Bike-Sharing Market Analysis

With Python

Lei Chen

Department of Computer Science
The University of Auckland New Zealand

Background

The bike-sharing system is a new generation of traditional bike rentals with automatically process from membership, rental to return. These systems make it easy for users to rent a bike from a specific location and then return to another location. Currently, there are more than 500 bike-sharing projects around the world, consisting of more than half a million bikes. Today, there is great interest in these systems because of their important role in transport, environment and health issues.

In addition, the data generated by these systems make them attractive for research. Unlike other transportation services, such as buses or subways, the travel, departure, and arrival locations of these systems are clearly documented. This feature makes bike-sharing systems as a virtual sensor networks that can be used to detect mobility in cities.

1. Business Understanding

1.1. Business Objectives

However, in China, the pursuit of rapid expansion in the market competition, resulted in a large number of bicycle disorderly release in most cities, which has been a huge waste of resources and disrupting the urban order. Thus, a prediction to the demand of sharing-bikes is necessary, not only for the companies concerning their profits, but also for the government, which are able to dynamically manage the overall amount of shared bikes in the market.

As we all know, Bike-sharing rental process is highly correlated to the commuting traffic, and weather settings. For instance, weather conditions, day of week, season, hour of the day, etc. can affect the rental behaviors. Meanwhile, count of rented bikes are also correlated to some events in the town.

Thus, this project will focus on below objectives:

- Predication of bike rental count based on various variables.

Tentatively, it will be judged a success with below Success Criteria:

- Reduce resource wasted

1.2. Assessing the Situation

The bike-sharing systems record almost all information of user's rental behavior, and some of them have released their data on public data websites where I can search for the appropriate data that can support my data-mining project.

This project will try to predict bike rental demand in the Capital Bikeshare program in Washington, D.C., with the shared bike data set from https://www.kaggle.com/marklvl/bike-sharing-dataset. In this iteration, I will use Python as my tool, and resources including online learning materials and guide from tutors are available.

There should be no real-world or legal risk as it is just a predicting project with historical public data.

1.3. Data Mining Goals

We can translate the business objectives into data mining terms. The goals for this project can be completed by using historical records about previous rentals to generate a model to predict bike-sharing demand trend, with R2 (coefficient of determination) score greater than 0.9 as success criteria.

1.4. Project Plan

The overview plan for the study is as shown in the table below.

Phase	**Time**	**Resources**	**Comments**
Business Understanding	1 week	Google	Case study for Housing market
Data Understanding	1 week	Kaggle, Python	Project Proposal (Iteration 1)
Data Preparation	0.5 week	Python	Learn data visualization with python
Data Transformation	0.5 week	Python	Learn data manipulation
Data Mining	0.5 week	Python	Learn data mining algorithms
Interpretation	0.5 week	Python	Pattern study and visualization

2. Data understanding

2.1. Data Collection

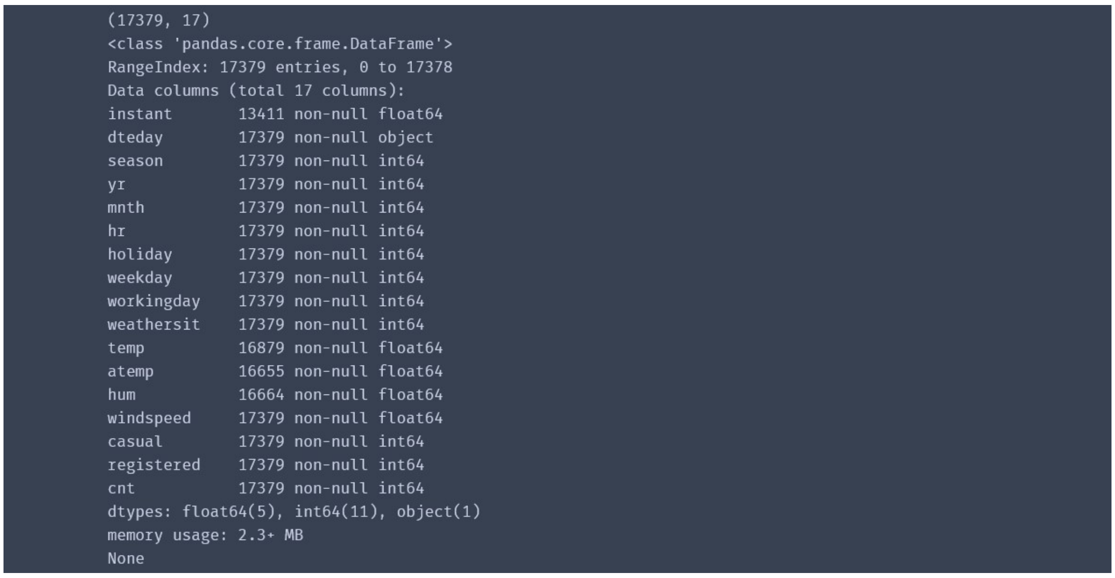
This project project uses bike-sharing data set downloaded from https://www.kaggle.com/marklvl/bike-sharing-dataset, with the original source available in http://capitalbikeshare.com/system-data and corresponding weather and holiday schedule information, which are extracted from http://www.freemeteo.com and http://dchr.dc.gov/page/holiday-schedule respectively.

2.2. Data Description

These data contain the two-year historical log, corresponding to years 2011 and 2012, from Capital Bikeshare system, Washington D.C., USA, stored in a CSV files.

Firstly, I import the data in Python and check its overall information, see Figure 1. There are 17379 rental records in total, with 17 columns of fields, including season, month, hours in a day, working day or not, weather situations and so on. Most fields are recorded well without NULL value, except fields 'instant' (13411 non-null values), 'temp', 'atemp' and 'hum'.

```
import pandas as pd
data = pd.read_csv('dataset.csv')
print(data.shape)
print(data.info())
```



```
(17379, 17)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17379 entries, 0 to 17378
Data columns (total 17 columns):
instant       13411 non-null float64
dteday        17379 non-null object
season        17379 non-null int64
yr            17379 non-null int64
mnth          17379 non-null int64
hr            17379 non-null int64
holiday       17379 non-null int64
weekday       17379 non-null int64
workingday    17379 non-null int64
weathersit    17379 non-null int64
temp          16879 non-null float64
atemp         16655 non-null float64
hum           16664 non-null float64
windspeed     17379 non-null float64
casual        17379 non-null int64
registered    17379 non-null int64
cnt           17379 non-null int64
dtypes: float64(5), int64(11), object(1)
memory usage: 2.3+ MB
None
```

Figure 1 Data Information

Table 1 lists the detailed attributes of the data and explanation of them according to data source website.

