titanic-survival-rate-analysis

April 26, 2024

1 Titanic Survival Rate

1.1 Imports

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import  missingno as msno
import seaborn as sns
```

1.2 Load Dataset

[2]:
```
df = pd.read_csv('titanic_train.csv')
df.head()
```

[2]:
```
   PassengerId  Survived  Pclass                                  Name   Sex  \
0          631         1       1  Barkworth, Mr. Algernon Henry Wilson  male
1          852         0       3                   Svensson, Mr. Johan  male
2           97         0       1             Goldschmidt, Mr. George B  male
3          494         0       1               Artagaveytia, Mr. Ramon  male
4          117         0       3                  Connors, Mr. Patrick  male

    Age  SibSp  Parch    Ticket     Fare Cabin Embarked
0  80.0      0      0     27042  30.0000   A23        S
1  74.0      0      0    347060   7.7750   NaN        S
2  71.0      0      0  PC 17754  34.6542    A5        C
3  71.0      0      0  PC 17609  49.5042   NaN        C
4  70.5      0      0    370369   7.7500   NaN        Q
```

Getting information about the dataset

[3]:
```
df.shape
```

[3]:
```
(891, 12)
```

[4]:
```
df.dtypes
```

```
[4]: PassengerId      int64
     Survived         int64
     Pclass           int64
     Name            object
     Sex             object
     Age            float64
     SibSp            int64
     Parch            int64
     Ticket          object
     Fare           float64
     Cabin           object
     Embarked        object
     dtype: object
```

```
[5]: df.describe()
```

```
[5]:        PassengerId    Survived      Pclass         Age       SibSp  \
     count   891.000000  891.000000  891.000000  891.000000  891.000000
     mean    446.000000    0.383838    2.308642   29.361582    0.523008
     std     257.353842    0.486592    0.836071   13.019697    1.102743
     min       1.000000    0.000000    1.000000    0.420000    0.000000
     25%     223.500000    0.000000    2.000000   22.000000    0.000000
     50%     446.000000    0.000000    3.000000   28.000000    0.000000
     75%     668.500000    1.000000    3.000000   35.000000    1.000000
     max     891.000000    1.000000    3.000000   80.000000    8.000000

                 Parch        Fare
     count  891.000000  891.000000
     mean     0.381594   32.204208
     std      0.806057   49.693429
     min      0.000000    0.000000
     25%      0.000000    7.910400
     50%      0.000000   14.454200
     75%      0.000000   31.000000
     max      6.000000  512.329200
```

1.3 Preprocessing

1.3.1 Handling Missing Values

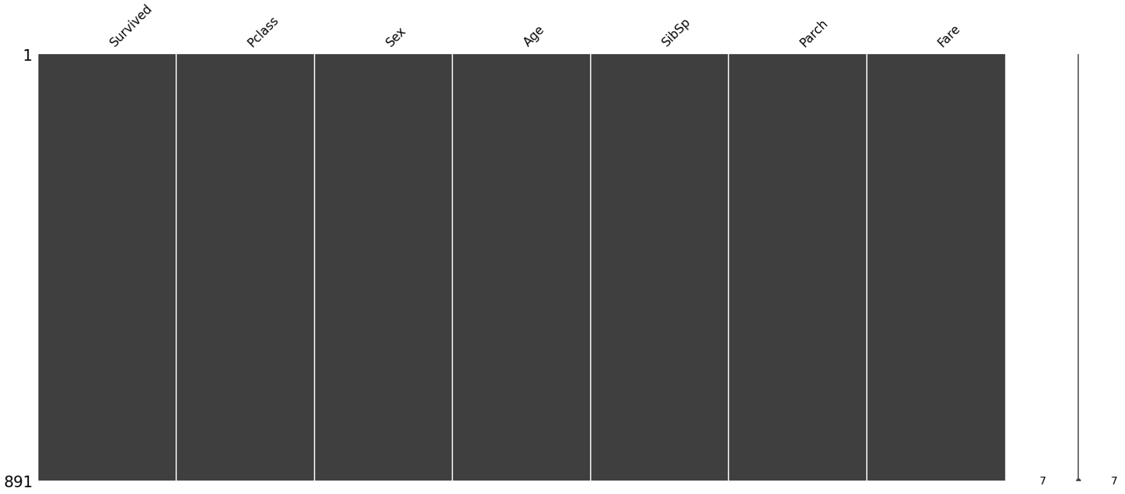
Removing irrelevant columns

```
[6]: df = df.drop(columns=['PassengerId', 'Name', 'Cabin', 'Ticket', 'Embarked'],␣
       ↪axis=1)
```

Check to see if there are any missing values

```
[7]: msno.matrix(df)
```

```
[7]: <Axes: >
```



```
[8]: df.isnull().sum()
```

```
[8]: Survived    0
     Pclass      0
     Sex         0
     Age         0
     SibSp       0
     Parch       0
     Fare        0
     dtype: int64
```

Binning the ages into categories. The pd.cut function is used to segregate array elements into different bins. Each bin is a range of ages, and they are labeled as 'Infant', 'Teen', '20s', '30s', '40s', '50s', and 'Elder'. This can be useful for later analyzing this feature

```
[9]: df['Age'] = df['Age'].astype(int)
```

```
[10]: df['Age'] = pd.cut(x = df['Age'], bins = [0, 5, 20, 30, 40, 50, 60, 100],␣
       ↪labels = ['Infant', 'Teen', '20s', '30s', '40s', '50s', 'Elder'])
```

1.3.2 Encoding Categorical Features

Encoding sex feature using map function

```
[12]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Sex       891 non-null    object
 3   Age       884 non-null    category
 4   SibSp     891 non-null    int64
 5   Parch     891 non-null    int64
 6   Fare      891 non-null    float64
dtypes: category(1), float64(1), int64(4), object(1)
memory usage: 43.1+ KB
```

[13]:

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

[13]:

```
array(['male', 'female'], dtype=object)
```

[14]:

```
df['Sex'] = df['Sex'].map({'male': 1, 'female': 0})
```

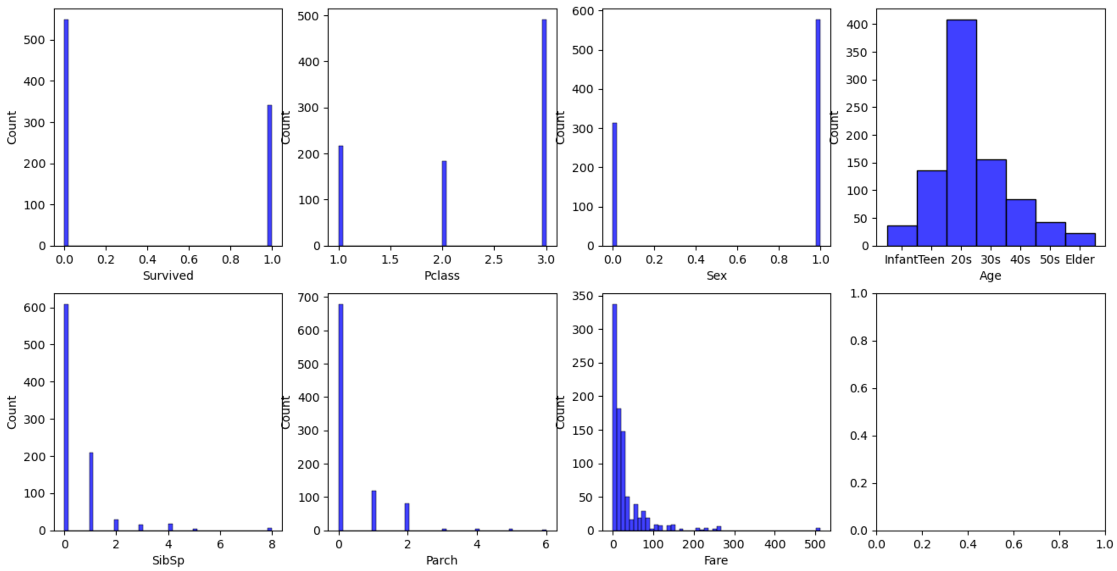
1.4 Exploratory Data Analysis (EDA)

1.4.1 Single Variable Analysis

Plotting histogram for each feature

[16]:

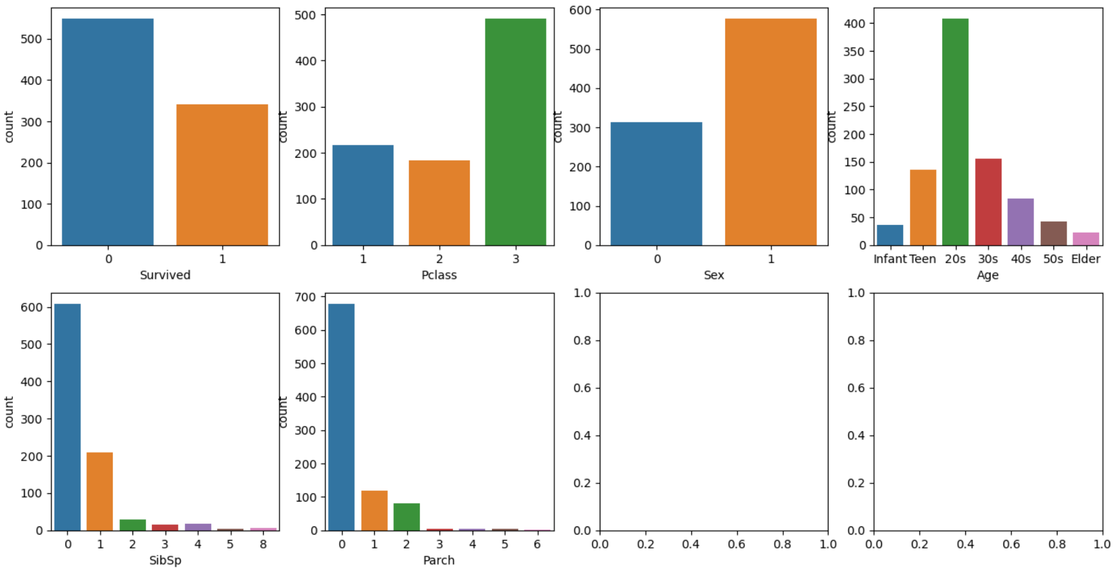
```
fig, axes = plt.subplots(2, 4, figsize=(16, 8))
axes_flat = axes.flatten()

for i, col in enumerate(df.columns):
    ax = axes_flat[i]
    sns.histplot(df[col], bins=50, color='b', ax=ax)
```



Plotting count plot for each feature

[28]:

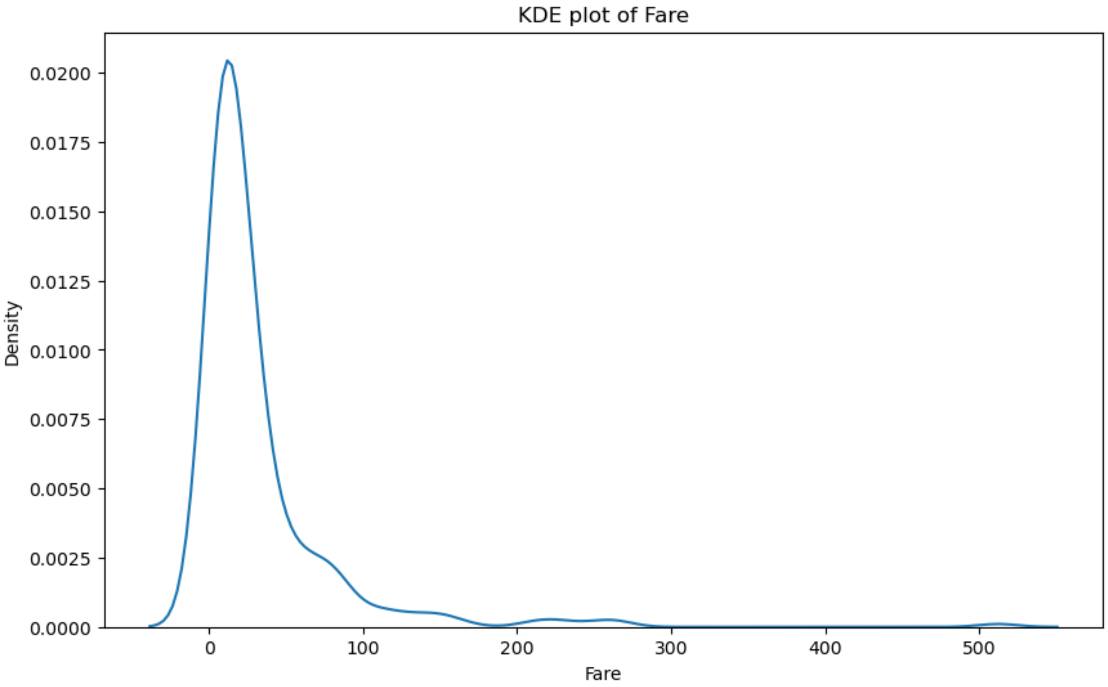
```
fig, axes = plt.subplots(2, 4, figsize = (16, 8))
axes_flat = axes.flatten()
columns_except_fare = [col for col in df.columns if col != 'Fare']

for i, col in enumerate(columns_except_fare):
    ax = axes_flat[i]
    sns.countplot(x = df[col], data = df, ax = ax)
```



Plotting KDE plot for 'Fare' feature

[33]:

```
plt.figure(figsize = (10,6))
sns.kdeplot(data = df, x = 'Fare')
plt.title('KDE plot of Fare')
plt.show()
```


KDE plot of Fare
Density
0.0000
0.0025
0.0050
0.0075
0.0100
0.0125
0.0150
0.0175
0.0200
0
100
200
300
400
500
Fare

1.4.2 Two Variable Analysis

1.4.3 1. Survived: This is the target variable we are trying to predict (0 = No; 1 = Yes). The correlation of other variables with ‘Survived’ gives an idea of their impact on survival rate. For instance, ‘Sex’ and ‘Survived’ seem to have a strong positive correlation, indicating that the gender of the passengers could have influenced their survival chances.

1.4.4 2. Pclass (Passenger Class): This is a proxy for socio-economic status (1 = 1st class (Upper); 2 = 2nd class (Middle); 3 = 3rd class (Lower)). It has a negative correlation with ‘Survived’, suggesting that lower class passengers were less likely to survive.

1.4.5 3. Sex: This is a binary variable (0 = Male; 1 = Female). It has a strong positive correlation with ‘Survived’, indicating that males had a higher survival rate than females.

1.4.6 4. Age: This is a continuous variable indicating the age of the passenger. The correlation between ‘Age’ and ‘Survived’ is not very strong, suggesting that age alone might not be a good predictor of survival.

1.4.7 5. SibSp: This variable indicates the number of siblings/spouses aboard. It doesn’t have a strong correlation with ‘Survived’, suggesting that the number of siblings/spouses aboard might not have a significant impact on survival.

1.4.8 6. Parch: This variable indicates the number of parents/children aboard. Like ‘SibSp’, ‘Parch’ also doesn’t have a strong correlation with ‘Survived’.

1.4.9 7. Fare: This is a continuous variable representing the passenger fare. It has a positive correlation with ‘Survived’, suggesting that passengers who paid higher fares were more likely to survive, possibly because they were in higher passenger classes.

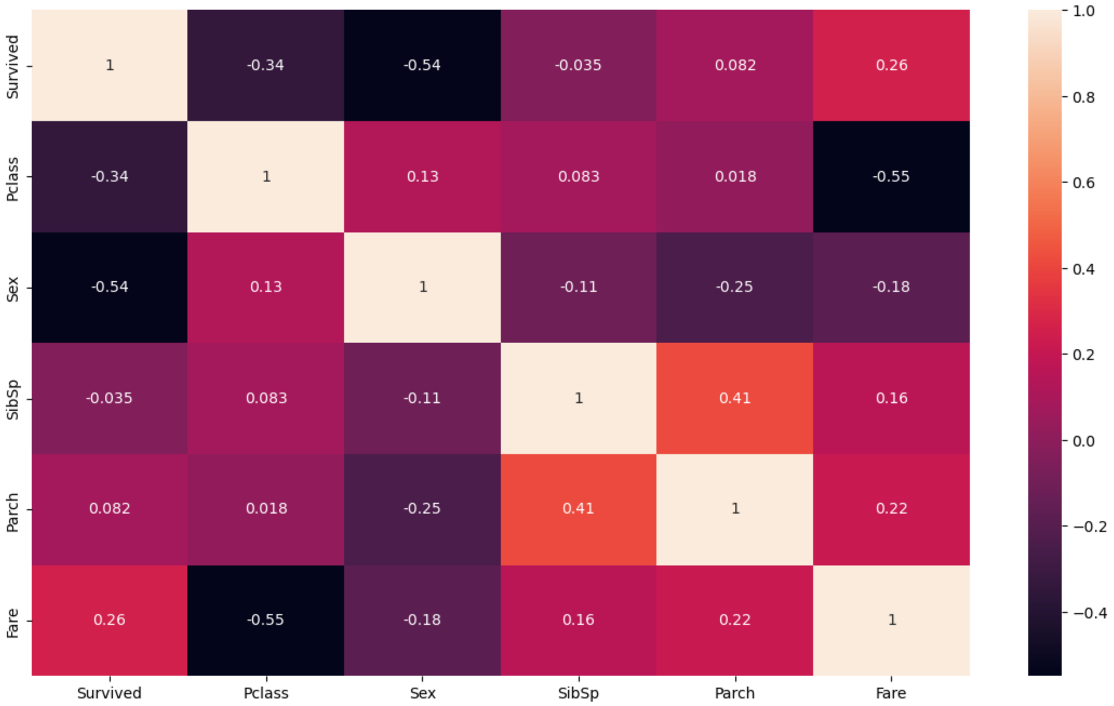
[18]:

```python
plt.figure(figsize=(14, 8))
corr = df.corr()
sns.heatmap(corr, annot=True)
```

```
C:\Users\niloo\AppData\Local\Temp\ipykernel_9032\92660599.py:2: FutureWarning:
The default value of numeric_only in DataFrame.corr is deprecated. In a future
version, it will default to False. Select only valid columns or specify the
value of numeric_only to silence this warning.
  corr = df.corr()
```

[18]:

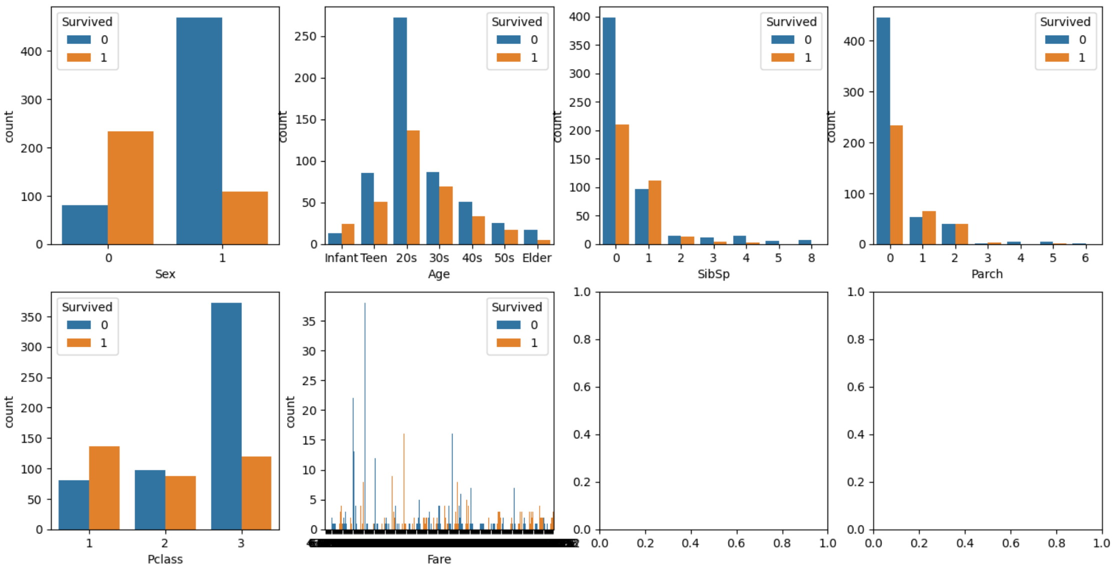
```
<Axes: >
```



[19]:
```
corr[['Survived']].abs().sort_values(by='Survived')
```

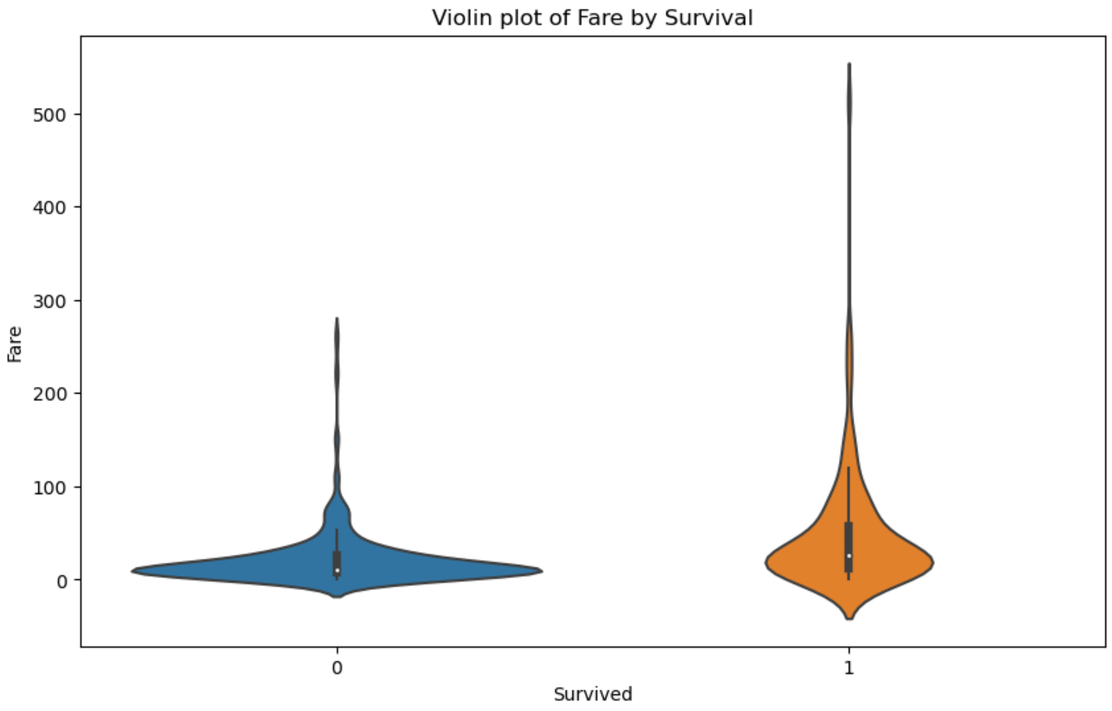
[19]:
```
          Survived
SibSp     0.035322
Parch     0.081629
Fare      0.257307
Pclass    0.338481
Sex       0.543351
Survived  1.000000
```

[30]:
```
cols = ['Sex', 'Age', 'SibSp', 'Parch', 'Pclass', 'Fare']

fig, axes = plt.subplots(2, 4, figsize = (16, 8))
axes_flat = axes.flatten()

for i, col in enumerate(cols):
    ax = axes_flat[i]
    sns.countplot(x = col, data = df, hue = 'Survived', ax = ax)
```



[29]:

```python
plt.figure(figsize=(10,6))
sns.violinplot(x = 'Survived', y = 'Fare', data = df)
plt.title('Violin plot of Fare by Survival')
plt.show()
```



Further preprcosseing the ‘Age’ feature

[34]:
```
age_mapping = {
    'infant': 0,
    'teen': 1,
    '20s': 2,
    '30s': 3,
    '40s': 4,
    '50s': 5,
    'elder': 6}

df['Age'] = df['Age'].map(age_mapping)
```

[35]:
```
df.dropna(subset=['Age'], axis= 0, inplace = True)
```

Splitting the df into X and y variables

[36]:
```
y = df['Survived']
x = df.drop(columns=['Survived'])
```

1.5 Model Training (ID3 from Scratch)

1.5.1 Utilies

[37]:
```
def entropy(labels):
    p = labels.value_counts() / len(labels)
    return -sum(p * np.log2(p))
```

[38]:
```
def information_gain(data, feature, target):

    # Entropy of parent
    entropy_parent = entropy(data[target])

    # Entropy of child
    entropy_child = 0
    for value in data[feature].unique():
        subset = data[data[feature] == value]
        wi = len(subset) / len(data)
        entropy_child += wi * entropy(subset[target])

    return entropy_parent - entropy_child
```

[46]:
```
target = 'Survived'
columns_except_target = [col for col in df.columns if col != target]
for col in columns_except_target:
    ig = information_gain(df, col, target)
    print(f'IG of {col} : {ig}')
```

```
IG of Pclass : 0.09858485512428761
IG of Sex : 0.24224166039520068
IG of Age : 0.0067630583143030165
IG of SibSp : 0.023079820377931992
IG of Parch : 0.017940673237646476
IG of Fare : 0.43126033782434336
```

In a decision tree, the attribute with the highest Information Gain (IG) is typically selected as the root node. In this case, the attribute 'Fare' has the highest IG, so it should be selected as the root node. The goal of using IG is to decide which feature splits the data best. We're looking for the feature that best separates the survivors from the non-survivors. In this case, it appears 'Fare' does this the best.

[49]:

```python
class Node:

    def __init__(self, feature = None, label = None):
        self.feature = feature
        self.label = label
        self.children = {}

    def __repr__(self):
        if self.feature is not None:
            return f'DecisionNode(feature = "{self.feature}", children = {self.
 ↪children})'
        else:
            return f'LeafNode(label = "{self.label}")'

Node(feature='Fare')
```

[49]:

```
DecisionNode(feature = "Fare", children = {})
```

[50]:

```python
def make_tree(data, target):

    # leaf node?
    if len(data[target].unique()) == 1:
        return Node(label = data[target].iloc[0])

    features = data.drop(target, axis=1).columns
    if len(features) == 0 or len(data) == 0:
        return Node(label=data[target].mode()[0])

    # calculate information gain
    gains = [information_gain(data, feature, target) for feature in features]

    # greedy search to find best feature
    max_gain_idx = np.argmax(gains)
    best_feature = features[max_gain_idx]

    # make a node
    node = Node(feature=best_feature)

    # loop over the best feature
    for value in data[best_feature].unique():
        subset = data[data[best_feature] == value].drop(best_feature, axis=1)

        node.children[value] = make_tree(subset, target)

    return node
```

[51]:
```
tree = make_tree(df, 'Survived')
tree
```

[51]:
```
DecisionNode(feature = "Fare", children = {75.25: LeafNode(label = "1"), 79.2:
DecisionNode(feature = "Age", children = {5.0: LeafNode(label = "1"), 4.0:
LeafNode(label = "0"), 2.0: DecisionNode(feature = "Sex", children = {0:
LeafNode(label = "1"), 1: LeafNode(label = "0")})}), 39.0: DecisionNode(feature
= "Sex", children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")}),
26.55: DecisionNode(feature = "Age", children = {5.0: DecisionNode(feature =
"Sex", children = {1: DecisionNode(feature = "Pclass", children = {1:
DecisionNode(feature = "SibSp", children = {0: DecisionNode(feature = "Parch",
children = {0: LeafNode(label = "0")})})}), 0: LeafNode(label = "1")}), 4.0:
DecisionNode(feature = "Pclass", children = {1: DecisionNode(feature = "Sex",
children = {1: DecisionNode(feature = "SibSp", children = {0:
DecisionNode(feature = "Parch", children = {0: LeafNode(label = "1")})})})}),
3.0: LeafNode(label = "1"), 2.0: DecisionNode(feature = "Pclass", children = {1:
DecisionNode(feature = "Sex", children = {1: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"1")})})})})}), 7.25: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "0"), 0: LeafNode(label = "1")}), 13.5: DecisionNode(feature =
"Sex", children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")}),
146.5208: LeafNode(label = "1"), 153.4625: DecisionNode(feature = "Sex",
children = {0: LeafNode(label = "1"), 1: LeafNode(label = "0")}), 29.7:
DecisionNode(feature = "Age", children = {5.0: LeafNode(label = "0"), 3.0:
LeafNode(label = "0"), 2.0: LeafNode(label = "1")}), 113.275:
DecisionNode(feature = "Sex", children = {1: LeafNode(label = "0"), 0:
LeafNode(label = "1")}), 12.35: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "0"), 0: LeafNode(label = "1")}), 10.5: DecisionNode(feature =
"Sex", children = {0: DecisionNode(feature = "Age", children = {5.0:
LeafNode(label = "0"), 4.0: LeafNode(label = "1"), 3.0: LeafNode(label = "1"),
2.0: LeafNode(label = "1")}), 1: LeafNode(label = "0")}), 30.6958:
LeafNode(label = "0"), 35.5: DecisionNode(feature = "Age", children = {5.0:
LeafNode(label = "1"), 4.0: LeafNode(label = "0"), 2.0: LeafNode(label = "1")}),
83.1583: LeafNode(label = "1"), 8.05: DecisionNode(feature = "Age", children =
{5.0: LeafNode(label = "0"), 4.0: DecisionNode(feature = "Pclass", children =
{3: DecisionNode(feature = "Sex", children = {1: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"0")})})})}), 3.0: DecisionNode(feature = "Pclass", children = {3:
DecisionNode(feature = "Sex", children = {1: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"0")})})})}), 2.0: LeafNode(label = "0")}), 16.0: LeafNode(label = "1"), 30.5:
DecisionNode(feature = "Age", children = {5.0: DecisionNode(feature = "Pclass",
children = {1: DecisionNode(feature = "Sex", children = {1: DecisionNode(feature
= "SibSp", children = {0: DecisionNode(feature = "Parch", children = {0:
LeafNode(label = "0")})})})})}), 3.0: LeafNode(label = "1"), 2.0: LeafNode(label =
"1")}), 51.8625: DecisionNode(feature = "Sex", children = {1: LeafNode(label =
"0"), 0: LeafNode(label = "1")}), 77.2875: LeafNode(label = "0"), 26.0:
DecisionNode(feature = "Sex", children = {1: DecisionNode(feature = "Age",
children = {5.0: LeafNode(label = "0"), 4.0: LeafNode(label = "0"), 3.0:
DecisionNode(feature = "SibSp", children = {0: LeafNode(label = "0"), 1:
DecisionNode(feature = "Pclass", children = {2: DecisionNode(feature = "Parch",
children = {0: LeafNode(label = "0")})})})}), 2.0: LeafNode(label = "0")}), 0:
DecisionNode(feature = "SibSp", children = {0: LeafNode(label = "1"), 1:
DecisionNode(feature = "Parch", children = {0: DecisionNode(feature = "Age",
children = {4.0: DecisionNode(feature = "Pclass", children = {2: LeafNode(label
= "0")}), 3.0: LeafNode(label = "1"), 2.0: LeafNode(label = "1")}), 1:
LeafNode(label = "0")})})}), 14.0: LeafNode(label = "0"), 78.2667:
LeafNode(label = "1"), 59.4: LeafNode(label = "1"), 23.0: LeafNode(label = "1"),
51.4792: LeafNode(label = "1"), 79.65: DecisionNode(feature = "Sex", children =
{1: LeafNode(label = "0"), 0: LeafNode(label = "1")}), 13.0:
DecisionNode(feature = "Sex", children = {1: DecisionNode(feature = "Age",
children = {5.0: LeafNode(label = "0"), 4.0: DecisionNode(feature = "Pclass",
children = {2: DecisionNode(feature = "SibSp", children = {0:
DecisionNode(feature = "Parch", children = {0: LeafNode(label = "0")})})})}), 3.0:
DecisionNode(feature = "Pclass", children = {2: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"0")})})}), 2.0: DecisionNode(feature = "Pclass", children = {2:
DecisionNode(feature = "SibSp", children = {0: DecisionNode(feature = "Parch",
children = {0: LeafNode(label = "0")})})})}), 0: DecisionNode(feature = "Age",
children = {4.0: LeafNode(label = "1"), 3.0: DecisionNode(feature = "Pclass",
children = {2: DecisionNode(feature = "SibSp", children = {0:
DecisionNode(feature = "Parch", children = {0: LeafNode(label = "1")})})})}), 2.0:
DecisionNode(feature = "Pclass", children = {2: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"1")})})})})}), 93.5: LeafNode(label = "1"), 12.525: LeafNode(label = "0"),
61.3792: LeafNode(label = "0"), 7.75: DecisionNode(feature = "Sex", children =
{1: DecisionNode(feature = "Age", children = {5.0: LeafNode(label = "0"), 3.0:
LeafNode(label = "0"), 2.0: DecisionNode(feature = "SibSp", children = {0:
DecisionNode(feature = "Pclass", children = {3: DecisionNode(feature = "Parch",
children = {0: LeafNode(label = "0")})})}), 1: LeafNode(label = "0")})}), 0:
DecisionNode(feature = "Age", children = {4.0: LeafNode(label = "0"), 2.0:
DecisionNode(feature = "Parch", children = {0: DecisionNode(feature = "Pclass",
children = {3: DecisionNode(feature = "SibSp", children = {0: LeafNode(label =
"1")})})}), 2: LeafNode(label = "0")})})}), 7.0542: LeafNode(label = "0"),
77.9583: LeafNode(label = "1"), 28.7125: LeafNode(label = "0"), 247.5208:
DecisionNode(feature = "Sex", children = {0: LeafNode(label = "1"), 1:
LeafNode(label = "0")}), 55.9: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "0"), 0: LeafNode(label = "1")}), 106.425: DecisionNode(feature
= "Sex", children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")}),
133.65: LeafNode(label = "1"), 76.7292: LeafNode(label = "1"), 89.1042:
LeafNode(label = "1"), 0.0: DecisionNode(feature = "Pclass", children = {3:
DecisionNode(feature = "Age", children = {4.0: LeafNode(label = "0"), 3.0:
LeafNode(label = "0"), 2.0: LeafNode(label = "1")}), 1: LeafNode(label = "0"),
2: LeafNode(label = "0")}), 56.9292: LeafNode(label = "1"), 110.8833:
DecisionNode(feature = "Sex", children = {1: LeafNode(label = "0"), 0:
LeafNode(label = "1")}), 25.9292: LeafNode(label = "1"), 39.6:
DecisionNode(feature = "Sex", children = {0: LeafNode(label = "1"), 1:
LeafNode(label = "0")}), 52.0: DecisionNode(feature = "Sex", children = {1:
DecisionNode(feature = "Age", children = {4.0: DecisionNode(feature = "SibSp",
children = {1: DecisionNode(feature = "Pclass", children = {1:
DecisionNode(feature = "Parch", children = {0: LeafNode(label = "0")})})}), 0:
LeafNode(label = "0")}), 3.0: LeafNode(label = "0"), 2.0: LeafNode(label =
"0")}), 0: LeafNode(label = "1")}), 34.375: LeafNode(label = "0"), 65.0:
LeafNode(label = "1"), 7.8542: DecisionNode(feature = "Age", children = {4.0:
LeafNode(label = "0"), 3.0: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "1"), 0: LeafNode(label = "0")}), 2.0: DecisionNode(feature =
"Sex", children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")})}), 14.5:
DecisionNode(feature = "Pclass", children = {3: LeafNode(label = "0"), 2:
LeafNode(label = "1")}), 38.5: LeafNode(label = "0"), 34.0208: LeafNode(label =
"0"), 15.0: LeafNode(label = "0"), 25.5875: LeafNode(label = "0"), 52.5542:
LeafNode(label = "1"), 9.0: LeafNode(label = "0"), 61.175: LeafNode(label =
"0"), 7.225: DecisionNode(feature = "Sex", children = {1: DecisionNode(feature =
"Age", children = {4.0: LeafNode(label = "0"), 3.0: LeafNode(label = "0"), 2.0:
DecisionNode(feature = "Pclass", children = {3: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"0")})})})}), 0: LeafNode(label = "1")}), 28.5: LeafNode(label = "0"), 83.475:
DecisionNode(feature = "Sex", children = {1: LeafNode(label = "0"), 0:
LeafNode(label = "1")}), 6.975: DecisionNode(feature = "Age", children = {4.0:
LeafNode(label = "0"), 2.0: LeafNode(label = "1")}), 27.9: LeafNode(label =
"0"), 14.4542: LeafNode(label = "0"), 26.25: DecisionNode(feature = "Sex",
children = {0: LeafNode(label = "1"), 1: LeafNode(label = "0")}), 164.8667:
LeafNode(label = "1"), 16.1: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "0"), 0: DecisionNode(feature = "Pclass", children = {3:
DecisionNode(feature = "Age", children = {2.0: DecisionNode(feature = "SibSp",
children = {1: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"1")})})})})}), 27.7208: DecisionNode(feature = "Sex", children = {0:
LeafNode(label = "1"), 1: LeafNode(label = "0")}), 90.0: DecisionNode(feature =
"Age", children = {4.0: LeafNode(label = "0"), 3.0: LeafNode(label = "1")}),
7.925: DecisionNode(feature = "SibSp", children = {0: DecisionNode(feature =
"Age", children = {4.0: LeafNode(label = "1"), 3.0: DecisionNode(feature =
"Pclass", children = {3: DecisionNode(feature = "Sex", children = {1:
```

```
DecisionNode(feature = "Parch", children = {0: LeafNode(label = "0")})})}), 2.0:
DecisionNode(feature = "Sex", children = {0: DecisionNode(feature = "Pclass",
children = {3: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"1")})}), 1: LeafNode(label = "0")})}), 2: LeafNode(label = "0"), 1:
LeafNode(label = "0")}), 57.9792: LeafNode(label = "1"), 46.9: LeafNode(label =
"0"), 211.3375: LeafNode(label = "1"), 6.45: LeafNode(label = "0"), 8.4042:
LeafNode(label = "0"), 27.0: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "0"), 0: LeafNode(label = "1")}), 8.6625: DecisionNode(feature
= "Age", children = {4.0: LeafNode(label = "0"), 3.0: LeafNode(label = "0"),
2.0: DecisionNode(feature = "Sex", children = {0: LeafNode(label = "0"), 1:
DecisionNode(feature = "SibSp", children = {0: DecisionNode(feature = "Pclass",
children = {3: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"0")})}), 2: LeafNode(label = "0")})})}), 227.525: DecisionNode(feature = "Sex",
children = {0: LeafNode(label = "1"), 1: LeafNode(label = "0")}), 7.65:
DecisionNode(feature = "Sex", children = {1: LeafNode(label = "0"), 0:
LeafNode(label = "1")}), 26.2875: LeafNode(label = "1"), 7.55:
DecisionNode(feature = "Sex", children = {1: LeafNode(label = "0"), 0:
DecisionNode(feature = "Pclass", children = {3: DecisionNode(feature = "Age",
children = {2.0: DecisionNode(feature = "SibSp", children = {0:
DecisionNode(feature = "Parch", children = {0: LeafNode(label = "0")})})})})}),
20.2125: LeafNode(label = "0"), 19.5: LeafNode(label = "1"), 134.5:
LeafNode(label = "1"), 39.6875: LeafNode(label = "0"), 7.125: LeafNode(label =
"0"), 14.1083: LeafNode(label = "0"), 9.475: LeafNode(label = "0"), 15.75:
LeafNode(label = "1"), 15.5: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "0"), 0: DecisionNode(feature = "Age", children = {3.0:
LeafNode(label = "0"), 2.0: LeafNode(label = "1")})}), 31.0:
DecisionNode(feature = "Sex", children = {1: DecisionNode(feature = "Age",
children = {3.0: LeafNode(label = "1"), 2.0: LeafNode(label = "0")}), 0:
LeafNode(label = "1")}), 7.8958: DecisionNode(feature = "Age", children = {3.0:
LeafNode(label = "0"), 2.0: DecisionNode(feature = "Sex", children = {1:
DecisionNode(feature = "Pclass", children = {3: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"0")})})}), 0: LeafNode(label = "0")})}), 31.275: LeafNode(label = "0"), 24.15:
DecisionNode(feature = "Sex", children = {1: LeafNode(label = "0"), 0:
DecisionNode(feature = "Parch", children = {1: LeafNode(label = "0"), 0:
LeafNode(label = "1")})}), 29.125: LeafNode(label = "0"), 71.2833:
LeafNode(label = "1"), 31.3875: LeafNode(label = "1"), 80.0: LeafNode(label =
"1"), 7.05: LeafNode(label = "0"), 53.1: DecisionNode(feature = "Sex", children
= {1: DecisionNode(feature = "Age", children = {3.0: LeafNode(label = "0"), 2.0:
LeafNode(label = "1")}), 0: LeafNode(label = "1")}), 9.5875: LeafNode(label =
"0"), 12.875: LeafNode(label = "0"), 135.6333: DecisionNode(feature = "Sex",
children = {0: LeafNode(label = "1"), 1: LeafNode(label = "0")}), 120.0:
LeafNode(label = "1"), 27.75: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "0"), 0: LeafNode(label = "1")}), 71.0: LeafNode(label = "1"),
17.4: LeafNode(label = "1"), 26.3875: LeafNode(label = "1"), 40.125:
LeafNode(label = "0"), 15.55: LeafNode(label = "0"), 7.4958: LeafNode(label =
"0"), 512.3292: LeafNode(label = "1"), 78.85: DecisionNode(feature = "Sex",
children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")}), 21.0:
DecisionNode(feature = "SibSp", children = {0: LeafNode(label = "1"), 1:
LeafNode(label = "0"), 3: LeafNode(label = "1")}), 20.25: LeafNode(label = "1"),
6.4375: LeafNode(label = "0"), 6.4958: LeafNode(label = "0"), 32.5:
LeafNode(label = "1"), 14.4: LeafNode(label = "0"), 15.85: DecisionNode(feature
= "Sex", children = {0: LeafNode(label = "1"), 1: LeafNode(label = "0")}),
8.6542: LeafNode(label = "0"), 12.275: LeafNode(label = "0"), 20.525:
DecisionNode(feature = "Sex", children = {1: LeafNode(label = "0"), 0:
LeafNode(label = "1")}), 7.775: DecisionNode(feature = "Age", children = {3.0:
LeafNode(label = "0"), 2.0: DecisionNode(feature = "Sex", children = {1:
DecisionNode(feature = "SibSp", children = {0: DecisionNode(feature = "Pclass",
children = {3: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"0")})}), 1: DecisionNode(feature = "Pclass", children = {3:
DecisionNode(feature = "Parch", children = {0: LeafNode(label = "0")})})}), 0:
DecisionNode(feature = "Pclass", children = {3: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"0")})})})})}), 9.5: DecisionNode(feature = "Age", children = {3.0:
LeafNode(label = "0"), 2.0: DecisionNode(feature = "Pclass", children = {3:
DecisionNode(feature = "Sex", children = {1: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"0")})})})})}), 86.5: LeafNode(label = "1"), 5.0: LeafNode(label = "0"),
30.0708: LeafNode(label = "0"), 56.4958: DecisionNode(feature = "Age", children
= {3.0: LeafNode(label = "1"), 2.0: DecisionNode(feature = "Pclass", children =
{3: DecisionNode(feature = "Sex", children = {1: DecisionNode(feature = "SibSp",
children = {0: DecisionNode(feature = "Parch", children = {0: LeafNode(label =
"1")})})})})}), 76.2917: LeafNode(label = "1"), 73.5: LeafNode(label = "0"),
8.3625: LeafNode(label = "0"), 18.0: LeafNode(label = "0"), 57.0: LeafNode(label
= "1"), 8.6833: LeafNode(label = "1"), 37.0042: LeafNode(label = "0"), 50.4958:
LeafNode(label = "0"), 12.475: LeafNode(label = "1"), 24.0: DecisionNode(feature
= "Sex", children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")}),
7.2292: DecisionNode(feature = "Sex", children = {1: DecisionNode(feature =
"Pclass", children = {3: DecisionNode(feature = "Age", children = {2.0:
DecisionNode(feature = "SibSp", children = {0: DecisionNode(feature = "Parch",
children = {0: LeafNode(label = "0")})})})}), 0: LeafNode(label = "1")}),
10.4625: LeafNode(label = "0"), 15.2458: DecisionNode(feature = "Sex", children
= {0: DecisionNode(feature = "Pclass", children = {3: DecisionNode(feature =
"Age", children = {2.0: DecisionNode(feature = "SibSp", children = {0:
DecisionNode(feature = "Parch", children = {2: LeafNode(label = "0")})})})}), 1:
LeafNode(label = "1")}), 66.6: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "0"), 0: LeafNode(label = "1")}), 7.875: LeafNode(label = "0"),
7.0458: LeafNode(label = "0"), 21.075: LeafNode(label = "0"), 9.4833:
LeafNode(label = "0"), 30.0: DecisionNode(feature = "Pclass", children = {1:
DecisionNode(feature = "Sex", children = {1: DecisionNode(feature = "Age",
children = {2.0: DecisionNode(feature = "SibSp", children = {0:
DecisionNode(feature = "Parch", children = {0: LeafNode(label = "1")})})})}), 2:
LeafNode(label = "1")}), 82.1708: DecisionNode(feature = "Sex", children = {1:
LeafNode(label = "0"), 0: LeafNode(label = "1")}), 47.1: LeafNode(label = "0"),
12.65: LeafNode(label = "1"), 22.525: LeafNode(label = "0"), 7.7958:
DecisionNode(feature = "Pclass", children = {3: DecisionNode(feature = "Sex",
children = {1: DecisionNode(feature = "Age", children = {2.0:
DecisionNode(feature = "SibSp", children = {0: DecisionNode(feature = "Parch",
children = {0: LeafNode(label = "0")})})})})})}), 33.0: DecisionNode(feature =
"Sex", children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")}), 8.4583:
LeafNode(label = "0"), 7.8792: LeafNode(label = "1"), 21.6792: LeafNode(label =
"0"), 7.7875: LeafNode(label = "1"), 22.3583: LeafNode(label = "1"), 7.3125:
LeafNode(label = "0"), 69.55: LeafNode(label = "0"), 55.0: LeafNode(label =
"1"), 25.925: LeafNode(label = "0"), 25.4667: LeafNode(label = "0"), 15.05:
LeafNode(label = "0"), 50.0: LeafNode(label = "0"), 23.25: LeafNode(label =
"1"), 35.0: LeafNode(label = "0"), 7.7292: LeafNode(label = "0"), 6.8583:
LeafNode(label = "0"), 8.1125: LeafNode(label = "1"), 19.9667: LeafNode(label =
"0"), 7.725: LeafNode(label = "0"), 14.4583: LeafNode(label = "0"), 15.1:
LeafNode(label = "0"), 7.6292: LeafNode(label = "0"), 221.7792: LeafNode(label =
"0"), 13.8625: LeafNode(label = "1"), 7.8292: DecisionNode(feature = "Sex",
children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")}), 8.7125:
LeafNode(label = "0"), 42.4: LeafNode(label = "0"), 7.7333: DecisionNode(feature
= "Sex", children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")}),
8.1375: LeafNode(label = "0"), 7.7375: DecisionNode(feature = "Sex", children =
{0: LeafNode(label = "1"), 1: LeafNode(label = "0")}), 23.45: LeafNode(label =
"0"), 6.95: LeafNode(label = "0"), 11.1333: LeafNode(label = "1"), 211.5:
LeafNode(label = "0"), 13.8583: LeafNode(label = "1"), 20.575: LeafNode(label =
"0"), 18.7875: LeafNode(label = "1"), 7.8875: LeafNode(label = "0"), 17.8:
LeafNode(label = "0"), 55.4417: LeafNode(label = "1"), 91.0792: LeafNode(label =
"1"), 151.55: DecisionNode(feature = "SibSp", children = {1: LeafNode(label =
"0"), 0: LeafNode(label = "1")}), 41.5792: DecisionNode(feature = "Sex",
children = {1: LeafNode(label = "0"), 0: LeafNode(label = "1")}), 7.7417:
LeafNode(label = "0"), 7.1417: LeafNode(label = "1"), 8.85: LeafNode(label =
"0"), 263.0: LeafNode(label = "1"), 69.3: LeafNode(label = "1"), 16.7:
LeafNode(label = "1"), 18.75: LeafNode(label = "1"), 49.5042: LeafNode(label =
"1"), 19.2583: LeafNode(label = "1"), 63.3583: LeafNode(label = "1"), 15.0458:
LeafNode(label = "0"), 9.225: LeafNode(label = "0"), 13.7917: LeafNode(label =
"1"), 11.5: LeafNode(label = "0"), 9.35: LeafNode(label = "0"), 29.0:
LeafNode(label = "1"), 9.8375: LeafNode(label = "0"), 7.5208: LeafNode(label =
"0"), 49.5: LeafNode(label = "1"), 10.5167: LeafNode(label = "0"), 7.8:
LeafNode(label = "0"), 9.825: LeafNode(label = "0"), 8.4333: LeafNode(label =
"0"), 262.375: LeafNode(label = "1")})
```

[52]:

```
tree.feature
```

[52]:

```
'Fare'
```

[53]:

```
def plot_tree(tree, g):
    root_node = tree.feature
    if root_node is None:
        return g
    g.node(root_node, nohtml(root_node))
    child_nodes = tree.children.keys()
    for i, child in enumerate(child_nodes):
        node = tree.children[child]
        name = str(node.feature) if node.feature is not None else␣
↪str(child)+str(node.label)
        label = node.feature if node.feature is not None else node.label
        g.node(name, nohtml(label))
        g.edge(root_node, name, label=child)
        plot_tree(node, g)
    return g
```

1.6 Decision Tree (sklearn)

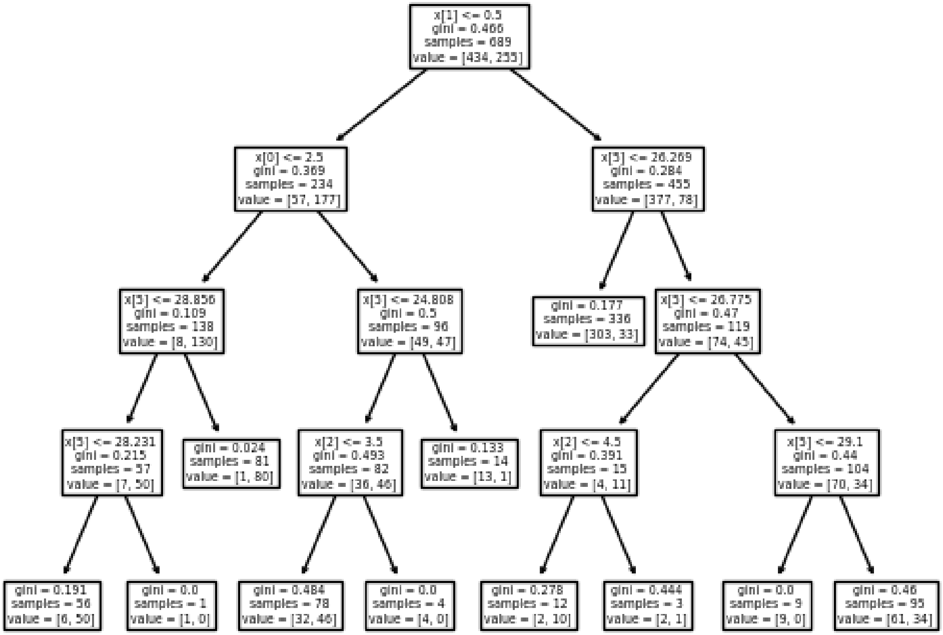
[54]:
```
from sklearn import tree
```

[55]:
```
clf = tree.DecisionTreeClassifier(max_depth=4, random_state=42, ccp_alpha=0.001)
```

[56]:
```
clf.fit(x, y)
```

[56]:
```
DecisionTreeClassifier(ccp_alpha=0.001, max_depth=4, random_state=42)
```

[57]:
```
tree.plot_tree(clf);
```



[58]:

```
clf.score(x, y)
```

[58]:

```
0.8403483309143687
```

1.7 Evaluation

[59]:

```
df_test = pd.read_csv('titanic_test.csv')
df_test.head()
```

[59]:

```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

1.8 Preprocessing

[60]:

```
df_test = df_test.drop(columns = ['PassengerId', 'Name', 'Cabin', 'Ticket',␣
 ↪'Embarked'], axis = 1)
```

[64]:

```
df_test['Age'] = df_test['Age'].replace(np.nan, df_test['Age'].median(axis = 0))
```

[65]:

```
df_test['Age'] = df_test['Age'].astype(int)
```

[78]:

```
df_test['Sex'] = df_test['Sex'].map({'male': 1, 'female': 0})
```

[67]:

```
df_test['Age'] = pd.cut(x = df_test['Age'], bins=[0, 5, 20, 30, 40, 50, 60,␣
 ↪100], labels = [0,1,2,3,4,5,6])
```

[68]:

```
df_test.dropna(subset=['Age'], axis= 0, inplace = True)
```

Splitting the df into X and y variables

[80]:
```
x = df_test.drop(columns = ['Survived'])
y = df_test['Survived']
```

Model evaluation on test df

[90]:
```
from sklearn.metrics import accuracy_score
```

[82]:
```
y_pred = clf.predict(x)
```

[83]:
```
accuracy_score(y, y_pred)
```

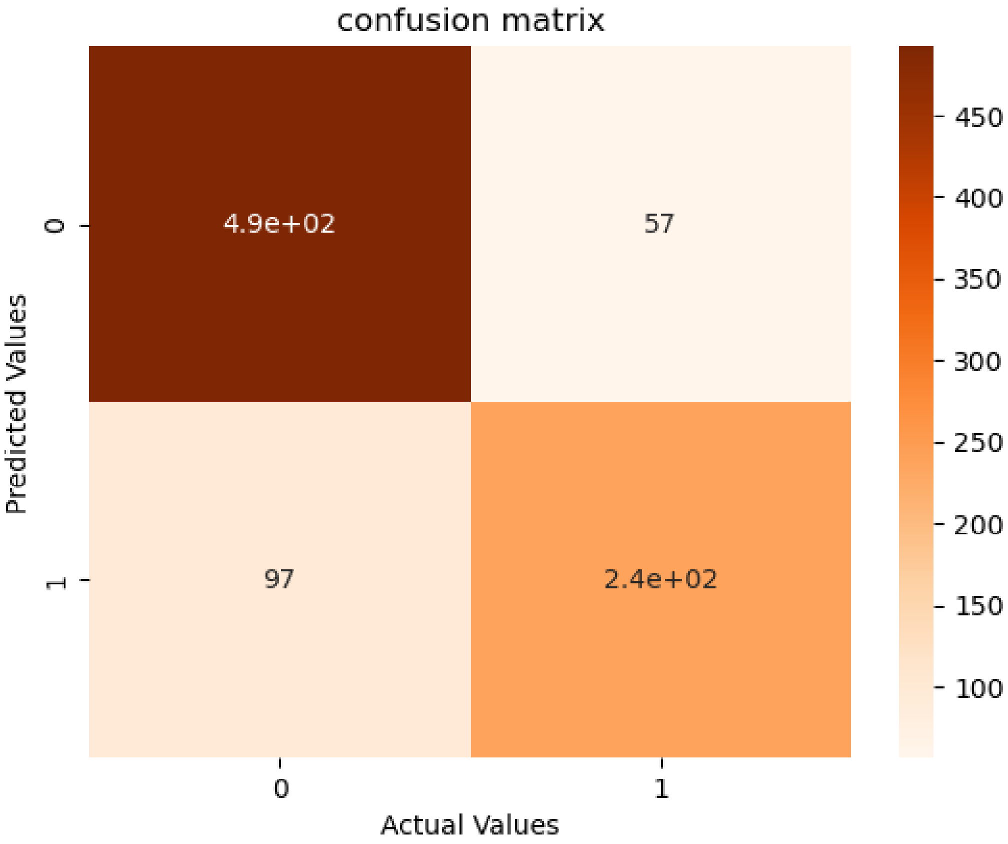
[83]:
```
0.8257918552036199
```

Confusion Matrix

[89]:
```
from sklearn.metrics import confusion_matrix
```

[88]:
```
sns.heatmap(confusion_matrix(y, y_pred), annot = True, cmap = 'Oranges')
plt.ylabel('Predicted Values')
plt.xlabel('Actual Values')
plt.title('Confusion Matrix')
```

[88]:
```
Text(0.5, 1.0, 'confusion matrix')
```


confusion matrix
Predicted Values
Actual Values
0
1
0
1
4.9e+02
57
97
2.4e+02
450
400
350
300
250
200
150
100