



Experiment No. 7
Apply Dimensionality Reduction on Adult Census Income Dataset and analyze the performance of the model
Date of Performance: 12/10/2023
Date of Submission: 16/10/2023



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 dimensionality 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.



Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

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, Marriedspouse-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, OutlyingUS(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, Trinidad & Tobago, Peru, Hong, Holand-Netherlands.

```
In [2]: import numpy as np
import pandas as pd
```

```
df = pd.read_csv("adult.csv")
df.head()
```

Out[2]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship	race	gender
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	Black	
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	
4	18	?	103497	Some-college	10	Never-married	?	Own-child	White	Female

```
In [3]: df.describe()
```

Out[3]:

	age	fnlwgt	educational-num	capital-gain	capital-loss	hours-per-week
count	48842.000000	4.884200e+04	48842.000000	48842.000000	48842.000000	48842.000000
mean	38.643585	1.896641e+05	10.078089	1079.067626	87.502314	40.422382
std	13.710510	1.056040e+05	2.570973	7452.019058	403.004552	12.391444
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.175505e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.781445e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.376420e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.490400e+06	16.000000	99999.000000	4356.000000	99.000000

```
In [4]: df.shape
```

Out[4]: (48842, 15)

]:

In [5

df.info

```
Out[5]: <bound method DataFrame.info of          age      workclass  fnlwgt      educati
on  educational-num \
0      25      Private  226802      11th          7
1      38      Private  89814      HS-grad        9
2      28  Local-gov  336951  Assoc-acdm       12
3      44      Private  160323  Some-college  10
4      18          ?  103497  Some-college  10
...    ...      ...      ...      ...      ...
48837   27      Private  257302  Assoc-acdm       12
48838   40      Private  154374      HS-grad        9
48839   58      Private  151910      HS-grad        9
48840   22      Private  201490      HS-grad        9
48841   52  Self-emp-inc  287927      HS-grad        9

      marital-status      occupation  relationship  race  gender \
0      Never-married  Machine-op-inspct  Own-child  Black  Male
1  Married-civ-spouse  Farming-fishing  Husband  White  Male
2  Married-civ-spouse  Protective-serv  Husband  White  Male
3  Married-civ-spouse  Machine-op-inspct  Husband  Black  Male
4      Never-married          ?  Own-child  White  Female
...    ...      ...      ...      ...      ...
48837  Married-civ-spouse  Tech-support  Wife  White  Female
48838  Married-civ-spouse  Machine-op-inspct  Husband  White  Male
48839      Widowed  Adm-clerical  Unmarried  White  Female
48840      Never-married  Adm-clerical  Own-child  White  Male
48841  Married-civ-spouse  Exec-managerial  Wife  White  Female

      capital-gain  capital-loss  hours-per-week  native-country  income
0              0              0              40  United-States  <=50K
1              0              0              50  United-States  <=50K
2              0              0              40  United-States  >50K
3          7688              0              40  United-States  >50K
4              0              0              30  United-States  <=50K
...    ...      ...      ...      ...      ...
48837              0              0              38  United-States  <=50K
48838              0              0              40  United-States  >50K
48839              0              0              40  United-States  <=50K
48840              0              0              20  United-States  <=50K
48841          15024              0              40  United-States  >50K

[48842 rows x 15 columns]>
```

]:

```
In [7] df.isnull().sum()
```

```
Out[7]: age                0
workclass          2799
fnlwgt             0
education          0
educational-num    0
marital-status     0
occupation         2809
relationship       0
race               0
gender             0
capital-gain       0
capital-loss       0
hours-per-week     0
native-country     857
income            0
dtype: int64
```

```
In [9]: for col in ['workclass', 'occupation', 'native-country']:
        df[col].fillna(df[col].mode()[0], inplace=True)
df.isnull().sum()
```

```
Out[9]: age                0
workclass                0
fnlwgt                   0
education                0
educational-num          0
marital-status           0
occupation               0
relationship             0
race                     0
gender                   0
capital-gain             0
capital-loss             0
hours-per-week           0
native-country           0
income                   0
dtype: int64
```

```
In [11]: from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, ran
```

```
In [14]: from sklearn import preprocessing
categorical = ['workclass', 'education', 'marital-status', 'occupation', 'rela
for feature in categorical:
    label = preprocessing.LabelEncoder()
    X_train[feature] = label.fit_transform(X_train[feature])
    X_test[feature] = label.transform(X_test[feature])
```

In []:

```
15 from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = pd.DataFrame(scaler.fit_transform(X_train), columns = X.columns)
X_test = pd.DataFrame(scaler.transform(X_test), columns = X.columns)
X_train.head()
```

Out[15]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship
0	-0.849978	-1.887643	-0.551219	1.212393	-0.027733	-0.406325	-1.554732	0.969833
1	0.241031	-0.094859	1.687545	-2.650223	-1.587187	-0.406325	-1.049322	0.969833
2	-0.486308	1.697924	-1.434052	-0.590161	0.362131	-0.406325	-0.543912	-0.899325
3	-0.195373	-0.094859	-0.384485	1.212393	-0.027733	0.922720	-0.796617	-0.276272
4	-0.704510	-0.094859	1.608144	0.182362	-0.417596	1.587242	1.730434	1.592886

```
In [18]: from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

LR = LogisticRegression()
LR.fit(X_train, y_train)
```

Out[18]: LogisticRegression()

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
In [20]: y_pred = LR.predict(X_test)
accuracy_score(y_test, y_pred)
```

Out[20]: 0.8221524602470484

```
In [21]: from sklearn.decomposition import PCA
pca = PCA()
```

```
In [22]: X_train = pca.fit_transform(X_train)
pca.explained_variance_ratio_
```

Out[22]: array([0.14740223, 0.10130193, 0.08096753, 0.07933632, 0.07433976,
0.07314763, 0.07066221, 0.06753572, 0.06516078, 0.06093536,
0.06003764, 0.04864317, 0.04289137, 0.02763835])

```
In [24]: X = df.drop(['income'], axis=1)
y = df['income']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, ran
```


In []:

```
25 categorical = ['workclass', 'education', 'marital-status', 'occupation', 'rela
for feature in categorical:
    label = preprocessing.LabelEncoder()
    X_train[feature] = label.fit_transform(X_train[feature])
    X_test[feature] = label.transform(X_test[feature])
```

```
In [26]: X_train = pd.DataFrame(scaler.fit_transform(X_train), columns = X.columns)
```

```
In [27]: pca= PCA()
pca.fit(X_train)
cumsum = np.cumsum(pca.explained_variance_ratio_)
dim = np.argmax(cumsum >= 0.90) + 1
print('The number of dimensions required to preserve 90% of variance is',dim)
```

The number of dimensions required to preserve 90% of variance is 12

```
In [28]: X = df.drop(['income', 'native-country', 'hours-per-week'], axis=1)
y = df['income']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, ran
```

```
In [30]: categorical = ['workclass', 'education', 'marital-status', 'occupation', 'rela
for feature in categorical:
    label = preprocessing.LabelEncoder()
    X_train[feature] = label.fit_transform(X_train[feature])
    X_test[feature] = label.transform(X_test[feature])
X_train = pd.DataFrame(scaler.fit_transform(X_train), columns = X.columns)
X_test = pd.DataFrame(scaler.transform(X_test), columns = X.columns)
```

```
In [31]: LR2 = LogisticRegression()
LR2.fit(X_train, y_train)
```

```
Out[31]: LogisticRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
In [32]: y_pred = LR2.predict(X_test)
accuracy_score(y_test, y_pred)
```

```
Out[32]: 0.8229031597625059
```


In []:

```
33 from sklearn.metrics import confusion_matrix
import pandas as pd
confusion = confusion_matrix(y_test, y_pred)
df_confusion = pd.DataFrame(confusion, columns=['Predicted No', 'Predicted Yes'])
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
<=50K	0.84	0.94	0.89	11138
>50K	0.71	0.44	0.54	3515
accuracy			0.82	14653
macro avg	0.78	0.69	0.72	14653
weighted avg	0.81	0.82	0.81	14653



Conclusion:

1. The Accuracy score obtained by applying principal component analysis on the testing data is 0.82 which means our model is 82% accurate on the testing data.
2. Precision measures the accuracy of the positive predictions and the precision score obtained by our model is 0.84
3. Recall measures the ability of the model to correctly identify all relevant instances and the Recall score obtained by our model is 0.94
4. F1-score is the harmonic mean of precision and recall and provides a balance between the 2 metrics and the F1-score obtained by our model is 0.89

