In [6]:

```
#import necessary packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
```

In [7]:

```
cd pwd
```

```
  File "<ipython-input-7-4c892a18346e>", line 1
    cd pwd
         ^
SyntaxError: invalid syntax
```

In [21]:

```
cd /home/chandu/downloads/
```

```
[Errno 2] No such file or directory: '/home/chandu/downloads/'
/home/chandu
```

In [22]:

```
cd /home/chandu/downloads/
```

```
[Errno 2] No such file or directory: '/home/chandu/downloads/'
/home/chandu
```

In [ ]:

```
cd /home/chandu/Downloads
```

In [8]:

```
data=pd.read_csv('creditcard.csv')
```

In [9]:

```
print(data.columns)
```

```
Index(['Time', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10',
       'V11', 'V12', 'V13', 'V14', 'V15', 'V16', 'V17', 'V18', 'V19', 'V20',
       'V21', 'V22', 'V23', 'V24', 'V25', 'V26', 'V27', 'V28', 'Amount',
       'Class'],
      dtype='object')
```

In [10]:

```
print(data.shape)
```

```
(284807, 31)
```

In [11]:

```
data.shape
```

Out[11]:

```
(284807, 31)
```

```
(284807, 31)
```

In [13]:

```python
print(data.describe)
```

```
<bound method NDFrame.describe of              Time         V1         V2        V3        V4        V
5  \
0             0.0  -1.359807  -0.072781  2.536347  1.378155 -0.338321
1             0.0   1.191857   0.266151  0.166480  0.448154  0.060018
2             1.0  -1.358354  -1.340163  1.773209  0.379780 -0.503198
3             1.0  -0.966272  -0.185226  1.792993 -0.863291 -0.010309
4             2.0  -1.158233   0.877737  1.548718  0.403034 -0.407193
...           ...        ...        ...       ...       ...       ...
284802   172786.0 -11.881118  10.071785 -9.834783 -2.066656 -5.364473
284803   172787.0  -0.732789  -0.055080  2.035030 -0.738589  0.868229
284804   172788.0   1.919565  -0.301254 -3.249640 -0.557828  2.630515
284805   172788.0  -0.240440   0.530483  0.702510  0.689799 -0.377961
284806   172792.0  -0.533413  -0.189733  0.703337 -0.506271 -0.012546

              V6        V7        V8        V9  ...       V21       V22  \
0       0.462388  0.239599  0.098698  0.363787  ... -0.018307  0.277838
1      -0.082361 -0.078803  0.085102 -0.255425  ... -0.225775 -0.638672
2       1.800499  0.791461  0.247676 -1.514654  ...  0.247998  0.771679
3       1.247203  0.237609  0.377436 -1.387024  ... -0.108300  0.005274
4       0.095921  0.592941 -0.270533  0.817739  ... -0.009431  0.798278
...          ...       ...       ...       ...  ...       ...       ...
284802 -2.606837 -4.918215  7.305334  1.914428  ...  0.213454  0.111864
284803  1.058415  0.024330  0.294869  0.584800  ...  0.214205  0.924384
284804  3.031260 -0.296827  0.708417  0.432454  ...  0.232045  0.578229
284805  0.623708 -0.686180  0.679145  0.392087  ...  0.265245  0.800049
284806 -0.649617  1.577006 -0.414650  0.486180  ...  0.261057  0.643078

             V23       V24       V25       V26       V27       V28  Amount  \
0      -0.110474  0.066928  0.128539 -0.189115  0.133558 -0.021053  149.62
1       0.101288 -0.339846  0.167170  0.125895 -0.008983  0.014724    2.69
2       0.909412 -0.689281 -0.327642 -0.139097 -0.055353 -0.059752  378.66
3      -0.190321 -1.175575  0.647376 -0.221929  0.062723  0.061458  123.50
4      -0.137458  0.141267 -0.206010  0.502292  0.219422  0.215153   69.99
...          ...       ...       ...       ...       ...       ...     ...
284802  1.014480 -0.509348  1.436807  0.250034  0.943651  0.823731    0.77
284803  0.012463 -1.016226 -0.606624 -0.395255  0.068472 -0.053527   24.79
284804 -0.037501  0.640134  0.265745 -0.087371  0.004455 -0.026561   67.88
284805 -0.163298  0.123205 -0.569159  0.546668  0.108821  0.104533   10.00
284806  0.376777  0.008797 -0.473649 -0.818267 -0.002415  0.013649  217.00

        Class
0           0
1           0
2           0
3           0
4           0
...       ...
284802      0
284803      0
284804      0
284805      0
284806      0

[284807 rows x 31 columns]>
```

