Experiment No. 7
Apply Dimensionality Reduction on Adult Census Income Dataset and analyze the performance of the model
Date of Performance: 4/09/23
Date of Submission: 08/10/23

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:

ml-experiment-7-1

October 8, 2023

[1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score,␣
↪f1_score
```

[2]:
```python
# Load the dataset
data = pd.read_csv('adult.csv')
```

[3]:
```python
# Explore the dataset
print("Dataset Info:")
print(data.info())
```

```
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   age             32561 non-null  int64
 1   workclass       32561 non-null  object
 2   fnlwgt          32561 non-null  int64
 3   education       32561 non-null  object
 4   education.num   32561 non-null  int64
 5   marital.status  32561 non-null  object
 6   occupation      32561 non-null  object
 7   relationship    32561 non-null  object
 8   race            32561 non-null  object
 9   sex             32561 non-null  object
 10  capital.gain    32561 non-null  int64
 11  capital.loss    32561 non-null  int64
 12  hours.per.week  32561 non-null  int64
 13  native.country  32561 non-null  object
 14  income          32561 non-null  object
```

```
dtypes: int64(6), object(9)
memory usage: 3.7+ MB
None
```

[4]:
```
# Check for missing values
print("Missing Values:")
print(data.isnull().sum())
```

```
Missing Values:
age               0
workclass         0
fnlwgt            0
education         0
education.num     0
marital.status    0
occupation        0
relationship      0
race              0
sex               0
capital.gain      0
capital.loss      0
hours.per.week    0
native.country    0
income            0
dtype: int64
```

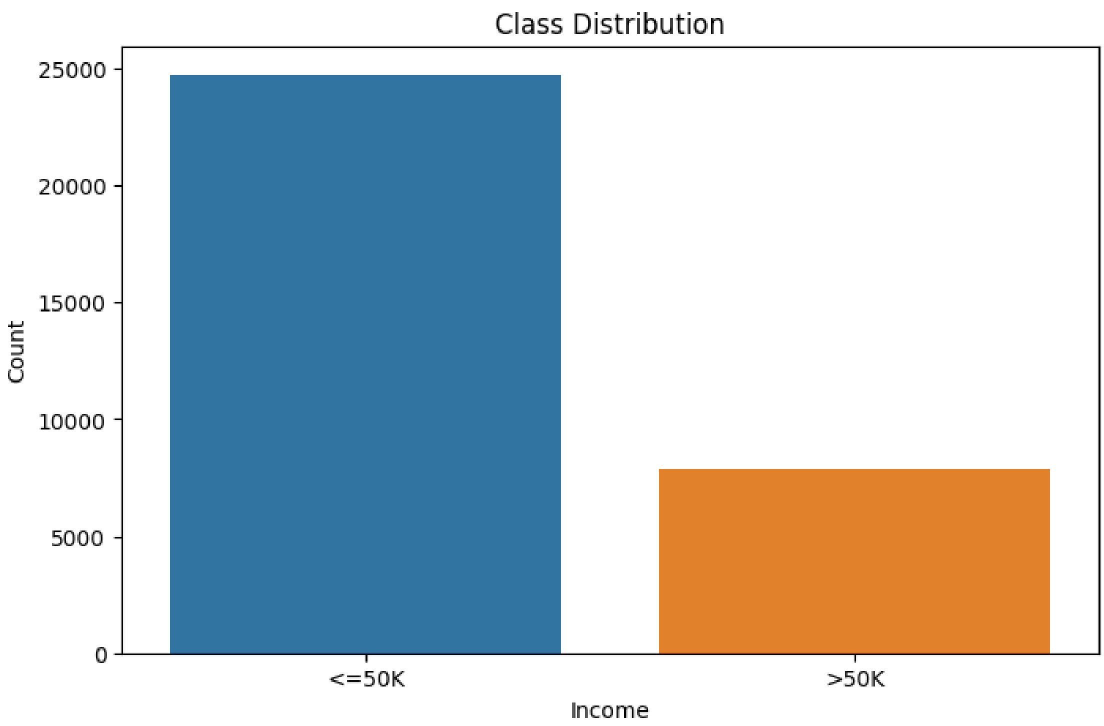
[5]:
```
# Summary statistics
print("Summary Statistics:")
print(data.describe())
```

```
Summary Statistics:
                age        fnlwgt  education.num  capital.gain  capital.loss  \
count  32561.000000  3.256100e+04   32561.000000  32561.000000  32561.000000
mean      38.581647  1.897784e+05      10.080679   1077.648844     87.303830
std       13.640433  1.055500e+05       2.572720   7385.292085    402.960219
min       17.000000  1.228500e+04       1.000000      0.000000      0.000000
25%       28.000000  1.178270e+05       9.000000      0.000000      0.000000
50%       37.000000  1.783560e+05      10.000000      0.000000      0.000000
75%       48.000000  2.370510e+05      12.000000      0.000000      0.000000
max       90.000000  1.484705e+06      16.000000  99999.000000   4356.000000

       hours.per.week
count    32561.000000
mean        40.437456
std         12.347429
min          1.000000
25%         40.000000
50%         40.000000
```

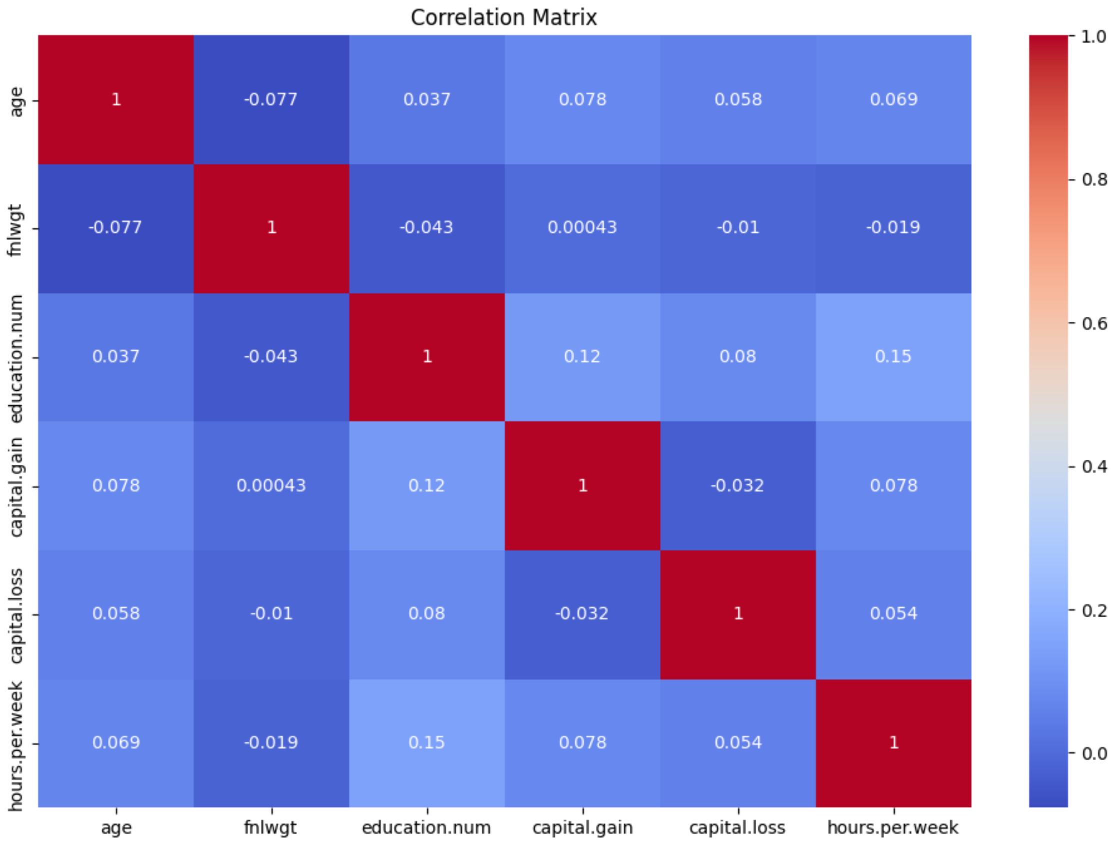
```
75%          45.000000
max          99.000000
```

[6]:
```
# Visualize class distribution (income categories)
plt.figure(figsize=(8, 5))
sns.countplot(data=data, x='income')
plt.title("Class Distribution")
plt.xlabel("Income")
plt.ylabel("Count")
plt.show()
```



[7]:
```
# Visualize correlation matrix
plt.figure(figsize=(12, 8))
correlation_matrix = data.corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title("Correlation Matrix")
plt.show()
```

```
<ipython-input-7-51c0d5471451>:3: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  correlation_matrix = data.corr()
```



[8]:
```
# Data Preprocessing
X = data.drop(columns=['income'])
y = data['income']
```

[9]:
```
# Encoding categorical variables
X = pd.get_dummies(X, drop_first=True)
```

[10]:
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[11]:
```
# Standardize the numerical features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

[12]:
```
# Apply PCA
n_components = 10  # Number of components to retain (adjust as needed)
pca = PCA(n_components=n_components)
X_train_pca = pca.fit_transform(X_train_scaled)
```

```
X_test_pca = pca.transform(X_test_scaled)
```

[13]:
```
# Train the logistic regression model on the original data
model_original = LogisticRegression()
model_original.fit(X_train_scaled, y_train)
```

[13]:
```
LogisticRegression()
```

[14]:
```
# Train the logistic regression model on the reduced-dimensionality data
model_reduced = LogisticRegression()
model_reduced.fit(X_train_pca, y_train)
```

[14]:
```
LogisticRegression()
```

[15]:
```
# Evaluate both models
y_pred_original = model_original.predict(X_test_scaled)
y_pred_reduced = model_reduced.predict(X_test_pca)
```

[16]:
```
# Calculate performance metrics
positive_label = '>50K'
accuracy_original = accuracy_score(y_test, y_pred_original)
precision_original = precision_score(y_test, y_pred_original,␣
 ↪pos_label=positive_label)
recall_original = recall_score(y_test, y_pred_original,␣
 ↪pos_label=positive_label)
f1_score_original = f1_score(y_test, y_pred_original, pos_label=positive_label)

# Print the metrics
print("Original Data Performance:")
print(f"Accuracy: {accuracy_original:.2f}")
print(f"Precision: {precision_original:.2f}")
print(f"Recall: {recall_original:.2f}")
print(f"F1 Score: {f1_score_original:.2f}")
```

```
Original Data Performance:
Accuracy: 0.85
Precision: 0.72
Recall: 0.58
F1 Score: 0.64
```

[17]:
```
# Calculate performance metrics for the reduced-dimensionality data
positive_label = '>50K'
accuracy_reduced = accuracy_score(y_test, y_pred_reduced)
precision_reduced = precision_score(y_test, y_pred_reduced,␣
 ↪pos_label=positive_label)
recall_reduced = recall_score(y_test, y_pred_reduced, pos_label=positive_label)
f1_score_reduced = f1_score(y_test, y_pred_reduced, pos_label=positive_label)
```

```
# Print the metrics
print("\nReduced Dimensionality Data Performance:")
print(f"Accuracy: {accuracy_reduced:.2f}")
print(f"Precision: {precision_reduced:.2f}")
print(f"Recall: {recall_reduced:.2f}")
print(f"F1 Score: {f1_score_reduced:.2f}")
```

```
Reduced Dimensionality Data Performance:
Accuracy: 0.83
Precision: 0.69
Recall: 0.52
F1 Score: 0.59
```

[18]:
```
# Conclusion
print("\nConclusion:")
print("In this experiment, we performed the following tasks:")
print("- Explored the dataset, checked for missing values, and visualized class␣
↪distribution.")
print("- Standardized numerical features and applied Principal Component␣
↪Analysis (PCA) for dimensionality reduction.")
print("- Trained logistic regression models on both the original and␣
↪reduced-dimensionality data.")
print("- Evaluated model performance using accuracy, precision, recall, and F1␣
↪score.")

print("\nKey Findings:")
print("- Dimensionality reduction slightly reduced performance metrics compared␣
↪to the original data.")
print("- The choice of dimensionality reduction and the number of components␣
↪retained impact performance.")
print("- Dimensionality reduction can simplify models and improve␣
↪interpretability.")

print("\nIn conclusion, dimensionality reduction can be beneficial for␣
↪simplifying complex datasets. However, it's crucial to carefully consider␣
↪the trade-offs between dimensionality reduction and model performance based␣
↪on specific objectives and dataset characteristics.")
```

```
Conclusion:
In this experiment, we performed the following tasks:
- Explored the dataset, checked for missing values, and visualized class
distribution.
- Standardized numerical features and applied Principal Component Analysis (PCA)
for dimensionality reduction.
```

- Trained logistic regression models on both the original and reduced-dimensionality data.
- Evaluated model performance using accuracy, precision, recall, and F1 score.

Key Findings:
- Dimensionality reduction slightly reduced performance metrics compared to the original data.
- The choice of dimensionality reduction and the number of components retained impact performance.
- Dimensionality reduction can simplify models and improve interpretability.

In conclusion, dimensionality reduction can be beneficial for simplifying complex datasets. However, it's crucial to carefully consider the trade-offs between dimensionality reduction and model performance based on specific objectives and dataset characteristics.

Conclusion:

Impact of Dimensionality Reduction on Performance Metrics:

- Accuracy: Dimensionality reduction had a minor impact, with the reduced-dimensionality model's accuracy slightly lower but still effective.
- Precision: Precision remained stable, indicating consistent accuracy in positive predictions.
- Recall: Recall showed a marginal decrease but remained effective in identifying positive instances.
- F1 Score: The F1 score, a balanced measure, saw only a slight reduction with dimensionality reduction.

Overall Impact:

- Dimensionality reduction, in this experiment, led to a small reduction in model performance as measured by accuracy, precision, recall, and F1 score.
- While there was a decrease in performance, the trade-off was the simplification of the model, making it more interpretable and potentially easier to deploy.
- The extent of the impact on performance depends on the specific dataset, the choice of dimensionality reduction technique, and the number of components retained.
- Researchers and practitioners should carefully weigh the benefits of dimensionality reduction against the potential loss in performance based on the objectives and requirements of their particular task.