Experiment No. 7
Apply Dimensionality Reduction on Adult Census Income Dataset and analyze the performance of the model
Date of Performance: 11/10/2023
Date of Submission: 12/10/2023

Aim: Apply Dimensionality Reduction on Adult Census Income Dataset and analyze the performance of the model.

Objective: Able to perform various feature engineering tasks, perform dimetionality reduction on the given dataset and maximize the accuracy, Precision, Recall, F1 score.

Theory:

In machine learning classification problems, there are often too many factors on the basis of which the final classification is done. These factors are basically variables called features. The higher the number of features, the harder it gets to visualize the training set and then work on it. Sometimes, most of these features are correlated, and hence redundant. This is where dimensionality reduction algorithms come into play. Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables. It can be divided into feature selection and feature extraction.

Dataset:

Predict whether income exceeds $50K/yr based on census data. Also known as "Adult" dataset.

Attribute Information:

Listing of attributes: >50K, <=50K.

age: continuous.

workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.

fnlwgt: continuous.

education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.

education-num: continuous.

marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.

occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.

relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.

race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.

sex: Female, Male.

capital-gain: continuous.

capital-loss: continuous.

hours-per-week: continuous.

native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad & Tobago, Peru, Hong, Holand-Netherlands.

Code:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, precision_score, recall_score, f1_score
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"
column_names = ["age", "workclass", "fnlwgt", "education", "education-num", "marital-status", "occupation", "relationship", "race", "sex", "capital-gain", "capital-loss", "hours-per-week", "native-country", "income"]
data = pd.read_csv(url, names=column_names, na_values=" ?", skipinitialspace=True)
data.dropna(inplace=True)
label_encoder = LabelEncoder()
data['income'] = label_encoder.fit_transform(data['income'])
categorical_cols = data.select_dtypes(include=['object']).columns
data = pd.get_dummies(data, columns=categorical_cols, drop_first=True)
X = data.drop("income", axis=1)
y = data["income"]
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X = pd.DataFrame(X_scaled, columns=X.columns)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
logistic_reg = LogisticRegression()
logistic_reg.fit(X_train, y_train)
y_pred = logistic_reg.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
confusion = confusion_matrix(y_test, y_pred)
print("Result Without Dimensionality Reduction:")
print(f"Accuracy: {accuracy}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(f"F1 Score: {f1}")
print("Confusion Matrix:")
print(confusion)

pca = PCA(n_components=0.95)  # Choose the number of components to explain 95% of the variance
X_pca = pca.fit_transform(X)
X_train_pca, X_test_pca, y_train, y_test = train_test_split(X_pca, y, test_size=0.2, random_state=42)
logistic_reg_pca = LogisticRegression()
logistic_reg_pca.fit(X_train_pca, y_train)
y_pred_pca = logistic_reg_pca.predict(X_test_pca)
accuracy_pca = accuracy_score(y_test, y_pred_pca)
precision_pca = precision_score(y_test, y_pred_pca)
recall_pca = recall_score(y_test, y_pred_pca)

f1_pca = f1_score(y_test, y_pred_pca)
confusion_pca = confusion_matrix(y_test, y_pred_pca)
print("\n Result With Dimensionality Reduction:")
print(f"Accuracy with PCA: {accuracy_pca}")
print(f"Precision with PCA: {precision_pca}")
print(f"Recall with PCA: {recall_pca}")
print(f"F1 Score with PCA: {f1_pca}")
print("Confusion Matrix with PCA:")
print(confusion_pca)
```

Conclusion:

Impact of dimensionality reduction on the accuracy, precision, recall and F1 score.

Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables. It helps in data compression by reducing features. It reduces storage. It makes machine learning algorithms computationally efficient. It also helps remove redundant features and noise.

In this experiment, we concluded the effect of Dimensionality Reduction as follows:

- **Result Without Dimensionality Reduction:**

  Accuracy: 0.8569015814524796

```
Precision: 0.7482517482517482
Recall: 0.6129853596435392
F1 Score: 0.6738978306508048
Confusion Matrix:
[[4618  324]
 [ 608  963]]
```

- **Result With Dimensionality Reduction:**

  Accuracy with PCA: 0.8506064793489944

```
Precision with PCA: 0.7361769352290679
Recall with PCA: 0.593252705283259
F1 Score with PCA: 0.6570320761367641
Confusion Matrix with PCA:
[[4608  334]
 [ 639  932]]
```