In [14]:

```python
data=data.sample(frac=0.1,random_state=1)
print(data.shape)
```

```
(28481, 31)
```

In [15]:

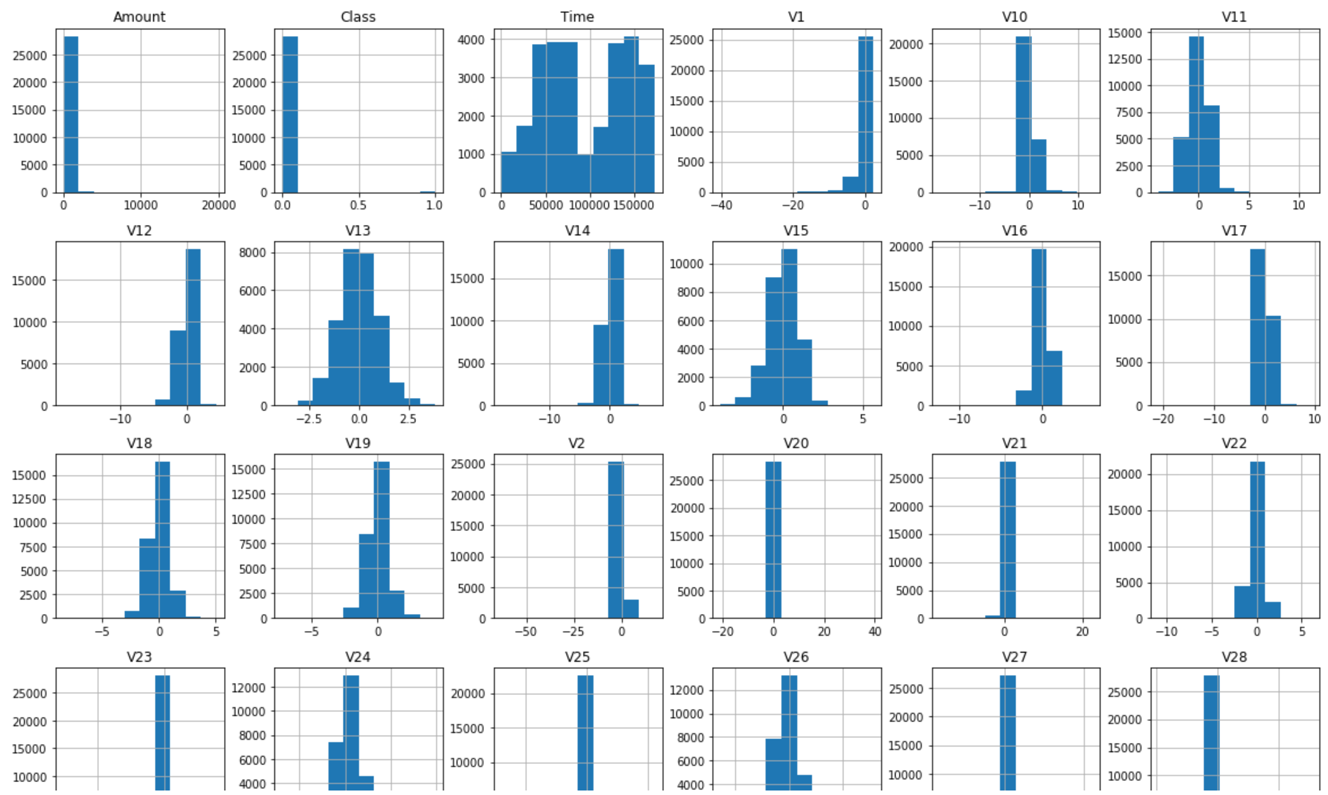
```python
print(data.describe)
```

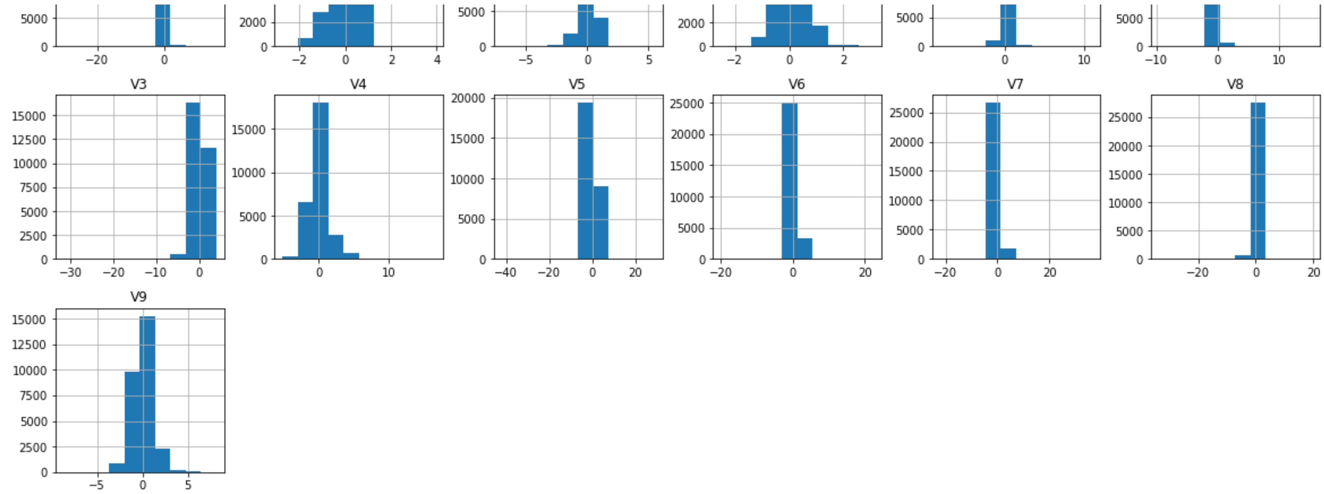
```
<bound method NDFrame.describe of              Time        V1        V2        V3        V4        V
```

```
<bound method NDFrame.describe of            Time        V1        V2        V3        V4        \
V6  \
169876  119907.0 -0.611712 -0.769705 -0.149759 -0.224877  2.028577 -2.019887
127467   78340.0 -0.814682  1.319219  1.329415  0.027273 -0.284871 -0.653985
137900   82382.0 -0.318193  1.118618  0.969864 -0.127052  0.569563 -0.532484
21513    31717.0 -1.328271  1.018378  1.775426 -1.574193 -0.117696 -0.457733
134700   80923.0  1.276712  0.617120 -0.578014  0.879173  0.061706 -1.472002
...          ...       ...       ...       ...       ...       ...       ...
2032      1574.0 -0.615776  0.654356  2.618793  0.857434 -0.487340  0.593957
240932  150813.0 -3.517229  3.326821 -3.590262  0.674769 -0.679266 -0.469516
3701      3169.0 -0.315540  1.054303  1.484711  1.138262  0.394713 -0.168883
153365   98752.0 -3.580417  4.100916 -2.577720 -1.476718 -0.006201 -2.008418
97365    66187.0  1.213349  0.227172 -0.886860  1.345683  2.254592  3.788565

              V7        V8        V9  ...       V21       V22       V23  \
169876  0.292491 -0.523020  0.358468  ... -0.075208  0.045536  0.380739
127467  0.321552  0.435975 -0.704298  ... -0.128619 -0.368565  0.090660
137900  0.706252 -0.064966 -0.463271  ... -0.305402 -0.774704 -0.123884
21513   0.681867 -0.031641  0.383872  ... -0.220815 -0.419013 -0.239197
134700  0.373692 -0.287204 -0.084482  ... -0.160161 -0.430404 -0.076738
...          ...       ...       ...  ...       ...       ...       ...
2032   -0.095191  0.426786  0.011607  ...  0.010440  0.113631 -0.313035
240932 -1.135362  2.778095 -2.404956  ...  0.455767  0.388102  0.268986
3701    0.737923 -0.061284 -0.952381  ...  0.005626  0.094740  0.024370
153365  0.887262  0.304192  2.879710  ... -0.194866  0.571678 -0.001519
97365  -0.521816  0.891366 -0.776104  ...  0.102366  0.116553 -0.166854

             V24       V25       V26       V27       V28  Amount  Class
169876  0.023440 -2.220686 -0.201146  0.066501  0.221180    1.79      0
127467  0.401147 -0.261034  0.080621  0.162427  0.059456    1.98      0
137900 -0.495687 -0.018148  0.121679  0.249050  0.092516    0.89      0
21513   0.009967  0.232829  0.814177  0.098797 -0.004273   15.98      0
134700  0.258708  0.552170  0.370701 -0.034255  0.041709    0.76      0
...          ...       ...       ...       ...       ...     ...    ...
2032   -0.015388  0.213878 -0.268579  0.117815  0.075734    9.99      0
240932  0.382692 -0.653335  2.192962 -0.953907 -0.137082    0.76      0
3701    0.091800 -0.463470 -0.457328  0.194541  0.166039   19.60      0
153365  0.009117  0.321669  0.034900  0.785417  0.353092    0.92      0
97365   1.015984  0.755462  0.169925 -0.005633  0.017400   19.34      0

[28481 rows x 31 columns]>
```

In [16]:

```
data.hist(figsize=(20,20))
plt.show()
```





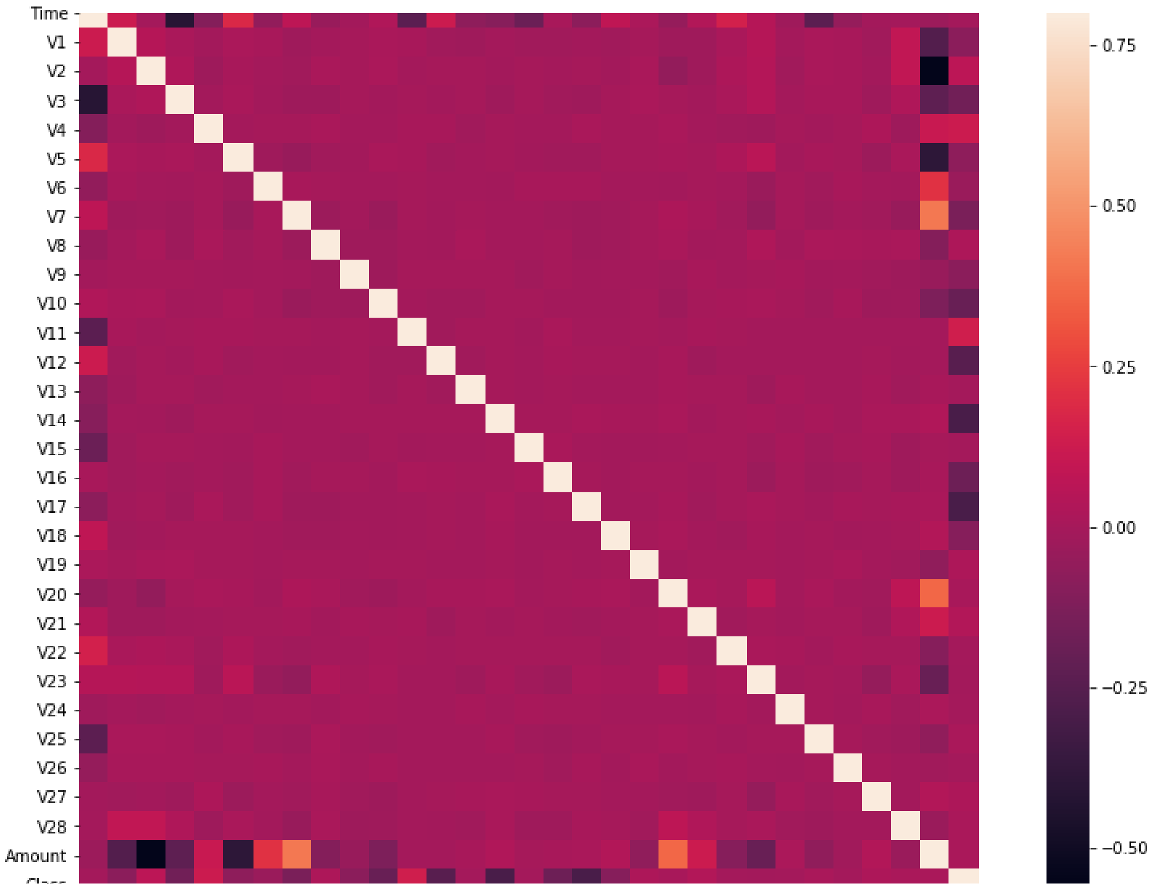

In [24]:

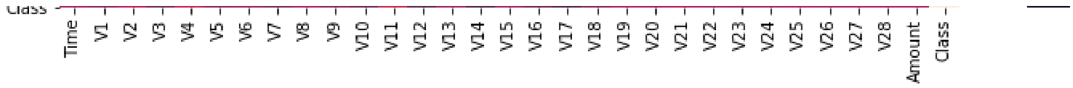
```
#determine the number of fraudulent cases in dataset
Fraud=data[data['Class']==1]
valid=data[data['Class']==0]
outlier_fraction=len(Fraud)/float(len(valid))
print(outlier_fraction)
print("Fraud transactions:{}".format(len(Fraud)))
print("valid transations:{}".format(len(valid)))
```

```
0.0017234102419808666
Fraud transactions:49
valid transations:28432
```

In [25]:

```
#correlationmatrix
datacorr=data.corr()
fig=plt.figure(figsize=(15,10))
sns.heatmap(datacorr,vmax=0.8,square=True)
plt.show()
```






In [42]:

```
# Get all the columns from the dataFrame
columns = data.columns.tolist()

# Filter the columns to remove data we do not want
columns = [c for c in columns if c not in ["Class"]]

# Store the variable we'll be predicting on
target = "Class"

X = data[columns]
Y = data[target]

# Print shapes
print(X.shape)
print(Y.shape)
```

```
(28481, 30)
(28481,)
```

In [43]:

```
from sklearn.metrics import classification_report, accuracy_score
from sklearn.ensemble import IsolationForest
from sklearn.neighbors import LocalOutlierFactor

# define random states
state = 1

# define outlier detection tools to be compared
classifiers = {
    "Isolation Forest": IsolationForest(max_samples=len(X),
                                        contamination=outlier_fraction,
                                        random_state=state),
    "Local Outlier Factor": LocalOutlierFactor(
        n_neighbors=20,
        contamination=outlier_fraction)}
```

In [44]:

```
# Fit the model
plt.figure(figsize=(9, 7))
n_outliers = len(Fraud)


for i, (clf_name, clf) in enumerate(classifiers.items()):

    # fit the data and tag outliers
    if clf_name == "Local Outlier Factor":
        y_pred = clf.fit_predict(X)
        scores_pred = clf.negative_outlier_factor_
    else:
        clf.fit(X)
        scores_pred = clf.decision_function(X)
        y_pred = clf.predict(X)

    # Reshape the prediction values to 0 for valid, 1 for fraud.
    y_pred[y_pred == 1] = 0
    y_pred[y_pred == -1] = 1

    n_errors = (y_pred != Y).sum()

    # Run classification metrics
    print('{}: {}'.format(clf_name, n_errors))
    print(accuracy_score(Y, y_pred))
    print(classification_report(Y, y_pred))
```

```
/home/chandu/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/iforest.py:247: FutureWarning:
```

```
/home/chandu/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/iforest.py:247: FutureWarning:
behaviour="old" is deprecated and will be removed in version 0.22. Please use behaviour="new", whi
ch makes the decision_function change to match other anomaly detection algorithm API.
  FutureWarning)
/home/chandu/anaconda3/lib/python3.7/site-packages/sklearn/ensemble/iforest.py:415:
DeprecationWarning: threshold_ attribute is deprecated in 0.20 and will be removed in 0.22.
  " be removed in 0.22.", DeprecationWarning)
```

```
Isolation Forest: 71
0.99750711000316
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     28432
           1       0.28      0.29      0.28        49

    accuracy                           1.00     28481
   macro avg       0.64      0.64      0.64     28481
weighted avg       1.00      1.00      1.00     28481

Local Outlier Factor: 97
0.9965942207085425
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     28432
           1       0.02      0.02      0.02        49

    accuracy                           1.00     28481
   macro avg       0.51      0.51      0.51     28481
weighted avg       1.00      1.00      1.00     28481



<Figure size 648x504 with 0 Axes>
```

In [ ]: