Seoul Bike Sharing System: A prediction on the rented bikes count using weather and time information

Georges Taing B0039821, Stephen Somsack B00398581

I. Introduction

In the past decades, bicycle sharing systems or public bike share schemes have become a popular alternative to other public shared transports (bus, subway, etc.) in cities. Not only do public bike share systems present benefits by reducing vehicle emissions, traffic congestion and fuel consumption, the users of public shared bikes also find health benefits and transport flexibility (short-term rental and checkout, "as-needed" service) along with financial savings.

However, although bike sharing systems can save its users' time thanks to an easy-to-use rental service, finding an available and accessible bike to use is not necessarily an easy task and can be time-consuming. Hence the importance for a public bike sharing rental service to know and keep track of the bikes' availability, such as the total number of available bikes at a given place or city, so that the supply of rental bikes can be handled more easily.

In this project, we worked on a problem that aims at predicting the bike sharing demand of the Seoul Bike Sharing system (Seoul, South Korea). The idea in this project is to predict the bike sharing demand using the weather information.

We found that this regression problem is an interesting problem to work on because of its concrete application (bike sharing systems) and because of the availability of an interesting dataset to work on (the latter will be described in detail in the next section).

I.2. Related Works

These last years, some studies have attempted to conduct bike related forecasting using different types of prediction models. Singhvi et al. [1] presented a log-log regression model to predict the bike usage during morning peak hours in New York City. Taxi usage, weather, and spatial factors were also taken into account to improve the accuracy. On the same dataset, Wang [2] compared the regional bike rental demand forecasting results by using various machine learning models and found that Neural Network-based and tree-based models generally reach most high prediction accuracy

A forecast of the bike rental demand in Washington D.C in 2017. The data is a combination of usage patterns and weather data. The main models used were multiple linear regression and random forest. They found the classic multiple linear regression was very inaccurate and that is not appropriate for their predictions. While a random forest-based model was more accurate and showed a great accuracy [3].

Deep Learning models were also considered by many researchers. Lin et al. [4] proposed a Graph Convolutional Neural Network with Data-driven Graph Filter model which is noteworthy in learning hidden correlations between stations in order to predict hourly demand for each station in the bike-sharing network. Those approaches provide often better results nowadays but in order to apply deep learning models, more computational time and power are required.

II. Dataset

II.1. Presentation

The dataset we worked on in this project is a public dataset found on UCI Machine Learning Repository's website (https://archive.ics.uci.edu/ml/datasets/Seoul+Bike+Sharing+Demand#).
The dataset contains a count of public bikes rented hourly in Seoul Bike sharing System with the corresponding weather data and holidays information. It is a multivariate dataset with a total number of 14 attributes. The attributes in the dataset are "integer" and "real" types, and are as follow:

Date	Dew point temperature (Celsius)
Rented bike count	Solar radiation (MJ/m2)
Hour (hour of the day)	Rainfall (mm)
Temperature (in Celsius)	Snowfall (cm)
Humidity (%)	Seasons
Wind speed (m/s)	Holiday (Holiday/No Holiday)

Visibility (10m)	Functional Day (Non Functional hours/Functional Hours)

Table 1. Features in the dataset

These features are interesting for the purpose of the study since we want to predict the bike sharing demand using the weather conditions of Seoul.

II.2. Exploratory Data Analysis (EDA)

In this section, we will present quantitative and visual methods to explain different aspects of the data.

First, let's have a general view on the correlation of the different features of the dataset. For that, we plotted the Pearson correlation coefficients which measures the linear correlation between two features. The covariance is measured and normalised, giving a value between -1 (blue) and 1 (red).
A value of 1 means the two features are correlated positively, 0 means the two features are not correlated, and -1 means the two features are correlated oppositely.

The Pearson correlation coefficient has been calculated for all the features with numerical values (all of them except "date" and "holiday")

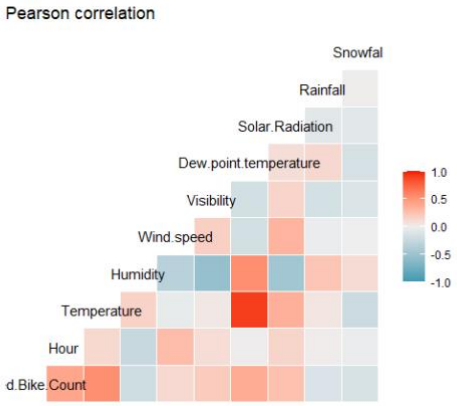

Figure 1. Pearson correlation matrix of the numerical features

Without much surprise, the feature "temperature" and "dew point temperature" are highly correlated, and the same behaviour is shown between "humidity" and "dew point temperature". The features "Hour" and "rented bike count" are also very correlated.

II.2.a. 'Date' feature

We had the date under the following format "Day/Month/Year" as a character attribute. With the help of the 'lubridate' package, we thought that it would be interesting to create new features that will represent the day of the week or the month when bikes are rented.

With these features, we could plot the number of rented bikes per hour during the day per day of the week or per month. In other terms, it allowed us to see the evolution of bike demand in Seoul at different scales of time (hourly, daily, monthly) throughout the year.

In the following graph, we plotted the average number of rented bikes as a function of the hour of the day, for each day of the week.

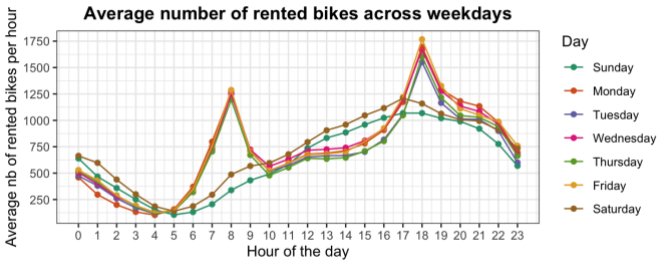

Figure 1. Average number of rented bikes hourly for everyday of the week

We saw that the rents follow the same trends during workdays from Monday to Friday: the number of rented bikes increases when people go to work (7 - 8 a.m.) or when they go back home (6p.m.). The fact that we don't observe those patterns during the weekend supports the behaviour that the number of rented bikes rises because people are working.

II.2.b. 'Holiday' feature

In the following histogram, we plotted the number of rented bikes as a function of the hour of the day, with a holiday vs. no holiday comparison.
The distribution supports the idea that the peaks around 7-8 a.m. and around 6 p.m. are respectively due to people going to work and going back from work during "no holiday" days. On the other hand, the days that are holidays do not present any noticeable peaks.

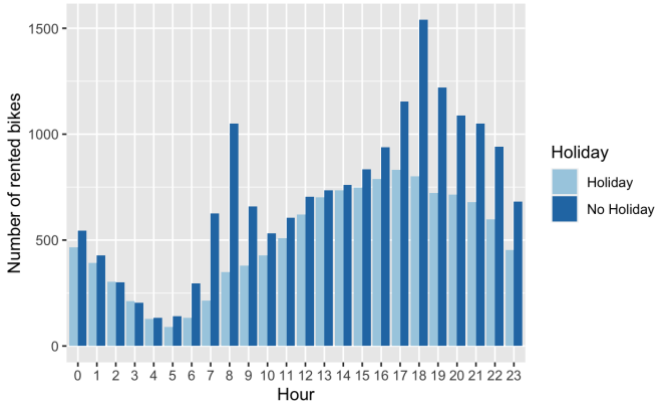

Figure 2. Number of rented bikes during holiday/non holiday periods

II.2.c. Months and Seasons

The two following graphs present respectively the average number of rented bikes for each month of the year and the average number of rented bikes for each season, both as a function of the hour of the day. We can notice that the number of rentals is higher during warmer periods of the year (Summer, Spring and Autumn), compared to Winter for which the number of rented bikes is almost ten times lower. It is understandable since people tend to use bikes with good weather conditions (no frozen routes, etc.).

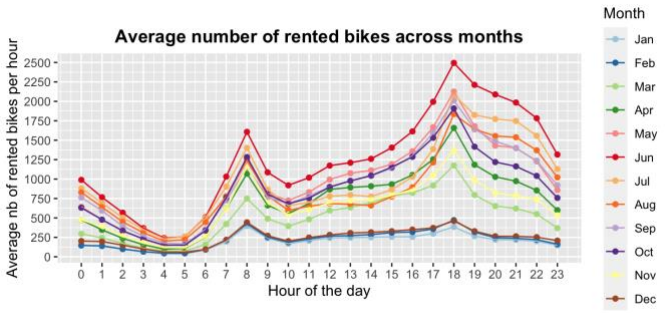

Figure 3. Average number of rented bikes hourly for every month of the year

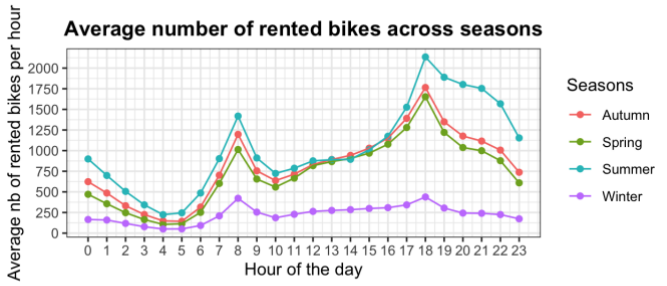

Figure 4. Average number of rented bikes hourly for every season of the year

III. Methodology

III.1. Feature scaling

We did not apply any feature scaling because the models we are using are not distance-based. Indeed, the algorithms that we will consider in our study are regression models (linear, Poisson, quasi-Poisson, negative binomial) and Random Forest.

III.2. Outliers

We did not notice really obvious and problematic outliers in the data. Even though some records showed a high number for the count of rented bikes, we kept them because they might have represented special occasions (New Year, National Day, Christmas or other Holidays…). Therefore, we didn't modify any records in our dataset.

III.3. Feature Selection

III.3.a. Boruta algorithm

We found it interesting to explore some feature selection algorithms and we chose to use the Boruta algorithm. Basically, the Boruta algorithm creates "shadow features" which are a randomized version of the features and make them compete with the initial features to evaluate the feature importance. Feature importance is evaluated by choosing a classifier (for instance, Random Forest) and we perform a feature importance measure (such as the Mean Accuracy Decrease*).

We compare the feature importance measures and a feature is relevant if it does better than the best randomized feature. We say that a feature is a "hit" when its feature importance is above the best the shadow feature or "no hit" when this is not the case. Finally, we repeat the previous process n times to have n repetitions that follow a normal distribution.

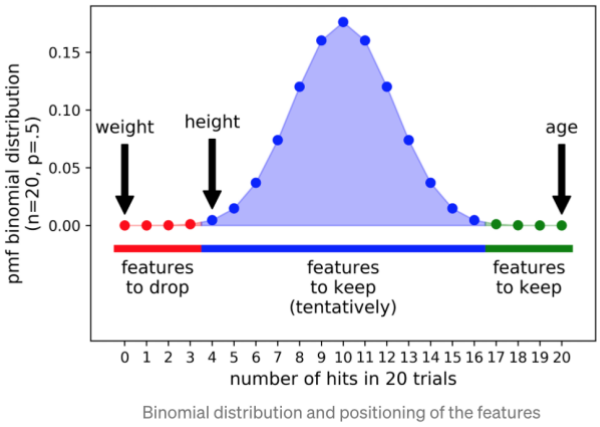

Figure 5. 3 areas for the feature selection

We have 3 areas: the left tail (features to drop), the right tail (features to keep) and the bell (we have the choice to keep them or not).

*Mean Decrease Accuracy: How much the tree nodes that use a specific feature (in other words, when split by this feature) reduce the accuracy loss (on average = its means across all trees in the forest)

For visual purposes, the graph representing the importance score for each feature given by the Boruta algorithm has been added to the annex of this report (**Section VI**).

The Boruta algorithm did not highlight any irrelevant features in our dataset. So, we did not drop any features. However, the findings from the Boruta algorithms told us that the "Hour" feature has the strongest feature importance.

III.3.b. Recursive Feature Elimination (RFE)

We also used another algorithm for feature selection, the Recursive Feature Elimination (RFE) which is quite popular. Technically, RFE is a wrapper-style feature selection algorithm that also applies filter-based feature selection internally. The algorithm works by searching for a subset of features by starting with all features in the training dataset and successfully dropping features until the desired number is reached. This is achieved by fitting a chosen machine learning algorithm used in the core of the model, listing features by importance, eliminating the least important features, and re-fitting the model.

This method was interesting because we wanted to know the optimal number of features to keep in order to build our model which was 31. We also got the top 5 most relevant features (that was also obtained with Boruta): Hour, Humidity, Temperature, DaySunday, Rainfall.

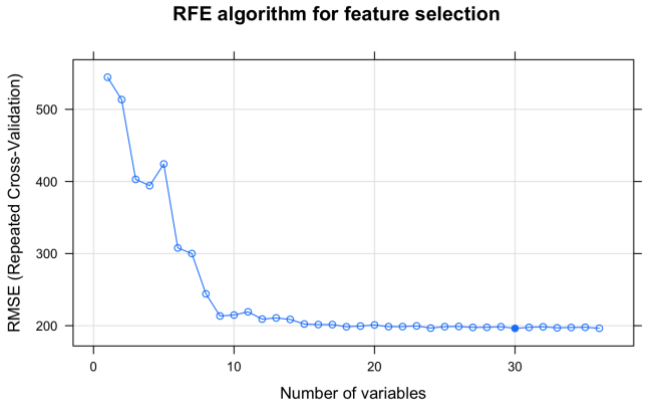

Figure 6. RMSE as a function of the number of variables

III.4. Data splitting

To evaluate and to compare our models, we split the dataset into two sets: a training set (80%) and a testing set (20%). The first set was used for training and fitting the model on the data. The second set was used to evaluate the performance of the models with metrics such as RMSE (Root Mean Squared Error) or RMSLE (Root Mean Squared Log Error).

III.5. Model selection

Classic parametric tests do not work on count-based data because having continuous data is required. We need to use a generalized linear model (GLM). GLMs is a flexible generalization of the classic linear regression that allows for response variables that have error distribution models other than a normal distribution. The most common approaches for handling count-based data are Poisson and Negative Binomial regressions.

III.5.1. Poisson regression

Poisson distribution is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time or space if these events occur with a known constant mean rate and independently of the time since the last event.
The probability mass function is given by:

$$P_X(k) = \frac{e^{-\lambda t} * (\lambda t)^k}{k!}$$

In a Poisson regression model where the event rate λ is not constant, the job is to fit the observed counts vector y to the regression matrix X (matrix representation of the weather features) by using a function expressing the rate vector λ as a function of the regression coefficients β and the regression matrix X. The mathematical function used in a Poisson regression is:

$$\lambda = e^{X\beta}$$

Poisson regression is a Generalized Linear Model to model count-based data. The Poisson regression has the following assumptions:

- The residuals follow a Poisson distribution
- We model ln(Y) with Y the number of rented bikes per hour, as linear function
- The mean (residuals) is equal to the variance (residuals)

III.5.2. Negative Binomial regression

Though Poisson regression is a quite good model for count-based data, the performance of the model is lowered by the strict assumption (criterion) that the variance is equal to the mean (equi-dispersion assumption). However, in real-world data, the variance is often greater than the mean (overdispersion). Negative binomial is a generalisation of it
In a Negative Binomial regression model, the variance is expressed as function of a new parameter α, where the general form is:
Variance = mean + α * mean ^{p}
We decided to use this model for its alleged better performance

The negative binomial regression model has been implemented by using the MASS package on R.

