Understanding and predicting bike rental behavior using regression methods

Authors: Alexandru Damian, Liliana Alice Canter, Daniel Gotthardt, Ana Borda

Course: Data Science and Machine Learning

Institution: University of Southern Denmark

Lecturers: Julia Pahl
Marina Meireles Pereira
Jie Cai

Handed in: 07.01.2022

1. INTRODUCTION

Over the last decade, bike rental services have experienced a growth in popularity. Renting a bike may have become a more attractive option than buying one. We can think about several possible reasons that make cycling more attractive than other means of transport like, avoiding traffic jams in the cities, fastness to move around the cities or easier and free parking as examples. What might make renting a bike more attractive than owning one can be the removal of the initial purchase and maintenance cost of bikes, the no need of insurance for the bike in case it gets stolen and other practical things that are neglected when not owning an asset. However, there are also drawbacks of renting instead of owning, like the dependency on other people's rental behavior and the uncertainty that this trigger on having or not having an available bike to rent when necessary. We believe that people have similar renting patterns, led by the daily working life routine and external factors such as the weather condition, so it may be a common problem not to have a free bike to rent, when necessary, which may be an issue for the renting company regarding customer satisfaction.

Bike-sharing systems have automated the whole process of rental, return, and memberships, making it very accessible and convenient for individuals. Such systems generate lots of data that, when analyzed, provides insights about mobility in the city and could help city planners and rental companies in their decision-making process. For the analysis presented in this project we set up some questions that we believe are interesting and important for bike renting companies to be able to provide a good service for citizens:

- Can we uncover patterns in the data that explain citizen's renting behavior?
- Which are the variables that most affect the number of rented bikes?
- How accurately can the bike rental demand be predicted on a given day and hour?

This project, as part of the *Data Science and Machine Learning* course, tries to answer the questions mentioned above by using different data analysis methods learnt in the course, selecting the methods that are most appropriate for the type of dataset and purpose of the analysis. The aim of the project can be divided into two. On one side, we gain knowledge about different data analysis methods that can be applied for this purpose, and we also learn how to apply and evaluate them depending on the purpose of the project. On the other side, we learn about bike rental behavior, which is crucial, as explained before, to ensure a good quality service by cities and bike renting companies. This would at the same time help promote this

mode of transport protecting the environment from harmful air pollutants caused by combustion engine vehicles, especially in big cities.

The rest of the report contains the following sections. In Section 2 we both describe the data that will be used for the analysis and the type of analysis that is developed on this data, like the definition of the response variable. Section 3 describes how the project execution has been planned and developed, followed by the execution of the project in Section 4. The latter is divided in more subsections as it will be seen later, in which different methods are applied. Section 5 provides a conclusion on the findings and learning from the project development and the results of it. The rest of the sections from the table of contents are reference list and appendixes.

2. CHARACTERIZATION OF THE PROJCET

2.1. DATA DESCRIPTION

The dataset in which this project is based is called "Bike Sharing Dataset" and it was extracted from https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset. As the authors of the dataset state in the description, the dataset was originally extracted from https://www.capitalbikeshare.com/system-data and was complemented with weather situation data from http://www.freemeteo.com. The dataset represents information about number of bikes rented in Washington D.C., USA between years 2011 and 2012. The dataset consists of two tables containing the same information, which is the number of bikes rented by casual, registered and total users. One of them contains daily information and the other one hourly information. We will refer to this tables as **Day** and **Hour**.

Day has 16 features and 731 rows, where each row represents one day, from 1st January 2011 to 31st December 2012. **Hour** has 17 features and 17,389 rows where each of the rows in **Day** is broken to 24 rows, each of them representing an entire hour. The features are the same except for **Hour** where the hour is included as well. One thing we noticed is that the **Hour** dataset should have 17.544 rows (731*24), but it has 17,389 instead. This means that there are 155 data points missing, which we guess they were missing values that have already been cleaned by the owners of the dataset. All the features that both tables contain, one for each column, are presented in Table 1.

Table 1. Description of all features

Features	Description
instant	Record index from 1 to the total number of rows
dteday	Represents the date on year-month-day format from 1st January 2011 to 31st December 2012
season	Qualitative feature representing the season of the year using values from 1 to 4: Winter (1), Spring (2), Summer (3), Autumn (4)
yr	Qualitative feature representing the years 2011 and 2012 by using values 0 or 1: 2011 (0), 2012 (1)
mnth	Qualitative feature representing the months by using values from 1 to 12: January (1), …, December (12)
hr	Only the table **Hour** contains this column. It represents the 24 hours of a day by using numbers from 0 to 23: 00.00 (0), …, 23.00 (23)
holiday	Qualitative feature representing weather is holiday or not: Holiday (1), Else (0). Data for this column is taken from http://dchr.dc.go/page/holiday-schedule.
weekday	Qualitative feature representing the week of the day by using numbers from 0 to 6: Monday (1), …, Sunday (0)
workingday	Qualitative feature representing weather is a working day or not by suing 1 or 0: Working day (1), Weekend or Holiday (0)
weathersit	Qualitative feature representing the weather situation by using numbers from 1 to 4: Clear or Few Clouds or Partly Cloudy (1), Mist + Cloudy or Mist + Broken Clouds or Mist + Few Clouds or Mist (2), Light Snow or Light Rain + Thunderstorm + Scattered Clouds or Light Rain + Scattered Clouds (3) Heavy Rain + Ice Pallets + Thunderstorm + Mist or Snow + Fog (4)
temp	Environment measured temperature in Celsius divided by 41, being 41 the maximum value measured
atemp	Feeling temperature in Celsius divided by 50, being 50 the maximum value measured
hum	Normalized humidity. Measured values divided by 100
windspeed	Normalized windspeed. Measured values divided by 67
casual	Count of casual users on that day/hour
registered	Count of registered users on that day/hour
cnt	Total users on that day/hour

2.2. CASE DESCRIPTION

As it can be seen in the previous section, this dataset contains a mixture of quantitative and qualitative features, and the type of method to apply will depend on which feature we choose as a response. In our case, as explained in the introduction, we want to understand the behavior and try to predict how many bike rentals occur in a certain day or hour, depending on different weather and seasonal conditions. Therefore, we define the feature **cnt** as the response, which represents the total users that have rented a bike in that day or hour. We only look at the total users and not to the casual and registered since we are not interested on the distinction of these two types of users, but in the total amount instead. However, from a business perspective it would be very interesting to understand how casual users and registered users behave, if they follow different patterns, if they are not equally affected by the different features, etc.

Because the response feature **cnt** is quantitative and because we want to predict that quantitative value, this project is defined as a regression problem which is at the same time a supervised learning problem, since all data points have a known label. As it will be seen in the following section both linear and multiple linear regression are used. By using these approaches several models are analyzed and evaluated based on the training and test error and prediction accuracy. Other methods are also applied, like Poisson's regression and time series analysis, to see if they give better solutions when it comes to the prediction.

3. METHODOLOGY

The following section shows the different steps and logic that has been followed for the analysis of the data. As it will be seen, different regression methods have been used and evaluated, to see which of these methods is more appropriate to reach the aim of the project. To accomplish the aim of this project we need to first identify if there is any pattern that our data follows regarding the total amount of users that rents a bike in a daily or hourly basis, in order to understand the user behavior and the features that most define this behavior. Secondly, we also want to check how accurately we can predict the number of users that will rent a bike in a certain day or hour, based on different values of the features selected for the different models. The same will be done for the **Day** and **Hour** data tables to understand both behaviors, daily and hourly, since the patterns and prediction accuracy may be different.

The following section is divided into 5 subsections, which can be associated to the different steps that have been followed for the project development. Section 4.1 shows some preliminary plots from the presented datasets to have a first overview of how the data looks like, to understand how the features may be affecting the response and to have a first impression of which regression methods could perform well. Section 4.2 presents the results from applying simple linear regression for both tables and section 4.3 presents the results from multiple linear regression, which involves feature selection and model assessment method application. Because of the not very successful results, Poisson's regression and time series analysis were added. Poisson's regression can improve the accuracy on the **Hour** dataset, since as it will be seen, the histogram of the feature **cnt** shows that **cnt** follows a distribution similar to Poisson´s. Moreover, we think that time series analysis could be a better method regarding the nature of our dataset and type of analysis.

4. EXECUTION OF THE DATA SCIENCE PROJECT

4.1. UNDERSTANDING THE DATA

As explained, the aim of this section is to have a first look at the behavior of the data to understand it better and to make a pre-analysis of it. The first thing that is interesting to see is which distribution does our response follow, since this gives an idea of which regression method may suit better for fitting the data and predicting the response. The distribution of the response **cnt** can be seen by building a histogram, and that is what is shown in ***Figure 1*** and ***Figure 2***. The function *hist()* has been used in R. The code is provided in *Histograms*.

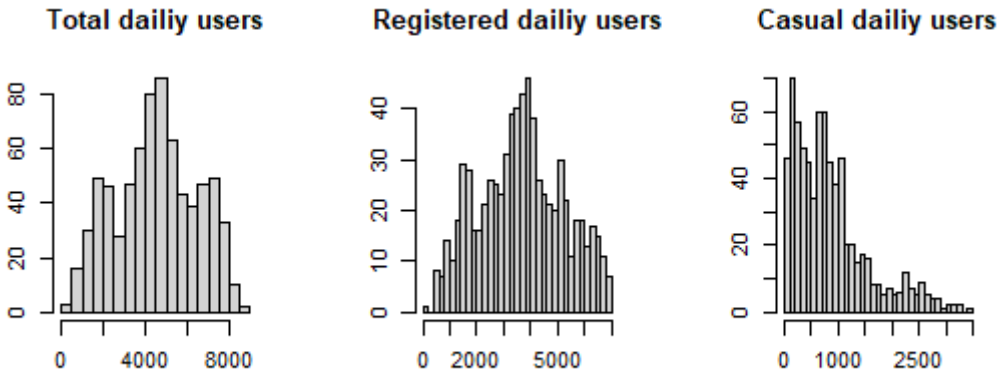


Figure 1. Histogram for total, registered and casual users in Day

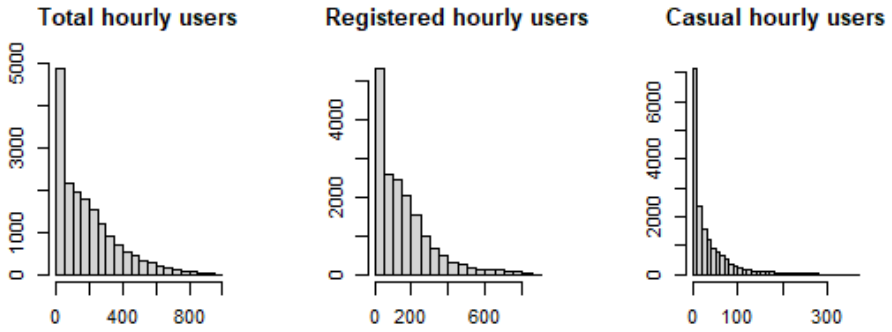


Figure 2. Histogram for total, registered and casual users in Hour

If we look at the histograms in **Error! Reference source not found.**, which shows the number of daily casual, registered and total users, we can see that the number of total and registered users follow a normal distribution, or at least something close to normal. However, for the casual type number of users renting, we can say that the response follows an exponential distribution or Poisson distribution with lambda 1 or 2. What we can infer from this is that, for the total and registered users a least square linear regression model could initially perform well to fit the data and predict the number of users that will rent a bike daily. If the casual user's amount wanted to be predicted though, the least square linear regression model may not be the most appropriate one. For this case, Poisson's regression could be used as suggested on page 170 of [1]. However, as explained before, only the total amount of users will be considered in this project.

If we do the same for the hourly case, which is shown in **Error! Reference source not found.**, none of the responses follow a normal distribution, which again means that linear regression may not perform well in this dataset for our purpose. Instead, Poisson's regression model could do better.

Another interesting thing to check is the scatterplot and correlation matrix, to see if there could be any linearity between any feature and the response, and to check for correlation between the features. The first two columns, **instant** and **dteday** have been erased from the original datasets. The first column is just the row number, and the second column is the date, which is not a numerical variable and thus the scatterplot and correlation matrix does not make sense. This applies for both datasets. The code is provided in *Scatterplot and Correlation matrix***.** The full scatterplot for **Day** is provided in the appendix section,

since the pictures are very small to see. However, we show in the next figures the relationship of **cnt** and some features selected based on highest apparent relationship.

Focusing on **Day** there are some features that clearly affect the response variable. First, from the **yr** we can conclude that there are more users in 2012 compared to 2011. This only shows that there might be an increasing tendency over the years, but we cannot use this information, we would need to have data from other years to confirm the possible increasing tendency. The season and month, which we know that are correlated, also affect in the number of users renting a bike. In **Figure 4** and **Figure 3** it can be seen that during the winter (so season 1 and months 12, 1 and 2) the number of users is lower than in spring, summer and autumn, being summer the season where the user amount is the maximum and July the most popular month for renting bikes. We think this can be explained by worse weather condition and lower average temperature during the wintertime.

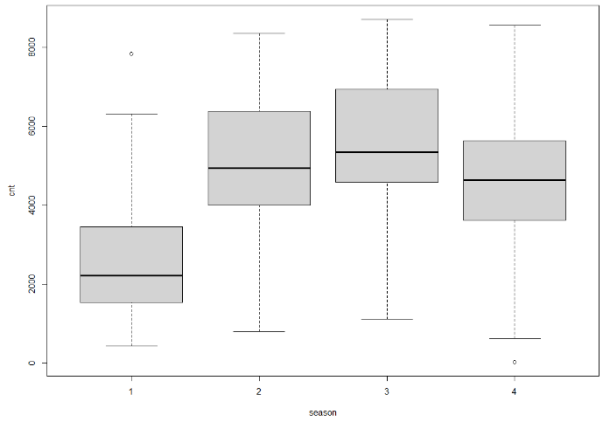


Figure 4. Season vs total users

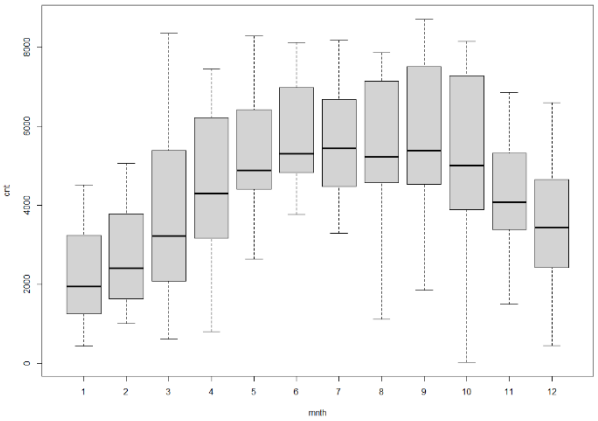


Figure 3. Month vs total users

Looking at Error! Reference source not found. we can confirm this statement, where it is shown that the worse the weather becomes, especially for very bad conditions (weathersit = 3) the lower the user amount becomes. The temperature and feeling temperature, which are correlated, change the amount of rentals, being this amount higher the higher the temperature is, which is also connected to the seasonal pattern or behavior seen before. This relationship is shown in Error! Reference source not found. , and at the first glance it may seem that the total users (**cnt**) and the feeling temperature (**atemp**) have a linear relationship, which will be tested in the next section. Lastly, looking at holiday in **Error! Reference source not found.,** it seems that more people rent bikes when it is not holiday (0) and less people do on holiday (1), even though the difference is not very strong. The variance is also higher when is not holiday.

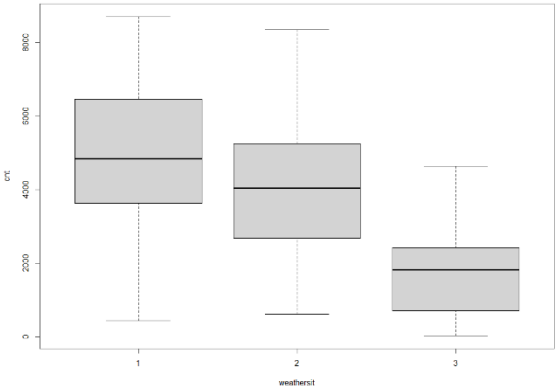


Figure 5. Boxplot for weather situation vs total users

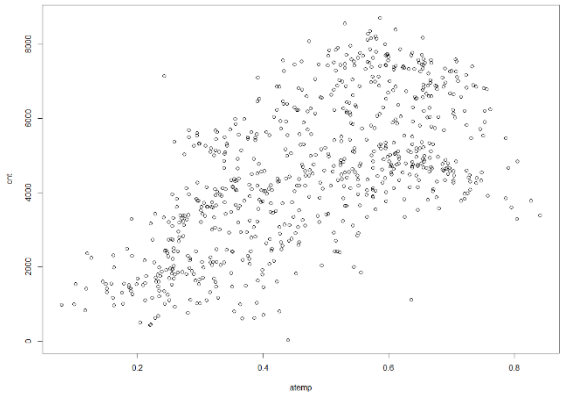


Figure 7. Feeling temperature vs total users

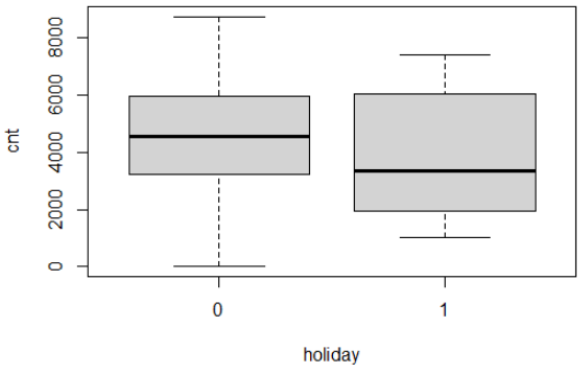


Figure 6. Boxplot for holiday vs total users

Summarizing, we can conclude that a) in colder seasons and months less people tend to rent bikes, b) as the weather condition worsens the number of users also decreases and that c) on holiday less people rent bikes. However, looking at the correlation matrix shown in **Figure 8** there is no strong correlation between the response variable and other features. The maximum correlation occurs with the year (**yr**) and with the temperature (**temp** and **atemp**) which account for 0,57 for the case of **yr** and 0.63 for **temp** and **atemp.**

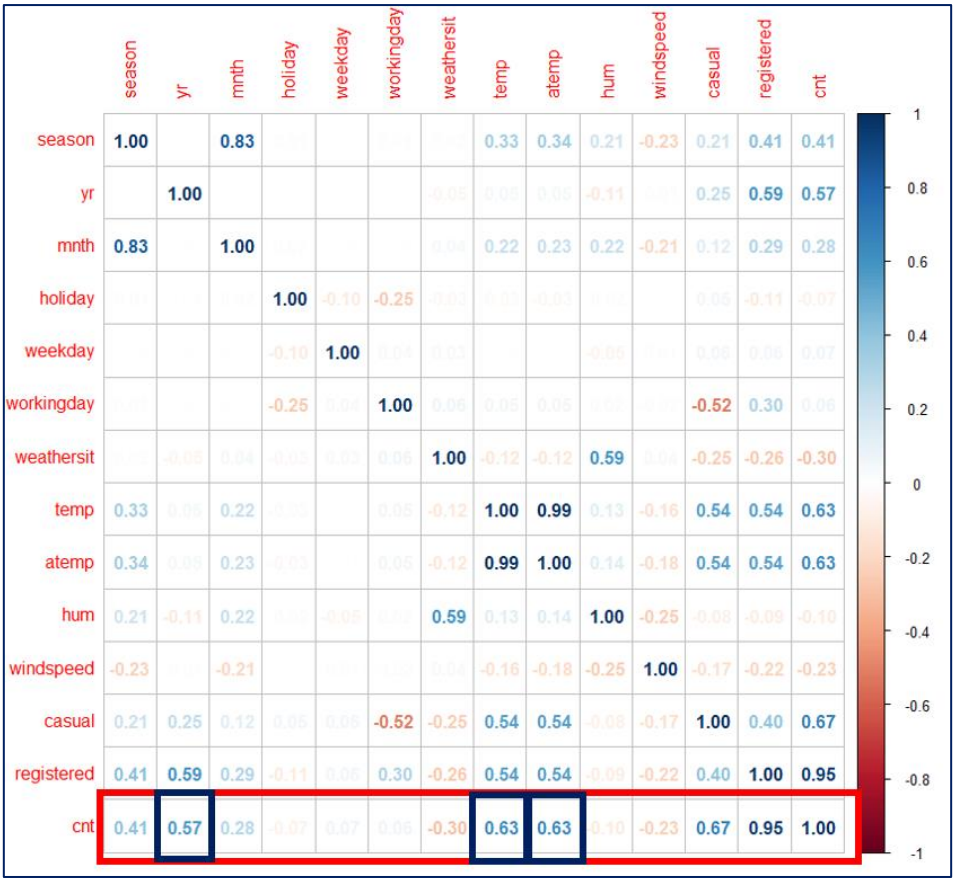


Figure 8. Correlation matrix for Day

For the case of **Hour**, the scenario is very similar. The same features that most impact the number of daily users is the same that most impact on the daily basis. No figure is shown since it is very similar to **Day** but with more variation. The values in the correlation matrix are different, which are shown in **Figure 9**, still being the temperature and feeling temperature the features with maximum correlation with **cnt**, with a value of 0.4 which it cannot be considered a high number. The feature **hr** also has a similar correlation to **cnt** with a value of 0.39. The relationship between the hours and the total user amount can be seen in **Figure 10**. This figure explains very well that there is a pattern that users follow on an hourly basis. At 5am the number of users start to slowly increase, reaching a peak at 8am, which may be due to the beginning of the working day. There is another visible peak occurring at 5pm and 6pm, possibly because of the end of the working day. From 6pm on, at night hours, the users decrease steadily reaching the minimum at 4am on average.

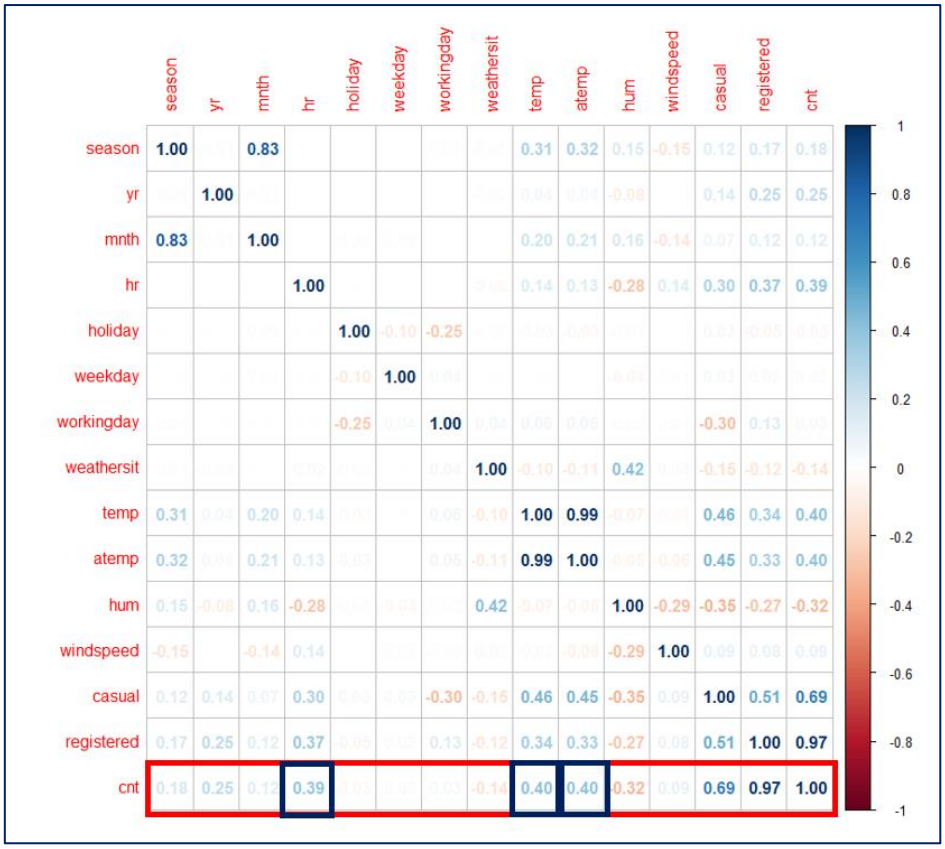


Figure 9. Correlation matrix for Hour

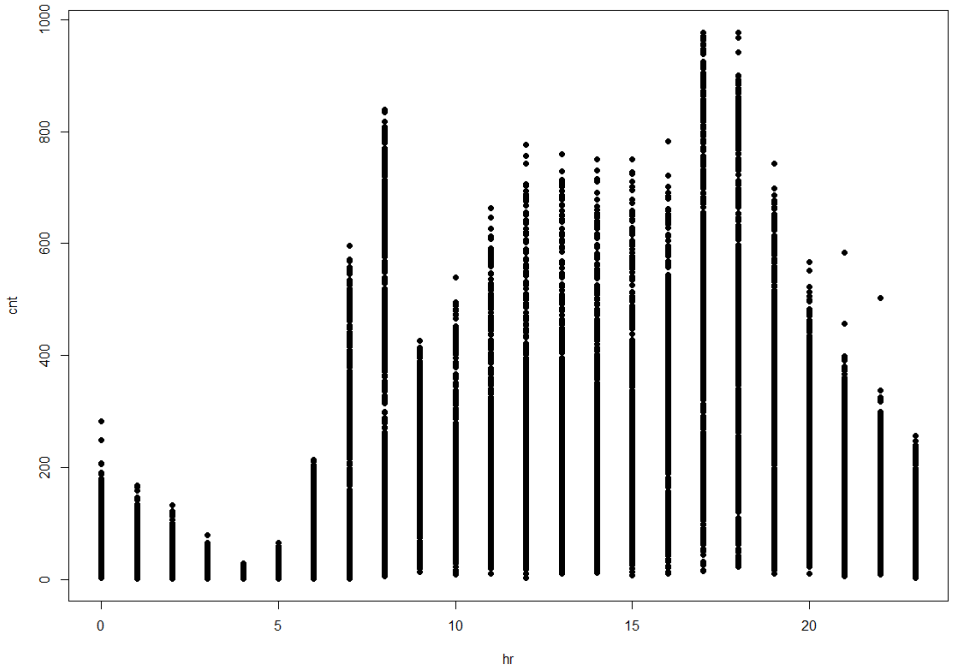


Figure 10. Hour vs total users

Filtering the dataframe **Hour** by using the code at *Filter weekend*, we observe a very similar pattern, but the peak at 8am, 5pm and 6pm are not ocurring. This explains that the peaks occur because of the working starting and ending hours which does not happen in the weekends for most of the businesses. There is no clear peak but the mayority of users rent bikes between 12am and 5pm similarly to **Figure 10**.

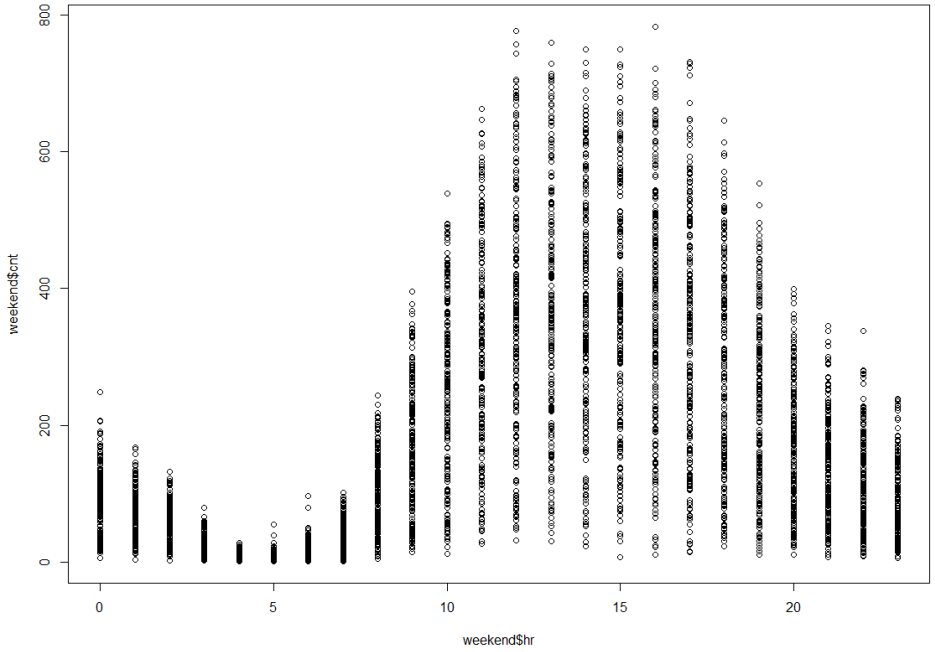


Figure 11. Hours vs total users only for weekends

4.2. SIMPLE LINEAR REGRESSION

The first method that will be applied is the simple linear regression. Linear regression is not always the best method for prediction, since it is a very low complex and low flexible method with low variance but high bias. However, linear regression can be useful to better understand the behavior of the data explored since as it is a low flexible method, the models resulting from it are not complex and therefore more interpretable.

This method will be applied two times since both **Day** and **Hour** dataset will be analyzed separately. Like that, we will try to build a linear regression model that may fit the data and predict the response both for the daily and hourly case, since it could be interesting to see how different the models look and perform when it comes to test and train error. The

response that we look at is the variable **cnt** in both cases, which is the total amount of users that have rented a bike at that day or hour.

There are two things to consider before comparing how well the linear models perform in both datasets. The first thing to consider is that **Hour** has much more rows than **Day**, and this may affect the quality of the fit and prediction playing in favor of **Hour** dataset. The second thing to bear in mind is that we have already seen in the previous section that the response **cnt** does not seem to follow a normal distribution in the case of **Hour**, so linear regression may not be the best method, playing in favor of **Day** dataset this time.

4.2.1. SIMPLE LINEAR REGRESSION ON THE DAY DATASET

As seen in the previous section, the features that most seem to have more correlation with the response variable **cnt** is the environment temperature (**temp**) and the feeling temperature (**atemp**). These two are highly correlated because **atemp** is calculated based on **temp**. Therefore, only **atemp** will be analyzed, since we think that people react more to the felt temperature rather than to the temperature shown in the thermometers. If we look at **Figure 12** it seems that there may be a linear relation between **atemp** and **cnt**, but if we look at the results from the model shown in **Figure 13** and commented in the following lines a linear model performs poorly when predicting the response variable in new data. The data has been divided into training and test data sets approximately 80% for training and 20% for testing. Code in *Training and testing dataset 80/20.*

In **Figure 12** the relationship between **atemp** and **cnt** is shown together with two lines. The black line represents the tendency line or best approximation line to the data. We can see that in the beginning is a straight line but, in the end, it has a bit of curvature. The red line is the resulting line from the simple linear model. The summary for this model is shown in **Figure 13**. We see here that the p value is low enough and F statistics high enough, which means that we can reject the null hypothesis of non-relation leading us to conclude that there is a relationship between **atemp** and **cnt**. The R squared though is not very good (0.4), so most of the data, 60%, cannot be explained by this model.

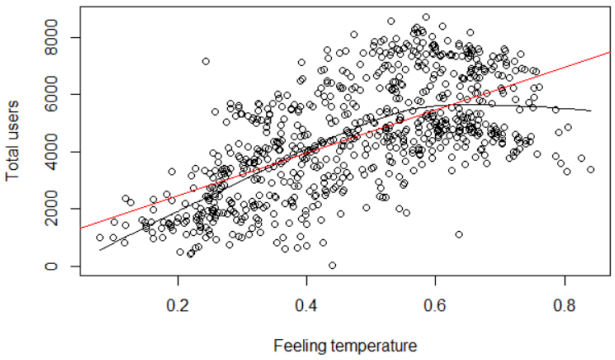


Figure 12. Feeling temperature vs total users. Red line: linear regression. Black line: trend line

```
Call:
lm(formula = cnt ~ atemp, data = train1)

Residuals:
    Min      1Q  Median      3Q     Max
-4604.6 -1086.0   -88.8  1054.5  4373.6

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    958.6      187.5   5.113 4.28e-07 ***
atemp         7491.1      373.7  20.044  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1496 on 598 degrees of freedom
Multiple R-squared:  0.4019,    Adjusted R-squared:  0.4009
F-statistic: 401.8 on 1 and 598 DF,  p-value: < 2.2e-16
```

Figure 13. Result from linear regression with feeling temperature

The red line **Figure 14** represents the line that the data points should have to fit if the predicted values where exactly the same as the real values. We can see that the predicted data points using the linear model are quite disperse from this ideal line, so we cannot say that the prediction is good. In **Figure 15** the residuals are plotted against the fitted values by the model. What we can conclude from here is that as the residuals are not equally distributed from the red horizontal line, the homoscedasticity property is not satisfied. This means that the variance of the residuals is not the same for every fitted value. The values for the residuals are also quite high, which means that the pre-assumption of linearity between **atemp** and **cnt** is not satisfied.

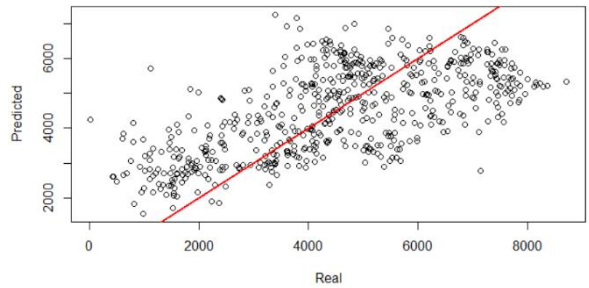


Figure 14. Real vs predicted values

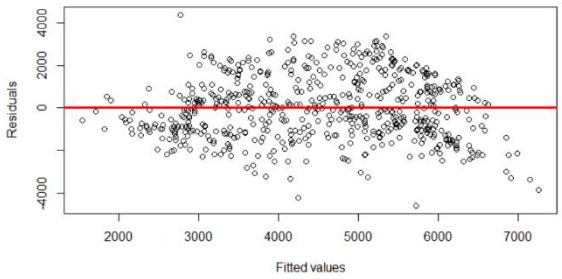


Figure 15. Fitted values vs residuals

Regarding the normality of the residuals, the Quantile-Quantile plot in **Figure 16** shows that the standardized residuals are normally distributed, since the Q-Q line is close to 45-degree straight line. The parameters of the normal distribution though are not the ones corresponding to the standard normal distribution, since the line does not cross the (0,0) coordinate point.

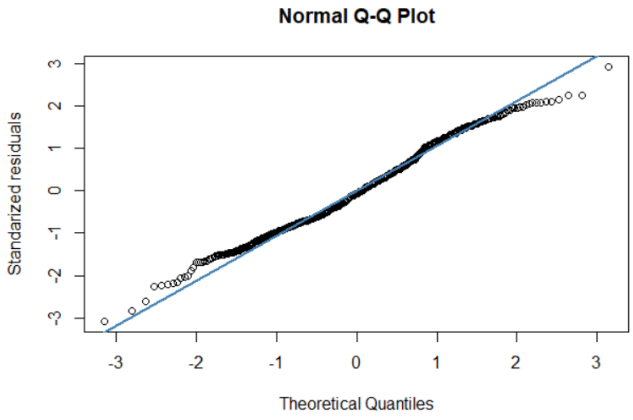


Figure 16. Normal quantile-quantile plot for residuals

We can conclude from the previous that the simple linear model with the feature **atemp** is not very appropriate for accurate prediction of total users, but if we look at the next **Figure 17** the predicted values in the test data follow a similar pattern compared to the original test data. However, it does not predict well the two peak values of 6000 and 8000 corresponding to the difference between years 2011 (black) and 2012 (red), it takes an average 7000 peak value for both years instead.

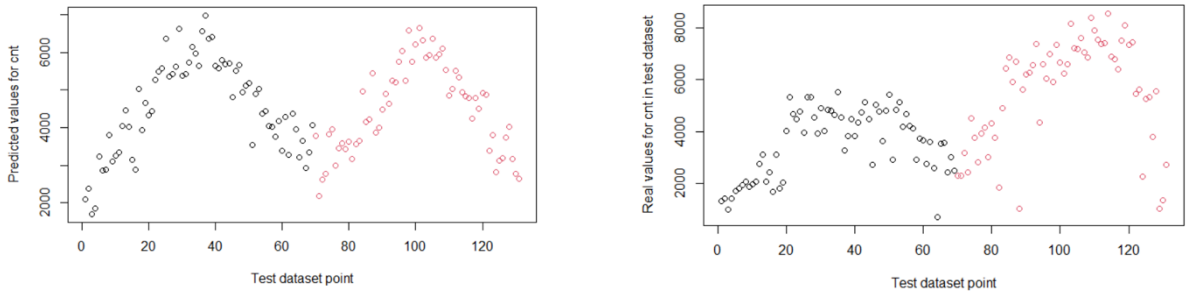


Figure 17. Right: Pattern for predicted values of total users. Left: Pattern for real values of total users. Black:2011 & Red: 2012

As the linear model does not give good prediction, we have tried other models like quadratic, cubic, logarithmic and squared root. All the code for the linear model and these mentioned models is shown in *Linear, quadratic, cubic, logarithmic, and squared root models in Day*. The results for train and test RMSE (Root Mean Squared Error) of all the mentioned models is shown in **Table 2**. To seeds have been used for the random train and test data split. The table shows that for both random splits of the train and test datasets the model that gives both the minimum train and test error is the cubic model. A 4^{th} degree polynomial was also tried to see if this would improve the results compared to the cubic model. Both train and test RMSE where lower for the 4^{th} degree polynomial but the difference was negligible.

Table 2. Train and test RMSE for different models using two seeds

Model	Train RMSE		Test RMSE	
	Seed 1	Seed 2	Seed 1	Seed 2
Linear	1493.91	1490.02	1537.115	1554.298
Quadratic	1429.056	1421.611	1473.798	1506.417
Cubic	1412.795	1398.874	1433.13	1492.747
Logarithmic	1457.177	1452.659	1508.926	1528.903
Squared error	1467.769	1462.947	1512.845	1534.087

```
Call:
lm(formula = cnt ~ poly(atemp, 3, raw = TRUE), data = train1)

Residuals:
    Min      1Q  Median      3Q     Max
-4727.6  -993.0  -116.8  1075.6  4817.3

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                1306.6      893.8   1.462  0.14432
poly(atemp, 3, raw = TRUE)1  -4873.7     6575.5  -0.741  0.45887
poly(atemp, 3, raw = TRUE)2  49025.9    14992.5   3.270  0.00114 **
poly(atemp, 3, raw = TRUE)3 -47403.9    10726.1  -4.419 1.18e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1404 on 596 degrees of freedom
Multiple R-squared:  0.4729,    Adjusted R-squared:  0.4702
F-statistic: 178.2 on 3 and 596 DF,  p-value: < 2.2e-16
```

Figure 19. Summary from cubic model with feeling temperature

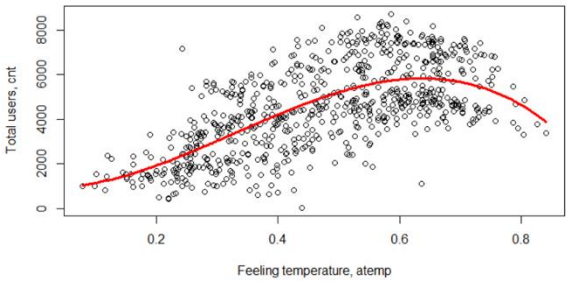


Figure 18. Feeling temperature vs total users. Red line: resulting cubic model

In conclusion, the model that best (but not good enough) fits and predicts the data regarding the feature **atemp** is the cubic model. The summary for this model is shown in **Figure 18**. Something to note from the summary is that the second- and third-degree variables from the model are marked as statistically relevant, but not the first-degree variable. This means that adding the second- and third-degree variables in the polynomial model does improve the result. Also, the R-squared value is higher compared to the first-degree linear model in **Figure 13,** which means that the model better explains the original data**.** We can see this in **Figure 19** where the model better captures the data compared to the red line in **Figure 12**. Equation 1 shows the resulting cubic model.

Equation 1. Resulting model from cubic regression for total users with feeling temperature

$$Cnt = 1306{,}6 - 4873{,}7 * atemp + 49025{,}9 * atemp^2 - 47403{,}9 * atemp^3$$

4.2.2. SIMPLE LINEAR REGRESSION ON THE HOUR DATASET

Like the previous dataset, the features that most seem to correlate with the response variable **cnt** are the environment temperature (**temp**) and the feeling temperature (**atemp**), in a lower

level compared to **Day**. Following the same reasoning, only **atemp** will be analyzed. The data has been divided into training and test data sets approximately 80% for training and 20% for testing as showed in *Training and testing dataset 80/20* but with the hour dataset.

If we look at **Figure 20**, where the relationship between **atemp** and **cnt** is shown, we can already say that there is no linear relationship, so it does not make sense to build a simple linear model. Compared to the **atemp** vs **cnt** graph in Day, the feeling temperature is discretized. We think that the reason for this may be that as the data is collected from hour to hour, there is more possibility of having repeated values of the feeling temperature compared to if it is collected daily. The result of applying simple linear regression with **atemp** is shown in **Figure 21** where only a 0.16 R squared is achieved.

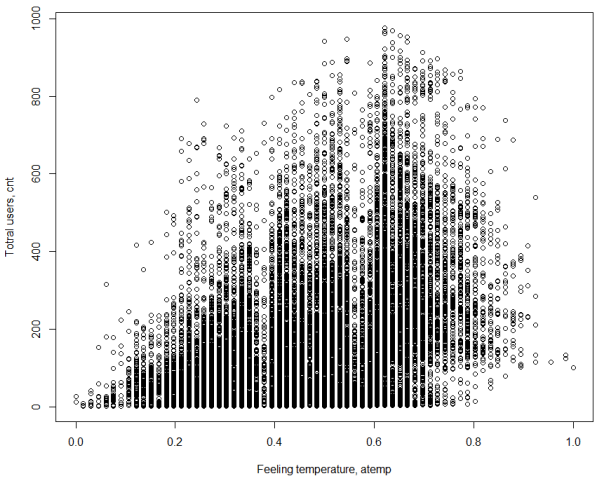


Figure 20. Feeling temperature cs total users in Hour

```
Call:
lm(formula = cnt ~ atemp, data = train2)

Residuals:
    Min      1Q  Median      3Q     Max
-311.61 -109.58  -33.41   76.64  744.31

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   -9.578      4.250  -2.253   0.0242 *
atemp        416.532      8.389  49.650   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 165.1 on 12998 degrees of freedom
Multiple R-squared:  0.1594,    Adjusted R-squared:  0.1594
F-statistic:  2465 on 1 and 12998 DF,  p-value: < 2.2e-16
```

Figure 21. Summary from simple linear regression with feeling temperature

Just for curiosity, we wanted to check what happens if we filter hour by hour, and there are three scenarios that we have seen. The code for this part is available at *Filtering by hour*. **Figure 22** plots the feeling temperature and total users only when the hour is 1 am. We can distinguish two groups of data points, and after various trials we discovered that these groups are defined by the factor **weekday**. Looking at **Figure 23** we can see that the pattern for rentals is different for days 6 and 0, which correspond to Saturday and Sunday (yellow and black). The reason for this can be that people tend to make more late-night plans during the weekend than during the weekdays. What we can read from **Figure 23** is that at late night hours, like 1am, the feeling temperature affects more to total users on Saturday and Sundays compared to days between Monday and Friday.

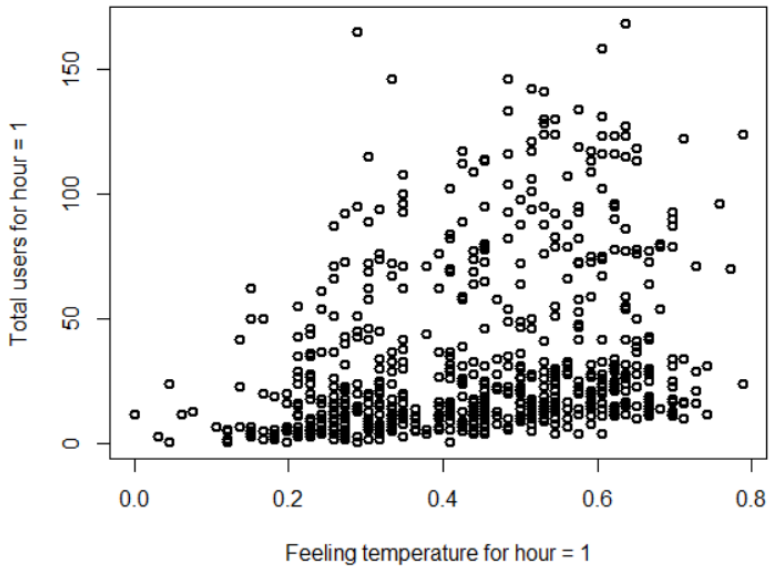


Figure 22. Feeling temperature vs total users. Hour = 1

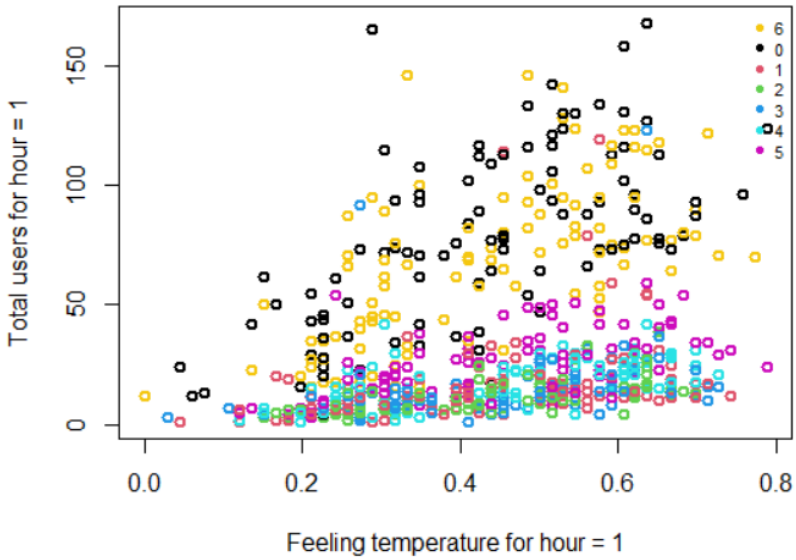


Figure 23. Feeling temperature vs total users. Hour = 1. By day of the week

If we now look at **Figure 25** instead, for early mornings the pattern changes. The pattern is still defined by the weekday factor, but on Saturdays and Sundays we see less users in early mornig hours, like 8am, compared to number of user during the week. This makes sense, since it is during the weekdays that people need to wake up early to go to work. Finally, in the evenings, like around 17.00, there is no clear distinction between weekends or weekdays, as we can see in **Figure 24**, so we can say that users are similarly affecet by the feeling temperature on the weekend and weekdays.

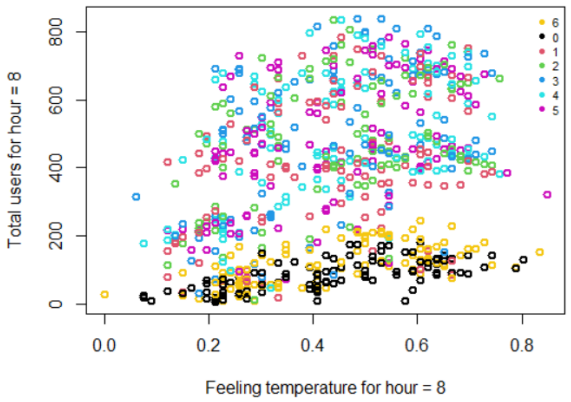


Figure 25. Feeling temperature vs total users. Hour = 8. By day of the week

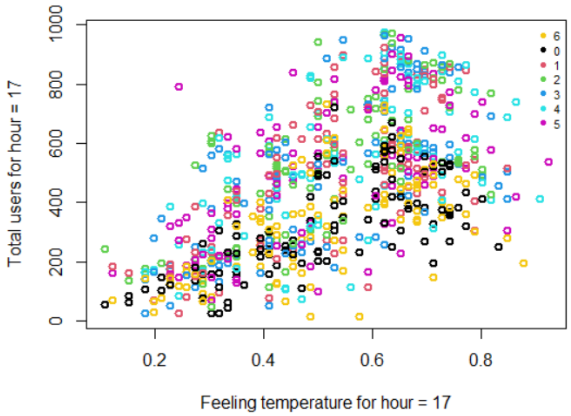


Figure 24. Feeling temperature vs total users. Hour = 17. By day of the week

It could be a possibility to apply simple linear regression for these different groups, so for each hour from 0 to 23 and for weekend and weekdays if there was any distinction, because it seems that there could be a linear relationship between **atemp** and **cnt** in those subgroups. For example, in **Figure 23** and **Figure 25** we can see that the data could be explained by 2 linear regressions, one for weekends and another one for weekdays. So for each hour with this similar pattern we would have 2 linear regressions. For the case of hours behaving similar to **Figure 24,** the data could be explained by one linear regression for each hour. However, we think that having more than 20 linear regression models is not the best solution when there are other options such as multiple linear regression that already take into consideration if it is weekday or weekend, if it is 8am or 8pm, etc. One clear conclusion from this is that the features **hr** and **weekday** are important for the response **cnt**, so a multiple linear regression model would be a better solution than several simple linear regressions. We show this in the next section.

4.3. MULTIPLE LINEAR REGRESSION

As seen in the previous subsection, simple linear regression with **atemp**, which is the feature that most seemed to have a linear relationship with the response variable **cnt,** does not give good results regarding prediction accuracy, neither regarding the fitting. We can see this in **Table 2** where both the train and test root mean squared error are shown to be very high. For the **Hour** dataset it has not even been applied since the R2 is very far from 1, which gives a hint on how bad the model will predict values. In both datasets, simple linear regression is not accurate enough for prediction. However, it was useful to better understand the pattern that our data follows due to easy interpretability of the model.

In order to improve the model's fit and prediction accuracy, multiple linear is applied. Multiple linear regression follows the same logic as simple linear regression but more than just one feature is incorporated into the model. Our first impression is that the models can have better results for both the R2 and the train RMSE since the more features you consider, the more information you are given to better fit the real data. We are aware however that this could also lead to overfitting, giving a poorer prediction on unforeseen data and thus a higher test RMSE. Comparison of results are commented in a later section.

When applying multiple linear regression, one of the first things is to decide how many features will be added to the model, and which of them. The number of features is important since a greater number of them would add complexity to the interpretation of the model, but if not enough are added the accuracy of the model may not be good. This could be done by trial and error, or by trying all possible combinations for each number of features. This is very time consuming, but there are some methods that help us in this process of "feature selection engineering" that can easily be applied in R. In this project we have used best subset selection, backward and forward stepwise selection, validation set approach and k-fold cross validation using 10 folds. Each of these methods apply the feature selection in a different way, so each of them gives different solutions both for the size of the model with minimum error and for the features to consider for the same size.

For the case of best subset, backward and forward stepwise selection **Figure 26, Figure 27, Figure 28, Figure 29, Figure 30** and **Figure 31** show how different criteria selects the "best" model size. These different criteria are the maximization of the R-squared (RSQ) and adjusted the R-squared, and minimization of CP (Mallow's Cp) and BIC (Bayesian Information Criterion). Each of these criteria penalizes differently for each added feature. The results are summarized in **Table 3** for Day and **Table 4** for Hour**.**

For validation set approach and k-fold cross validation methods, which directly estimate the test error, Error! Reference source not found., Error! Reference source not found.**,** Error! Reference source not found. and Error! Reference source not found. depict how the error varies while adding more features to the model. Further comments on the specific figures are commented in the following lines.

All the code for this part is available from page *55*.

4.3.1. FEATURE SELECTION FOR DAY DATASET

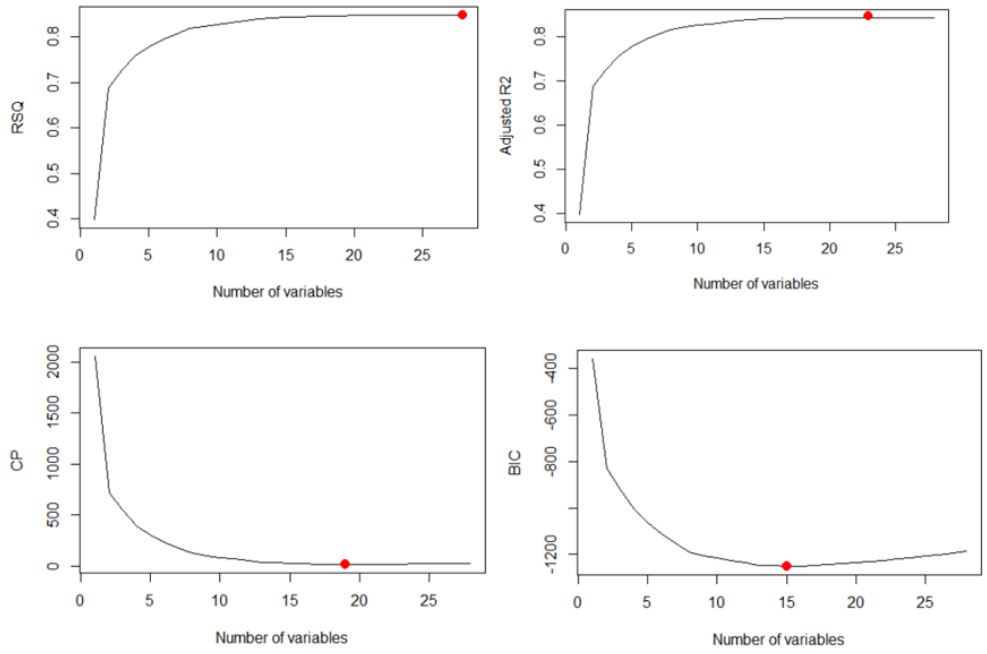


Figure 26. Maximum R2 and Adjusted R2 & minimum Cp and Bic for best subset selection in Day

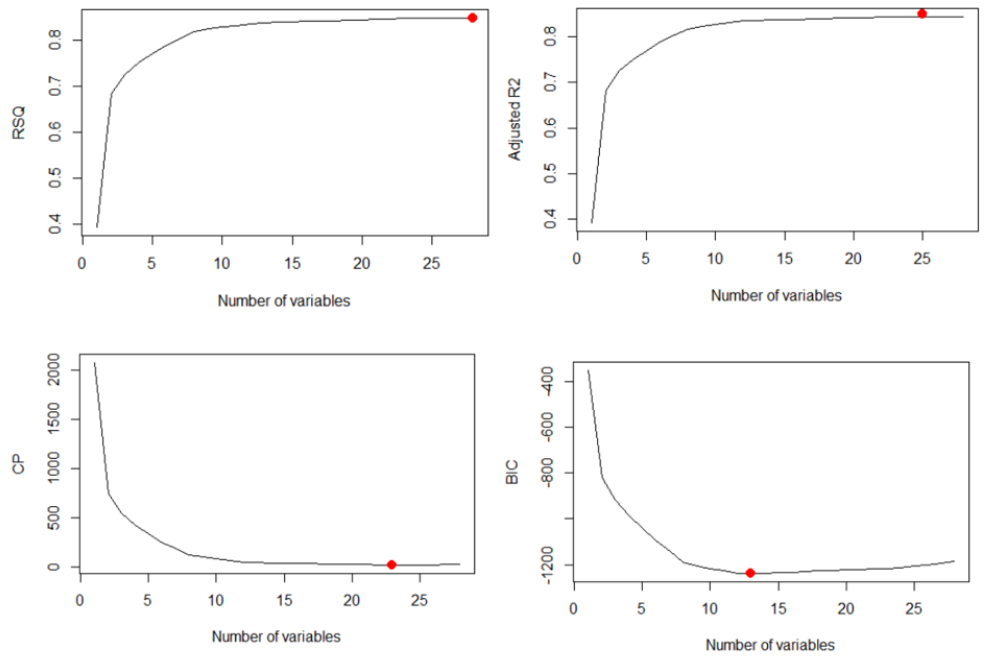


Figure 27. Maximum R2 and Adjusted R2 & minimum Cp and Bic for backward stepwise selection in Day

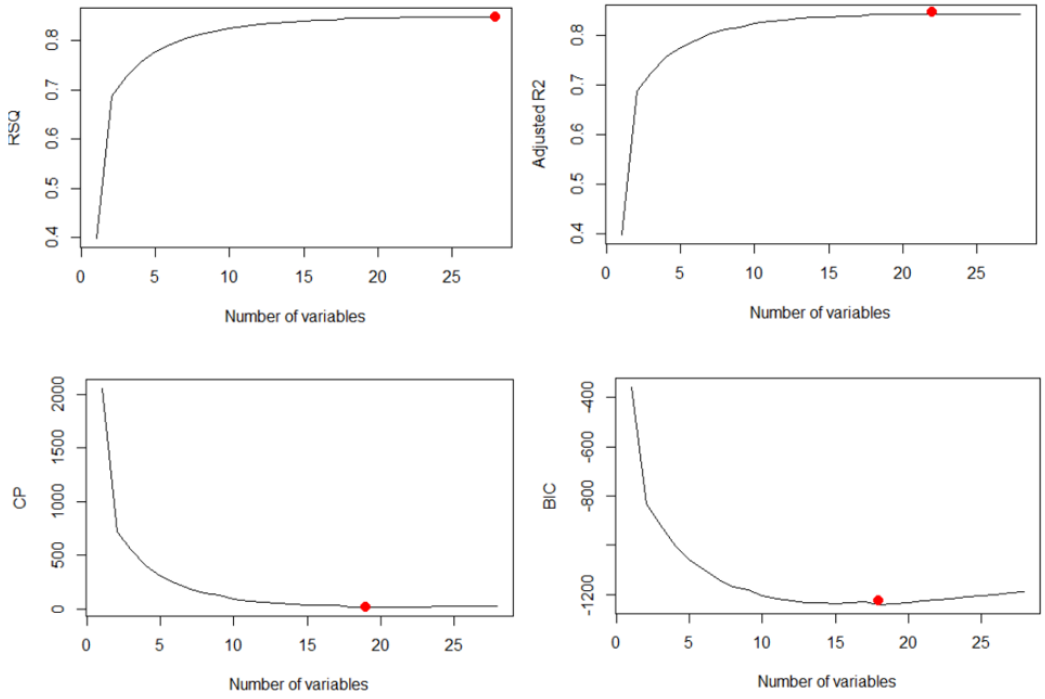


Figure 28. Maximum R2 and Adjusted R2 & minimum Cp and Bic for forward stepwise selection in Day

Table 3. Summary of selected model size for each feature selection method in Day

Method	R2	Adjusted R2	CP	BIC
Best subset	28	23	19	15
Backward	28	25	23	13
Forward	28	22	19	18

Looking at each row, which corresponds to the first three methods for feature selection, we can see that the criteria that most restricts the amount of features for the model is the Bayesian information criterion. This means that this criterion penalizes more for each extra feature incorporated. The key question here is which model size to select. The first thing to decide is which criteria is going to be used, that is, how restrictive we want to be with the number of features. We keep in mind that the higher the number of features, the more complex (or less interpretable) the model becomes, which means that even the train error is lower, the test error can be higher. Let's say we decide to be restrictive and select the minimum BIC. The second decision to be made is, do we take 15, 13 or 18 features? This are the numbers that each of the three methods give as a result. If we are looking for the minimum test error, then we would have to take each model and apply it for the test data and see how the test RMSE results. The

model with the minimum test RMSE would be the most adequate one for prediction. Same explanation applies to **Table 4**.

As mentioned before, not only these three methods have been used for feature selection but also the validation set approach and k-fold cross validation. These two methods are known as cross validation resampling methods that are used for model assessment and they estimate the test error in a les underestimated way than subset selection methods. However, these methods can also be used for feature selection, based on the test error that each model size gives. We can see this in Error! Reference source not found. and Error! Reference source not found.. For the case of validation set approach, the model size which gives the minimum test error is 16. For the k-fold cross validation method, this minimum test error is achieved with 24. However, it is also something to note that, for example in Error! Reference source not found. even if the minimum test error is achieved with 16 predictors, the difference in this test error may be negligible if we compare to the test error with 2 predictors**,** as it seems that from the second added feature the error does not vary so much.

