In [1]:

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

In [2]:

```
df = pd.read_csv('citibike_tripdata.csv')
df.sample(15)
```

Out[2]:

	tripduration	starttime	stoptime	start station id	start station name	end station id	end station name	bikeid	name_
17819	427	2018-05-17 08:23:19	2018-05-17 08:30:26	3210	Pershing Field	3195	Sip Ave	26173	Annu
31672	157	2018-05-29 21:16:21	2018-05-29 21:18:59	3186	Grove St PATH	3213	Van Vorst Park	29654	Annu
23452	449	2018-05-22 18:24:51	2018-05-22 18:32:20	3185	City Hall	3269	Brunswick & 6th	33599	Annu
4896	133	2018-05-04 21:01:51	2018-05-04 21:04:05	3211	Newark Ave	3209	Brunswick St	32769	Annu
2326	687	2018-05-02 19:56:32	2018-05-02 20:08:00	3211	Newark Ave	3195	Sip Ave	26281	Annu
27921	252	2018-05-25 21:10:51	2018-05-25 21:15:04	3195	Sip Ave	3678	Fairmount Ave	29608	Annu fro
12762	295	2018-05-11 10:55:56	2018-05-11 11:00:52	3679	Bergen Ave	3195	Sip Ave	26268	
6166	272	2018-05-06 08:02:29	2018-05-06 08:07:02	3203	Hamilton Park	3186	Grove St PATH	29479	Annu
16426	176	2018-05-15 11:28:23	2018-05-15 11:31:20	3278	Monmouth and 6th	3272	Jersey & 3rd	29444	Annu
7443	283	2018-05-07 12:19:22	2018-05-07 12:24:05	3183	Exchange Place	3275	Columbus Drive	29244	(
32294	488	2018-05-30 11:11:47	2018-05-30 11:19:55	3202	Newport PATH	3639	Harborside	33573	Annu
26696	480	2018-05-25 07:28:03	2018-05-25 07:36:03	3192	Liberty Light Rail	3185	City Hall	29438	Annu
29026	273	2018-05-27 06:44:15	2018-05-27 06:48:49	3213	Van Vorst Park	3209	Brunswick St	33666	Annu
17562	575	2018-05-16 18:27:42	2018-05-16 18:37:18	3198	Heights Elevator	3207	Oakland Ave	29538	J
30685	520	2018-05-29 08:51:19	2018-05-29 08:59:59	3212	Christ Hospital	3185	City Hall	26219	Annu

In [3]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32428 entries, 0 to 32427
Data columns (total 10 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   tripduration         32428 non-null  int64 
 1   starttime            32428 non-null  object
 2   stoptime             32428 non-null  object
 3   start station id     32428 non-null  int64 
 4   start station name   32428 non-null  object
 5   end station id       32428 non-null  int64 
 6   end station name     32428 non-null  object
 7   bikeid               32428 non-null  int64 
 8   name_localizedValue  32428 non-null  object
 9   usertype             32428 non-null  object
dtypes: int64(4), object(6)
memory usage: 2.5+ MB
```

In [4]:

```
df['name_localizedValue'].unique()
```

Out[4]:

```
array(['Annual Membership', '24 Hour',
       'FREE Bonus Month with Annual Membership',
       'Annual Membership from Citi Bike App',
       '$25 Off Annual Membership', 'Join Citi Bike for $14.95/month',
       'Single Ride', 'JCBS Employee', 'Citi Bike Annual Membership',
       'Annual Membership - Save 15%', '3 Day', 'Day Pass',
       'NYCHA Membership (Renewal)', 'Join Citi Bike for $14/month',
       'Citi Bike for Business Annual Membership',
       '$35 Off Annual Membership', 'JCHA Annual Membership $5/month',
       'CDCU', 'NYCHA Membership', '3-Day Pass from Citi Bike App',
       'Single Ride POS',
       'Annual Membership - Save $25 to Celebrate Expansion',
       'Annual Membership - 10% Off', 'CDCU (renewal)',
       '$99 Annual Membership', 'Motivate Employee', 'NYCBS Employee',
       'NYCHA POS',
       'Bike Angels Reward Memberships - Annual member with removed 2 minu
te delay',
       '24 Hour from Citi Bike Squad', 'Free Day Pass', 'First Ride Fre
e'],
      dtype=object)
```

In [5]:

```
df.loc[df['usertype'] == 'Customer']
```

Out[5]:

	tripduration	starttime	stoptime	start station id	start station name	end station id	end station name	bikeid	name_l
1	1482	2018-05-01 01:31:10	2018-05-01 01:55:53	3681	Grand St	3185	City Hall	33593	
3	190	2018-05-01 02:03:29	2018-05-01 02:06:40	3185	City Hall	3186	Grove St PATH	29662	
35	367	2018-05-01 06:28:03	2018-05-01 06:34:10	3278	Monmouth and 6th	3186	Grove St PATH	26235	
139	699	2018-05-01 07:45:27	2018-05-01 07:57:07	3268	Lafayette Park	3214	Essex Light Rail	33562	
151	334	2018-05-01 07:52:39	2018-05-01 07:58:14	3201	Dey St	3195	Sip Ave	26178	
...	...	...	...	...	...	...	...	...	
32348	842	2018-05-30 12:15:18	2018-05-30 12:29:21	3194	McGinley Square	3195	Sip Ave	29463	3-Day
32352	278	2018-05-30 12:21:07	2018-05-30 12:25:46	3214	Essex Light Rail	3214	Essex Light Rail	33680	
32353	1004	2018-05-30 12:21:20	2018-05-30 12:38:04	3214	Essex Light Rail	3183	Exchange Place	29646	
32358	557	2018-05-30 12:28:48	2018-05-30 12:38:06	3214	Essex Light Rail	3183	Exchange Place	26198	
32391	4387	2018-05-30 13:12:04	2018-05-30 14:25:12	3213	Van Vorst Park	3213	Van Vorst Park	26272	

2375 rows × 10 columns

In [6]:

```
#Localizedvalue and usertype very much correlated, so an remove localizedvalue column
#Remove start/stop time, station id, bike id, start station name, stop name

tripdata = df.drop(columns = ['starttime', 'stoptime', 'start station id', 'start station
                              'end station name', 'bikeid', 'name_localizedValue'], axis=

tripdata['usertype'] = tripdata['usertype'].apply(lambda x : 1 if x == 'Subscriber' else
tripdata
```

Out[6]:

	tripduration	usertype
0	338	1
1	1482	0
2	232	1
3	190	0
4	303	1
...	...	...
32423	396	1
32424	313	1
32425	316	1
32426	1130	1
32427	369	1

32428 rows × 2 columns

In [7]:

```
from sklearn.model_selection import train_test_split
```

In [8]:

```
y = tripdata[['usertype']]
X = tripdata[['tripduration']]
```

In [9]:

```
X_train, X_val, y_train, y_val = train_test_split(X,y, test_size=0.3, random_state=42)
```

In [10]:

```
from sklearn.linear_model import LogisticRegression

lr = LogisticRegression()
```

In [11]:

```
lr.fit(X_train, y_train)
```

```
C:\Users\Debmalya\anaconda3\lib\site-packages\sklearn\utils\validation.py:
993: DataConversionWarning: A column-vector y was passed when a 1d array w
as expected. Please change the shape of y to (n_samples, ), for example us
ing ravel().
  y = column_or_1d(y, warn=True)
```

Out[11]:

```
LogisticRegression()
```

In [12]:

```
y_pred = lr.predict(X_val)
```

In [13]:

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

In [14]:

```
acc = accuracy_score(y_val, y_pred)
acc
```

Out[14]:

```
0.931236509404872
```

In [15]:

```
pre = precision_score(y_val, y_pred)
pre
```

Out[15]:

```
0.9323766260582284
```

In [16]:

```
recall = recall_score(y_val, y_pred)
recall
```

Out[16]:

```
0.9984521835268104
```

In [17]:

```
f1 = f1_score(y_val, y_pred)
f1
```

Out[17]:

```
0.964283807591693
```

In [ ]:

```

```