1. Preliminary understanding of data

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sn

Biks = pd.read_csv('datasets\\MontrealBikeLane.csv',index_col='Date',parse_dates=True)
weather = pd.read_csv('datasets\\WeatherInfo.csv',index_col='Date/Time',parse_dates=True)
print(Biks)
print(weather)
print(Biks.info())
print(weather.info())
```

We can see that the data for the bikes is 319 lines of data, 22 lanes. Weather includes 27 features such as time, temperature, somatosensory temperature, humidity and wind speed, as shown in Fig. 1. As you can see from Fig. 2, there are no missing values in these data.

```
             Time  Berri1  Boyer  ...  Totem_Laurier  University  Viger
Date                              ...
2015-01-01  00:00      58     12  ...             78          21      6
2015-02-01  00:00      75      7  ...             57          77      4
2015-03-01  00:00      79      7  ...            174          40      5
2015-04-01  00:00      10      1  ...             20           6      0
2015-05-01  00:00      42      0  ...             41          56     10
...           ...     ...    ...  ...            ...         ...    ...
2015-11-11  00:00    3044   1931  ...           1527        2860    356
2015-12-11  00:00    1751    930  ...            955        1777    198
2015-11-13  00:00    1818    906  ...           1040        1727    258
2015-11-14  00:00     979    759  ...            805         737     73
2015-11-15  00:00     913    749  ...            804         685     63

[319 rows x 22 columns]
            Year  Month  ...  Spd of Max Gust Flag Unnamed: 27
Date/Time                ...
2015-01-01  2015      1  ...                   NaN         NaN
2015-01-02  2015      1  ...                   NaN         NaN
2015-01-03  2015      1  ...                   NaN         NaN
2015-01-04  2015      1  ...                   NaN         NaN
2015-01-05  2015      1  ...                   NaN         NaN
...          ...    ...  ...                   ...         ...
2015-12-27  2015     12  ...                   NaN         NaN
2015-12-28  2015     12  ...                   NaN         NaN
2015-12-29  2015     12  ...                   NaN         NaN
2015-12-30  2015     12  ...                   NaN         NaN
2015-12-31  2015     12  ...                   NaN         NaN

[365 rows x 27 columns]
```

Fig. 1 data information

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 319 entries, 2015-01-01 to 2015-11-15
Data columns (total 22 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Time                          319 non-null    object
 1   Berri1                        319 non-null    int64
 2   Boyer                         319 non-null    int64
 3   Brébeuf                       319 non-null    int64
 4   CSC (Côte Sainte-Catherine)   319 non-null    int64
 5   Maisonneuve_1                 62 non-null     float64
 6   Maisonneuve_2                 319 non-null    int64
 7   Maisonneuve_3                 319 non-null    int64
 8   Notre-Dame                    319 non-null    int64
 9   Parc                          319 non-null    int64
 10  Parc U-Zelt Test              52 non-null     float64
 11  PierDup                       319 non-null    int64
 12  Pont_Jacques_Cartier          209 non-null    float64
 13  Rachel / Hôtel de Ville       319 non-null    int64
 14  Rachel / Papineau             319 non-null    int64
 15  René-Lévesque                 319 non-null    int64
 16  Saint-Antoine                 319 non-null    int64
 17  Saint-Laurent U-Zelt Test     50 non-null     float64
 18  Saint-Urbain                  319 non-null    int64
 19  Totem_Laurier                 319 non-null    int64
 20  University                    319 non-null    int64
 21  Viger                         319 non-null    int64
dtypes: float64(4), int64(17), object(1)
memory usage: 57.3+ KB
None
<class 'pandas.core.frame.DataFrame'>
```

Fig. 2 data missing

2. Data processing

```
#Differentiate between weekdays and weekends  0 is Monday
berri_bikes = Biks
berri_bikes.index
berri_bikes.index.day
berri_bikes.index.weekday
berri_bikes.loc[:,'weekday'] = berri_bikes.index.weekday
print(berri_bikes)
```

Based on how often we use bikes in our daily lives, we think it is the weekday that has some influence on the number of bikes. Therefore, we added the new feature of the weekday based on the date. The processed data is shown in Fig. 3.

```
                  Time  Berri1  Boyer  ...  University  Viger  weekday
Date                                   ...
2015-01-01  00:00       58     12  ...          21      6        3
2015-02-01  00:00       75      7  ...          77      4        6
2015-03-01  00:00       79      7  ...          40      5        6
2015-04-01  00:00       10      1  ...           6      0        2
2015-05-01  00:00       42      0  ...          56     10        4
...           ...      ...    ...  ...         ...    ...      ...
2015-11-11  00:00     3044   1931  ...        2860    356        2
2015-12-11  00:00     1751    930  ...        1777    198        4
2015-11-13  00:00     1818    906  ...        1727    258        4
2015-11-14  00:00      979    759  ...         737     73        5
2015-11-15  00:00      913    749  ...         685     63        6

[319 rows x 23 columns]
```

Fig. 3 data processing

3. Data analysis

```
Biks.plot(figsize=(15, 10))
plt.show()
```

(1) First, let's look at the number of bikes in each lane in 2015. We found that the number of lanes has a certain pattern, and the pattern between lanes is very similar, as show in Fig. 4. So, we'll take one of these lanes and analyze it in detail.

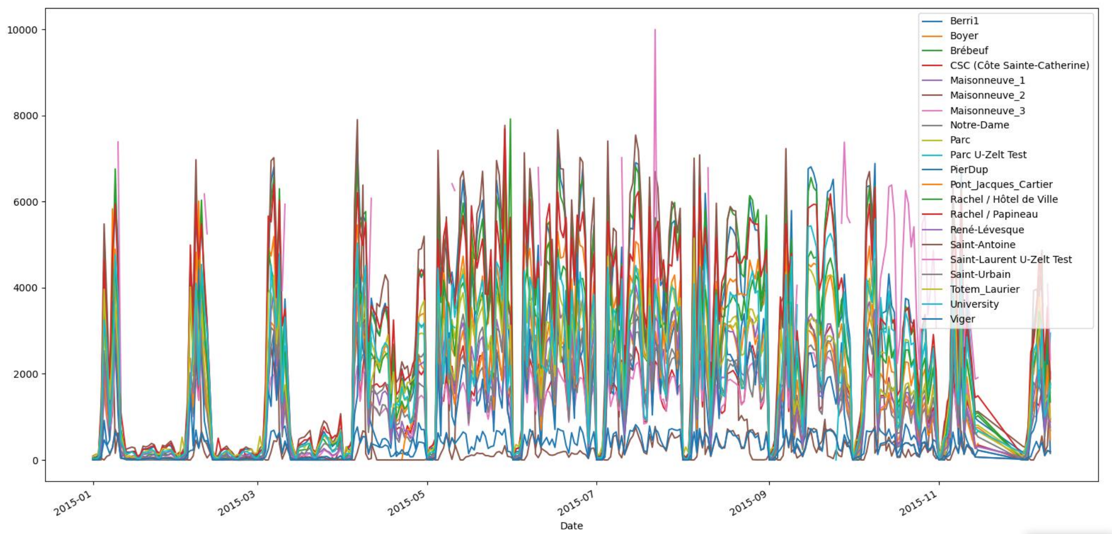
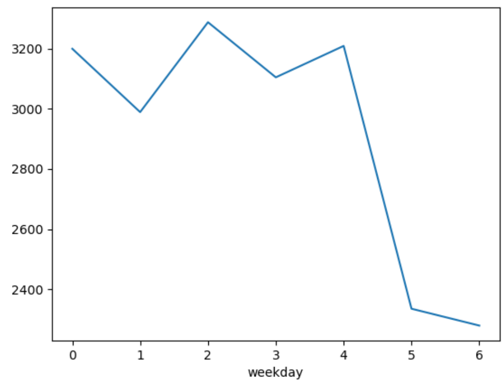


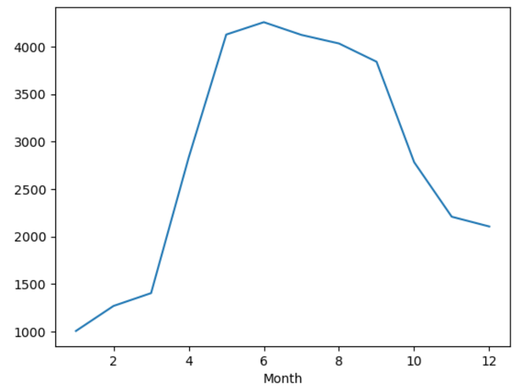
Fig. 4 The number of bicycles in each lane in 2015year

```
all_df=berri_bikes.join(weather)
print(all_df)

all_df.groupby(['weekday'])['Berri1'].mean().plot(kind='line')
plt.show()

all_df.groupby(['Month'])['Berri1'].mean().plot(kind='line')
plt.show()

all_df.groupby(['Day'])['Berri1'].mean().plot(kind='line')
plt.show()

all_df.groupby(['Max Temp (°C)'])['Berri1'].mean().plot(kind='line')
plt.show()
```

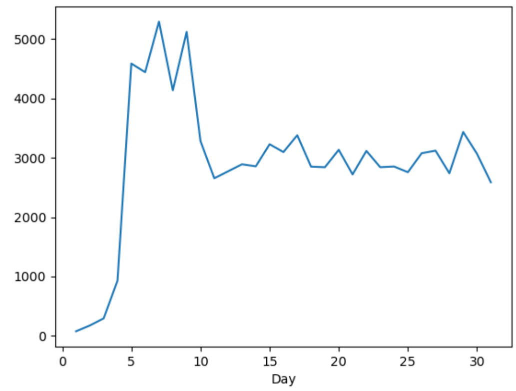
(2) Then, we analyze the influence of weekday, Month, Day and Max Temp on the number of bicycles in Berri1 lane, as show in Fig. 5.



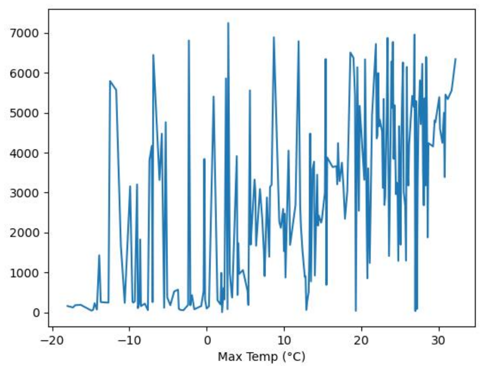
(a)



(b)



(c)



(d)

Fig. 5 The influence of each feature on the number of bicycles

From the Fig. 5, we can draw some conclusions. The demand for bicycles on weekdays is greater than that on weekends, and the demand for bicycles at the beginning of the month is smaller than that at the end of the month. Taken together, it can be seen that there are a certain proportion of office workers among the cycling crowd, and this part of users' demand for rental cars on non-holidays and working days will be released. Further analyzing the bicycle demand of each month, the demand of spring and winter months is lower than that of summer and autumn, and it can be seen that the temperature change of each quarter is highly correlated with the bicycle demand of each quarter. In addition, we also analyzed the influence of the maximum temperature on the number of bicycles, as shown in Fig. 5 (d). It can be preliminarily concluded that temperature is an important factor affecting the demand for bicycles.

```
corr = all_df.corr()
mask = np.array(corr)
mask[np.tril_indices_from(mask)] = False

plt.subplots(figsize=(10, 10))
sn.heatmap(corr, mask=mask,vmax=.8, square=True,annot=True)
plt.ylim(0,len(corr))
plt.tight_layout()

plt.show()
print(all_df.info())
```

(3) **Correlation analysis:**

We drew the relevant thermal maps of max temp, min temp and other characteristics, as shown in Fig. 6.

Fig. 6 Thermal map

From the Fig. 6, We can draw some conclusions:

- Snow on Grnd Flag, Max temp flag, Year is not a really useful numeric feature, as can be seen from its correlation with Berri1.
- Heat Deg Day was negatively correlated with max temp, min Temp and Mean Temp. Snow on Grnd shows the same negative correlation. Meanwhile, Mean Temp is positively correlated with max temp and min Temp. So we can delete Heat Deg Day, Mean Temp, Snow on Grnd.

```python
from sklearn.model_selection import train_test_split

dropFeatures = ["Time","Year","Data Quality","Max Temp Flag","Min Temp Flag","Mean Temp (°C)","Mean Temp Flag","Heat Deg Days Flag",
                "Cool Deg Days Flag","Total Rain (mm)","Total Rain Flag","Total Snow Flag","Total Precip (mm)","Total Precip Flag",
               "Snow on Grnd (cm)","Snow on Grnd Flag","Dir of Max Gust (10s deg)","Dir of Max Gust Flag","Spd of Max Gust (km/h)","Spd of Max Gust Flag","Unnamed: 27"
              ,"Heat Deg Days (°C)","Cool Deg Days (°C)","Total Snow (cm)"
]

X1 = all_df.drop(dropFeatures, axis=1).copy()
X = X1.drop("Berri1", axis=1).copy()
yLabels = X1.drop(X, axis=1).copy()

train_x, test_x, train_y, test_y = train_test_split(X, yLabels, test_size=0.3, random_state=42)

train=pd.concat([train_x, train_y], axis=1)
test=pd.concat([test_x, test_y], axis=1)
train_x.to_csv('train.txt', index=False, sep='\t')
test_x.to_csv('test.txt', index=False, sep='\t')
```

(4) **Regression analysis:**

According to the above analysis, we processed the data, eliminated the irrelevant features, and finally put the relevant features into the symbolic regression model for prediction. Fig. 7 shows the data characteristics used for symbolic regression training. We will use PFGP algorithm to train and test it. Please see the **src** folder for specific codes.

	weekday	Month	Day	Max Temp (°C)	Min Temp (°C)
2015-01-01 00:00:00	3	1	1	-3.00000	-7.60000
2015-02-01 00:00:00	6	2	1	-14.20000	-20.20000
2015-03-01 00:00:00	6	3	1	-3.40000	-17.30000
2015-04-01 00:00:00	2	4	1	2.00000	-6.70000
2015-05-01 00:00:00	4	5	1	19.30000	7.60000
2015-06-01 00:00:00	0	6	1	14.00000	9.10000
2015-07-01 00:00:00	2	7	1	23.00000	15.30000
2015-08-01 00:00:00	5	8	1	25.80000	16.00000
2015-09-01 00:00:00	1	9	1	27.20000	14.20000
2015-10-01 00:00:00	3	10	1	14.40000	4.20000
2015-11-01 00:00:00	6	11	1	12.90000	5.50000
2015-12-01 00:00:00	1	12	1	4.10000	-7.30000
2015-01-13 00:00:00	1	1	13	-17.00000	-23.10000
2015-01-14 00:00:00	2	1	14	-14.40000	-23.30000
2015-01-15 00:00:00	3	1	15	-4.70000	-20.80000
2015-01-16 00:00:00	4	1	16	-2.40000	-20.20000
2015-01-17 00:00:00	5	1	17	-14.90000	-23.90000
2015-01-18 00:00:00	6	1	18	2.70000	-15.20000
2015-01-19 00:00:00	0	1	19	1.40000	-13.60000
2015-01-20 00:00:00	1	1	20	-13.70000	-18.30000

X 格式: %s

已着色单元

Fig. 7 Features used for symbolic regression

Contact information of the participants:

Group Member 1: Junlan Dong, Email: eru-dd@foxmail.com

Group Member 3: Lianjie Zhong, Email: jackyzhong99@qq.com

Group Member 4: Nikola Gligorovski, Email: nikola-gligorovski@qq.com

Group Member 2: Jinghui Zhong, Email: jinghuizhong@suct.edu.cn