Field	Explaination
instant	record index
dteday	date (yyyy-mm-dd)
season	season (1:springer, 2:summer, 3:fall, 4:winter)
yr	year (0: 2011, 1:2012)
mnth	month (1 to 12)
hr	hour (0 to 23)
holiday	holiday is 1, otherwise is 0

weekday	day of the week (0 to 6)
workingday	working day is 1, otherwise is 0
weathersit	weather conditions (1 to 4)
temp	Normalized temperature in Celsius.
atemp	Normalized feeling temperature in Celsius.
hum	Normalized humidity.
windspeed	Normalized wind speed.
casual	count of casual users
registered	count of registered users
cnt	count of total rental bikes

Table 1 Fields Detail

2.3. Data Exploration

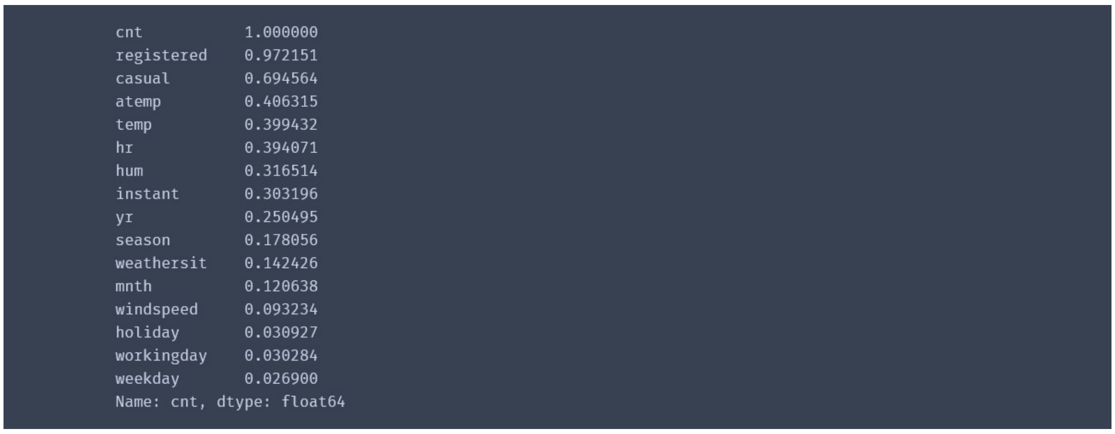
To predict the rental demand, we can start with checking the correlation of ‘cnt’ (overall count of rental bikes) against other fields.

As seen from Figure 2, the weather (including temperature, humidity) has a significant effect on the number of rentals.

In addition, time factors such as year, month, and season also have a significant effect on count, while the correlation coefficients of holiday and weekday and count are minimal.

Fields ‘registered’ and ‘casual’ are just components of the overall rental count, so we can ignore them although they are highly correlated with the mining goal.

```
corr = data.corr()
influnce = corr['cnt'].sort_values(ascending=False)
print(influnce)
```



```
cnt           1.000000
registered    0.972151
casual        0.694564
atemp         0.406315
temp          0.399432
hr            0.394071
hum           0.316514
instant       0.303196
yr            0.250495
season        0.178056
weathersit    0.142426
mnth          0.120638
windspeed     0.093234
holiday       0.030927
workingday    0.030284
weekday       0.026900
Name: cnt, dtype: float64
```

Figure 2 Correlation Coefficients

To confirm above assumptions and further understand the data, I built some plots to demonstrate the relationships of some relevant fields and the prediction goal – total rental counts.

```
sn.barplot(data['weathersit'], data['cnt'])
plt.title('the influnce of weather')
plt.savefig("the influnce of weather.jpg")
```

```
sn.boxplot(data['yr'], data['cnt'])
plt.title('the influnce of year')
plt.savefig("the influnce of year.jpg")

sn.pointplot(data['mnth'], data['cnt'])
plt.title('the influnce of month')
plt.savefig("the influnce of month.jpg")

sn.boxplot(data['season'], data['cnt'])
plt.title('the influnce of season')
plt.savefig("the influnce of season.jpg")

sn.barplot(data['hr'], data['cnt'])
plt.title('the influnce of hours in a day')
plt.savefig("the influnce of hours in a day.jpg")
```

Figure 3 shows that the effect of weather conditions on rental behavior is obvious.

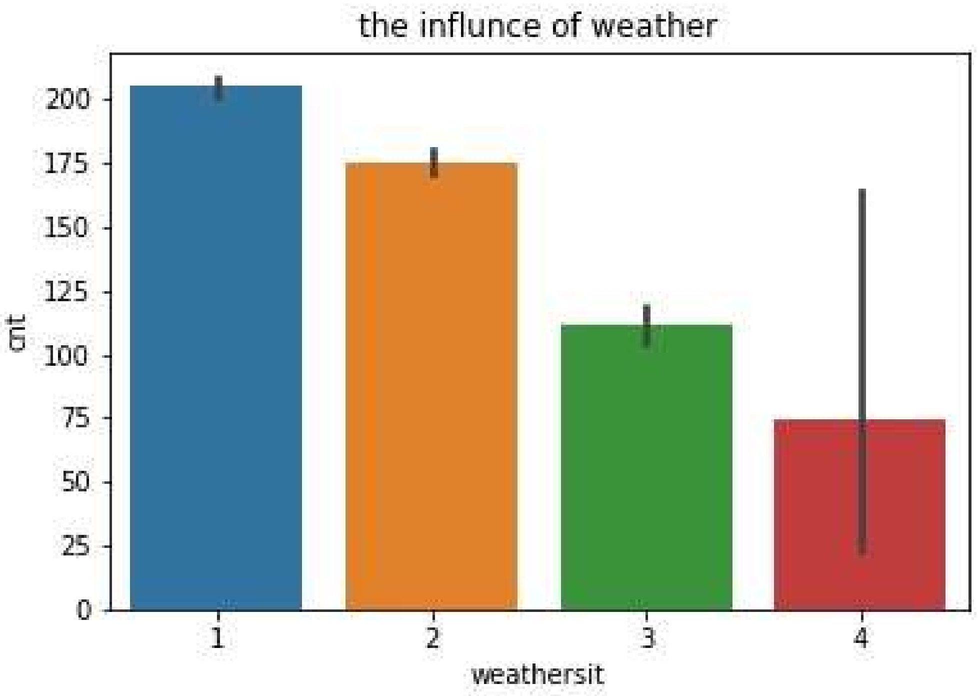


Figure 3

Figure 4 demonstrates that the number of rentals in 2012 is significantly higher than in 2011, showing that shared bikes have become more and more popular.

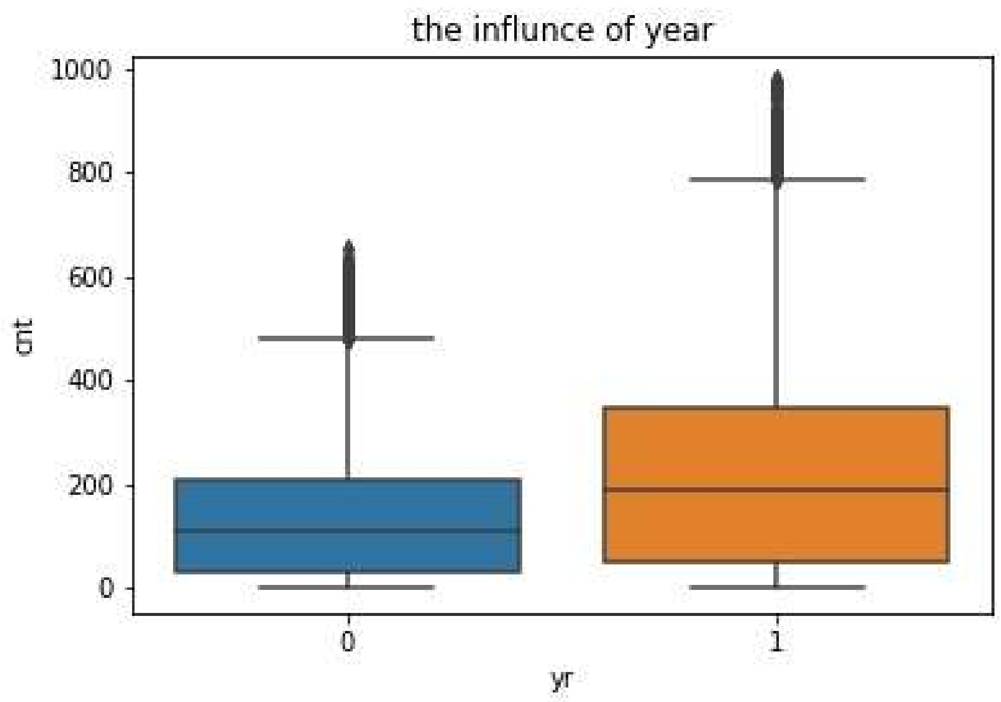


Figure 4

Meanwhile, the impact of months on rental counts is clearly the same in 2011 and 2012, with a rapid increase in monthly rentals from January to peak in June, but a sharp decline after October, which is a clearly seasonal trend, see Figure 5.

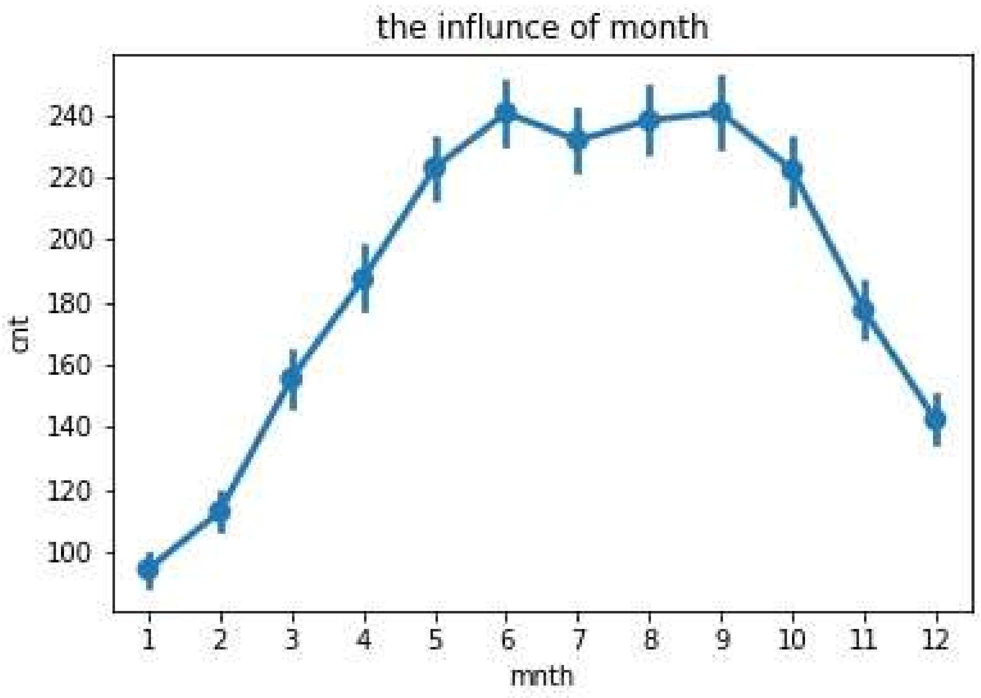


Figure 5

Figure 6 shows the impact of the season on the number of rentals, which indicates the similar effect of the month and the season. As the month is more detailed, the season field can be ignored in following process.

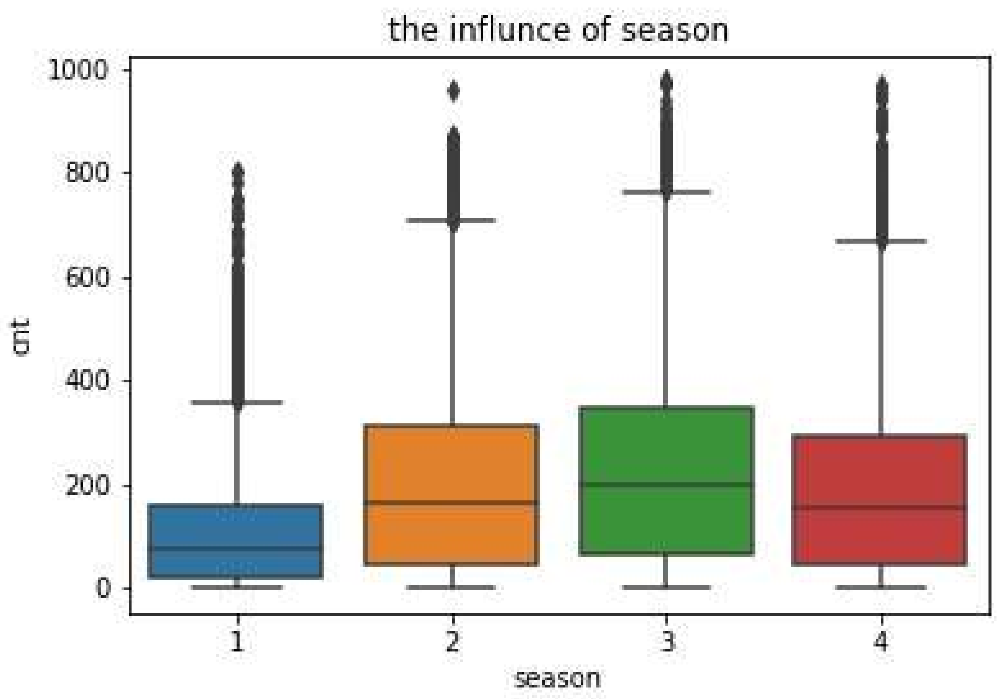


Figure 6

In terms of hours within a day, Figure 7 shows two peaks every day, around 8 a.m. and 17 p.m., just in time for the rush hour of working days.

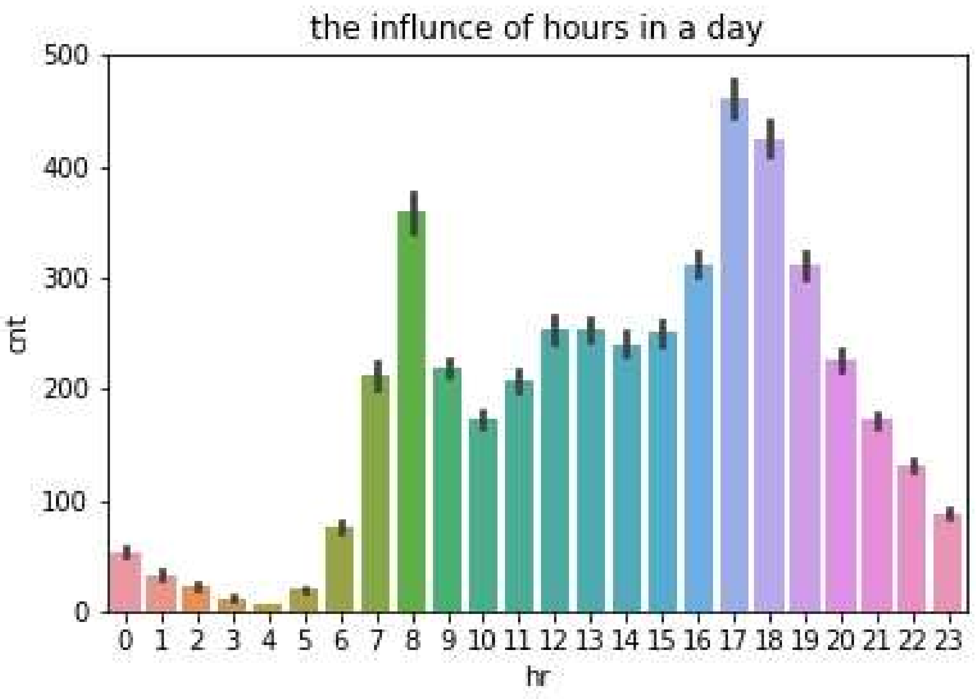


Figure 7

2.4. Data Quality Audit

As seen from Figure 1, the overall data quality is pretty good, except 3 fields with Null values (3968 missing values for ‘instant’, and 500, 724, 715 for temp, atemp, hum respectively).

So according to data exploration and data audit result, the following steps will:

- Remove fields with more than 20% missing values
- Impute missing values for other fields
- Ignore ‘casual’, ‘registered’ fields as they do not make sense for the mining goal.
- Ignore ‘season’ filed as it plays the same role as ‘month’.

3. Data Preparation

According to the above exploration of the data, I will start to conduct the work of data preparation in this chapter, including data select, data clean, data construct and data format, to deal with the data quality issues mentioned in chapter 2.4.

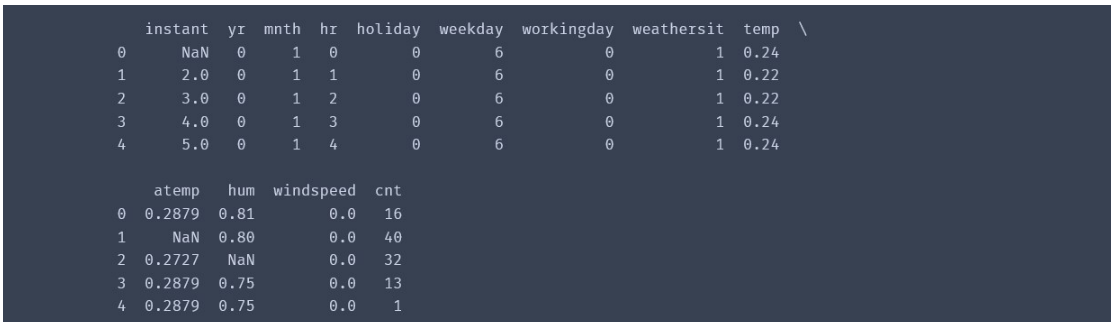
3.1. Select the data

In general, there are two ways to select data: selecting rows and selecting fields.

For the fields selection, as analyzed in the previous section, 'dteday', 'registered', 'causal' do not make sense for the data mining goals, because this project focuses only on the prediction of the total rental numbers. So we can just disposal these 3 fields in the following data manipulation, When exploring the data, we have found that the impact of the season on the total rentals, is pretty similar to that of the month. As the month is more detailed, we can also ignore the 'season' field as discussed before.

Figure 8 shows the selected data frame after the fields selecting.

```
data = data.drop(['dteday', 'registered', 'casual', 'season'], axis=1)
print(data.head())
```



```
   instant  yr  mnth  hr  holiday  weekday  workingday  weathersit  temp  \
0      NaN   0     1   0        0        6           0           1  0.24
1      2.0   0     1   1        0        6           0           1  0.22
2      3.0   0     1   2        0        6           0           1  0.22
3      4.0   0     1   3        0        6           0           1  0.24
4      5.0   0     1   4        0        6           0           1  0.24

    atemp   hum  windspeed  cnt
0  0.2879  0.81        0.0   16
1     NaN  0.80        0.0   40
2  0.2727   NaN        0.0   32
3  0.2879  0.75        0.0   13
4  0.2879  0.75        0.0    1
```

Figure 8 Data Selected

3.2. Clean the data

As the result of Data Audit, there is some Null data in 'instant', 'temp', 'atemp' and 'hum' fields. In the process of data cleaning, I need to remove 'instant' fields as it has more than 20% values missing.

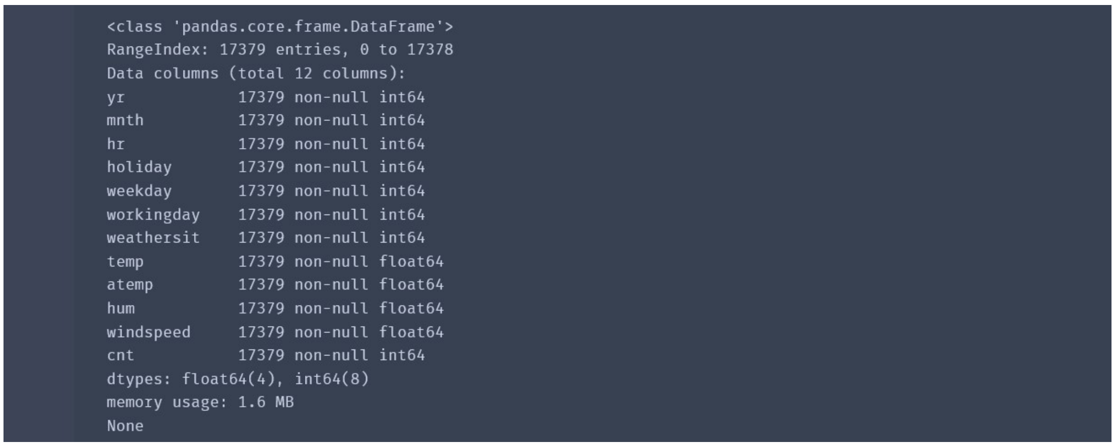
```
data = data.drop(['instant'], axis=1)
```

3.3. Construct the data

Figure 1 shows only 500, 724, 715 missing values for temp, atemp, hum fields, and considering there maybe extreme weather conditions, it is better to impute them with mid-range value of these fields.

```
data['temp'] = data['temp'].fillna(data['temp'].median())
data['atemp'] = data['atemp'].fillna(data['atemp'].median())
data['hum'] = data['hum'].fillna(data['hum'].median())
print(data.info())
```

Figure 9 shows the data information after cleaning and constructing.



