In [1]:
```
# Import necessary libraries
import warnings
warnings.filterwarnings("ignore")
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:
```
# Read the dataset from the csv file
df = pd.read_csv('creditcard.csv')
```

1. Exploratory Data Analysis

In [3]:
```
# Display the first few rows
df.head()
```

Out[3]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V2
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.1104
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.1012
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.9094
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.1903
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.1374

5 rows × 31 columns

In [4]:
```
# Check datatypes
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
 20  V20     284807 non-null  float64
 21  V21     284807 non-null  float64
 22  V22     284807 non-null  float64
 23  V23     284807 non-null  float64
 24  V24     284807 non-null  float64
 25  V25     284807 non-null  float64
 26  V26     284807 non-null  float64
 27  V27     284807 non-null  float64
 28  V28     284807 non-null  float64
 29  Amount  284807 non-null  float64
 30  Class   284807 non-null  int64  
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

In [5]:
```
# Checking missing values
```

```
df.isnull().sum()
```

Out[5]:
```
Time      0
V1        0
V2        0
V3        0
V4        0
V5        0
V6        0
V7        0
V8        0
V9        0
V10       0
V11       0
V12       0
V13       0
V14       0
V15       0
V16       0
V17       0
V18       0
V19       0
V20       0
V21       0
V22       0
V23       0
V24       0
V25       0
V26       0
V27       0
V28       0
Amount    0
Class     0
dtype: int64
```

In [6]:
```
# Check the duplicate values
df[df.duplicated(keep='first')]
```

Out[6]:

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22
33	26.0	-0.529912	0.873892	1.347247	0.145457	0.414209	0.100223	0.711206	0.176066	-0.286717	...	0.046949	0.208105
35	26.0	-0.535388	0.865268	1.351076	0.147575	0.433680	0.086983	0.693039	0.179742	-0.285642	...	0.049526	0.206537
113	74.0	1.038370	0.127486	0.184456	1.109950	0.441699	0.945283	-0.036715	0.350995	0.118950	...	0.102520	0.605089
114	74.0	1.038370	0.127486	0.184456	1.109950	0.441699	0.945283	-0.036715	0.350995	0.118950	...	0.102520	0.605089
115	74.0	1.038370	0.127486	0.184456	1.109950	0.441699	0.945283	-0.036715	0.350995	0.118950	...	0.102520	0.605089
...	...	...	...	...	...	...	...	...	...	...	...	...	...
282987	171288.0	1.912550	-0.455240	-1.750654	0.454324	2.089130	4.160019	-0.881302	1.081750	1.022928	...	-0.524067	-1.337510
283483	171627.0	-1.464380	1.368119	0.815992	-0.601282	-0.689115	-0.487154	-0.303778	0.884953	0.054065	...	0.287217	0.947825
283485	171627.0	-1.457978	1.378203	0.811515	-0.603760	-0.711883	-0.471672	-0.282535	0.880654	0.052808	...	0.284205	0.949659
284191	172233.0	-2.667936	3.160505	-3.355984	1.007845	-0.377397	-0.109730	-0.667233	2.309700	-1.639306	...	0.391483	0.266536
284193	172233.0	-2.691642	3.123168	-3.339407	1.017018	-0.293095	-0.167054	-0.745886	2.325616	-1.634651	...	0.402639	0.259746

1081 rows × 31 columns

In [7]:
```
# Display the statistics of each column such as min, max, mean, Standard deviation and quartiles.
df.describe()
```

Out[7]:

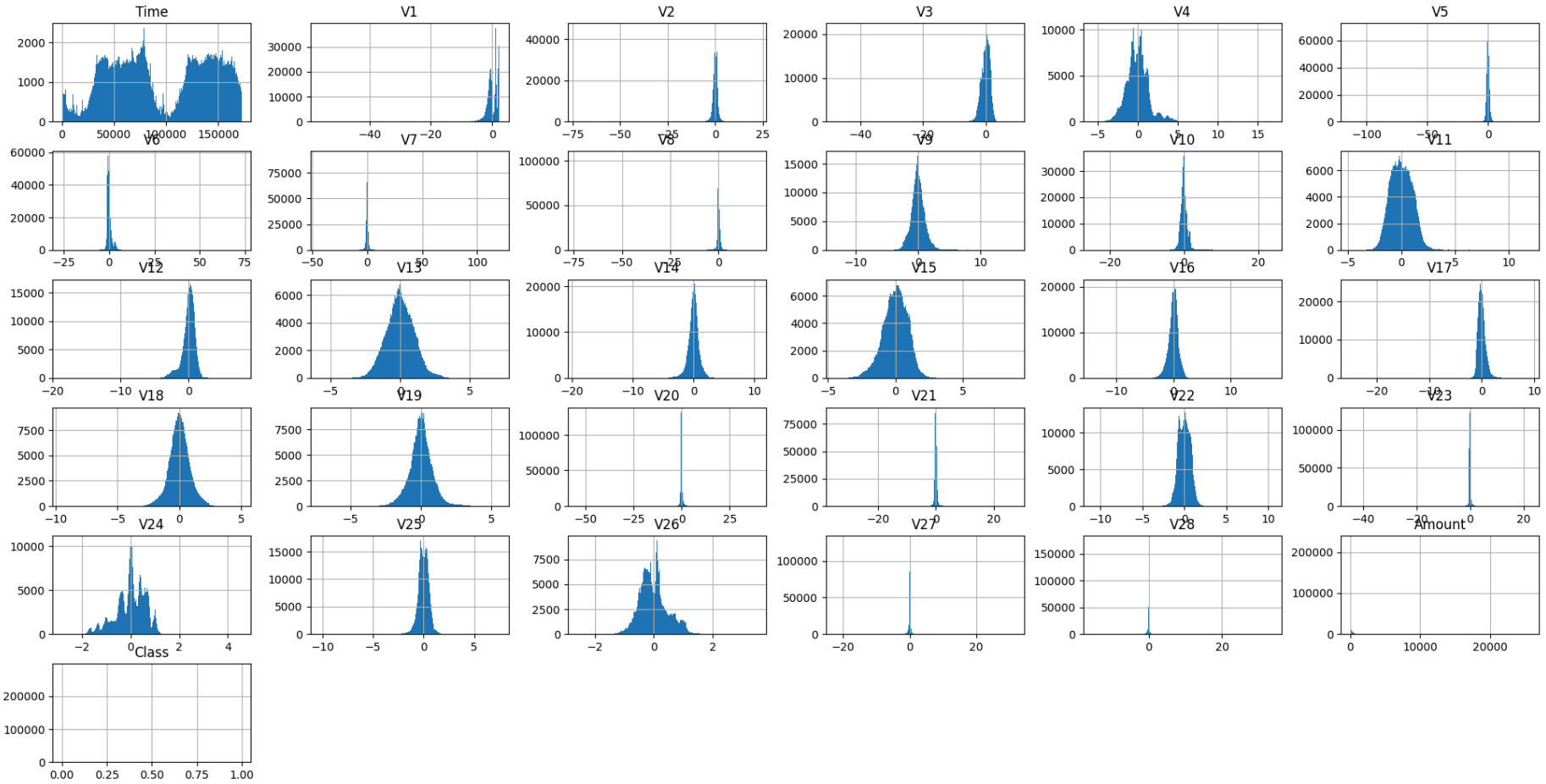
	Time	V1	V2	V3	V4	V5	V6	V7	
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.84807(
mean	94813.859575	1.168375e-15	3.416908e-16	-1.379537e-15	2.074095e-15	9.604066e-16	1.487313e-15	-5.556467e-16	1.21348
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247e+00	1.332271e+00	1.237094e+00	1.19435:
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433e+02	-2.616051e+01	-4.355724e+01	-7.32167
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.915971e-01	-7.682956e-01	-5.540759e-01	-2.08629
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.433583e-02	-2.741871e-01	4.010308e-02	2.23580
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.119264e-01	3.985649e-01	5.704361e-01	3.27345
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167e+01	7.330163e+01	1.205895e+02	2.00072

8 rows × 31 columns

In [8]:
```
# Class is imbalanced
df['Class'].value_counts()
```

Out[8]:
```
Class
0    284315
1       492
Name: count, dtype: int64
```

In [9]:
```
# Plot the histogram of each columns
df.hist(bins=250, figsize=(24,12))
plt.show()
```



2. Data preprocessing

In [10]:
```
# Drop duplicate values
df = df.drop_duplicates(keep='first')
df[df.duplicated(keep='first')]
```

Out[10]:

Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V24	V25	V26	V27	V28	Amount	Class

