unit-2assign-2

November 18, 2024

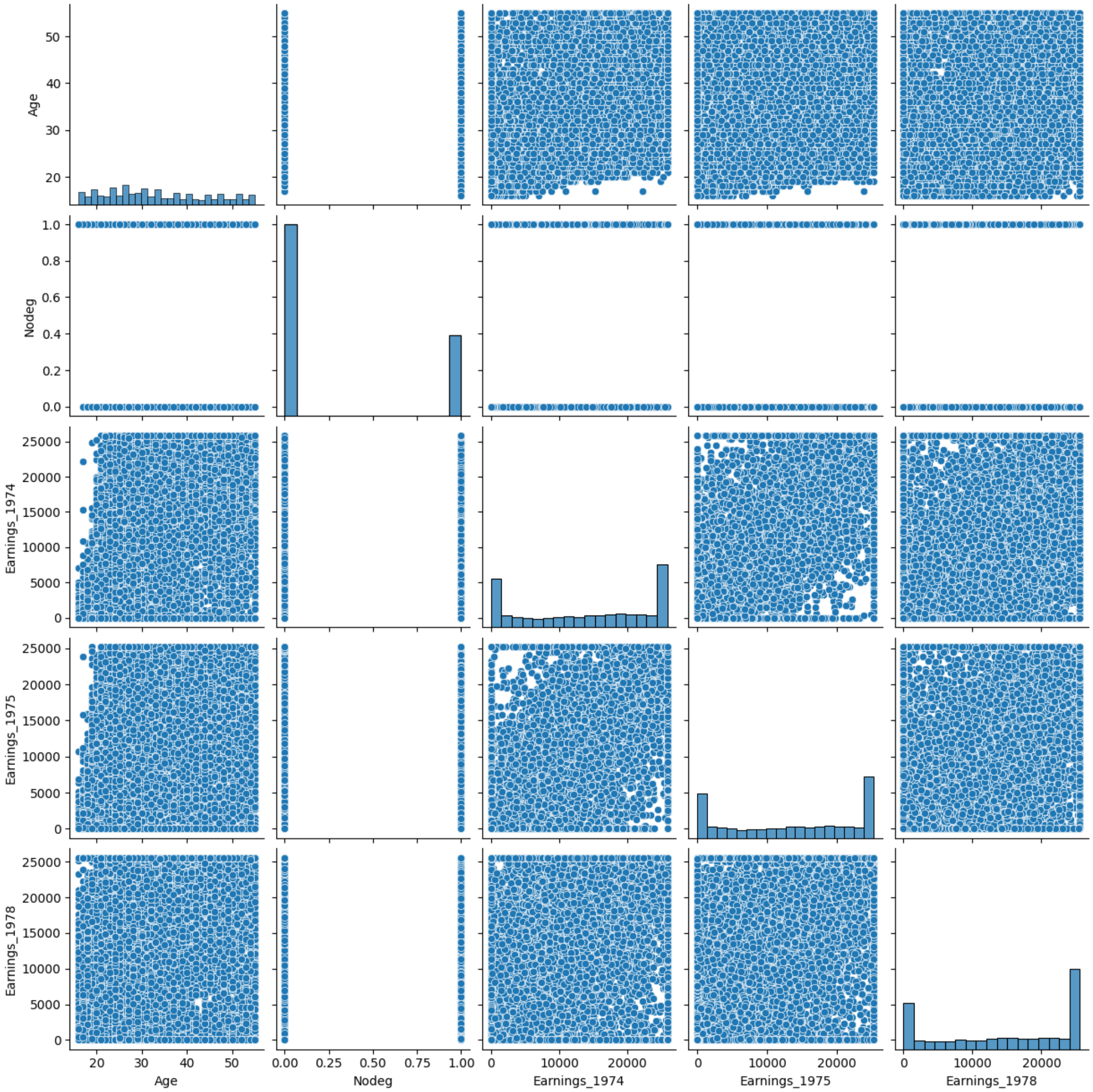
[1]:
```
import pandas as pd

# Load the dataset
data = pd.read_csv("NvWCPg2u12Tza1js.csv")

# Preview the dataset
print(data.head())

# Summary statistics
print(data.describe())

# Check for missing values
print(data.isnull().sum())
```

```
   Age          Eduacation      Race          Hisp MaritalStatus  Nodeg  \
0   45  LessThanHighSchool  NotBlack  NotHispanic       Married      1
1   21        Intermediate  NotBlack  NotHispanic    NotMarried      0
2   38          HighSchool  NotBlack  NotHispanic       Married      0
3   48  LessThanHighSchool  NotBlack  NotHispanic       Married      1
4   18  LessThanHighSchool  NotBlack  NotHispanic       Married      1

   Earnings_1974  Earnings_1975  Earnings_1978
0      21516.670      25243.550      25564.670
1       3175.971       5852.565      13496.080
2      23039.020      25130.760      25564.670
3      24994.370      25243.550      25564.670
4       1669.295      10727.610       9860.869
                Age         Nodeg  Earnings_1974  Earnings_1975  Earnings_1978
count  15992.000000  15992.000000   15992.000000   15992.000000   15992.000000
mean      33.225238      0.295835   14016.800304   13650.803376   14846.659673
std       11.045216      0.456432    9569.795893    9270.403225    9647.391524
min       16.000000      0.000000       0.000000       0.000000       0.000000
25%       24.000000      0.000000    4403.452250    4398.823000    5669.298000
50%       31.000000      0.000000   15123.580000   14557.110000   16421.975000
75%       42.000000      1.000000   23584.180000   22923.737500   25564.670000
max       55.000000      1.000000   25862.320000   25243.550000   25564.670000
Age               0
Eduacation        0
```

```
Race              0
Hisp              0
MaritalStatus     0
Nodeg             0
Earnings_1974     0
Earnings_1975     0
Earnings_1978     0
dtype: int64
```

[2]:

```python
import seaborn as sns
import matplotlib.pyplot as plt

# Plot pairwise relationships
sns.pairplot(data)
plt.show()
```



[8]:

```
X = X.apply(pd.to_numeric, errors='coerce')  # Converts non-numeric values to␣
 ↪NaN
print(X.isnull().sum())  # Check for any NaN introduced
X = X.fillna(0)  # Replace NaN with 0
```

```
Age                             0
Nodeg                           0
Eduacation_Intermediate         0
Eduacation_LessThanHighSchool   0
Education_PostGraduate          0
Eduacation_graduate             0
Race_black                      0
Hisp_hispanic                   0
MaritalStatus_NotMarried        0
dtype: int64
```

[11]:

```
from sklearn.model_selection import train_test_split

# Prepare data
# Replace 'Earnings_1978' with the desired target variable
X = data.drop(['Earnings_1974', 'Earnings_1975', 'Earnings_1978'], axis=1)
y = data['Earnings_1978']  # Use the appropriate column as the target variable

# Split the dataset 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)

# Display the shapes of the splits to verify
print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"y_train shape: {y_train.shape}")
print(f"y_test shape: {y_test.shape}")
```

```
X_train shape: (12793, 6)
X_test shape: (3199, 6)
y_train shape: (12793,)
y_test shape: (3199,)
```

[15]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# Step 1: Load and preprocess the data
# Assuming `data` is already loaded
# Replace 'Earnings_1978' with the desired target variable
X = data.drop(['Earnings_1974', 'Earnings_1975', 'Earnings_1978'], axis=1)
y = data['Earnings_1978']

# Step 2: Encode categorical variables
X = pd.get_dummies(X, drop_first=True)  # Convert categorical columns to dummy␣
 ↪variables

# Step 3: Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Step 4: Train the model
model = LinearRegression()
model.fit(X_train, y_train)  # Fit the linear regression model

# Step 5: Evaluate the model
score = model.score(X_test, y_test)  # R-squared score
print(f"R-squared score: {score}")
```

```
R-squared score: 0.09200104293622746
```

[17]:

```
import joblib
y_pred = model.predict(X_test)
# Save the model
joblib.dump(model, 'linear_regression_model.pkl')

# Save predictions
predictions = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
predictions.to_csv('predictions.csv', index=False)
```