DPI Project: London bike rentals

Davide Cremonini - 14412

GitLab repository:

https://gitlab.inf.unibz.it/Davide.Cremonini/pdi2020-assignment-project

Table of contents

1. Motivation scenario

Nowadays, as urbanization reaches unprecedented levels, road congestions and air pollution have become serious issues. Among the many solutions that have emerged in recent years, perhaps the most promising is represented by bike sharing. To be clear, bike sharing refers to rental schemes, whereby civilians can pick up, ride and drop off bicycles at numerous points across the city.

Bike sharing not only represents one of the most interesting and promising method of reducing city traffic, but it can also benefit the economy (impact on businesses and neighbourhood, reduced expenditure on healthcare, as stated in this article: https://medium.com/urbansharing/the-economic-benefits-of-bike-sharing-f69c230e5a9d) and the personal health of the customers (as stated in this research paper https://www.sciencedirect.com/science/article/pii/S0160412017321566).

Starting from this good points, London is one of the cities most willing to use the bike rental system as a solution for air pollution and road congestions. Therefore, public authorities and bike rentals service managers would like to monitor past data in order to obtain useful information to improve the system, involving more and more citizens and tourists and improve the general living conditions of the city. In addition, by identifying possible patterns in the use of the service, they would like to have a program that can give an estimate of the number of bikes that will be rented in the short term, based on information about the day and weather forecast.

For this purpose, I am responsible for the realization of such analysis and software for the forecasting of future rentals. I set myself the goal of analysing the data regarding the number of bikes that were rented in London (from January 2012 to January 2015). Moreover, my work will be based on the hypothesis that the number of bike rentals are related to some variables like the hour of the day, period of the year and weather condition, and I will try to train a Machine Learning model to try to predict the future bike shares.

The analysis of past data, the discovery of some pattern and the predictions of the future use could help the bike sharing general management (advertising, incentives or discounts for improve low bike shares, take full advantage of predicted high shares in a specific period of the year, improve service and bike availability, and many other useful management choices). Some questions that the city administration wants to have answered, and that will be answered by statistical analysis of past data are:

- Are bike shares influenced by the type of day and period?
- How are bike shares distributed over the hours?
- How is the request for bike rentals during the periods of the year?
- How are bike shares distributed over weather condition?
- How do temperature and wind affect the number of bike rentals?

2. Data description

For the realization of the project, three different types of data are to be taken into consideration:

Data for analysis and statistics:
- Bike rentals usage
- Historical weather conditions

Data for future prediction:
- Weather forecasts

2.1 Bike rentals usage

Data concerning the bike rentals usage in the city of London is retrieved from the government data source (https://cycling.data.tfl.gov.uk).

Data covers three complete years, from January 2012 to January 2015, but is spread across several independent files of different sizes and covering different time ranges. Summarizing, there are 52 files, with the smallest of 8 MB and the largest of 132 MB, reaching a total of 3.35 GB.

The following table presents the columns together with data types of the common schema that is shared among the 52 files:

Column	Data type
Rental ID	Integer
Duration	Integer
Bike ID	Integer
End Date	String
EndStation ID	Integer
EndStation Name	String
Start Date	String
StartStation ID	Integer
StartStation Name	String

After a quick exploratory analysis, the data results to be very dense, with almost no missing values. The only exception concerns the variables Endstation ID and EndStation name, where the data is sometimes absent.

Although the table contains interesting information such as the station of departure or duration of the trip, which could be the subject of study for a future project, the objective of the analysis of the realized project focuses on the use of rental bikes at a certain time, combining it with information about the weather. For this reasons, as we will see in the implementation section, data is grouped together by hour and the number of bike rented will be counted.

2.2 Historical weather conditions

Data concerning the historical weather conditions regarding the city of London is retrieved from OpenWeatherMap.org (https://openweathermap.org/history-bulk).

Data contains hourly weather information in London and covers 15 complete years, from 2005 to 2019, with a total of 131601 observations and 26 variables. Information is organized in the following way:

Column	Data type	Column	Data type
Unnamed:0	Integer	grnd_level	Float
dt	Integer	humidity	Integer
dt_iso	String	wind_speed	Float
timezone	Integer	wind_deg	Integer
city_name	String	rain_1h	Float
lat	Float	rain_3h	Float
lon	Float	snow_1h	Float
temp	Float	snow_3h	Float
feels_like	Float	clouds_all	Integer
temp_min	Float	weather_id	Integer
temp_max	Float	weather_main	String
pressure	Float	weather_description	String
sea_level	Float	weather_icon	String

Also this dataset results to be very dense. The exceptions are represented by the variables sea_level, grnd_level, rain_1h, rain_3h, snow_1h and snow_3h, where data is almost exclusively absent.

2.3 Weather forecasts

Recalling that one of the objectives of the project is also training a ML model to predict future shares, we need future weather information. For this reason, we retrieve those information again from OpenWeatherMap.org, which offers API to gather future weather information. More specifically, it offers 5 days forecast at a 3 hour interval (in JSON format): https://openweathermap.org/forecast5.

The response of the API call will then be appropriately processed within the program, as we will see in details in the appropriate section.

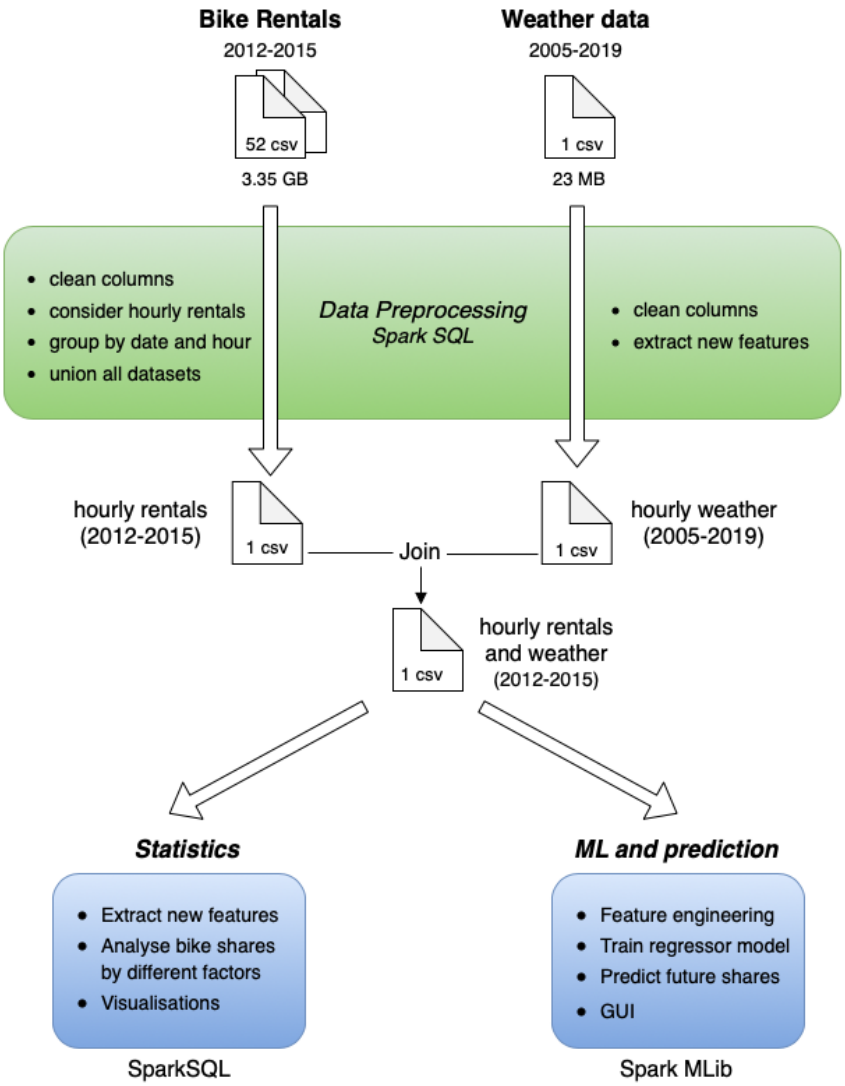
2.4 Data quality

The various datasets does not contain particular data quality issues. The only issues derives from the fact that the three types of data should be combined in the application, needing therefore to operate on some data to correctly merge and join them.