4.3.2. FEATURE SELECTION FOR HOUR DATASET

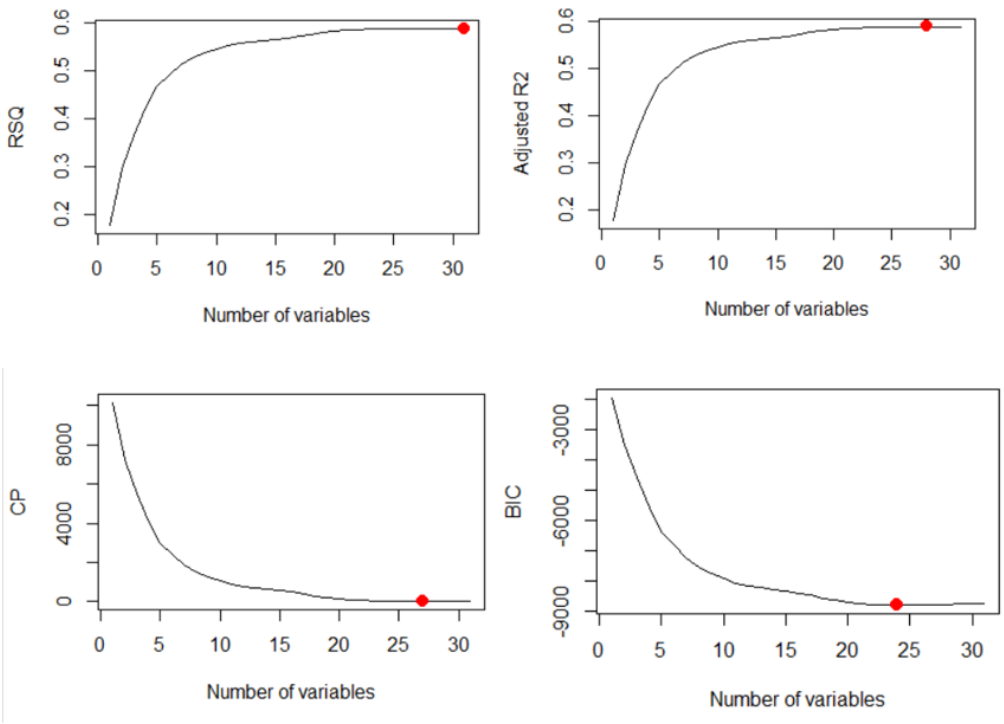


Figure 29. Maximum R2 and Adjusted R2 & minimum Cp and Bic for best subset selection in Hour

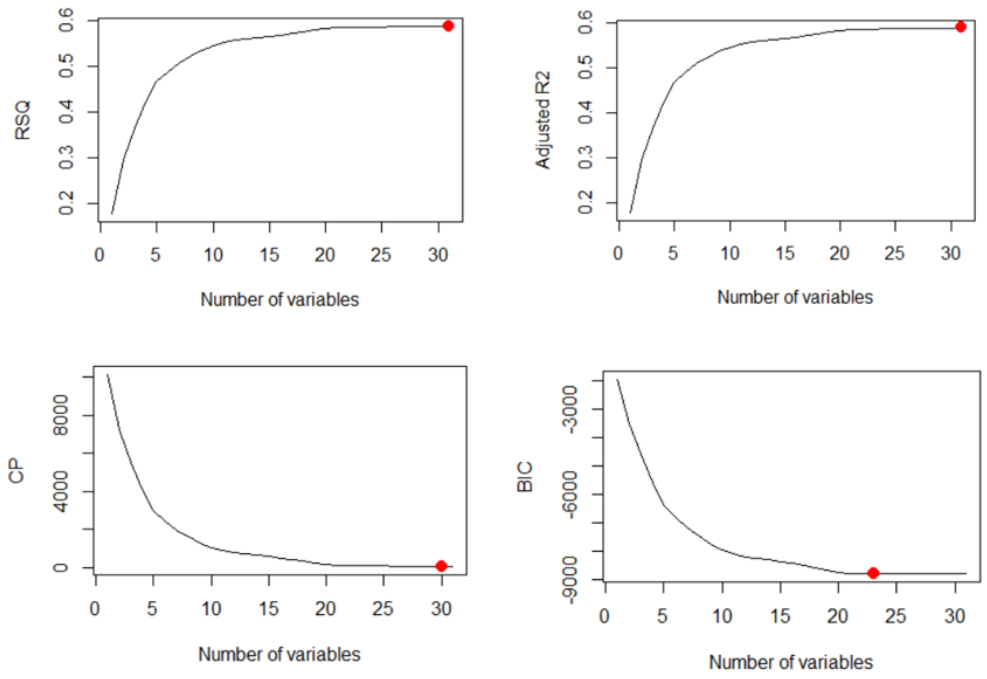


Figure 30. Maximum R2 and Adjusted R2 & minimum Cp and Bic for backward stepwise selection in Hour

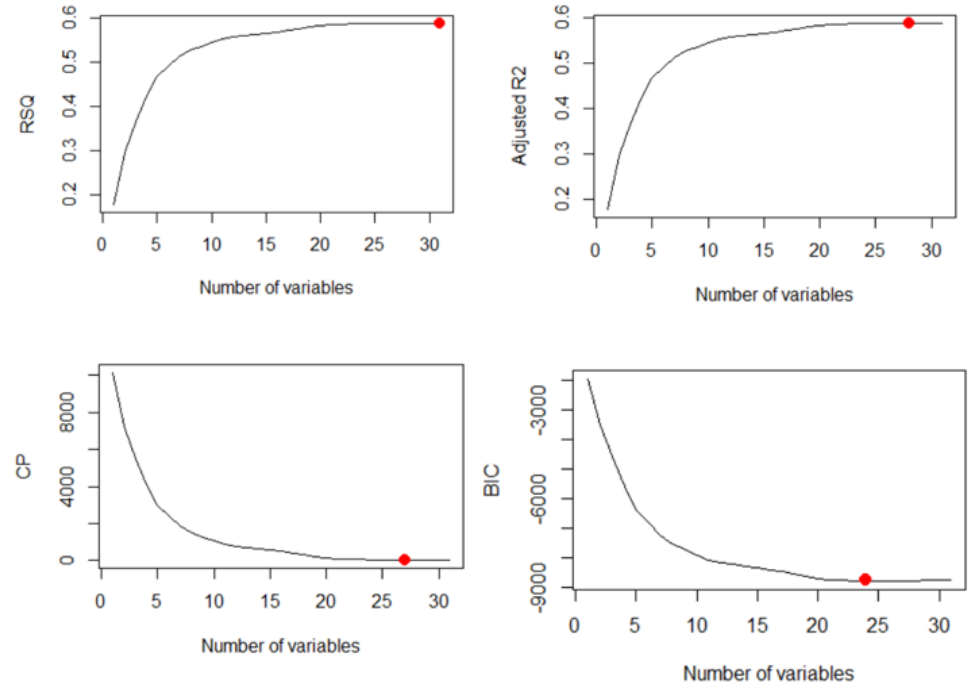


Figure 31. Maximum R2 and Adjusted R2 & minimum Cp and Bic for forward stepwise selection in Hour

Table 4. Summary of selected model size for each feature selection method in Hour

Method	R2	Adjusted R2	CP	BIC
Best subset	31	28	27	24
Backward	31	31	30	23
Forward	31	28	27	24

The same explanation applies to this table compared to the one in the **Day** dataset. However, if we look at the results from validation set approach and k-fold cross validation**,** Error! Reference source not found. and Error! Reference source not found. have a different pattern in **Hour**. There is a first lower peak of the test error in both figures when the feature amount is 9, but when increasing the features, the test error increases again.

4.3.3. MODEL ASSESMENT OF THE DAY DATASET

After the feature selection process is completed, we want to figure out which model delivers the best results by using so called *resampling methods*. The process of using these methods to evaluate the performance of a model is called model assessment. Basically, resampling methods continuously draw samples from the data set and a execute the given model on the picked data. In general, they acquire high computational effort, but due to sharp technological developments during the past decades, this sort of model evaluation can be easily applied these days (James, et al., 2021).

For this project it was decided to use validation set approach as well as K-fold cross validation for both data sets to assess the performance of the models. As a feature selection input serve the results of the best subset-, forward stepwise and backwards stepwise selection. An overview of the calculated test errors is given at the end of this chapter.

1. *Validation Set approach*

Best Subset Selection:

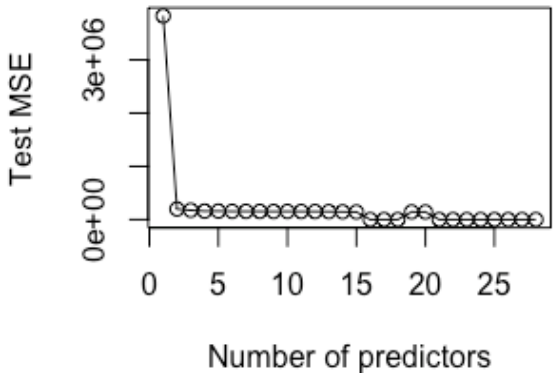


Figure 32: Validation set approach on the best subset selection of the day data set

Number of Predictors for minimum test error: 16

Minimum test error: 2.689003e-21

Forward Stepwise Selection:

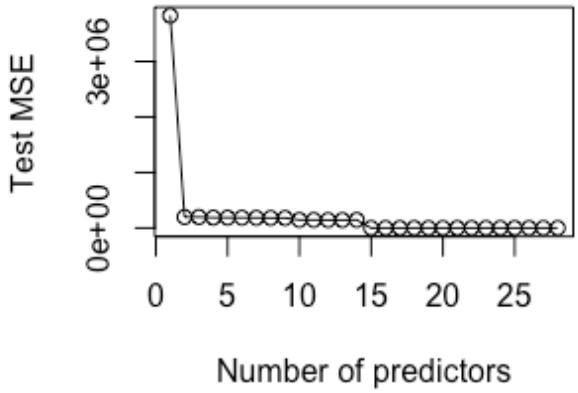


Figure 33: Validation set approach on the forward stepwise selection of the day data set

Number of Predictors for minimum test error: 16

Minimum test error: 1.933308e-23

Backward Stepwise Selection:

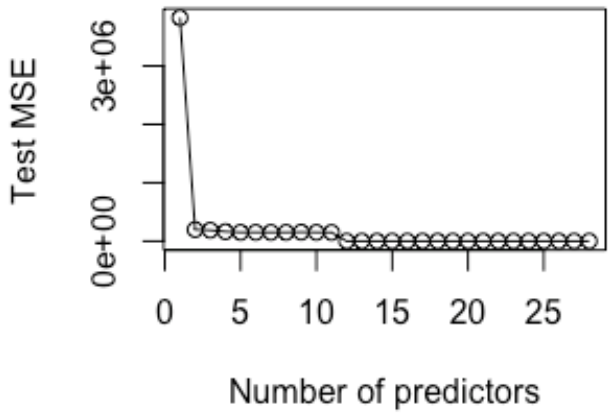


Figure 34: Validation set approach on the backward stepwise selection of the day data set

Number of Predictors for minimum test error: 12

Minimum test error: 1.862143e-23

2. *K-fold cross validation*

Best Subset Selection:

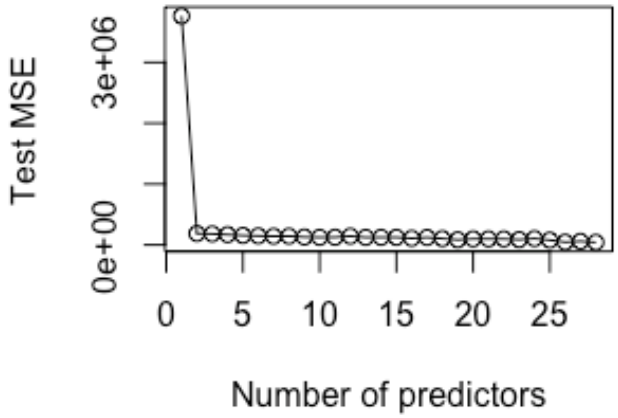


Figure 35: K-fold cross validation (K = 10) on the best subset selection of the day data set

Number of Predictors for minimum test error: 28

Minimum test error: 42243.98

Forward Stepwise Selection:

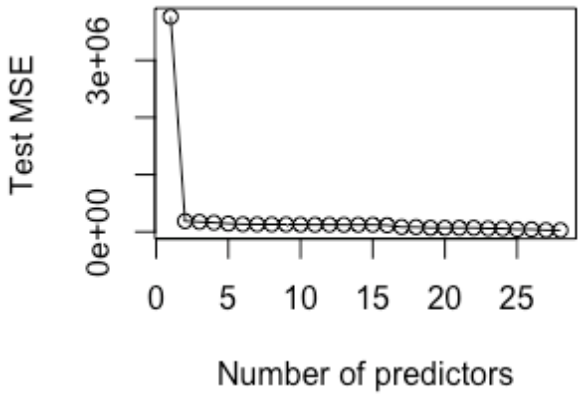


Figure 36: K-fold cross validation on forward stepwise selection of the day data set

Number of Predictors for minimum test error: 28

Minimum test error: 28321.86

Backward Stepwise Selection:

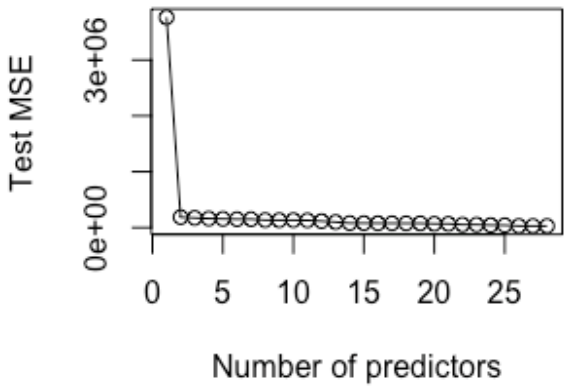


Figure 37: K-fold cross validation on the backward stepwise selection of the day data set

Number of Predictors for minimum test error: 28

Minimum test error: 28362.11

The following Table summarizes the test errors as well as the number of necessary features, which were calculated by applying validation set approach and K-fold cross validation. It is shown that the lowest test error as well as the minimum number of features can be achieved by using the backward stepwise feature selection.

Table 5: Overview of the calculated test errors for the model as well as the number of predictors of the day data set, which should be included

	Validation Set approach	**K-fold cross validation**
Best Subset Selection	2.689003e-21 (16)	42243.98 (28)
Forward stepwise selection	1.933308e-23 (16)	28321.86 (28)
Backward stepwise selection	1.862143e-23 (12)	28362.11 (28)

4.3.4. MODEL ASSESMENT FOR THE HOUR DATASET

Now where the minimal test error is calculated for the day dataset, the same procedure is executed with the hour dataset. Analog to the proceedings in the previous section again the validation-set-approach and K-fold cross validation with K = 10 is applied.

3. *Validation Set approach*

Based on best subset selection:

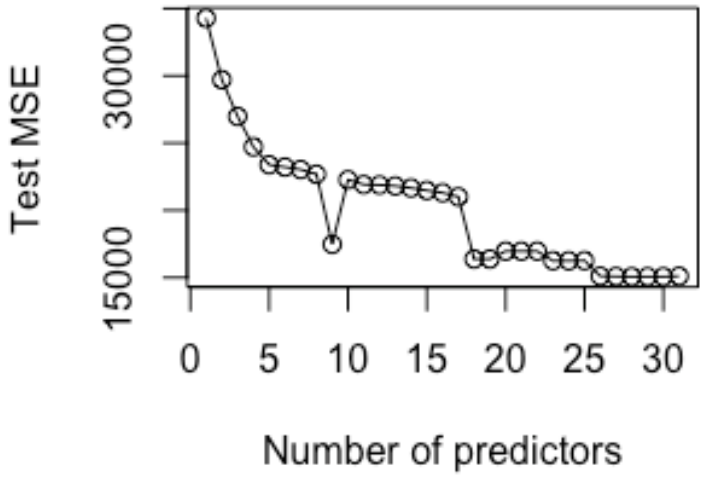


Figure 38: Validation set approach on the best subset selection of the hour data set

Number of Predictors for minimum test error: 28

Minimum test error: 15066.93

Based on stepwise forward selection:

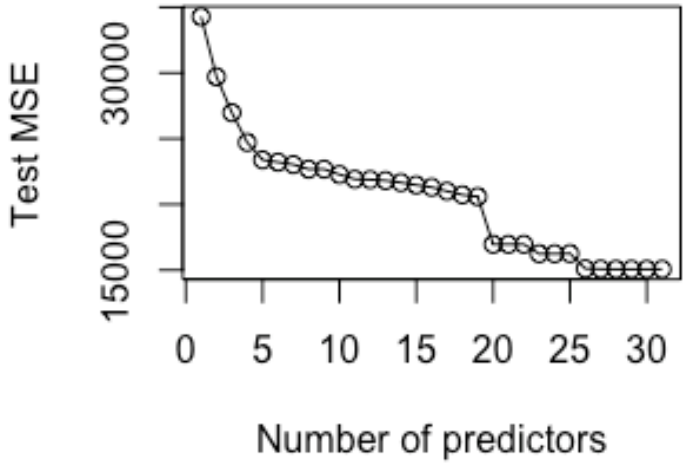


Figure 39: Validation step approach on the stepwise forward selection of the hour data set

Number of Predictors for minimum test error: 28

Minimum test error: 15066.93

Based on stepwise backward selection:

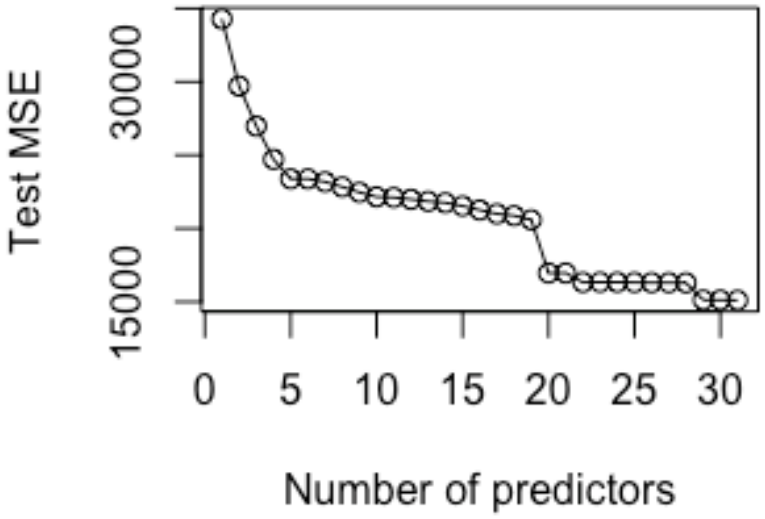


Figure 40: Validation set approach on the stepwise backward selection of the hour data set

Number of Predictors for minimum test error: 31

Minimum test error: 15105.78

4. K-Fold cross validation (K=10)

In the next step a K-fold cross validation is executed, whereby best subset-, forward stepwise and backward stepwise served as a base for the feature selection.

Best Subset selection:

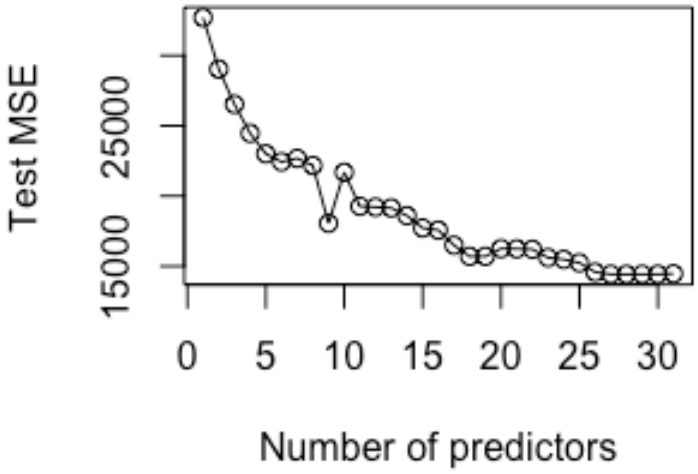


Figure 41: K-fold cross validation (K=10) on the best subset selection of the hour data set

Number of Predictors for minimum test error: 28

Minimum test error: 14421.99

Forward stepwise selection:

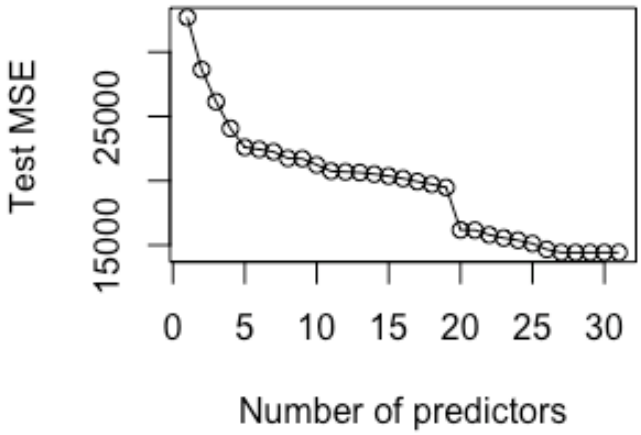


Figure 42: K-fold cross validation on the forward stepwise selection of the hour data set

Number of Predictors for minimum test error: 28

Minimum test error: 14421.99

Backward stepwise selection:

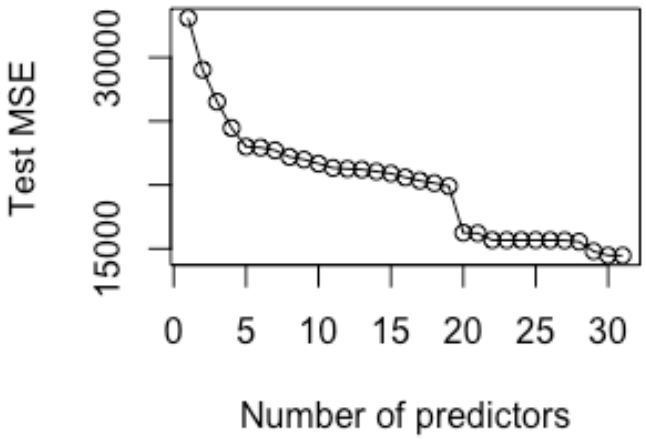


Figure 43: K-fold cross validation on the backward stepwise selection of the hour data set

Number of Predictors for minimum test error: 31

Minimum test error: 14465.09

In the following table an overview of the calculated test errors as well as the number of necessary predictors is given. It is shown that best-subset selection and the forward stepwise

selection give the same test error for both resampling methods. Furthermore, these two methods give a lower test error as well as a lower number of necessary predictors compared to the backward stepwise selection.

Table 6: Overview of the calculated test errors for the model as well as the number of predictors of the hour data set, which should be included

	Validation Set approach	**K-fold cross validation**
Best Subset Selection	15066.93 (28)	14421.99 (28)
Forward stepwise selection	15066.93 (28)	14421.99 (28)
Backward stepwise selection	15105.78 (31)	14465.09 (31)

4.4. POISSON'S REGRESSION

Poisson regression is a generalized linear model form of regression analysis. It is used to model count data. For it to be applied it is assumed that the response variable has a Poisson distribution. It is also assumed in case of a Poisson regression that the variance is equal with the mean. As stated earlier in the report, the distribution of total users is a Poisson one, therefore ana analysis will be done both on day and hour datasets. To see if the variance and the mean are equal ana analysis was done which is shown in **Table 8** - Overdispersion Day for **Day** dataset and **Table 7** - Overdispersion Hour for **Hour** dataset.

Table 8 - Overdispersion Day

```
Overdispersion test Obs.Var/Theor.Var Statistic
       poisson data          833.1478  608197.9

Overdispersion test p-value
       poisson data       0
```

Table 7 - Overdispersion Hour

```
Overdispersion test Obs.Var/Theor.Var
       poisson data          173.6563

Overdispersion test Statistic p-value
       poisson data   3017799       0
```

From the p-value we can see that there is an overdispersion in both tables which means that Poisson regression is not the best option to predict the total number of users. Still, for the purpose of this report the regression will be done.

Table 10 - Summary Poisson Regression Day

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-69.494   -7.420    0.756    7.888   56.616

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -9.983e+00  4.666e-02 -213.97   <2e-16 ***
dteday       1.348e-08  3.491e-11  386.19   <2e-16 ***
season2      2.784e-01  2.274e-03  122.42   <2e-16 ***
season3      8.401e-02  2.846e-03   29.51   <2e-16 ***
season4      1.997e-01  2.067e-03   96.62   <2e-16 ***
holiday1    -1.707e-01  3.729e-03  -45.79   <2e-16 ***
workingday1  2.841e-02  1.237e-03   22.96   <2e-16 ***
weathersit2 -9.016e-02  1.498e-03  -60.19   <2e-16 ***
weathersit3 -7.178e-01  5.503e-03 -130.45   <2e-16 ***
temp         7.091e-01  2.357e-02   30.09   <2e-16 ***
atemp        6.425e-01  2.597e-02   24.74   <2e-16 ***
hum         -3.496e-01  5.554e-03  -62.95   <2e-16 ***
windspeed   -5.494e-01  8.144e-03  -67.46   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 668801  on 730  degrees of freedom
Residual deviance: 143922  on 718  degrees of freedom
AIC: 151346

Number of Fisher Scoring iterations: 4
```

Table 9 - Summary Poisson Regression Hour

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-24.888   -4.548   -1.131    3.100   27.788

Coefficients:
              Estimate Std. Error  z value Pr(>|z|)
(Intercept)  3.353991   0.006154  544.988  < 2e-16 ***
hr1         -0.462656   0.008182  -56.545  < 2e-16 ***
hr2         -0.830273   0.009313  -89.154  < 2e-16 ***
hr3         -1.493086   0.012163 -122.760  < 2e-16 ***
hr4         -2.091914   0.015857 -131.923  < 2e-16 ***
hr5         -0.938692   0.009785  -95.930  < 2e-16 ***
hr6          0.414774   0.006616   62.695  < 2e-16 ***
hr7          1.431091   0.005664  252.668  < 2e-16 ***
hr8          1.917330   0.005423  353.561  < 2e-16 ***
hr9          1.380268   0.005648  244.368  < 2e-16 ***
hr10         1.100078   0.005804  189.546  < 2e-16 ***
hr11         1.235068   0.005708  216.379  < 2e-16 ***
hr12         1.401991   0.005626  249.210  < 2e-16 ***
hr13         1.377070   0.005641  244.132  < 2e-16 ***
hr14         1.310421   0.005679  230.743  < 2e-16 ***
hr15         1.349403   0.005663  238.286  < 2e-16 ***
hr16         1.574208   0.005562  283.004  < 2e-16 ***
hr17         1.988757   0.005419  367.001  < 2e-16 ***
hr18         1.928944   0.005424  355.652  < 2e-16 ***
hr19         1.644814   0.005507  298.675  < 2e-16 ***
hr20         1.355291   0.005642  240.195  < 2e-16 ***
hr21         1.105998   0.005798  190.767  < 2e-16 ***
hr22         0.855694   0.006004  142.517  < 2e-16 ***
hr23         0.479234   0.006419   74.656  < 2e-16 ***
season2      0.316960   0.002223  142.581  < 2e-16 ***
season3      0.238896   0.002710   88.139  < 2e-16 ***
season4      0.454967   0.001958  232.384  < 2e-16 ***
holiday1    -0.157698   0.003725  -42.334  < 2e-16 ***
workingday1  0.025813   0.001237   20.859  < 2e-16 ***
weathersit2 -0.036946   0.001414  -26.125  < 2e-16 ***
weathersit3 -0.459637   0.002845 -161.557  < 2e-16 ***
weathersit4 -0.430809   0.067023   -6.428 1.29e-10 ***
temp         0.406274   0.019030   21.350  < 2e-16 ***
atemp        0.962882   0.020748   46.409  < 2e-16 ***
hum         -0.350530   0.003927  -89.259  < 2e-16 ***
windspeed   -0.156621   0.004824  -32.469  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 2891591  on 17378  degrees of freedom
Residual deviance:  760407  on 17343  degrees of freedom
AIC: 871380

Number of Fisher Scoring iterations: 5
```

Based on the analysis above, in Table 9 - Summary Poisson Regression Hour and **Table 10** - Summary Poisson Regression Day, we can see that the variables have a significant impact on the response variable. The null deviance shows how well the response variable is predicted, and we can see that there is a difference in the deviance in Table 10 - Summary Poisson Regression Day which indicates not the best fit. When we look at **Table 9** - Summary Poisson Regression Hour for the hour dataset, the difference is larger. Since there is no R Squared for this regression the pseudo–R Squared is calculated by dividing the residual deviance to the null variance and deduct from one the result. For the **Day** dataset is 0.78 and for the **Hour** dataset

Table 11 - Coefficients Day

(Intercept)	dteday	season2	season3	season4	holiday1	workingday1	weathersit2
4.619798e-05	1.000000e+00	1.321074e+00	1.087835e+00	1.221007e+00	8.450508e-01	1.028818e+00	9.157808e-01
weathersit3	temp	atemp	hum	windspeed			
4.878163e-01	2.032164e+00	1.901193e+00	7.049451e-01	5.773134e-01			

11 - Coefficients Day and **Table 12** - Coefficients Hour for the **Day** and **Hour** datasets.

For example, the number of users increases with 22.1 % in season four compared with season one. If we look at the **weathersit3** the number of users is decreasing with approximately 51.2 % (100 – 48.8) from the ones using the bikes in **weathersit1**.

Table 12 - Coefficients Hour

(Intercept)	dteday	season2	season3	season4	hr1
3.183097e-07	1.000000e+00	1.324197e+00	1.114961e+00	1.216586e+00	6.279855e-01
hr2	hr3	hr4	hr5	hr6	hr7
4.329449e-01	2.225782e-01	1.219242e-01	3.869476e-01	1.502870e+00	4.167471e+00
hr8	hr9	hr10	hr11	hr12	hr13
6.801845e+00	4.002435e+00	3.044256e+00	3.508678e+00	4.174329e+00	4.079334e+00
hr14	hr15	hr16	hr17	hr18	hr19
3.821883e+00	3.976298e+00	4.975699e+00	7.501389e+00	7.040592e+00	5.260470e+00
hr20	hr21	hr22	hr23	holiday1	workingday1
3.923358e+00	3.048282e+00	2.364304e+00	1.619231e+00	8.392193e-01	1.029261e+00
weathersit2	weathersit3	weathersit4	temp	atemp	hum
9.423608e-01	6.193310e-01	6.341230e-01	1.346413e+00	2.765688e+00	8.056713e-01
windspeed					
9.144637e-01					

Compared with the analysis for day dataset, in the above table we can see that, for example, in season four is an increase of 21.6 % instead of 22.1 %. Now we will use the model to predict the number of users, and a preview of it is shown in **Table 13** - Prediction Poisson Day for the day dataset.

Table 13 - Prediction Poisson Day

1	2	3	4	5	6	7	8	9	10	11
1773.7357	1771.0458	1746.9030	1770.0077	1898.9997	1923.2895	1660.6064	1434.4619	1472.3806	1659.9135	1554.5876
12	13	14	15	16	17	18	19	20	21	22
1560.9658	1619.2602	1779.9552	1725.1629	1849.7447	1292.9924	1544.2353	1715.6956	1767.2464	1617.0123	1550.1296
23	24	25	26	27	28	29	30	31	32	33
1529.5146	1643.1167	1722.5564	753.6986	1805.2297	1596.6755	1743.6899	1841.3141	1588.2445	1644.6151	1588.0498
34	35	36	37	38	39	40	41	42	43	44

We can see that in day 20 it is predicted to be approximately 1,767 total users.

Table 14 - Prediction Poisson Hour

1	2	3	4	5	6	7
22.478702	13.847146	9.546481	5.068557	2.776462	7.987391	33.138445
8	9	10	11	12	13	14
88.796937	154.891816	99.059275	78.525110	83.456626	111.832549	108.506386
15	16	17	18	19	20	21
101.793975	102.423133	124.102953	191.402375	114.380518	85.460895	95.143445
22	23	24	25	26	27	28
74.318065	56.625550	41.606397	25.725862	15.673787	10.412772	5.705321
29	30	31	32	33	34	35
3.125270	24.932503	104.172520	112.653285	97.666296	69.753835	80.675562
36	37	38	39	40	41	42
96.635672	97.326694	58.334191	58.923812	75.344263	188.536588	169.733397

In **Table 14** - Prediction Poisson Hour we can see the predictions for the hour data set. For example, in day 40 there should be around 75 users.

Since the variance and the mean are not the same as mentioned before, a Quasi-Poisson regression is made also. This one is a generalization of the Poisson regression.

The results are presented in Table 15 - Summary Quasi- Poisson Hour for the **Hour** and **Table 16** - Summary Quasi-Poisson Day for the **Day** datasets.

Table 15 - Summary Quasi- Poisson Hour

Table 16 - Summary Quasi-Poisson Day

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-69.494   -7.420    0.756    7.888   56.616

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -9.983e+00  6.349e-01 -15.723  < 2e-16 ***
dteday       1.348e-08  4.751e-10  28.379  < 2e-16 ***
season2      2.784e-01  3.095e-02   8.996  < 2e-16 ***
season3      8.401e-02  3.873e-02   2.169 0.030422 *
season4      1.997e-01  2.812e-02   7.100 3.00e-12 ***
holiday1    -1.707e-01  5.074e-02  -3.365 0.000807 ***
workingday1  2.841e-02  1.684e-02   1.687 0.091978 .
weathersit2 -9.016e-02  2.039e-02  -4.423 1.13e-05 ***
weathersit3 -7.178e-01  7.488e-02  -9.586  < 2e-16 ***
temp         7.091e-01  3.207e-01   2.211 0.027361 *
atemp        6.425e-01  3.533e-01   1.818 0.069439 .
hum         -3.496e-01  7.559e-02  -4.626 4.43e-06 ***
windspeed   -5.494e-01  1.108e-01  -4.957 8.93e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for quasipoisson family taken to be 185.1818)

    Null deviance: 668801  on 730  degrees of freedom
Residual deviance: 143922  on 718  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 4
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-24.888   -4.548   -1.131    3.100   27.788

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.353991   0.041007  81.792  < 2e-16 ***
hr1         -0.462656   0.054518  -8.486  < 2e-16 ***
hr2         -0.830273   0.062052 -13.380  < 2e-16 ***
hr3         -1.493086   0.081041 -18.424  < 2e-16 ***
hr4         -2.091914   0.105658 -19.799  < 2e-16 ***
hr5         -0.938692   0.065200 -14.397  < 2e-16 ***
hr6          0.414774   0.044081   9.409  < 2e-16 ***
hr7          1.431091   0.037739  37.920  < 2e-16 ***
hr8          1.917330   0.036134  53.062  < 2e-16 ***
hr9          1.380268   0.037635  36.675  < 2e-16 ***
hr10         1.100078   0.038671  28.447  < 2e-16 ***
hr11         1.235068   0.038032  32.474  < 2e-16 ***
hr12         1.401991   0.037485  37.401  < 2e-16 ***
hr13         1.377070   0.037585  36.639  < 2e-16 ***
hr14         1.310421   0.037841  34.630  < 2e-16 ***
hr15         1.349403   0.037733  35.762  < 2e-16 ***
hr16         1.574208   0.037064  42.473  < 2e-16 ***
hr17         1.988757   0.036107  55.079  < 2e-16 ***
hr18         1.928944   0.036139  53.376  < 2e-16 ***
hr19         1.644814   0.036694  44.825  < 2e-16 ***
hr20         1.355291   0.037596  36.048  < 2e-16 ***
hr21         1.105998   0.038630  28.630  < 2e-16 ***
hr22         0.855694   0.040007  21.389  < 2e-16 ***
hr23         0.479234   0.042772  11.204  < 2e-16 ***
season2      0.316960   0.014812  21.399  < 2e-16 ***
season3      0.238896   0.018060  13.228  < 2e-16 ***
season4      0.454967   0.013045  34.876  < 2e-16 ***
holiday1    -0.157698   0.024821  -6.353 2.16e-10 ***
workingday1  0.025813   0.008245   3.131  0.00175 **
weathersit2 -0.036946   0.009423  -3.921 8.86e-05 ***
weathersit3 -0.459637   0.018957 -24.246  < 2e-16 ***
weathersit4 -0.430809   0.446581  -0.965  0.33472
temp         0.406274   0.126796   3.204  0.00136 **
atemp        0.962882   0.138243   6.965 3.40e-12 ***
hum         -0.350530   0.026167 -13.396  < 2e-16 ***
windspeed   -0.156621   0.032141  -4.873 1.11e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for quasipoisson family taken to be 44.39723)

    Null deviance: 2891591  on 17378  degrees of freedom
Residual deviance:  760407  on 17343  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 5
```

Firstly, when comparing the results above with the ones from **Table 9** - Summary Poisson Regression Hour and **Table 10** - Summary Poisson Regression Day and, it can be seen that in the case of **Day** dataset, not all the variables seem to have a significance over response variable

1	2	3	4	5	6	7
22.478702	13.847146	9.546481	5.068557	2.776462	7.987391	33.138445
8	9	10	11	12	13	14
88.796937	154.891816	99.059275	78.525110	83.456626	111.832549	108.506386
15	16	17	18	19	20	21
101.793975	102.423133	124.102953	191.402375	114.380518	85.460895	95.143445
22	23	24	25	26	27	28
74.318065	56.625550	41.606397	25.725862	15.673787	10.412772	5.705321
29	30	31	32	33	34	35
3.125270	24.932503	104.172520	112.653285	97.666296	69.753835	80.675562

Table 17: Prediction Quasi Poisson Hour

96.635672	97.326694	58.334191	58.923812	75.344263	188.536588	169.733397

in the Quasi-Poisson regression. Secondly, it can be seen that the standard error is higher in the Quasi-Poisson regression and also the dispersion parameter. This proves that there was indeed an over dispersion in the data and that standard error was underestimated in the Poisson model. Still calculating the pseudo-R Squared we see that the results are the same 0.78 for the **Day** dataset and 0.74 for the **Hour** dataset which is quite interesting. The exponential value of coefficients is shown in **Table 18 -** Coefficients Quasi-Poisson Day and **Table 19 -** Coefficients Quasi-Poisson Hour.

Table 18 - Coefficients Quasi-Poisson Day

(Intercept)	dteday	season2	season3	season4	holiday1
4.619798e-05	1.000000e+00	1.321074e+00	1.087635e+00	1.221007e+00	8.430506e-01
workingday1	weathersit2	weathersit3	temp	atemp	hum
1.028818e+00	9.137808e-01	4.878163e-01	2.032164e+00	1.901193e+00	7.049451e-01
windspeed					
5.773134e-01					

Looking at **Table 11** - Coefficients Day and **Table 18** - Coefficients Quasi-Poisson Day in the **Day** dataset it is shown that the values have not changed, while looking at **Table 12** - Coefficients Hour and **Table 19** - Coefficients Quasi-Poisson Hour in the **Hour** dataset there are some values which remained the same and some that are changed.

Table 19 - Coefficients Quasi-Poisson Hour

(Intercept)	hr1	hr2	hr3	hr4	hr5	hr6	hr7	hr8	hr9
28.6167266	0.6296092	0.4359301	0.2246783	0.1234506	0.3911392	1.5140279	4.1832597	6.8027706	3.9759690
hr10	hr11	hr12	hr13	hr14	hr15	hr16	hr17	hr18	hr19
3.0044016	3.4386107	4.0632831	3.9632718	3.7077352	3.8551228	4.8269155	7.3064492	6.8822356	5.1800459
hr20	hr21	hr22	hr23	season2	season3	season4	holiday1	workingday1	weathersit2
3.8778909	3.0222400	2.3530066	1.6148367	1.3729479	1.2698461	1.5761221	0.8541078	1.0261489	0.9637277
weathersit3	weathersit4	temp	atemp	hum	windspeed				
0.6315126	0.6499834	1.5012135	2.6192343	0.7043149	0.8550277				

A prediction is done in **Table 20** - Prediction Quasi-Poisson Day and **Table 21** - Prediction Quasi-Poisson Hour for both datasets with the Quasi-Poisson regression model.

Table 20 - Prediction Quasi-Poisson Day

1	2	3	4	5	6	7	8
1773.7357	1771.0458	1746.9030	1770.0077	1898.9997	1923.2895	1660.6064	1434.4619
9	10	11	12	13	14	15	16
1472.3806	1659.9135	1554.5876	1560.9658	1619.2602	1779.9552	1725.1629	1849.7447
17	18	19	20	21	22	23	24
1292.9924	1544.2353	1715.6956	1767.2464	1617.0123	1550.1296	1529.5146	1643.1167
25	26	27	28	29	30	31	32
1722.5564	753.6986	1805.2297	1596.6755	1743.6899	1841.3141	1588.2445	1644.6151
33	34	35	36	37	38	39	40

As before, the data in the **Day** table is not changed compared with the results in the Poisson regression, but the results in the **Hour** table are different.

Table 21 - Prediction Quasi-Poisson Hour

1	2	3	4	5	6	7
22.478702	13.847146	9.546481	5.068557	2.776462	7.987391	33.138445
8	9	10	11	12	13	14
88.796937	154.891816	99.059275	78.525110	83.456626	111.832549	108.506386
15	16	17	18	19	20	21
101.793975	102.423133	124.102953	191.402375	114.380518	85.460895	95.143445
22	23	24	25	26	27	28
74.318065	56.625550	41.606397	25.725862	15.673787	10.412772	5.705321
29	30	31	32	33	34	35
3.125270	24.932503	104.172520	112.653285	97.666296	69.753835	80.675562
36	37	38	39	40	41	42
96.635672	97.326694	58.334191	58.923812	75.344263	188.536588	169.733397

In the *Poisson and Quasi-Poisson regression for Day and Hour* appendix the code can be seen for the previous two analysis.

TIME SERIES ANALYSIS

A time-series analysis has the goal of making informed predictions about future values, based on past values. Forecasting bike demand is useful for city planning purposes and for predicting the growth of companies offering bike renting possibilities.

The seasonality for hourly data has already been presented in *Figure 10. Hour vs total users* and *Figure 11. Hours vs total users only for weekends* and it is consistent over the years and when comparing working days to weekends. The pattern also holds when taking into account the growth of the application over the years.

Since the data used for this project consists of sets of numerical measurements of the same entity taken at spaced intervals over time (daily and hourly), this dataset is perfect for using a time series analysis.

This part will focus on time series analysis and time series forecasting for the day dataset, as this is the more important metric for decision-makers in regard to city planning. Stakeholders will also be able to anticipate future growth of the company.

For this analysis, Facebook Prophet was used.

Prophet implements a procedure for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects.

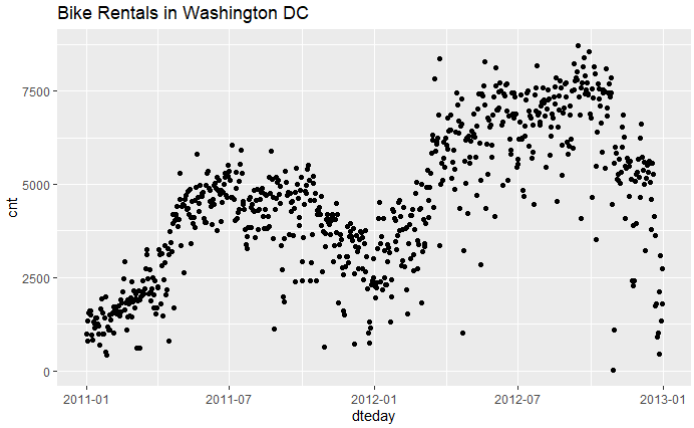


Figure 44: Daily bike rentals between January 1'st 2011 and January 10'th 2013

Time series components

- **Trend**

There is a growing trend of bike rentals in the data. The promise of saturation is the maximum potential number of "Capital Bikeshare" users in Washington D.C. Over the 2 year period, the number of users renting bikes from this application has increased significantly, meaning that more people are interested in alternative ways of getting around the city.

- **Seasonality**

The data presents a seasonality of 12. During the winter months, the number of bike rentals is lower, and during summer months the number of bike rentals hits peak values. This means that weather conditions associated with the different seasons have a great impact on the number of bike rentals. There is regularity in the data, as this seasonal pattern holds true over 2012.
The model is multiplicative, as it is expected that the number of bike rentals to grow from year to year due to how people are becoming more aware of their environmental impact and the benefits of using a bike.

- **Random variation**

In the months of July, for both 2011 and 2012, there seems to be a dip in the number of bike rentals. This variation could be explained by the fact that temperatures reach their maximum level during this month, which makes it unpleasant to use a bike in the city.

- **Irregularities and outliers**

There seem to be a fair number of outliers in the dataset, but they can be attributed to random variations, extreme weather, or events in the city.

Relative contribution of components

Seasonality is more important than trend, as it accounts for most of the variation.

Simple prediction that captures trend and seasonality

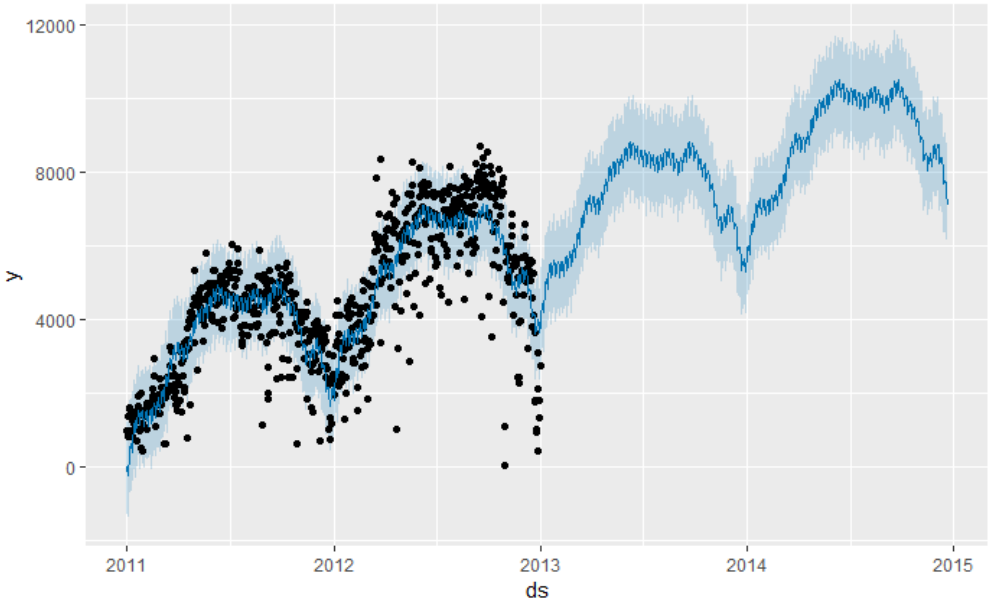


Figure 45: Prediction for the next 720 days

The black dots are actual data points and the blue line is predicted values.

The confidence bound is the lighter blue around the blue line.

The model seems decent, as most of the points fall within the bounds and the model seems to capture trend and seasonality.

This is in contrast with *Figure* 17, where the linear model was unable to account for the increasing trend in the data.

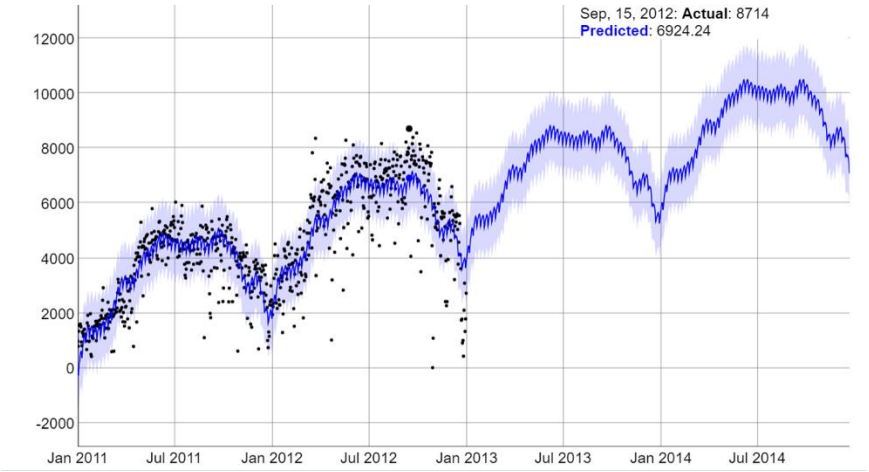


Figure 46:Predicted values vs actual values

It seems that for the second year, the data points during the months of June to October present more variance which makes it harder for the model to predict the actual values and increases the upper and lower bound.

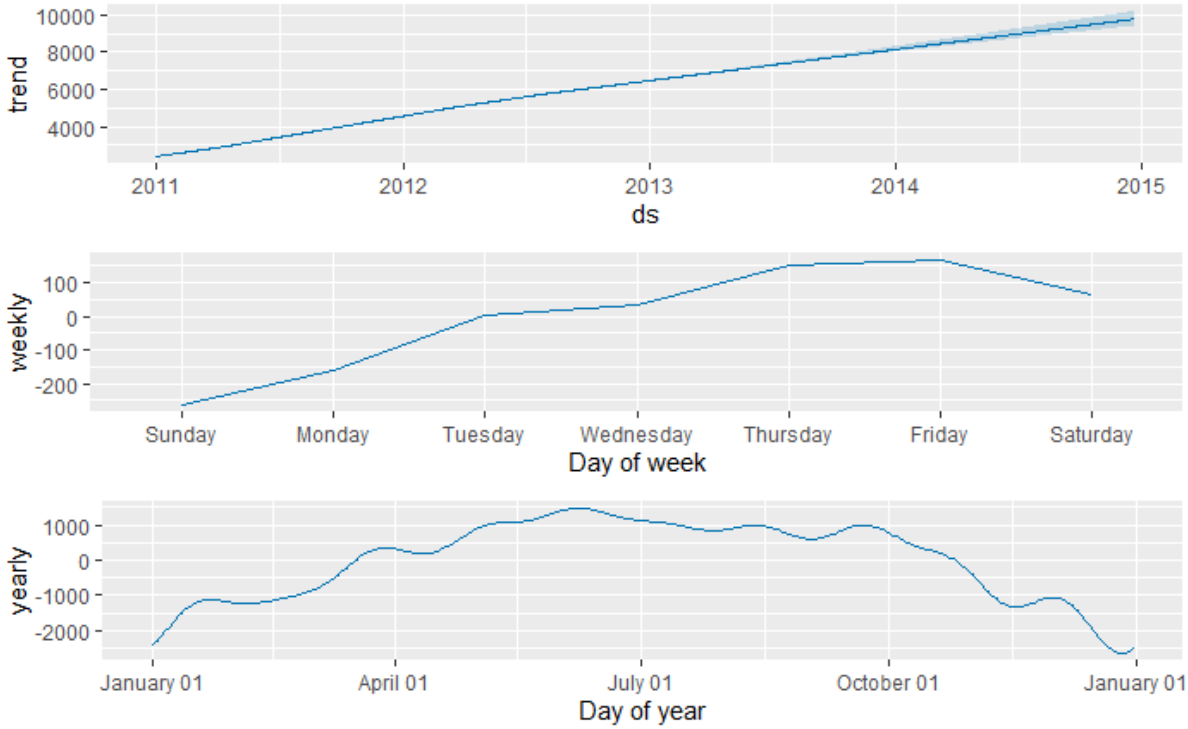


Figure 47: Forecast components

There is an increasing trend in the data.There is a pattern in rentals, as bike rentals on Thursday and Fridays are comparatively higher than rentals on Sunday and Monday.
There is also a seasonality of 12 in the data, with lower rentals in Winter and peaks during summer months.

Model performance

Adjusted R-squared: 0.7454

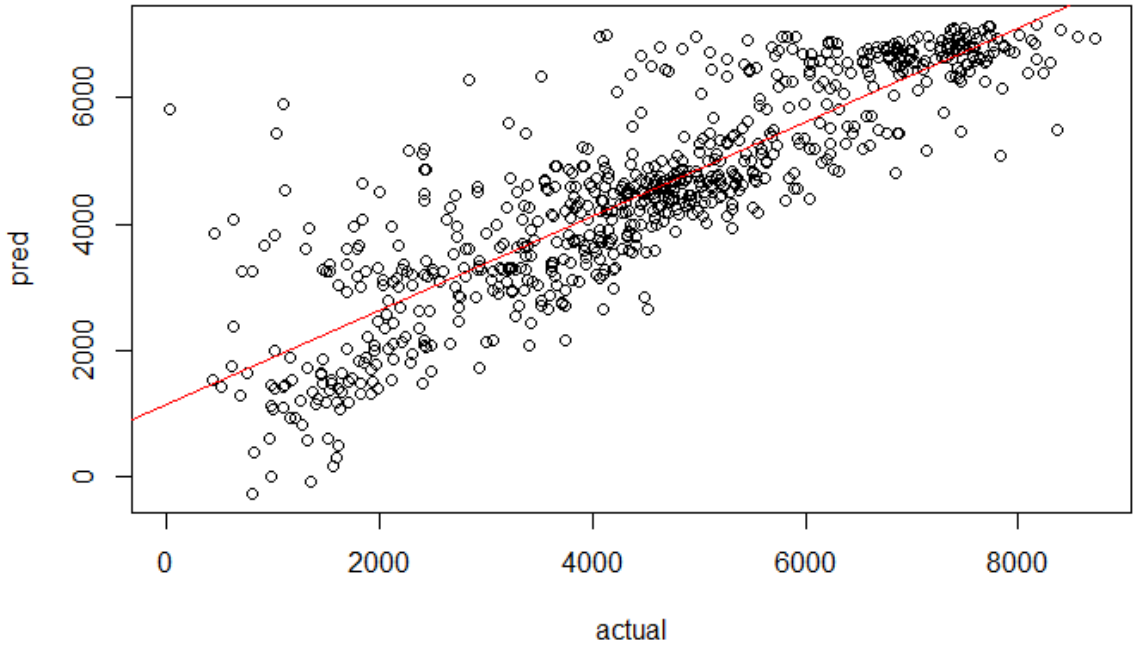


Figure 48: Actual values vs Predicted values

The model has decent performance, but there is a lot of scattering around the red line. To improve the model performance, more input information should be included. Prophet comes with an inbuilt method of validation, called cross_validation. Since the data is only for 2 years, it is not enough for using this method. Using first year as training data and the second year as test data, the following results are obtained.

	y <dbl>	ds <S3: POSIXct>	yhat <dbl>	yhat_lower <dbl>	yhat_upper <dbl>	cutoff <S3: POSIXct>
1	2236	2012-01-03	2072.246	1255.1468	2983.876	2012-01-02
2	2368	2012-01-04	1834.433	953.3220	2751.022	2012-01-02
3	3272	2012-01-05	1917.962	995.0189	2709.412	2012-01-02
4	4098	2012-01-06	2044.100	1204.0415	2968.933	2012-01-02
5	4521	2012-01-07	1960.162	1030.3637	2833.143	2012-01-02
6	3425	2012-01-08	1941.232	1088.6761	2813.430	2012-01-02

Figure 49: Cross validation data frame

Since it cannot see the trend in the data, the prediction has a very high Mean absolute percentage error(MAPE), which means that predictions more than a few days in the future are very bad.

	horizon <time>	mse <dbl>	rmse <dbl>	mae <dbl>	mape <dbl>	mdape <dbl>	smape <dbl>	coverage <dbl>
1	36 days	2309675	1519.762	1323.054	0.3698478	0.4112806	0.4667270	0.3333333
2	37 days	2319696	1523.055	1335.798	0.3739852	0.4112806	0.4715583	0.3333333
3	38 days	2376211	1541.496	1363.280	0.3787713	0.4112806	0.4782907	0.3055556
4	39 days	2376502	1541.591	1363.387	0.3771220	0.4031803	0.4757631	0.3055556
5	40 days	2261139	1503.708	1313.439	0.3664755	0.3942583	0.4602787	0.3333333
6	41 days	2101889	1449.789	1267.533	0.3672414	0.3942583	0.4510493	0.3333333

Figure 50: Performance metrics

Time series limitations

When it comes to time series forecasting, the further into the future the predictions, the less accurate they will be. There are lots of limitations when it comes to the forecast.

Time series assumes that the existing pattern in the data will continue, as it does not take into account the saturation point, marketing campaigns from the company or law maker changes that would make people more or less likely to use bikes.

If there is a change in circumstances, such as a new bike sharing app that competes with the existing one, or people decide to buy instead of renting bikes, then the model will be unable to predict them.

Another limitation is that for this analysis the data was limited to only 2 years, which is not very much.

Adding regressors to improve model performance

Taking into consideration the limitations presented above, going further into this analysis, the horizon for making the forecast will only be 21 days into the future. Since data points end at January 10’th, 2013; adding 21 more days allows a clear view of the month of January.

Limiting the analysis also means that the confidence of the predictions will be higher.

Used regressors:

- Holidays
- Temperature
- Humidity

1. Holiday regressor

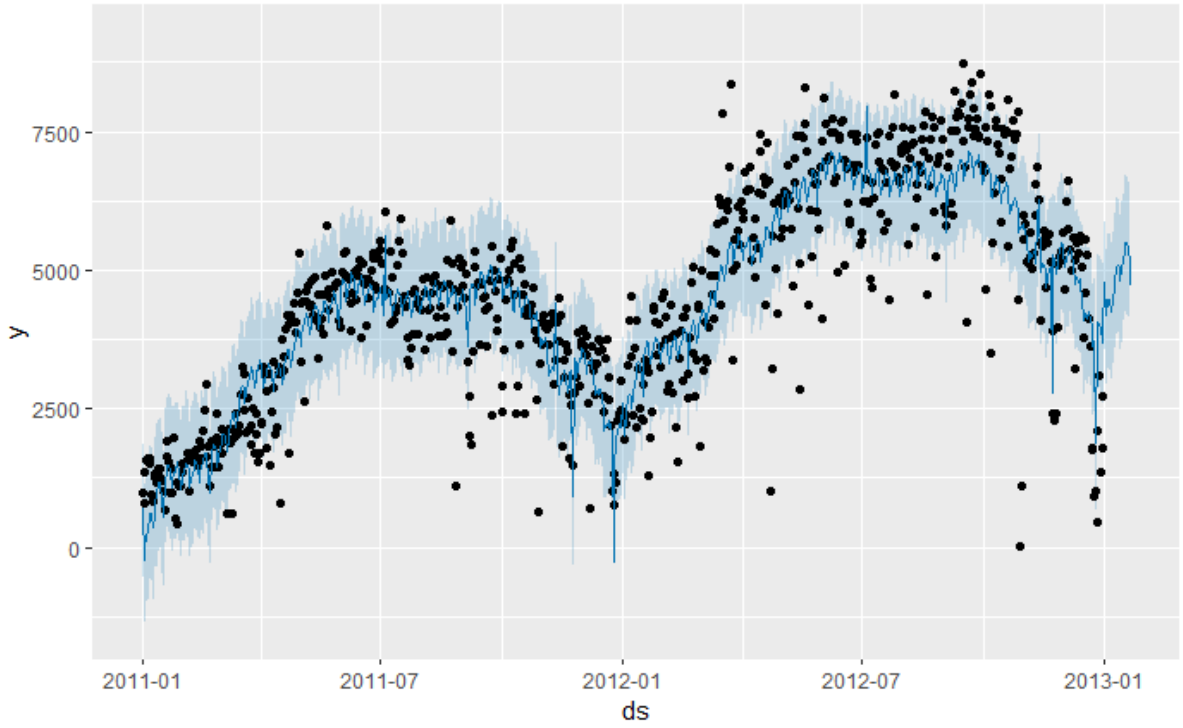


Figure 51: Model with US holiday regressor

Figure 52: Variance generated by holiday

Model performance - Adjusted R-squared: 0.7557

Adding the holiday regressor allowed the model to fit some of the outliers, which improved just a tiny bit of the model performance.

2. Holiday + Temperature

Regressors need to be available for the values that we want to predict.

Noticing that for the month of January, temperature values fall between 0.1 and 0.3, we generate a data frame with 21 values for the temperature to account for the 21 days in the future that we want to predict.

An improvement to this method would be to use a third-party source in order to get weather forecast data.

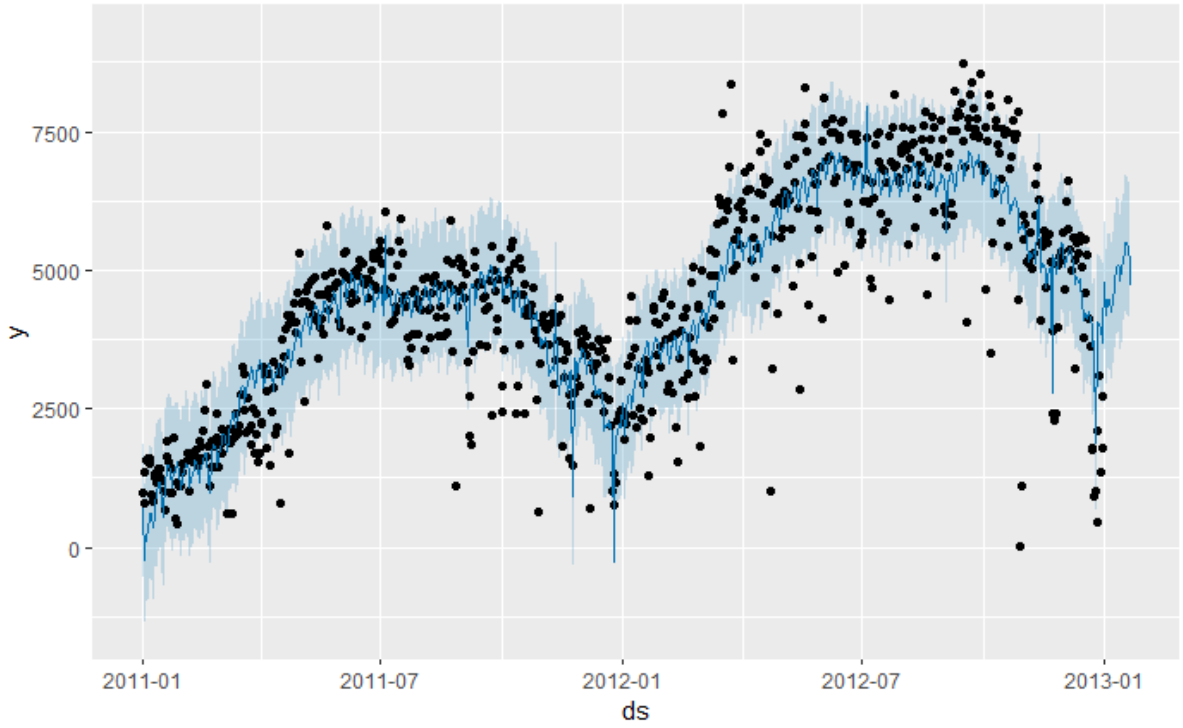


Figure 53: Model with US holiday and temperature regressors

This prediction seems very similar to the previous one, so the model improvement is expected to not be very significant.

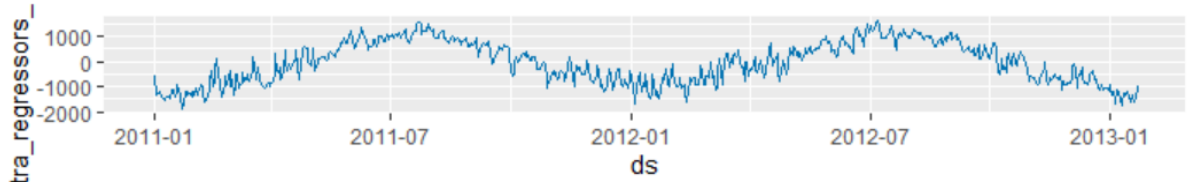


Figure 54: Variance generated by temperature

The temperature regressor is in additive mode.

Model performance - Adjusted R-squared: 0.7764

As expected from the plot, the model improvement isn't very significant.

3. Holiday + Temperature + Humidity

Again, it is necessary to create humidity values for the forecasting. Noticing that for the month of January, humidity values fall between 0.4 and 0.8, we generate a data frame with 21 values for humidity to account for the 21 days in the future that we want to predict.

An improvement to this would be to use a third-party source in order to get weather forecast data.

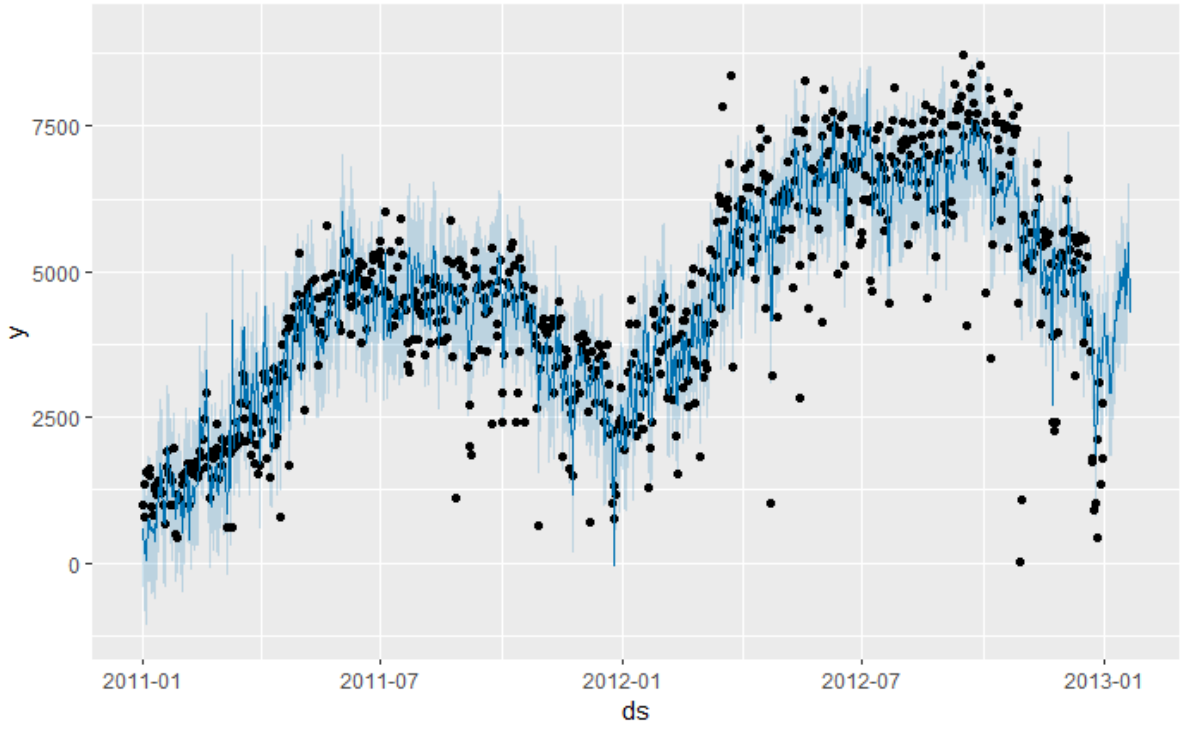


Figure 55: Model with all 3 regressors

Adding the humidity regressor added a lot more variance in the forecasting model, making it so that it covers more points. From the plotting, it is expected that this regressor will significantly improve the model's performance.

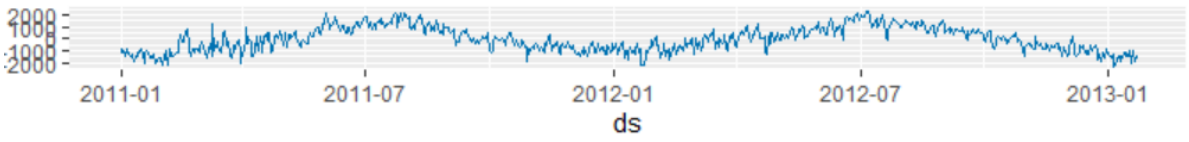


Figure 56: Variance generated by humidity

Model performance - Adjusted R-squared: 0.8144

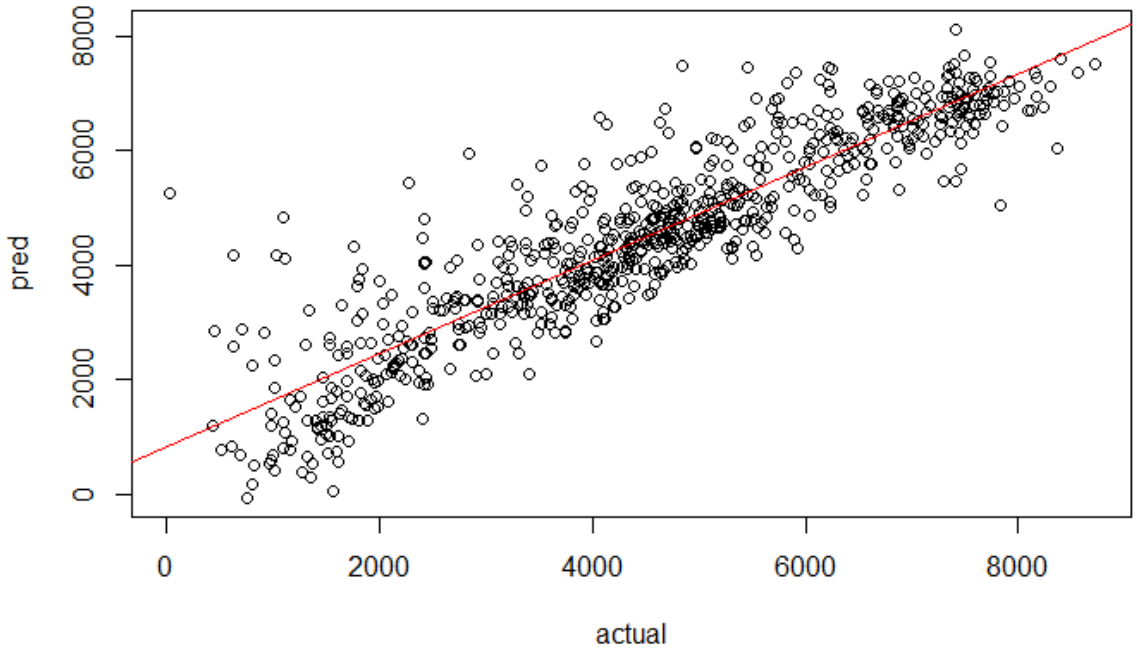


Figure 57: Actual values vs Predicted values after adding regressors

It is noticeable that there is a better linear fit for the data once the regressors have been added to the model. There is less variance.

Results

Adding more regressors didn't show better results for the data. Some of them worsened the model performance, while others increased the model performance but at the same time generated a very bad forecast.

For example, adding the wind regressor with generated values between 1 and 3 resulted in a very skewed representation of predicted values for the future.

Model performance - Adjusted R-squared: 0.8359

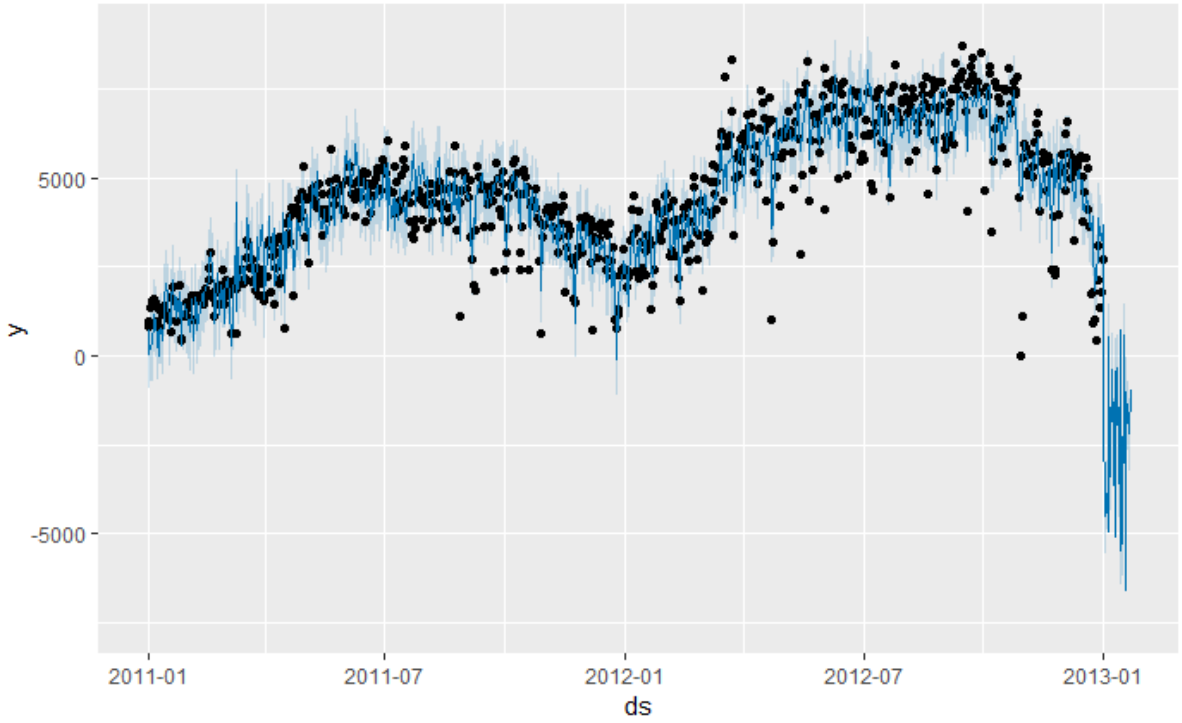


Figure 58: Model with previous 3 regressors + windspeed

Figure 59: Variance generated by windspeed

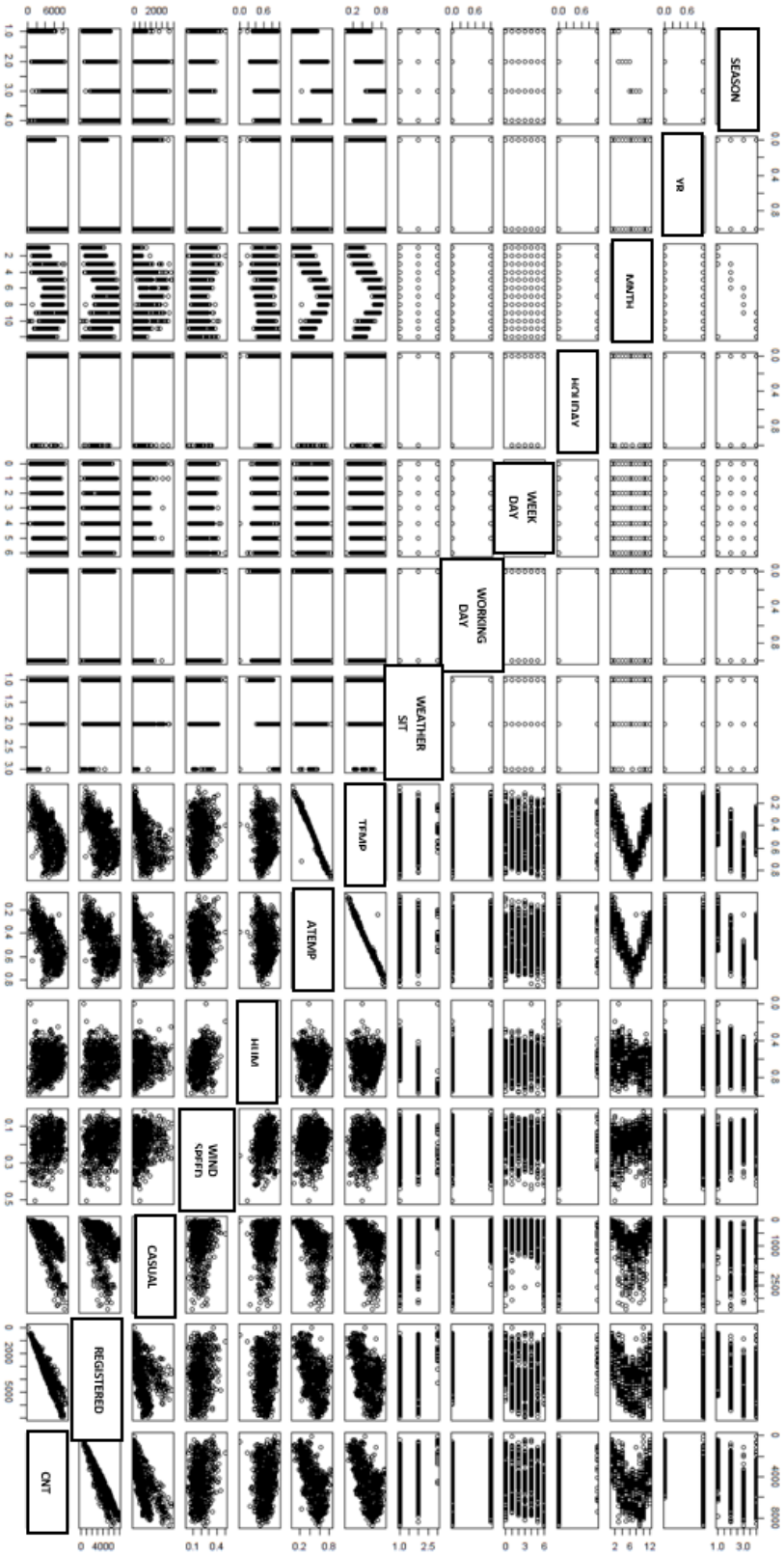
7. CONCLUSION

Reaching the end of our report some conclusion can be drawn. A dataset was selected based on number of attributes and rows to be the subject of analysis, respectively Bike Share. There are two datasets, one for **Day** and one for **Hour**, and variables are both quantitative and qualitative, containing information about number of bike rental users between years 2011 and 2012 on different weather conditions. Based on the information that the database provided some research questions were formulated. The main objective was to both discover patterns and better understand the data regarding the behavior of bike renting users and also to predict the number of users that will rent a bike based on different values for features like weather condition or seasonality. The report was structured in five main parts. Since the response variable is quantitative, this problem can be defined as a regression problem within the supervised learning. The number of total users is mostly influenced by the weather conditions. By doing a correlation matrix it is revealed that the highest correlation with the response variable is from temperature and felt temperature variables. A linear regression was applied, and the results showed that is not the best fit with a R-Squared of only 40 %. The Error improved quite a lot through multiple linear regression compared to simple linear regression. But the error is still quite high. After, a Poisson regression was applied. Using an overdispersion analysis it is found that this type of regression is not the best to predict the number of users, so a Quasi-Poisson is made also. The pseudo-R Squared is calculated for both regressions reaching a value of 0.78 for Day and 0.74 for Hour datasets in both cases. Still, some predictions are made.

8. REFERENCES

[1] „An Introduction to Statistical Learning, with Applications in R, Second Edition”. Authors: Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani.

9. APPENDIX


SEASON
YR
MNTH
HOLIDAY
WEEK DAY
WORKING DAY
WEATHER SIT
TEMP
ATEMP
HUM
WIND SPEED
CASUAL
REGISTERED
CNT

10. APPENDIX FOR CODE IN R

I. Histograms

```
########## Uploading data set ##########
day = read.csv("day.csv")
View(day)
attach(day)

########### Histogram cnt, registered and casual ##########
par(mfrow = c(2,3))
hist(cnt, xlab = "", ylab = "", main = "Total dailiy users", breaks = 30 )
hist(registered, xlab = "", ylab = "", main = "Registered dailiy users", breaks = 30 )
hist(casual, xlab = "", ylab = "", main = "Casual dailiy users", breaks = 30 )
```

```
########## Uploading and investigating data set ##########

hour = read.csv("hour.csv")
View(hour)
attach(hour)
names(hour)
dim(hour)
sum(is.na(hour$cnt)) #no missing observations


########### Histogram cnt, registered and casual ##########

par(mfrow = c(1,3))
hist(cnt, xlab = "", ylab = "", main = "Total hourly users", breaks = 30 )
hist(registered, xlab = "", ylab = "", main = "Registered hourly users", breaks = 30 )
hist(casual, xlab = "", ylab = "", main = "Casual hourly users", breaks = 30 )
```

II. Scatterplot and Correlation matrix

```
########## Scatter plot ##########
day <- day[,-c(1,2)]
pairs(day)

########## Correlation matrix ##########
cor(day)
View(cor(day))
par(mfrow = c(1,1))
corrplot(cor(day), method = "number")
```

```
########## Scatter plot and correlation matrix ##########
hour = hour[-c(1,2)]
pairs(hour)
par(mfrow = c(1,1))
cor(hour)
View(cor(hour))
corrplot(cor(hour), method = "number")
```

III. Some initial plots

```
###### Some plots ######

boxplot(cnt ~ season)
boxplot(cnt ~ mnth)
plot(mnth, cnt, pch = 19, col = season)
boxplot(cnt ~ workingday)
boxplot(cnt ~ weekday)
boxplot(cnt ~ holiday)
boxplot(cnt ~ weathersit)
plot(temp, cnt, col = season, pch = 19)
plot(atemp, cnt)
plot(hum, cnt)
plot(windspeed, cnt)
```

IV. Filter weekend

```
weekend = hour %>% filter(weekday == 0 | weekday == 6)
View(weekend)
plot(weekend$hr, weekend$cnt)
```

V. Training and testing dataset 80/20

```
########## DIVIDING TRAINING AND TESTING DATA ##########
# More or less 80/20
set.seed(1)
rand = sample(1:731, 600, replace = FALSE)
train1 <- day[rand,]
test1 <- day[-rand,]
```

VI. Linear, quadratic, cubic, logarithmic, and squared root models in Day

```
# Temperature feeling atemp linear
plot(atemp, cnt, ylab = "Total users", xlab = "Feeling temperature")
lines(lowess(atemp, cnt)) #shows best approximation line
lm.atemp = lm(cnt ~ atemp, data = train1)
summary(lm.atemp)
abline(lm.atemp, col = "red")
confint(lm.atemp) #confidence interval coefficients estimates
predict(lm.atemp,train1, interval = "confidence")
predict(lm.atemp, train1, interval = "prediction")
sqrt(mean((train1$cnt - predict(lm.atemp, train1))^2))

plot(train1$cnt, predict(lm.atemp, train1), xlab = "Real", ylab = "Predicted")
lines(train1$cnt, train1$cnt, col = "red")
plot(lm.atemp$fitted.values, lm.atemp$residuals, xlab = "Fitted values", ylab = "Residuals")
abline(h=0, col = "red", lwd = 3)
qqnorm(rstandard(lm.atemp), pch = 1, ylab = "Standarized residuals")
qqline(rstandard(lm.atemp), col = "steelblue", lwd = 2)

#For test, prediction
sqrt(mean((test1$cnt - predict(lm.atemp, test1))^2))
plot(test1$cnt, predict(lm.atemp, test1), xlab = "Real", ylab = "Predicted")
resultsaTemp = data.frame(test1[[14]],predict(lm.atemp, test1))
plot(resultsaTemp$predict.lm.atemp..test1., xlab = "Test dataset point", ylab = "Predicted values for cnt", col = test1$yr)
plot(resultsaTemp$test1..14.., xlab = "Test dataset point", ylab = "Real values for cnt in test dataset", col = test1$yr)
```

```
# Temperature feeling quadratic
plot(atemp, cnt)
lm.atemp2 = lm(cnt ~ poly(atemp,2, raw = TRUE), data = train1)
lines(train1$atemp, (predict(lm.atemp2)), col="red")
summary(lm.atemp2)
confint(lm.atemp2)
predict(lm.atemp2, train1)
sqrt(mean((train1$cnt - predict(lm.atemp2, train1))^2)) #lower than with simple, also with other seed

plot(train1$cnt, predict(lm.atemp2, train1), xlab = "Real", ylab = "Predicted")
plot(lm.atemp2$fitted.values, lm.atemp2$residuals, xlab = "Fitted", ylab = "Residuals")# not linear #not Homoscedas
qqnorm(rstandard(lm.atemp2), pch = 1, ylab = "Standarized residuals")
qqline(rstandard(lm.atemp2), col = "steelblue", lwd = 2)

#For test, prediction
sqrt(mean((test1$cnt - predict(lm.atemp2, test1))^2))
plot(test1$cnt, predict(lm.atemp2, test1))
resultsaTemp2 = data.frame(test1[[14]], predict(lm.atemp2, test1))
```

```
# Temperature feeling cubic
plot(atemp, cnt, xlab = "Feeling temperature, atemp", ylab = "Total users, cnt")
lm.atemp3 = lm(cnt ~ poly(atemp,3, raw = TRUE), data = train1)
lines(smooth.spline(train1$atemp, predict(lm.atemp3)), col="red", lwd = 3)
summary(lm.atemp3)
confint(lm.atemp3)
predict(lm.atemp3, train1)
sqrt(mean((train1$cnt - predict(lm.atemp3, train1))^2)) #lower than with simple, also with other seed

plot(train1$cnt, predict(lm.atemp3, train1), xlab = "Real", ylab = "Predicted")
plot(lm.atemp2$fitted.values, lm.atemp3$residuals, xlab = "Fitted", ylab = "Residuals")
qqnorm(rstandard(lm.atemp3), pch = 1, ylab = "Standarized residuals")
qqline(rstandard(lm.atemp3), col = "steelblue", lwd = 2)

#For test, prediction
sqrt(mean((test1$cnt - predict(lm.atemp3, test1))^2))
plot(test1$cnt, predict(lm.atemp3, test1))
resultsaTemp3 = data.frame(test1[[14]], predict(lm.atemp3, test1))
```

```
# Temperature feeling logaritmic
plot(log(atemp), cnt)
lm.atemplog = lm(cnt ~ log(atemp), data = train1)
lines(train1$atemp, (predict(lm.atemplog)), col="red")
summary(lm.atemplog)
confint(lm.atemplog)
predict(lm.atemplog, train1)
sqrt(mean((train1$cnt - predict(lm.atemplog, train1))^2)) #lower than with simple, also with other seed

plot(train1$cnt, predict(lm.atemplog, train1), xlab = "Real", ylab = "Predicted")
plot(lm.atemplog$fitted.values, lm.atemplog$residuals, xlab = "Fitted", ylab = "Residuals")# not linear #not Homosc
qqnorm(rstandard(lm.atemplog), pch = 1, ylab = "Standarized residuals")
qqline(rstandard(lm.atemplog), col = "steelblue", lwd = 2)

#For test, prediction
sqrt(mean((test1$cnt - predict(lm.atemplog, test1))^2))
plot(test1$cnt, predict(lm.atemplog, test1))
resultsatemplog = data.frame(test1[[14]], predict(lm.atemplog, test1))
```

```
# Temperature feeling squared root
plot(sqrt(atemp), cnt)
lm.atempsqrt = lm(cnt ~ sqrt(atemp), data = train1)
lines(train1$atemp, (predict(lm.atempsqrt)), col="red")
summary(lm.atempsqrt)
confint(lm.atempsqrt)
predict(lm.atempsqrt, train1)
sqrt(mean((train1$cnt - predict(lm.atempsqrt, train1))^2)) #lower than with simple, also with other seed

plot(train1$cnt, predict(lm.atempsqrt, train1), xlab = "Real", ylab = "Predicted")
plot(lm.atempsqrt$fitted.values, lm.atempsqrt$residuals, xlab = "Fitted", ylab = "Residuals")# not linear #not Homo
qqnorm(rstandard(lm.atempsqrt), pch = 1, ylab = "Standarized residuals")
qqline(rstandard(lm.atempsqrt), col = "steelblue", lwd = 2)

#For test, prediction
sqrt(mean((test1$cnt - predict(lm.atempsqrt, test1))^2))
plot(test1$cnt, predict(lm.atempsqrt, test1))
resultsatempsqrt = data.frame(test1[[14]], predict(lm.atempsqrt, test1))
```

VII. Filtering by hour

```
hour1 <- hour %>% filter(hr == 1)
View(hour1)
plot(hour1$atemp, hour1$cnt, col = hour8$weekday, lwd =2, xlab = "Feeling temperature for hour = 1", ylab = "Total users for hour = 1")

hour8 <- hour %>% filter(hr == 8)
View(hour8)
plot(hour8$atemp, hour8$cnt, col = hour8$weekday, lwd = 2, xlab = "Feeling temperature for hour = 8", ylab = "Total users for hour = 8")

hour17 <- hour %>% filter(hr == 17)
View(hour17)
plot(hour17$atemp, hour17$cnt, col = hour8$weekday, lwd = 2,xlab = "Feeling temperature for hour = 17", ylab = "Total users for hour = 17" )
```

FEATURE SELECTION

VIII. Best subset selection in Day and Hour

```
## BEST SUBSET SELECTION
#day <- day[, -c(12,13)]
str(day)
View(day)


fit.full <- regsubsets(cnt ~., data = day, nvmax = 28)
summary <- summary(fit.full)
summary
summary$rsq
names(summary)


par(mfrow = c(1,1))
plot(summary$rsq, xlab = "Number of variables", ylab = "RSQ", type = "l")
which.max(summary$rsq)
points(28,summary$rsq[28], col = "red", cex = 2, pch = 20 )

plot(summary$adjr2, xlab = "Number of variables", ylab = "Adjusted R2", type = "l")
which.max(summary$adjr2)
points(23,summary$rsq[23], col = "red", cex = 2, pch = 20 )

plot(summary$cp, xlab = "Number of variables", ylab = "CP", type = "l")
which.min(summary$cp)
points(19,summary$cp[19], col = "red", cex = 2, pch = 20 )

plot(summary$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
which.min(summary$bic)
points(15,summary$bic[15], col = "red", cex = 2, pch = 20 )
```

```
## BEST SUBSET SELECTION
hoursub <- hoursub[, -c(12,13)]
str(hoursub)
View(hoursub)
hoursub [c("season", "yr",  "hr", "holiday", "weekday", "workingday", "weathersit" )]
 <- lapply(hoursub[c("season", "yr",  "hr", "holiday", "weekday", "workingday", "weathersit")], factor)
attach(hoursub)

fit.full <- regsubsets(cnt ~., data = hoursub, nvmax = 40)
summary <- summary(fit.full)
summary
summary$rsq
names(summary)
```

IX. Backward stepwise selection in Day and Hour

```
## BACKWARDS STEPWISE
fit.back <- regsubsets(cnt ~., data = day, nvmax = 28, method = "backward")
summaryBC <- summary(fit.back)
summaryBC$rsq
names(summaryBC)

plot(summaryBC$rsq, xlab = "Number of variables", ylab = "RSQ", type = "l")
which.max(summaryBC$rsq)
points(28,summaryBC$rsq[28], col = "red", cex = 2, pch = 20 )

plot(summaryBC$adjr2, xlab = "Number of variables", ylab = "Adjusted R2", type = "l")
which.max(summaryBC$adjr2)
points(25,summaryBC$rsq[25], col = "red", cex = 2, pch = 20 )

plot(summaryBC$cp, xlab = "Number of variables", ylab = "CP", type = "l")
which.min(summaryBC$cp)
points(23,summaryBC$cp[23], col = "red", cex = 2, pch = 20 )

plot(summaryBC$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
which.min(summaryBC$bic)
points(13,summaryBC$bic[13], col = "red", cex = 2, pch = 20 )
```

```
## BACKWARDS STEPWISE
fit.back <- regsubsets(cnt ~., data = hoursub, nvmax = 40, method = "backward")
summaryBC <- summary(fit.back)
summaryBC
names(summaryBC)

plot(summaryBC$rsq, xlab = "Number of variables", ylab = "RSQ", type = "l")
which.max(summaryBC$rsq)
points(31,summaryBC$rsq[31], col = "red", cex = 2, pch = 20 )

plot(summaryBC$adjr2, xlab = "Number of variables", ylab = "Adjusted R2", type = "l")
which.max(summaryBC$adjr2)
points(31,summaryBC$rsq[31], col = "red", cex = 2, pch = 20 )

plot(summaryBC$cp, xlab = "Number of variables", ylab = "CP", type = "l")
which.min(summaryBC$cp)
points(30,summaryBC$cp[30], col = "red", cex = 2, pch = 20 )

plot(summaryBC$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
which.min(summaryBC$bic)
points(23,summaryBC$bic[23], col = "red", cex = 2, pch = 20 )
```

X. Forward stepwise selection in Day and Hour

```
## FORWARD STEPWISE

fit.for <- regsubsets(cnt ~., data = day, nvmax = 28, method = "forward")
summaryFW <- summary(fit.for)
summaryFW$rsq
names(summaryFW)

plot(summaryFW$rsq, xlab = "Number of variables", ylab = "RSQ", type = "l")
which.max(summaryFW$rsq)
points(28,summaryFW$rsq[28], col = "red", cex = 2, pch = 20 )

plot(summaryFW$adjr2, xlab = "Number of variables", ylab = "Adjusted R2", type = "l")
which.max(summaryFW$adjr2)
points(22,summaryBC$rsq[22], col = "red", cex = 2, pch = 20 )

plot(summaryFW$cp, xlab = "Number of variables", ylab = "CP", type = "l")
which.min(summaryFW$cp)
points(19,summaryFW$cp[19], col = "red", cex = 2, pch = 20 )

plot(summaryFW$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
which.min(summaryFW$bic)
points(18,summaryBC$bic[18], col = "red", cex = 2, pch = 20 )
```

```
## FORWARD STEPWISE

fit.for <- regsubsets(cnt ~., data = hoursub, nvmax = 40, method = "forward")
summaryFW <- summary(fit.for)
summaryFW
names(summaryFW)

plot(summaryFW$rsq, xlab = "Number of variables", ylab = "RSQ", type = "l")
which.max(summaryFW$rsq)
points(31,summaryFW$rsq[31], col = "red", cex = 2, pch = 20 )

plot(summaryFW$adjr2, xlab = "Number of variables", ylab = "Adjusted R2", type = "l")
which.max(summaryFW$adjr2)
points(28,summaryBC$rsq[28], col = "red", cex = 2, pch = 20 )

plot(summaryFW$cp, xlab = "Number of variables", ylab = "CP", type = "l")
which.min(summaryFW$cp)
points(27,summaryFW$cp[27], col = "red", cex = 2, pch = 20 )

plot(summaryFW$bic, xlab = "Number of variables", ylab = "BIC", type = "l")
which.min(summaryFW$bic)
points(24,summaryBC$bic[24], col = "red", cex = 2, pch = 20 )
```

XI. Predict function for reagsubsets

```
# Predict function for reagsubstets
predict.regsubsets = function(object,newdata,id,...){# define function for prediction
  form=as.formula(object$call[[2]])# object: variable passed to the fct regfit; call is a list of at least two ele
  mat=model.matrix(form,newdata)# define matrix
  coefi=coef(object,id=id)# define coefficients
  xvars=names(coefi)# display names of the predictors
  mat[,xvars]%*%coefi# display matrix with the coefficients
}
```

XII. Validation set approach for Day and Hour

```
## VALIDATON SET APPROACH
fit.best=regsubsets(cnt~., data=train1,nvmax=28)# perform best subset selection on training data
test.mat=model.matrix(cnt~.,data=test1)# make a model matrix from the test data

val.errors=rep(NA,28)# replicate for all val.errors from NA to 28
for(i in 1:28){# for all i from 1 to 28
  coefi=coef(fit.best,id=i)# extract the coefficients from regfit.best with increasing nr. of pred.
  pred=test.mat[,names(coefi)]%*%coefi# get the prediction of the coefficients using test.mat
  val.errors[i]=mean((test1$cnt-pred)^2)# compute the test MSE
  }
val.errors# display all test MSEs
which.min(val.errors)# give the one with the minimal test MSE
coef(fit.best, 16)
plot(val.errors, xlab = "Number of predictors", ylab = "Test MSE")
lines(val.errors)
```

```
## VALIDATON SET APPROACH
fit.best = regsubsets(cnt~., data=train3,nvmax=31)# perform best subset selection on training data
test.mat = model.matrix(cnt~.,data=test3)# make a model matrix from the test data

val.errors=rep(NA,31)# replicate for all val.errors from NA to 31
for(i in 1:31){# for all i from 1 to 31
  coefi=coef(fit.best,id=i)# extract the coefficients from regfit.best with increasing nr. of pred.
  pred=test.mat[,names(coefi)]%*%coefi# get the prediction of the coefficients using test.mat
  val.errors[i]=mean((test3$cnt-pred)^2)# compute the test MSE
}

val.errors# display all test MSEs
which.min(val.errors)# give the one with the minimal test MSE
coef(fit.best, 28)
plot(val.errors, xlab = "Number of predictors", ylab = "Test MSE")#weird
lines(val.errors)
```

XIII. K-fold cross validation for Day and Hour

```
## K-FOLD
k <- 10
set.seed(1)
folds <- sample(1:k, nrow(day), replace = TRUE)
cv.errors <- matrix(NA, k, 28, dimnames = list(NULL, paste(1:28))) #null matrix to store results

for(j in 1:k)  {# define loop for best subset selection on training data
  best.fit = regsubsets(cnt ~.,data = day[folds!=j,],nvmax=28)# best subset selection on training data set
  for(i in 1:28){# define loop for best subset selection on test data
    pred = predict(best.fit,day[folds==j,],id=i)# prediction/model testing of best subset selection
     cv.errors[j,i] = mean((day$cnt[folds==j]-pred)^2)# give cross-val. 10x29-matrix as test MSE
  }
}

mean.cv.errors = apply(cv.errors,2,mean)
which.min(mean.cv.errors)
plot(mean.cv.errors, xlab = "Number of predictors", ylab = "Test MSE")
lines(mean.cv.errors)
```

```
## K-FOLD
k <- 10
set.seed(1)
folds <- sample(1:k, nrow(hoursub), replace = TRUE)
cv.errors <- matrix(NA, k, 31, dimnames = list(NULL, paste(1:31))) #null matrix to store results

for(j in 1:k)  {# define loop for best subset selection on training data
  best.fit = regsubsets(cnt ~.,data = hoursub[folds!=j,],nvmax=31)# best subset selection on training data set
  for(i in 1:31){# define loop for best subset selection on test data
    pred = predict(best.fit,hoursub[folds==j,],id=i)# prediction/model testing of best subset selection
    cv.errors[j,i] = mean((hoursub$cnt[folds==j]-pred)^2)# give cross-val. 10x29-matrix as test MSE
  }
}

mean.cv.errors = apply(cv.errors,2,mean)
which.min(mean.cv.errors)
plot(mean.cv.errors, xlab = "Number of predictors", ylab = "Test MSE")
lines(mean.cv.errors)
```

XIV. Filtering hours that we think are less crucial

```
#filtering the hours that we dont think is so important due to computatinal time
hoursub <- hour %>% filter(hr != 0 & hr != 1 & hr != 2 & hr != 3 & hr != 4 & hr != 5 & hr != 20 & hr != 21 & hr != 21 & hr != 22 & hr != 23)
hoursub <- hoursub[,-3] #we dont consider the month, only the season, since R takes a long time and season and month are correlated
```

XV. Poisson and Quasi-Poisson regression for Day and Hour

```
day [c("season", "yr", "mnth", "holiday", "weekday", "workingday", "weathersit" )] <- lapply(day[c("season", "yr", "mnth", "holiday", "weekday",
                                                                                                   "workingday", "weathersit")], factor)
qcc.overdispersion.test(day$cnt, type = "poisson")

fit <- glm(cnt ~ dteday + season + holiday + workingday + weathersit + temp + atemp + hum + windspeed, data = day, family = poisson)
summary(fit)

poi.mod <- glm(cnt ~ dteday + season + holiday + workingday + weathersit + temp + atemp + hum + windspeed, family = poisson, data = day)
exp(poi.mod$coef)

predict(poi.mod, type = "response")

model <- glm(cnt ~ dteday + season + holiday + workingday + weathersit + temp + atemp + hum + windspeed, family = quasipoisson, data = day)
exp(model$coef)

fit <- glm(cnt ~ dteday + season + holiday + workingday + weathersit + temp + atemp + hum + windspeed, data = day, family = quasipoisson(link = "log"))
summary(fit)

predict(model, type = "response")
```

```
library(readxl)
hour <- read_excel("Master/Data Science and Machine Learning/Project/Bike-Sharing-Dataset/hour.xlsx")
View(hour)

hour [c("season", "yr", "mnth","hr", "holiday", "weekday", "workingday", "weathersit" )] <- lapply(hour[c("season", "yr", "mnth", "hr", "holiday", "weekday",
                                                                                                          "workingday", "weathersit")], factor)
qcc.overdispersion.test(hour$cnt, type = "poisson")

fit <- glm(cnt ~ hr + season + holiday + workingday + weathersit + temp + atemp + hum + windspeed, data = hour, family = poisson)
summary(fit)

poi.mod <- glm(cnt ~ dteday + season + hr + holiday + workingday + weathersit + temp + atemp + hum + windspeed, family = poisson, data = hour)
exp(poi.mod$coef)


predict(poi.mod, type = "response")

model <- glm(cnt ~ hr + season + holiday + workingday + weathersit + temp + atemp + hum + windspeed, data = hour, family = quasipoisson)
exp(model$coef)

fit <- glm(cnt ~ hr + season + holiday + workingday + weathersit + temp + atemp + hum + windspeed, data = hour, family = quasipoisson(link = "log"))
summary(fit)

predict(model, type = "response")

```