

```
In [1]: import numpy as np
import pandas as pd
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import f1_score

import matplotlib.pyplot as plt
%matplotlib inline
```

```
In [2]: df=pd.read_csv('prefinaloutput.csv')
```

```
In [3]: df.head(50)
```

Out[3]:

	amount	oldbalanceOrg	oldbalanceDest	isFraud	isFlaggedFraud for Operations	Time	Instrument or Mode_CREDIT- CARD	N
0	0.000121	0.000864	0.000000	0	0	0.939583	1	
1	0.000930	0.001620	0.000000	0	1	0.918056	1	
2	0.055328	0.019373	0.358175	0	1	0.203472	0	
3	0.013412	0.552665	0.846995	0	1	0.530556	0	
4	0.000895	0.001013	0.000000	0	1	0.322917	1	
5	0.000243	0.000427	0.000000	0	0	0.354167	1	
6	0.001011	0.004672	0.000000	0	1	0.791667	1	
7	0.000022	0.007654	0.000000	0	0	0.984722	1	
8	0.010432	0.019373	0.010583	0	1	0.386806	0	
9	0.062700	0.019373	0.247145	0	1	0.856250	0	
10	0.018434	0.245232	0.508086	0	1	0.461806	0	
11	0.008400	0.325994	1.000000	0	1	0.483333	0	
12	0.011435	0.019373	0.023937	0	1	0.268056	0	
13	0.014191	0.019373	0.568306	0	1	0.084722	0	
14	0.002054	0.000939	0.000000	0	1	0.593750	1	
15	0.002112	0.001364	0.005596	0	1	0.695139	0	
16	0.007724	0.000077	0.025762	0	1	0.904167	0	
17	0.000008	0.001629	0.000000	0	0	0.293750	1	
18	0.000783	0.001140	0.000000	0	1	0.611806	1	
19	0.027621	0.531522	0.017031	0	1	0.856944	0	
20	0.000504	0.001821	0.000000	0	1	0.457639	1	
21	0.001168	0.000076	0.000000	0	1	0.024306	1	
22	0.012354	0.945736	0.013058	0	1	0.551389	0	
23	0.020224	0.527536	0.013042	0	1	0.526389	0	
24	0.002420	0.003227	0.000000	0	1	0.290278	1	
25	0.000698	0.000794	0.000000	0	1	0.600000	1	
26	0.000287	0.000011	0.000000	0	0	0.132639	1	
27	0.005498	0.001646	0.002396	0	1	0.340972	0	
28	0.026164	0.019373	0.325268	0	1	0.960417	0	
29	0.001251	0.019373	0.000000	0	1	0.002083	1	
30	0.003738	0.001610	0.003891	0	1	0.276389	0	

	amount	oldbalanceOrg	oldbalanceDest	isFraud	isFlaggedFraud for Operations	Time	Instrument or Mode_CREDIT- CARD	N
31	0.052143	0.019373	0.085251	0	1	0.809028	0	
32	0.000403	0.000463	0.000000	0	0	0.370833	1	
33	0.000019	0.002400	0.000000	0	0	0.174306	1	
34	0.000048	0.000168	0.000000	0	0	0.689583	1	
35	0.002425	0.000465	0.000000	0	1	0.918056	1	
36	0.011753	0.687853	0.016835	0	1	0.538889	0	
37	0.023077	0.706116	0.026484	0	1	0.475000	0	
38	0.000336	0.000693	0.000000	0	0	0.450694	1	
39	0.003391	0.017888	0.562842	0	1	0.631250	0	
40	0.001476	0.019373	0.000000	0	1	0.031250	1	
41	0.011776	0.626915	0.014575	0	1	0.863889	0	
42	0.005056	0.000004	0.000000	0	1	0.644444	0	
43	0.005074	0.019373	0.000000	0	1	0.918750	1	
44	0.038094	0.194751	0.253290	0	1	0.519444	0	
45	0.000013	0.000020	0.000000	0	0	0.230556	1	
46	0.000403	0.002252	0.000000	0	0	0.678472	1	
47	0.002363	0.354576	0.016878	0	1	0.015278	0	
48	0.017088	0.002429	0.000000	0	1	0.227083	0	
49	0.023968	0.013110	0.573770	0	1	0.177778	0	

```
In [4]: X=df.drop('isFraud',axis=1)
```

```
In [5]: X
```

Out[5]:

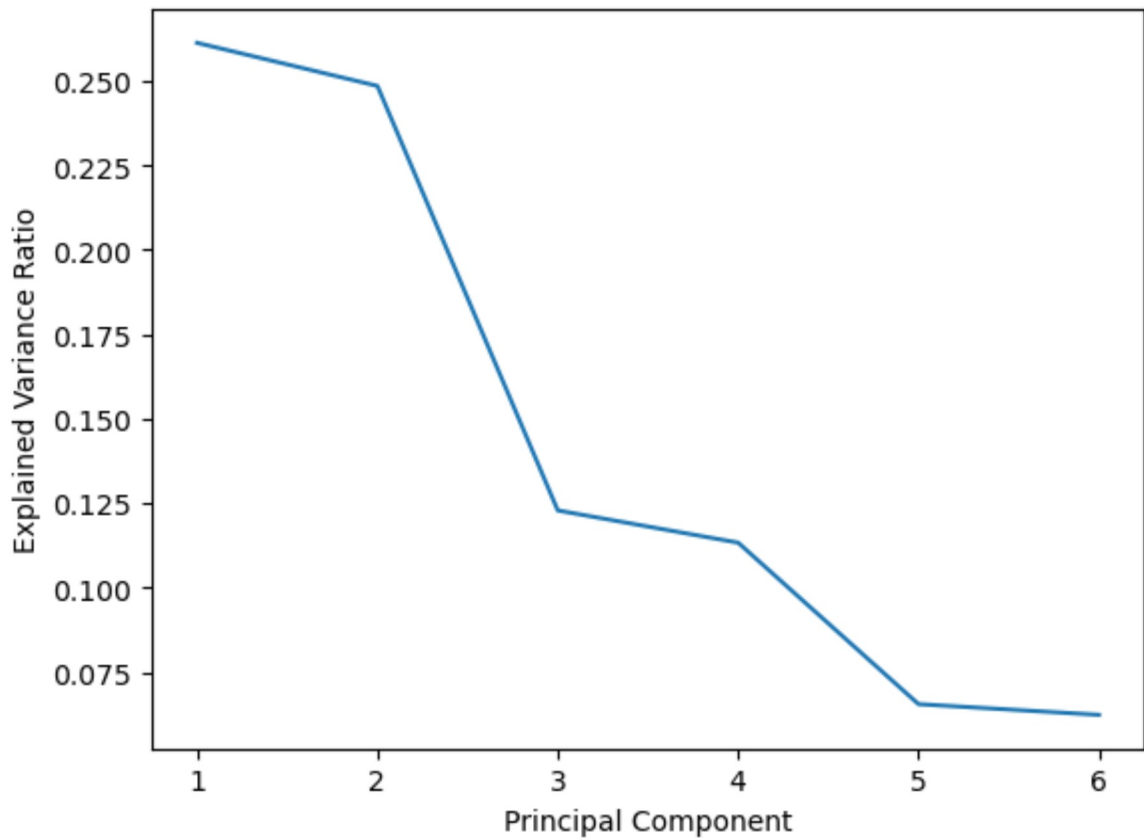
	amount	oldbalanceOrg	oldbalanceDest	isFlaggedFraud for Operations	Time	Instrument or Mode_CREDIT- CARD	Instrun Mode_DE C
0	0.000121	0.000864	0.000000	0	0.939583	1	
1	0.000930	0.001620	0.000000	1	0.918056	1	
2	0.055328	0.019373	0.358175	1	0.203472	0	
3	0.013412	0.552665	0.846995	1	0.530556	0	
4	0.000895	0.001013	0.000000	1	0.322917	1	
...
994	0.000877	0.004998	0.000000	1	0.056250	1	
995	0.013326	0.193793	0.003701	1	0.050694	0	
996	0.002676	0.002068	0.005572	1	0.995833	0	
997	0.003723	0.010631	0.298527	1	0.349306	0	
998	0.006072	0.004700	0.000000	1	0.605556	0	

999 rows × 17 columns

In [6]: `y=df['isFraud']`In [7]: `from sklearn.decomposition import PCA
pca = PCA(n_components=6)`In [8]: `pca.fit(X)
X_pca = pca.transform(X)
X_pca`

Out[8]: array([[-0.96201235, -0.47023336, -0.55869341, -0.54528875, 0.56520966,
 -0.14989069],
 [-0.71830703, -0.54848174, 0.18092689, 0.42231408, -0.14996667,
 -0.00502351],
 [0.93026908, -0.8686877 , 0.73396292, 0.40699991, -0.07444297,
 1.22669418],
 ...,
 [1.00081662, 1.12656688, 0.19373739, 0.53937436, 0.14922751,
 -0.37633214],
 [0.57540244, -0.80854093, 0.4119488 , 0.4919727 , 0.01053284,
 -0.23827955],
 [1.32126537, 1.06403993, 0.20211662, 0.47913645, 0.45485953,
 0.98020457]])

In [9]: `plt.plot(range(1, pca.n_components_ + 1), pca.explained_variance_ratio_)
plt.xlabel('Principal Component')
plt.ylabel('Explained Variance Ratio')
plt.show()`



```
In [10]: X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size = 0.
svclassifier = SVC(C=1000.0, kernel='poly', gamma= 1, degree=2)
svclassifier.fit(X_train, y_train)
y_pred = svclassifier.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

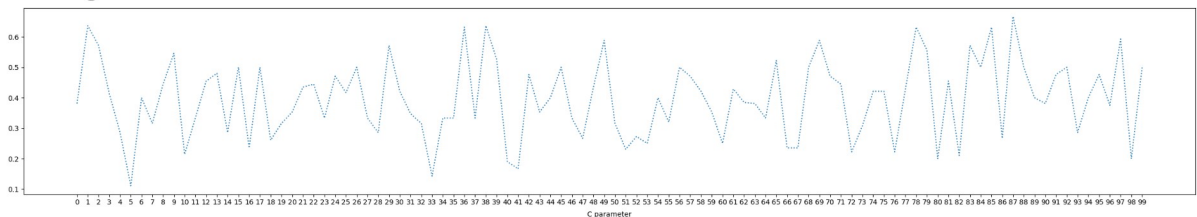
```
[[237  3]
 [  5  5]]
```

	precision	recall	f1-score	support
0	0.98	0.99	0.98	240
1	0.62	0.50	0.56	10
accuracy			0.97	250
macro avg	0.80	0.74	0.77	250
weighted avg	0.97	0.97	0.97	250

```
In [13]: cresult=[]
ci=[]
for j in range(50):
    X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size
    svclassifier = SVC(C=1000.0, kernel='poly', gamma= 1, degree=2)
    svclassifier.fit(X_train, y_train)
    y_pred = svclassifier.predict(X_test)
    cresult.append(f1_score(y_test, y_pred))
    ci.append(j)

print("average of f1 score "+str(np.mean(cresult)))
plt.figure().set_figwidth(30)
plt.plot(cresult, linestyle = 'dotted')
plt.xticks(range(len(ci)), ci)
plt.xlabel("C parameter", labelpad=7)
plt.show()
```

average of f1 score 0.3966412894178722



```
In [16]: X_train, X_test, y_train, y_test = train_test_split(X_pca, y, stratify=y, test_size
svclassifier = SVC(C=1000.0, kernel='poly', gamma= 1, degree=2)
svclassifier.fit(X_train, y_train)
y_pred = svclassifier.predict(X_test)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[240  0]
 [ 10  0]]
```

	precision	recall	f1-score	support
0	0.96	1.00	0.98	240
1	0.00	0.00	0.00	10
accuracy			0.96	250
macro avg	0.48	0.50	0.49	250
weighted avg	0.92	0.96	0.94	250

C:\Users\hfcpa\anaconda3\lib\site-packages\sklearn\metrics_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.

```
_warn_prf(average, modifier, msg_start, len(result))
```

C:\Users\hfcpa\anaconda3\lib\site-packages\sklearn\metrics_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.

```
_warn_prf(average, modifier, msg_start, len(result))
```

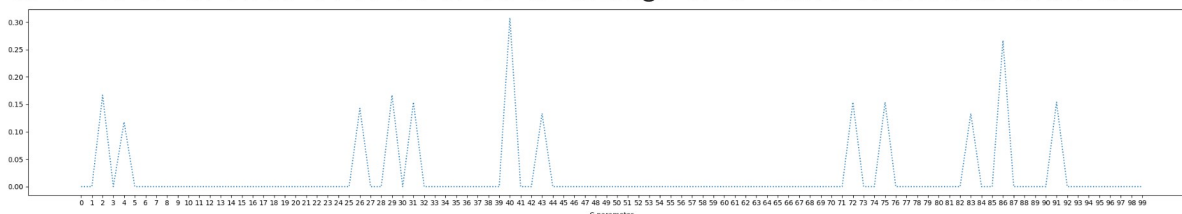
C:\Users\hfcpa\anaconda3\lib\site-packages\sklearn\metrics_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.

```
_warn_prf(average, modifier, msg_start, len(result))
```

```
In [17]: cresult=[]
ci=[]
for j in range(50):
    X_train, X_test, y_train, y_test = train_test_split(X_pca, y, stratify=y, test_s
    svclassifier = SVC(C=1000.0, kernel='poly', gamma= 1, degree=2)
    svclassifier.fit(X_train, y_train)
    print(j, end=" ")
    y_pred = svclassifier.predict(X_test)
    cresult.append(f1_score(y_test, y_pred))
    ci.append(j)

print("average of f1 score "+str(np.mean(cresult)))
plt.figure().set_figwidth(30)
plt.plot(cresult, linestyle = 'dotted')
plt.xticks(range(len(ci)), ci)
plt.xlabel("C parameter", labelpad=7)
plt.show()
```

```
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58
59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86
87 88 89 90 91 92 93 94 95 96 97 98 99 average of f1 score 0.020502477914242617
```



In []: