

|                                                                                                       |
|-------------------------------------------------------------------------------------------------------|
| Experiment No. 3                                                                                      |
| Apply Decision Tree Algorithm on Adult Census Income Dataset and analyze the performance of the model |
| Date of Performance: 07/08/2023                                                                       |
| Date of Submission: 21/08/2023                                                                        |

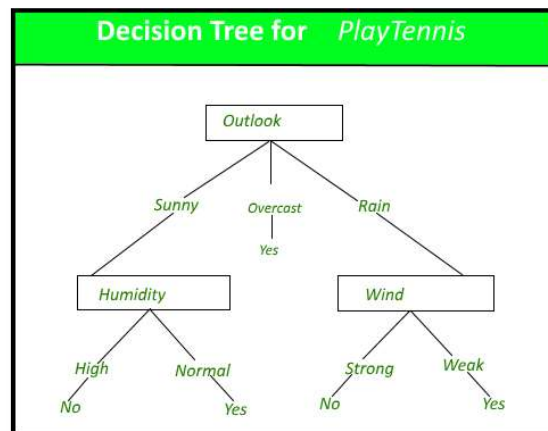


**Aim:** Apply Decision Tree Algorithm on Adult Census Income Dataset and analyze the performance of the model.

**Objective:** To perform various feature engineering tasks, apply Decision Tree Algorithm on the given dataset and maximize the accuracy, Precision, Recall, F1 score. Improve the performance by performing different data engineering and feature engineering tasks.

**Theory:**

Decision Tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.



**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, Trinidad&Tobago, Peru, Hong, Holand-Netherlands.

**Conclusion:**

1.Categorical attributes in the dataset, includes workclass, education, marital status, relationship, race, sex, native country, and income, were preprocessed using label encoding. Label encoding is a technique that converts categorical variables into a numerical format, making them suitable for machine learning algorithms.

2.Hyperparameter tuning was performed on the decision tree model. The following parameters were tuned:

- a. Max Depth: The maximum depth of the tree, which controls its complexity. It was adjusted to find the optimal tree depth.
- b. Min Samples Leaf: The minimum number of samples required at a leaf node was tuned to optimize the model's performance.
- c. Min Samples Split: This parameter, specifying the minimum number of samples needed to split an internal node, was adjusted to improve model accuracy.

3.The accuracy of the model is 81% , precision is 83% , recall is 94% and F1 score is 89% . Based on the confusion matrix it can observed that the Type I error is 540 and Type II error is 1955.

```
from google.colab import drive
```

```
# Mount Google Drive
```

```
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import pandas as pd
```

```
import numpy as np
```

```
import seaborn as sns
```

```
import matplotlib.pyplot as plt
```

```
df=pd.read_csv("/content/drive/MyDrive/dataset/adult.csv")
```

```
df.head()
```

|   | age      | workclass | fnlwgt | education    | educational-num | marital-     |
|---|----------|-----------|--------|--------------|-----------------|--------------|
|   | status \ |           |        |              |                 |              |
| 0 | 25       | Private   | 226802 | 11th         | 7               | Never-       |
| 1 | 38       | Private   | 89814  | HS-grad      | 9               | Married-civ- |
| 2 | 28       | Local-gov | 336951 | Assoc-acdm   | 12              | Married-civ- |
| 3 | 44       | Private   | 160323 | Some-college | 10              | Married-civ- |
| 4 | 18       | ?         | 103497 | Some-college | 10              | Never-       |
|   |          |           |        |              |                 | married      |

|   | occupation        | relationship | race  | gender | capital-gain |
|---|-------------------|--------------|-------|--------|--------------|
|   | capital-loss \    |              |       |        |              |
| 0 | Machine-op-inspct | Own-child    | Black | Male   | 0            |
| 1 | Farming-fishing   | Husband      | White | Male   | 0            |
| 2 | Protective-serv   | Husband      | White | Male   | 0            |
| 3 | Machine-op-inspct | Husband      | Black | Male   | 7688         |
| 4 | ?                 | Own-child    | White | Female | 0            |

|   | hours-per-week | native-country | income |
|---|----------------|----------------|--------|
| 0 | 40             | United-States  | <=50K  |
| 1 | 50             | United-States  | <=50K  |
| 2 | 40             | United-States  | >50K   |
| 3 | 40             | United-States  | >50K   |
| 4 | 30             | United-States  | <=50K  |

```
df.shape
```

```
(48842, 15)
```

