cse519_hw2_Kurakula_Mohith_112504214

September 26, 2019

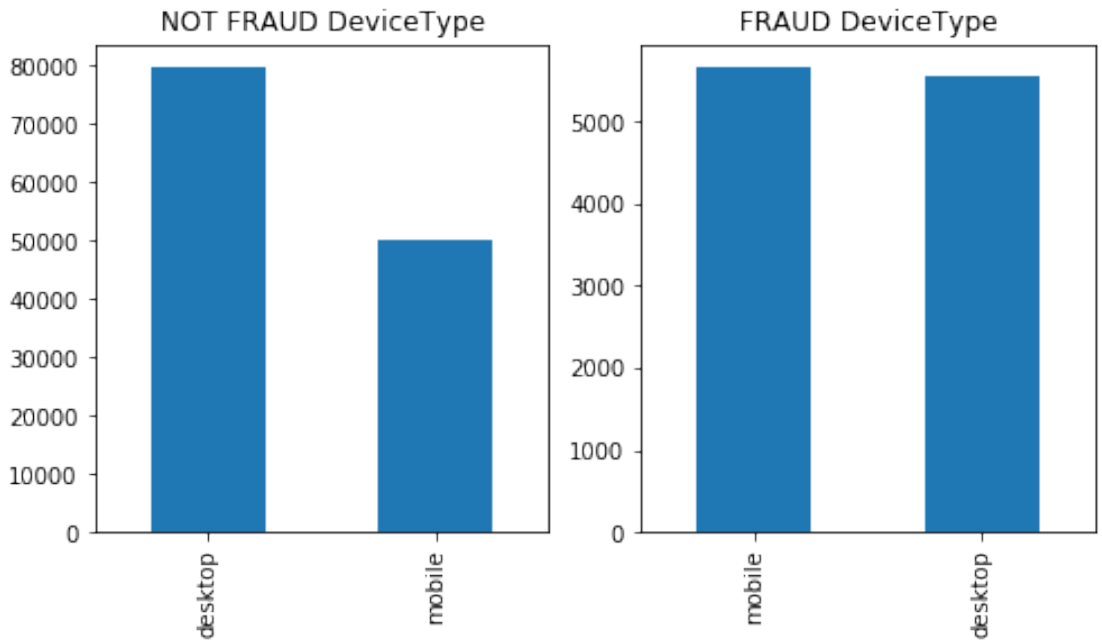
1 Homework 2 - IEEE Fraud Detection

2 # Part 1 - Fraudulent vs Non-Fraudulent Transaction

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
t_transaction=pd.read_csv("train_transaction.csv")
t_identity=pd.read_csv("train_identity.csv")
t_test_transaction=pd.read_csv("test_transaction.csv")
t_test_identity=pd.read_csv("test_identity.csv")
df_test=pd.merge(t_test_transaction, t_test_identity, on='TransactionID',␣
 ↪how='left')
df=pd.merge(t_transaction, t_identity, on='TransactionID', how='left')
```

[2]:
```
df_fraud=df.loc[df['isFraud']== 1]
df_nfraud=df.loc[df['isFraud']== 0]
plt.subplot(1, 2, 1)
df_nfraud['DeviceType'].value_counts().nlargest(6).plot(kind='bar',title="NOT␣
 ↪FRAUD DeviceType",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['DeviceType'].value_counts().nlargest(6).plot(kind='bar',title="FRAUD␣
 ↪DeviceType",figsize=(8,4))
```

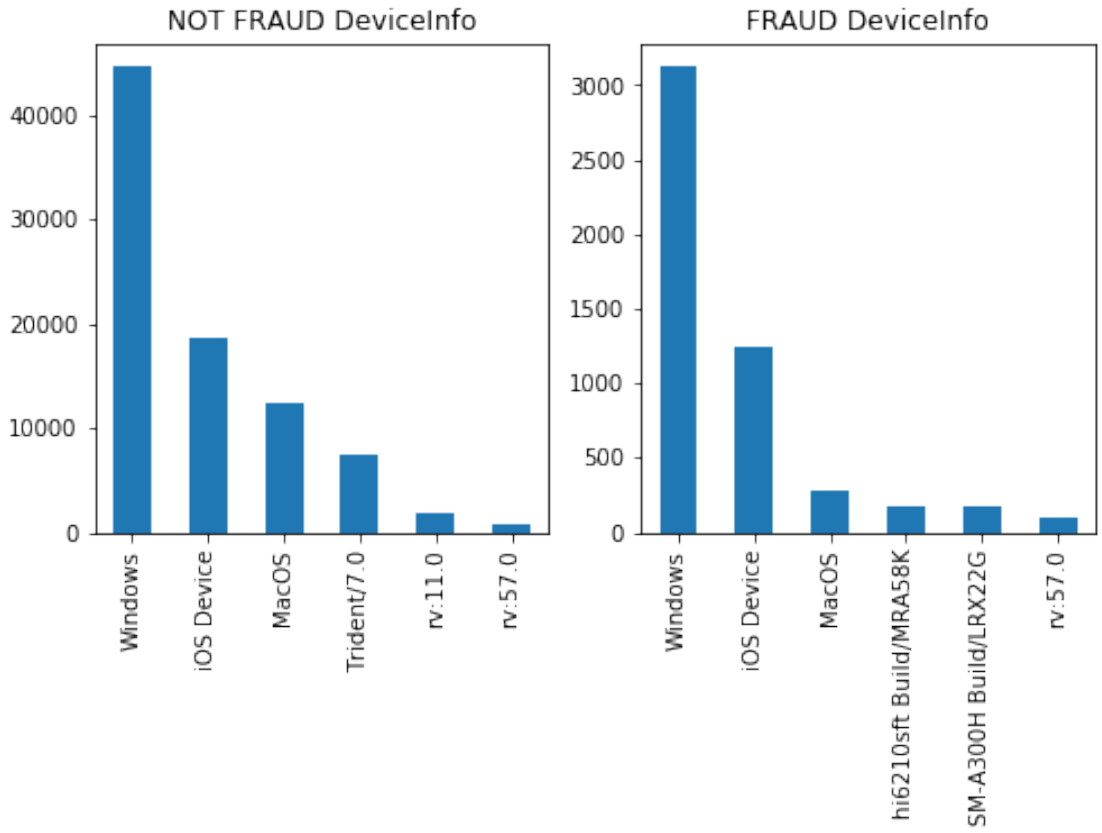
[2]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d249f044e0>
```



Above is the plot for DeviceType for fraud and Non fraud data. In the above plot it is clear that frequency of the desktops is more in the Not fraud that means desktops are being more used(in both) and more safer compared to mobiles and whereas in fraud plot the chances of occuring fraud in both the mobile and desktop are almost same that means desktops are more safe to use

[3]:
```
plt.subplot(1, 2, 1)
df_nfraud['DeviceInfo'].value_counts().nlargest(6).plot(kind='bar',title="NOT␣
 ↪FRAUD DeviceInfo",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['DeviceInfo'].value_counts().nlargest(6).plot(kind='bar',title="FRAUD␣
 ↪DeviceInfo",figsize=(8,4))
```

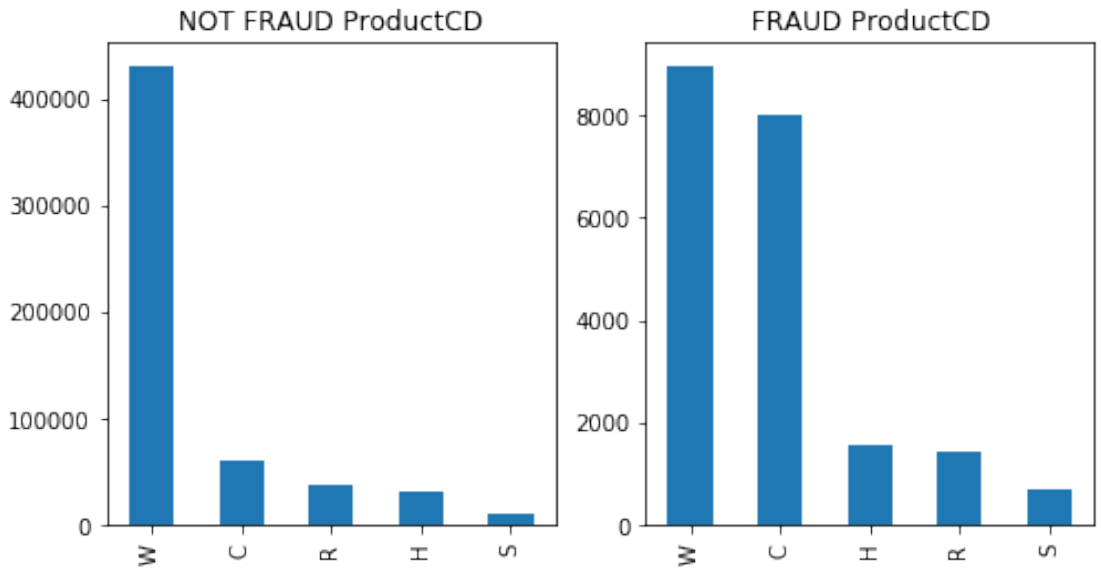
[3]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d24ada4c88>
```



Above is the plot for Deviceinfo for fraud and Non fraud data. In the above plot it is clear that frequency of the windows is more in the both Not fraud and fraud compared to others that means windows laptops are being used more and whereas if we take the ratio of the coressponding then windows is more safe than others

[4]:
```
plt.subplot(1, 2, 1)
df_nfraud['ProductCD'].value_counts().nlargest(6).plot(kind='bar',title="NOT␣
 ↪FRAUD ProductCD",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['ProductCD'].value_counts().nlargest(6).plot(kind='bar',title="FRAUD␣
 ↪ProductCD",figsize=(8,4))
```

[4]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d24ae90b00>
```



