Software Engineering Tools Lab

Assignment No-2

(Module 2- Software Development Frameworks)

Batch – T8

Sr. No.	Name	PRN
1	Sayali Yogesh Desai	2020BTECS00206
2	Tejal Satish Pansare	2020BTECS00203

1. List of Frameworks/IDEs/Software's

   a. Eclipse

   b. Android SDK

   c. Node.js

   d. DotNet

   e. Ruby on Rails

   f. Anaconda

   g. Google Colab

For every Frameworks/IDEs/Software's given above provide the answers for below questions

We are choosing **Node.js** as Frameworks/IDEs/Software's for below questions

1	Original author	Ryan Dhal
2	Developers	Open JS Foundation
3	Initial release	May 27, 2009; 12 years ago
4	Stable release	17.4.0/ January 18, 2022; 21 days ago
5	Preview release	0.10.42/ February 2016
6	Repository (with cloud support)	https://github.com/nodejs/node
7	Written in (Languages)	C, C++, JavaScript
8	Operating System support	z/OS, Linux, MacOS, Microsoft Windows, SmartOS, FreeBSD, OpenBSD, IBM AIX
9	Platform, portability	Cross-Platform, iTwin.js
10	Available in (Total languages)	1

11	List of languages supported	JavaScript (CoffeeScript, Dart, TypeScript, ClojureScript and others)
12	Type (Programming tool, integrated development environment etc.)	Runtime Environment
13	Website	https://nodejs.org/
14	Features	Single Threaded, Asynchronous, Event Driven, Open Source, Fast Performance, Highly Scalable, No Buffering, Caching, Licensed
15	Size (in MB, GB etc.)	By default, Node.js (up to 11. x) uses a maximum heap size of 700MB and 1400MB on 32-bit and 64-bit platforms, respectively.
16	Privacy and Security	NPM Phishing, Regular expressions Denial of Service (DOS)
17	Type of software (Open source/License)	License
18	If License- Provide details	MIT License
19	Latest version	Node v17.4.0
20	Cloud support (Yes/No)	Yes
21	Applicability	▪ Internet of Things ▪ Real-Time Chats ▪ Complex Single-Page Applications ▪ Real-Time Collaboration Tools ▪ Streaming apps ▪ Microservices Architecture
22	Drawbacks (if any)	▪ Inability to process CPU bound ▪ Cell back hell issue ▪ Application Programming Interface is not scalable ▪ Performance bottlenecks with heavy computation

1. Implement linear regression problem using Google colab

(Perform preprocessing, training and testing)

Dataset 1- https://www.kaggle.com/spittman1248/cdc-data-nutrition-physical-activity-obesity

Dataset 2- https://archive.ics.uci.edu/ml/datasets/Air+Quality

Dataset 3- https://archive.ics.uci.edu/ml/datasets/Appliances+energy+prediction

Dataset 4- https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset

Dataset 5- https://archive.ics.uci.edu/ml/datasets/Demand+Forecasting+for+a+store

Dataset 6- https://archive.ics.uci.edu/ml/datasets/Hungarian+Chickenpox+Cases

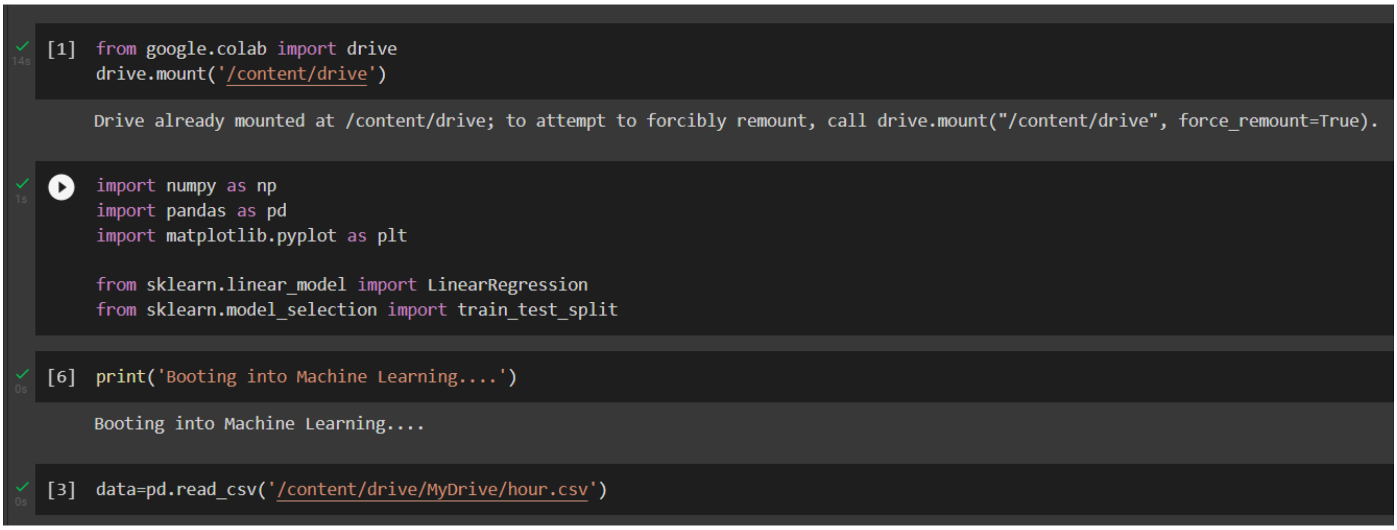
Dataset 7- https://archive.ics.uci.edu/ml/datasets/KDD+Cup+1998+Data

Dataset 8- https://archive.ics.uci.edu/ml/datasets/Water+Quality+Prediction

We have used Dataset no. 4 i.e. https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset for implementation of linear regression problem using Google colab.

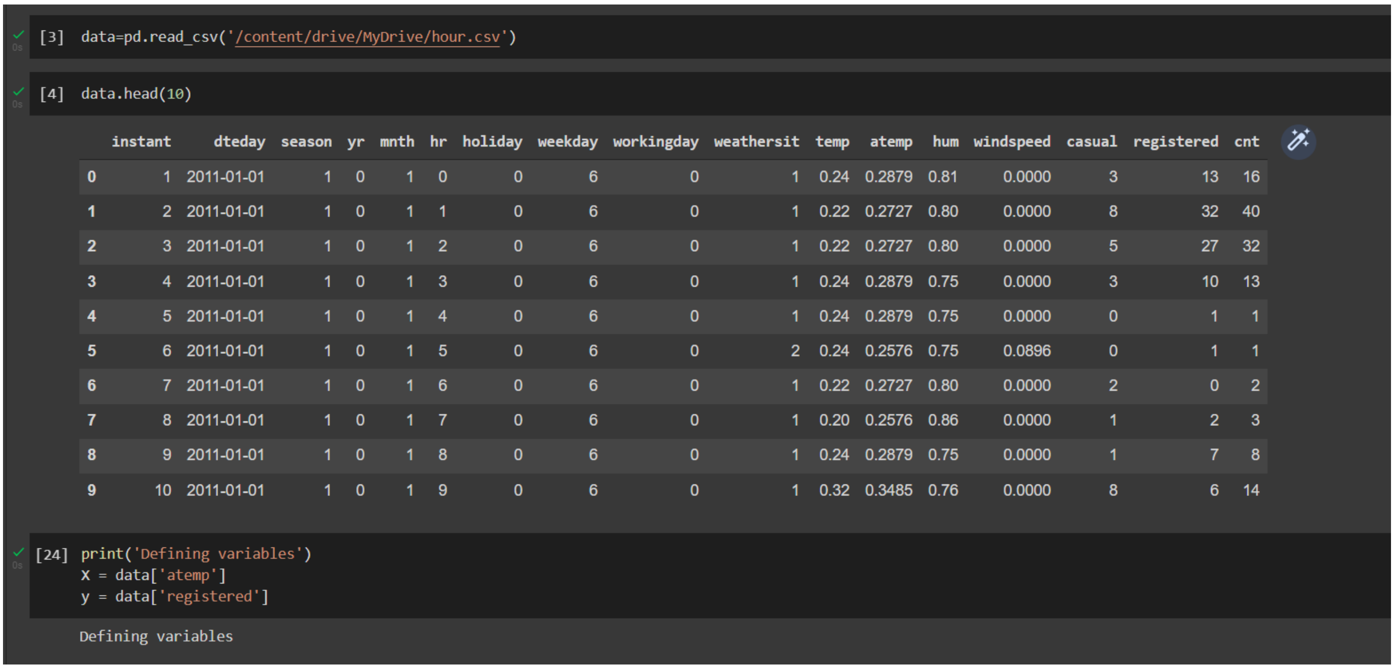
My Performance: https://colab.research.google.com/drive/1nZ-l8Cj74cL_7zBmk8yyGbzrflY-tDD_?usp=sharing

- **Preprocessing:**



```
[1] from google.colab import drive
    drive.mount('/content/drive')

    Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

    import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt

    from sklearn.linear_model import LinearRegression
    from sklearn.model_selection import train_test_split

[6] print('Booting into Machine Learning....')

    Booting into Machine Learning....

[3] data=pd.read_csv('/content/drive/MyDrive/hour.csv')
```

- **Training:**

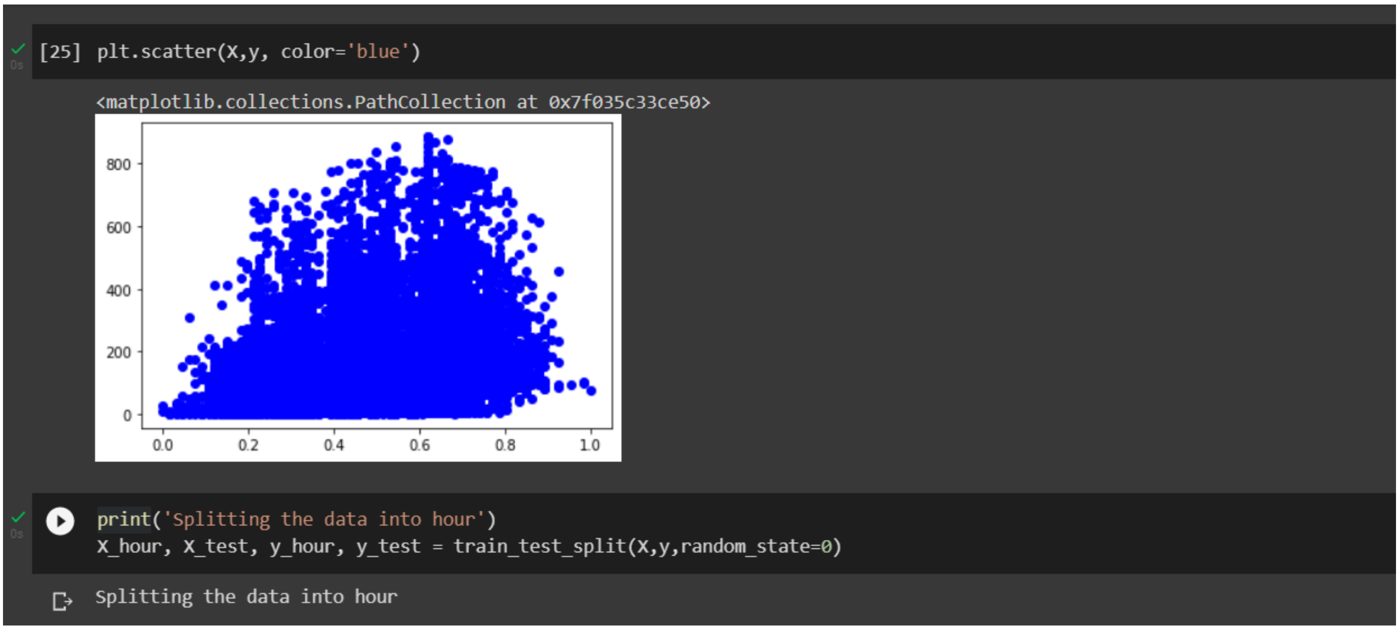


```
[3]  data=pd.read_csv('/content/drive/MyDrive/hour.csv')
```

```
[4]  data.head(10)
```

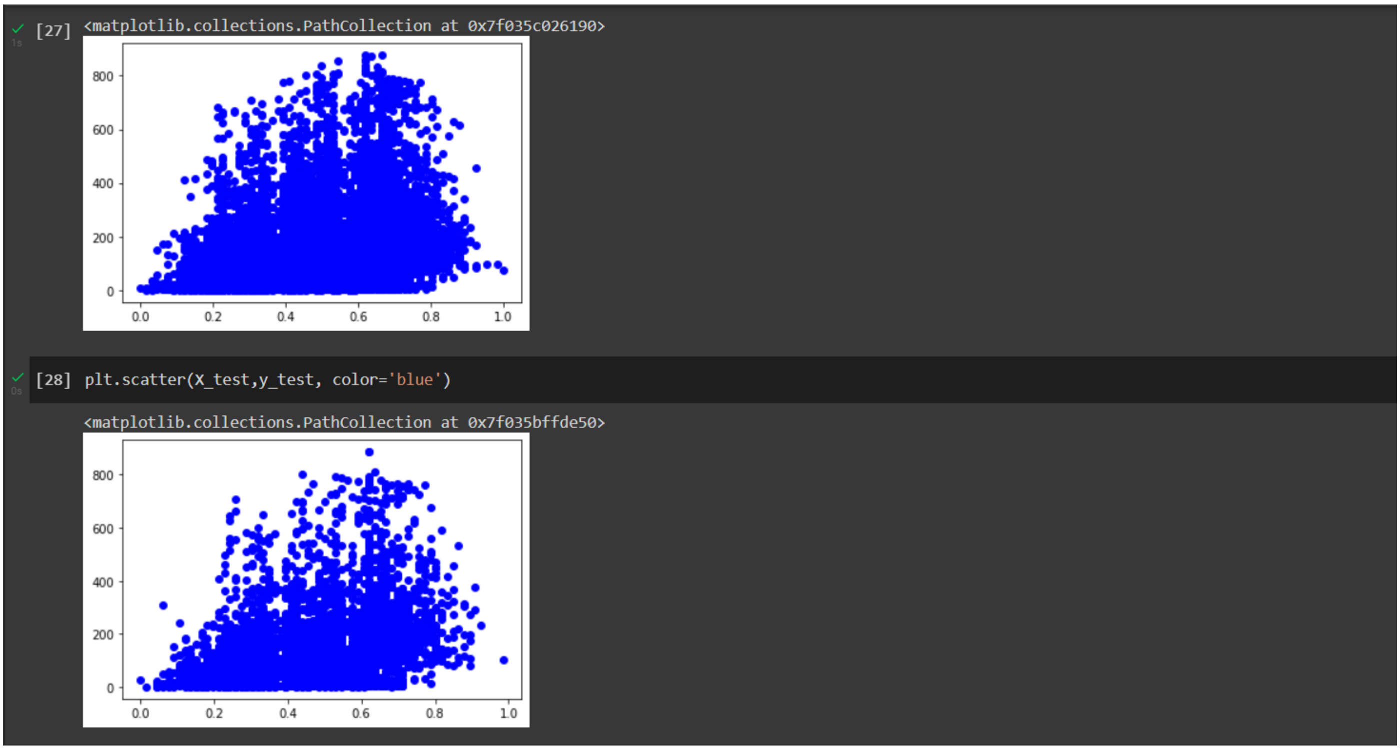
	instant	dteday	season	yr	mnth	hr	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	0	6	0	1	0.24	0.2879	0.81	0.0000	3	13	16
1	2	2011-01-01	1	0	1	1	0	6	0	1	0.22	0.2727	0.80	0.0000	8	32	40
2	3	2011-01-01	1	0	1	2	0	6	0	1	0.22	0.2727	0.80	0.0000	5	27	32
3	4	2011-01-01	1	0	1	3	0	6	0	1	0.24	0.2879	0.75	0.0000	3	10	13
4	5	2011-01-01	1	0	1	4	0	6	0	1	0.24	0.2879	0.75	0.0000	0	1	1
5	6	2011-01-01	1	0	1	5	0	6	0	2	0.24	0.2576	0.75	0.0896	0	1	1
6	7	2011-01-01	1	0	1	6	0	6	0	1	0.22	0.2727	0.80	0.0000	2	0	2
7	8	2011-01-01	1	0	1	7	0	6	0	1	0.20	0.2576	0.86	0.0000	1	2	3
8	9	2011-01-01	1	0	1	8	0	6	0	1	0.24	0.2879	0.75	0.0000	1	7	8
9	10	2011-01-01	1	0	1	9	0	6	0	1	0.32	0.3485	0.76	0.0000	8	6	14

```
[24]  print('Defining variables')
      X = data['atemp']
      y = data['registered']

      Defining variables
```



```
[25]  plt.scatter(X,y, color='blue')

      <matplotlib.collections.PathCollection at 0x7f035c33ce50>
```

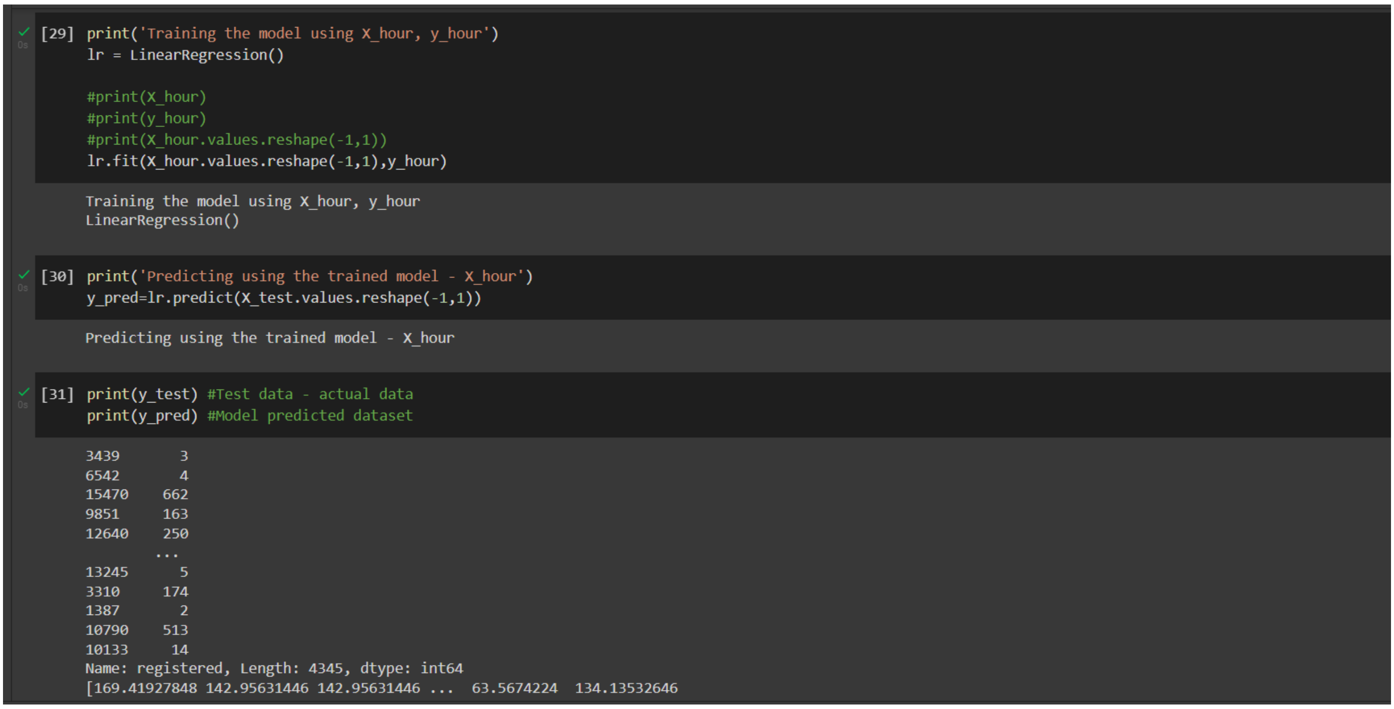
```
print('Splitting the data into hour')
X_hour, X_test, y_hour, y_test = train_test_split(X,y,random_state=0)

Splitting the data into hour
```



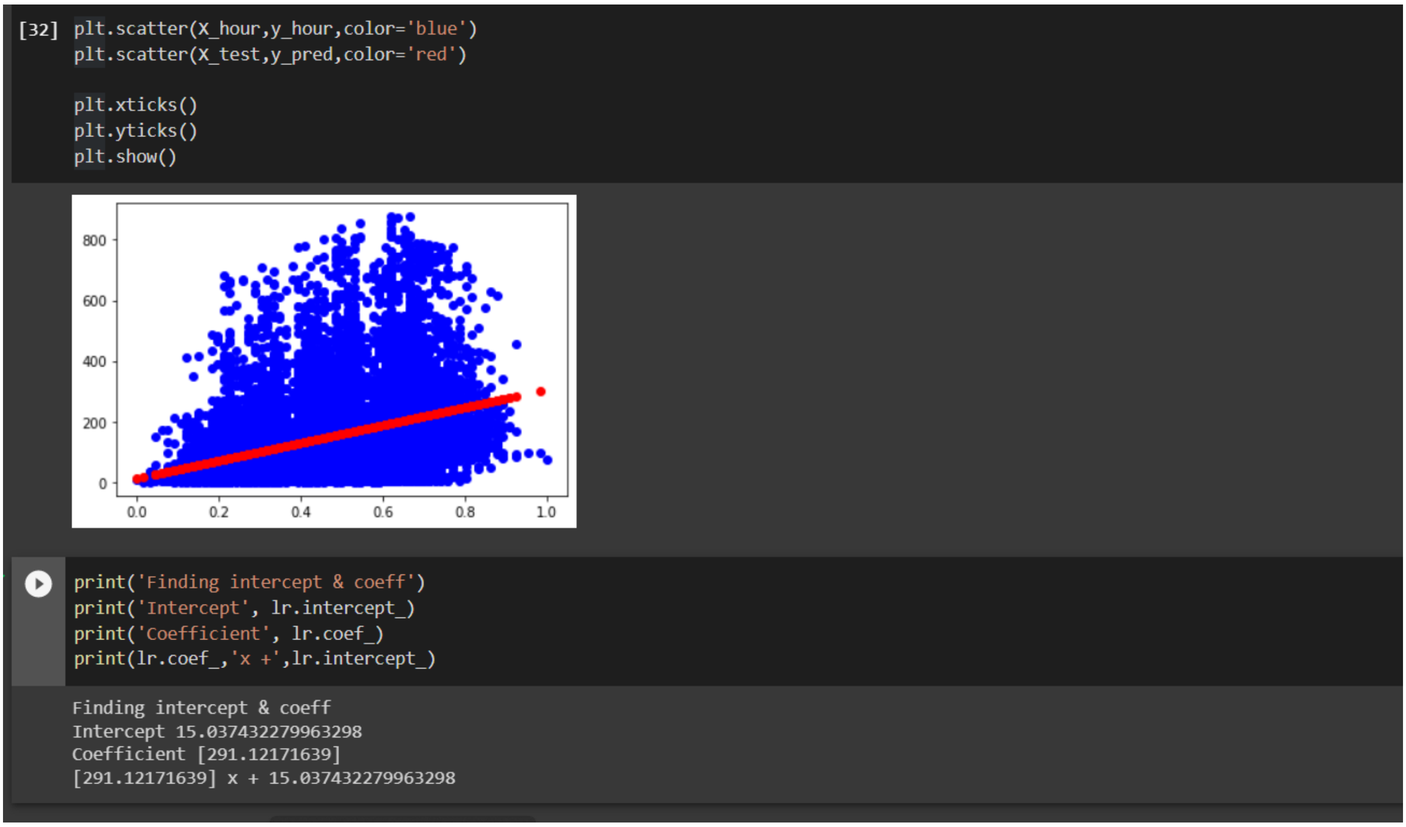
```
[27] <matplotlib.collections.PathCollection at 0x7f035c026190>
```

```
[28] plt.scatter(X_test,y_test, color='blue')

     <matplotlib.collections.PathCollection at 0x7f035bffde50>
```

- **Testing:**



```
[29] print('Training the model using X_hour, y_hour')
     lr = LinearRegression()

     #print(X_hour)
     #print(y_hour)
     #print(X_hour.values.reshape(-1,1))
     lr.fit(X_hour.values.reshape(-1,1),y_hour)

     Training the model using X_hour, y_hour
     LinearRegression()

[30] print('Predicting using the trained model - X_hour')
     y_pred=lr.predict(X_test.values.reshape(-1,1))

     Predicting using the trained model - X_hour

[31] print(y_test) #Test data - actual data
     print(y_pred) #Model predicted dataset

     3439       3
     6542       4
     15470    662
     9851     163
     12640    250
             ...
     13245      5
     3310     174
     1387       2
     10790    513
     10133     14
     Name: registered, Length: 4345, dtype: int64
     [169.41927848 142.95631446 142.95631446 ...  63.5674224  134.13532646
```



```
plt.scatter(X_hour,y_hour,color='blue')
plt.scatter(X_test,y_pred,color='red')

plt.xticks()
plt.yticks()
plt.show()
```

```
print('Finding intercept & coeff')
print('Intercept', lr.intercept_)
print('Coefficient', lr.coef_)
print(lr.coef_,'x +',lr.intercept_)
```

```
Finding intercept & coeff
Intercept 15.037432279963298
Coefficient [291.12171639]
[291.12171639] x + 15.037432279963298
```