## Data Preprocessing

```
df.isnull().sum()
```

```
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
```

```
df['workclass'].unique()
```

```
array(['Private', 'Local-gov', '?', 'Self-emp-not-inc', 'Federal-gov',
      'State-gov', 'Self-emp-inc', 'Without-pay', 'Never-worked'],
      dtype=object)
```

```
df.describe()
```

|       | age          | fnlwgt       | educational-num | capital-gain \ |
|-------|--------------|--------------|-----------------|----------------|
| count | 48842.000000 | 4.884200e+04 | 48842.000000    | 48842.000000   |
| mean  | 38.643585    | 1.896641e+05 | 10.078089       | 1079.067626    |
| std   | 13.710510    | 1.056040e+05 | 2.570973        | 7452.019058    |
| min   | 17.000000    | 1.228500e+04 | 1.000000        | 0.000000       |
| 25%   | 28.000000    | 1.175505e+05 | 9.000000        | 0.000000       |
| 50%   | 37.000000    | 1.781445e+05 | 10.000000       | 0.000000       |
| 75%   | 48.000000    | 2.376420e+05 | 12.000000       | 0.000000       |
| max   | 90.000000    | 1.490400e+06 | 16.000000       | 99999.000000   |

|       | capital-loss | hours-per-week |
|-------|--------------|----------------|
| count | 48842.000000 | 48842.000000   |
| mean  | 87.502314    | 40.422382      |
| std   | 403.004552   | 12.391444      |
| min   | 0.000000     | 1.000000       |
| 25%   | 0.000000     | 40.000000      |
| 50%   | 0.000000     | 40.000000      |
| 75%   | 0.000000     | 45.000000      |
| max   | 4356.000000  | 99.000000      |

```
df['capital-gain'].unique()
```

```
array([    0,  7688,  3103,  6418,  7298,  3908, 14084,  5178, 15024,
        99999, 2597,  2907,  4650,  6497,  1055,  5013, 27828,  4934,
        4064,  3674,  2174, 10605,  3418,   114,  2580,  3411,  4508,
        4386,  8614, 13550,  6849,  2463,  3137,  2885,  2964,  1471,
       10566,  2354,  1424,  1455,  3325,  4416, 25236,   594,  2105,
        4787,  2829,   401,  4865,  1264,  1506, 10520,  3464,  2653,
       20051,  4101,  1797,  2407,  3471,  1086,  1848, 14344,  1151,
        2993,  2290, 15020,  9386,  2202,  3818,  2176,  5455, 11678,
        7978,  7262,  6514, 41310,  3456,  7430,  2414,  2062, 34095,
        1831,  6723,  5060, 15831,  2977,  2346,  3273,  2329,  9562,
        2635,  4931,  1731,  6097,   914,  7896,  5556,  1409,  3781,
        3942,  2538,  3887, 25124,  7443,  5721,  1173,  4687,  6612,
        6767,  2961,   991,  2036,  2936,  2050,  1111,  2228, 22040,
        3432,  6360,  2009,  1639, 18481,  2387])
```

```
df['capital-gain'].nunique()
```

```
123
```

```
df['capital-gain'].value_counts()
```

```
0          44807
15024        513
7688         410
7298         364
99999        244
...
1111          1
7262          1
22040         1
1639          1
2387          1
```

```
Name: capital-gain, Length: 123, dtype: int64
```

```
df['capital-loss'].value_counts()
```

```
0          46560
1902        304
1977        253
1887        233
2415         72
...
2465         1
2080         1
155          1
1911         1
2201         1
```

```
Name: capital-loss, Length: 99, dtype: int64
```

```
df['fnlwgt'].value_counts()
203488    21
190290    19
120277    19
125892    18
126569    18
..
188488     1
285290     1
293579     1
114874     1
257302     1
Name: fnlwgt, Length: 28523, dtype: int64

df['race'].unique()
array(['Black', 'White', 'Asian-Pac-Islander', 'Other',
       'Amer-Indian-Eskimo'], dtype=object)
```

## Drop columns fnlwgt , race , capital-gain , capital-loss

```
df.duplicated().sum()
47

df['workclass'].value_counts()
Private          33906
Self-emp-not-inc  3862
Local-gov        3136
?                2799
State-gov        1981
Self-emp-inc     1695
Federal-gov      1432
Without-pay      21
Never-worked     10
Name: workclass, dtype: int64

df.replace('?', np.nan, inplace = True)
df.dropna(inplace=True)

df['workclass'].value_counts()
Private          33262
Self-emp-not-inc  3795
Local-gov        3100
State-gov        1946
Self-emp-inc     1645
Federal-gov      1406
```



```
Without-pay          21
Name: workclass, dtype: int64
```

```
df.shape
```

```
(45222, 15)
```

```
df=df.drop_duplicates()
```

```
df.shape
```

```
(45175, 15)
```

```
df.head()
```

|   | age | workclass | fnlwgt | education    | educational-num | marital-status \   |
|---|-----|-----------|--------|--------------|-----------------|--------------------|
| 0 | 25  | Private   | 226802 | 11th         | 7               | Never-married      |
| 1 | 38  | Private   | 89814  | HS-grad      | 9               | Married-civ-spouse |
| 2 | 28  | Local-gov | 336951 | Assoc-acdm   | 12              | Married-civ-spouse |
| 3 | 44  | Private   | 160323 | Some-college | 10              | Married-civ-spouse |
| 5 | 34  | Private   | 198693 | 10th         | 6               | Never-married      |

|   | occupation        | relationship  | race  | gender | capital-gain |
|---|-------------------|---------------|-------|--------|--------------|
| 0 | Machine-op-inspct | Own-child     | Black | Male   | 0            |
| 1 | Farming-fishing   | Husband       | White | Male   | 0            |
| 2 | Protective-serv   | Husband       | White | Male   | 0            |
| 3 | Machine-op-inspct | Husband       | Black | Male   | 7688         |
| 5 | Other-service     | Not-in-family | White | Male   | 0            |

|   | hours-per-week | native-country | income |
|---|----------------|----------------|--------|
| 0 | 40             | United-States  | <=50K  |
| 1 | 50             | United-States  | <=50K  |
| 2 | 40             | United-States  | >50K   |
| 3 | 40             | United-States  | >50K   |
| 5 | 30             | United-States  | <=50K  |

```
df.drop(['fnlwgt', 'race', 'capital-gain', 'capital-loss'],axis=1,inplace=True)
```

```

df.head()

```

|   | age | workclass | education    | educational-num | marital-           |
|---|-----|-----------|--------------|-----------------|--------------------|
| 0 | 25  | Private   | 11th         | 7               | Never-married      |
| 1 | 38  | Private   | HS-grad      | 9               | Married-civ-spouse |
| 2 | 28  | Local-gov | Assoc-acdm   | 12              | Married-civ-spouse |
| 3 | 44  | Private   | Some-college | 10              | Married-civ-spouse |
| 5 | 34  | Private   | 10th         | 6               | Never-married      |

|   | occupation        | relationship  | gender | hours-per-week | native- |
|---|-------------------|---------------|--------|----------------|---------|
| 0 | Machine-op-inspct | Own-child     | Male   | 40             | United- |
| 1 | Farming-fishing   | Husband       | Male   | 50             | United- |
| 2 | Protective-serv   | Husband       | Male   | 40             | United- |
| 3 | Machine-op-inspct | Husband       | Male   | 40             | United- |
| 5 | Other-service     | Not-in-family | Male   | 30             | United- |

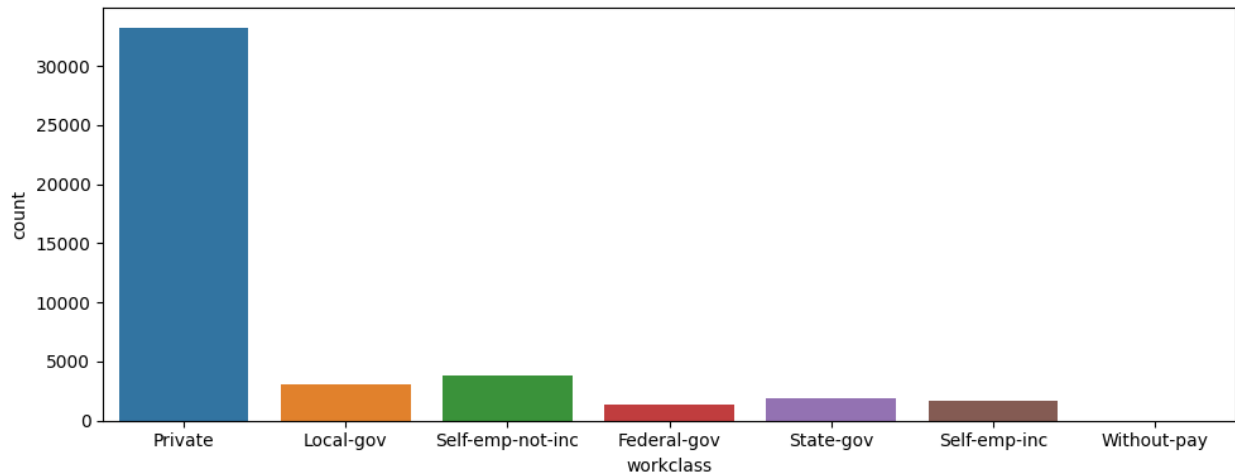
|   | income |
|---|--------|
| 0 | <=50K  |
| 1 | <=50K  |
| 2 | >50K   |
| 3 | >50K   |
| 5 | <=50K  |

```

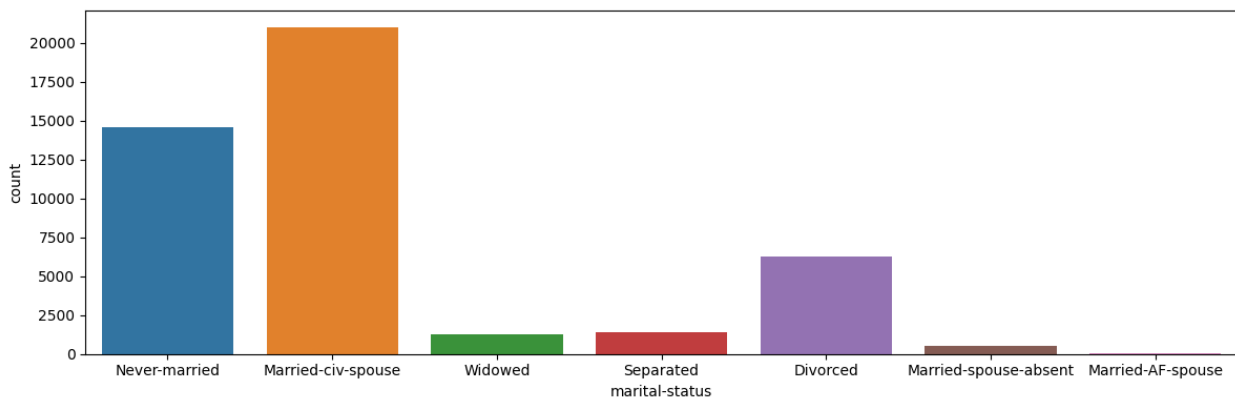
df.shape
(45175, 11)

plt.figure(figsize=(10, 4))
sns.countplot(df , x='workclass' )
plt.tight_layout()

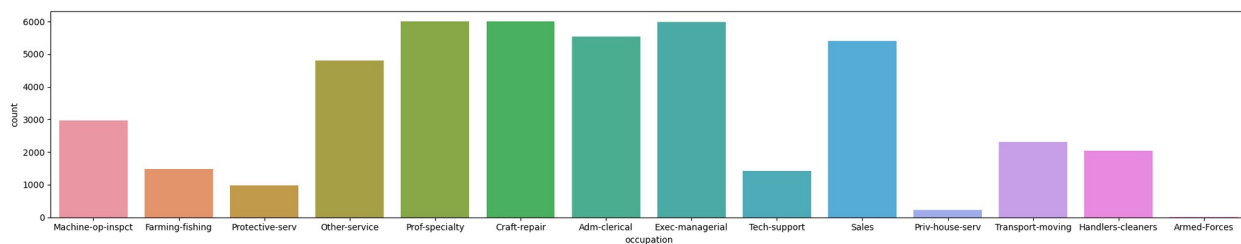
```



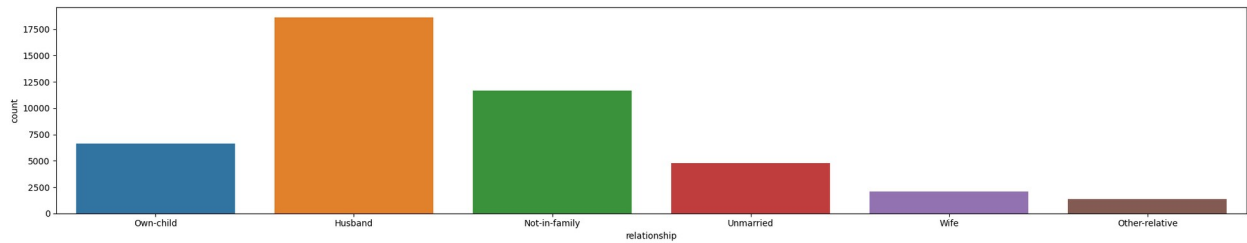
```
plt.figure(figsize=(12, 4))
sns.countplot(df , x='marital-status' )
plt.tight_layout()
```



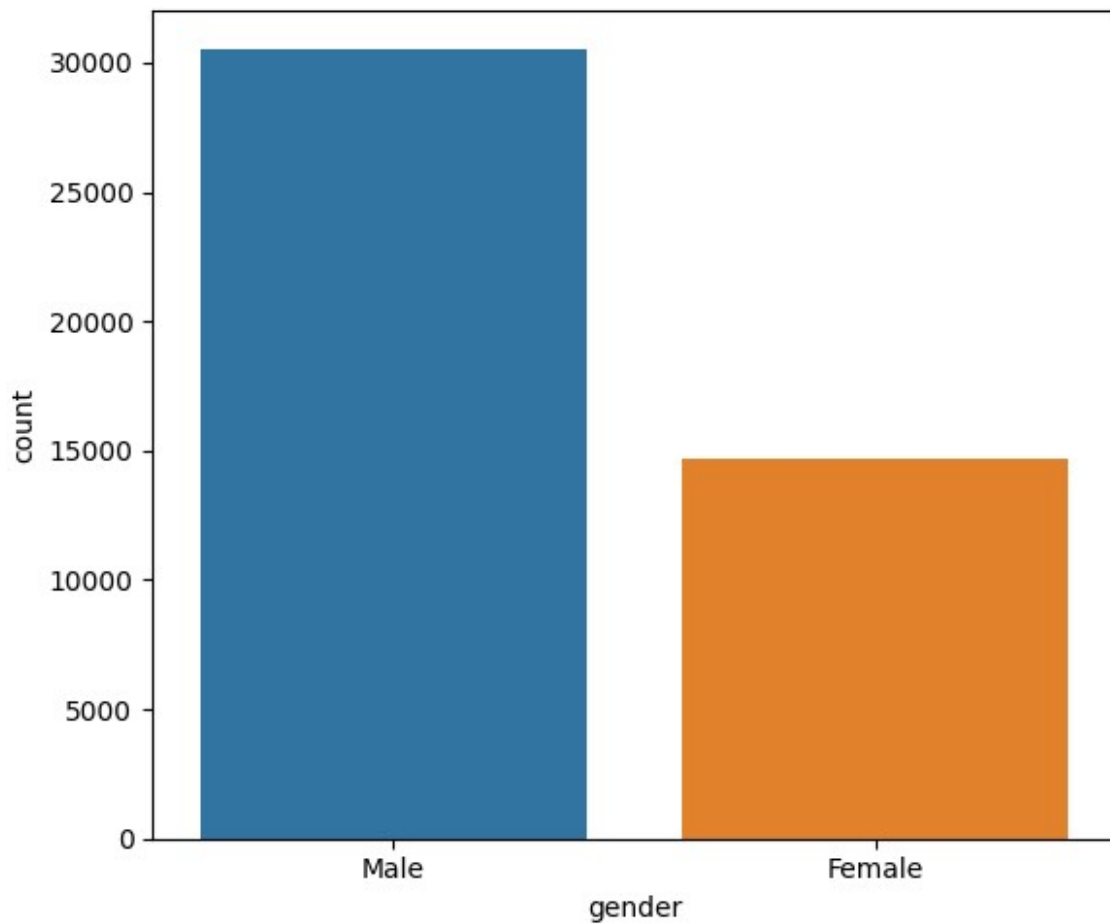
```
plt.figure(figsize=(20, 4))
sns.countplot(df , x='occupation' )
plt.tight_layout()
```



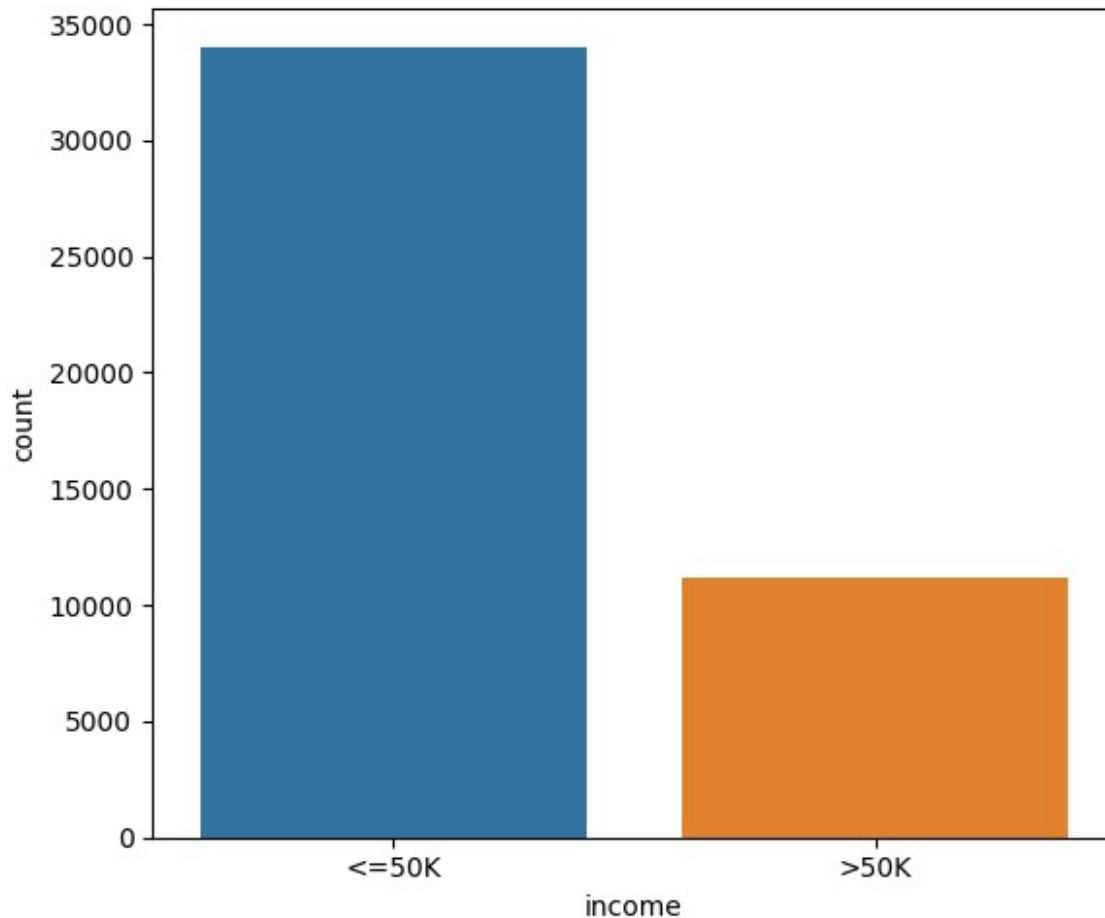
```
plt.figure(figsize=(20, 4))
sns.countplot(df , x='relationship')
plt.tight_layout()
```



```
plt.figure(figsize=(6, 5))  
sns.countplot(df , x='gender')  
plt.tight_layout()
```



```
plt.figure(figsize=(6, 5))  
sns.countplot(df , x='income')  
plt.tight_layout()
```



```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()

df['workclass'] = label_encoder.fit_transform(df['workclass'])
df['marital-status'] = label_encoder.fit_transform(df['marital-status'])
df['occupation'] = label_encoder.fit_transform(df['occupation'])
df['relationship'] = label_encoder.fit_transform(df['relationship'])
df['gender'] = label_encoder.fit_transform(df['gender'])
df['native-country'] = label_encoder.fit_transform(df['native-country'])
df['income'] = label_encoder.fit_transform(df['income'])

df['education'] = label_encoder.fit_transform(df['education'])
df.head()
```

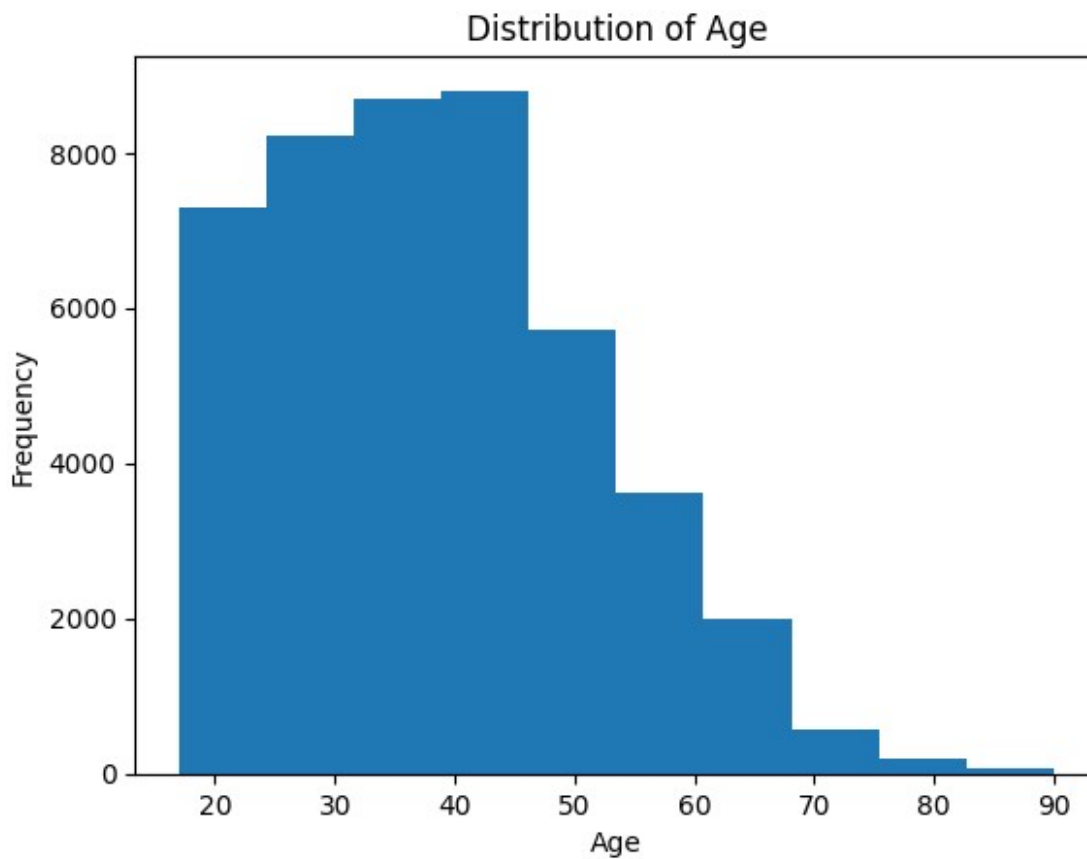
|   | age | workclass | education | educational-num | marital-status |
|---|-----|-----------|-----------|-----------------|----------------|
| 0 | 25  | 2         | 1         | 7               | 4              |
| 1 | 38  | 2         | 11        | 9               | 2              |

|    |    |   |    |    |   |
|----|----|---|----|----|---|
| 4  |    |   |    |    |   |
| 2  | 28 | 1 | 7  | 12 | 2 |
| 10 |    |   |    |    |   |
| 3  | 44 | 2 | 15 | 10 | 2 |
| 6  |    |   |    |    |   |
| 5  | 34 | 2 | 0  | 6  | 4 |
| 7  |    |   |    |    |   |

|   | relationship | gender | hours-per-week | native-country | income |
|---|--------------|--------|----------------|----------------|--------|
| 0 | 3            | 1      | 40             | 38             | 0      |
| 1 | 0            | 1      | 50             | 38             | 0      |
| 2 | 0            | 1      | 40             | 38             | 1      |
| 3 | 0            | 1      | 40             | 38             | 1      |
| 5 | 1            | 1      | 30             | 38             | 0      |

```
plt.hist(df['age'])
# add labels and title
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title('Distribution of Age')
Text(0.5, 1.0, 'Distribution of Age')
```



```
df.head()
```

```
   age  workclass  education  educational-num  marital-status
0    25         2         1             7         4
6
1    38         2        11             9         2
4
2    28         1         7            12         2
10
3    44         2        15            10         2
6
5    34         2         0             6         4
7
```

```
   relationship  gender  hours-per-week  native-country  income
0              3       1             40             38       0
1              0       1             50             38       0
2              0       1             40             38       1
3              0       1             40             38       1
5              1       1             30             38       0
```

```
df['income'] = df['income'].astype('category')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
Int64Index: 45175 entries, 0 to 48841
```

```
Data columns (total 11 columns):
```

| #  | Column          | Non-Null | Count    | Dtype    |
|----|-----------------|----------|----------|----------|
| 0  | age             | 45175    | non-null | int64    |
| 1  | workclass       | 45175    | non-null | int64    |
| 2  | education       | 45175    | non-null | int64    |
| 3  | educational-num | 45175    | non-null | int64    |
| 4  | marital-status  | 45175    | non-null | int64    |
| 5  | occupation      | 45175    | non-null | int64    |
| 6  | relationship    | 45175    | non-null | int64    |
| 7  | gender          | 45175    | non-null | int64    |
| 8  | hours-per-week  | 45175    | non-null | int64    |
| 9  | native-country  | 45175    | non-null | int64    |
| 10 | income          | 45175    | non-null | category |

```
dtypes: category(1), int64(10)
```

```
memory usage: 3.8 MB
```

```
#Train Test Split
```

```
from sklearn.model_selection import train_test_split
```

```
# Putting independent variables/features to X
```

```
X = df.drop('income',axis=1)
```

```
# Putting response/dependent variable/feature to y
```

```
y = df['income']
```

```
X.head()
```

|   | age | workclass | education | educational-num | marital-status |
|---|-----|-----------|-----------|-----------------|----------------|
| 0 | 25  | 2         | 1         | 7               | 4              |
| 1 | 38  | 2         | 11        | 9               | 2              |
| 2 | 28  | 1         | 7         | 12              | 2              |
| 3 | 44  | 2         | 15        | 10              | 2              |
| 5 | 34  | 2         | 0         | 6               | 4              |

|   | relationship | gender | hours-per-week | native-country |
|---|--------------|--------|----------------|----------------|
| 0 | 3            | 1      | 40             | 38             |
| 1 | 0            | 1      | 50             | 38             |
| 2 | 0            | 1      | 40             | 38             |
| 3 | 0            | 1      | 40             | 38             |
| 5 | 1            | 1      | 30             | 38             |

```
y.head()
```

|   |   |
|---|---|
| 0 | 0 |
| 1 | 0 |
| 2 | 1 |
| 3 | 1 |
| 5 | 0 |

```
Name: income, dtype: category
```

```
Categories (2, int64): [0, 1]
```

```
X_train,X_test,y_train,y_test =  
train_test_split(X,y,test_size=0.30,random_state=101)
```

```
X_train.head()
```

|       | age | workclass | education | educational-num | marital-status |
|-------|-----|-----------|-----------|-----------------|----------------|
| 47222 | 34  | 2         | 11        | 9               | 2              |
| 48391 | 33  | 2         | 11        | 9               | 0              |
| 30359 | 64  | 0         | 11        | 9               | 2              |



|       |    |   |   |   |   |
|-------|----|---|---|---|---|
| 22468 | 33 | 2 | 0 | 6 | 4 |
| 5     |    |   |   |   |   |
| 21211 | 34 | 1 | 6 | 5 | 4 |
| 7     |    |   |   |   |   |

|       | relationship | gender | hours-per-week | native-country |
|-------|--------------|--------|----------------|----------------|
| 47222 | 0            | 1      | 60             | 29             |
| 48391 | 1            | 1      | 60             | 38             |
| 30359 | 0            | 1      | 8              | 38             |
| 22468 | 4            | 1      | 40             | 38             |
| 21211 | 3            | 1      | 40             | 38             |

## Import model

```
from sklearn.tree import DecisionTreeClassifier
```

## Training model

```
dt_default = DecisionTreeClassifier(max_depth=5)
dt_default.fit(X_train,y_train)
```

```
DecisionTreeClassifier(max_depth=5)
```

## Checking Accuracy

```
from sklearn.metrics import
classification_report,confusion_matrix,accuracy_score
```

```
y_pred_default = dt_default.predict(X_test)
```

```
print(classification_report(y_test,y_pred_default))
```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0            | 0.83      | 0.94   | 0.89     | 10237   |
| 1            | 0.71      | 0.41   | 0.52     | 3316    |
| accuracy     |           |        | 0.81     | 13553   |
| macro avg    | 0.77      | 0.68   | 0.70     | 13553   |
| weighted avg | 0.80      | 0.81   | 0.80     | 13553   |

```
print(confusion_matrix(y_test,y_pred_default))
```

```
[[9635  540]
 [1955 1423]]
```