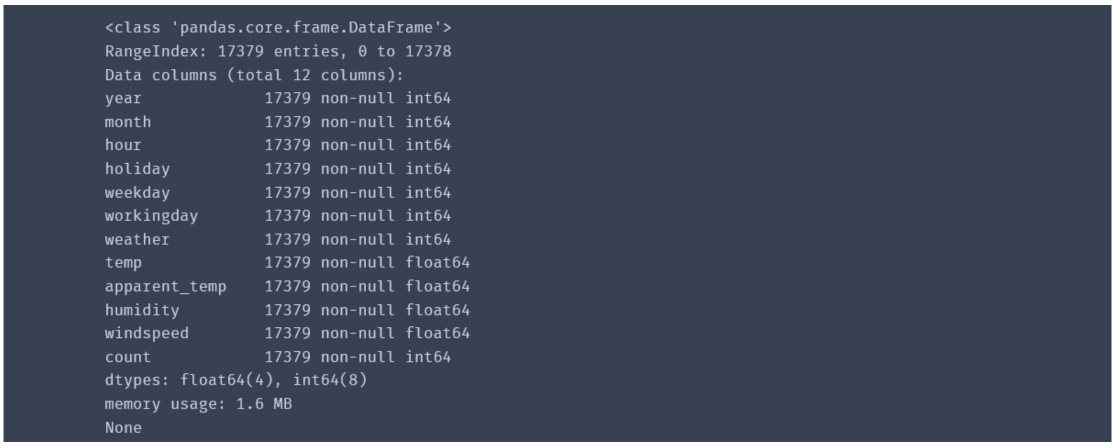
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17379 entries, 0 to 17378
Data columns (total 12 columns):
yr            17379 non-null int64
mnth          17379 non-null int64
hr            17379 non-null int64
holiday       17379 non-null int64
weekday       17379 non-null int64
workingday    17379 non-null int64
weathersit    17379 non-null int64
temp          17379 non-null float64
atemp         17379 non-null float64
hum           17379 non-null float64
windspeed     17379 non-null float64
cnt           17379 non-null int64
dtypes: float64(4), int64(8)
memory usage: 1.6 MB
None
```

Figure 9 Data Cleaned and Constructed

3.4. Format the data

During visualization built previously, index of some fields are not so clear, so in this step I will normalize the naming of fields for better interpretation in the following steps.

```
data.columns = ['year', 'month', 'hour', 'holiday', 'weekday', 'workingday',
                        'weather', 'temp', 'apparent_temp', 'humidity', 'windspeed', 'count']
print(data.info())
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17379 entries, 0 to 17378
Data columns (total 12 columns):
year             17379 non-null int64
month            17379 non-null int64
hour             17379 non-null int64
holiday          17379 non-null int64
weekday          17379 non-null int64
workingday       17379 non-null int64
weather          17379 non-null int64
temp             17379 non-null float64
apparent_temp    17379 non-null float64
humidity         17379 non-null float64
windspeed        17379 non-null float64
count            17379 non-null int64
dtypes: float64(4), int64(8)
memory usage: 1.6 MB
None
```

Figure 10 Data Formatted

4. Data Transformation

4.1. Reduce the data

Considering fields ‘holiday’, ‘weekday’ and ‘workingday’ seem have low correlations with overall rental counts in Figure 2, maybe we should check the impact of working before further processing.

When further exploring the data, I find that the rental counts are almost the same each day in a week, see Figure 11. So it is reasonable to reduce this field.

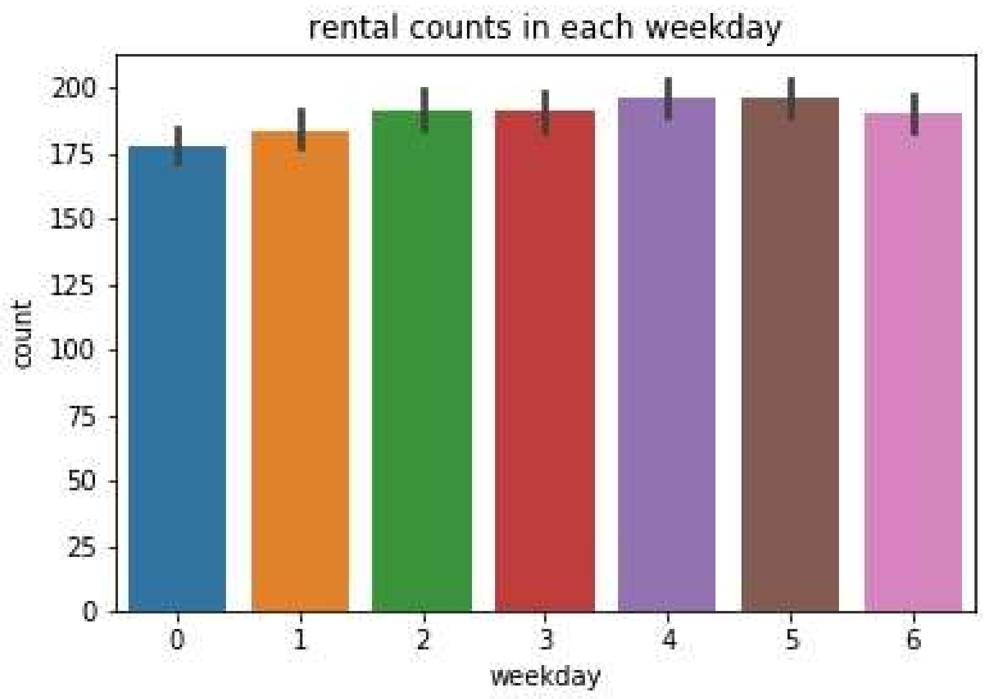
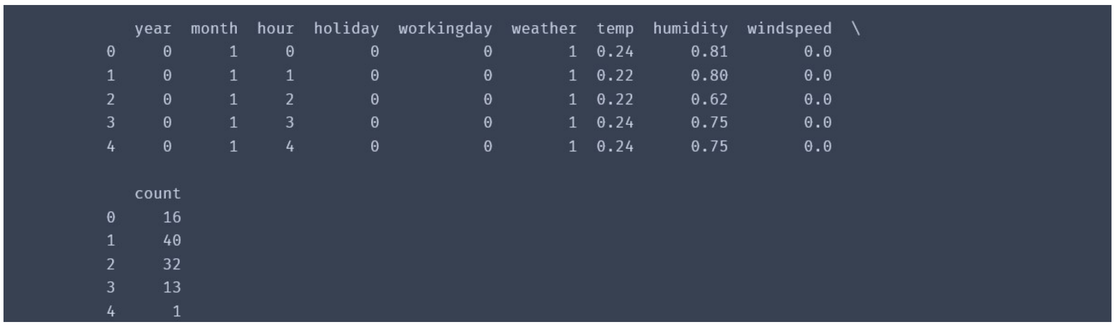


Figure 10 Rental counts in each weekday

Also I noticed that ‘temp’, ‘apparent_temp’ and ‘humidity’ are all weather attributes, it is reasonable to compare the relationship between any two of them, and between the total counts. It is clear in the Figure 11 that ‘temp’ and ‘apparent_temp’, are roughly linear. As showed in Figure 2, they also have similar correlation with rental counts (0.406 and 0.399, respectively), so we can further reduce the data by removing ‘apparent_temp’ field.

```
data = data.drop(['weekday', 'apparent_temp'], axis=1)
print(data.head())
```



```
   year  month  hour  holiday  workingday  weather  temp  humidity  windspeed  \
0     0      1     0        0           0        1  0.24      0.81        0.0
1     0      1     1        0           0        1  0.22      0.80        0.0
2     0      1     2        0           0        1  0.22      0.62        0.0
3     0      1     3        0           0        1  0.24      0.75        0.0
4     0      1     4        0           0        1  0.24      0.75        0.0

   count
0     16
1     40
2     32
3     13
4      1
```

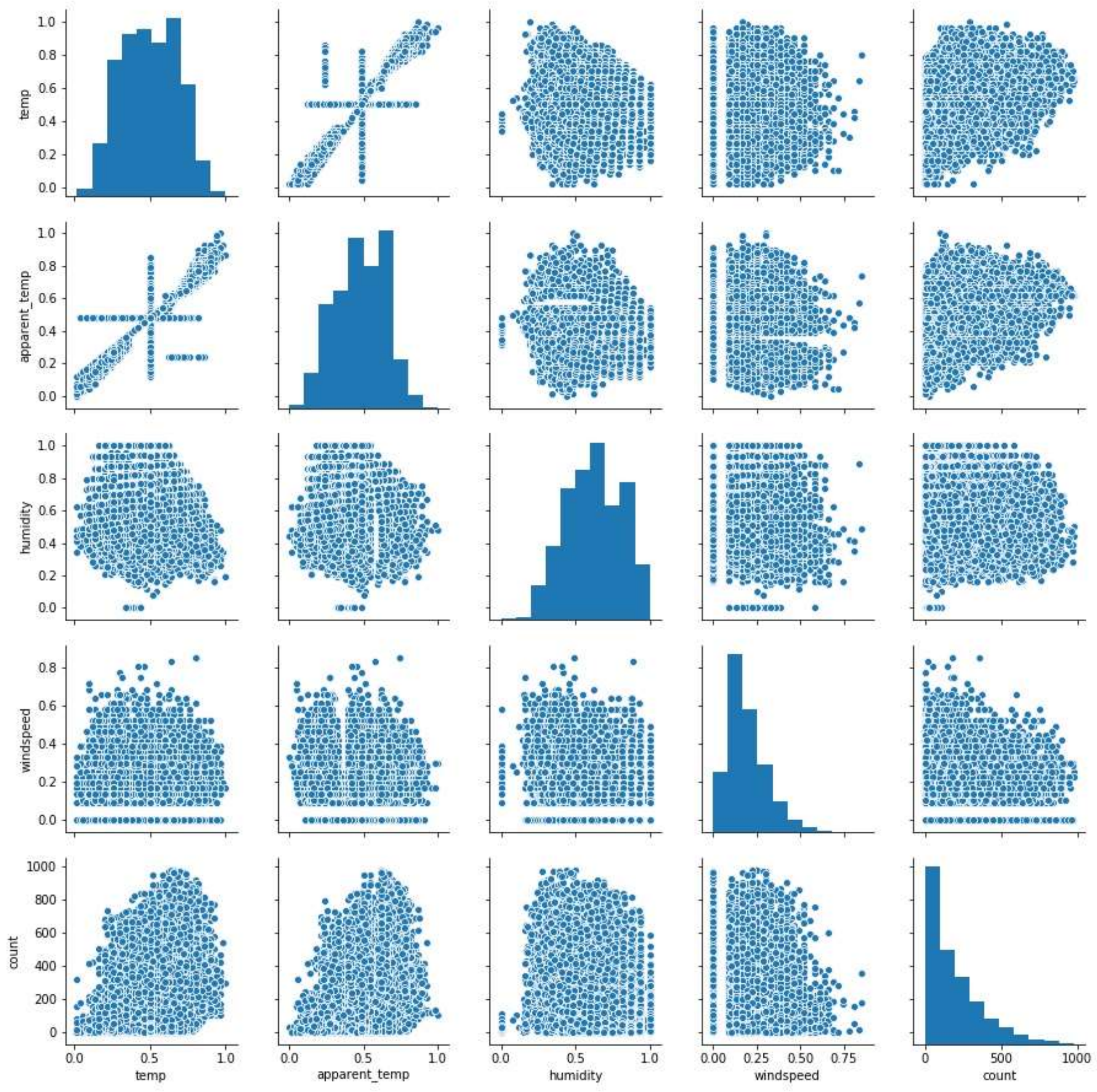


Figure 11 the Influence of weather attributes

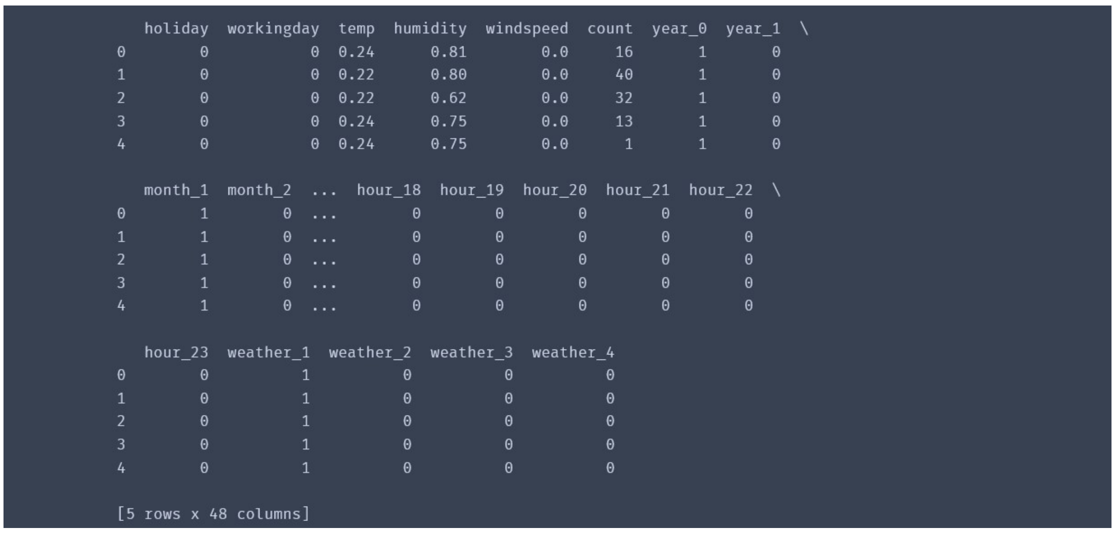
4.2. Project the data

In summary, the project will extract 9 fields including year, month, hour, workingday, holiday, weather, temp, humidity, and windspeed for further data mining process, where year, month, hour, workingday, holiday, and weather are discrete variables. Since workingday and holiday are already binary values, the remaining discrete variables require transformation in a one-hot method, see Figure 12.

```
col_trans = ['year', 'month', 'hour', 'weather']
data_trans = pd.get_dummies(data, columns=col_trans)
print(data_trans.head())
```

Then I split the data into observations (the relevant fields that may affect the rental counts) and predictors (overall rental counts) for the further data mining use.

```
y = data_trans['count']
x = data_trans.drop('count', axis=1)
```



```
   holiday  workingday  temp  humidity  windspeed  count  year_0  year_1  \
0        0           0  0.24      0.81        0.0     16       1       0
1        0           0  0.22      0.80        0.0     40       1       0
2        0           0  0.22      0.62        0.0     32       1       0
3        0           0  0.24      0.75        0.0     13       1       0
4        0           0  0.24      0.75        0.0      1       1       0

   month_1  month_2  ...  hour_18  hour_19  hour_20  hour_21  hour_22  \
0        1        0  ...        0        0        0        0        0
1        1        0  ...        0        0        0        0        0
2        1        0  ...        0        0        0        0        0
3        1        0  ...        0        0        0        0        0
4        1        0  ...        0        0        0        0        0

   hour_23  weather_1  weather_2  weather_3  weather_4
0        0          1          0          0          0
1        0          1          0          0          0
2        0          1          0          0          0
3        0          1          0          0          0
4        0          1          0          0          0

[5 rows x 48 columns]
```

Figure 12 Project the data

5. Data Mining Method Select

Generally, there are three main types of modelling methods: classification, association and segmentation. Each method has certain advantages and suits for different types of problems.

5.1. Match the goals to methods of data mining

Classification models utilize existing historical data to predict future possible outcomes.
Association models are used to find out possible relationships or patterns between different entities (attributes).
Segmentation models or clustering models are often used to group data instead of predicting a specific result, the data in each group has a similar interesting or pattern.
The data mining goal of this project is to predict the future possible demand of bike-rental, which means it needs existing historical data to predict the overall rental counts, so classification method would be the choice.

5.2. Select the appropriate data-mining methods

According to project goals, the best method for this data mining is classification. Representative methods for classification are random trees, regression, and neural networks. As the overall rental counts are continuous values to predict, I should choose the appropriate regression data-mining algorithms in the following data mining steps.

6. Data Mining Algorithm Select

6.1. Exploratory of algorithms

The data-mining goals of this project are predicting the possible bike-rental demands. As discussed in Chapter 5, there are a variety of regression algorithms in classification for different data mining objectives. In the following steps, I will test some popular algorithms and choose the most appropriate one.

6.2. Select algorithm

To find the best algorithm for the data-mining goal, I choose 3 popular regression algorithms: Random Forest Regression, XGBoost Tree, and Neural Net Regression, then use R2 (coefficient of determination) to access their performances.

6.3. Build Models

I build 3 different models with data extracted from Chapter 4.2, using default paraments for random forest regression model and XGBoost tree model, and 3 layers of neural network with maximum 500 times iteration for neural network regression. And evaluate them to check if they can achieve success criteria.

Random Forest Regression Modelling:

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
rfr = RandomForestRegressor()
rfr.fit(x, y)
rfr.pred = rfr.predict(x)
print('R2 Score of Random Forest Regression: ', r2_score(y, rfr.pred))
```

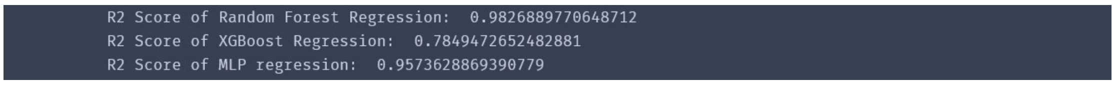
XGBoost Tree Modelling:

```
from xgboost import XGBRegressor
xgbr = XGBRegressor()
xgbr.fit(x, y)
xgbr.pred = xgbr.predict(x)
print('R2 Score of XGBoost Regression: ', r2_score(y, xgbr.pred))
```

Neural Network Regression Model Fit:

```
from sklearn.neural_network import MLPRegressor
mlp = MLPRegressor(hidden_layer_sizes=(50,50,50), max_iter=500)
mlp.fit(x,y)
mlp.pred = mlp.predict(x)
print('R2 Score of MLP regression: ', r2_score(y, mlp.pred))
```

Below is the R2 scores of 3 models.



```
R2 Score of Random Forest Regression:  0.9826889770648712
R2 Score of XGBoost Regression:  0.7849472652482881
R2 Score of MLP regression:  0.9573628869390779
```

As the models built in last section, both Random Forest and Neural Network Model meet the success criteria with high correlation of rental counts (R2 score is 0.983 and 0.957 respectively), we will select them for further data mining.

7. Data Mining

7.1. Create and justify test designs

To justify the model with the success criteria, in this step, I partition the data set into two parts: 80% as training set and 20% as test set.

```
from sklearn.model_selection import train_test_split
xTrain, xTest, yTrain, yTest = train_test_split(x, y, test_size=0.2, random_state=10)
```

7.2. Conduct data mining

In this section, I fit a random forest regression model and a neural network regression model with training set and output the predictions based on test set.

Random Forest Regression Modelling:

```
rfr.fit(xTrain, yTrain)
rfr.pred = rfr.predict(xTest)
```

Neural Network Regression Modelling:

```
mlp.fit(xTrain,yTrain)
mlp.pred = mlp.predict(xTest)
```

7.3. Search for patterns

In Chapter 2, we have checked the rental counts based on months, hours in a day, and weather situations. These fields are all considered as important as their high relevance with rental counts. Although Figure 7 shows a clear pattern that there are two peak hours each day, the pattern of day-to-day changes between peaks is not that obvious, which may need take the impact of holidays or working days into consideration, which we did not check before.

Below codes build a plot to demonstrate the difference between working days and non-working days and we can see that ‘workingday’, ‘holiday’ and ‘weekday’ have the same pattern against rental counts in Figure 13.

```
axes = plt.subplots(2, 1, figsize = (16,10))
ax1 = plt.subplot(2, 1, 1)
sn.pointplot(data['hour'], data['count'], hue=data['workingday'], ax=ax1)
ax1.set_title('the influnce of hour in workingday')
ax2 = plt.subplot(2, 2, 3)
sn.pointplot(data['hour'], data['count'], hue=data['holiday'], ax=ax2)
ax2.set_title('the influnce of hour in holiday')
ax3 = plt.subplot(2, 2, 4)
sn.pointplot(data['hour'], data['count'], hue=data['weekday'], ax=ax3)
ax3.set_title('the influnce of hour in weekday')
plt.savefig("the influnce of working or not.jpg")
```

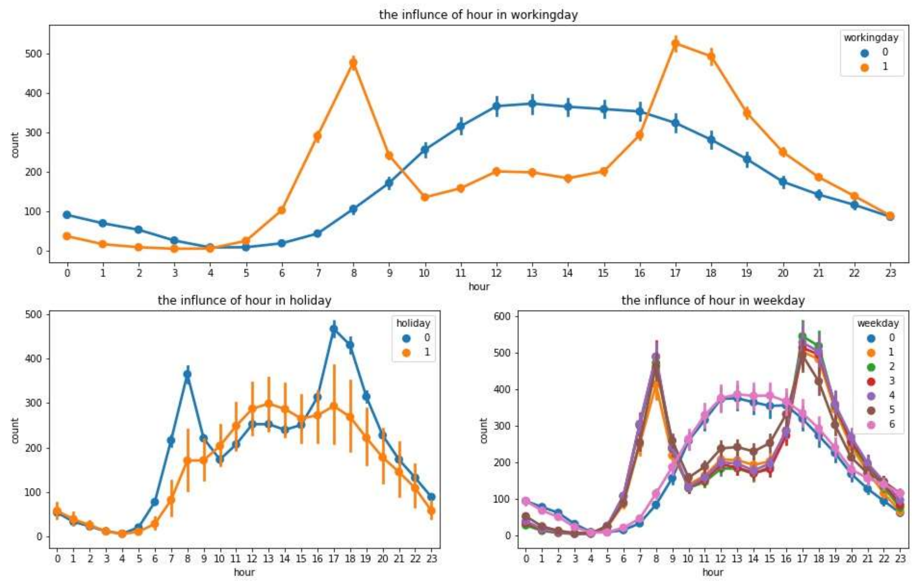


Figure 13 Working or Not

8. Interpretation

8.1. Study the mined patterns

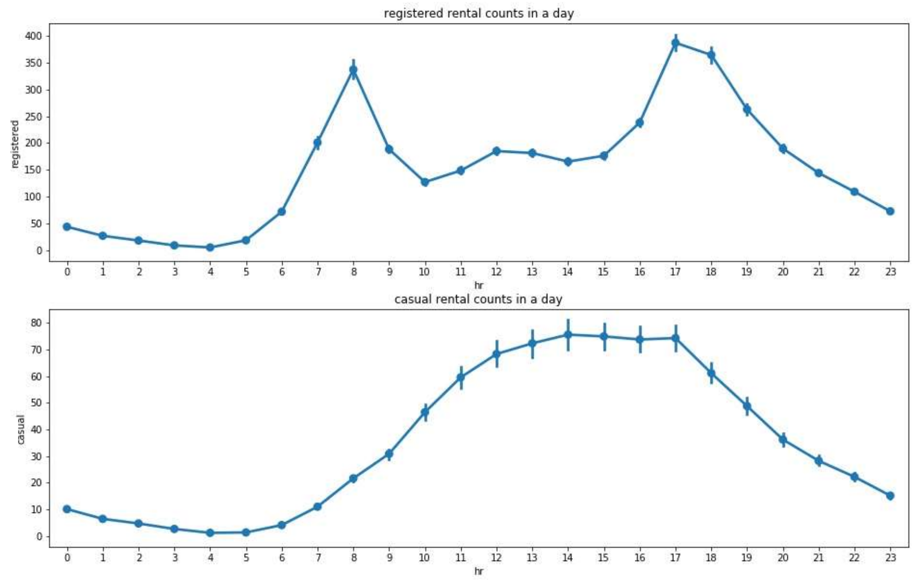
Figure 13 in Chapter 7.3 shows that working days have high rush-hour rental counts, and low counts in the rest of a day, while the rental counts are high in noon and afternoon of non-working days, which meets our assumption of people's rental behavior.

8.2. Visualize the data, results, models and patterns

In previous steps we have built some significant plots to visualize and interpret the relationships between different fields in the data.

Actually, if we further check the casual and registered rental counts respectively, the familiar pattern happens, see Figure 14. It may indicate that registered users are mainly made up of the commuters, whose behavior is more routine and regular. The unregistered users' behavior is more casual, but still has clear pattern to be identified and predicted.

This model is generated based on total rental counts prediction where registered users take a large proportion than casual scenarios. However, the pattern found about the rental behavior in non-working days, or that of unregistered users may also be useful, as it provides a clue to detect some abnormal fluctuation of rental counts, like during some specific holidays, celebrations or important events. Such prediction is necessary and useful to improve the flexibility of the bike-sharing market management.



Pic 15 Registered or Casual

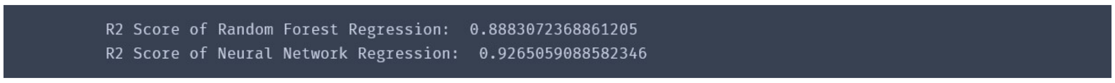
8.3. Interpret the data, results, models and patterns

Once we identify the behavior pattern of rental users, we can release the sharing-bikes dynamically to reach a better input-output ratio. What's more, government then is able to manage and the bike-sharing market more efficiently, regulate the market operation mode, and improve the market mechanism.

8.4. Assess and evaluate results, models and patterns

With the fitted model based on training set, we can check the coefficient of determination of predicted total rental counts with test set.

The evaluated coefficient of determination score for both of them as below. Both models' R2 score drop, especially the random forest regression model, which is 0.888, less than 0.9 and cannot meet the success criteria anymore. So finally, we will use Neural Network Regression model (scored 0.927) for this project.



```
R2 Score of Random Forest Regression:  0.8883072368861205
R2 Score of Neural Network Regression:  0.9265059088582346
```

8.5. Iterate prior steps

During the whole steps of this project, I made several iterations to make sure the data mining process is more effective and has better performance. They are documented as below:

Iteration 1:

During data reduction, I went back to data understanding stage to re-explored 'weekday' and 'atemp' fields, and dropped these 2 fields accordingly.

Iteration 2:

When building models, I tried several values for hiden_layer sizes and max_tier paraments of Neural Network Regression model to find a better performance.

```
# modelling with default parament values
mlp = MLPRegressor()
mlp.fit(x,y)
mlp.pred = mlp.predict(x)
```

The R2 score is 0.909, lower than the model in Chapter 6.3, as default maximum iterations (200) reached but the optimization hasn't converged yet.

Iteration 3:

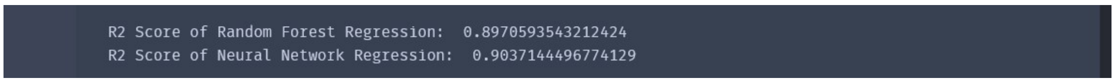
When creating test designs, I changed training set to 70% and test set to 30% , and the evaluation shows that the score of Random Trees had a little rise to 0.897, but that of Neural Network Regression model dropped to 0.904.

```
xTrain, xTest, yTrain, yTest = train_test_split(x, y, test_size=0.3, random_state=10)

rfr.fit(xTrain, yTrain)
rfr.pred = rfr.predict(xTest)

mlp.fit(xTrain,yTrain)
mlp.pred = mlp.predict(xTest)

print('R2 Score of Random Forest Regression: ', r2_score(yTest, rfr.pred))
print('R2 Score of Neural Network Regression: ', r2_score(yTest, mlp.pred))
```



```
R2 Score of Random Forest Regression:  0.8970593543212424
R2 Score of Neural Network Regression:  0.9037144496774129
```

Iteration 4:

When searching and visualizing patterns, I also re-explored the data to find impacting patterns of working or holidays, registered rental or casual rental.

Disclaimer: