

Experiment - 9

Student Name: Aniket Kumar

Branch: CSE

Semester: 5th Semester

Subject Name: Machine Learning Lab

UID: 20BCS5306

Section/Group: 703-B

Subject Code: 20CSP-317

Aim: Implementation of Principal component Analysis.

Objective: To prepare a model with Principal component Analysis.

Data Set Chosen: Principal component Analysis

Result and output:

Implementation of Principal component Analysis

```
In [1]: import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

```
dataset = pd.read_csv('Wine.csv')
dataset.head()
```

Out[1]:

	Alcohol	Malic_Acid	Ash	Ash_Alcanity	Magnesium	Total_Phenols	Flavanoids	Nonflavanoid_Phenols	Proanthocyanins	Colc
0	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	
2	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	
3	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	
4	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	

```
In [3]: X = dataset.iloc[:, 0:13].values
        y = dataset.iloc[:, 13].values

        from sklearn.model_selection import train_test_split

        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 0)

        from sklearn.preprocessing import StandardScaler
        sc = StandardScaler()

        X_train = sc.fit_transform(X_train)
        X_test = sc.transform(X_test)

        from sklearn.decomposition import PCA

        pca = PCA(n_components = 2)

        X_train = pca.fit_transform(X_train)
        X_test = pca.transform(X_test)

        explained_variance = pca.explained_variance_ratio_

        from sklearn.linear_model import LogisticRegression

        classifier = LogisticRegression(random_state = 0)
        classifier.fit(X_train, y_train)

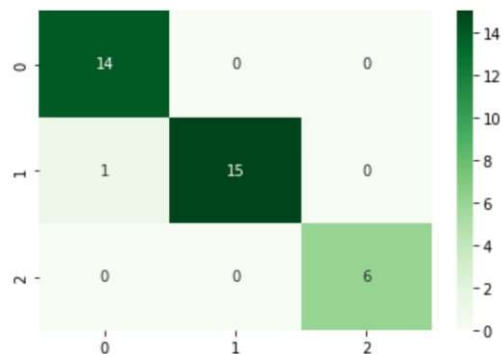
        y_pred = classifier.predict(X_test)

        from sklearn.metrics import accuracy_score
        from sklearn.metrics import confusion_matrix
        from sklearn.metrics import classification_report

        cm = confusion_matrix(y_test, y_pred)
        cm

        sns.heatmap(confusion_matrix(y_test, y_pred), annot = True, cmap = 'Greens')
```

Out[3]: <AxesSubplot:>



```
In [4]: cr = classification_report(y_test, y_pred)
cr
```

```
Out[4]: '          precision    recall  f1-score   support\n\n 0.97          14\n 0          1.00          1.00           6\n macro avg          0.98          0.98          0.98          36\n 97          36'
```

```
In [6]: ac = accuracy_score(y_test, y_pred)
ac
```

```
Out[6]: 0.9722222222222222
```

Result: Accuracy of the model is approximately 95%.