In [7]:

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

In [8]:

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

In [9]:

```
data.head()
```

Out[9]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533

5 rows × 31 columns

In [4]:

```
data.shape ##no of transactions and columns
```

Out[4]:

```
(284807, 31)
```

In [10]:

```
data.describe()
```

Out[10]:

	Time	V1	V2	V3	V4	
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070
mean	94813.859575	3.919560e-15	5.688174e-16	-8.769071e-15	2.782312e-15	-1.552563
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.91597
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.43358
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.11926
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167

8 rows × 31 columns

In [11]:

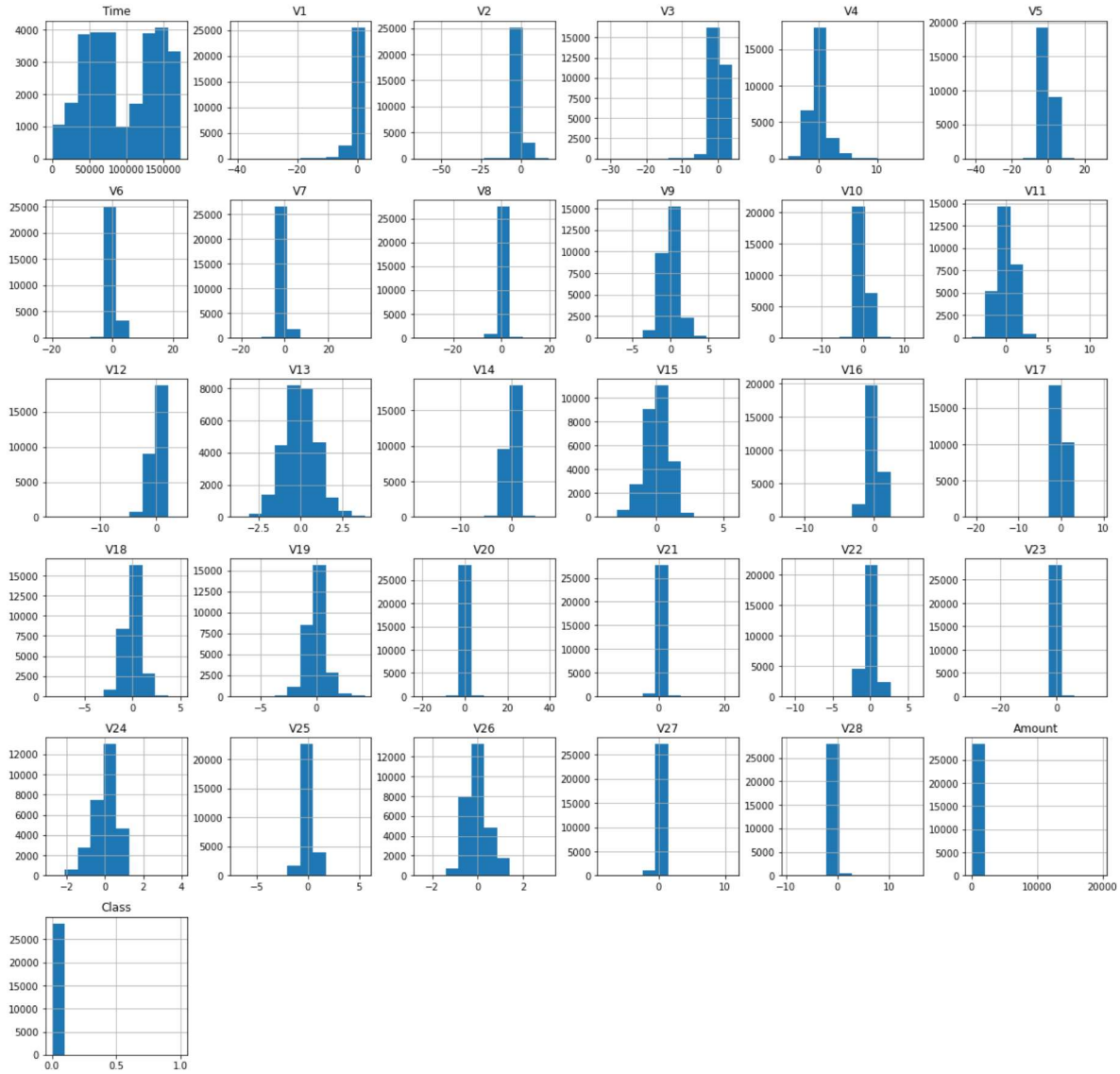
```
data = data.sample(frac = 0.1, random_state = 1) ##working with 10% of the whole dataset fo

print(data.shape)
```

```
(28481, 31)
```

In [13]:

```
data.hist(figsize = (20,20)) ##histograms for each parameter
plt.show()
```



In [14]:

```
# Determine no of fraud cases in dataset
Fraud = data[data['Class'] == 1]
Valid = data[data['Class'] == 0]

outlier_fraction = len(Fraud) / float(len(Valid))
```

In [15]:

```
print(outlier_fraction)
print('Fraud Cases:{}'.format(len(Fraud)))
print('Valid Cases:{}'.format(len(Valid)))
```

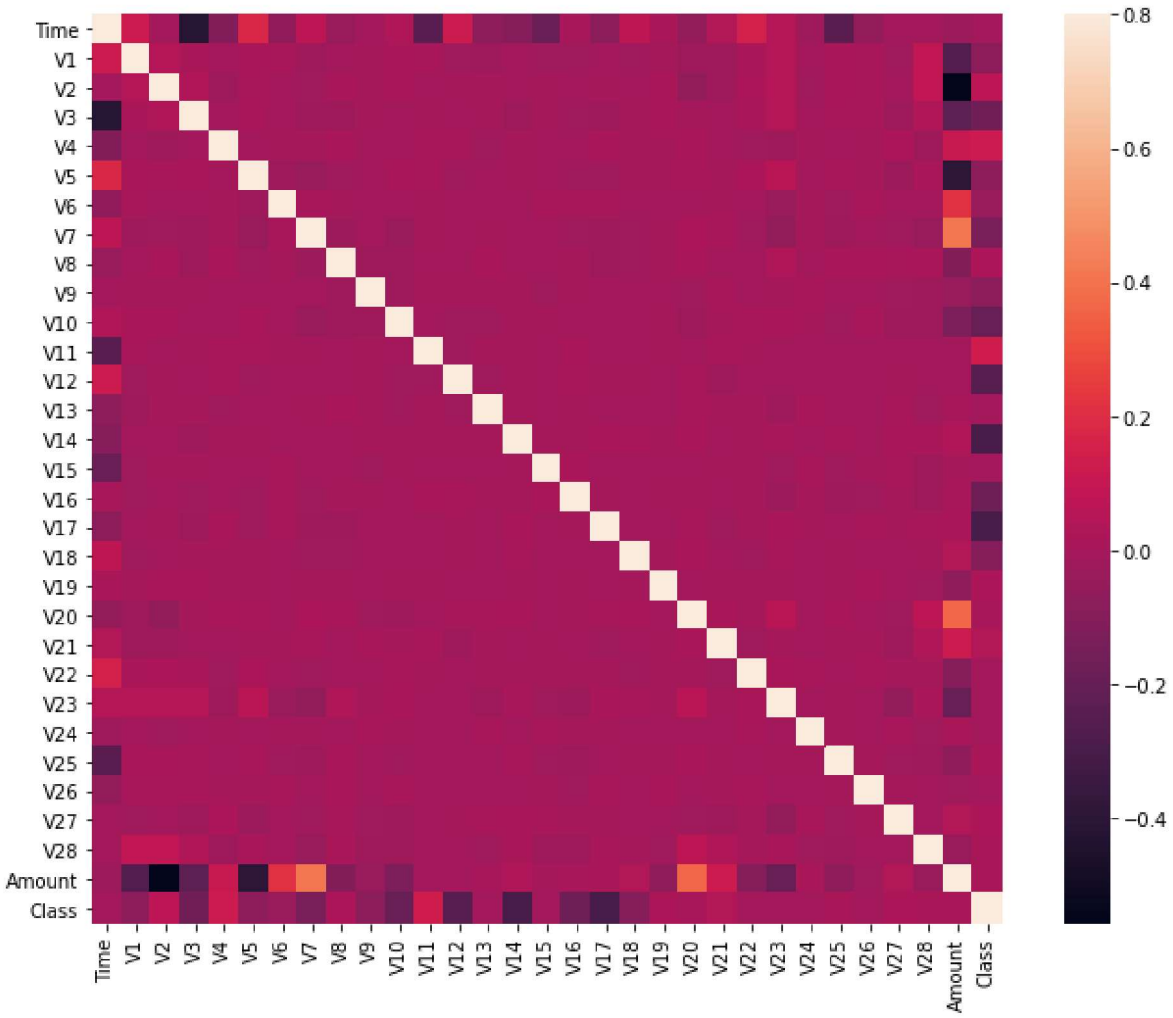
```
0.0017234102419808666
Fraud Cases:49
Valid Cases:28432
```

In [16]:

```
# Correlation Matrix

corrmat = data.corr()
fig = plt.figure(figsize = (12,9))

sns.heatmap(corrmat, vmax = .8, square = True)
plt.show()
```



In [18]:

```
# get all the columns from the dataframe
columns = data.columns.tolist()

#removing the columns from where we do not need the data

columns = [c for c in columns if c not in ["Class"]]
target = "Class"

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

print(X.shape)
print(Y.shape)
```

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

In [20]:

```
from sklearn.metrics import classification_report, accuracy_score
from sklearn.ensemble import IsolationForest              #used for anomaly detection
from sklearn.neighbors import LocalOutlierFactor          # anomaly detection methods

# define a random state
state = 1

# defining the outlier detection methods

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 [21]:

```
# fit the model

n_outliers = len(Fraud)

for i, (clf_name, clf) in enumerate (classifiers.items()):
    if clf_name == "Local Outlier Factor":
        y_pred = clf.fit_predict(X)
        scores_pred = clf.negative_outlier_factor_
    else:
        clf.fit(X)
        sores_pred = clf.decision_function(X)
        y_pred = clf.predict(X)
    y_pred[y_pred == 1] = 0
    y_pred[y_pred == -1] = 1

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

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

```
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
```

In [ ]: