```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import pointbiserialr, spearmanr
from sklearn.feature_selection import SelectKBest
from sklearn.model_selection import cross_val_score
from sklearn.tree import DecisionTreeClassifier
```

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

	age	workclass	fnlwgt	education	education.num	marital.status	occupatio
0	90	?	77053	HS-grad	9	Widowed	
1	82	Private	132870	HS-grad	9	Widowed	Exec manageria
2	66	?	186061	Some-college	10	Widowed	
3	54	Private	140359	7th-8th	4	Divorced	Machine op-insp
4	41	Private	264663	Some-college	10	Separated	Pro specialt
...	...	...	...	...	...	...	.
32556	22	Private	310152	Some-college	10	Never-married	Protective ser
32557	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech suppo
32558	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine op-insp
32559	58	Private	151910	HS-grad	9	Widowed	Adm clerica
32560	22	Private	201490	HS-grad	9	Never-married	Adm clerica

32561 rows × 15 columns

Data Analysis

```
df.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
```

```
df.describe()
```

	age	fnlwgt	education.num	capital.gain	capital.loss	hours
count	32561.000000	3.256100e+04	32561.000000	32561.000000	32561.000000	32
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	

```
df.head()
```

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	r
0	90	?	77053	HS-grad	9	Widowed	?	
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	
2	66	?	186061	Some-college	10	Widowed	?	
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	

```
#checking total missing values in the whole dataset of attributes
df_missing = (df=='?').sum()
df_missing
```

```
age                 0
workclass        1836
fnlwgt              0
education           0
education.num       0
marital.status      0
occupation       1843
relationship        0
race                0
sex                 0
capital.gain        0
capital.loss        0
hours.per.week      0
native.country    583
income              0
dtype: int64
```

```
percent_missing = (df=='?').sum() * 100/len(df)
percent_missing
```

```
age               0.000000
workclass         5.638647
fnlwgt            0.000000
education         0.000000
education.num     0.000000
marital.status    0.000000
occupation        5.660146
relationship      0.000000
race              0.000000
sex               0.000000
capital.gain      0.000000
capital.loss      0.000000
hours.per.week    0.000000
native.country    1.790486
income            0.000000
dtype: float64
```

```
df.apply(lambda x: x !='?',axis=1).sum() #rows not containing "?"
```

```
age               32561
workclass         30725
fnlwgt            32561
education         32561
education.num     32561
marital.status    32561
occupation        30718
relationship      32561
race              32561
sex               32561
capital.gain      32561
capital.loss      32561
hours.per.week    32561
native.country    31978
income            32561
dtype: int64
```

```
df = df[df['workclass'] !='?']  #dropping missing value rows
df.head()
```

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	r
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	
5	34	Private	216864	HS-grad	9	Divorced	Other-service	
6	38	Private	150601	10th	6	Separated	Adm-clerical	

```
df_categorical = df.select_dtypes(include=['object'])
df_categorical.apply(lambda x: x=='?',axis=1).sum()
```

```
workclass           0
education           0
marital.status      0
occupation          7
relationship        0
race                0
sex                 0
native.country    556
income              0
dtype: int64
```

```
df = df[df['occupation'] !='?']
df = df[df['native.country'] !='?']
#final check for null values
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 30162 entries, 1 to 32560
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   age             30162 non-null  int64
 1   workclass       30162 non-null  object
 2   fnlwgt          30162 non-null  int64
 3   education       30162 non-null  object
 4   education.num   30162 non-null  int64
 5   marital.status  30162 non-null  object
 6   occupation      30162 non-null  object
 7   relationship    30162 non-null  object
 8   race            30162 non-null  object
 9   sex             30162 non-null  object
 10  capital.gain    30162 non-null  int64
 11  capital.loss    30162 non-null  int64
 12  hours.per.week  30162 non-null  int64
 13  native.country  30162 non-null  object
 14  income          30162 non-null  object
dtypes: int64(6), object(9)
memory usage: 3.7+ MB
```

Data preprocessing

```
from sklearn import preprocessing

# encoding categorical variables using label Encoder
df_categorical = df.select_dtypes(include=['object'])
df_categorical.head()
```

	workclass	education	marital.status	occupation	relationship	race	sex	r
1	Private	HS-grad	Widowed	Exec-managerial	Not-in-family	White	Female	
3	Private	7th-8th	Divorced	Machine-op-inspct	Unmarried	White	Female	
4	Private	Some-college	Separated	Prof-specialty	Own-child	White	Female	

```
le = preprocessing.LabelEncoder()
df_categorical = df_categorical.apply(le.fit_transform)
df_categorical.head()
```

	workclass	education	marital.status	occupation	relationship	race	sex	nati
1	2	11	6	3	1	4	0	
3	2	5	0	6	4	4	0	
4	2	15	5	9	3	4	0	
5	2	11	0	7	4	4	0	
6	2	0	5	0	4	4	1	

```
#dropping duplicate columns which had categorical values
df = df.drop(df_categorical.columns,axis=1)
df = pd.concat([df,df_categorical],axis=1)
df.head()
```

	age	fnlwgt	education.num	capital.gain	capital.loss	hours.per.week	workcla
1	82	132870	9	0	4356	18	
3	54	140359	4	0	3900	40	
4	41	264663	10	0	3900	40	
5	34	216864	9	0	3770	45	
6	38	150601	6	0	3770	40	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 30162 entries, 1 to 32560
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   age             30162 non-null  int64
 1   fnlwgt          30162 non-null  int64
 2   education.num   30162 non-null  int64
 3   capital.gain    30162 non-null  int64
 4   capital.loss    30162 non-null  int64
 5   hours.per.week  30162 non-null  int64
 6   workclass       30162 non-null  int64
 7   education       30162 non-null  int64
 8   marital.status  30162 non-null  int64
 9   occupation      30162 non-null  int64
 10  relationship    30162 non-null  int64
 11  race            30162 non-null  int64
 12  sex             30162 non-null  int64
 13  native.country  30162 non-null  int64
 14  income          30162 non-null  int64
dtypes: int64(15)
memory usage: 3.7 MB
```

```
# converting target variable income to categorical
df['income'] = df['income'].astype('category')
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 30725 entries, 1 to 32560
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype   
---  ------          --------------  -----   
 0   age             30725 non-null  int64   
 1   workclass       30725 non-null  object  
 2   fnlwgt          30725 non-null  int64   
 3   education       30725 non-null  object  
 4   education.num   30725 non-null  int64   
 5   marital.status  30725 non-null  object  
 6   occupation      30725 non-null  object  
 7   relationship    30725 non-null  object  
 8   race            30725 non-null  object  
 9   sex             30725 non-null  object  
 10  capital.gain    30725 non-null  int64   
 11  capital.loss    30725 non-null  int64   
 12  hours.per.week  30725 non-null  int64   
 13  native.country  30725 non-null  object  
 14  income          30725 non-null  category
dtypes: category(1), int64(6), object(8)
memory usage: 3.5+ MB
<ipython-input-56-f55635cbcbc0>:2: SettingWithCopyWarning: 
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-co
  df['income'] = df['income'].astype('category')
```

```
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(3)
```

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	r
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	

```
y.head(3)
```

```
1    <=50K
3    <=50K
4    <=50K
Name: income, dtype: category
Categories (2, object): ['<=50K', '>50K']
```

splitting the values in test & train datasets

```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.30,random_state=99)

X_train.head()
```

	age	workclass	fnlwgt	education	education.num	marital.status	occupatio
32493	51	Private	177669	Some-college	10	Married-civ-spouse	Sale
4725	20	Private	27337	HS-grad	9	Never-married	Handlers cleaner
6380	38	Private	258761	HS-grad	9	Divorced	Exec manageria
10782	43	Private	120277	HS-grad	9	Married-civ-spouse	Craft-repa
29936	21	Private	199419	Some-college	10	Never-married	Adm clerica

```
from sklearn.tree import DecisionTreeClassifier

# max_depth which is 5 so that we can plot and read the tree.
dt_default = DecisionTreeClassifier(max_depth=5)
dt_default.fit(X_train,y_train)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-60-6cfc0f7d38c9> in <cell line: 5>()
      3 # max_depth which is 5 so that we can plot and read the tree.
      4 dt_default = DecisionTreeClassifier(max_depth=5)
----> 5 dt_default.fit(X_train,y_train)

                              5 frames
/usr/local/lib/python3.10/dist-packages/pandas/core/generic.py in __array__(self,
dtype)
   2068
   2069     def __array__(self, dtype: npt.DTypeLike | None = None) -> np.ndarray:
-> 2070         return np.asarray(self._values, dtype=dtype)
   2071
   2072     def __array_wrap__(

ValueError: could not convert string to float: 'Private'
```

SEARCH STACK OVERFLOW

```
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))  #classifier
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-61-7c9dee1e4662> in <cell line: 3>()
      1 from sklearn.metrics import
classification_report,confusion_matrix,accuracy_score
      2
----> 3 y_pred_default = dt_default.predict(X_test)
      4 print(classification_report(y_test,y_pred_default))  #classifier

                              5 frames
/usr/local/lib/python3.10/dist-packages/pandas/core/generic.py in __array__(self,
dtype)
   2068
   2069     def __array__(self, dtype: npt.DTypeLike | None = None) -> np.ndarray:
-> 2070         return np.asarray(self._values, dtype=dtype)
   2071
   2072     def __array_wrap__(

ValueError: could not convert string to float: 'Private'
```

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

```
[[6553  314]
 [1039 1143]]
0.8504807161012267
```

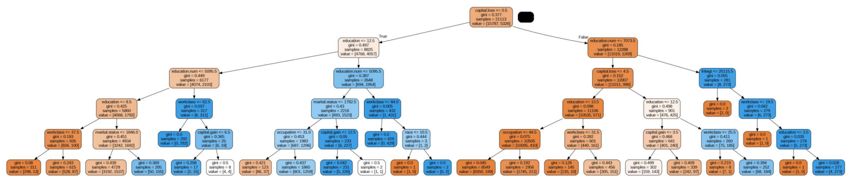
```
from IPython.display import Image
from six import StringIO
from sklearn.tree import export_graphviz
import pydotplus,graphviz

# Putting features
features = list(df.columns[1:])
features
```

```
['workclass',
 'fnlwgt',
 'education',
 'education.num',
 'marital.status',
 'occupation',
 'relationship',
 'race',
 'sex',
 'capital.gain',
 'capital.loss',
 'hours.per.week',
 'native.country',
 'income']
```

```
dot_data = StringIO()
export_graphviz(dt_default, out_file=dot_data,feature_names=features, filled=True,rounded=True)

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```


capital.loss <= 0.5 gini = 0.377 samples = 21113 value = [15787, 5326]
True
False
education <= 12.5 gini = 0.497 samples = 8825 value = [4768, 4057]
education.num <= 7073.5 gini = 0.185 samples = 12288 value = [11019, 1269]
education.num <= 5095.5 gini = 0.449 samples = 6177 value = [4074, 2103]
education.num <= 5095.5 gini = 0.387 samples = 2648 value = [694, 1954]
capital.loss <= 4.5 gini = 0.152 samples = 12007 value = [11011, 996]
fnlwgt <= 25115.5 gini = 0.055 samples = 281 value = [8, 273]
education <= 8.5 gini = 0.425 samples = 5860 value = [4068, 1792]
workclass <= 62.5 gini = 0.037 samples = 317 value = [6, 311]
marital.status <= 1782.5 gini = 0.43 samples = 2216 value = [693, 1523]
workclass <= 84.0 gini = 0.005 samples = 432 value = [1, 431]
education <= 13.5 gini = 0.098 samples = 11106 value = [10535, 571]
education <= 12.5 gini = 0.498 samples = 901 value = [476, 425]
gini = 0.0 samples = 2 value = [2, 0]
workclass <= 19.5 gini = 0.042 samples = 279 value = [6, 273]
workclass <= 37.5 gini = 0.193 samples = 926 value = [826, 100]
marital.status <= 1846.0 gini = 0.451 samples = 4934 value = [3242, 1692]
gini = 0.0 samples = 292 value = [0, 292]
capital.gain <= 6.5 gini = 0.365 samples = 25 value = [6, 19]
occupation <= 31.0 gini = 0.453 samples = 1983 value = [687, 1296]
capital.gain <= 12.5 gini = 0.05 samples = 233 value = [6, 227]
gini = 0.0 samples = 429 value = [0, 429]
race <= 10.5 gini = 0.444 samples = 3 value = [1, 2]
occupation <= 44.5 gini = 0.075 samples = 10505 value = [10095, 410]
workclass <= 31.5 gini = 0.392 samples = 601 value = [440, 161]
capital.gain <= 3.5 gini = 0.468 samples = 641 value = [401, 240]
workclass <= 25.5 gini = 0.411 samples = 260 value = [75, 185]
gini = 0.0 samples = 1 value = [1, 0]
education <= 3.5 gini = 0.035 samples = 278 value = [5, 273]
gini = 0.08 samples = 311 value = [298, 13]
gini = 0.243 samples = 615 value = [528, 87]
gini = 0.439 samples = 4729 value = [3192, 1537]
gini = 0.369 samples = 205 value = [50, 155]
gini = 0.208 samples = 17 value = [2, 15]
gini = 0.5 samples = 8 value = [4, 4]
gini = 0.421 samples = 123 value = [86, 37]
gini = 0.437 samples = 1860 value = [601, 1259]
gini = 0.042 samples = 231 value = [5, 226]
gini = 0.5 samples = 2 value = [1, 1]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 2 value = [0, 2]
gini = 0.045 samples = 8549 value = [8350, 199]
gini = 0.192 samples = 1956 value = [1745, 211]
gini = 0.128 samples = 145 value = [135, 10]
gini = 0.443 samples = 456 value = [305, 151]
gini = 0.499 samples = 302 value = [159, 143]
gini = 0.409 samples = 339 value = [242, 97]
gini = 0.219 samples = 8 value = [7, 1]
gini = 0.394 samples = 252 value = [68, 184]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.028 samples = 277 value = [4, 273]

Colab paid products - Cancel contracts here

0s completed at 10:49 AM