III.5.3. Random Forest

Random forests are an ensemble learning algorithm for both classification and regression. The main idea behind Random Forest is building many decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees. In our study, we used of the Random Forest algorithm, which is generally a powerful model even without any tuning of the hyperparameters. We tried to tune some

hyperparameters such as the number of trees (ntree) and the number of variables randomly sampled as candidates at each split (mtry) with a grid search.

IV. Results

IV.1 Models' performance

We evaluated the different models with the testing set on 2 metrics. The first one is RMSE which computes the root of the sum of the squared errors between the predictions and the actual outputs. The mathematical expression of the RMSE is given by:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}$$

This metric allows us to consider Linear Regression even if it is not a good model for count data because it can output negative values. Indeed, some values in our predictions with Linear Regression are negative. That is the reason why, the Linear Regression model can't have a RMSLE which uses the log function; therefore, it can't be calculated.
The mathematical expression of the RMSLE is given by:

$$RMSLE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(log(\hat{y}_i + 1) - log(y_i + 1))^2}$$

The interest in RMSLE is that it penalises more predictions that are below the true output and that is relevant in a system that aims at predicting the number of required bikes for the city. It is better to have too many available bikes than not enough for the inhabitants.
With both metrics, Random Forest is by far the best model with the lowest error.

	RMSE	RMSLE
Poisson regression	371	0.733
Negative Binomial regression	428	0.811
Linear regression	416	NaN (impossible to compute)
Random Forest	187	0.693

Table 2. RMSE and RMSLE for each model used.

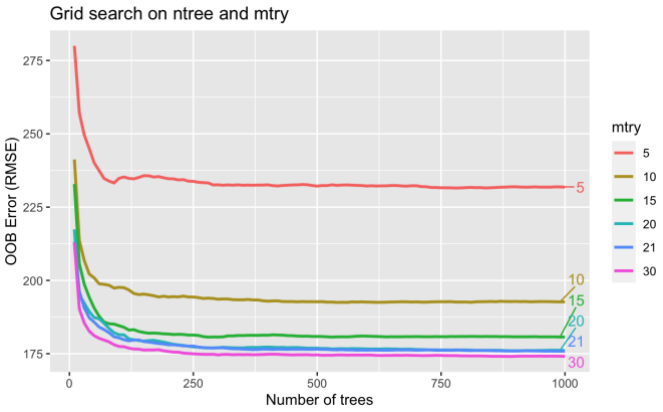

Figure 7. RMSE for each mtry

With the Random Forest model, we found that the optimal mtry was 30 which is coherent with the optimal number of features (31). Basically, the Random Forest model was already "optimal" when we used it out-of-bag.

IV.2. Residuals Analysis

In the context of a regression problem, we can perform a residual analysis on the Poisson and Negative Binomial models. We used the DHARMa package that handles residual analysis for GLMs.

IV.2.1 Poisson regression

In the QQ plot of the residuals, we expect the observed residuals to follow a y=x function represented by the red line on the graph. We clearly saw that our values were close to 0.0-0.4 and 0.6-1.0 and that residuals were not on the red line. That means that a Poisson regression model was not a good fit because the data are overdispersed.

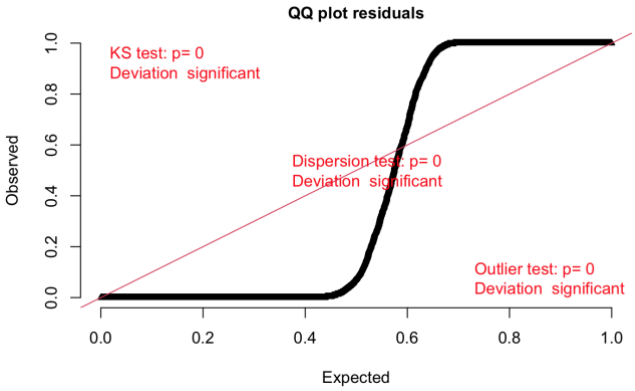

Figure 8. Graph of the QQ plot residuals for the Poisson regression

Another hint for overdispersion is the deviance that we obtained from our model. A rule of thumb is to have a deviance not too far from the value of degrees of freedom. The Poisson model has a residual deviance far greater than the number of degrees of freedom (1155312 for 6983).

```
    Null deviance: 3991664  on 7006  degrees of freedom
Residual deviance: 1155312  on 6983  degrees of freedom
```

The DHARMa package can also compute the overdispersion. We computed a dispersion of 189 (we need a value below 1) for a p-value < 2.2e-16.

IV.2.2. Negative Binomial

We saw that the QQ plot of the residuals was better with a Negative Binomial model. Indeed, the observed residuals fitted better the red line.

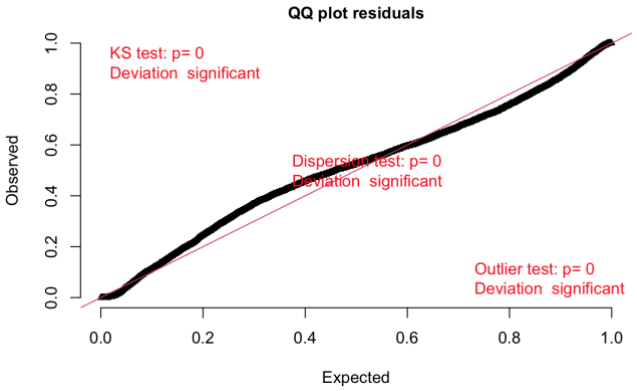

Figure 9. Graph of the QQ plot residuals for the Negative Binomial regression

The deviance was also better. We had a residual deviance of 7246.1 for 6984 degrees of freedom which was quite close.

```
    Null deviance: 23433.6  on 7006  degrees of freedom
Residual deviance:  7246.1  on 6984  degrees of freedom
```

Moreover, the dispersion computed with the DHARMa package is equal to 0.527 for a p-value < 2.2e-16. The Negative Binomial model considered better the overdispersion in the data.

V. Discussion

Our study aimed at forecasting the number of rented bikes during the day in the city of Seoul. The project began with an exploratory dataset analysis where we studied various statistical characteristics on the features. We did not notice any obvious outliers that needed us to drop some records. Moreover, the algorithms that we used did not require a feature scaling. The exploration was also an opportunity to apply feature engineering, the date feature was a core component for creating new features related to the day of the week and the month of the year. Another key step in our work was to compare the feature importance of our variables by using the Boruta technique and the RFE algorithm. It allowed us to highlight the most relevant features and to quantify the impact on the predictions of each one of them. We proposed many methods that were based on Generalized Linear Models and the Poisson distribution as we had a count-based data as output. Testing a conventional linear regression model was interesting because it showed the problem that the output of linear regression can be negative which is irrelevant for a count-based data. Besides, to evaluate and to compare our models, we used two metrics for regression: RMSE and RMSLE. The second metric was really pertinent because it penalises predictions that are below the true output and that is relevant for a renting system. For the results, the GLM showed poor performance when compared to a Random Forest model even without hyperparameter tuning.

Some proposals for future study on this project would be to try to create some neural networks models considering that many recent studies on forecasting displayed great performance with them. Building a Recurrent Neural Network (RNN) based on a Long Short-Term Memory (LSTM) architecture would be interesting. Indeed, LSTM networks are well-suited for working on time series data which is the case here.

References

[1] D. Singhvi et al., “Predicting Bike Usage for New York City’s Bike Sharing System,” 2015

[2] Wang, W. « Forecasting bike rental demand using New York Citi Bike data, » M.S Thesis, School of Computing College of Science of Health, TU Dublin, Dublin, Ireland, 2016.

[3] Y. Feng and S. Wang, “A forecast for bike rental demand based on random forests and multiple linear regression”, presented at the IEEE/ACIS 16th International Conference on Computer and Information Science, Wuhan, China, May 24-26, 2017.

[4] L. Lin, Z. He, and S. Peeta, ‘Predicting station-level hourly demand in a large-scale bike-sharing network: A graph convolutional neural network approach’, Transportation Research Part C: Emerging Technologies, vol. 97, pp. 258–276, Dec. 2018.

VI. ANNEX

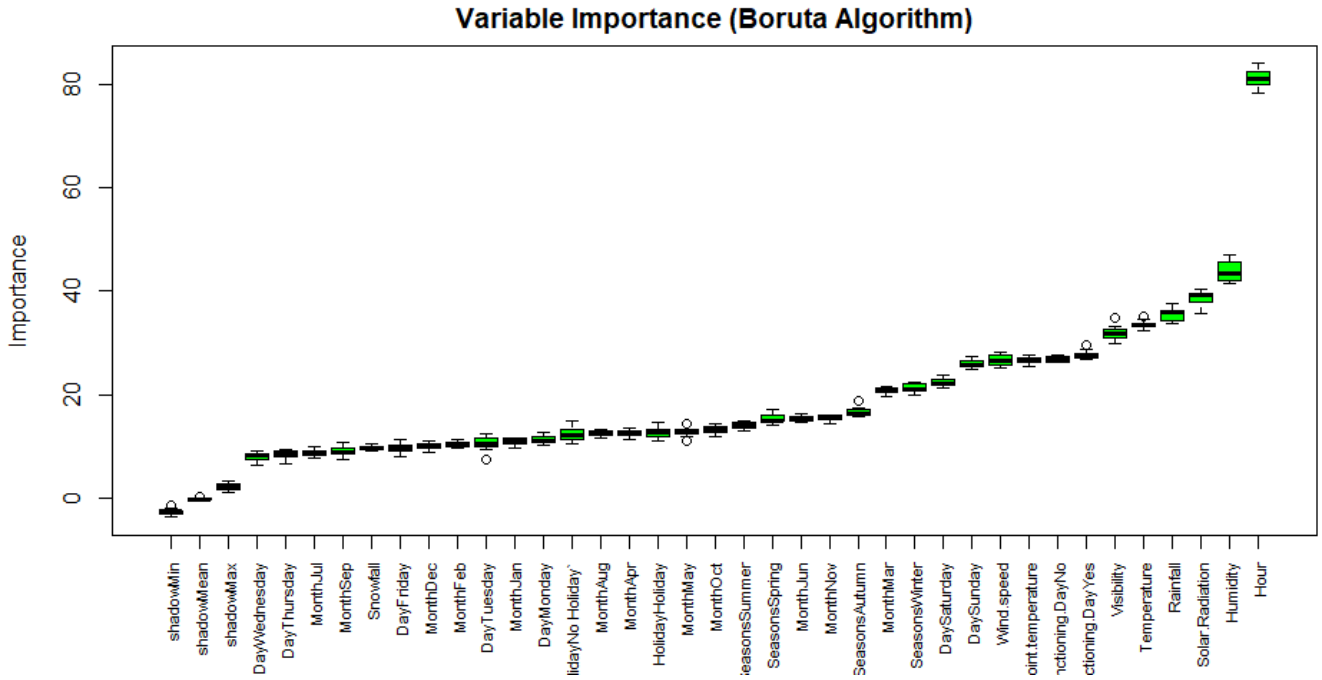

Figure 10. Importance of the features according to the Boruta algorithm