In [82]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as py
from sklearn.linear_model import LogisticRegression
```

In [83]:
```
df=pd.read_csv("C9_Data - C9_Data.csv")
df
```

Out[83]:

	row_id	user_id	timestamp	gate_id
0	0	18	2022-07-29 09:08:54	7
1	1	18	2022-07-29 09:09:54	9
2	2	18	2022-07-29 09:09:54	9
3	3	18	2022-07-29 09:10:06	5
4	4	18	2022-07-29 09:10:08	5
...	...	...	...	...
37513	37513	6	2022-12-31 20:38:56	11
37514	37514	6	2022-12-31 20:39:22	6
37515	37515	6	2022-12-31 20:39:23	6
37516	37516	6	2022-12-31 20:39:31	9
37517	37517	6	2022-12-31 20:39:31	9

37518 rows × 4 columns

In [84]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 37518 entries, 0 to 37517
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   row_id     37518 non-null  int64 
 1   user_id    37518 non-null  int64 
 2   timestamp  37518 non-null  object
 3   gate_id    37518 non-null  int64 
dtypes: int64(3), object(1)
memory usage: 1.1+ MB
```

In [85]:
```
df1=df.fillna(value=0)
df1
```

Out[85]:

	row_id	user_id	timestamp	gate_id
0	0	18	2022-07-29 09:08:54	7
1	1	18	2022-07-29 09:09:54	9
2	2	18	2022-07-29 09:09:54	9
3	3	18	2022-07-29 09:10:06	5
4	4	18	2022-07-29 09:10:08	5
...	...	...	...	...
37513	37513	6	2022-12-31 20:38:56	11
37514	37514	6	2022-12-31 20:39:22	6
37515	37515	6	2022-12-31 20:39:23	6
37516	37516	6	2022-12-31 20:39:31	9
37517	37517	6	2022-12-31 20:39:31	9

37518 rows × 4 columns

In [86]:
```
df1.columns
```

Out[86]:
```
Index(['row_id', 'user_id', 'timestamp', 'gate_id'], dtype='object')
```

In [87]:
```
df2=df1[['row_id', 'user_id','gate_id']]
df2
```

Out[87]:

	row_id	user_id	gate_id
0	0	18	7
1	1	18	9
2	2	18	9
3	3	18	5
4	4	18	5
...	...	...	...
37513	37513	6	11
37514	37514	6	6
37515	37515	6	6
37516	37516	6	9
37517	37517	6	9

37518 rows × 3 columns

linear

In [88]:
```
x=df2[['row_id', 'user_id',]]
y=df2[ 'gate_id']
```

In [89]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [90]:
```
from sklearn.linear_model import LinearRegression
```

In [91]:
```
lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[91]:
```
LinearRegression()
```

In [92]:
```
print(lr.intercept_)
```

```
7.307956253097548
```

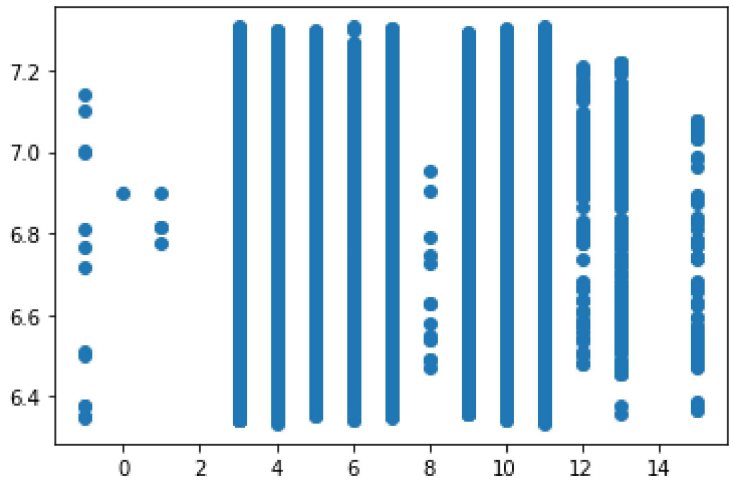
In [93]:
```
coeff =pd.DataFrame(lr.coef_,x.columns,columns=["Co-efficient"])
coeff
```

Out[93]:

	Co-efficient
row_id	-0.000007
user_id	-0.012718

In [94]:
```
prediction =lr.predict(x_test)
py.scatter(y_test,prediction)
```

Out[94]:
```
<matplotlib.collections.PathCollection at 0x1b7c08ab190>
```



In [95]:
```
print(lr.score(x_test,y_test))
```

```
0.005460436791627221
```

In [96]:
```
print(lr.score(x_train,y_train))
```

```
0.0055159346403285126
```

LOgistic

In [97]:
```
feature_matrix=df2.iloc[:,0:3]
target_vector=df2.iloc[:,1]
```

In [98]:
```
feature_matrix.shape
```

Out[98]:
```
(37518, 3)
```

In [99]:
```
target_vector.shape
```

Out[99]:
```
(37518,)
```

In [100]:
```
from sklearn.preprocessing import StandardScaler
```

In [101]:
```
fs=StandardScaler().fit_transform(feature_matrix)
```

In [102]:
```
logr =LogisticRegression()
logr.fit(fs,target_vector)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:
763: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[102]:
```
LogisticRegression()
```

In [103]:
```
observation=[[1.4,5.0,11]]
```

In [104]:
```
prediction=logr.predict(observation)
print(prediction)
```

```
[57]
```

In [105]:
```
logr.classes_
```

Out[105]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 14, 15, 17, 18,
       19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35,
       36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52,
       53, 54, 55, 56, 57], dtype=int64)
```

In [106]:
```
logr.predict_proba(observation)[0][0]
```

Out[106]:
```
6.157365479828746e-197
```

In [107]:
```
logr.predict_proba(observation)[0][1]
```

Out[107]:
```
5.748600667868822e-174
```