. From this, it is evident that both temp and atemp are highly correlated, but this was expected. For the purpose of exploring data both variables were included in the first test. From the test, the obtained multiple R-squared value was 0.3795 giving an accuracy of 37.95% concluding that the model is not a good fit for the data. The technique was then tested again without the temp variable. This, however, did not make any difference to the model accuracy as it gave the same accuracy percentage. The model was then tested again using various independent variables with the aim of increasing the prediction rate. The final accuracy achieved using the model was 34%. This indicates that the regression technique is not a good fit for the data and that any predictions made using this would have a high error rate. Investigating further, the QQ plot indicated that the data might be positively serially correlated, which would indicate as to why our linear model was performing so poorly. Both a Rank Von Neumann test and a Durbin-Watson test for serial correlation was performed after reading Bartels The Rank Von Neumann Test As A Test For Autocorrelation in Regression Models. [8] Bartels concluded that the DW test is more robust to deviations of normality than the alternatives, such as The Geary Test, and the critical values of DW are less affected by nonnormality than those of RVN or G, and the DW test is generally more powerful. [8] However, both of the tests result in rejecting the null hypothesis that the data does not have a serial correlation in favour of the alternative hypothesis that the data does have a serial correlation. A DW statistic of 0.8 shows a positive serial correlation, looking at a QQ plot shows this clearly.

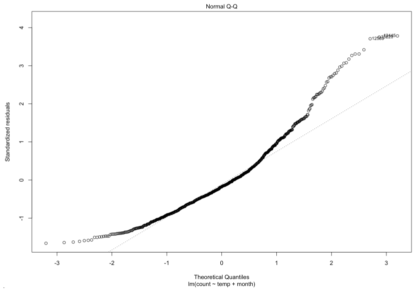
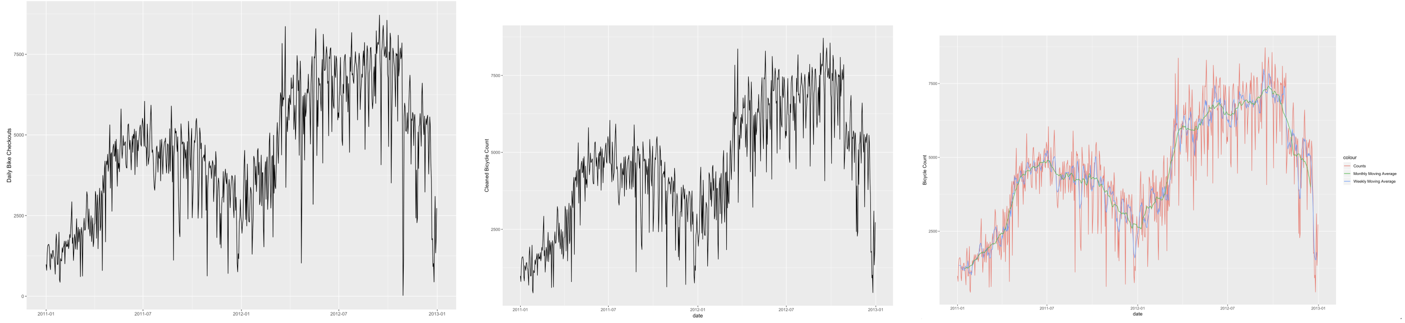


Fig. 9: Multiple Linear Regression: QQ plot

This Positive Serial Correlation indicated that Multiple Linear Regression would not be suitable to predict our data. To overcome this issue both ARIMA and ARIMAX were applied.



(a) Time Series

(b) Time Series Cleaned

(c) Weighted Moving Average

Fig. 10: ARIMA

E. ARIMA

ARIMA is an Auto-Regressive Integrated Moving Average. It is a forecasting model that relies on historical data to make future predictions. It compares current variables and historical variables, at a value of -1. These values are known as lags, lag1 is -1, lag 2 is -2.. up to lagN. It relies on three parameters, p, d and q. Auto-Regressive means the use of historical data to predict for the target variable Y. This is denoted by the p value. The d represents the amount of differencing for the Integrated Moving Average. Differencing a series is subtracting current and historical values d times. Differencing is used to stabilise the series when the data isnt stationary, q represents the error. A simple ARIMA model assumes non-seasonality, which means a test for seasonality and the data may need to be de-seasonalised. However ARIMA models with seasonal data is possible. ARIMA also assumes that the data is stationary, and stationarity must be tested before fitting an ARIMA model. The first model fitted will be a non-seasonal, stationary ARIMA model fitted with the dataset. First the dataset was read and restructured. The date and count variables were used to create a time series of bikes used over time.

This graph shows the positive serial correlation of our data, as the number of bikes being used increases over time. The series has a lot of outliers in the downward spikes in the graph that need to be cleaned up before further progress can be made. This can be done using the tsclean() function.

This graph is the same as before, but showing that the outliers have been removed and the trend can be seen more clearly. Next a weighted moving average is plotted to show the trend of the data Weekly and Monthly.

This graph shows that even though the individual count varies by day, there is a trend of the number of bikes used increasing, and a seasonal trend where the usage dips in the Winter months of 2011, only to rise again during the Summer of 2012 and eventually fall during the Winter months. This shows that the data has a seasonal aspect to it which will be investigated later. Next the data needs to be decomposed. Decomposing a time series is extracting three variables, seasonal, trend and cycle.

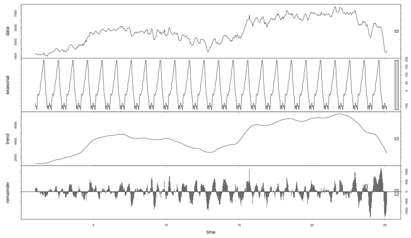
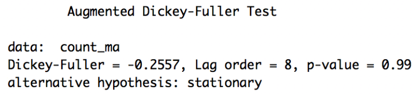


Fig. 11: Decomposed Time Series

This graph has a lot of information. The top box is just displaying the data over time. The Next box shows the seasonality, we can see that values peak and fall repeatedly. The third box shows the trend of the data over time and its general increase, with the final box being the error of residuals over time, showing that later predictions may be more difficult than earlier predictions. ARIMA relies on data being Stationary. A timeseries is stationary when its mean, variance and autocovariance are time irrelevant. ARIMA functions on using historical values to predict future values, and therefore the series must be Stationary for accurate results. The Augmented Dickey-Fuller test is a statistical test for stationarity. The null hypothesis is that the series is not stationary, the alterative hypothesis being that the series is stationary.

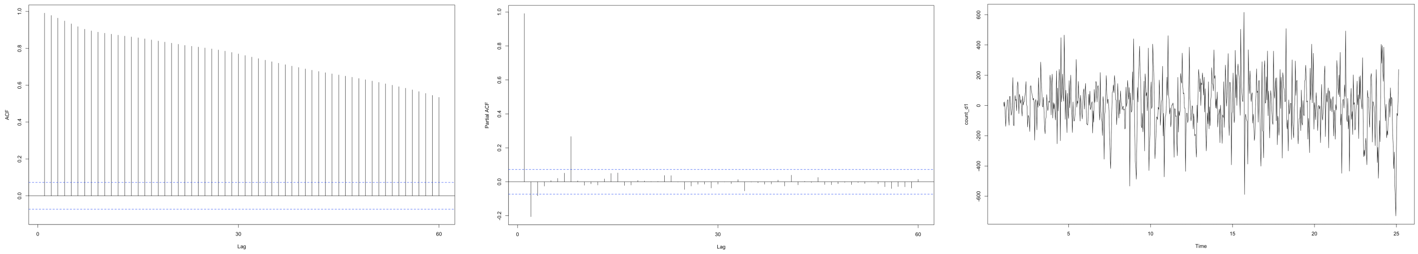


```
	Augmented Dickey-Fuller Test

data:  count_ma
Dickey-Fuller = -0.2557, Lag order = 8, p-value = 0.99
alternative hypothesis: stationary
```

Fig. 12: Augmented Dickey-fuller Test for Stationary series

Performing an Augmented Dickey-Fuller test results in a p-value of 0.99, which means the null hypothesis cannot be rejected, and therefore our data is not stationary. Due to having non-stationary timeseries, this must be dealt with through the use of Differencing. This means selecting the correct value of d for ARIMA. Autocorrelation plots (ACF) are a visual tool in determining the stationarity of a series. These plots can assist in choosing the d and q values for an ARIMA model. ACF plots show a correlation between a series and its lags.



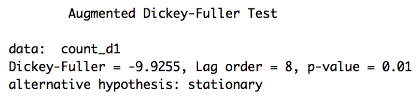
(a) ACF Plot (b) PACF plot (c) Stationary Time Series

Fig. 13: ARIMA Plots

A high correlation between the series and its lags can be seen. This is expected, the lags are the same data just a single value changed. Therefore high correlation is expected. There are also Partial Autocorrelation plots (PACF). These plots function similarly to ACF, they display correlations between a series and its lags, however this graph attempts to show correlations that are not explained by previous lags. PACF plot can assist in choosing a p value.

The PACF plot shows spikes at lag 1 and lag 7, which indicate that that a d value of 1 or 7 be chosen. Differencing the series with a value of 1 produces this series.

Which looks stationary. Performing the Augmented Dickey-Fuller test on the differenced series confirms that the series is stationary.



```
        Augmented Dickey-Fuller Test

data:  count_d1
Dickey-Fuller = -9.9255, Lag order = 8, p-value = 0.01
alternative hypothesis: stationary
```

Fig. 14: Augmented Dickey-fuller Test for Stationary series

With a p-value of 0.01, this is lower than our significant value of 0.05, so the null hypothesis that the series is not stationary is rejected in favour of the alternative hypothesis that the series is stationary. Creating ACF and PACF plots for this differenced series shows significant autocorrelations at lag 1, 2 and 7. This indicates that ARIMA models with a d of 1, 2, and 7 might be best.

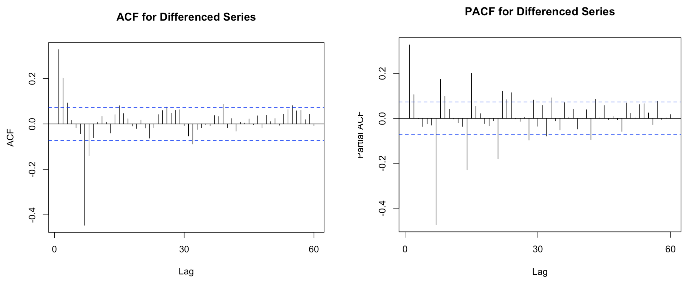


Fig. 15: Differenced ACF PACF plots

From there the auto.arima() function is used on the series. The auto.arima() function decides values of p, d and q for the model itself by choosing what It thinks are best. This is to confirm if auto.arima() agrees with the assumptions made above, such as using a differencing of 1.

The result of auto.arima() confirms the assumptions made above and uses a differencing of 1. Looking at the ACF and PACF plots there are large spikes at a lag of 7, and therefore an ARIMA model with a p or q of 7 is likely to give strong results. Then we fit an ARIMA model with the arima() function, with a p, d and q values of (1, 1, 7). A forecast of the next 30 values with this model produces acceptable results.

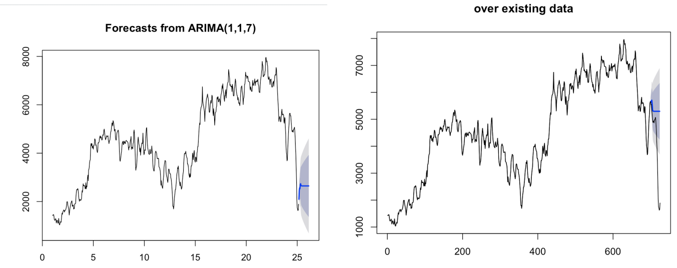


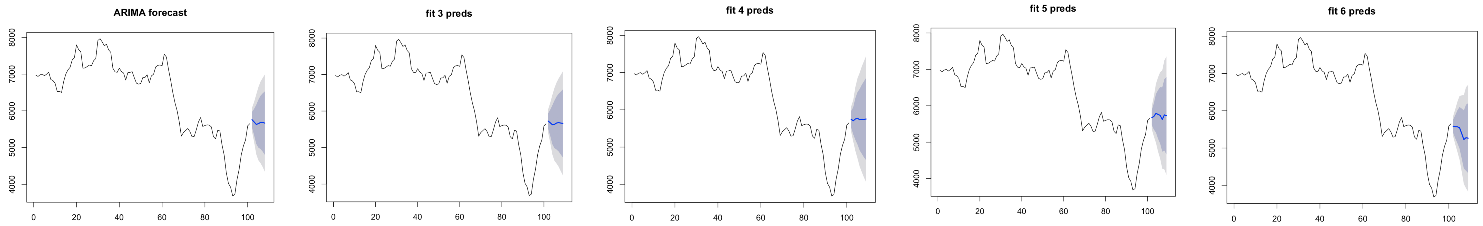
Fig. 16: ARIMA forecasts

However, the downside of ARIMA models shows here. After a certain amount of predictions, the model reverts to the mean. This is represented in the flat line at the end of this graph. A new model with the same p, d and q values were fitted onto most of the data, withholding 25 data points, and then the model was used to predict 25 points in the future. This was so a comparison between predicted and actual values could be shown.

The model starts off strong by following the predicted closely, but after a while eventually reverts back to the mean. This again is the issue with the ARIMA model, the further into the future predictions are made, the weaker the model is at making predictions and predictions eventually revert to the mean. This model had an RMSE of 118.0124.

F. ARIMAX

ARIMAX is a variation of ARIMA, it stands for Auto Regressive Integrated Moving Average with Exogeneous Input. The addition of Exogeneous Input means adding on other variables that will help the forecast being made.



(a) Baseline forecast

(b) Exogenous variables: temp

(c) Exogenous variables: temp, atemp, humidity, windspeed

(d) Exogenous variables: windspeed, humidity

(e) Final ARIMAX

Fig. 17: ARIMAX Plots

Using the auto.arima() function, a matrix with extra variables, such as Temperature, Humidity and Windspeed were added to the model to gain more accuracy. Another change that was made was to train the ARIMAX model on a smaller subset of the data to try and gain more accurate results and to avoid reverting the mean as quickly. First as a baseline, an ARIMA model (3, 1, 2) trained on a subset of the data was performed and an RMSE of 158.6698 was achieved, A worse result than our previous ARIMA model.

However, this is before we added any Exogeneous Input. Adding Temperature alone as a single Exogeneous input didnt do much for our prediction, an RMSE of 158.505 didnt vary much from our previous prediction without Temperature. With the graphs looking nearly identical.

Adding more variables made the model worse, with an RMSE of 166.8204, with extra variables of Temp, Atemp, Windspeed and Humidity. The graph even shows a more straight line prediction than previous models, with wider blue and dark blue confidence intervals. Removing Temp and Atemp seemed to make the model marginally stronger, with an RMSE of 154.4033, but still nowhere close to the ARIMA model.

A final ARIMAX model was fitted, however instead of using the auto.arima() function, the p, d and q values used for the ARIMA model were used instead. This ARIMAX model, with Windspeed and Humidity as exogeneous variables, and an order of (1, 1, 7) resulted in an RMSE of 134.2396.

G. Statistical Analysis

To further the report, a statistical analysis was conducted. As the data is unpaired data a Mann-Whitney U test was used to determine whether there is a significant difference in the median ranks between the casual bike users and registered bike users. The hypothesis for this test is outlined below.

$$\text{Ho: } M_{\text{casual}} = M_{\text{registered}} \text{ (null)}$$

$$\text{H1: } M_{\text{casual}} \neq M_{\text{registered}} \text{ (alternative)}$$

The alpha value that will be used is 0.05. Using an alpha value of 0.05 gives a 5% chance of committing a type I error. A type I error is the rejection of a true null hypothesis. The p-value obtained was 2.2e-16, less than the alpha value so therefore the null hypothesis (Ho) is rejected in favour of the alternative hypothesis (H1). This means that the test has found a significant difference between both casual and registered bike users.

$$U = 0.6871, p < 0.05$$

This test could be furthered with the introduction of added variables to determine whether season and weekday influence the difference between the two types of users.

V. Conclusion

Ultimately, multiple data analysis and machine learning techniques were conducted to attempt to understand and make future predictions using the bike sharing data set. Beginning with Decision Trees, Random Forests and Clustering to explore the data and decide on a model. When the chosen model Multiple Linear Regression revealed poor results and low accuracy, time series analysis was adopted and implemented to overcome the problems Multiple Linear had with data that had serial correlation.

Fitting ARIMA and ARIMAX models to our data didn't lead to a perfect solution, but a decent ARIMA model was built.The Exogeneous Variables of ARIMAX didn't improve the ARIMA model, adding more Exogeneous Variables proved to reduce accuracy, not increase it.

VI. Future Work

Future work on this data could include a more robust implementation of ARIMAX. Significant feature engineering could lead to more helpful Exogenous variables. A stronger model might be required to facilitate better predictions, as there are many factors contributing to the bike usage per day that neither Multiple Linear Regression, ARIMA or ARIMAX didn't use. A Recursive Neural Network could potentially lead to a more accurate model, being able to handle multiple factors. With the knowledge gained from this report, future work would definitely include an attempt to fit a neural network to the data.

References

[1] Kaggle. Bike sharing demand, 2019.

[2] S.Ganjoo Medium. Self-organizing maps, 2018.

[3] M. Steinbach P. Tan and V. Kumar. In *Introduction to data mining*, volume 1st ed, 2014.

[4] A. Trevino Datascience.com. Introduction to k-means clustering, 2019.

[5] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer Texts in Statistics. Springer, 2013.

[6] I. Reinstein Kdnuggets.com. Random forests(r) explained. 2019.

[7] M. Kuhn Topepo.github.io. The caret package. 2019.

[8] K. Grace-Martin The Analysis Factor. Assessing the fit of regression models - the analysis factor. 2019.