3. Architecture of solution and technology used

3.1 General Architecture

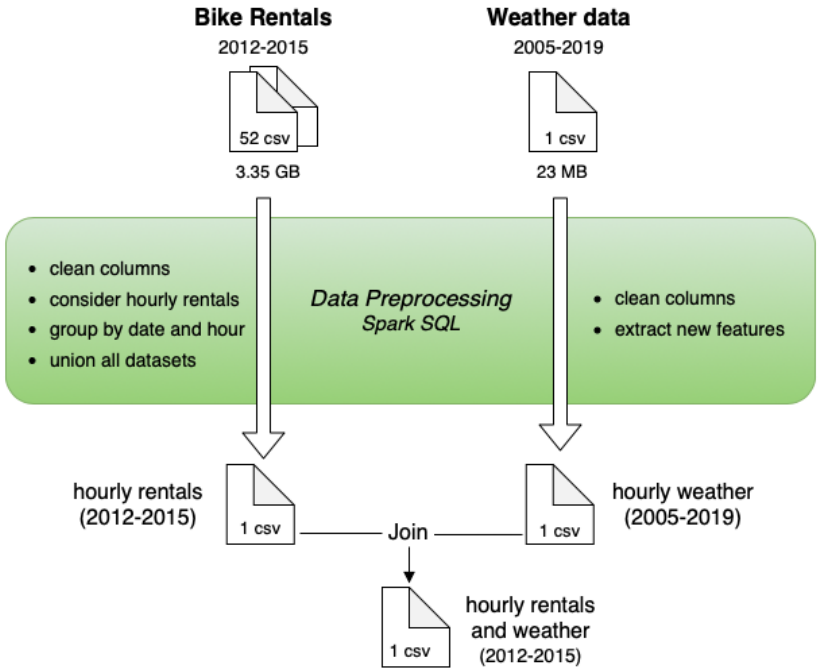
In this section, we provide more details on the description of the overall architecture of the solution. The project structure can be divided into three main parts, each one responsible for a different operation in the overall architecture. The solution architecture and workflow can be graphically represented by the following diagram:



The overall project is developed using IntelliJ as IDE. The technology used is Spark (using Spark SQL and Spark MLib implemented via Java as programming language). Moreover, Java is also used for developing a Graphical User Interface for predicting and showing future bike rentals. Finally, Google Data Studio is used to produce visualizations of the statistics summaries produced.

In the following subsections, we provide more details on the workflow and architecture of each of the different three phases of the project.

3.2 Data Preprocessing



The preprocessing phase takes advantage of the Spark SQL library, in particular using its core data structure: typed Datasets, which are distributed collections of structured data, storing Java-Bean compliant and serializable objects, and more specifically Dataframes, particular Datasets organized into named columns. The Dataset API is available for Java.

There are 5 developed classes responsible for the preprocessing phase:

- Main class that controls the workflow of the process
  - SparkDriver_Preprocessing
- Data preparation class, with principal methods for process datasets and join them
  - DataPreprocess
- Java-Bean for creating typed Datasets
  - HourlyRental
  - HourlyWeather
  - HourlyWeatherAdvanced

3.2.1 Bike Rentals preprocessing

Each of the 52 csv files is preprocessed in the following way:

- Each file is read from the specific directory using the SparkSession read csv function and stored in a Dataframe.
- Each Dataframe is cleaned and processed: the granularity of the considered rentals is increased from minutes to hours, using the withColumn function of the Dataframe class.
- Each Dataframe is then grouped by date and hour, producing therefore hourly bike rentals count.
- All Dataframes are merged together using the union function of the Dataframe class and encoded as Datasets of HourlyRental objects.

3.2.2 Historical weather preprocessing

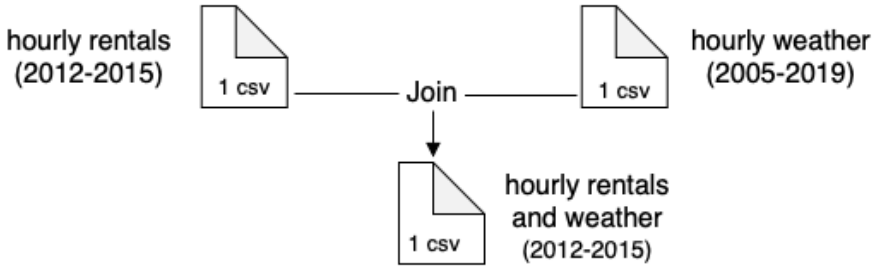
The historical weather data of the city of London is preprocessed in the following way:

- The file is read using the SparkSession read csv function, filtered by selecting the columns of interest and stored in a Dataset of HourlyWeather objects using the appropriate encoder.

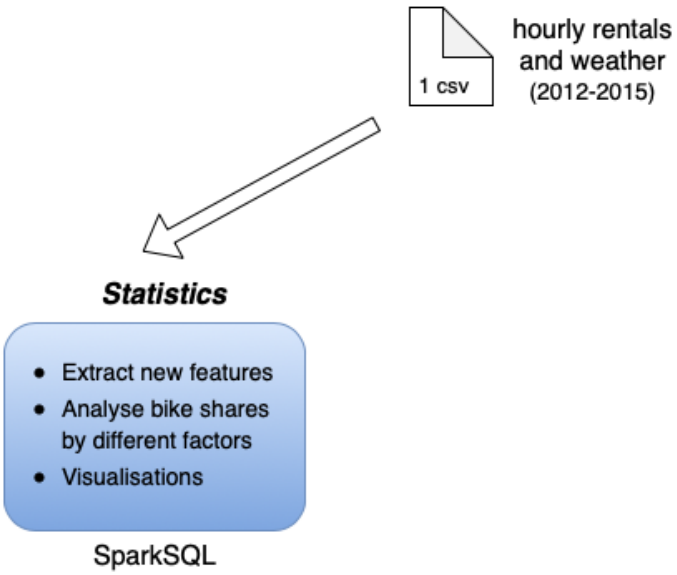
- Some new features are extracted: the Dataset of HourlyWeather objects is transformed into another one of HourlyWeatherAdvanced objects using the map function. The date format is changed to match the same format of the hourly rentals Dataset, and new features are extracted from the original ones, like the day of the week and the type of the day.

3.2.3 Dataset joined

Finally, the preprocessed typed Datasets are joined into a Dataframe using the join function on the date column, to obtain the hourly rentals information together with weather conditions.



3.3 Statistics



From the resulted hourly rentals and weather dataset obtained in the first pre-processing phase, the first objective is to collect useful statistics to discover some patterns and try to answer the initial questions expressed in the Motivation section.

The classes responsible for gathering such statistics are 2:

- Main class that controls the workflow of the process
  - SparkDriver_Statistics
- Data analyser class, responsible for extract new features and statistics
  - DataAnalyser

Again, the statistics analysis phase takes advantage of the Spark SQL library, in particular using Dataframes and associated functions.

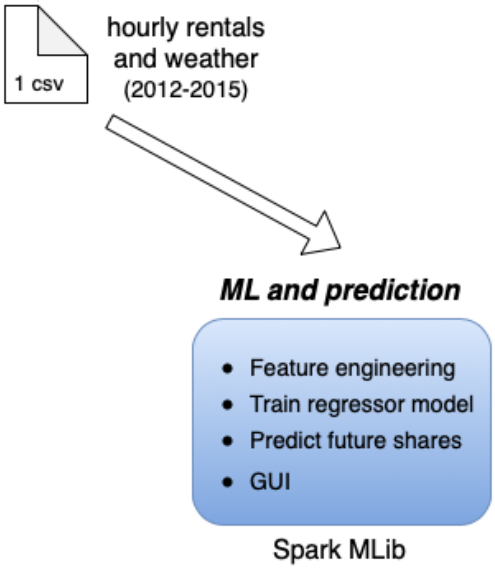
The workflow of this analysis phase is the following:

- With the use of several withColumn and when functions, new columns are extracted from the original ones:
  - String variable representing timestamp is casted, and decomposed into several columns (year, month, day, hour, ecc).
  - Some other features are extracted from the information about months, hours and days, like season, period of the day and type of the day.
  - Information like temperature and wind speed are grouped into ranges.

- Produce several Dataframes containing statistics summary obtained using aggregation functions, filtering and grouping of the Dataframe obtained in the previous point.

- The produced Dataframes are saved as csv files to be visualized using Google Data Studio.

3.4 ML and prediction



The final objective of the project is to predict future bike rentals, by training a ML model that learns from the past data (those who are produced by the pre-processing phase) and predicts bike rentals of the future 5 days.

The technology used in this phase are several. The most dominant one is Spark, taking advantages of both Spark SQL library (to handle the Dataframes and perform feature engineering) and Spark MLib (to train a model and perform predictions). Moreover, since a GUI is developed, other libraries are used like Java Swing components togethers with external libraries like JSON reader (to handle API responses from OpenWeatherMap.org), JXDatePicker and JFreeChart.

The developed classes in this phases are 7:

- Main class that controls the workflow of the process
  - AppManager
- Support classes
  - WeatherForecast: represents an instance of hourly forecast, containing the various variables and methods
  - Regression: responsible for performing the regression (training and prediction)
  - Forecast analyser: responsible for handle the response from the API and create suitable objects
- GUI classes
  - MainPanel
  - IntroPanel
  - DayRentalsPanel

In the following subsections we illustrate the details of the ML pipeline

3.4.1 Feature Engineering

In order to prepare the ML model, the dataset produced in the first phase needs further modifications, and therefore some feature engineering techniques are applied.

The pipeline for preparing the dataset is the following:

Columns	Technique	Description
Hour Month	Cosine and sine calculation	Cyclical variables like hour and month needs to be handled in a particular way. For this reasons, the cosine and sine of their values are computed. To understand the reason, close times like 23 and 00 needs to appear similar, and this similarity is lost by taking their value as it is.

Columns	Technique	Description
Day of week Weather Season Day Period Day type	String indexer + One-hot encoding	Categorical variables needs to be mapped to numerical ones before applying the MLib algorithms. Therefore, they are transformed into integers using the String indexer and then into vectors using one hot encoding.

Columns	Technique	Description
Feels_like Temp Temp min Temp max Pressure Humidity Wind speed	Vector assembler + Standard scaler	Numerical values are normalized, using first a vector assembler to combine the column in a single vector column, and then a standard scaler.

Columns	Technique	Description
All columns produced by the previous transformers	Vector assembler	All column produced by the previous transformers are those used to train a ML model. Therefore they are combined using a Vector assembler into a final **features** column.

3.4.2 Model training

At the end of the previously illustrated pipeline, the Dataframe is ready for the application of Spark MLib algorithms, as it contains two fundamental columns: “count”, that represents the hourly number of bike rentals, and “features”, that contains the vector of the features that are used for training the MLib algorithm.

The ML task we have to perform is **regression**, since we are trying to predict a real value, rather than performing classification. Among the available set of regression algorithms that Spark MLib library offers, the chosen one is **Random forest regression.** Random forests are ensembles of decision trees. In this sense, they combine many decision trees in order to reduce the risk of overfitting, and can be used both for regression and classification.

Before the real prediction of the future 5 days bike shares, we want to evaluate the performance of the Random forest regressor, since our datasets (with more than 26000 observation) allows us to split it into train and test without the risk of loosing accuracy. Therefore, we split the dataset in two sets, with the following distribution: 80% for training and 20% for testing.

After the splitting, the Random Forest regressor is created. In term of parameters, we decided to model it with 50 decision trees with a max depth of the trees set to 10, since they appear to be the parameters that lead to the highest performance.

Finally, we train the regressor using the training set and predict the values of bike shares for the test set. Since the test set is labelled, we can evaluate the performance of the model. The metric used for the evaluation is RMSE, which stands for Root Mean Squared Error and is calculates as:

$$RMSE = \sqrt[2]{\sum_{i=1}^{N} \frac{(Predicted_i - Actual_i)^2}{N}}$$

The result of the evaluation is printed to the standard output by the program. The value resulted to be around 350, which is not bad for the prediction of a value that ranges from 1 to 8000 (even if it could be improved with a more thorough analysis of the performances of the MLib algorithm, with more experiments and a more detailed parameter tuning phase).

3.4.3 Future 5 days prediction

After stating that the model achieves a satisfactory performance, it can be used for predicting the bike shares for the future 5 days.

The first step is to retrieve them from the OpenWeatherMap.org API, and this step is under the responsibility of the ForecastAnalyser class, which makes the request to the website and parses the result, by creating a list of WeatherForecast objects, where each one represents the weather forecast for a specific hour. Moreover, each objects has a variable for storing the associated number of bikes rented, and at the beginning it is set to -1.

The list of weather forecasts is then transformed into a Dataframe, with the use of the RowFactory to manually creating the rows and StructType to define the schema, combined then with the use of the createDataFrame() function of the SparkSession class.

The created Dataframe is not ready for the prediction phase, since it is not in the same format of the Dataframes used to train and evaluate the regressor. Therefore, the Dataframe is transformed by applying the transform() method of the previously created pipeline, to replicate the same chain of transformations.

Finally, the Random Forest Regressor trained and evaluated before, is then used to predict the values of the resulted Dataframe that contains the future 5 days weather forecasts. The bike shares count of the different WeatherForecast objects is updated with the predicted one.

At the end, we obtain a list of hourly weather forecast objects with the associated bike rentals predicted by the Random Forest Regressor. Those objects are then used and presented to the user with a GUI that will be illustrated in the next section.