0 rows × 31 columns

In [11]:
```
df.columns
```

Out[11]:
```
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 [12]:
```
# Create clip columns for outlier treatment
clip_columns = df[['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']]
clip_columns.describe()
```

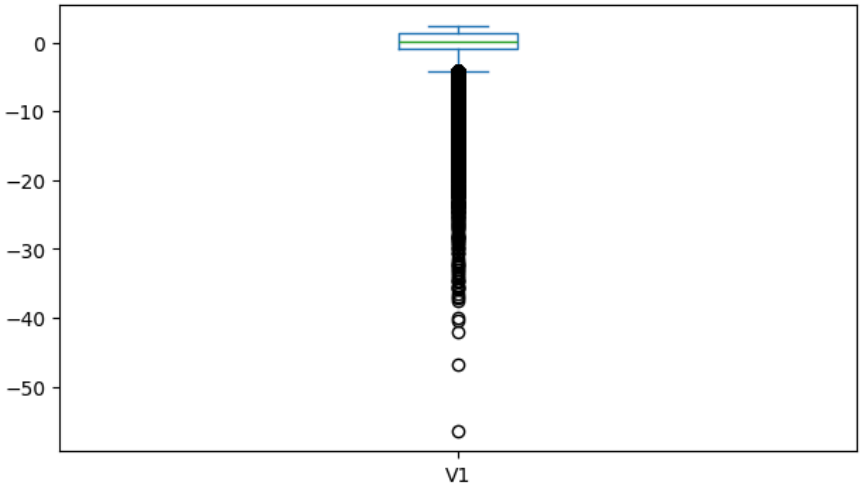
Out[12]:

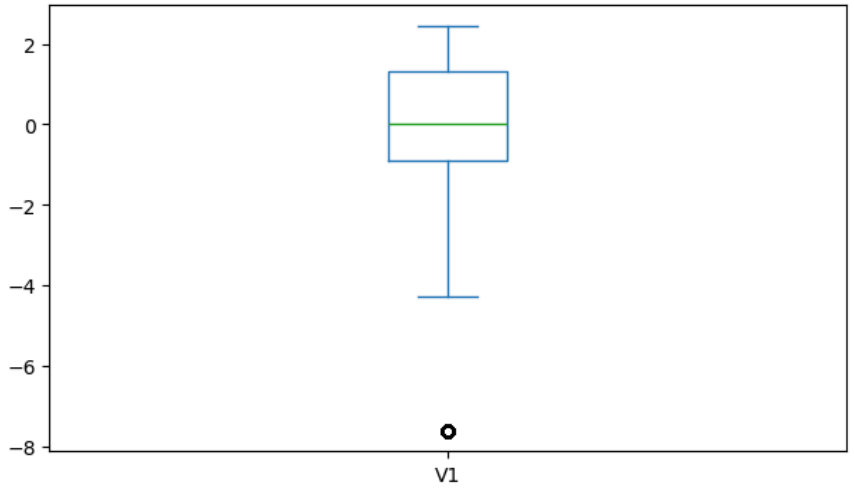
	V1	V2	V3	V4	V5	V6	V7	V8	
count	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.0
mean	0.005917	-0.004135	0.001613	-0.002966	0.001828	-0.001139	0.001801	-0.000854	-0.0
std	1.948026	1.646703	1.508682	1.414184	1.377008	1.331931	1.227664	1.179054	1.0
min	-56.407510	-72.715728	-48.325589	-5.683171	-113.743307	-26.160506	-43.557242	-73.216718	-13.4
25%	-0.915951	-0.600321	-0.889682	-0.850134	-0.689830	-0.769031	-0.552509	-0.208828	-0.6
50%	0.020384	0.063949	0.179963	-0.022248	-0.053468	-0.275168	0.040859	0.021898	-0.0
75%	1.316068	0.800283	1.026960	0.739647	0.612218	0.396792	0.570474	0.325704	0.5
max	2.454930	22.057729	9.382558	16.875344	34.801666	73.301626	120.589494	20.007208	15.5

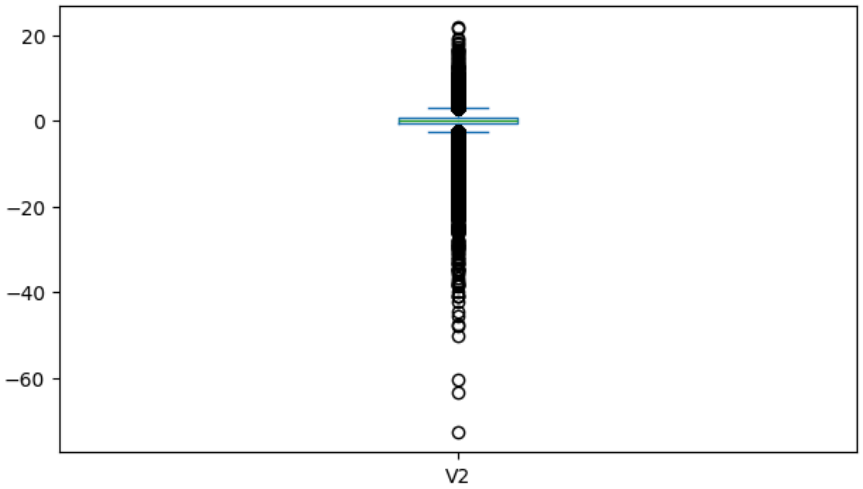
8 rows × 28 columns

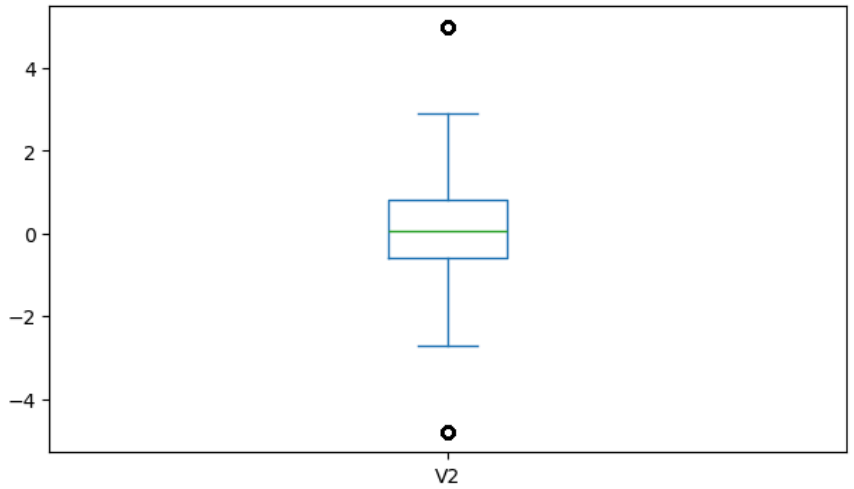
In [13]:
```
# Handle the outlier for clip columns and display the change
for i in clip_columns:
    plt.figure(figsize=(16,9))
    plt.subplot(2,2,1)
    df[i].plot.box();
    df[i][df[i]<(df[i].quantile(0.25)-1.5*(df[i].quantile(0.75)-df[i].quantile(0.25))) ] = (df[i].quantile(0.25)-3*(df[i].quan
```

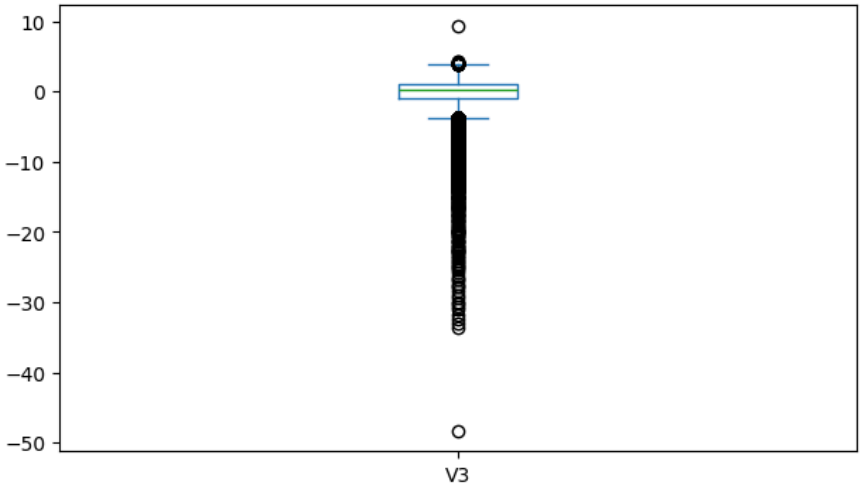
```
    df[i][df[i]>(df[i].quantile(0.75)+1.5*(df[i].quantile(0.75)-df[i].quantile(0.25)))] = (df[i].quantile(0.75)+3*(df[i].quant
    plt.subplot(2,2,2)
    df[i].plot.box();
```

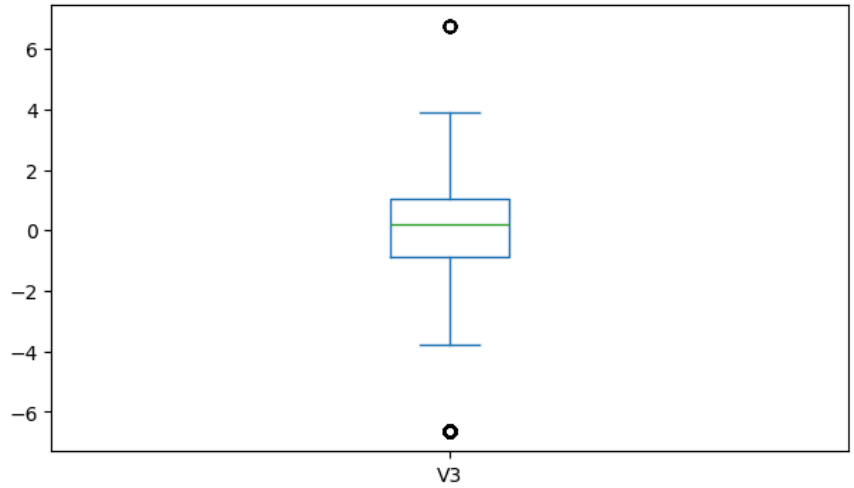


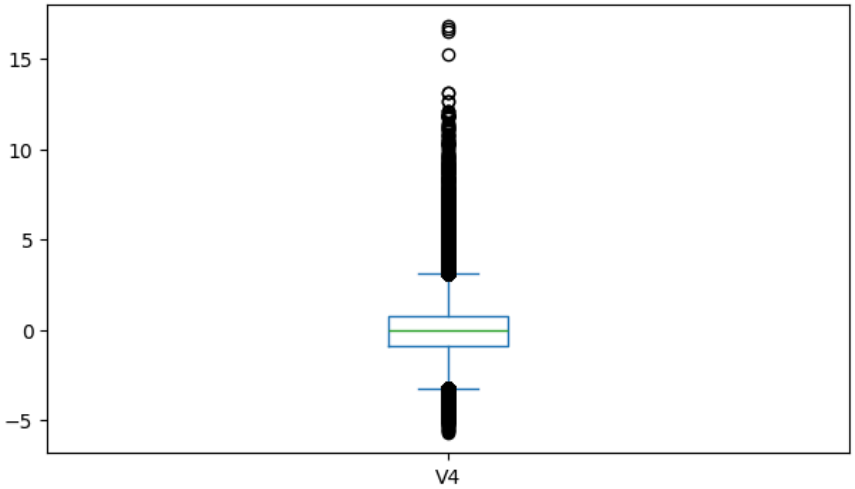


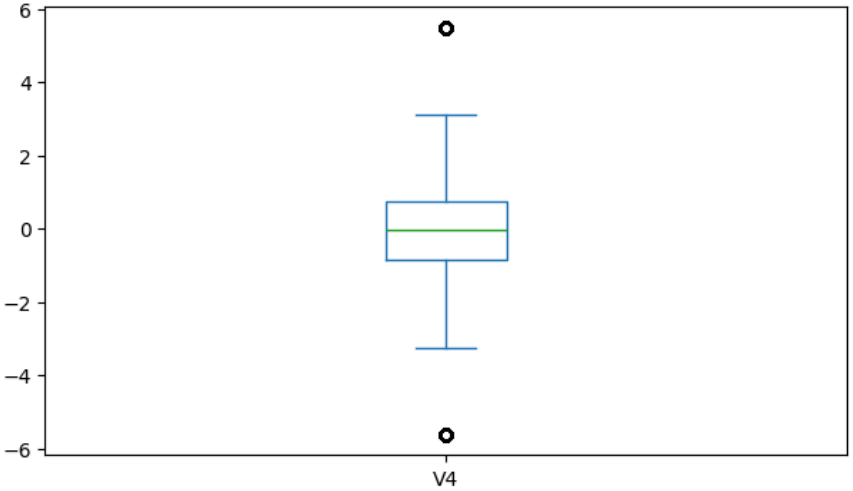


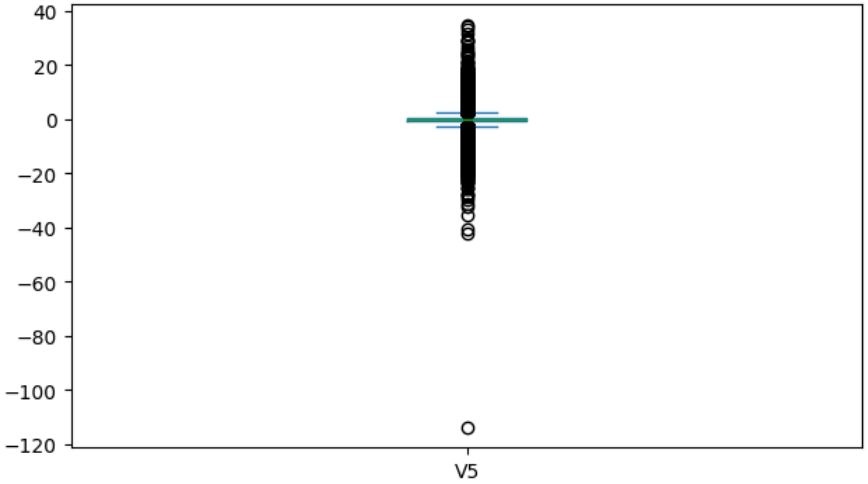


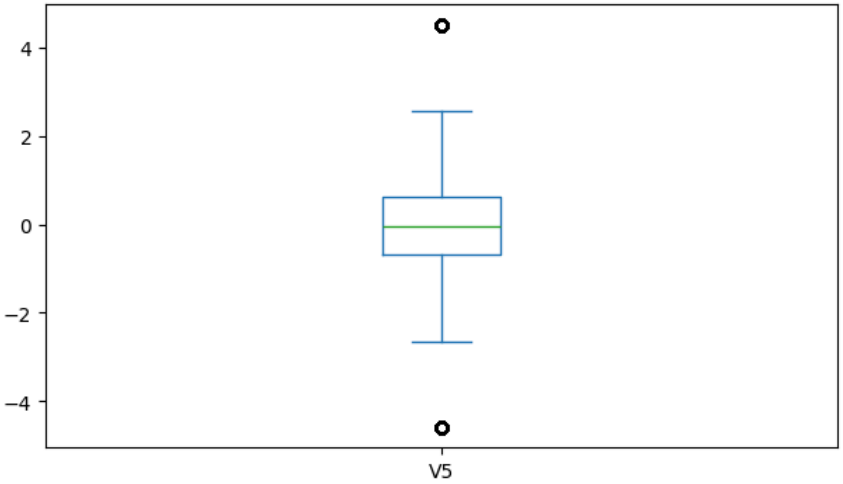

10
0
−10
−20
−30
−40
−50
V3

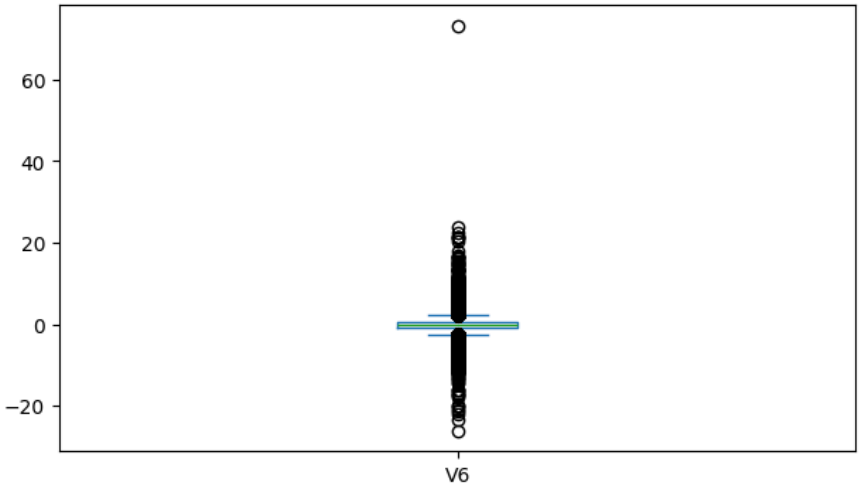

6
4
2
0
−2
−4
−6
V3

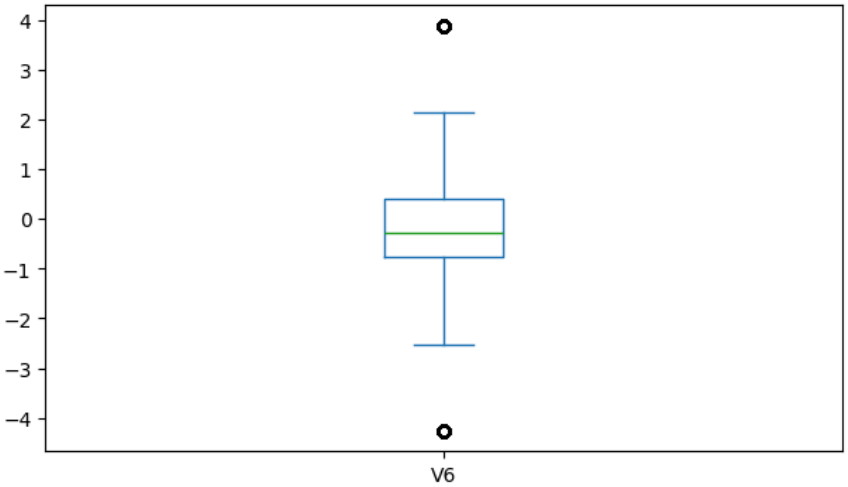

15
10
5
0
−5
V4

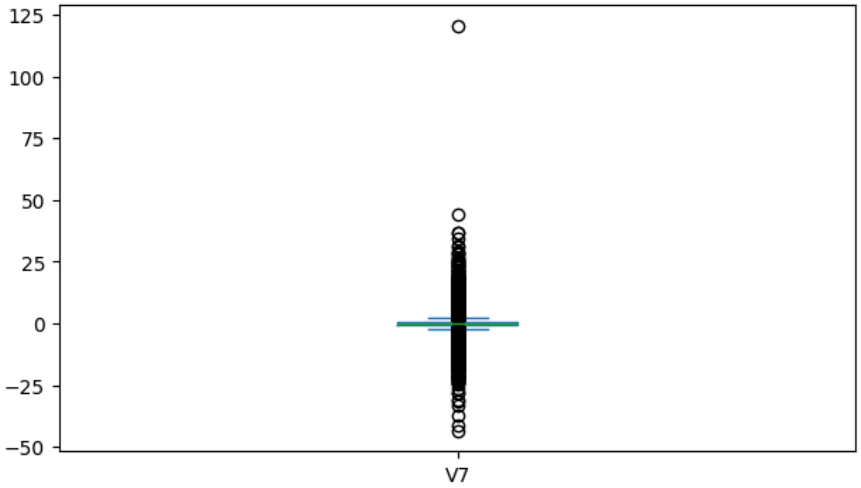

6
4
2
0
−2
−4
−6
V4

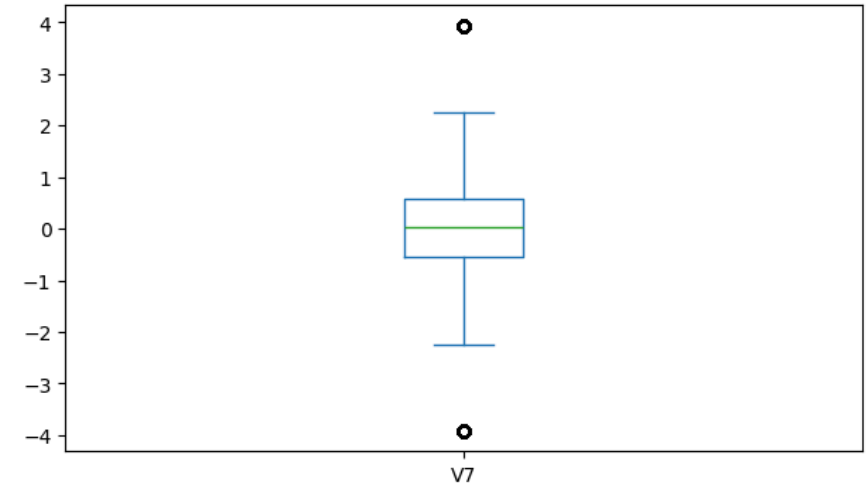

40
20
0
−20
−40
−60
−80
−100
−120
V5

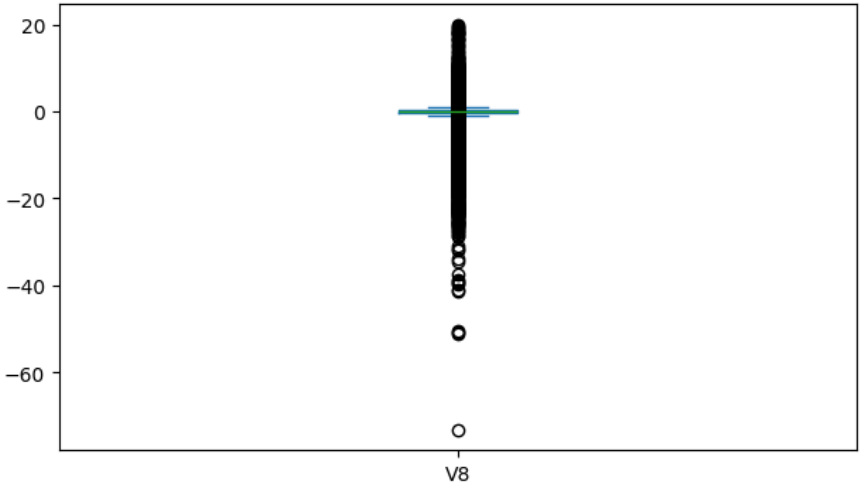

4
2
0
−2
−4
V5

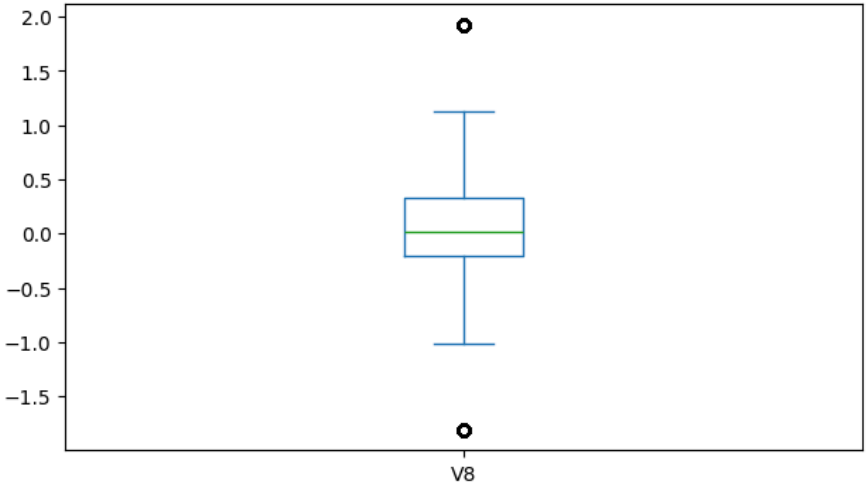

60
40
20
0
−20
V6

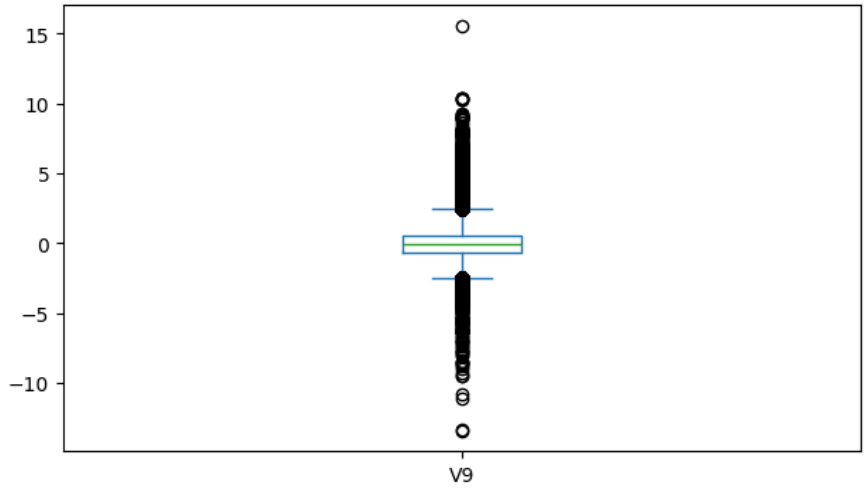

4
3
2
1
0
−1
−2
−3
−4
V6

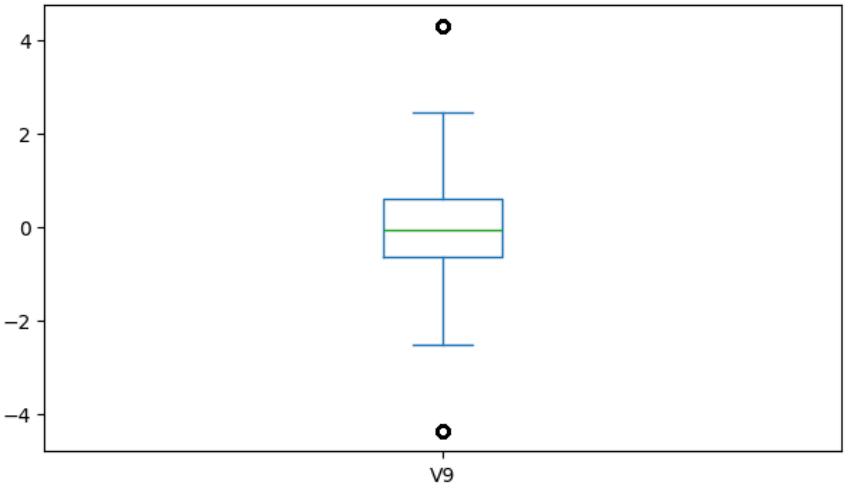

125
100
75
50
25
0
−25
−50
V7

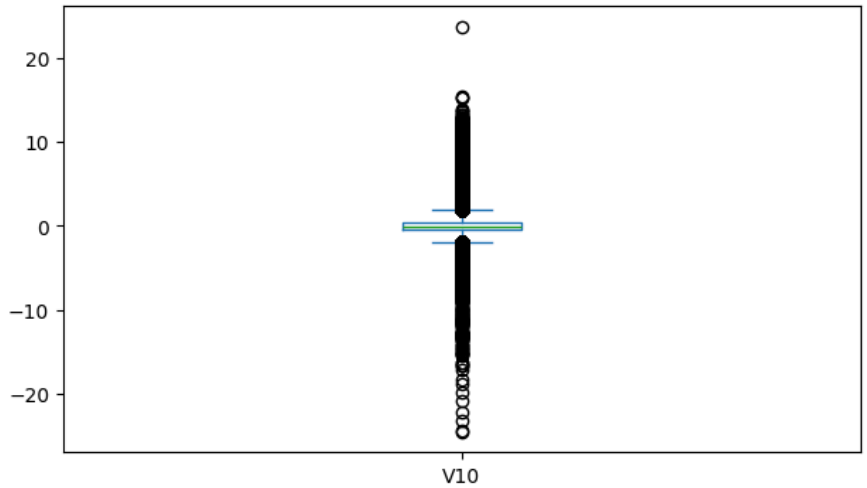

4
3
2
1
0
−1
−2
−3
−4
V7

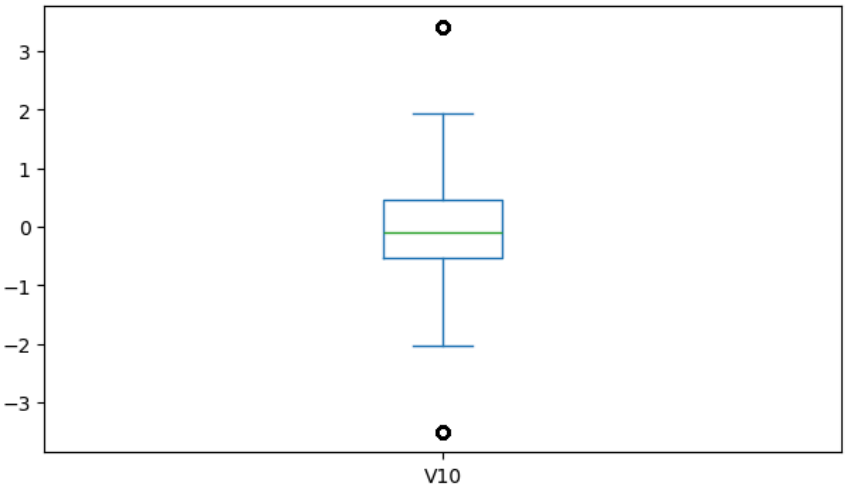

20
0
−20
−40
−60
V8

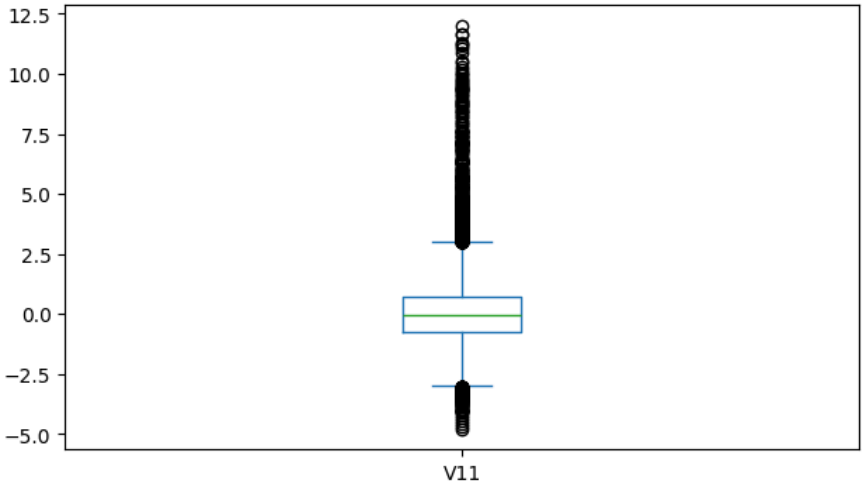

2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
V8

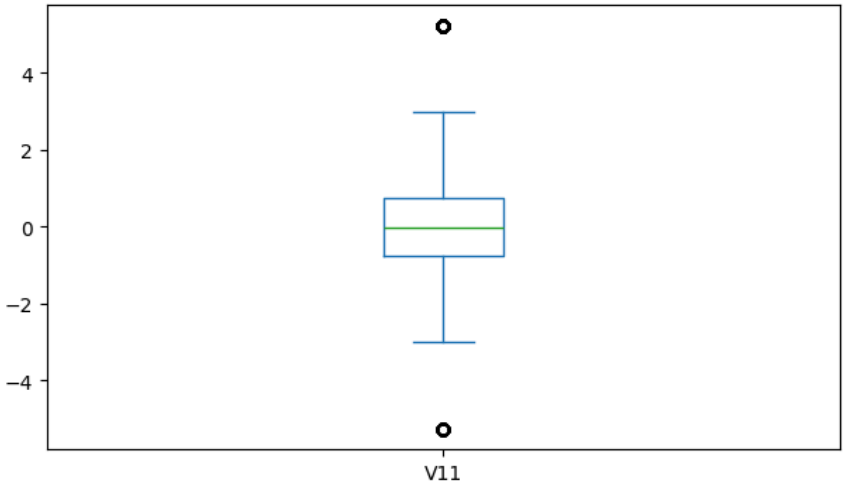

15
10
5
0
−5
−10
V9

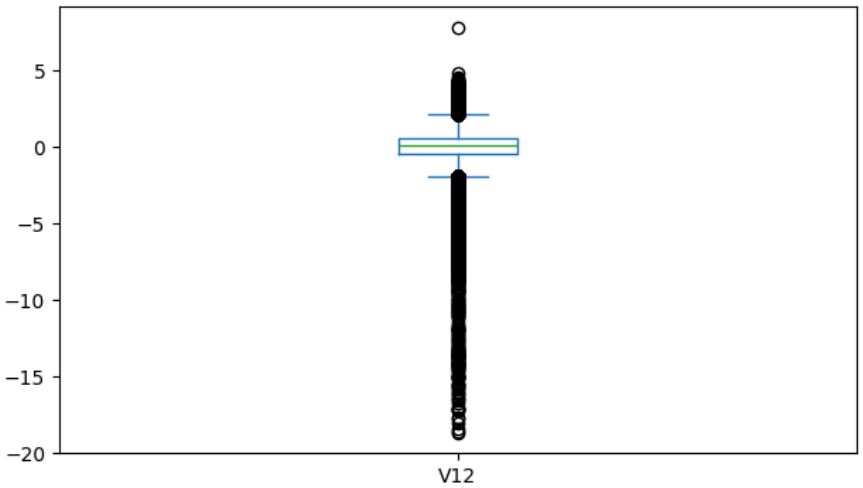

4
2
0
−2
−4
V9

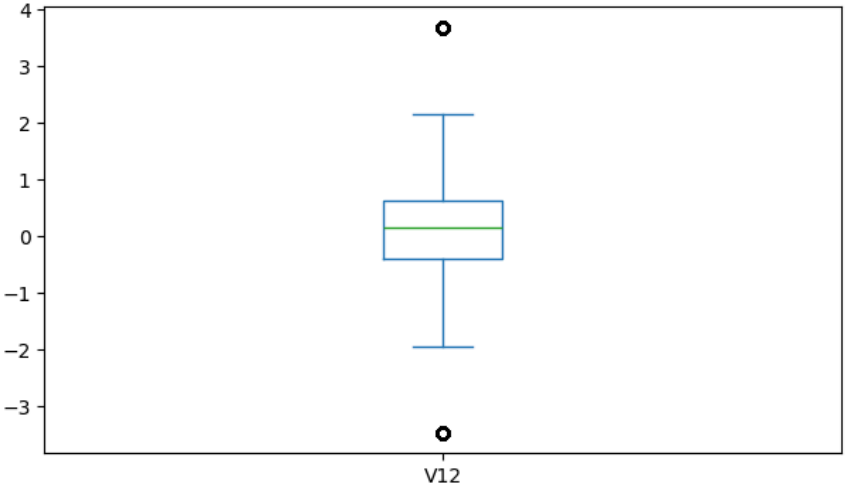

20
10
0
−10
−20
V10

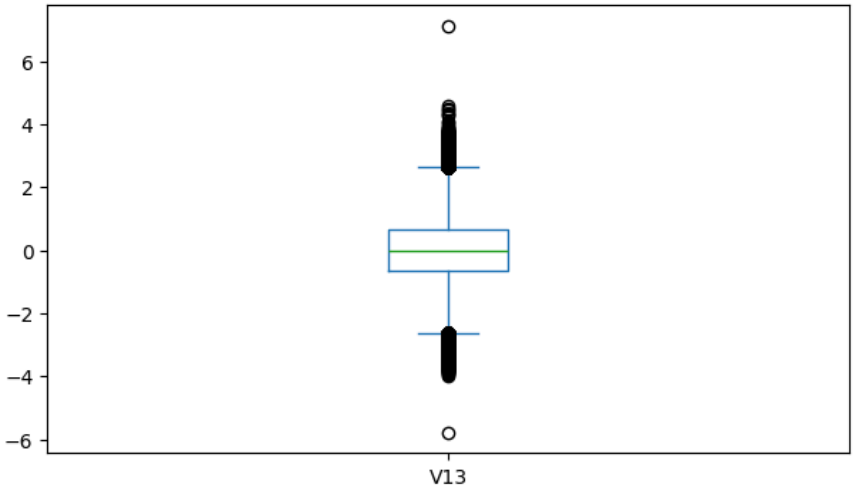

3
2
1
0
−1
−2
−3
V10

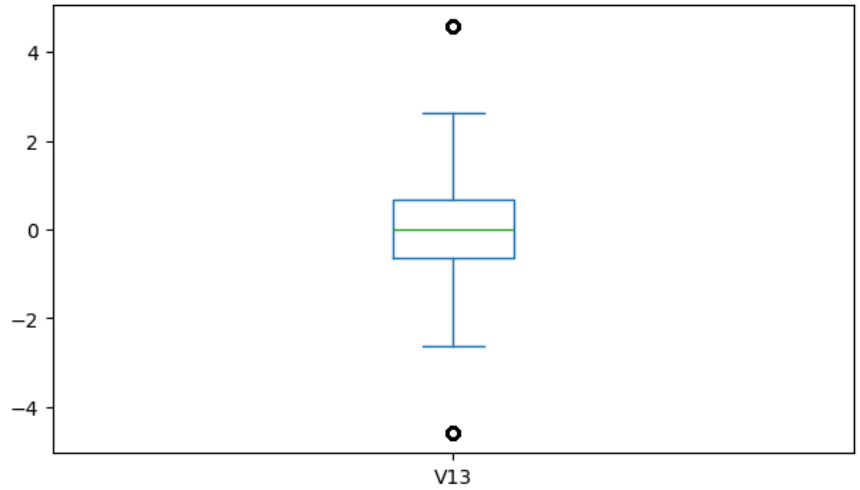

12.5
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
V11

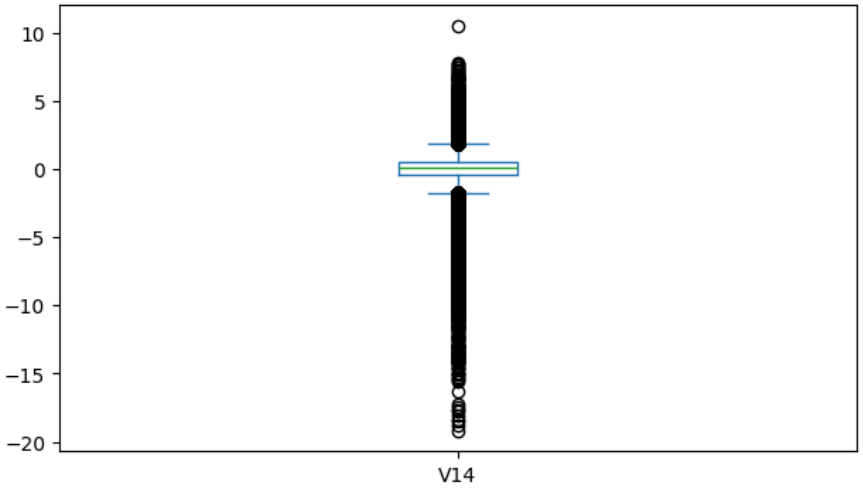

4
2
0
−2
−4
V11

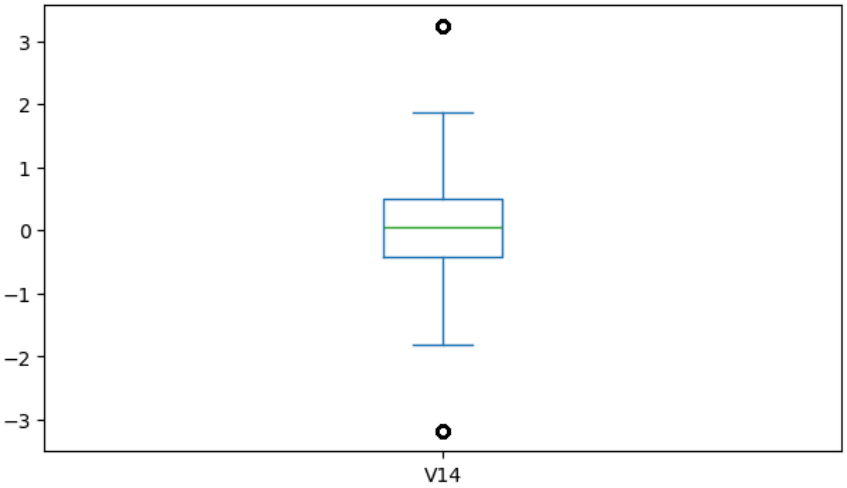

5
0
−5
−10
−15
−20
V12

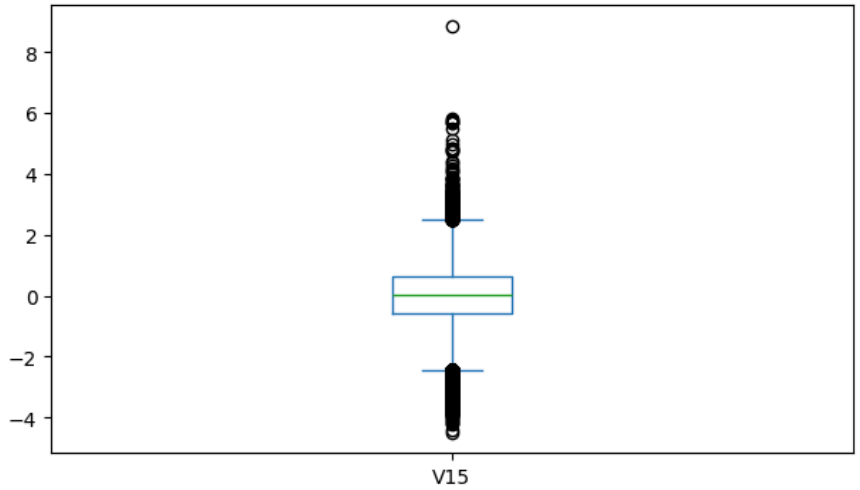

4
3
2
1
0
−1
−2
−3
V12

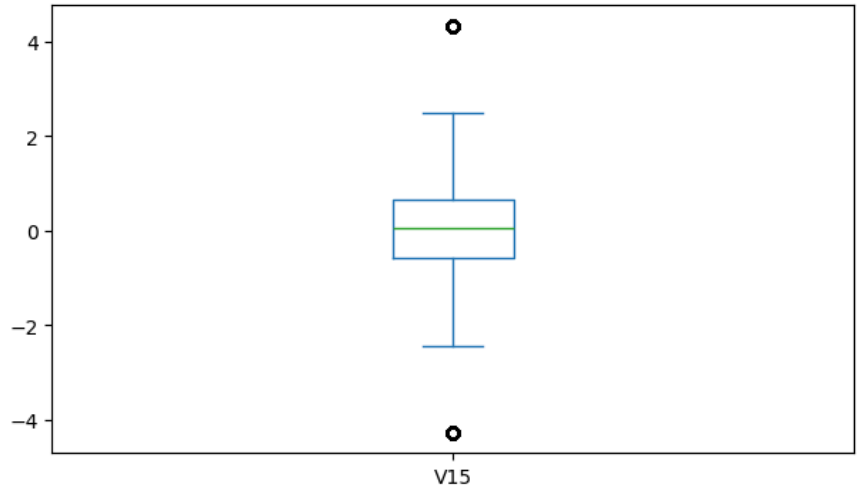

6
4
2
0
−2
−4
−6
V13

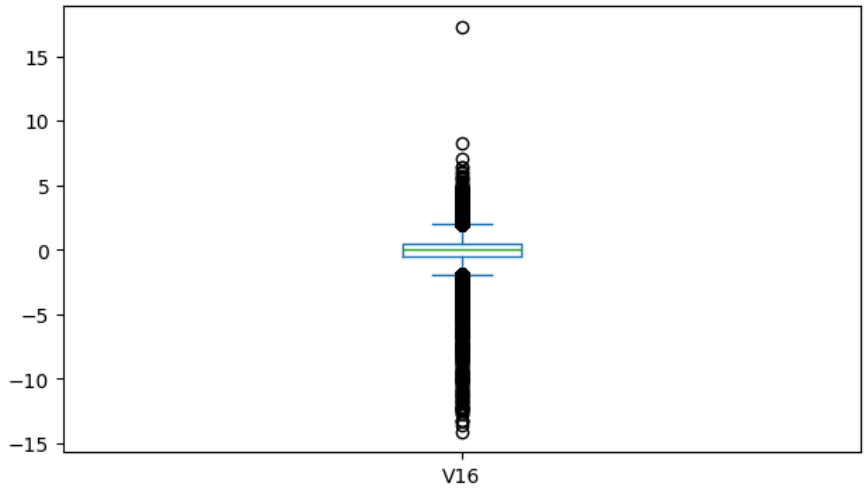

4
2
0
−2
−4
V13

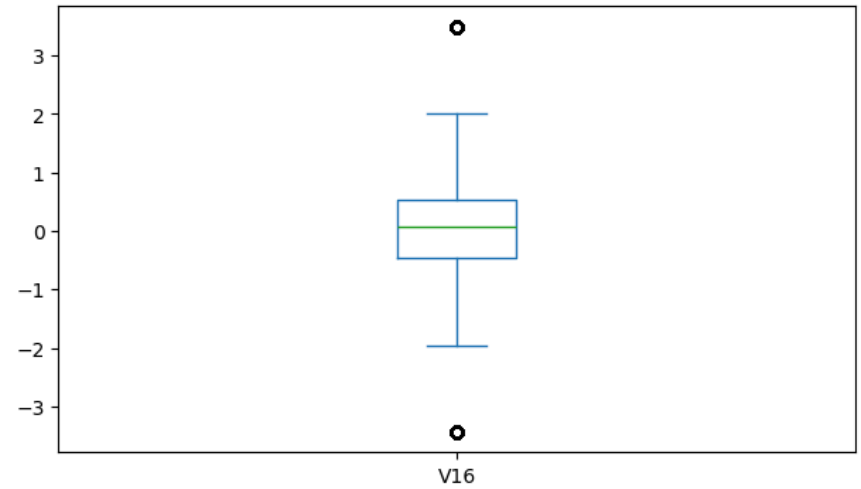

10
5
0
−5
−10
−15
−20
V14

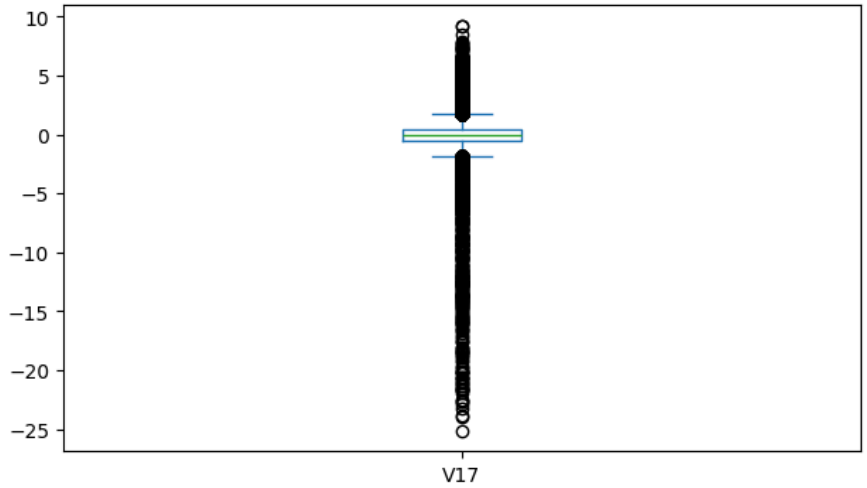

3
2
1
0
−1
−2
−3
V14

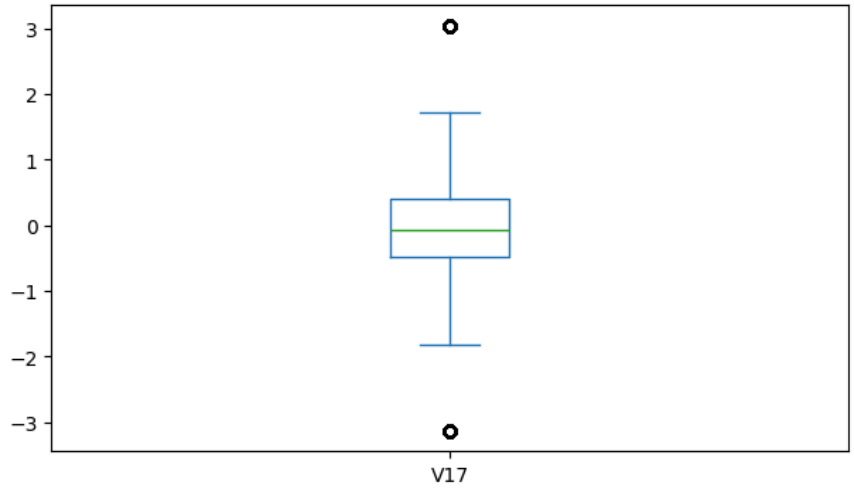

8
6
4
2
0
−2
−4
V15

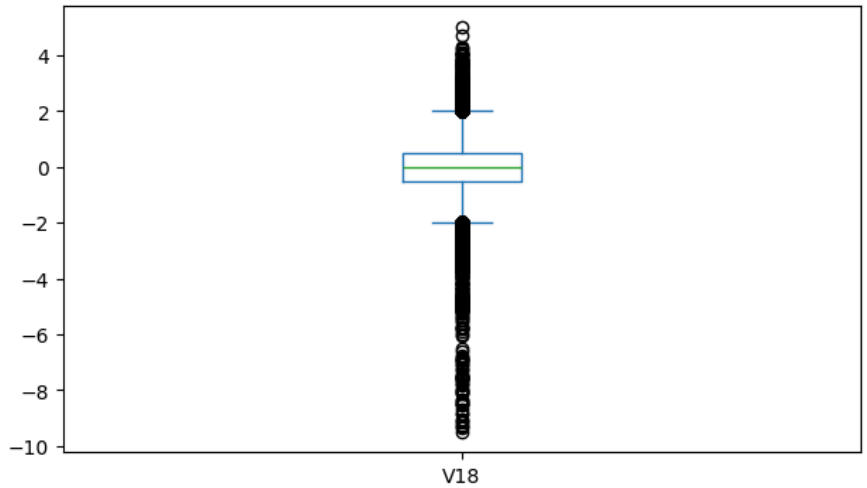

4
2
0
−2
−4
V15

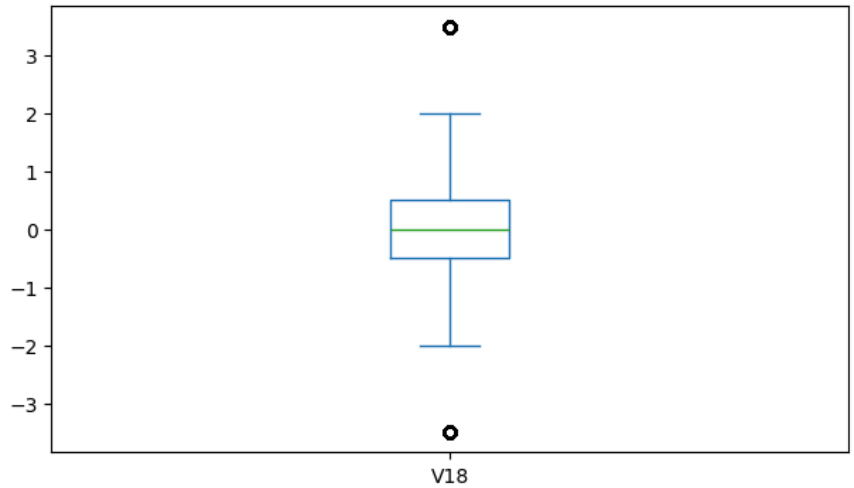

15
10
5
0
−5
−10
−15
V16

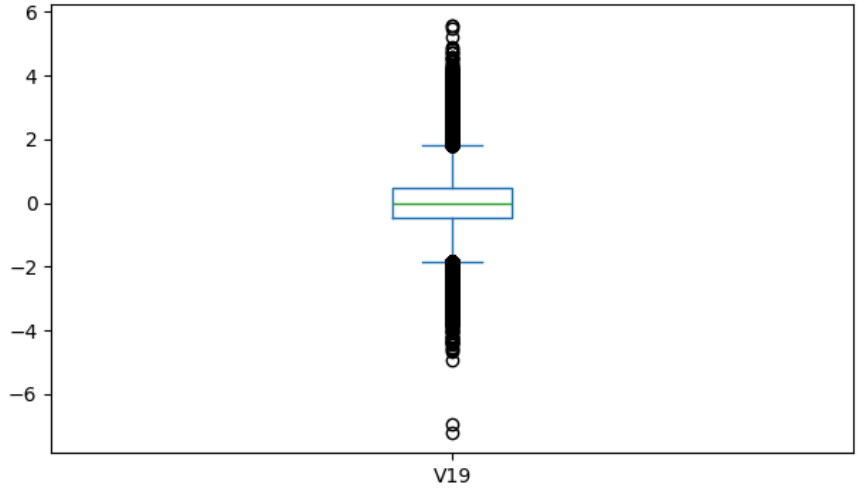

3
2
1
0
−1
−2
−3
V16

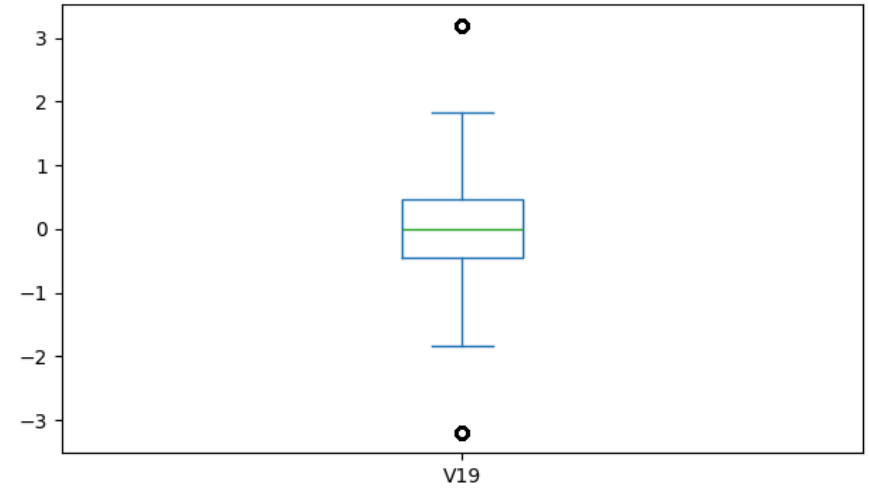

10
5
0
−5
−10
−15
−20
−25
V17

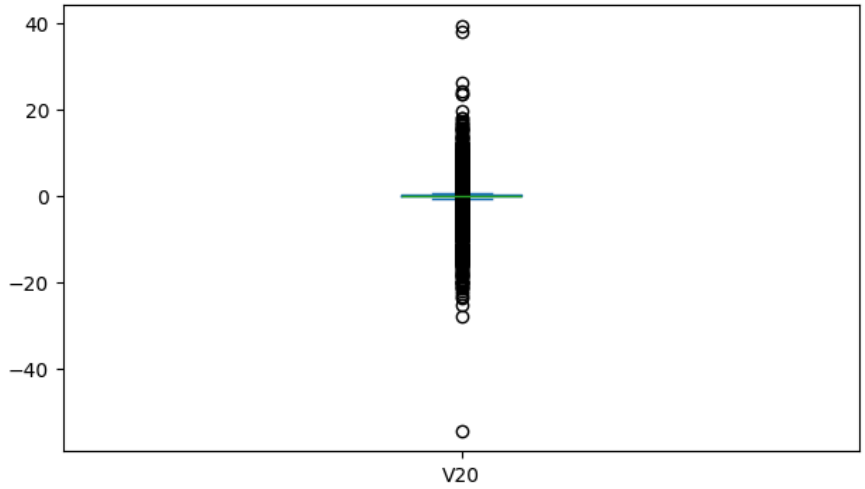

3
2
1
0
−1
−2
−3
V17

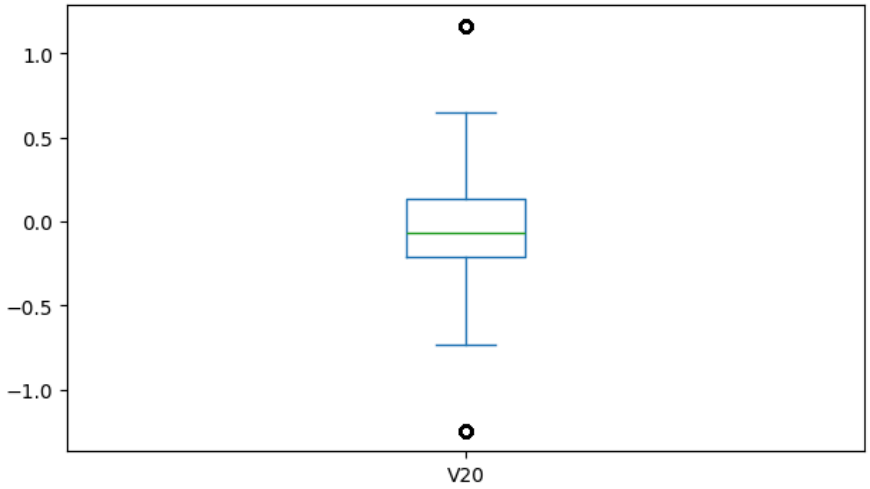

4
2
0
−2
−4
−6
−8
−10
V18

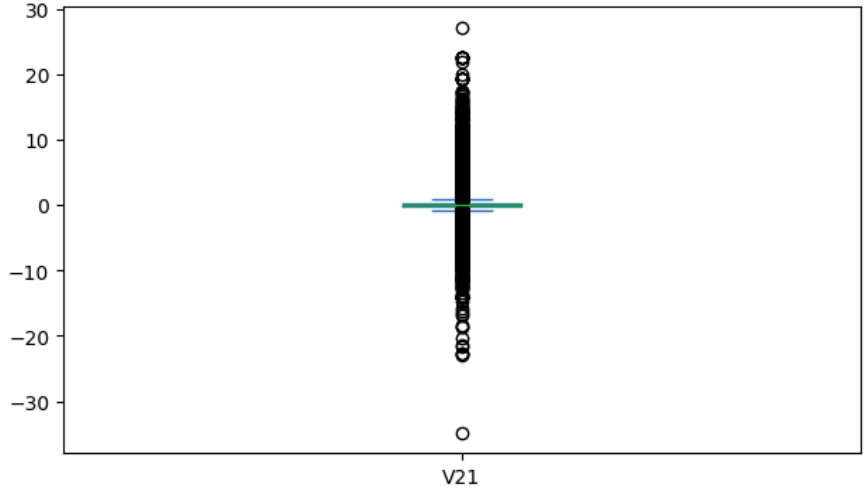

3
2
1
0
−1
−2
−3
V18

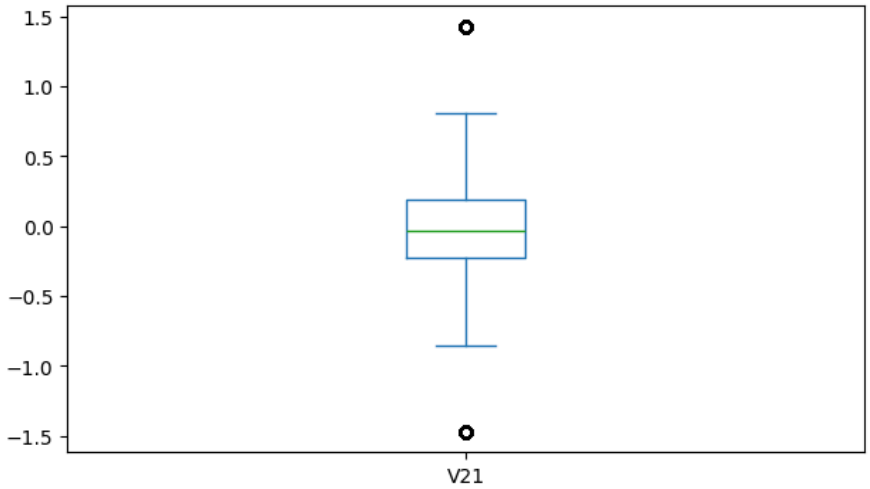

6
4
2
0
−2
−4
−6
V19

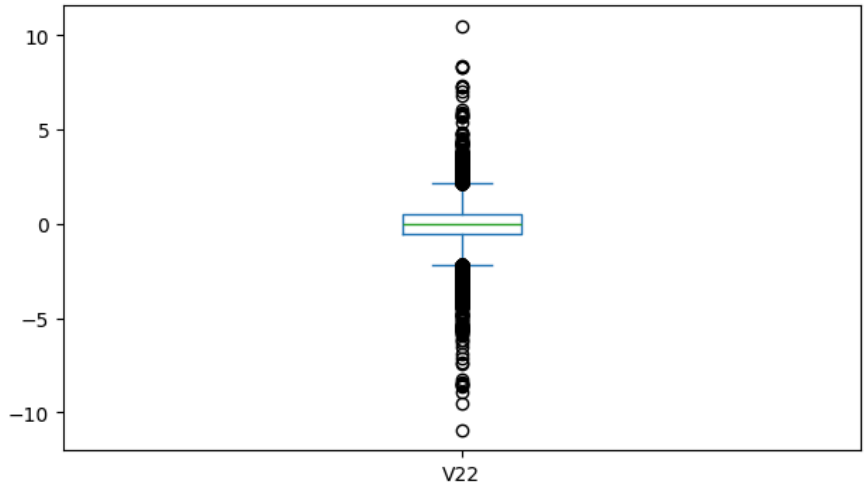

3
2
1
0
−1
−2
−3
V19

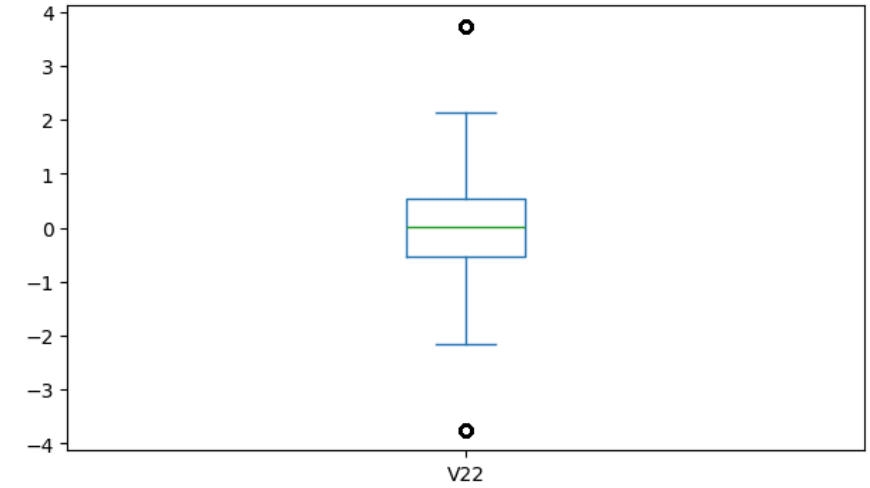

40
20
0
−20
−40
V20

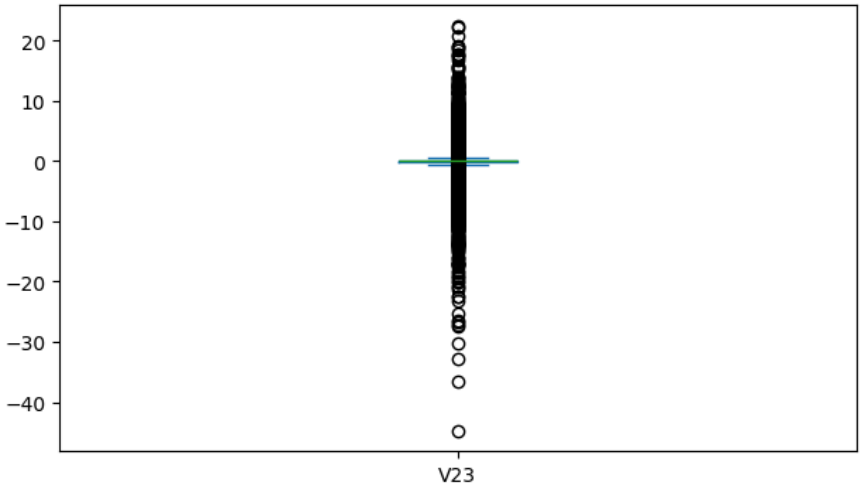

1.0
0.5
0.0
−0.5
−1.0
V20

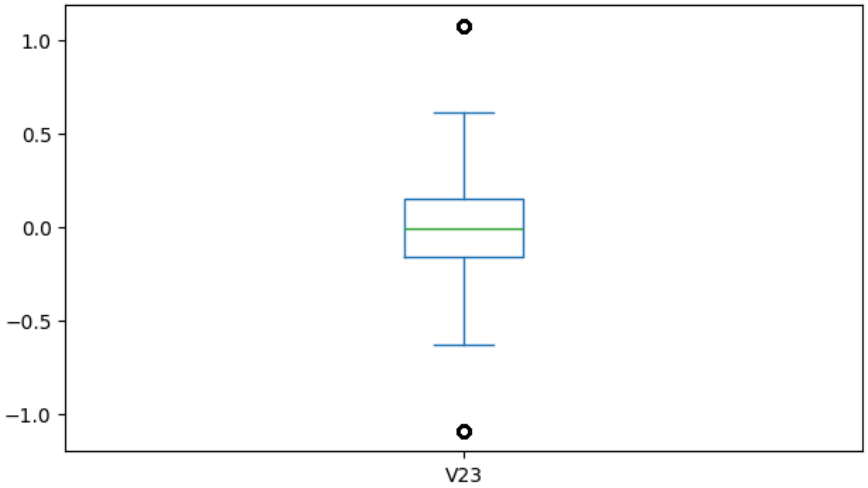

30
20
10
0
−10
−20
−30
V21

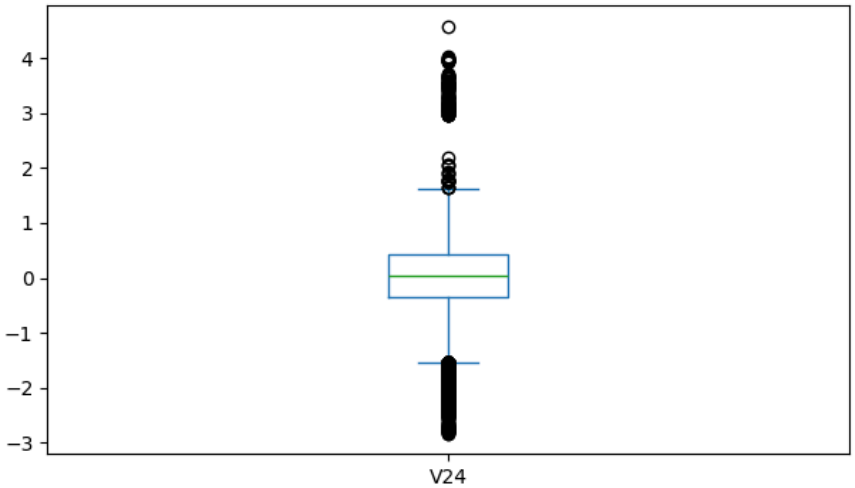

1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
V21

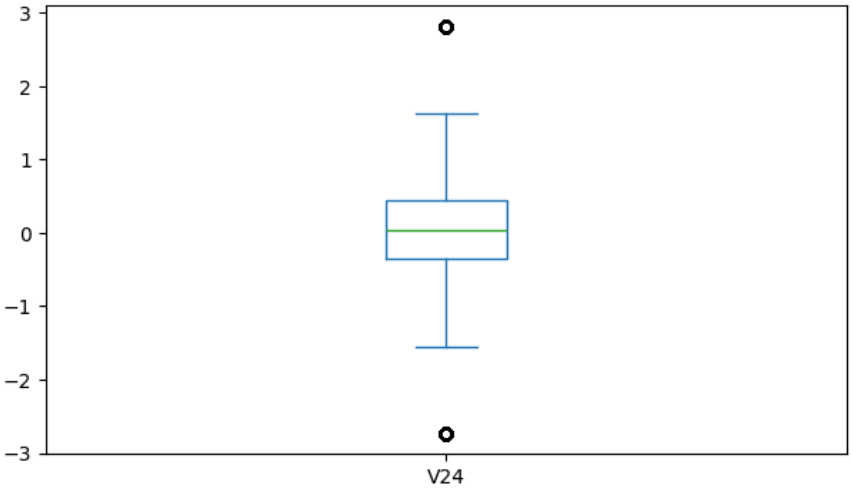

10
5
0
−5
−10
V22

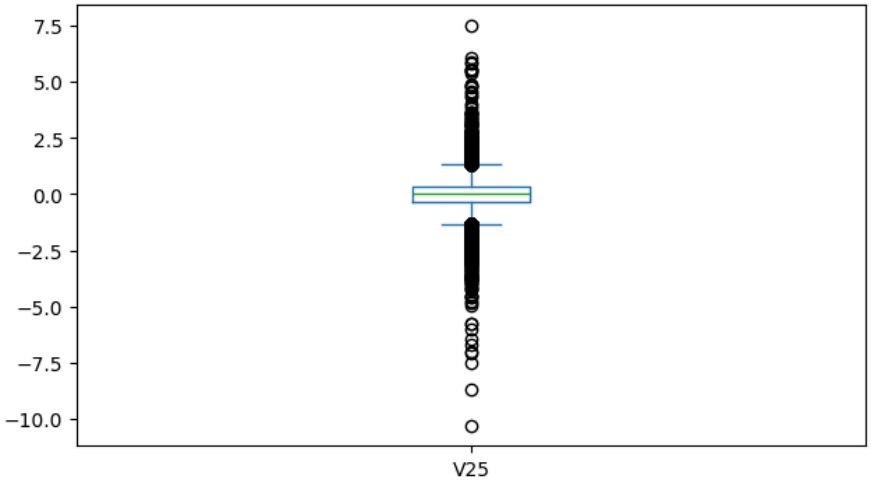

4
3
2
1
0
−1
−2
−3
−4
V22

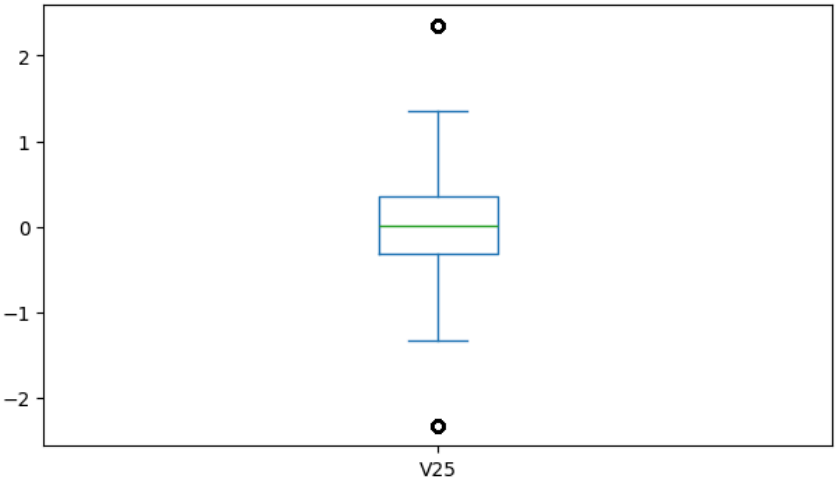

20
10
0
−10
−20
−30
−40
V23

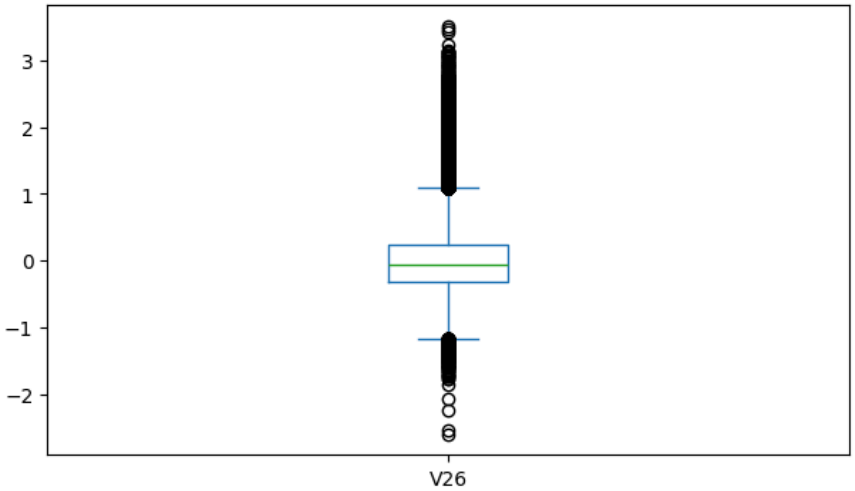

1.0
0.5
0.0
−0.5
−1.0
V23

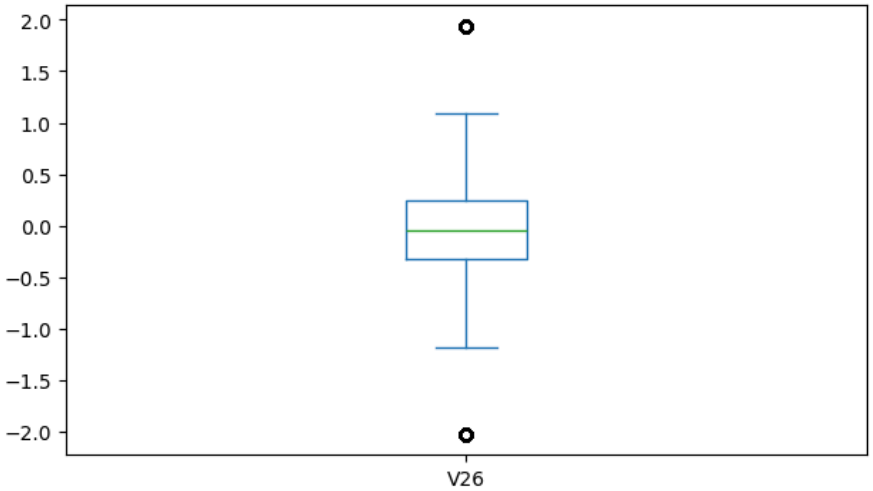

4
3
2
1
0
−1
−2
−3
V24

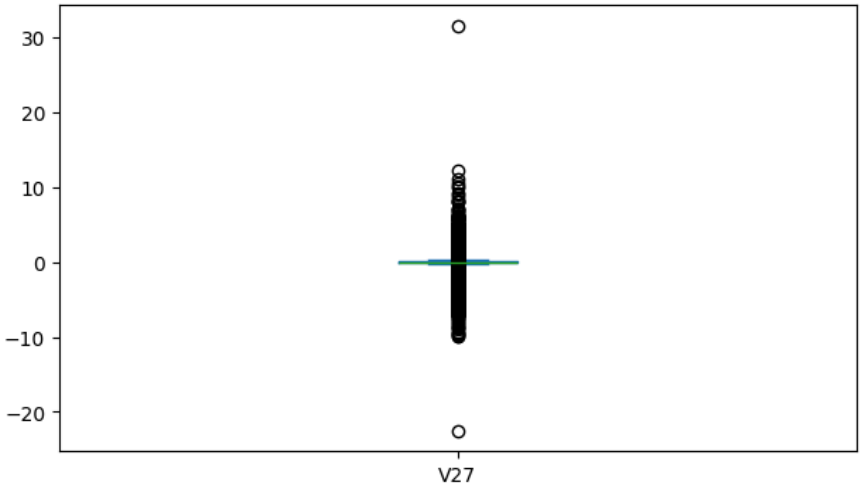

3
2
1
0
−1
−2
−3
V24

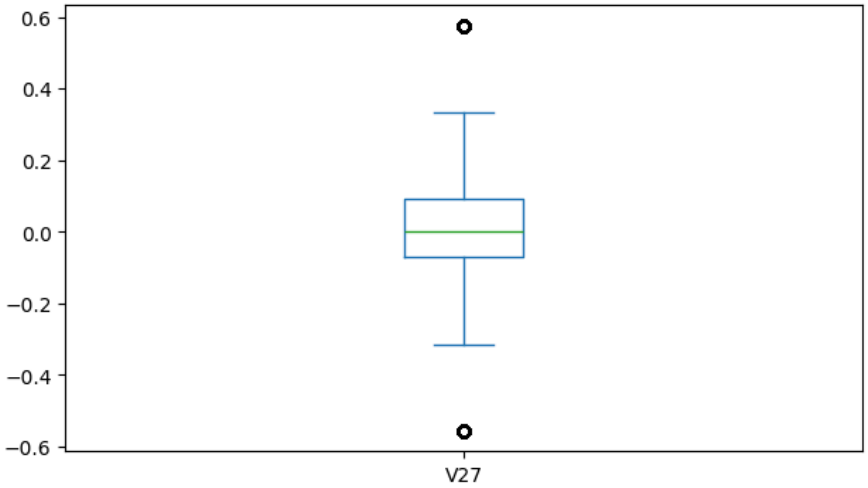

7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
V25

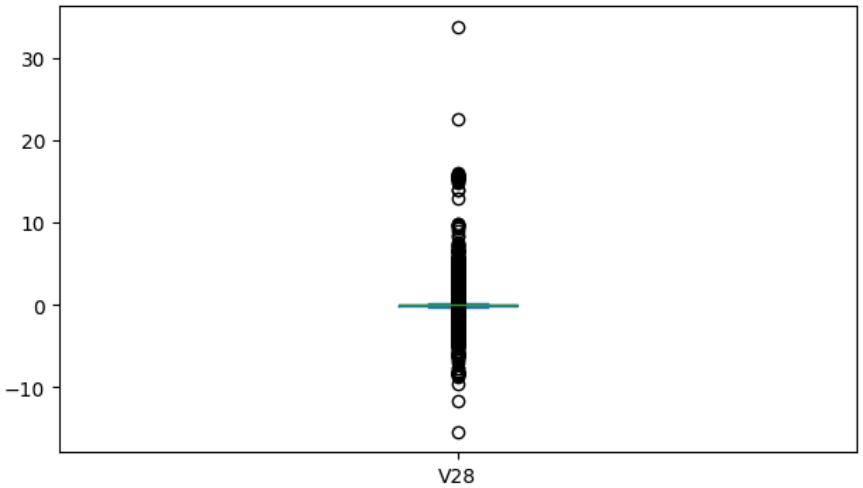

2
1
0
−1
−2
V25

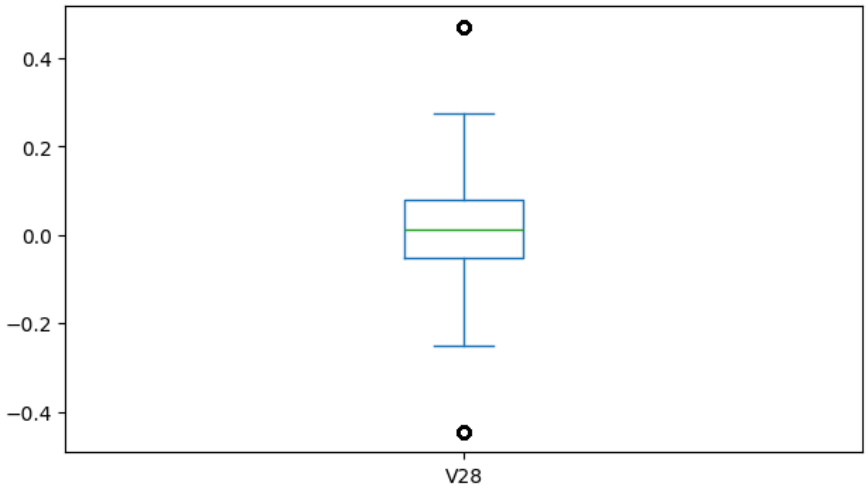

3
2
1
0
−1
−2
V26


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
V26









3. Feature Engineering

In [14]:
```
df['Time']
```

Out[14]:
```
0             0.0
1             0.0
2             1.0
3             1.0
4             2.0
             ...
284802    172786.0
284803    172787.0
284804    172788.0
284805    172788.0
284806    172792.0
Name: Time, Length: 283726, dtype: float64
```

In [15]:
```
#Create two columns for hour and day column from time column

# Convert time in seconds to hours (0 to 23 hours only)
df['Hour'] = ((df['Time']%(3600*24))//3600)

# Convert time in seconds to days (0 and 1 days only)
df['Day'] = df['Time']//(3600*24)

# Drop Time column
df = df.drop('Time',axis=1)
```

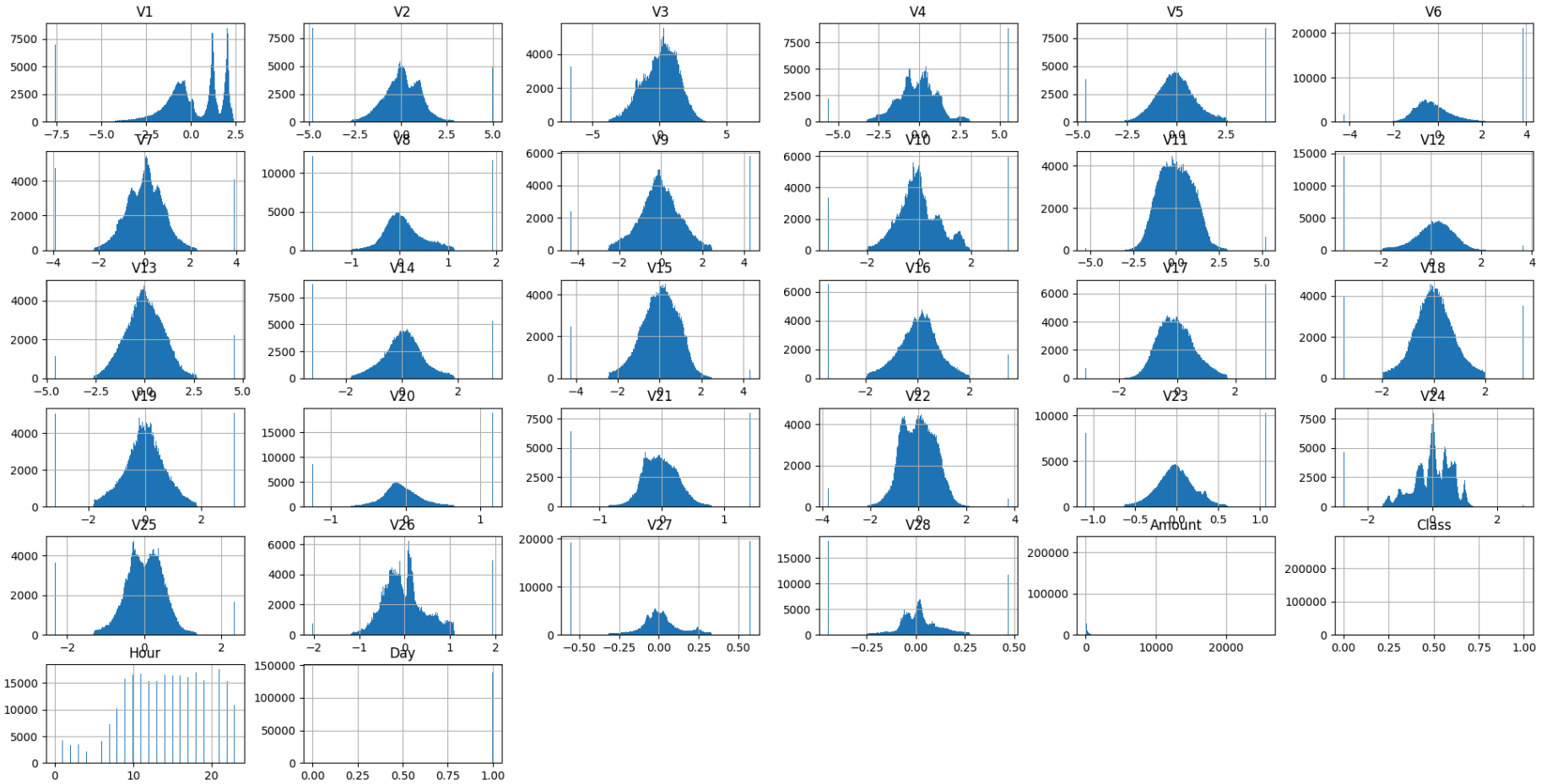
In [16]:
```
df.describe()
```

Out[16]:

	V1	V2	V3	V4	V5	V6	V7	V8	
count	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.000000	283726.0
mean	-0.002086	0.021258	-0.005236	0.024806	0.029934	0.027500	0.005745	0.066238	0.0
std	1.873219	1.419041	1.491115	1.588539	1.325419	1.367914	1.034625	0.657939	1.1
min	-7.612009	-4.802131	-6.639608	-5.619479	-4.595973	-4.266499	-3.921459	-1.812425	-4.3
25%	-0.915951	-0.600321	-0.889682	-0.850134	-0.689830	-0.769031	-0.552509	-0.208828	-0.6
50%	0.020384	0.063949	0.179963	-0.022248	-0.053468	-0.275168	0.040859	0.021898	-0.0
75%	1.316068	0.800283	1.026960	0.739647	0.612218	0.396792	0.570474	0.325704	0.5
max	2.454930	5.002093	6.776886	5.508991	4.518361	3.894261	3.939424	1.929300	4.3

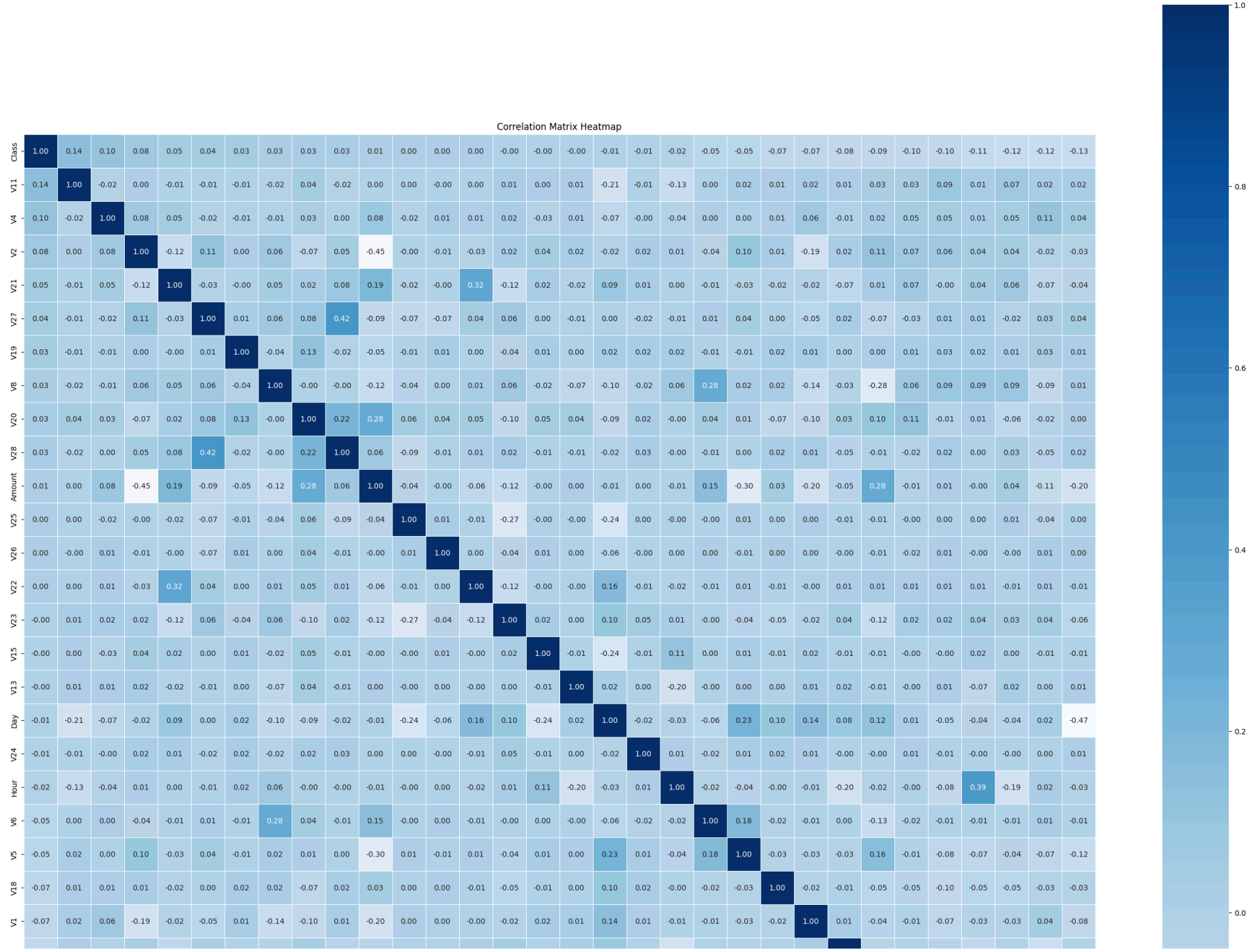
8 rows × 32 columns

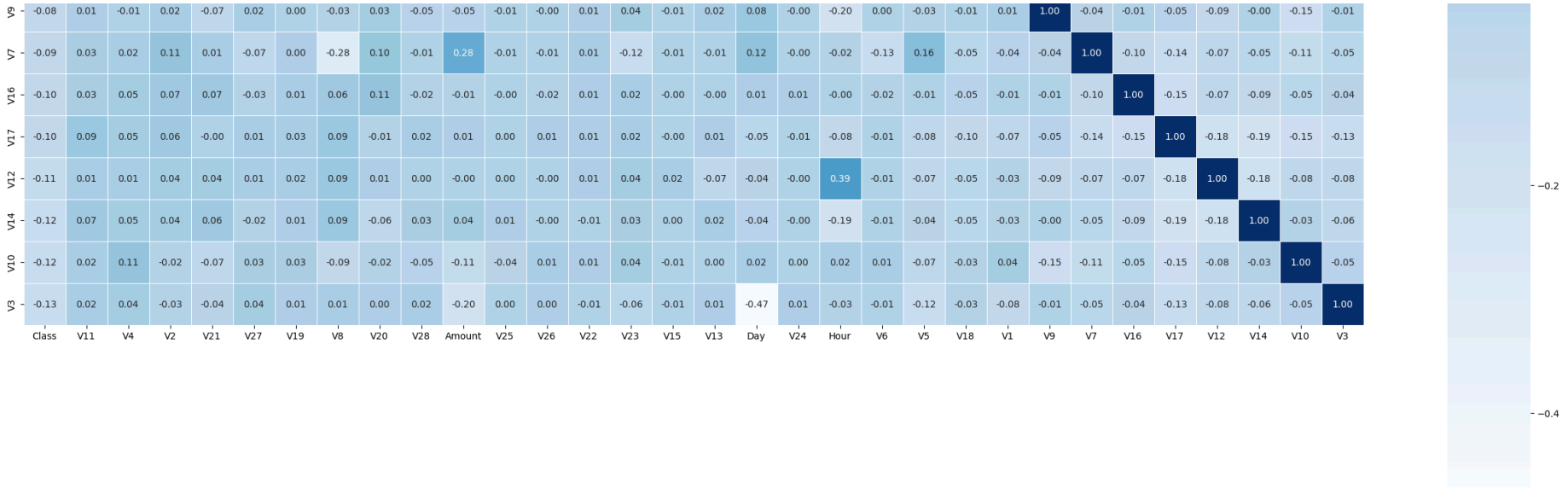
In [17]:
```
# Plot the histogram of each columns
df.hist(bins=250, figsize=(24,12))
plt.show()
```



In [18]:
```
# Compute the correlation matrix
corr_matrix =df.corr()

# Sort the correlation matrix with respect to the column of interest
sorted_correlation = corr_matrix['Class'].sort_values(ascending=False)

# # Extract the columns in order of their correlation with the column of interest
correlated_columns = sorted_correlation.index

# Create a subset of the correlation matrix with the sorted columns
subset_corr_matrix = corr_matrix.loc[correlated_columns, correlated_columns]
```

```python
# Set up the Matplotlib figure
plt.figure(figsize=(32, 32))

# Create a heatmap using Seaborn
sns.heatmap(subset_corr_matrix, annot=True, cmap='Blues', linewidths=0.5, fmt=".2f", square=True)

# Add a title
plt.title("Correlation Matrix Heatmap")

# Show the plot
plt.show()
```


Correlation Matrix Heatmap
Class V11 V4 V2 V21 V27 V19 V8 V20 V28 Amount V25 V26 V22 V23 V15 V13 Day V24 Hour V6 V5 V18 V1
1.0
0.8
0.6
0.4
0.2
0.0



4. Model training and evaluation

In [19]:
```
# Create features and target columns
X = df.drop('Class',axis=1)
y = df['Class']
```

In [20]:
```
# Split the data into train and test set
from sklearn.model_selection import train_test_split

random_state = 325

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,  random_state=random_state, stratify=y)
```

In [21]:
```
# Apply StandardScaler for train data
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [22]:
```
# Apply SMOTE technique to balance the training data
from imblearn.over_sampling import SMOTE
```

```
smote = SMOTE(random_state=random_state)
X_train, y_train = smote.fit_resample(X_train, y_train)
```

In [23]:
```
# Building and train a RandomForestClassifier model on the training data
from sklearn.ensemble import RandomForestClassifier

RF1 = RandomForestClassifier(n_jobs=-1,
                             random_state=random_state
                             )

RF1.fit(X_train,y_train)
```

Out[23]:
```
RandomForestClassifier(n_jobs=-1, random_state=325)
```

In [24]:
```
# Save and load the model to make predicitons
import joblib

joblib.dump(RF1,'RF1.joblib')

RF1 = joblib.load('RF1.joblib')
```

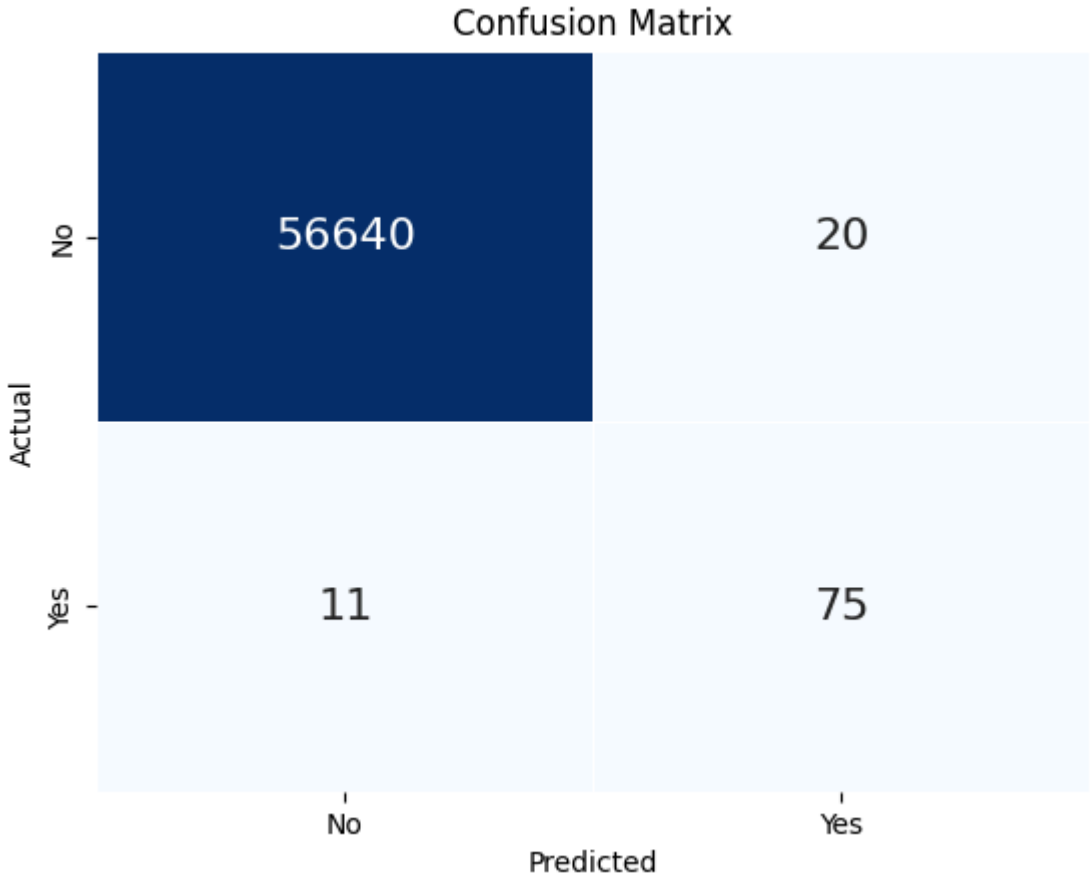
In [25]:
```
# Evaluate the Random Forest model by making predictions on test data
from sklearn.metrics import accuracy_score, f1_score, classification_report, confusion_matrix,balanced_accuracy_score

y_pred = RF1.predict(X_test)

print("Random Forest Evaluation:")
print(classification_report(y_pred,y_test))
```

```
Random Forest Evaluation:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56660
           1       0.79      0.87      0.83        86

    accuracy                           1.00     56746
   macro avg       0.89      0.94      0.91     56746
weighted avg       1.00      1.00      1.00     56746
```