From the above above plots it is clear that product W is being purchased more abundantly than compared to other products and the product W is more safer to purchase (when we compare it by ratio)

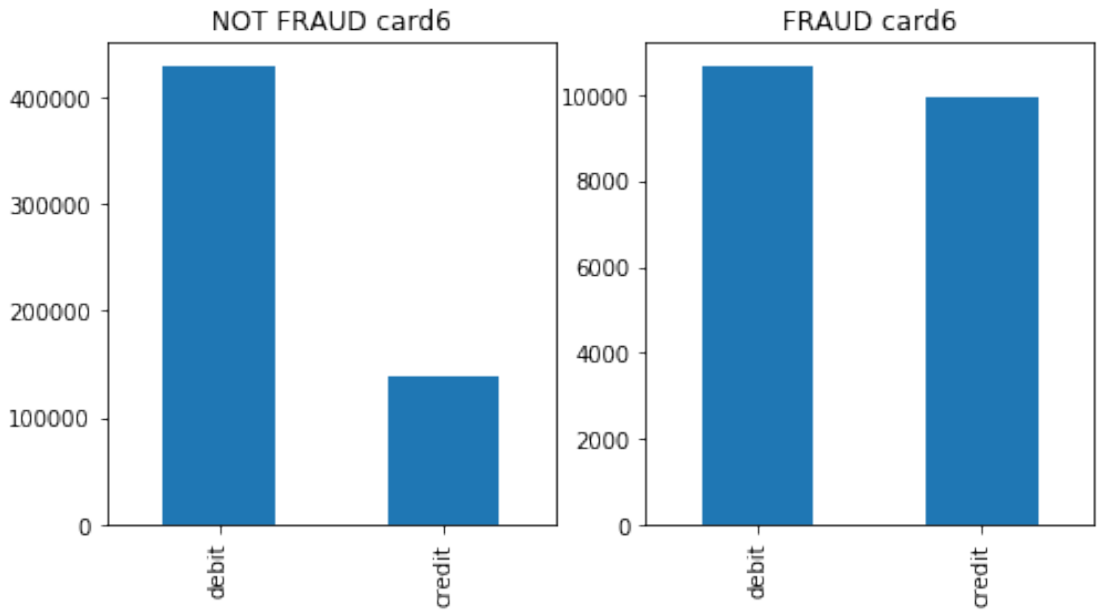
[5]:
```
plt.subplot(1, 2, 1)
df_nfraud['card4'].value_counts().nlargest(6).plot(kind='bar',title="NOT FRAUD␣
 ↪card4",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['card4'].value_counts().nlargest(6).plot(kind='bar',title="FRAUD␣
 ↪card4",figsize=(8,4))
```

[5]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d249f40f28>
```

From the above above plots it is clear that visa cards are used more compared to other cards and the visa cards are more safer to use(when we compare it by ratio)

[6]:
```
plt.subplot(1, 2, 1)
df_nfraud['card6'].value_counts().nlargest(2).plot(kind='bar',title="NOT FRAUD␣
 ↪card6",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['card6'].value_counts().nlargest(2).plot(kind='bar',title="FRAUD␣
 ↪card6",figsize=(8,4))
```

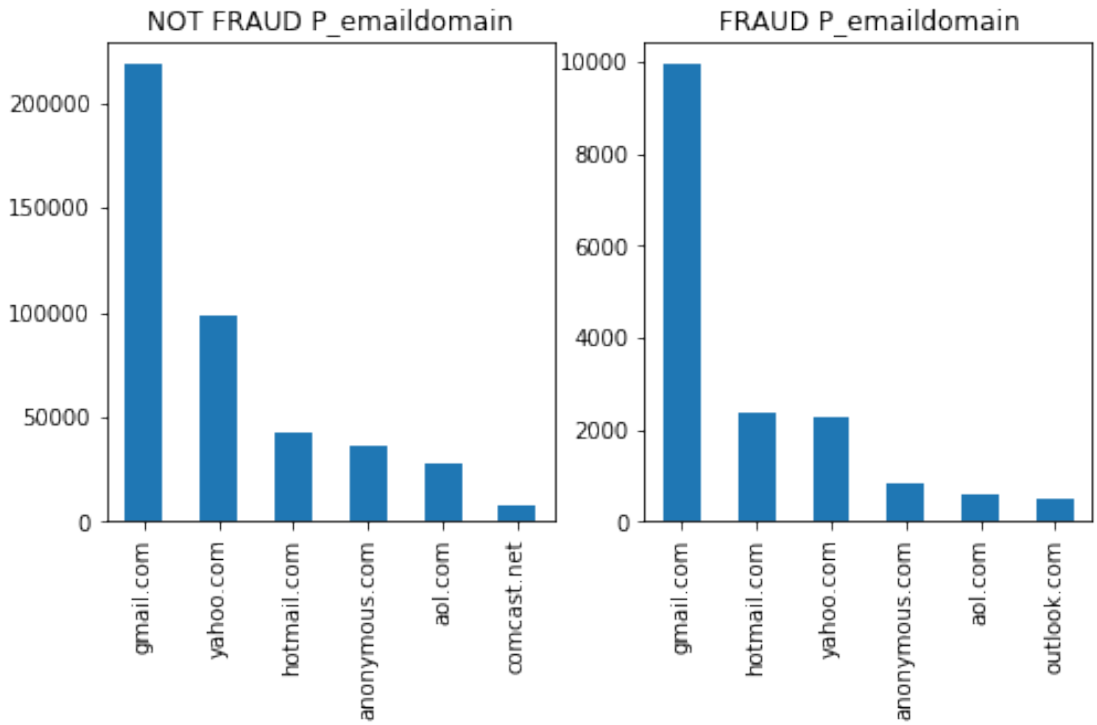
[6]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d24ac2df60>
```



From the above above plots it is clear that debit cards are used more compared to credit and the debit cards are more safer to use(when we compare it by ratio)

[7]:
```
plt.subplot(1, 2, 1)
df_nfraud['P_emaildomain'].value_counts().nlargest(6).
 ↪plot(kind='bar',title="NOT FRAUD P_emaildomain",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['P_emaildomain'].value_counts().nlargest(6).
 ↪plot(kind='bar',title="FRAUD P_emaildomain",figsize=(8,4))
```

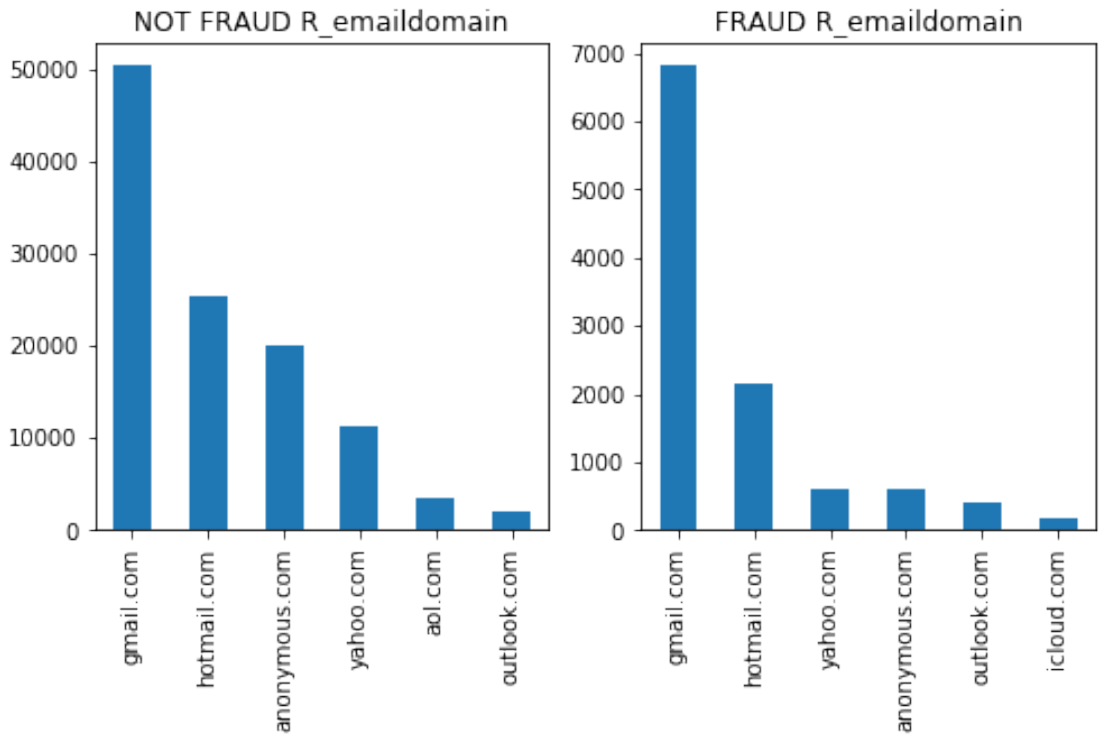
[7]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d24ade3ef0>
```



In the above above plots the users who use gmail.com are more compared to other mailing addresses and using gmail.com as mailing address is more safe compared to others(when we compare it by ratio)

[8]:
```
plt.subplot(1, 2, 1)
df_nfraud['R_emaildomain'].value_counts().nlargest(6).
 ↪plot(kind='bar',title="NOT FRAUD R_emaildomain",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['R_emaildomain'].value_counts().nlargest(6).
 ↪plot(kind='bar',title="FRAUD R_emaildomain",figsize=(8,4))
```

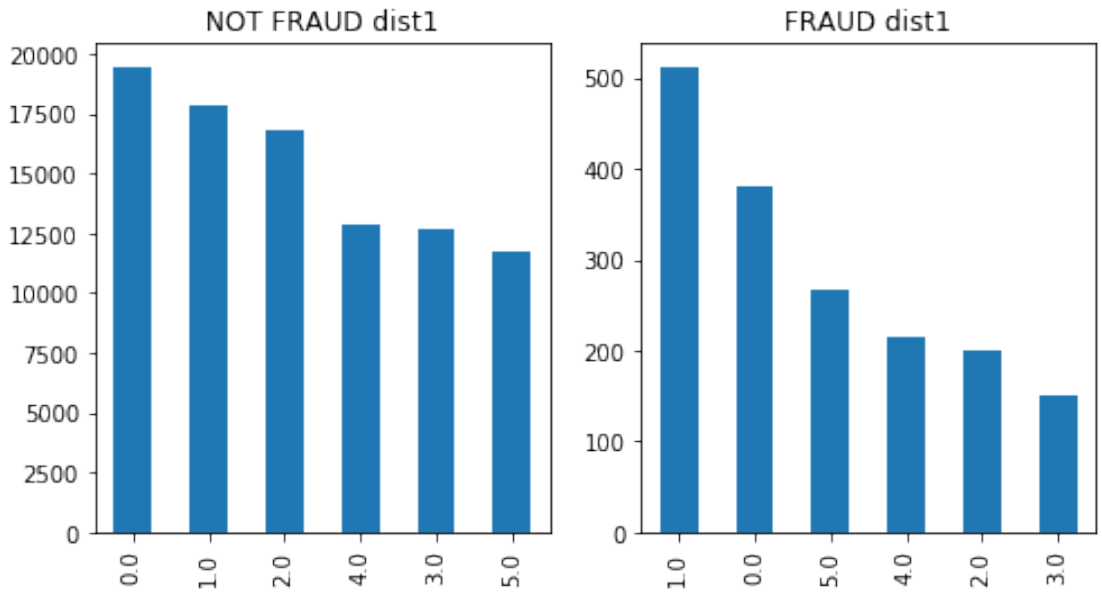
[8]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d24aa35400>
```



In the above above plots the users who use gmail.com are more compared to other mailing addresses and using gmail.com as a mailing address is more safe compared to others(when we compare it by ratio)

[9]:
```
plt.subplot(1, 2, 1)
df_nfraud['dist1'].value_counts().nlargest(6).plot(kind='bar',title="NOT FRAUD␣
 ↪dist1",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['dist1'].value_counts().nlargest(6).plot(kind='bar',title="FRAUD␣
 ↪dist1",figsize=(8,4))
```

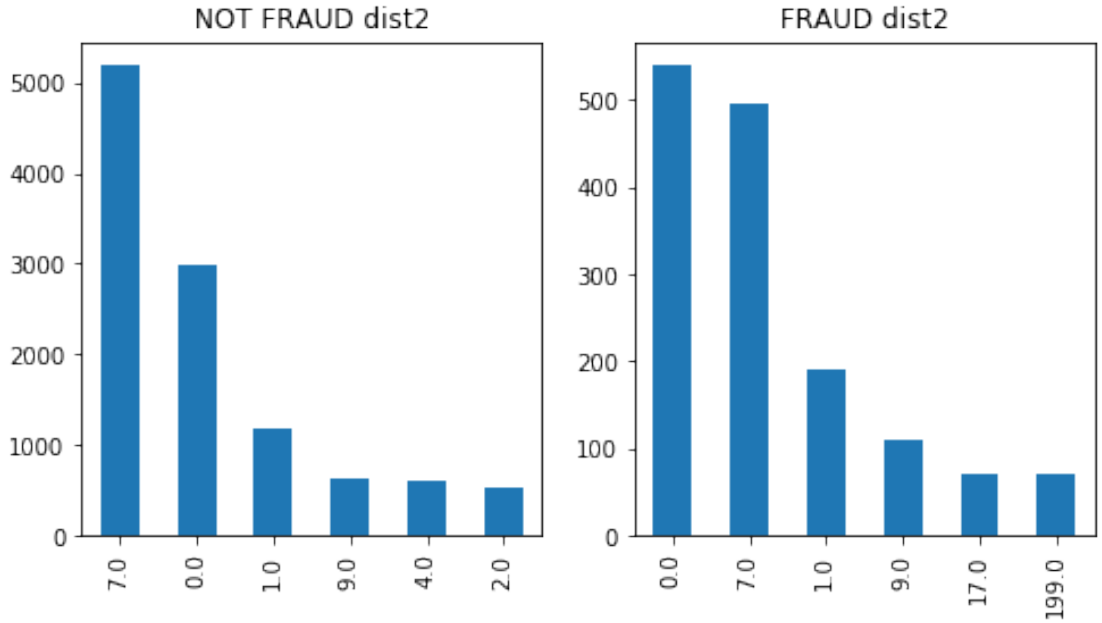
[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d24ad86898>
```



No interesting insights

[10]:
```
plt.subplot(1, 2, 1)
df_nfraud['dist2'].value_counts().nlargest(6).plot(kind='bar',title="NOT FRAUD␣
 ↪dist2",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['dist2'].value_counts().nlargest(6).plot(kind='bar',title="FRAUD␣
 ↪dist2",figsize=(8,4))
```

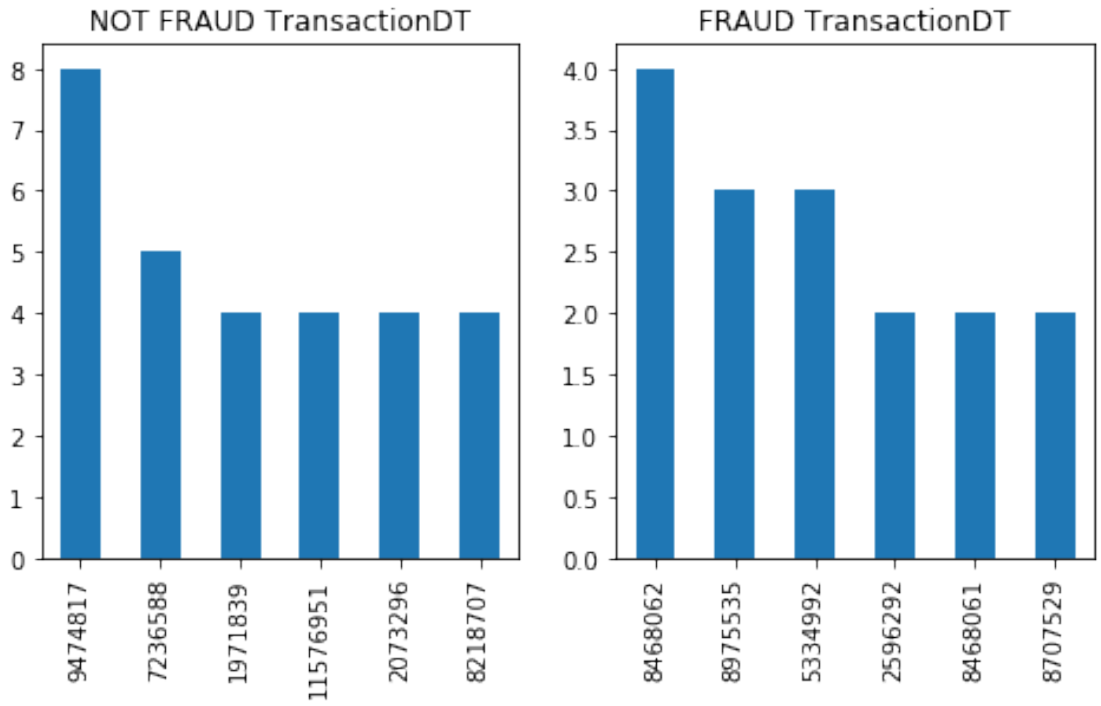
[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d249fd2860>
```



