

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [2]: from sklearn.linear_model import LogisticRegression
```

```
In [3]: df=pd.read_csv("C9 Data csv").dropna()
df
```

Out[3]:

	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

```
In [4]: df.dropna(inplace=True)
```

```
In [5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 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: 1.4+ MB
```

```
In [6]: feature_matrix = df[['row_id', 'user_id']]
target_vector = df['gate_id']
```

```
In [7]: feature_matrix.shape
```

```
Out[7]: (37518, 2)
```

```
In [8]: target_vector.shape
```

```
Out[8]: (37518,)
```

```
In [9]: from sklearn.preprocessing import StandardScaler
```

```
In [10]: fs = StandardScaler().fit_transform(feature_matrix)
```

```
In [11]: 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[11]: LogisticRegression()
```

```
In [12]: feature_matrix.shape
```

```
Out[12]: (37518, 2)
```

```
In [13]: target_vector.shape
```

```
Out[13]: (37518,)
```

```
In [14]: from sklearn.preprocessing import StandardScaler
```

```
In [15]: fs = StandardScaler().fit_transform(feature_matrix)
```

Regression

```
In [16]: 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[16]: LogisticRegression()
```

```
In [17]: observation=df[['row_id','user_id']]
```

```
In [18]: prediction = logr.predict(observation)  
prediction
```

```
Out[18]: array([-1, -1, -1, ..., 16, 16, 16], dtype=int64)
```

```
In [19]: logr.classes_
```

```
Out[19]: array([-1,  0,  1,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16],  
          dtype=int64)
```

```
In [20]: logr.predict_proba(observation)[0][1]
```

```
Out[20]: 1.7263815682078809e-09
```

```
In [21]: from sklearn.linear_model import Ridge,Lasso
```

```
In [22]: x = df[['row_id','user_id']]  
y = df['gate_id']
```

```
In [23]: 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 [24]: rr=Ridge(alpha=10)  
rr.fit(x_train,y_train)  
rr.score(x_test,y_test)  
rr.score(x_train,y_train)
```

```
Out[24]: 0.0052547442671367905
```

```
In [25]: from sklearn.linear_model import LinearRegression  
lr= LinearRegression()  
lr.fit(x_train,y_train)
```

Out[25]: LinearRegression()

```
In [26]: lr.intercept_
```

Out[26]: 7.302588405159492

```
In [27]: coeff = pd.DataFrame(lr.coef_,x.columns,columns=['Co-efficient'])  
coeff
```

Out[27]:

	Co-efficient
row_id	-0.000007
user_id	-0.012368

```
In [28]: prediction = lr.predict(x_test)  
plt.scatter(y_test,prediction)
```

Out[28]: <matplotlib.collections.PathCollection at 0x1c697d95700>

