Homework 2 - IEEE Fraud Detection

```
# It is defined by the kaggle/python docker image: https://github
# For example, here's several helpful packages to load in

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Ent

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# Any results you write to the current directory are saved as outp
```

```
/kaggle/input/ieee-fraud-detection/train_identity.csv
/kaggle/input/ieee-fraud-detection/test_identity.csv
/kaggle/input/ieee-fraud-detection/test_transaction.csv
/kaggle/input/ieee-fraud-detection/sample_submission.csv
/kaggle/input/ieee-fraud-detection/train_transaction.csv
```

For all parts below, answer all parts as shown in the Google document for H
answer the questions. We also ask that code be commented to make it easi

audulent Transactior

Saved successfully!

```
# TODO: code and runtime results

import pandas as pd
```

```
train_transaction = pd.read_csv('/kaggle/input/ieee-fraud-detecti
train_transaction.head()
```

	TransactionID	isFraud	TransactionDT	TransactionAm
0	2987000	0	86400	68.5
1	2987001	0	86401	29.(
2	2987002	0	86469	59.(
3	2987003	0	86499	50.(
4	2987004	0	86506	50.(

5 rows × 394 columns

```
isFraud = train_transaction.loc[train_transaction['isFraud']==1]
isNotFraud = train_transaction.loc[train_transaction['isFraud']==
```

```
isFraud.head()
```

Saved successfully!

```
TransactionID  isFraud  TransactionDT  Transaction
```

```
cols = [col for col in isFraud.columns if col in ['TransactionID',
isFraud1 = isFraud[cols]
isFraud1.head()
```

	TransactionID	isFraud	TransactionDT	Transaction
203	2987203	1	89760	445.
240	2987240	1	90193	37.
243	2987243	1	90246	37.
245	2987245	1	90295	37.
288	2987288	1	90986	155.

```
df1 = pd.read_csv('/kaggle/input/ieee-fraud-detection/train_identi
df1.head()
```

Saved successfully!

	TransactionID	id_01	id_02	id_03	id_04	id_05
0	2987004	0.0	70787.0	NaN	NaN	NaN
1	2987008	-5.0	98945.0	NaN	NaN	0.0
2	2987010	-5.0	191631.0	0.0	0.0	0.0
3	2987011	-5.0	221832.0	NaN	NaN	0.0
4	2987016	0.0	7460.0	0.0	0.0	1.0

5 rows × 41 columns

```
cols = [col for col in df1.columns if col in ['DeviceType','Devic
df2 = df1[cols]
df2.head()
```

Saved successfully!

DeviceType DeviceInfo

```
isFraud = pd.concat([df2,isFraud1], sort='False')
isFraud.tail()
```

	DeviceInfo	DeviceType	P_emaildomain	ProductC
590361	NaN	NaN	yahoo.com	W
590364	NaN	NaN	hotmail.com	C
590368	NaN	NaN	hotmail.com	H
590372	NaN	NaN	yahoo.com	W
590526	NaN	NaN	gmail.com	F

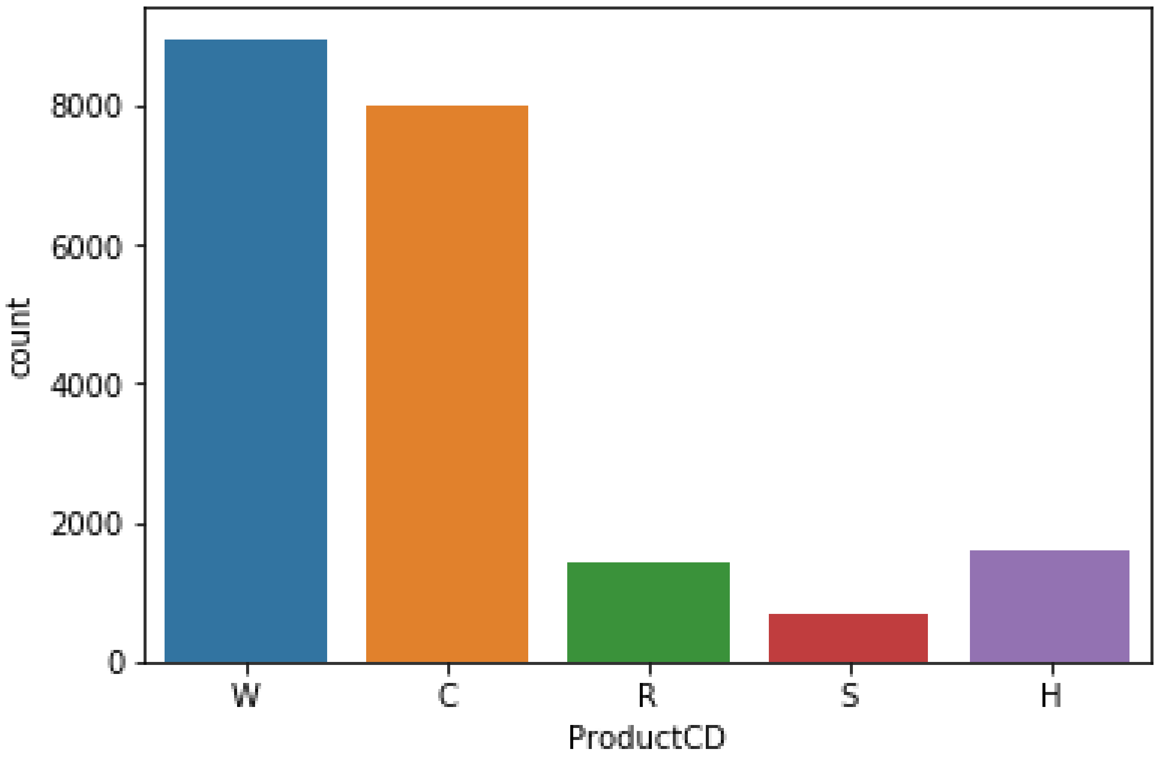
```
import numpy as np
np.log(isFraud['TransactionAmt']).hist(bins=100)
```

Saved successfully!

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f20cf431668
```

```
import seaborn as sns
sns.countplot(x='ProductCD', data= isFraud)
```

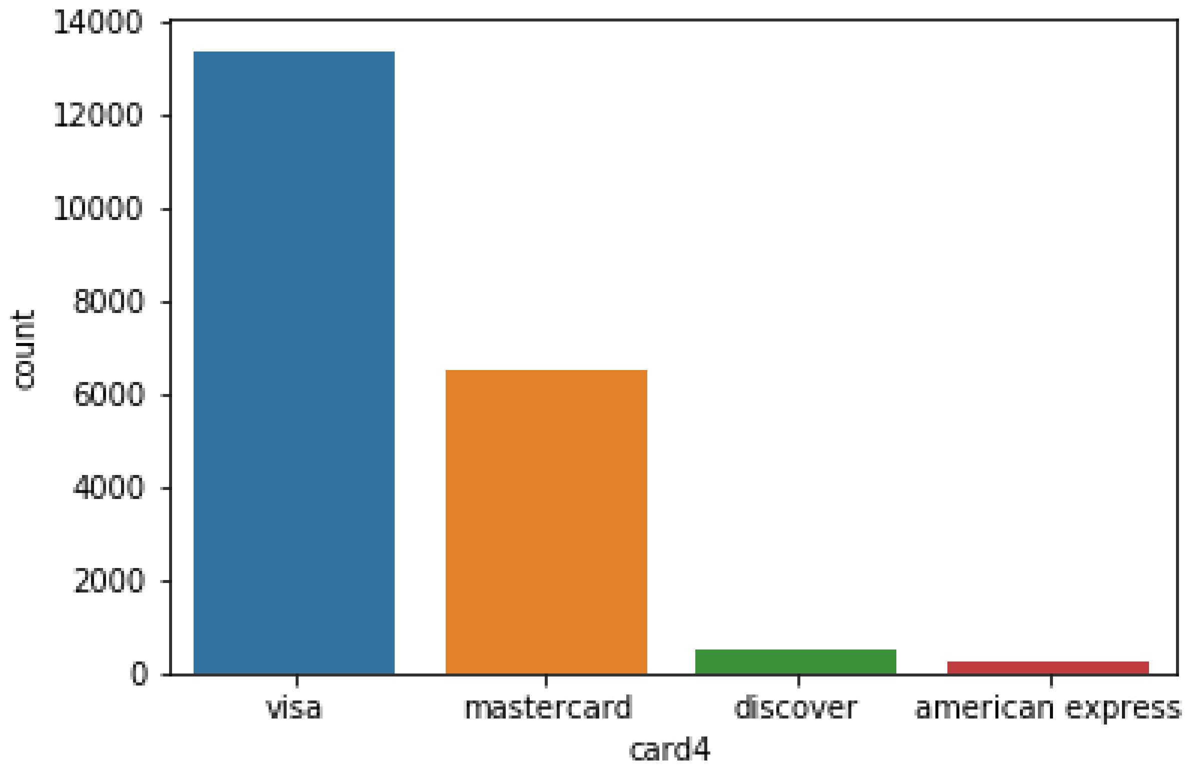
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f20cd0855f8
```



```
sns.countplot(x='card4', data= isFraud)
```

Saved successfully!

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f20cd042eb8
```

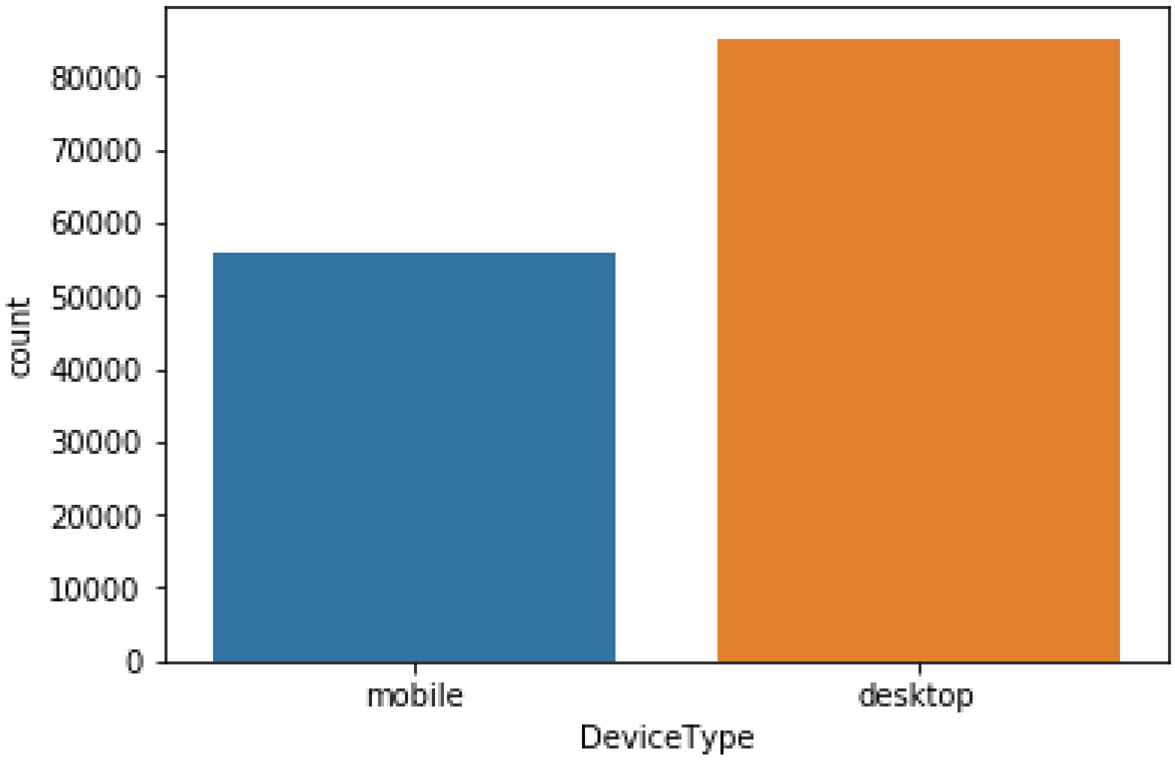


Visa cards have the highest instances of Fraud. But, Visa is also the most us

```
sns.countplot(x='DeviceType', data= isFraud)
```

Saved successfully!

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f20ccfc8eb8
```



Write your answer here

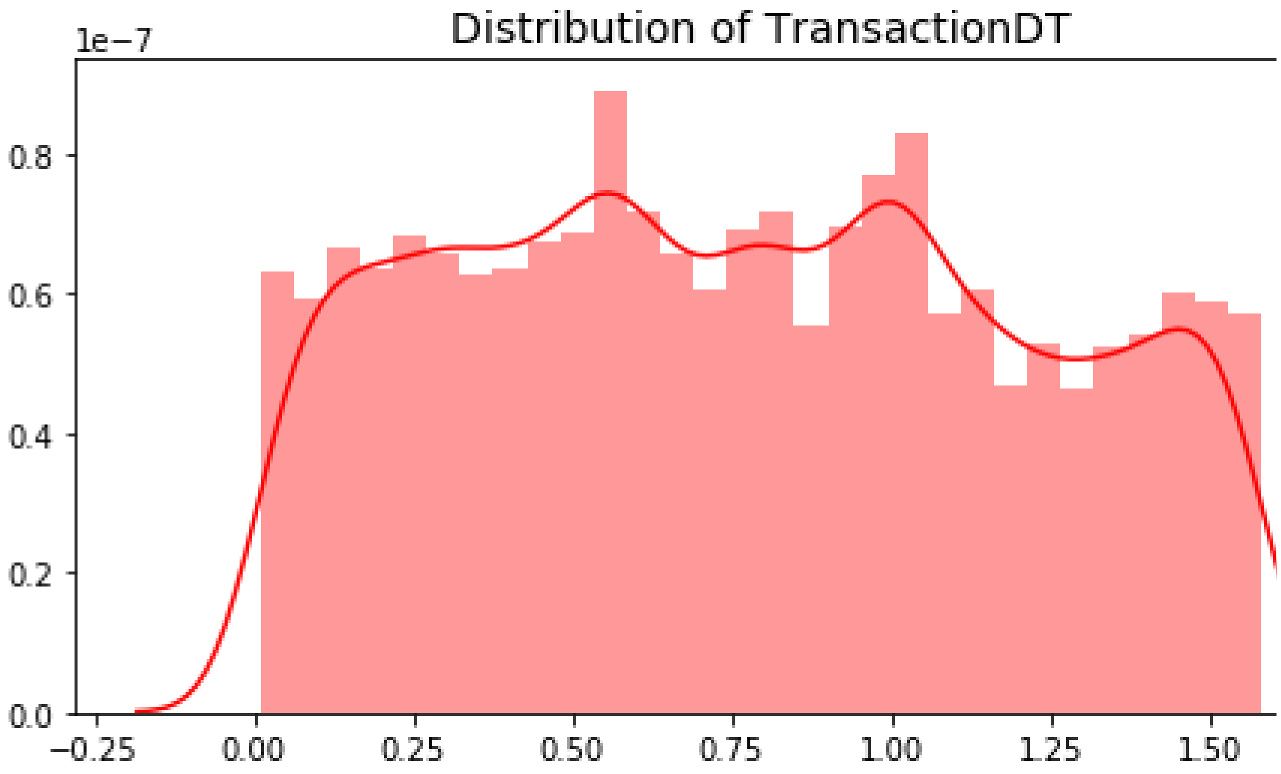
Part 2 - Transaction Frequency

```
# TODO: code to generate the frequency graph
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 2, figsize=(18,4))

#isFraud['TransactionDT'] = pd.to_numeric(isFraud['TransactionDT']
isFraud2 =isFraud
isFraud2 = isFraud2.dropna(subset=['TransactionDT'])
```

Saved successfully!

```
alues
sns.distplot(time_val, ax=ax[0], color='r')
ax[0].set_title('Distribution of TransactionDT', fontsize=14)
ax[1].set_xlim([min(time_val), max(time_val)])
```

```
sns.distplot(np.log(time_val), ax=ax[1], color='b')
ax[1].set_title('Distribution of LOG TransactionDT', fontsize=14)
ax[1].set_xlim([min(np.log(time_val)), max(np.log(time_val))])

plt.show()
```



Write your answer here

```
isFraud['Time'] = np.round(isFraud['TransactionDT']/(60*60),0)
isFraud['Time'] = np.round(isFraud['Time']%24,0)
isFraud.loc[isFraud['Time']==10]
l3 = []
l3 = isFraud['Time'].tolist()
plt.bar(isFraud['Time'], isFraud['addr2'], align='center', alpha=
```

Saved successfully!

```
<BarContainer object of 164896 artists>Error in callback
---------------------------------------------------------
KeyboardInterrupt                         Traceback (mos
/opt/conda/lib/python3.6/site-packages/ipykernel/pylab/b
    115         # ignore the tracking, just draw and clo
    116         try:
--> 117             return show(True)
    118         except Exception as e:
    119             # safely show traceback if in IPytho
```

23 frames

```
</opt/conda/lib/python3.6/site-packages/decorator.py:dec

/opt/conda/lib/python3.6/site-packages/matplotlib/transf
   1971         and :meth:`scale`.
   1972         """
-> 1973         a = np.cos(theta)
   1974         b = np.sin(theta)
   1975         rotate_mtx = np.array([[a, -b, 0.0], [b,

KeyboardInterrupt:
```

SEARCH STACK OVERFLOW

```
sns.countplot(x='DaysFromStart', data= isFraud)
isFraud['DaysFromStart'].unique()
```

Saved successfully!

```
-------------------------------------------------------
ValueError                                Traceback (mos
<ipython-input-17-0287a2d1dae7> in <module>
----> 1 sns.countplot(x='DaysFromStart', data= isFraud)
      2 isFraud['DaysFromStart'].unique()

 ────────────────────────────── 2 frames ──────────────
/opt/conda/lib/python3.6/site-packages/seaborn/categoric
    153                     if isinstance(input, string_type
    154                         err = "Could not interpret i
--> 155                         raise ValueError(err)
    156 
    157                 # Figure out the plotting orientatio

ValueError: Could not interpret input 'DaysFromStart'
```

SEARCH STACK OVERFLOW

Part 3 - Product Code

```
# TODO: code to analyze prices for different product codes
isFraud1 = isFraud
isFraud1['Rank'] = isFraud1['TransactionAmt'].rank(ascending=0)
#isFraud1.sort_values('TransactionAmt', inplace=True)
isFraud1["group_rank"] = isFraud1.groupby("ProductCD")["Transacti
isFraud1.head()
```

Saved successfully!

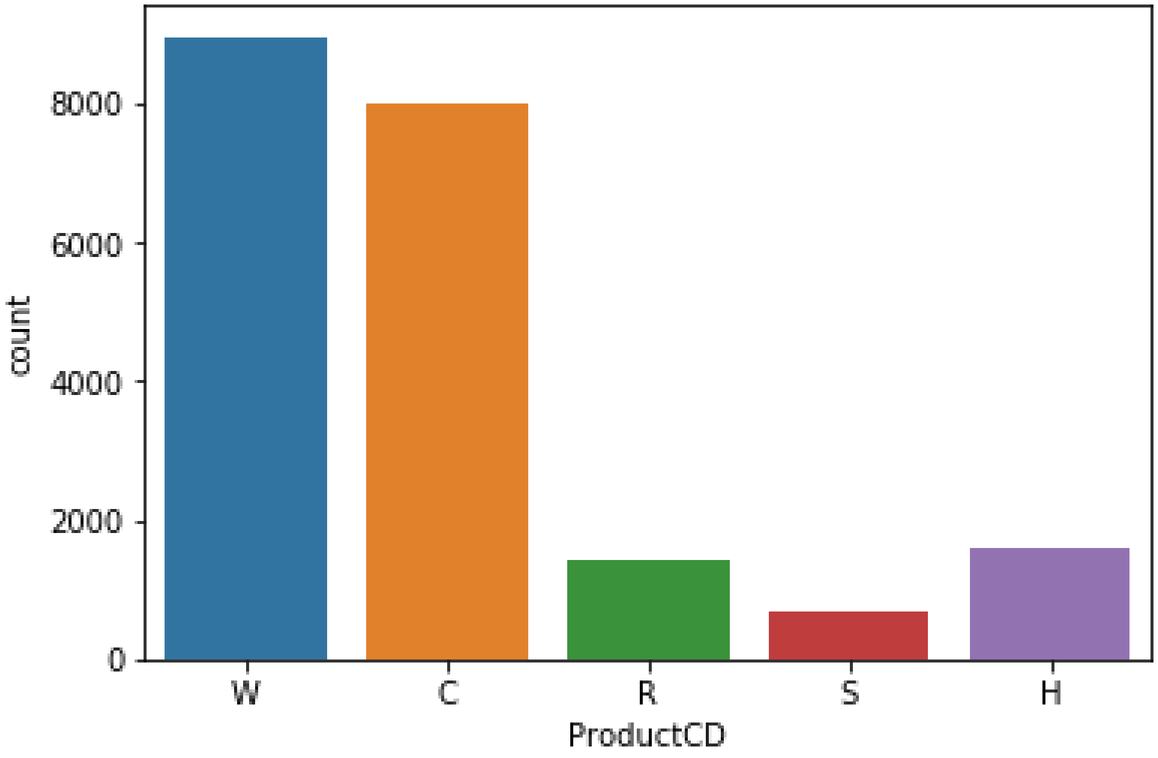
	DeviceInfo	DeviceType	P_emaildomain	ProductCD
0	SAMSUNG SM-G892A Build/NRD90M	mobile	NaN	NaN
1	iOS Device	mobile	NaN	NaN
2	Windows	desktop	NaN	NaN
3	NaN	desktop	NaN	NaN
4	MacOS	desktop	NaN	NaN

We see on ranking that the ProductCD which is W is most priced while that

```
sns.countplot(x='ProductCD', data = isFraud)
```

Saved successfully!

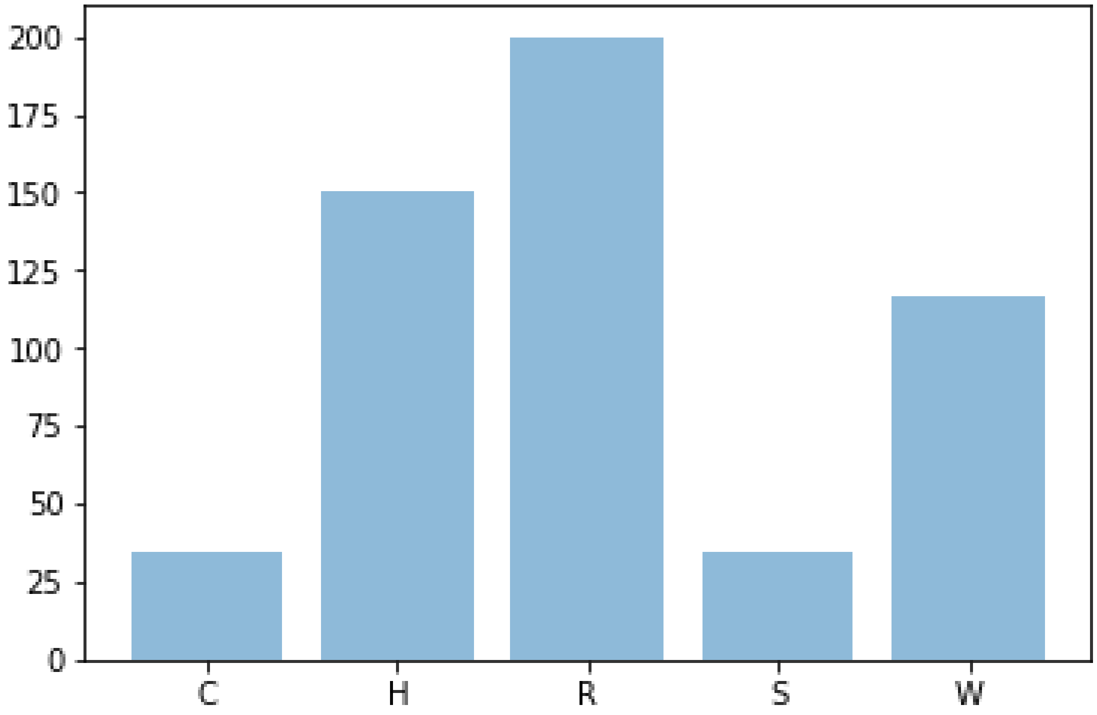
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ff04214a8
```



```
df2 = isFraud.groupby(['ProductCD']).median()
df2.head()
df3 = df2['TransactionAmt']
l1= df3.index.tolist()
l2 = []
l2.append(df3[0])
l2.append(df3[1])
l2.append(df3[2])
l2.append(df3[3])
l2.append(df3[4])
plt.bar(l1, l2, align='center', alpha=0.5)

#plt.show()
```

Saved successfully! ×

```
<BarContainer object of 5 artists>
```



We also see that W is the most bought thing which people have been faudec
most expensive.

Part 4 - Correlation Coefficient

```
# TODO: code to calculate correlation coefficient
isFraud4 = isFraud
isFraud4 = isFraud4.dropna(subset = ['TransactionDT'])
isFraud4 = isFraud4.dropna(subset = ['TransactionAmt'])
import scipy.stats as sp
spcor = sp.pearsonr(isFraud4['TransactionDT'], isFraud4['Transact:
spcor1 = sp.spearmanr(isFraud4['TransactionDT'], isFraud4['Transac
                                              nDT'], isFraud4['Transactior
```

Saved successfully!

```
(0.03975538815285562, 1.0867383084387185e-08)
```

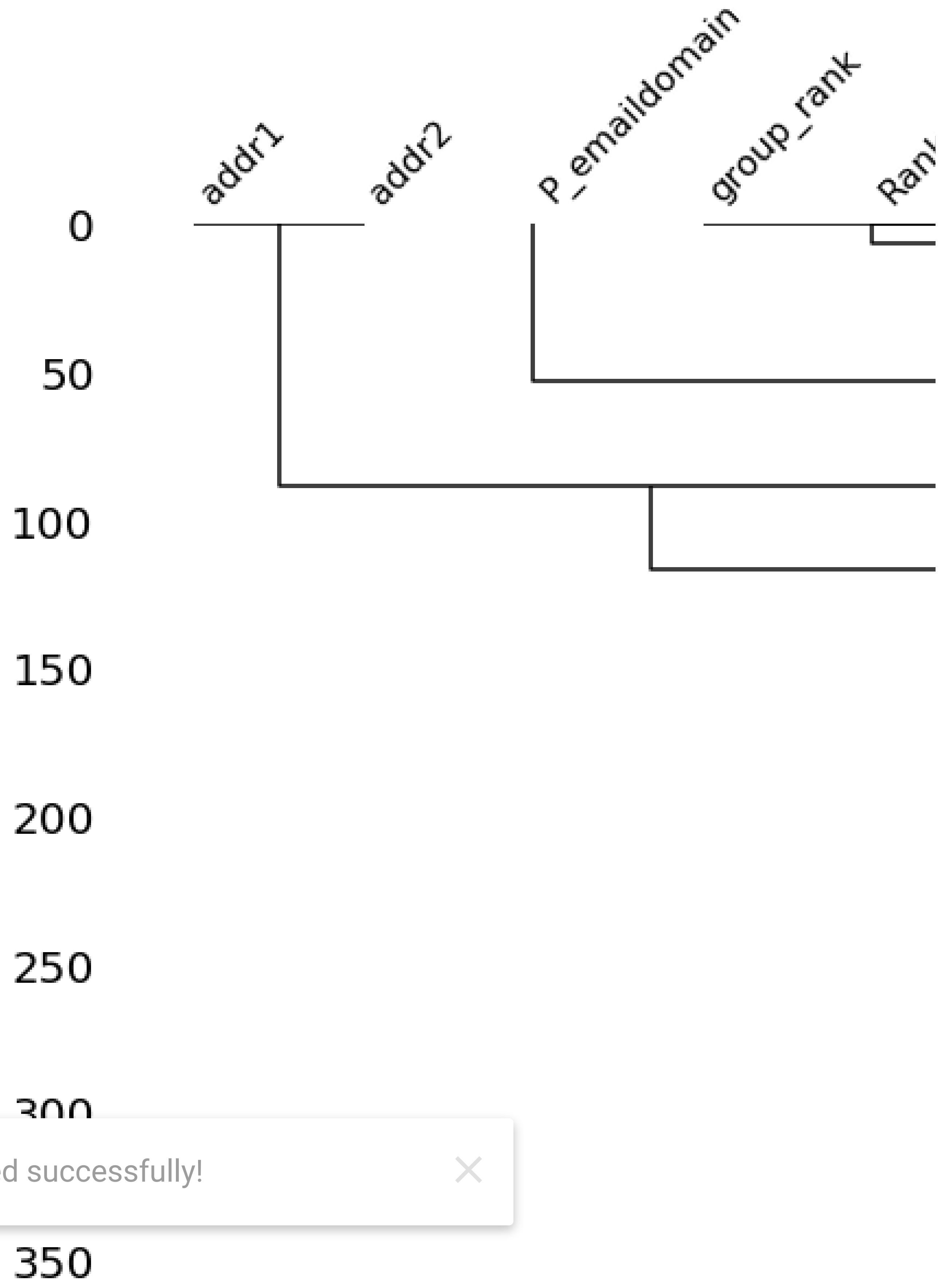
Write your answer here

Part 5 - Interesting Plot

```
# TODO: code to generate the plot here.

import missingno as msno
msno.dendrogram(isFraud)
```

Saved successfully!

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ff02de2b0
```



Saved successfully!

400

One method of finding correlation is the dendogram. Here, the hirerchical cl
each other can be found. From the plot, we can see and as expected, Device
for this is D=1-C wherein the C is the correlation while the D shows the dista
the correlated values. addr1, addr2 seem also correlated. These can also be

Part 6 - Prediction Model

```
import pandas as pd
test_transaction = pd.read_csv('/kaggle/input/ieee-fraud-detectior
cols = [col for col in test_transaction.columns if col in ['Transa
test_X = test_transaction[cols]
test_X.head()
```

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```

```
import pandas as pd
train_transaction = pd.read_csv('/kaggle/input/ieee-fraud-detectic
cols = [col for col in test_transaction.columns if col in ['Transa
X_train = train_transaction[cols]
```

Saved successfully!

```
# TODO: code for your final model
import pandas as pd
```

```
train_transaction = pd.read_csv('/kaggle/input/ieee-fraud-detecti
cols = [col for col in train_transaction.columns if col in ['isFra
y_train = train_transaction[cols]
y_train.head()
```

```
X_train.fillna(-1, inplace=True)
y_train.fillna(-1, inplace=True)
test_X.fillna(-1, inplace=True)
```

```
X_train['card6'] = pd.factorize(X_train['card6'])[0]+1
X_train['card4'] = pd.factorize(X_train['card4'])[0]+1
X_train['P_emaildomain'] = pd.factorize(X_train['P_emaildomain'])
X_train['R_emaildomain'] = pd.factorize(X_train['R_emaildomain'])
X_train['ProductCD'] = pd.factorize(X_train['ProductCD'])[0]+1
#X_train
test_X['card6'] = pd.factorize(test_X['card6'])[0]+1
test_X['card4'] = pd.factorize(test_X['card4'])[0]+1
test_X['P_emaildomain'] = pd.factorize(test_X['P_emaildomain'])[0
test_X['R_emaildomain'] = pd.factorize(test_X['R_emaildomain'])[0
test_X['ProductCD'] = pd.factorize(test_X['ProductCD'])[0]+1
test_X
```

```
df4 = pd.read_csv('/kaggle/input/ieee-fraud-detection/test_transa
```

```
model.fit(X_train, y_train)
f = model.predict(test_X)
```

```
df8=df4['TransactionID']
d = pd.DataFrame(f)
#df8.append(f)
df8 = pd.concat([df8,d], axis=1)
```

Saved successfully!

Part 7 - Final Result

Report the rank, score, number of entries, for your highest rank. Include a sn
to your Kaggle profile. Make sure to include a screenshot of your ranking. M

Kaggle Link: https://www.kaggle.com/golion/abhaygoyal-dsf/edit/run/2106

Highest Rank: 5983

Score: 0.5000

Number of entries: 1

INCLUDE IMAGE OF YOUR KAGGLE RANKING https://drive.google.com/ope

Saved successfully!

Saved successfully!