No interesting insights

[11]:
```
plt.subplot(1, 2, 1)
df_nfraud['TransactionDT'].value_counts().nlargest(6).
 ↪plot(kind='bar',title="NOT FRAUD TransactionDT",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['TransactionDT'].value_counts().nlargest(6).
 ↪plot(kind='bar',title="FRAUD TransactionDT",figsize=(8,4))
```

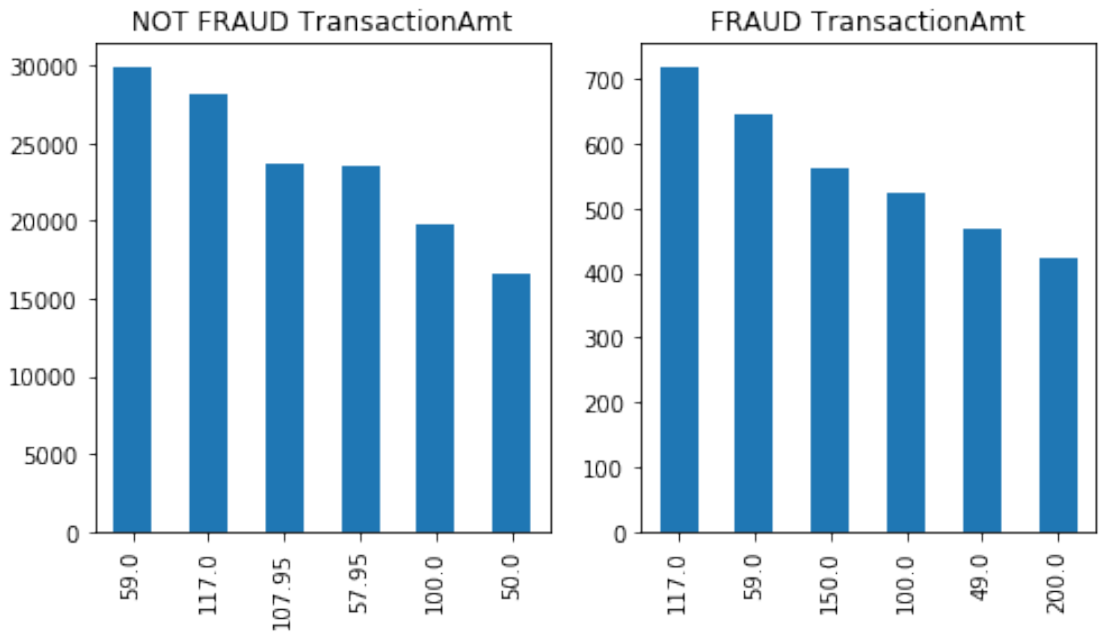
[11]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d249e91710>
```



No interesting insights

[12]:
```
plt.subplot(1, 2, 1)
df_nfraud['TransactionAmt'].value_counts().nlargest(6).
 ↪plot(kind='bar',title="NOT FRAUD TransactionAmt",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['TransactionAmt'].value_counts().nlargest(6).
 ↪plot(kind='bar',title="FRAUD TransactionAmt",figsize=(8,4))
```

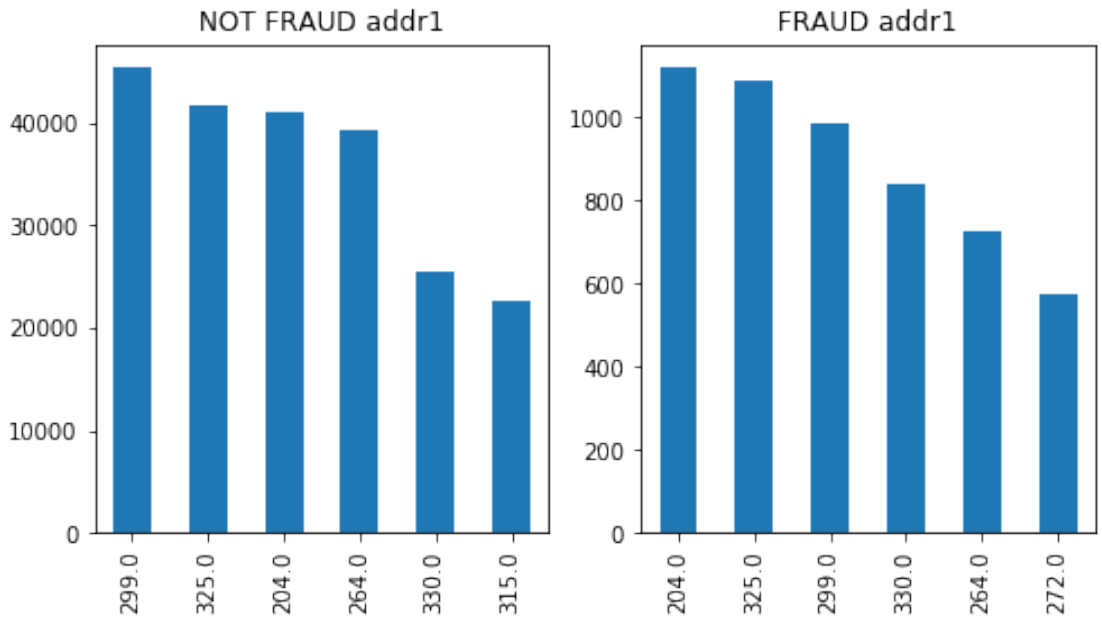
[12]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d249edf4a8>
```



In the above plot the product with transaction amt of 59.0 has more non fraud transactions and product with trans amt of 117.0 has more fraud transactions

[13]:
```
plt.subplot(1, 2, 1)
df_nfraud['addr1'].value_counts().nlargest(6).plot(kind='bar',title="NOT FRAUD␣
 ↪addr1",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['addr1'].value_counts().nlargest(6).plot(kind='bar',title="FRAUD␣
 ↪addr1",figsize=(8,4))
```

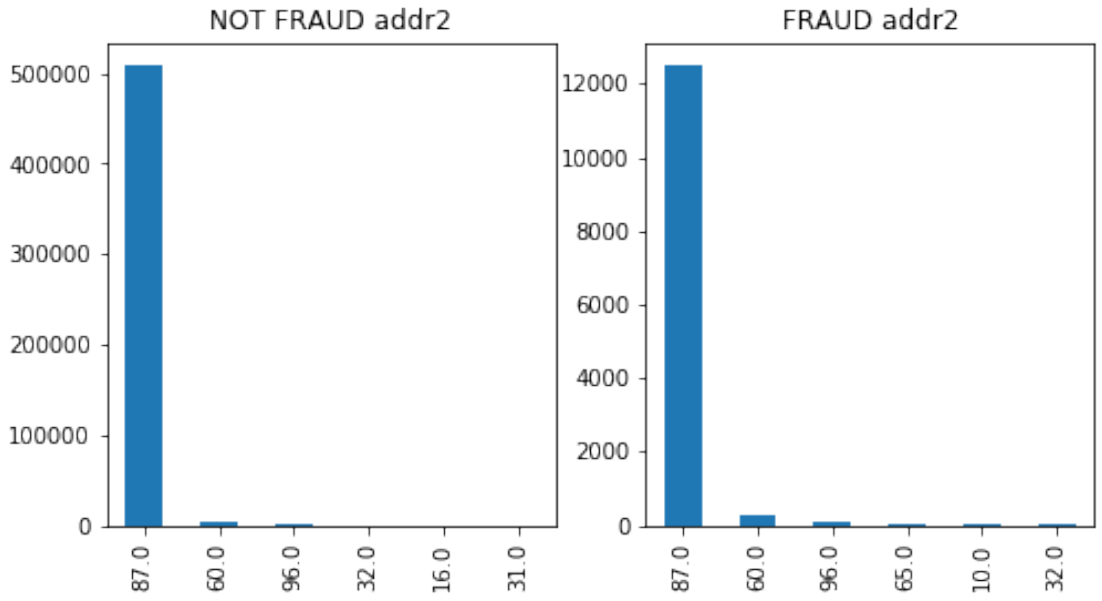
[13]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d2499588d0>
```



In the above plot the product with address or zip at of 299.0 has more non fraud transactions and product with address or zip with 204.0 has more fraud transactions.

[14]:
```
plt.subplot(1, 2, 1)
df_nfraud['addr2'].value_counts().nlargest(6).plot(kind='bar',title="NOT FRAUD␣
↪addr2",figsize=(8,4))
plt.subplot(1, 2, 2)
df_fraud['addr2'].value_counts().nlargest(6).plot(kind='bar',title="FRAUD␣
↪addr2",figsize=(8,4))
```

[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d249a36eb8>
```



In the above plot it is clear that the transactions are being mainly proceeded in the country with country code 87.0 remaining all has a very less number of transactions

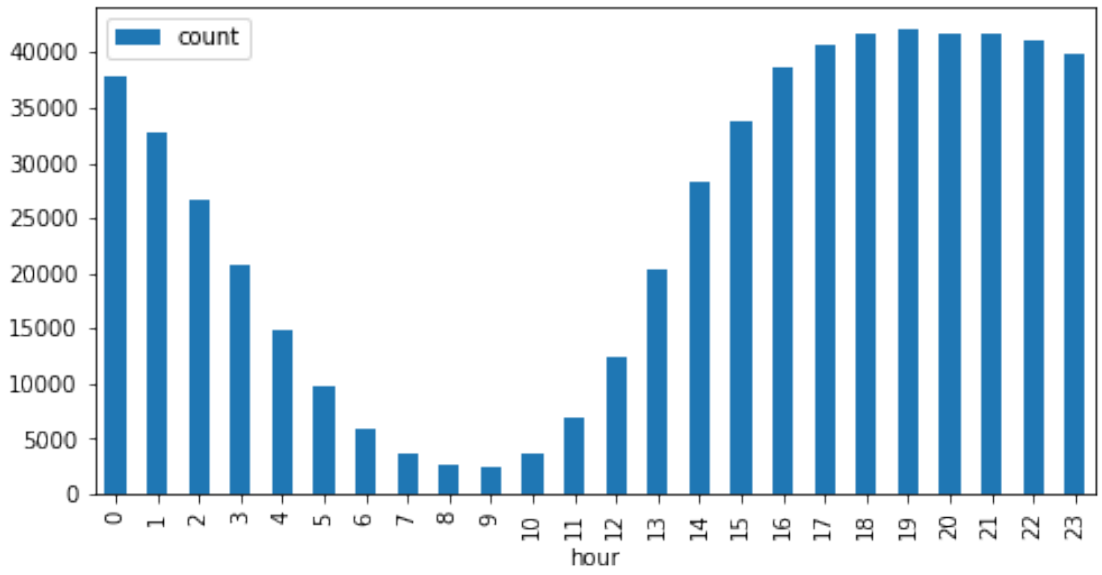
2.1 Part 2 - Transaction Frequency

[15]:
```
k=df["addr2"].value_counts().nlargest(n=1).values[0]
df_addr2=df_fraud.loc[df_fraud['addr2']== k]
df_sol1=pd.DataFrame()
df_sol1['TransactionDT']=(df['TransactionDT']//3600)%24
df1=df_sol1.groupby('TransactionDT')#.to_frame().value_counts()
l1=[]
l2=[]
for name,grp in df1:
    l1.append(name)
    l2.append(grp.shape[0])
#print(df1)
print(l1)
print(l2)

data = {"hour":l1,"count":l2}
df_fin=pd.DataFrame(data)
print(df_fin)
df_fin.plot(kind='bar',x="hour",y="count",figsize=(8,4))
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,
22, 23]
[37795, 32797, 26732, 20802, 14839, 9701, 6007, 3704, 2591, 2479, 3627, 6827,
12451, 20315, 28328, 33859, 38698, 40723, 41639, 42115, 41782, 41641, 41139,
39949]
    hour  count
0      0  37795
1      1  32797
2      2  26732
3      3  20802
4      4  14839
5      5   9701
6      6   6007
7      7   3704
8      8   2591
9      9   2479
10    10   3627
11    11   6827
12    12  12451
13    13  20315
14    14  28328
```

```
15     15  33859
16     16  38698
17     17  40723
18     18  41639
19     19  42115
20     20  41782
21     21  41641
22     22  41139
23     23  39949
```

[15]: `<matplotlib.axes._subplots.AxesSubplot at 0x1d249b174a8>`



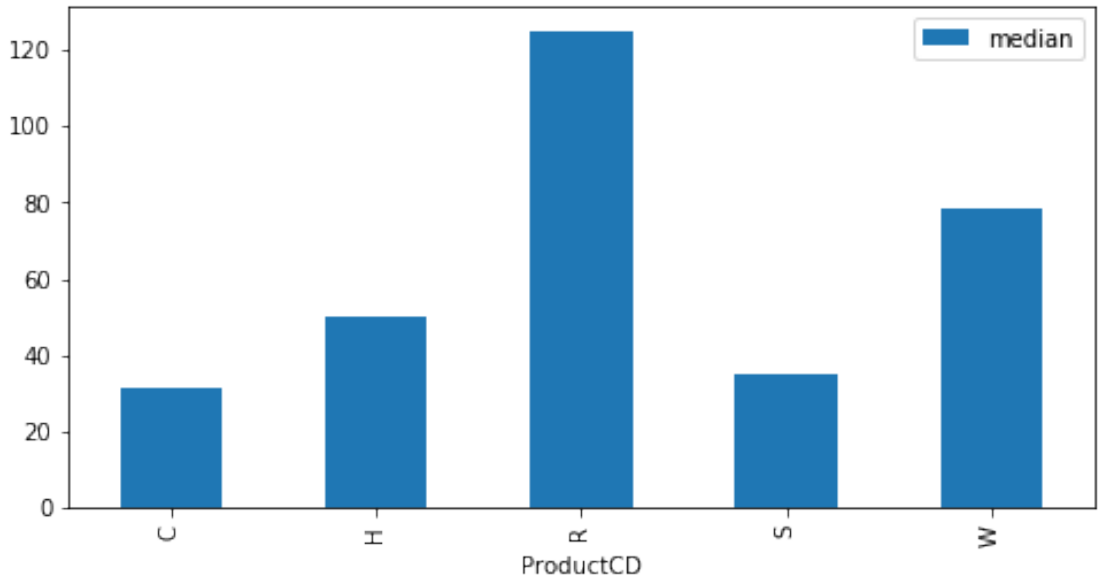
In the above graph the lowest transactions occurs between 9-10 hrs and in the time between 7-10 hrs customers are less active and it increases from then on and the maximum transactions takes place between 19-20 hrs and in the time between 16-24hrs customers are being more active.

2.2 Part 3 - Product Code

[16]:
```
df_sol2=pd.DataFrame()
df_sol2['ProductCD']=df['ProductCD']
df_sol2['TransactionAmt']=df['TransactionAmt']
df_p1=pd.DataFrame()
#df_3=df.ProductCD.unique()
l3=[]
l4=[]
df2 = df_sol2.groupby('ProductCD')
for n,g in df2:
    l3.append(n)
```

```
    l4.append(np.median(g['TransactionAmt']))
data1={"ProductCD":l3,"median":l4}
df_fin1=pd.DataFrame(data1)
df_fin1.plot(kind='bar',x="ProductCD",y="median",figsize=(8,4))
```

[16]: <matplotlib.axes._subplots.AxesSubplot at 0x1d249bd1780>



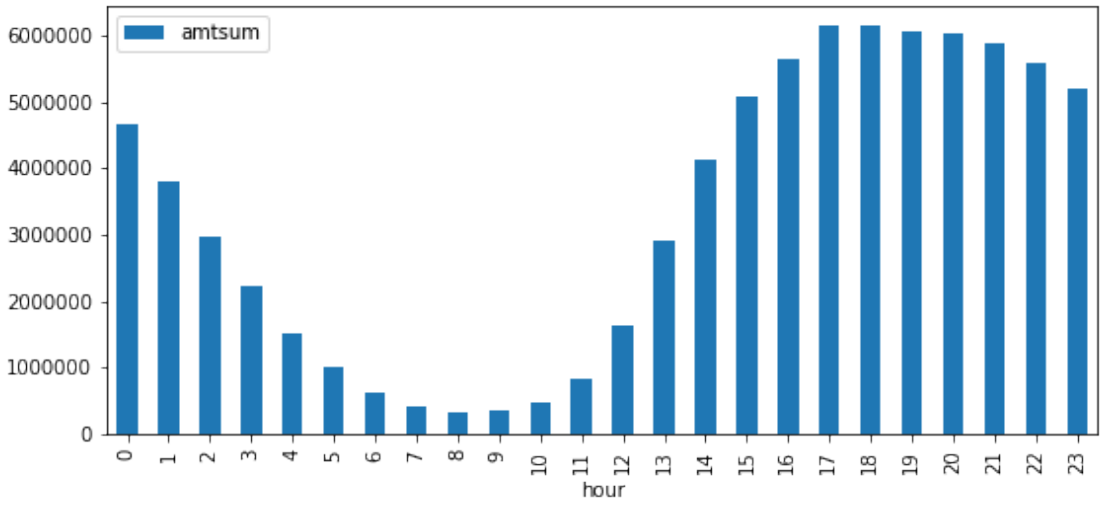
From the above it is clear R is the more expensive product whereas the C is the cheaper product

2.3 Part 4 - Correlation Coefficient

[17]:
```
df_sol3=pd.DataFrame()
df_sol3['TransactionDT']=(df['TransactionDT']//3600)%24
df_sol3['TransactionAmt']=df['TransactionAmt']
df3 = df_sol3.groupby('TransactionDT')
l5=[]
l6=[]
for n,grp in df3:
    l5.append(n)
    l6.append(np.sum(grp['TransactionAmt']))
data2= {"hour":l5,"amtsum":l6}
df_fin2=pd.DataFrame(data2)
df_fin2.plot(kind='bar',x="hour",y="amtsum",figsize=(9,4))
correlation=df_fin2['hour'].corr(df_fin2['amtsum'])
```

```
print("****************************************The correlation is:
 ↪**********************************************************")
print("correlation=",correlation)
print("****************************************The correlation is:
 ↪**********************************************************")
```

```
****************************************The correlation
is:**********************************************************
correlation= 0.6421174943084417
****************************************The correlation
is:**********************************************************
```



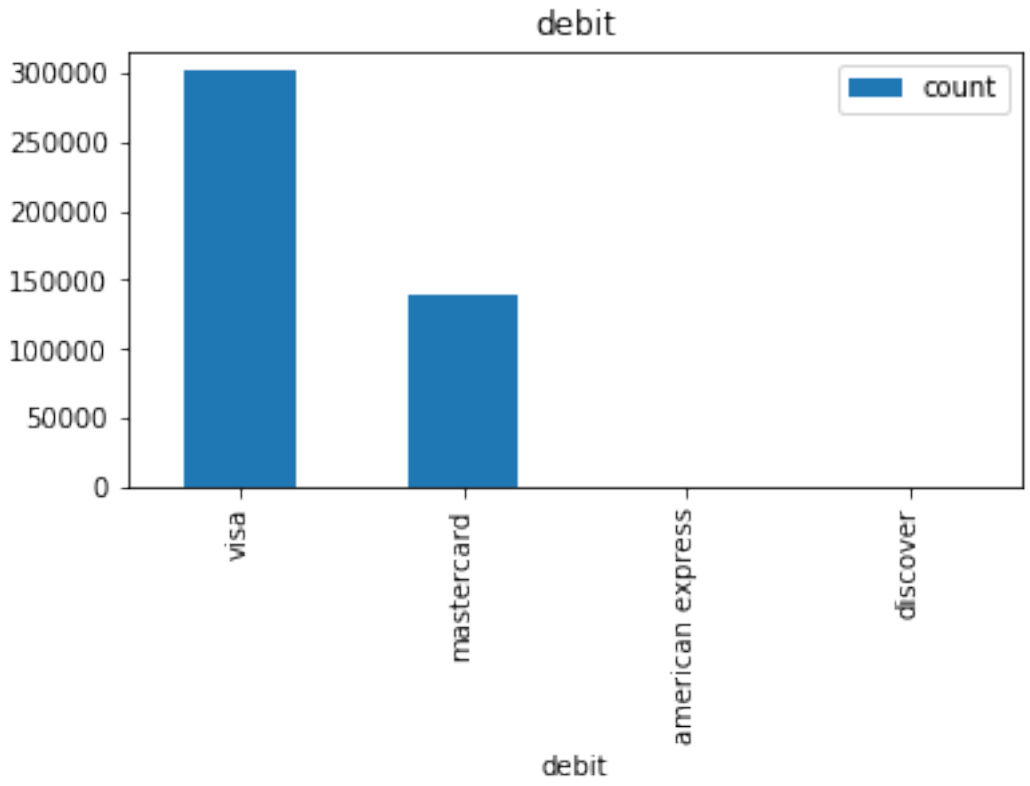
From the above the correlation coefficient between the hours and the sum of the transaction amounts at that time is correlation= 0.6421174943084417 and in between 7-11 hrs the amtsum is least and in 17-22 hrs the amtsum is the greatest

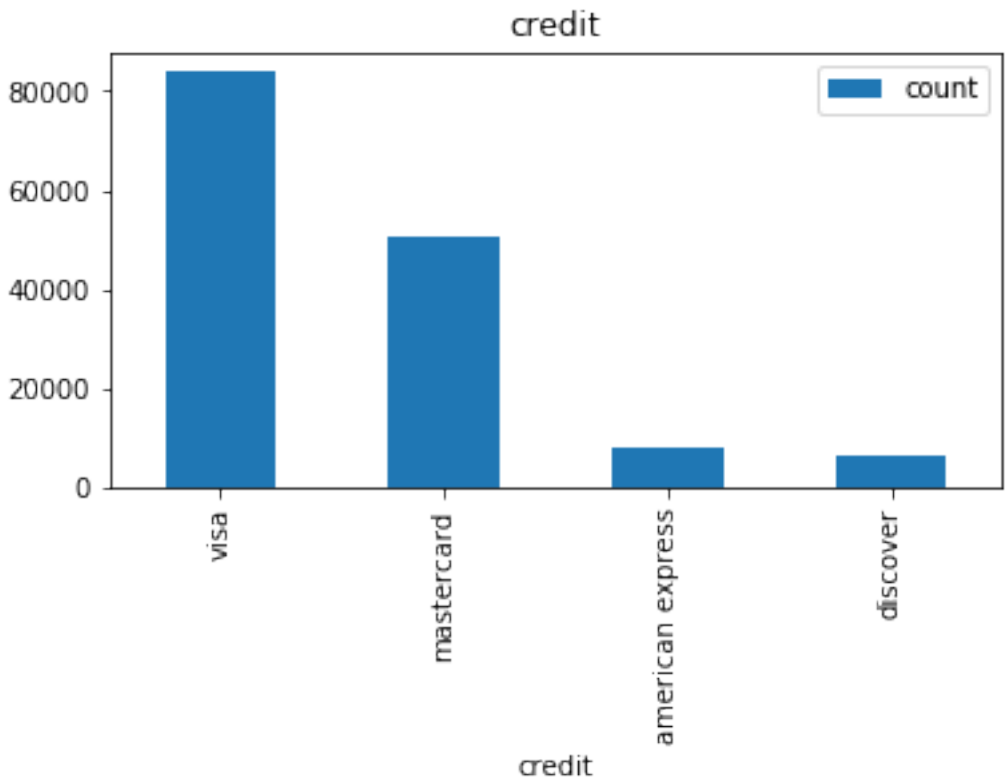
2.4 Part 5 - Interesting Plot

[18]:
```
df_sol4=pd.DataFrame()
df_sol4['card4']=df['card4']
df_sol4['card6']=df['card6']
debit_v=df_sol4.loc[(df_sol4['card4']== 'visa') & (df_sol4['card6']=='debit')]
credit_v=df_sol4.loc[(df_sol4['card4']== 'visa') & (df_sol4['card6']=='credit')]
debit_m=df_sol4.loc[(df_sol4['card4']== 'mastercard') &␣
 ↪(df_sol4['card6']=='debit')]
credit_m=df_sol4.loc[(df_sol4['card4']== 'mastercard') &␣
 ↪(df_sol4['card6']=='credit')]
debit_a=df_sol4.loc[(df_sol4['card4']== 'american express') &␣
 ↪(df_sol4['card6']=='debit')]
```

```
credit_a=df_sol4.loc[(df_sol4['card4']== 'american express') &␣
 ↪(df_sol4['card6']=='credit')]
debit_d=df_sol4.loc[(df_sol4['card4']== 'discover') &␣
 ↪(df_sol4['card6']=='debit')]
credit_d=df_sol4.loc[(df_sol4['card4']== 'discover') &␣
 ↪(df_sol4['card6']=='credit')]

data4={"debit":['visa','mastercard','american express','discover'],"count":
 ↪[debit_v.shape[0],debit_m.shape[0],debit_a.shape[0],debit_d.shape[0]]}
df_fin4=pd.DataFrame(data4)
df_fin4.plot(kind='bar',x="debit",y="count",title="debit",figsize=(6,3))

data5={"credit":['visa','mastercard','american express','discover'],"count":
 ↪[credit_v.shape[0],credit_m.shape[0],credit_a.shape[0],credit_d.shape[0]]}
df_fin5=pd.DataFrame(data5)
df_fin5.plot(kind='bar',x="credit",y="count",title="credit",figsize=(6,3))
```