4. Interface and system functionalities

In this section, we provide details description of the usage of the various tasks of the project, by providing the description of the command line calls and the expected results. Moreover, for the statistics task we provide some visualizations of the obtained datasets, and for the ML task we include some screens of the application.

Since it is a Maven project, the various calls are performed from command line. The initial step is to build the jar file. In this sense, since we include also external libraries (in particular for the realization of the GUI), we added an assembly plugin to include all the necessary external classes. Therefore, after navigating in the project folder, the command to execute at the beginning is the following:

```
mvn clean package assembly:single
```

Note that the project come with the jar file already created. Run the above command if changes to the project are performed.

4.1 Preprocessing

To run the first task of the project, i.e. pre-process the datasets and merge them, the command line call is the following:

```
spark-submit  --class data_analysis.SparkDriver_Preprocessing --
deploy-mode client --master local target/Project-1.0-SNAPSHOT-jar-
with-dependencies.jar
```

When the task is completed, the resulted dataset can be found in the output/preprocessing folder. This file is fundamental since it is the file that will be used as input for the other two steps.
Note that for reason of completeness, a copy of the resulted file named as “rental_weather.csv” is included in the input_data folder.

4.2 Statistics and visualizations

To run the second task for gathering statistical summaries, the command line call is the following:

```
spark-submit  --class data_analysis.SparkDriver_Statistics --
deploy-mode client --master local target/Project-1.0-SNAPSHOT-jar-
with-dependencies.jar <joined_dataset>
```

where `<joined_dataset>` represents the dataset output from pre-processing phase. To use the already included dataset, change this part with `input_data/rental_weather.csv`

The task will create several csv files containing statistical summaries that are then visualized using Google Data Studio, to answer the five questions included in the first section.
We illustrate the resulted visualizations:

Are bike shares influenced by the type of the day and the period?

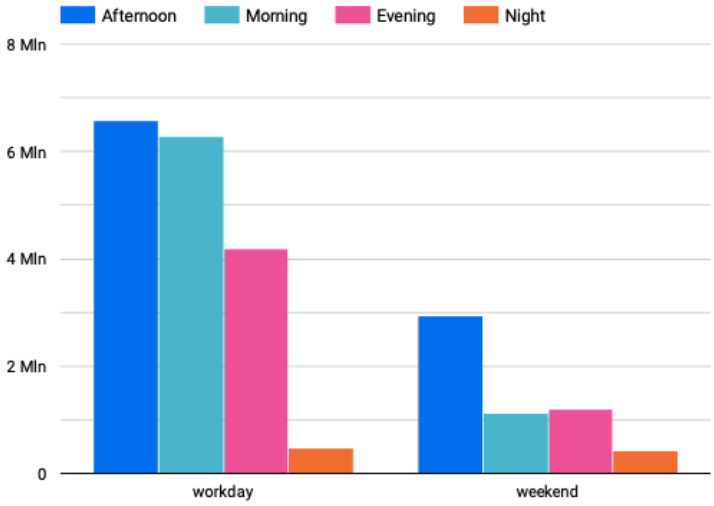
	type_day	day_per...	bikes_rented
1.	workday	Afternoon	6.576.167
2.	workday	Evening	4.195.308
3.	workday	Morning	6.270.199
4.	workday	Night	483.041
5.	weekend	Afternoon	2.937.422
6.	weekend	Evening	1.190.028
7.	weekend	Morning	1.132.210
8.	weekend	Night	415.217

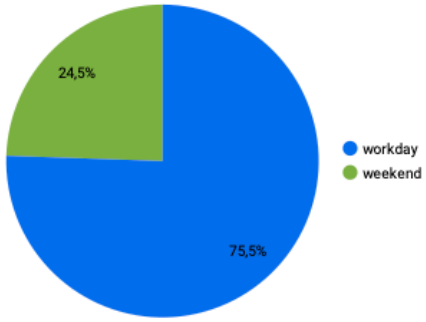
1 - 8 / 8

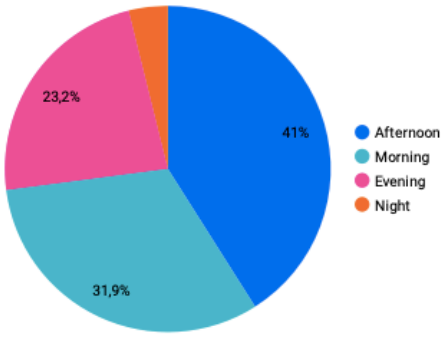
	type_day	day_period	bikes_rented
1.	workday	Afternoon	
2.	workday	Morning	
3.	workday	Evening	
4.	weekend	Afternoon	
5.	weekend	Evening	
6.	weekend	Morning	
7.	workday	Night	
8.	weekend	Night	

1 - 8 / 8

bikes_rented
23.199.592
from 2012 to 2015



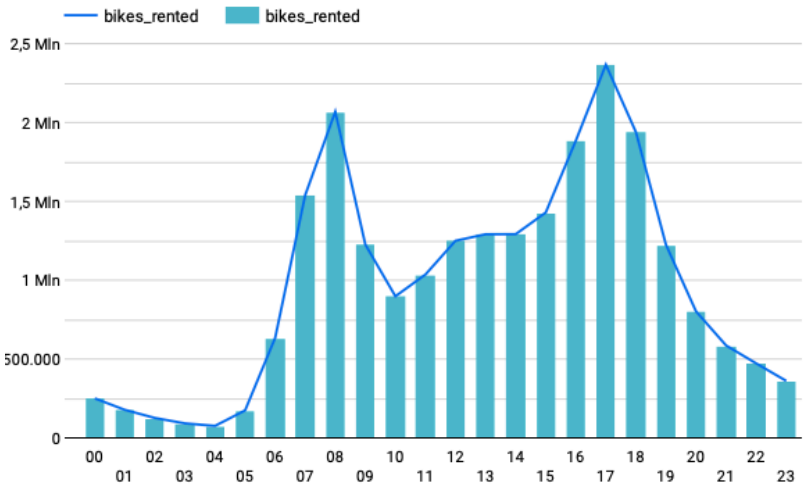


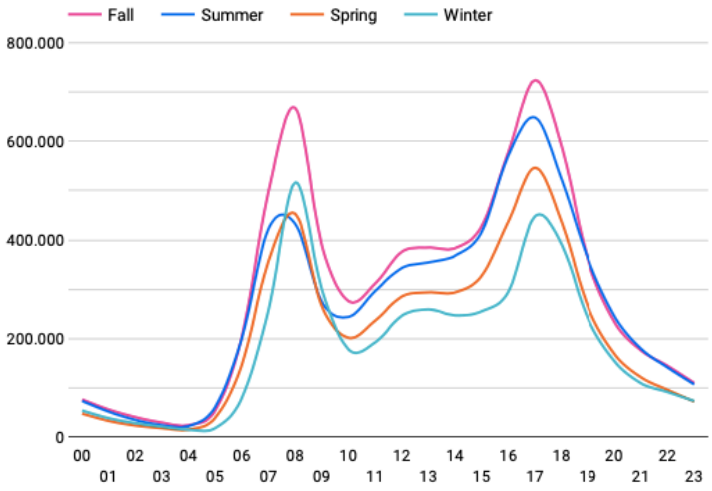


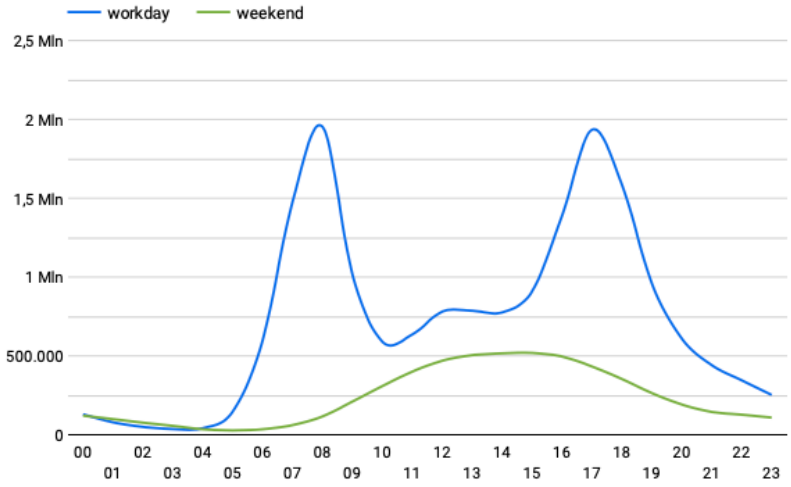
Conclusions:

- The majority of the 24 millions bikes rented were used during workdays. This can lead to the conclusion that bike sharing is used consistently by workers.
- Among the periods of the day, the majority of bikes are rented in the afternoon, both on the weekend than on workdays. An interesting fact, they are widely used in the morning during workdays, while on weekend the second place is occupied by the evening. In both cases, relatively few bikes are rented at night.

How are bike shares distributed over the hours?







Top 10 hours

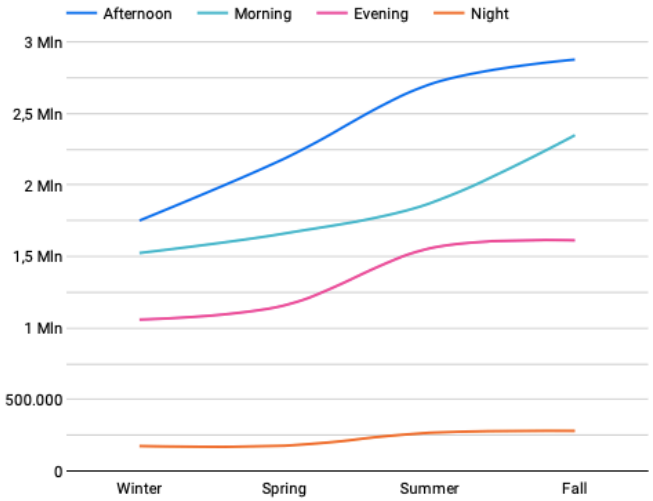
	hour	bikes_rented
1.	17	
2.	08	
3.	18	
4.	16	
5.	07	
6.	15	
7.	14	
8.	13	
9.	12	
10.	09	

1 - 24 / 24

Conclusions:

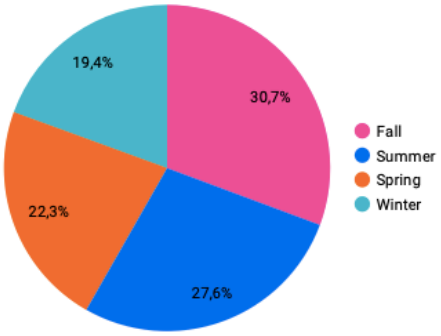
- We have already stated that during weekends and workdays the number of bikes rented is different, but here we obtain new interesting results. The hourly distribution of bike shares indicates that the bike sharing usage during workdays and weekends is very different. During workdays, there are two peaks, at 8 in the morning and at 17-18 in the afternoon. Intuitively, they correspond to the times when people go to or come back from work. As a conclusion, the hour at which bikes are rented is highly influenced by the type of the day.
- On the other hand, the hourly distribution is the same in each season, even if the values are different. It indicates that the hour at which bikes are rented is not influenced by the season.

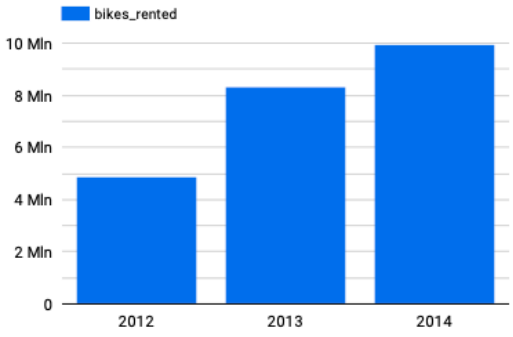
How is the request for bike rentals during the periods of the year?

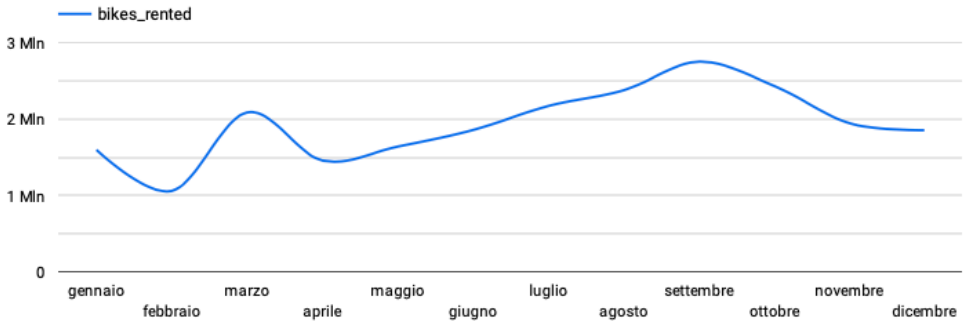


	season	bikes_rented
1.	Fall	
2.	Summer	
3.	Spring	
4.	Winter	

1 - 4 / 4



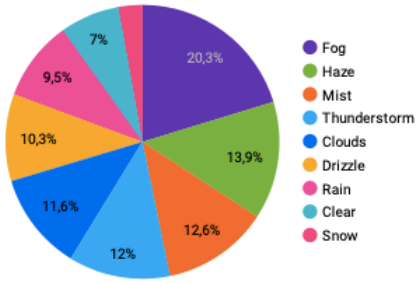




Conclusions:

- The plot about bikes rented in the different years shows a consistent increase of the usage of the service during time.
- We can see in which season the service is majorly used: Fall, followed by Summer, Spring and Winter. Moreover, the ranking is also valid for the different day periods.
- September appears to be the month in which the majority of bikes are rented (it is correspondent to the beginning of school). Another interesting peak is in March, that may indicate an increasing demand once winter terminates.

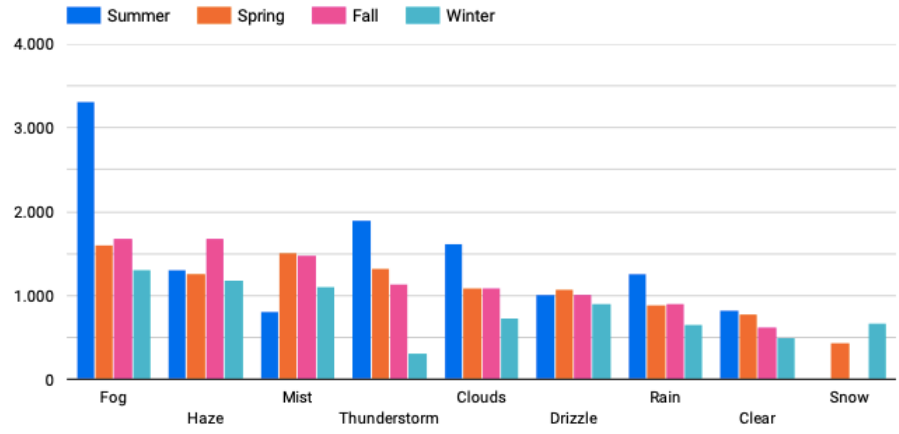
How are bike shares distributed over weather condition?



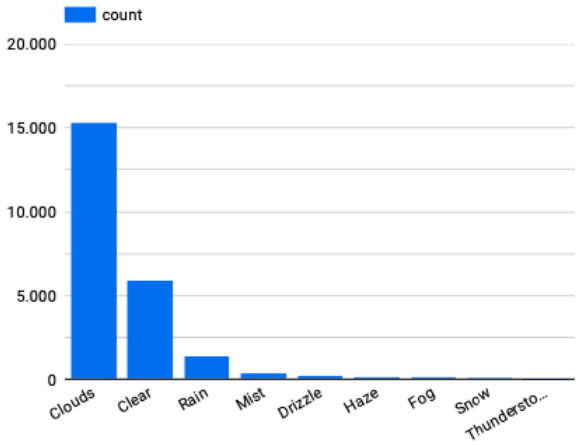
	weather_main	bikes_rented
1.	Fog	
2.	Haze	
3.	Mist	
4.	Thunderstorm	
5.	Clouds	
6.	Drizzle	
7.	Rain	
8.	Clear	
9.	Snow	

1 - 9 / 9

Bike rentals by weather condition and season



Frequency distribution of weather conditions



Conclusions:

- The analysis of bike rentals by weather condition shows unexpected results: on average, days with fog, haze, mist and thunderstorm are the best one. We have to pay attention to these results, as they may be outliers: in fact, if we look at the frequency distributions of the weather conditions, those conditions correspond to the least frequent. By combining those results, we may hypothesize that the low number of occurrences of those weather conditions corresponds to high number of bike rentals, but we need to analyse data in a wider range of years to take conclusions about the effective influence of those conditions.

How do temperature and wind affect the number of bike rentals?

Wind

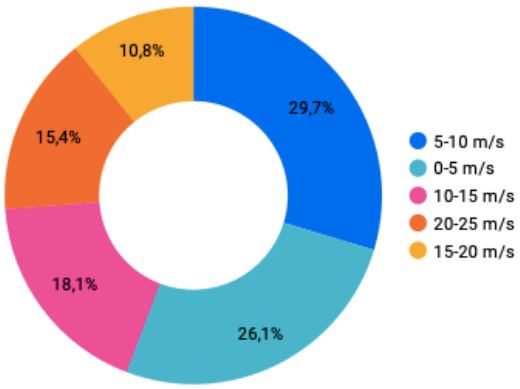
	wind_range	bikes_rented
1.	5-10 m/s	
2.	0-5 m/s	
3.	10-15 m/s	
4.	20-25 m/s	
5.	15-20 m/s	

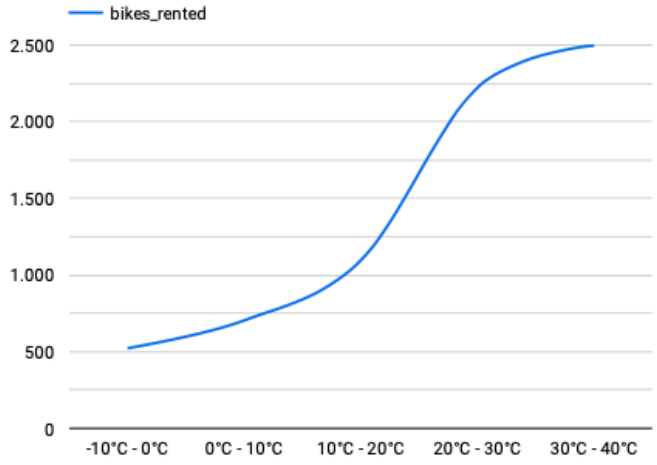
1 - 5 / 5

Temperature

	temp_range	bikes_rented
1.	30°C - 40°C	
2.	20°C - 30°C	
3.	10°C - 20°C	
4.	0°C - 10°C	
5.	-10°C - 0°C	

1 - 5 / 5





Conclusions:

- Wind seems to affect negatively the number of bikes rented, as the majority of bikes are shared with absence of wind or with a low value of it. The higher the wind speed, the less the number of bikes rented on average.
- The temperature influences the bike rentals. More specifically, the higher the temperature, the higher the number of bike rentals on average.

4.3 Future bike shares prediction

To run application for predicting and visualizing the future bike shares, the command line call is the following:

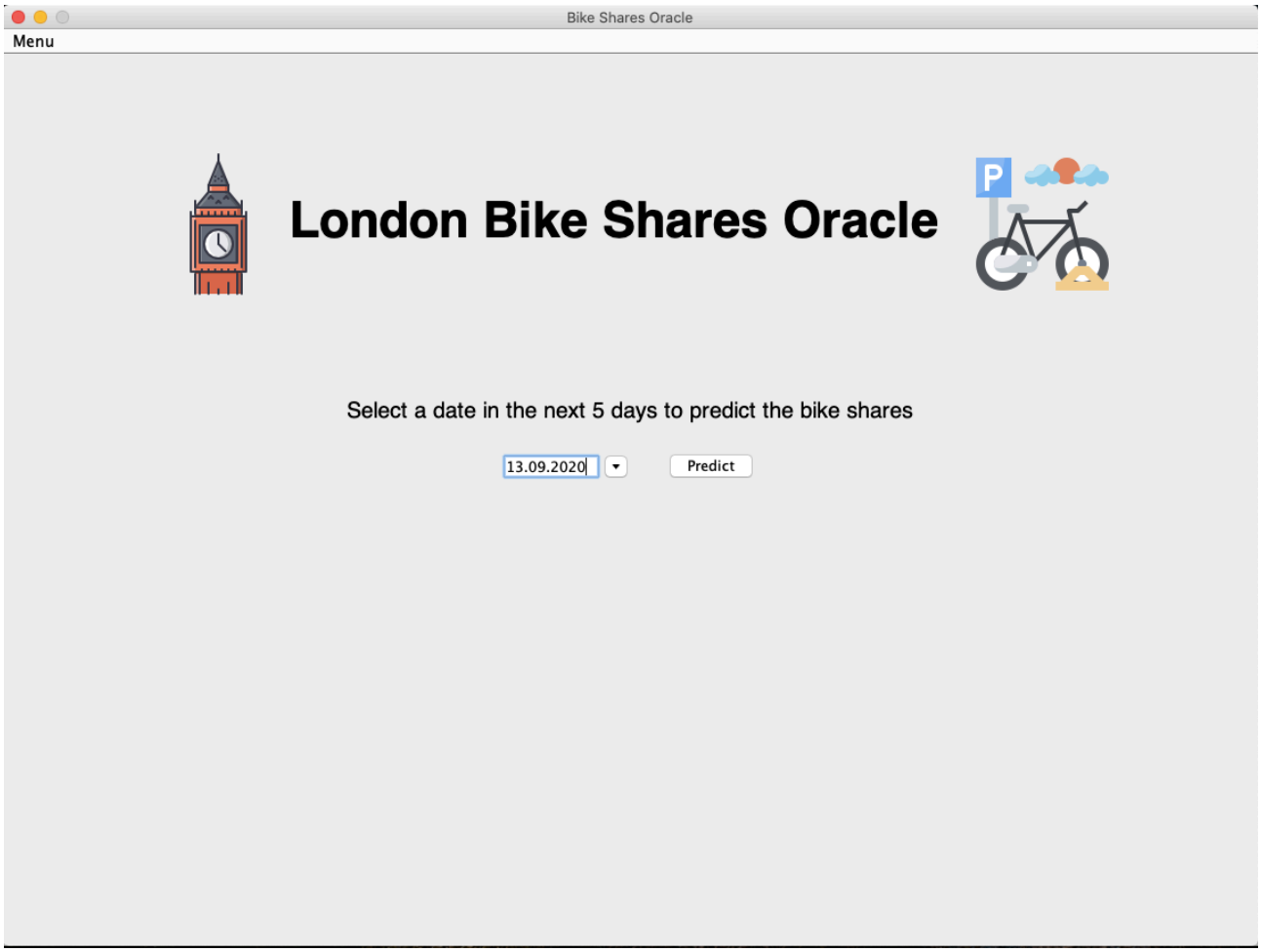
```
spark-submit  --class rentals_weather_gui.AppManager --deploy-mode client --master local target/Project-1.0-SNAPSHOT-jar-with-dependencies.jar <joined_dataset>
```

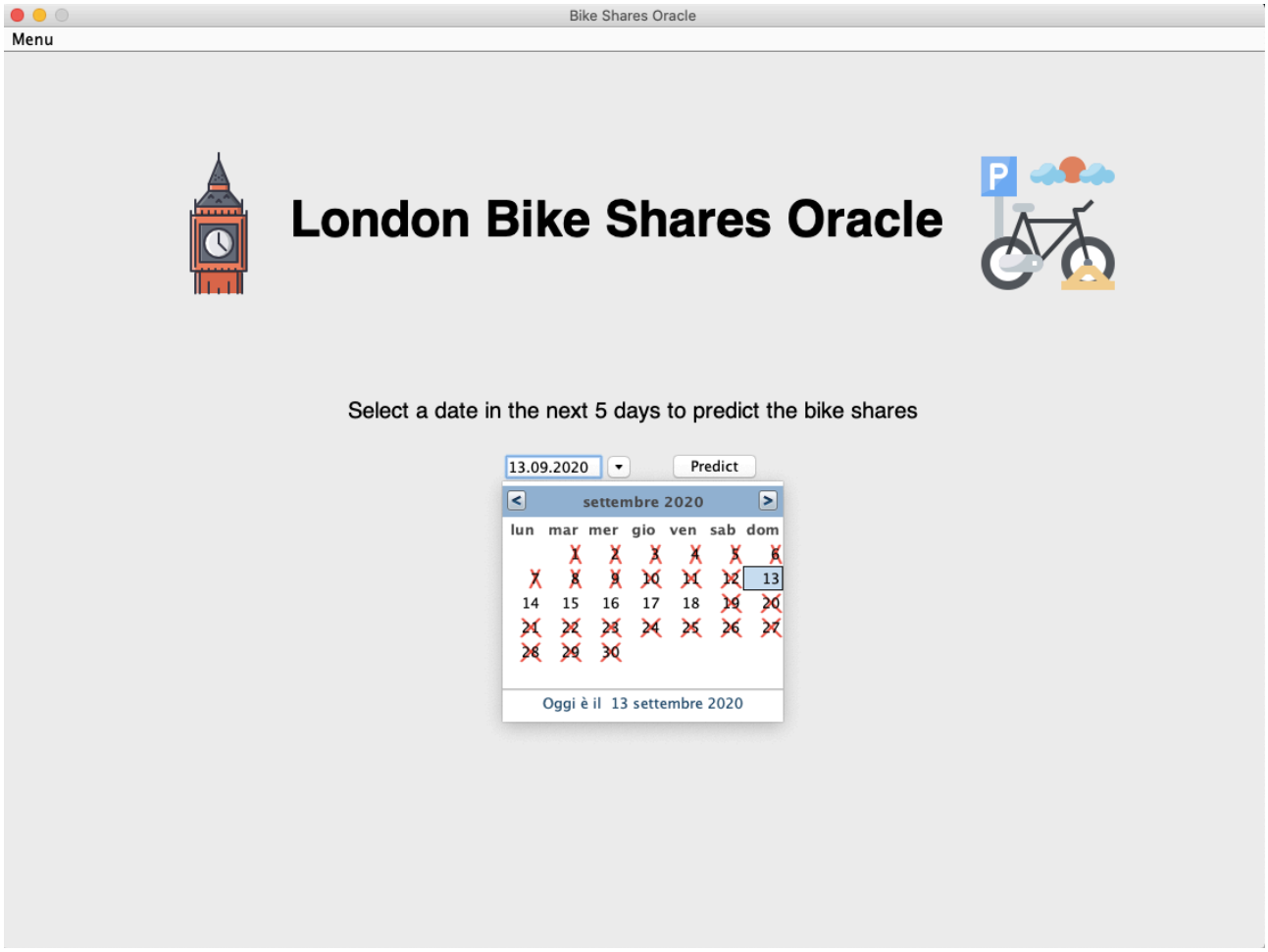
where `<joined_dataset>` represents the dataset output from pre-processing phase. To use the already included dataset, change this part with `input_data/rental_weather.csv`

The above command will perform the training of the Random Forest Regressor and the prediction of the future 5 days bike shares in the back end, and will then launch an application for visualizing the results.

4.3.1 Application GUI and usage

The application starts once the Random Forest Regressor performs training and predictions in the back end, and allows the user to select the day on which he wants to check the weather conditions together with the number of bike shares that are predicted by the system.

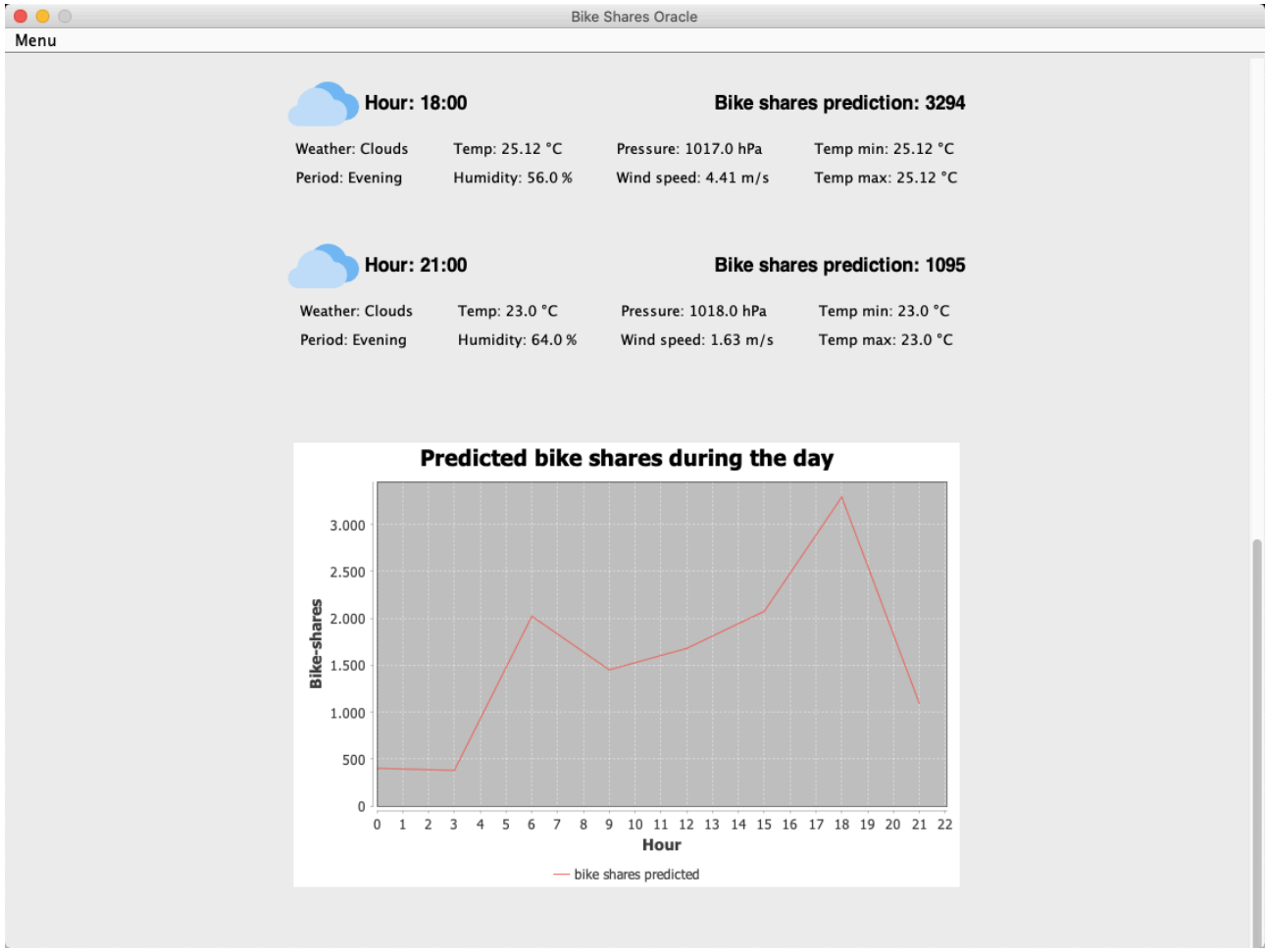




Once the user selects a day, the 3-hour weather situation is presented to him, together with the bike rentals prediction that has been realized by the Random Forest Regressor algorithm of Spark MLib.
In addition, he is shown the graph of the demand for rental bikes expected during the day. In this way. In the following provided screenshots, we can notice how the hourly distribution of bike shares in a workday is predicted, with a peak at 6 and 18.

In this way, the user can identify possible peaks or patterns and make strategic choices that can generally improve the service and make it increasingly used by citizens, going to improve the living conditions of the city and reduce pollution and traffic.


Bike Shares Oracle
Menu
Tuesday 15-9-2020
Hour: 00:00
Bike shares prediction: 401
Weather: Clear
Temp: 20.57 °C
Pressure: 1017.0 hPa
Temp min: 20.57 °C
Period: Night
Humidity: 57.0 %
Wind speed: 3.19 m/s
Temp max: 20.57 °C
Hour: 03:00
Bike shares prediction: 379
Weather: Clear
Temp: 19.81 °C
Pressure: 1016.0 hPa
Temp min: 19.81 °C
Period: Night
Humidity: 62.0 %
Wind speed: 2.67 m/s
Temp max: 19.81 °C
Hour: 06:00
Bike shares prediction: 2022
Weather: Clear
Temp: 19.18 °C
Pressure: 1016.0 hPa
Temp min: 19.18 °C
Period: Morning
Humidity: 62.0 %
Wind speed: 1.84 m/s
Temp max: 19.18 °C
Hour: 09:00
Bike shares prediction: 1451
Weather: Clouds
Temp: 23.84 °C
Pressure: 1017.0 hPa
Temp min: 23.84 °C
Period: Morning
Humidity: 47.0 %
Wind speed: 1.51 m/s
Temp max: 23.84 °C
Hour: 12:00
Bike shares prediction: 1680
Weather: Clouds
Temp: 27.66 °C
Pressure: 1017.0 hPa
Temp min: 27.66 °C


Bike Shares Oracle
Menu
Hour: 18:00
Bike shares prediction: 3294
Weather: Clouds
Temp: 25.12 °C
Pressure: 1017.0 hPa
Temp min: 25.12 °C
Period: Evening
Humidity: 56.0 %
Wind speed: 4.41 m/s
Temp max: 25.12 °C
Hour: 21:00
Bike shares prediction: 1095
Weather: Clouds
Temp: 23.0 °C
Pressure: 1018.0 hPa
Temp min: 23.0 °C
Period: Evening
Humidity: 64.0 %
Wind speed: 1.63 m/s
Temp max: 23.0 °C
Predicted bike shares during the day
Bike-shares
3.000
2.500
2.000
1.500
1.000
500
0
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
Hour
bike shares predicted

5. Conclusions and lesson learned

5.1 Conclusions

The project was born following the request of the London authorities and the system managers to improve the city's bicycle rental service, to make the city less polluted and trafficked.

The project made it possible to combine data about bicycle rental in London scattered in different files into a single dataset, combining it with meteorological data. This combination of data made it possible to analyse the number of bikes rented by comparing it with different factors, obtaining interesting statistics.

Here is a summary of the results obtained from the visualizations of the statistics:

- The majority of bikes are rented during workday, in particular in the afternoon or morning.
- Hourly distribution of bike shares changes from weekends to workdays, indicating that the type of the day highly influences the hourly usage of the service.
- The usage of the service increases over time. Fall is the season with the highest demand of bikes.
- Wind speeds affects negatively the number of bike rentals. Temperature affects positively the number of bike rentals.

The data was used to train a regression algorithm, used to predict future uses of the bike sharing service. This prediction allowed to realize a program that allows to visualize the weather conditions and the number of bikes rented at 3 hour intervals for the next 5 days.

5.2 Lesson learned and future works

The project allowed me to get to know Spark in more detail, especially the Spark SQL and Spark MLib libraries, and understand its effectiveness in analysing and processing large amounts of data. The learning process was not intuitive, and I encountered initial difficulties in becoming familiar with the framework. However, this familiarity was acquired as the project progressed.

In conclusion, I think that the choice to use Spark was the right one, because it allowed to realize the initial intentions and to achieve the objectives in an effective way.

Regarding the future works, I believe that there are ideas where we can work to improve the system and its usefulness. First, we could spend more time in the realization of the machine learning algorithm to improve forecast accuracy. In addition, we could consider several aspects of the initial dataset separately to get more statistics, such as using variables like the bicycle stations used and the duration of a trip.