In [189]:
```
from pyforest import *
active_imports()
from datetime import datetime
from datetime import date
import seaborn as sns
import scipy.stats as stats

```

```
import pandas as pd
import matplotlib.pyplot as plt
```

ETL

In [190]:
```
df_train=pd.read_csv('TrainingData_V1.csv')
df_test=pd.read_excel('TestingData_For_Candidate.xlsx')
```

In [191]:
```
df_test['return']=-1
```

In [192]:
```
df=pd.concat([df_train,df_test])
```

In [193]:
```
df.dropna(inplace=True)# removing the null values
```

In [194]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 82749 entries, 0 to 20054
Data columns (total 14 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   order_item_id  82749 non-null  int64  
 1   order_date     82749 non-null  object 
 2   delivery_date  82749 non-null  object 
 3   item_id        82749 non-null  int64  
 4   item_size      82749 non-null  object 
 5   item_color     82749 non-null  object 
 6   brand_id       82749 non-null  int64  
 7   item_price     82749 non-null  float64
 8   user_id        82749 non-null  int64  
 9   user_title     82749 non-null  object 
 10  user_dob       82749 non-null  object 
 11  user_state     82749 non-null  int64  
 12  user_reg_date  82749 non-null  object 
 13  return         82749 non-null  int64  
dtypes: float64(1), int64(6), object(7)
memory usage: 9.5+ MB
```

In [195]:
```
1 df.head()
```

Out[195]:

	order_item_id	order_date	delivery_date	item_id	item_size	item_color	brand_id	item_price	u
0	1	22-06-2016	27-06-2016	643	38	navy	30	49.9	
1	10	22-06-2016	27-06-2016	195	xxl	grey	46	19.9	
2	11	22-06-2016	05-07-2016	25	xxl	grey	5	79.9	
3	32	23-06-2016	26-06-2016	173	m	brown	20	19.9	
5	45	23-06-2016	26-06-2016	448	42	bordeaux	72	59.9	

In [196]:
```
1 df['user_dob']=pd.to_datetime(df.user_dob)
2 df['order_date']=pd.to_datetime(df.order_date)
3 df['delivery_date']=pd.to_datetime(df.delivery_date)
```

In [197]:
```
1 df['user_year'] = pd.DatetimeIndex(df['user_dob']).year
```

In [198]:
```
1 df.user_year.unique()
```

Out[198]:
```
array([1969, 1970, 1960, 1966, 1963, 1959, 1950, 1967, 1973, 1964, 1968,
       1962, 1965, 1900, 1972, 1971, 1949, 1974, 1952, 1946, 1956, 1955,
       1958, 1989, 1982, 1961, 1957, 1975, 1947, 1954, 1981, 1953, 1951,
       1976, 1983, 1942, 1978, 1977, 1980, 1948, 1986, 1939, 1988, 1945,
       1979, 1984, 1906, 1987, 1901, 1943, 1985, 1940, 1944, 1992, 1991,
       1925, 1941, 1933, 1990, 1937, 2010, 2011, 1994, 1934, 1935, 1938,
       1931, 1995, 1930, 1936, 1993, 2005, 1911, 1903, 1998, 1912, 1929,
       1907, 1999], dtype=int64)
```

In [199]:
```
1 df['Age']=date.today().year-df['user_year']
```

In [200]:
```
1 df.drop(['user_dob'],axis=1,inplace=True)
```

In [201]:
```
1 df['deliver']=(df['delivery_date']-df['order_date']).dt.days
```

In [202]:
```
1 len(df[df['deliver']<0]) #removing these rows as it gives ambigous result
```

Out[202]:
```
15707
```

In [203]:
```
1 df=df[df['deliver']>=0]
```

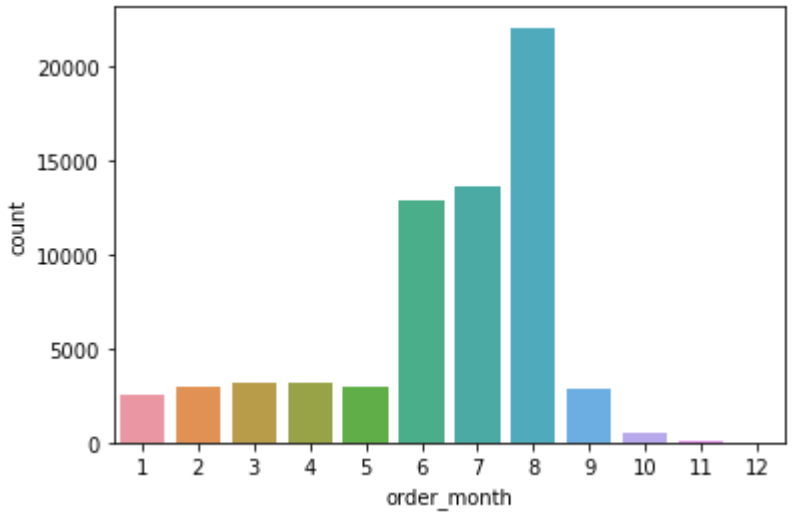
In [204]:
```
1 df['order_month'] = pd.DatetimeIndex(df['order_date']).month
```

EDA

In [205]:
```
1 sns.countplot(df.order_month)
```

```
C:\Users\Rahul\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variable as a keyword arg: x. From version 0.12, the o
nly valid positional argument will be `data`, and passing other arguments witho
ut an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[205]:
```
<AxesSubplot:xlabel='order_month', ylabel='count'>
```

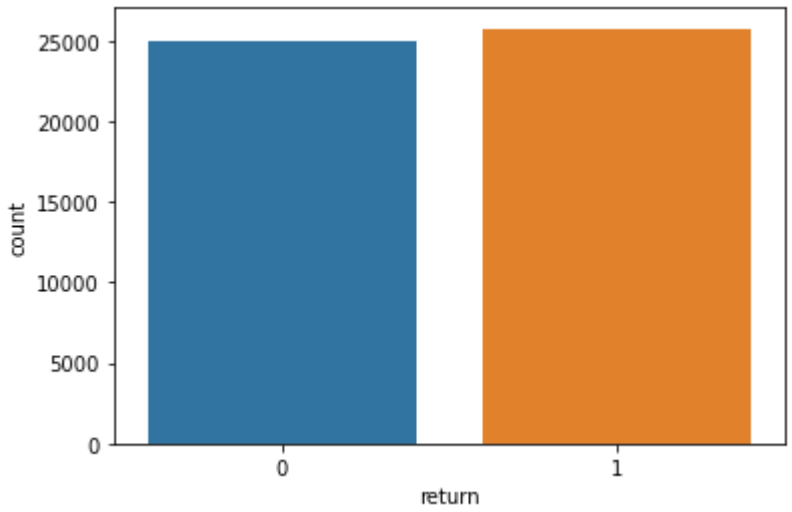


The graphy shows the no of order in each month where the x axis shows the order month and the y axis shows the total number of order in that month. It is evident from the bar chart that the majority of booking is done in the months june,July,August.August has the maximum no of order with more than 20000 orders.Next comes the month of july and then june.

In [206]:
```
1 sns.countplot(df[df['return']!=-1]['return'])# we can see that it is pretty
```

```
C:\Users\Rahul\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWar
ning: Pass the following variable as a keyword arg: x. From version 0.12, the o
nly valid positional argument will be `data`, and passing other arguments witho
ut an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[206]:
```
<AxesSubplot:xlabel='return', ylabel='count'>
```



In [207]:
```
1 df.brand_id.nunique()
```

Out[207]:
```
130
```

In [208]:
```
df[df['return']==1].groupby('brand_id')['return'].count().sort_values(ascend
```

Out[208]:
```
brand_id
3      2400
1      2291
11     1458
37     1179
5      1024
20      887
6       740
113     664
17      653
43      583
Name: return, dtype: int64
```

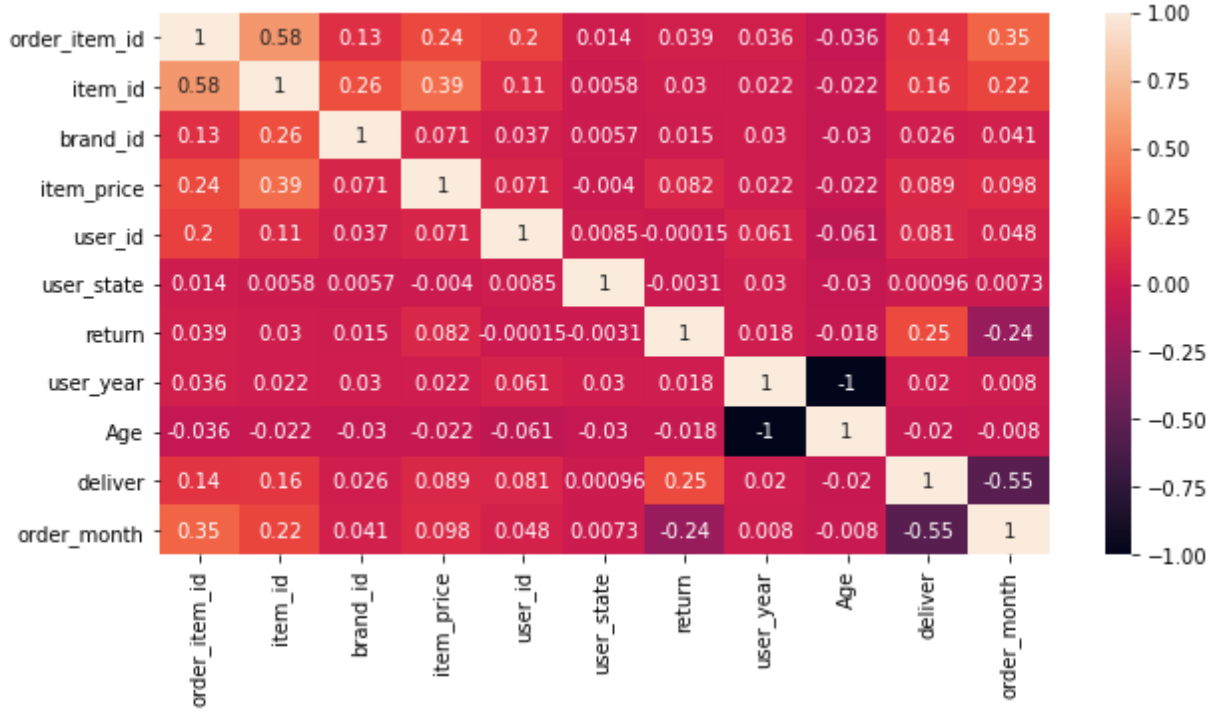
In [209]:
```
df[df['return']==1].groupby('brand_id')['return'].count().sort_values()[0:10
```

Out[209]:
```
brand_id
131    1
85     1
125    1
98     1
100    1
65     1
107    1
62     2
103    2
82     2
Name: return, dtype: int64
```

Feature Selection

In [210]:
```
1 plt.figure(figsize=[10,5])
2 sns.heatmap(df.corr(),annot=True)
```

Out[210]: <AxesSubplot:>



In [211]:
```
1 df.user_title.unique()
```

Out[211]: array(['Mrs', 'Family', 'Mr', 'Company', 'not reported'], dtype=object)

In [212]:
```
df.head()
```

Out[212]:

	order_item_id	order_date	delivery_date	item_id	item_size	item_color	brand_id	item_price	u
0	1	2016-06-22	2016-06-27	643	38	navy	30	49.9	
1	10	2016-06-22	2016-06-27	195	xxl	grey	46	19.9	
3	32	2016-06-23	2016-06-26	173	m	brown	20	19.9	
5	45	2016-06-23	2016-06-26	448	42	bordeaux	72	59.9	
6	48	2016-06-23	2016-06-26	32	l	white	3	21.9	

Chi Square

H0=There's no relationship between user_title and return H1=There's a relationship between user_title and return

In [213]:
```
contigency=pd.crosstab(df_train.user_title,df_train['return'], margins=True)
contigency
```

Out[213]:

return	0	1	All
user_title			
Company	41	64	105
Family	206	133	339
Mr	1814	1327	3141
Mrs	41160	35107	76267
not reported	62	31	93
All	43283	36662	79945

In [214]:
```
chi2, p, dof,ex=stats.chi2_contingency(contigeny,correction=False)
```

In [215]:
```
print('p_values: %.3f' % p)
print('chi2: %.3f' % chi2)
```

```
p_values: 0.000
chi2: 38.920
```

Hence we reject the null hypothesis as the p value is less than .05 %

Encoding

In [216]:
```
df=pd.get_dummies(df,columns=['user_title'],drop_first=True)

```

In [217]:
```
def delivery(on_time):
    if on_time>=5:
        return 'On_time'
    else:
        return'Late'

```

In [218]:
```
df.deliver=df.deliver.apply(delivery)
```

In [219]:
```
from sklearn import preprocessing

# label_encoder object knows how to understand word labels.
label_encoder = preprocessing.LabelEncoder()

# Encode labels in column 'species'.
df['deliver']= label_encoder.fit_transform(df['deliver'])

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

Out[219]:
```
array([1, 0])
```

Feature Selection

In [221]:
```
df=pd.get_dummies(df,columns=['user_state'],drop_first=True)
```

In [220]:
```
df.drop(columns=['order_date','delivery_date','user_reg_date','user_year','o
```

In [225]:
```
df_train=df[df['return']!=-1]
df_test=df[df['return']==-1]
```

In [228]:
```
X=df_train.drop('return',axis=1)
Y=df_train['return']
```

Modelling

In [230]:
```

#split data
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X,Y, test_size=0.20, ran
```

```
<frozen importlib._bootstrap>:219: RuntimeWarning: numpy.ufunc size changed, ma
y indicate binary incompatibility. Expected 192 from C header, got 216 from PyO
bject
```

In [231]:
```
# standardise
from sklearn import preprocessing
scalar = preprocessing.StandardScaler()

scalar.fit(x_train)
x_train_std = scalar.transform(x_train)
x_test_std = scalar.transform(x_test)
```

In [236]:
```
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier(n_estimators=500,bootstrap=True)
```

In [237]:
```
rf.fit(x_train_std,y_train)
```

Out[237]:
```
RandomForestClassifier(n_estimators=500)
```

In [245]:
```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

In [241]:
```
yp=rf.predict(x_train_std)
```

In [248]:
```
confusion_matrix(y_train,yp)
```

Out[248]:
```
array([[18704,  1197],
       [ 1018, 19612]], dtype=int64)
```

In [246]:
```
matrix = classification_report(y_train,yp,labels=[1,0])
print('Classification report : \n',matrix)
```

```
Classification report :
               precision    recall  f1-score   support

           1       0.94      0.95      0.95     20630
           0       0.95      0.94      0.94     19901

    accuracy                           0.95     40531
   macro avg       0.95      0.95      0.95     40531
weighted avg       0.95      0.95      0.95     40531
```

In [ ]:
```
ve 1made a sample model.I am getting a pretty good precision and recall score .With
```

In [251]:
```
1 df_test.drop('return',axis=1,inplace=True)
```

```
C:\Users\Rahul\anaconda3\lib\site-packages\pandas\core\frame.py:4163: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  return super().drop(
```

In [252]:
```
1 df_test=scalar.transform(df_test)
```

In [253]:
```
1 rf.predict(df_test)
```

Out[253]:
```
array([1, 0, 1, ..., 0, 1, 0], dtype=int64)
```

In [ ]:
```
1
```