In [26]:

```
# Calculate the balanced_accuracy score for the Random Forest model and plot the confusion matrix

print("The f1_score for the ML model is {} %".format(round(100*f1_score(y_pred,y_test,average='macro'),2)))

labels= ['No','Yes']
sns.heatmap(pd.DataFrame(confusion_matrix(y_pred,y_test)),annot=True,annot_kws = {'size':16}, fmt='d',
linewidths= 0.5 ,cmap = 'Blues' , cbar=False,xticklabels= labels, yticklabels= labels);
plt.title('Confusion Matrix')
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```

```
The f1_score for the ML model is 91.42 %
```



In [27]:
```
# The 5-fold cross validation score for the dataset

from sklearn.model_selection import cross_val_score

RF_scores = cross_val_score(RF1, X_train,y_train, cv=5,scoring='f1_macro')

def display_scores(scores):
    print("Scores:", scores.round(4))
    print("Mean:", round(scores.mean(),4))
    print("Standard Deviation:", round(scores.std(),4))

display_scores(RF_scores)
```

```
Scores: [0.9999 0.9999 0.9998 0.9999 0.9998]
Mean: 0.9999
Standard Deviation: 0.0
```

5. Hyperparameter tuning

In [28]:
```python
# Using RandomSearchCV to find the best hyperparameters
from sklearn.model_selection import RandomizedSearchCV

search_space = {'n_estimators':[100,500],
                'max_depth':[None,3],
                'max_leaf_nodes': [None,121],
                'min_samples_leaf': [1,2],
                'min_samples_split': [2,3]
               }

RF2 = RandomForestClassifier(n_jobs=-1,
                             random_state=random_state
                            )

RandomSearchCV_RF2 = RandomizedSearchCV(RF2,
                                        search_space,
                                        n_iter=30,
                                        cv=5,
                                        scoring='f1_macro',
                                        verbose=4)

RandomSearchCV_RF2.fit(X_train, y_train)

print('\n',RandomSearchCV_RF2.best_params_,'\n',RandomSearchCV_RF2.best_score_)
```

```
Fitting 5 folds for each of 30 candidates, totalling 150 fits
[CV 1/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 2/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 3/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 4/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.3min
[CV 5/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.3min
[CV 1/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 2/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 3/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 4/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 5/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 1/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 2/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 3/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 4/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 5/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 1/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.935 total tim
e= 2.8min
[CV 2/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 3/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 4/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 5/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.934 total tim
e= 2.8min
```

```
[CV 1/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.935 total tim
e=  33.3s
[CV 2/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.934 total tim
e=  34.6s
[CV 3/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.934 total tim
e=  34.8s
[CV 4/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.934 total tim
e=  33.3s
[CV 5/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.934 total tim
e=  34.7s
[CV 1/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.935 total ti
me= 2.8min
[CV 2/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 3/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 4/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 5/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 1/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=1.000 total
time=10.8min
[CV 2/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=1.000 total
time=10.7min
[CV 3/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=1.000 total
time=10.5min
[CV 4/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=1.000 total
time=10.7min
[CV 5/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=1.000 total
time=10.9min
[CV 1/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.935 total tim
e= 2.8min
[CV 2/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 3/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 4/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 5/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 1/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.993 total
```

```
time= 1.3min
[CV 2/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 3/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 4/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.993 total
time= 1.4min
[CV 5/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 1/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.992 total
time= 6.4min
[CV 2/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.993 total
time= 6.5min
[CV 3/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.993 total
time= 6.4min
[CV 4/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.993 total
time= 6.4min
[CV 5/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.993 total
time= 6.4min
[CV 1/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.935 total ti
me=  34.8s
[CV 2/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.934 total ti
me=  34.8s
[CV 3/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.934 total ti
me=  34.7s
[CV 4/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.934 total ti
me=  34.7s
[CV 5/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.934 total ti
me=  34.7s
[CV 1/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.935 total ti
me=  37.1s
[CV 2/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.934 total ti
me=  38.7s
[CV 3/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.934 total ti
me=  39.2s
[CV 4/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.934 total ti
me=  36.2s
[CV 5/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.934 total ti
me=  37.1s
[CV 1/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.935 total ti
me= 3.0min
```

```
[CV 2/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.934 total ti
me= 3.0min
[CV 3/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.934 total ti
me= 2.9min
[CV 4/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 5/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 1/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.935 total tim
e=  34.7s
[CV 2/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.934 total tim
e=  34.8s
[CV 3/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.934 total tim
e=  33.6s
[CV 4/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.934 total tim
e=  35.2s
[CV 5/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.934 total tim
e=  38.1s
[CV 1/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 2/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 3/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 4/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 5/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 1/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.992 total
time= 6.4min
[CV 2/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.993 total
time= 6.5min
[CV 3/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.993 total
time= 6.4min
[CV 4/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.993 total
time= 6.4min
[CV 5/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=0.993 total
time= 6.4min
[CV 1/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=1.000 total
time=10.9min
[CV 2/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=1.000 total
```

```
time=10.7min
[CV 3/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=1.000 total
time=10.5min
[CV 4/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=1.000 total
time=10.9min
[CV 5/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=1.000 total
time=10.8min
[CV 1/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.935 total tim
e=  34.9s
[CV 2/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.934 total tim
e=  34.7s
[CV 3/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.934 total tim
e=  34.7s
[CV 4/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.934 total tim
e=  34.7s
[CV 5/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.934 total tim
e=  34.6s
[CV 1/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.992 total
time= 6.4min
[CV 2/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.993 total
time= 6.5min
[CV 3/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.993 total
time= 6.4min
[CV 4/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.993 total
time= 6.4min
[CV 5/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.993 total
time= 6.5min
[CV 1/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 2/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.4min
[CV 3/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.4min
[CV 4/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 5/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 1/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.935 total ti
me= 2.8min
[CV 2/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.934 total ti
me= 2.8min
```

```
[CV 3/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 4/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 5/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 1/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.935 total tim
e=  35.0s
[CV 2/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.934 total tim
e=  34.6s
[CV 3/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.934 total tim
e=  34.9s
[CV 4/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.934 total tim
e=  34.9s
[CV 5/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.934 total tim
e=  34.8s
[CV 1/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 2/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 3/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 4/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 5/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=1.000 total
time= 2.2min
[CV 1/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=1.000 total
time=10.9min
[CV 2/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=1.000 total
time=10.8min
[CV 3/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=1.000 total
time=10.5min
[CV 4/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=1.000 total
time=11.2min
[CV 5/5] END max_depth=None, max_leaf_nodes=None, min_samples_leaf=1, min_samples_split=2, n_estimators=500;, score=1.000 total
time=11.3min
[CV 1/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.935 total ti
me=  35.3s
[CV 2/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.934 total ti
me=  35.0s
[CV 3/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.934 total ti
```

```
me=  33.8s
[CV 4/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.934 total ti
me=  35.3s
[CV 5/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=100;, score=0.934 total ti
me=  35.0s
[CV 1/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.935 total tim
e= 2.8min
[CV 2/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 3/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 4/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 5/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 1/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.993 total
time= 1.4min
[CV 2/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.993 total
time= 1.4min
[CV 3/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.993 total
time= 1.4min
[CV 4/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.993 total
time= 1.4min
[CV 5/5] END max_depth=None, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=2, n_estimators=100;, score=0.993 total
time= 1.3min
[CV 1/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.935 total ti
me=  35.2s
[CV 2/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.934 total ti
me=  35.1s
[CV 3/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.934 total ti
me=  35.1s
[CV 4/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.934 total ti
me=  35.0s
[CV 5/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=2, n_estimators=100;, score=0.934 total ti
me=  35.1s
[CV 1/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.935 total ti
me= 2.8min
[CV 2/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 3/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.934 total ti
me= 2.8min
```

```
[CV 4/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 5/5] END max_depth=3, max_leaf_nodes=None, min_samples_leaf=2, min_samples_split=3, n_estimators=500;, score=0.934 total ti
me= 2.8min
[CV 1/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.935 total tim
e= 2.8min
[CV 2/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 3/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 4/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.934 total tim
e= 2.8min
[CV 5/5] END max_depth=3, max_leaf_nodes=121, min_samples_leaf=1, min_samples_split=3, n_estimators=500;, score=0.934 total tim
e= 2.8min

{'n_estimators': 500, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_leaf_nodes': None, 'max_depth': None}
0.9998654027299141
```

In [29]:
```
RandomSearchCV_RF2.best_params_
```

Out[29]:
```
{'n_estimators': 500,
 'min_samples_split': 2,
 'min_samples_leaf': 1,
 'max_leaf_nodes': None,
 'max_depth': None}
```

In [29]:
```
RF2 = RandomForestClassifier(n_jobs=-1,
                             random_state=random_state,
                              n_estimators=500,
                              min_samples_split=2,
                              min_samples_leaf=1
                             )

RF2.fit(X_train,y_train)
```

Out[29]:

RandomForestClassifier

```
RandomForestClassifier(n_estimators=500, n_jobs=-1, random_state=325)
```

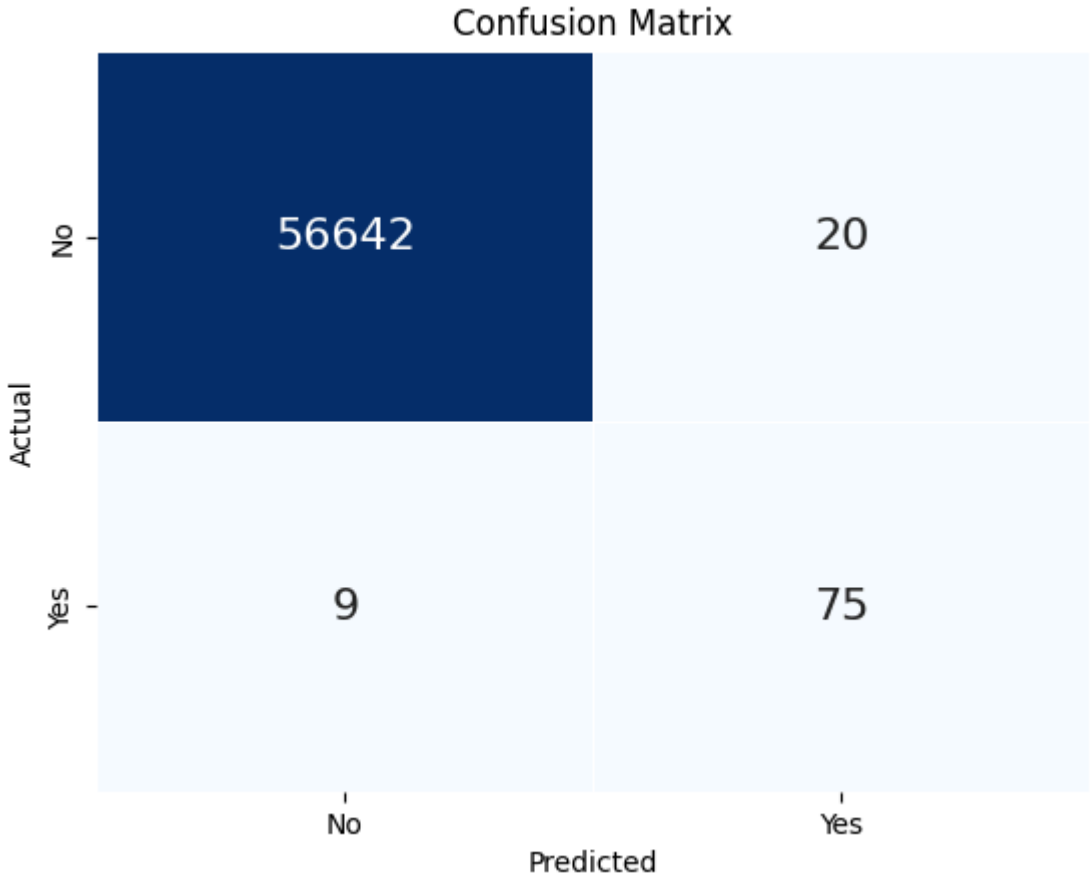
In [30]:
```
# Save and load the model to make predicitons

joblib.dump(RF2,'RF2.joblib')

RF2 = joblib.load('RF2.joblib')
```

In [31]:
```
# Evaluate the Random Forest model by making predictions on test data

y_pred = RF2.predict(X_test)

print("Random Forest Evaluation:")
print(classification_report(y_pred,y_test))
```

```
Random Forest Evaluation:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56662
           1       0.79      0.89      0.84        84

    accuracy                           1.00     56746
   macro avg       0.89      0.95      0.92     56746
weighted avg       1.00      1.00      1.00     56746
```

In [32]:
```
# Calculate the balanced_accuracy score for the Random Forest model and plot the confusion matrix

print("The f1_score for the ML model is {} %".format(round(100*f1_score(y_pred,y_test,average='macro'),2)))

labels= ['No','Yes']
sns.heatmap(pd.DataFrame(confusion_matrix(y_pred,y_test)),annot=True,annot_kws = {'size':16}, fmt='d',
linewidths= 0.5 ,cmap = 'Blues' , cbar=False,xticklabels= labels, yticklabels= labels);
plt.title('Confusion Matrix')
plt.ylabel('Actual')
plt.xlabel('Predicted')
plt.show()
```

```
The f1_score for the ML model is 91.89 %
```



In [35]:
```
# The 5-fold cross validation score for the dataset

from sklearn.model_selection import cross_val_score

RF_scores = cross_val_score(RF2, X_train,y_train, cv=5,scoring='f1_macro')

def display_scores(scores):
    print("Scores:", scores.round(4))
    print("Mean:", round(scores.mean(),4))
    print("Standard Deviation:", round(scores.std(),4))

display_scores(RF_scores)
```

```
Scores: [0.9999 0.9999 0.9998 0.9999 0.9998]
Mean: 0.9999
Standard Deviation: 0.0
```

In [33]:
```
# Compare the F1 score for old and new model
y_pred = RF1.predict(np.array(X_test))
print("The f1_score for the ML model 1 is {} %".format(round(100*f1_score(y_pred,y_test,average='macro'),2)))
y_pred = RF2.predict(np.array(X_test))
print("The f1_score for the ML model 2 is {} %".format(round(100*f1_score(y_pred,y_test,average='macro'),2)))
```

```
The f1_score for the ML model 1 is 91.42 %
The f1_score for the ML model 2 is 91.89 %
```

We see an improvment with the hyperparameter tuning

In [ ]:

In [ ]:

In [ ]: