

In [1]:

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 %matplotlib inline
5 import seaborn as sns
6 from IPython import get_ipython
7 import warnings
8 warnings.filterwarnings("ignore")
```

In [2]:

```
1 data = pd.read_csv('adults.txt')
```

In [3]:

```
1 data.head()
```

Out[3]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship	race	gender
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female

In [4]:

```
1 data.tail()
```

Out[4]:

	age	workclass	fnlwgt	education	educational-num	marital-status	occupation	relationship	rac
32556	27	Private	257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	Whit
32557	40	Private	154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	Whit
32558	58	Private	151910	HS-grad	9	Widowed	Adm-clerical	Unmarried	Whit
32559	22	Private	201490	HS-grad	9	Never-married	Adm-clerical	Own-child	Whit
32560	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	Whit

In [5]:

```
1 data.shape
```

Out[5]:

(32561, 15)

In [6]:

```
1 data.columns
```

Out[6]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'educational-num',  
      'marital-status', 'occupation', 'relationship', 'race', 'gender',  
      'capital-gain', 'capital-loss', 'hours-per-week', 'native-country',  
      'income'],  
      dtype='object')
```

In [7]:



```
1 data.duplicated().sum()
```

Out[7]:

24

In [8]:



```
1 data = data.drop_duplicates()
```

In [9]:



```
1 data.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 [10]:



```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 32537 entries, 0 to 32560
Data columns (total 15 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   age                   32537 non-null  int64
 1   workclass             32537 non-null  object
 2   fnlwgt               32537 non-null  int64
 3   education            32537 non-null  object
 4   educational-num      32537 non-null  int64
 5   marital-status       32537 non-null  object
 6   occupation           32537 non-null  object
 7   relationship         32537 non-null  object
 8   race                 32537 non-null  object
 9   gender               32537 non-null  object
10   capital-gain         32537 non-null  int64
11   capital-loss         32537 non-null  int64
12   hours-per-week       32537 non-null  int64
13   native-country       32537 non-null  object
14   income               32537 non-null  object
dtypes: int64(6), object(9)
memory usage: 4.0+ MB
```

In [11]:



```
1 data.describe()
```

Out[11]:

	age	fnlwgt	educational-num	capital-gain	capital-loss	hours-per-week
count	32537.000000	3.253700e+04	32537.000000	32537.000000	32537.000000	32537.000000
mean	38.585549	1.897808e+05	10.081815	1078.443741	87.368227	40.440329
std	13.637984	1.055565e+05	2.571633	7387.957424	403.101833	12.346889
min	17.000000	1.228500e+04	1.000000	0.000000	0.000000	1.000000
25%	28.000000	1.178270e+05	9.000000	0.000000	0.000000	40.000000
50%	37.000000	1.783560e+05	10.000000	0.000000	0.000000	40.000000
75%	48.000000	2.369930e+05	12.000000	0.000000	0.000000	45.000000
max	90.000000	1.484705e+06	16.000000	99999.000000	4356.000000	99.000000

In [12]:



```
1 data.nunique()
```

Out[12]:

```
age                73
workclass           9
fnlwt             21648
education          16
educational-num    16
marital-status      7
occupation         15
relationship        6
race                5
gender              2
capital-gain       119
capital-loss        92
hours-per-week     94
native-country      42
income              2
dtype: int64
```

In [13]:



```
1 data = data.drop(['fnlwt', 'educational-num'], axis=1)
2 col_names = data.columns
3 for c in col_names:
4     data = data.replace("?", np.NaN)
5 data = data.apply(lambda x: x.fillna(x.value_counts().index[0]))
```

In [15]:



```
1 from sklearn import preprocessing
```

In [16]:

```

1 data.replace(['Divorced', 'Married-AF-spouse',
2             'Married-civ-spouse', 'Married-spouse-absent',
3             'Never-married', 'Separated', 'Widowed'],
4             ['divorced', 'married', 'married', 'married',
5             'not married', 'not married', 'not married'],
6             inplace=True)
7 category_col = ['workclass', 'race', 'education', 'marital-status',
8               'occupation', 'relationship', 'gender',
9               'native-country', 'income']
10 labelEncoder = preprocessing.LabelEncoder()
11 mapping_dict = {}
12 for col in category_col:
13     data[col] = labelEncoder.fit_transform(data[col])
14
15     le_name_mapping = dict(zip(labelEncoder.classes_,
16                               labelEncoder.transform(labelEncoder.classes_)))
17     mapping_dict[col] = le_name_mapping
18 print(mapping_dict)

```

```

{'workclass': {'?': 0, 'Federal-gov': 1, 'Local-gov': 2, 'Never-worked': 3, 'Private': 4, 'Self-emp-inc': 5, 'Self-emp-not-inc': 6, 'State-gov': 7, 'Without-pay': 8}, 'race': {'Amer-Indian-Eskimo': 0, 'Asian-Pac-Islander': 1, 'Black': 2, 'Other': 3, 'White': 4}, 'education': {'10th': 0, '11th': 1, '12th': 2, '1st-4th': 3, '5th-6th': 4, '7th-8th': 5, '9th': 6, 'Assoc-acdm': 7, 'Assoc-voc': 8, 'Bachelors': 9, 'Doctorate': 10, 'HS-grad': 11, 'Masters': 12, 'Preschool': 13, 'Prof-school': 14, 'Some-college': 15}, 'marital-status': {'Divorced': 0, 'Married-AF-spouse': 1, 'Married-civ-spouse': 2, 'Married-spouse-absent': 3, 'Never-married': 4, 'Separated': 5, 'Widowed': 6}, 'occupation': {'?': 0, 'Adm-clerical': 1, 'Armed-Forces': 2, 'Craft-repair': 3, 'Exec-managerial': 4, 'Farming-fishing': 5, 'Handlers-cleaners': 6, 'Machine-op-inspct': 7, 'Other-service': 8, 'Priv-house-serv': 9, 'Prof-specialty': 10, 'Protective-serv': 11, 'Sales': 12, 'Tech-support': 13, 'Transport-moving': 14}, 'relationship': {'Husband': 0, 'Not-in-family': 1, 'Other-relative': 2, 'Own-child': 3, 'Unmarried': 4, 'Wife': 5}, 'gender': {'Female': 0, 'Male': 1}, 'native-country': {'?': 0, 'Cambodia': 1, 'Canada': 2, 'China': 3, 'Columbia': 4, 'Cuba': 5, 'Dominican-Republic': 6, 'Ecuador': 7, 'El-Salvador': 8, 'England': 9, 'France': 10, 'Germany': 11, 'Greece': 12, 'Guatemala': 13, 'Haiti': 14, 'Holand-Netherlands': 15, 'Honduras': 16, 'Hong': 17, 'Hungary': 18, 'India': 19, 'Iran': 20, 'Ireland': 21, 'Italy': 22, 'Jamaica': 23, 'Japan': 24, 'Laos': 25, 'Mexico': 26, 'Nicaragua': 27, 'Outlying-US(Guam-USVI-etc)': 28, 'Peru': 29, 'Philippines': 30, 'Poland': 31, 'Portugal': 32, 'Puerto-Rico': 33, 'Scotland': 34, 'South': 35, 'Taiwan': 36, 'Thailand': 37, 'Trinidad&Tobago': 38, 'United-States': 39, 'Vietnam': 40, 'Yugoslavia': 41}, 'income': {'<=50K': 0, '>50K': 1}}

```

In [17]:

```

1 from sklearn.model_selection import train_test_split
2 from sklearn.tree import DecisionTreeClassifier
3 from sklearn.metrics import accuracy_score

```

In [18]:

```
1 X = data.values[:, 0:12]
2 Y = data.values[:, 12]
```

In [19]:

```
1 X_train, X_test, y_train, y_test = train_test_split(
2     X, Y, test_size = 0.3, random_state = 100)
3
4 dt_clf_gini = DecisionTreeClassifier(criterion = "gini",
5                                     random_state = 100,
6                                     max_depth = 5,
7                                     min_samples_leaf = 5)
8
9 dt_clf_gini.fit(X_train, y_train)
10 y_pred_gini = dt_clf_gini.predict(X_test)
11
12 print ("Decision Tree using Gini Index\nAccuracy is ",
13       accuracy_score(y_test, y_pred_gini)*100 )
```

Decision Tree using Gini Index
Accuracy is 82.67772997336611



In []:

```

1  <html>
2  <body>
3      <h3>Income Prediction Form</h3>
4  <div>
5  <form action="/result" method="POST">
6      <label for="age">Age</label>
7      <input type="text" id="age" name="age">
8      <br>
9      <label for="w_class">Working Class</label>
10     <select id="w_class" name="w_class">
11         <option value="0">Federal-gov</option>
12         <option value="1">Local-gov</option>
13         <option value="2">Never-worked</option>
14         <option value="3">Private</option>
15         <option value="4">Self-emp-inc</option>
16         <option value="5">Self-emp-not-inc</option>
17         <option value="6">State-gov</option>
18         <option value="7">Without-pay</option>
19     </select>
20     <br>
21     <label for="edu">Education</label>
22     <select id="edu" name="edu">
23         <option value="0">10th</option>
24         <option value="1">11th</option>
25         <option value="2">12th</option>
26         <option value="3">1st-4th</option>
27         <option value="4">5th-6th</option>
28         <option value="5">7th-8th</option>
29         <option value="6">9th</option>
30         <option value="7">Assoc-acdm</option>
31         <option value="8">Assoc-voc</option>
32         <option value="9">Bachelors</option>
33         <option value="10">Doctorate</option>
34         <option value="11">HS-grad</option>
35         <option value="12">Masters</option>
36         <option value="13">Preschool</option>
37         <option value="14">Prof-school</option>
38         <option value="15">16 - Some-college</option>
39     </select>
40     <br>
41     <label for="marital_stat">Marital Status</label>
42     <select id="marital_stat" name="marital_stat">
43         <option value="0">divorced</option>
44         <option value="1">married</option>
45         <option value="2">not married</option>
46     </select>
47     <br>
48     <label for="occup">Occupation</label>
49     <select id="occup" name="occup">
50         <option value="0">Adm-clerical</option>
51         <option value="1">Armed-Forces</option>
52         <option value="2">Craft-repair</option>
53         <option value="3">Exec-managerial</option>
54         <option value="4">Farming-fishing</option>
55         <option value="5">Handlers-cleaners</option>
56         <option value="6">Machine-op-inspect</option>
57         <option value="7">Other-service</option>
58         <option value="8">Priv-house-serv</option>
59         <option value="9">Prof-specialty</option>

```



```
60 <option value="10">Protective-serv</option>
61 <option value="11">Sales</option>
62 <option value="12">Tech-support</option>
63 <option value="13">Transport-moving</option>
64 </select>
65 <br>
66 <label for="relation">Relationship</label>
67 <select id="relation" name="relation">
68 <option value="0">Husband</option>
69 <option value="1">Not-in-family</option>
70 <option value="2">Other-relative</option>
71 <option value="3">Own-child</option>
72 <option value="4">Unmarried</option>
73 <option value="5">Wife</option>
74 </select>
75 <br>
76 <label for="race">Race</label>
77 <select id="race" name="race">
78 <option value="0">Amer Indian Eskimo</option>
79 <option value="1">Asian Pac Islander</option>
80 <option value="2">Black</option>
81 <option value="3">Other</option>
82 <option value="4">White</option>
83 </select>
84 <br>
85 <label for="gender">Gender</label>
86 <select id="gender" name="gender">
87 <option value="0">Female</option>
88 <option value="1">Male</option>
89 </select>
90 <br>
91 <label for="c_gain">Capital Gain </label>
92 <input type="text" id="c_gain" name="c_gain">btw:[0-99999]
93 <br>
94 <label for="c_loss">Capital Loss </label>
95 <input type="text" id="c_loss" name="c_loss">btw:[0-4356]
96 <br>
97 <label for="hours_per_week">Hours per Week </label>
98 <input type="text" id="hours_per_week" name="hours_per_week">btw:[1-99]
99 <br>
100 <label for="native-country">Native Country</label>
101 <select id="native-country" name="native-country">
102 <option value="0">Cambodia</option>
103 <option value="1">Canada</option>
104 <option value="2">China</option>
105 <option value="3">Columbia</option>
106 <option value="4">Cuba</option>
107 <option value="5">Dominican Republic</option>
108 <option value="6">Ecuador</option>
109 <option value="7">El Salvador</option>
110 <option value="8">England</option>
111 <option value="9">France</option>
112 <option value="10">Germany</option>
113 <option value="11">Greece</option>
114 <option value="12">Guatemala</option>
115 <option value="13">Haiti</option>
116 <option value="14">Netherlands</option>
117 <option value="15">Honduras</option>
118 <option value="16">HongKong</option>
119 <option value="17">Hungary</option>
120 <option value="18">India</option>
```

```
121     <option value="19">Iran</option>
122     <option value="20">Ireland</option>
123     <option value="21">Italy</option>
124     <option value="22">Jamaica</option>
125     <option value="23">Japan</option>
126     <option value="24">Laos</option>
127     <option value="25">Mexico</option>
128     <option value="26">Nicaragua</option>
129     <option value="27">Outlying-US(Guam-USVI-etc)</option>
130     <option value="28">Peru</option>
131     <option value="29">Philippines</option>
132     <option value="30">Poland</option>
133     <option value="11">Portugal</option>
134     <option value="32">Puerto-Rico</option>
135     <option value="33">Scotland</option>
136     <option value="34">South</option>
137     <option value="35">Taiwan</option>
138     <option value="36">Thailand</option>
139     <option value="37">Trinidad&Tobago</option>
140     <option value="38">United States</option>
141     <option value="39">Vietnam</option>
142     <option value="40">Yugoslavia</option>
143     </select>
144     <br>
145     <input type="submit" value="Submit">
146 </form>
147 </div>
148 </body>
149 </html>
```

In [22]:



```
1 from flask import *
2 app = Flask(__name__)
```

In [23]:



```
1 # prediction function
2 def ValuePredictor(to_predict_list):
3     to_predict = np.array(to_predict_list).reshape(1, 12)
4     loaded_model = pickle.load(open("model.pkl", "rb"))
5     result = loaded_model.predict(to_predict)
6     return result[0]
7
8 @app.route('/result', methods = ['POST'])
9 def result():
10     if request.method == 'POST':
11         to_predict_list = request.form.to_dict()
12         to_predict_list = list(to_predict_list.values())
13         to_predict_list = list(map(int, to_predict_list))
14         result = ValuePredictor(to_predict_list)
15         if int(result) == 1:
16             prediction = 'Income more than 50K'
17         else:
18             prediction = 'Income less than 50K'
19         return render_template("result.html", prediction = prediction)
20
21 if __name__ == '__main__':
22     app.run(debug = True)
```