| Experiment No. 7 |
| --- |
| Apply Dimensionality Reduction on Adult Census Income  Dataset and analyze the performance of the model |
| Date of Performance: 17/9/2024 |
| Date of Submission: 24/9/2024 |

**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:





| # Separate the target variable (income) from the features  X = data.drop('income', axis=1) y = data['income'] |
| --- |
| categorical\_columns = X.select\_dtypes(include=['object']).columns label\_encoder = LabelEncoder()  X[categorical\_columns] = X[categorical\_columns].apply(label\_encoder.fit\_transform) |
| # Step 2: Apply PCA for dimensionality reduction  # You can adjust the number of components (n\_components) as needed n\_components = 14 # You can change this value  pca = PCA(n\_components=n\_components) X\_pca = pca.fit\_transform(X) |
| # Step 3: Train classifiers on Original and Reduced-dimensioned data  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  X\_pca\_train, X\_pca\_test, y\_pca\_train, y\_pca\_test = train\_test\_split(X\_pca, y, test\_size=0.2, random\_state= |
| # Initialize a Random Forest classifier  rf\_classifier = RandomForestClassifier(random\_state=42) |
| # Fit the classifier on the original data rf\_classifier.fit(X\_train, y\_train)  y\_pred = rf\_classifier.predict(X\_test) |
| # Calculate performance metrics for the original data accuracy\_original = accuracy\_score(y\_test, y\_pred)  precision\_original = precision\_score(y\_test, y\_pred, pos\_label='>50K') recall\_original = recall\_score(y\_test, y\_pred, pos\_label='>50K')  f1\_score\_original = f1\_score(y\_test, y\_pred, pos\_label='>50K') |
| # Fit the classifier on the reduced-dimensioned data rf\_classifier.fit(X\_pca\_train, y\_pca\_train)  y\_pca\_pred = rf\_classifier.predict(X\_pca\_test) |
| # Calculate performance metrics for the reduced-dimensioned data accuracy\_pca = accuracy\_score(y\_pca\_test, y\_pca\_pred)  precision\_pca = precision\_score(y\_pca\_test, y\_pca\_pred, pos\_label='>50K') recall\_pca = recall\_score(y\_pca\_test, y\_pca\_pred, pos\_label='>50K')  f1\_score\_pca = f1\_score(y\_pca\_test, y\_pca\_pred, pos\_label='>50K') |
| confusion\_matrix\_original = confusion\_matrix(y\_test, y\_pred) print("Confusion Matrix for Original Data:")  print(confusion\_matrix\_original) |

Confusion Matrix for Original Data: [[4627 349]

[ 604 933]]





Performance Metrics for Original Data: Accuracy: 0.8536772608628896

Precision: 0.7277691107644306

Recall: 0.6070266753415745

F1 Score: 0.6619368570415041



Performance Metrics for PCA-Reduced Data: Accuracy: 0.8532166436358053

Precision: 0.745977984758679

Recall: 0.5731945348080677

F1 Score: 0.648270787343635



Confusion Matrix for PCA-Reduced Data: [[4676 300]

[ 656 881]]

**Conclusion:**

Original Data:

Accuracy: 85.37%

Precision: 72.78%

Recall: 60.70%

F1 Score: 66.19%

PCA-Reduced Data:

Accuracy: 85.32%

Precision: 74.60%

Recall: 57.32%

F1 Score: 64.83%

In conclusion, both the original data and the PCA-reduced data exhibit similar overall model performance. The PCA-reduced data, despite having fewer dimensions, achieves a relatively competitive level of performance with a slightly improved precision, although at the expense of recall. The choice between using the original or PCA-reduced data depends on your specific goals and the trade-offs between model complexity, interpretability, and performance.