[18]: `<matplotlib.axes._subplots.AxesSubplot at 0x1d249d036a0>`



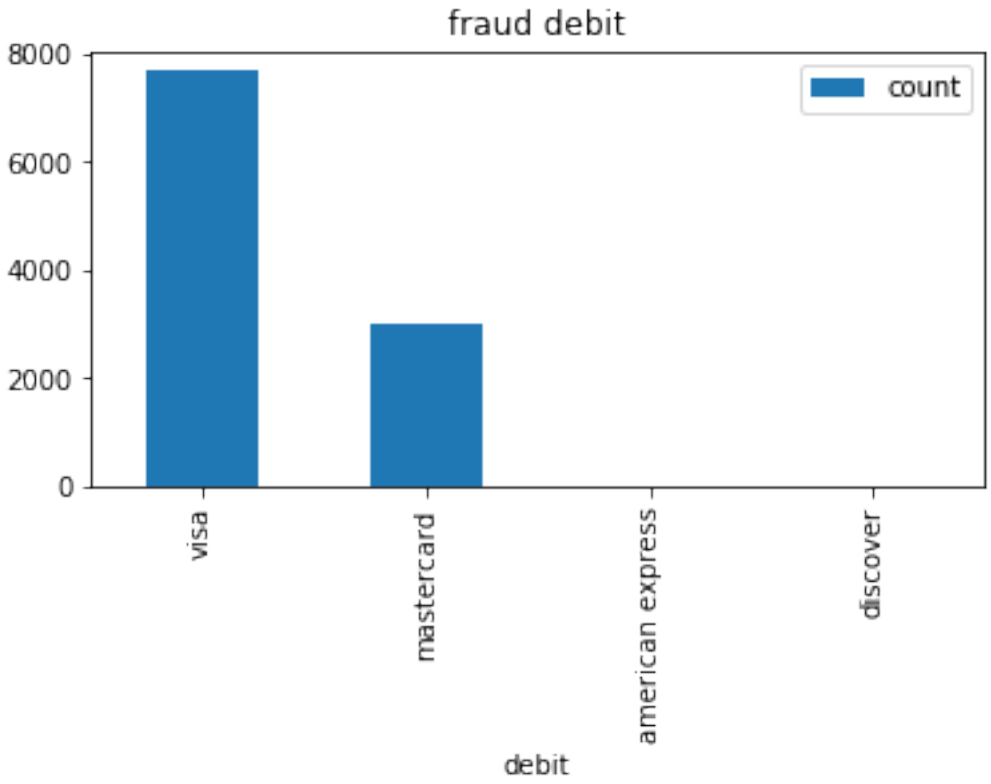


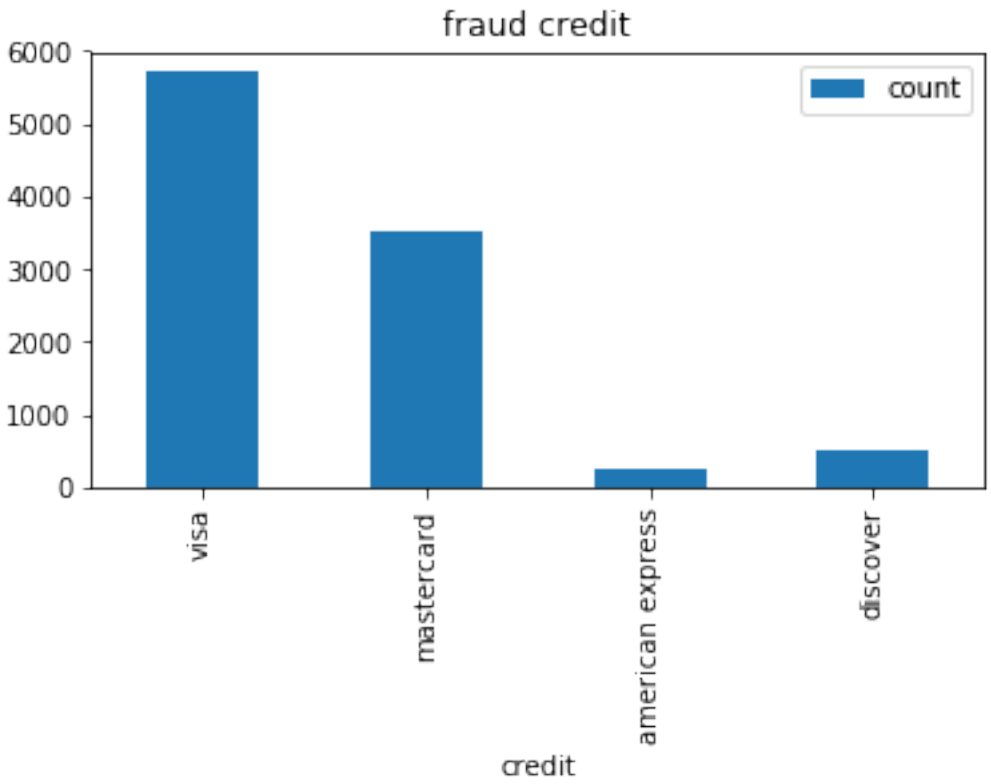
The above is the plot for the distribution of the cards in debit card and the credit card.For purchasing the products the customers are using more visa debit cards and credit cards.

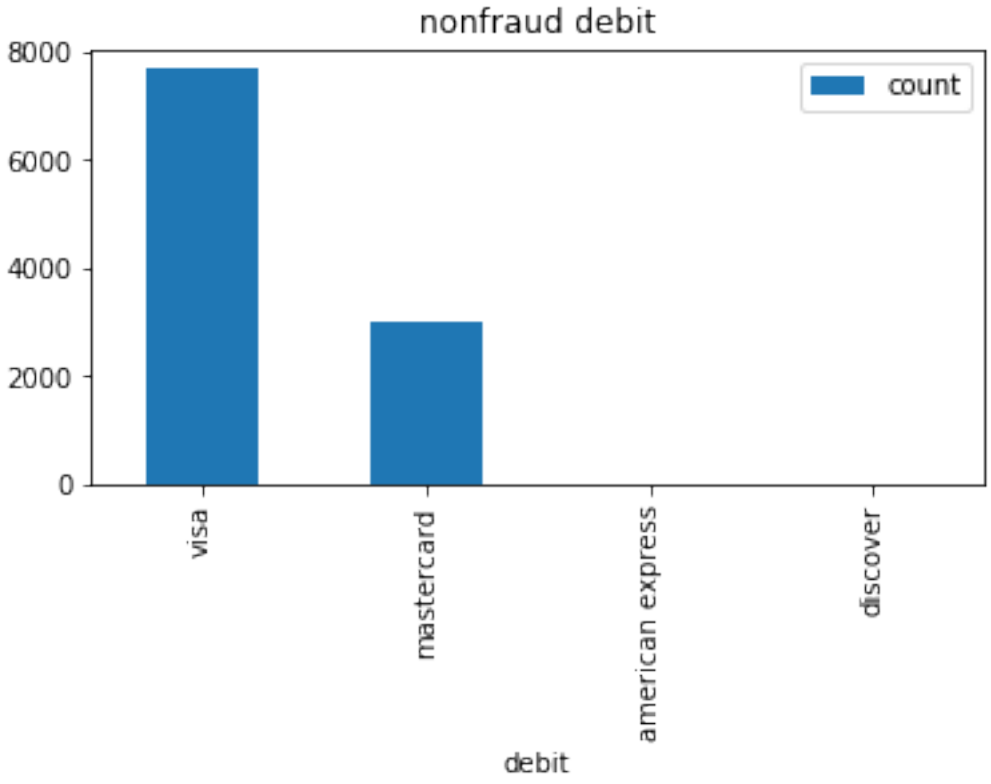
[19]:
```
df_fraud5=pd.DataFrame()
df_fraud5['card4']=df_fraud['card4']
df_fraud5['card6']=df_fraud['card6']
debit_vf=df_fraud5.loc[(df_fraud5['card4']== 'visa') &␣
 ↪(df_fraud5['card6']=='debit')]
credit_vf=df_fraud5.loc[(df_fraud5['card4']== 'visa') &␣
 ↪(df_fraud5['card6']=='credit')]
debit_mf=df_fraud5.loc[(df_fraud5['card4']== 'mastercard') &␣
 ↪(df_fraud5['card6']=='debit')]
credit_mf=df_fraud5.loc[(df_fraud5['card4']== 'mastercard') &␣
 ↪(df_fraud5['card6']=='credit')]
debit_af=df_fraud5.loc[(df_fraud5['card4']== 'american express') &␣
 ↪(df_fraud5['card6']=='debit')]
credit_af=df_fraud5.loc[(df_fraud5['card4']== 'american express') &␣
 ↪(df_fraud5['card6']=='credit')]
debit_df=df_fraud5.loc[(df_fraud5['card4']== 'discover') &␣
 ↪(df_fraud5['card6']=='debit')]
credit_df=df_fraud5.loc[(df_fraud5['card4']== 'discover') &␣
 ↪(df_fraud5['card6']=='credit')]
```

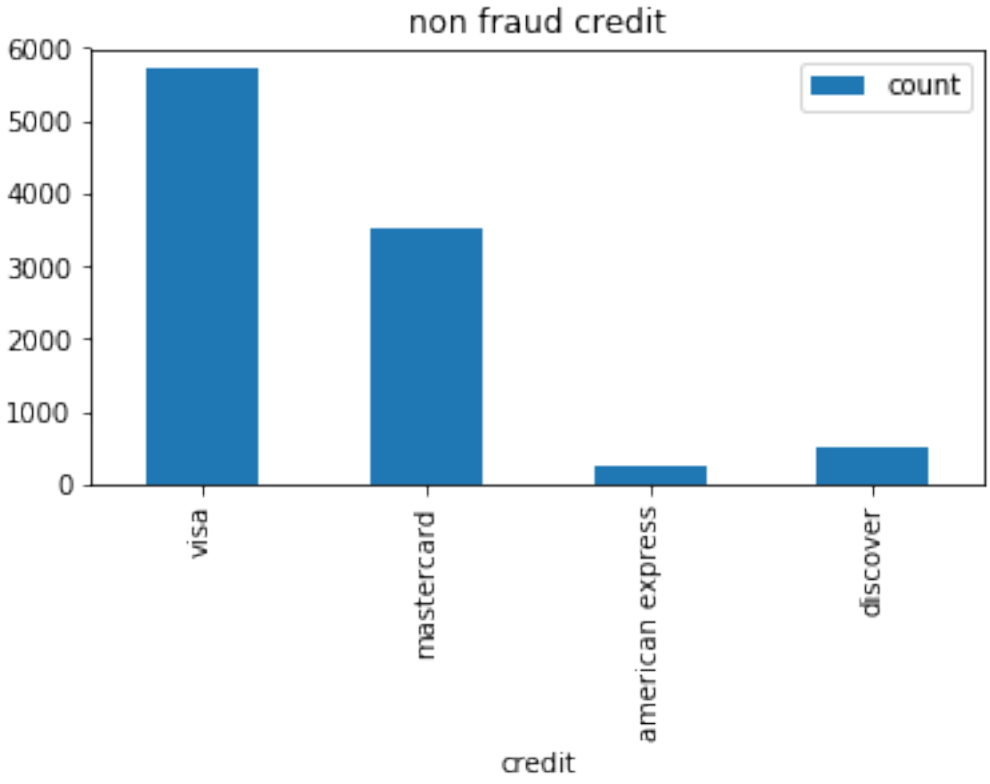
```
df_nfraud5=pd.DataFrame()
df_nfraud5['card4']=df_nfraud['card4']
df_nfraud5['card6']=df_nfraud['card6']
debit_vnf=df_nfraud5.loc[(df_nfraud5['card4']== 'visa') &␣
 ↪(df_nfraud5['card6']=='debit')]
credit_vnf=df_nfraud5.loc[(df_nfraud5['card4']== 'visa') &␣
 ↪(df_nfraud5['card6']=='credit')]
debit_mnf=df_nfraud5.loc[(df_nfraud5['card4']== 'mastercard') &␣
 ↪(df_nfraud5['card6']=='debit')]
credit_mnf=df_nfraud5.loc[(df_nfraud5['card4']== 'mastercard') &␣
 ↪(df_nfraud5['card6']=='credit')]
debit_anf=df_nfraud5.loc[(df_nfraud5['card4']== 'american express') &␣
 ↪(df_nfraud5['card6']=='debit')]
credit_anf=df_nfraud5.loc[(df_nfraud5['card4']== 'american express') &␣
 ↪(df_nfraud5['card6']=='credit')]
debit_dnf=df_nfraud5.loc[(df_nfraud5['card4']== 'discover') &␣
 ↪(df_nfraud5['card6']=='debit')]
credit_dnf=df_nfraud5.loc[(df_nfraud5['card4']== 'discover') &␣
 ↪(df_nfraud5['card6']=='credit')]

data6={"debit":['visa','mastercard','american express','discover'],"count":
 ↪[debit_vf.shape[0],debit_mf.shape[0],debit_af.shape[0],debit_df.shape[0]]}
df_fin6=pd.DataFrame(data6)
df_fin6.plot(kind='bar',x="debit",y="count",title='fraud debit',figsize=(6,3))
data7={"credit":['visa','mastercard','american express','discover'],"count":
 ↪[credit_vf.shape[0],credit_mf.shape[0],credit_af.shape[0],credit_df.
 ↪shape[0]]}
df_fin7=pd.DataFrame(data7)
df_fin7.plot(kind='bar',x="credit",y="count",title="fraud credit",figsize=(6,3))



data8={"debit":['visa','mastercard','american express','discover'],"count":
 ↪[debit_vnf.shape[0],debit_mnf.shape[0],debit_anf.shape[0],debit_dnf.
 ↪shape[0]]}
df_fin6=pd.DataFrame(data6)
df_fin6.plot(kind='bar',x="debit",y="count",title='nonfraud␣
 ↪debit',figsize=(6,3))
data9={"credit":['visa','mastercard','american express','discover'],"count":
 ↪[credit_vnf.shape[0],credit_mnf.shape[0],credit_anf.shape[0],credit_dnf.
 ↪shape[0]]}
df_fin7=pd.DataFrame(data7)
df_fin7.plot(kind='bar',x="credit",y="count",title="non fraud␣
 ↪credit",figsize=(6,3))
```

[19]: `<matplotlib.axes._subplots.AxesSubplot at 0x1d24c252588>`




fraud credit
6000
5000
4000
3000
2000
1000
0
count
visa
mastercard
american express
discover
credit


nonfraud debit
8000
6000
4000
2000
0
count
visa
mastercard
american express
discover
debit



The frequencey of the cards which are visa and all with (debit and fraud) is some what same as the visa and all, (debit and non fraud) so the probabilty is almost 1/2 for the fraud to takes place with any card.Visa is ued more in credit and debit cards in fraud and non fraud and the corresponding ratio also the highest.Relatively Visa cards are safer to use compared to others

2.5 Part 6 - Prediction Model

[20]:
```
df['dist1']=df['dist1'].fillna(df.loc[:,"dist1"].median())
df['dist2']=df['dist2'].fillna(df.loc[:,"dist2"].median())
df['R_emaildomain']=df['R_emaildomain'].fillna('mac.com',inplace=True)
df['P_emaildomain']=df['P_emaildomain'].fillna('mac.com',inplace=True)
df=df.fillna(method ='pad')
df=df.fillna(method ='bfill')
new_df=df.fillna(method ='ffill')
new1=new_df.fillna(-999)



df_test['dist1']=df_test['dist1'].fillna(df.loc[:,"dist1"].median())
df_test['dist2']=df_test['dist2'].fillna(df.loc[:,"dist2"].median())
```

```
df_test['R_emaildomain']=df_test['R_emaildomain'].fillna('mac.com',inplace=True)
df_test['P_emaildomain']=df_test['P_emaildomain'].fillna('mac.com',inplace=True)
df_test=df_test.fillna(method ='pad')
df_test=df_test.fillna(method ='bfill')
new_df2=df_test.fillna(method ='ffill')
new2=new_df2.fillna(-999)
result=new1['isFraud']
new1=new1.drop('isFraud',axis=1)
```

[21]:
```
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.metrics import roc_auc_score
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from xgboost import XGBClassifier
import xgboost as xgb


k=new1.select_dtypes(include='object').columns
le = LabelEncoder()
for i in k:
    new1[i]= le.fit_transform(new1[i].astype(str))
    new2[i]= le.fit_transform(new2[i].astype(str))



Xtrain, Xtest, ytrain, ytest = train_test_split(new1, result, test_size=0.43,␣
 ↪random_state=42)
log_reg = LogisticRegression()
log_reg.fit(Xtrain, ytrain)
prediction = log_reg.predict(Xtest)
print("accuracy: ",accuracy_score(ytest, prediction))
model = LogisticRegression(C=0.001, tol = 0.1,fit_intercept= True,␣
 ↪random_state=10,solver= 'liblinear')
model.fit(Xtrain,ytrain)
predictors=list(Xtrain)
pred=model.predict(Xtest)
```

```
C:\Users\Mohith\Anaconda3\lib\site-
packages\sklearn\linear_model\logistic.py:433: FutureWarning: Default solver
will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)

accuracy:  0.9646835976418977
```

[22]:
```
pred=model.predict(new2)
submission_test=pd.read_csv('sample_submission.csv',index_col='TransactionID')
submission_test['isFraud']=log_reg.predict_proba(new2)[:,1]
submission_test.to_csv('regression_log.csv')
```

2.6 Part 7 - Final Result

Kaggle Link: https://www.kaggle.com/mohith9626 Highest Rank: 5666 Score: 0.6690 Number of entries: 5