In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
master_titanic = pd.read_csv(r"D:\PG-DAI\MachineLearning\Dec 20 Decision Trees\Titanic.csv")
```

In [3]:
```
master_titanic.head()
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [8]:
```
x_titanic = master_titanic.iloc[:, [1,2,4,5,9]]
```

In [15]:
```
x_titanic
```

Out[15]:

	Survived	Pclass	Sex	Age	Fare
0	0	3	male	22.0	7.2500
1	1	1	female	38.0	71.2833
2	1	3	female	26.0	7.9250
3	1	1	female	35.0	53.1000
4	0	3	male	35.0	8.0500
...	...	...	...	...	...

	Survived	Pclass	Sex	Age	Fare
886	0	2	male	27.0	13.0000
887	1	1	female	19.0	30.0000
888	0	3	female	NaN	23.4500
889	1	1	male	26.0	30.0000
890	0	3	male	32.0	7.7500

891 rows × 5 columns

In [12]:
```
categorical_cols = ['Sex', 'Married', 'Education', 'Self_Employed','Property_Area','Loan_Status']


#import pandas as pd
df = pd.get_dummies(x_titanic, 'Sex')
```

In [16]:
```
del df['Sex_male']
```

In [17]:
```
inputs = df
```

In [19]:
```
x = df.loc[:,df.columns != 'Survived']
```

In [20]:
```
y = df['Survived']
```

In [22]:
```
x['Age'].replace(to_replace=np.nan, value=x.Age.mean(), inplace=True, limit=None, regex=False, method='pad')
```

```
C:\ProgramData\Anaconda3\lib\site-packages\pandas\core\generic.py:6619: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  return self._update_inplace(result)
```

In [23]:
```
x.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Pclass      891 non-null    int64
 1   Age         891 non-null    float64
 2   Fare        891 non-null    float64
 3   Sex_female  891 non-null    uint8
dtypes: float64(2), int64(1), uint8(1)
memory usage: 21.9 KB
```

In [25]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, random_state=7)
```

In [26]:
```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
```

In [27]:
```
model.fit(X_train,y_train)
model.score(X_test,y_test)
```

Out[27]:
```
0.7443946188340808
```

In [28]:
```
model.predict(X_test[0:10])
```

Out[28]:
```
array([0, 0, 0, 0, 1, 1, 0, 0, 0, 1], dtype=int64)
```

In [29]:
```
model.predict_proba(X_test[:10])
```

Out[29]:
```
array([[0.96074687, 0.03925313],
       [0.9290383 , 0.0709617 ],
       [0.97063236, 0.02936764],
       [0.96829032, 0.03170968],
       [0.02164717, 0.97835283],
```

```
       [0.22418857, 0.77581143],
       [0.96858177, 0.03141823],
       [0.9680319 , 0.0319681 ],
       [0.93850945, 0.06149055],
       [0.02199788, 0.97800212]])
```

In [30]:

```
from sklearn.model_selection import cross_val_score
cross_val_score(GaussianNB(),X_train, y_train, cv=5)
```

Out[30]:

```
array([0.79104478, 0.79850746, 0.79850746, 0.83458647, 0.7518797 ])
```

In [48]:

```
from sklearn.model_selection import GridSearchCV
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import StratifiedKFold


skf = StratifiedKFold(n_splits=200)
params = {}
gd = GridSearchCV(model, cv=skf, param_grid=params, return_train_score=True)

gd.fit(X_train,y_train)
print(gd.best_params_)
print(gd.best_score_)
```

```
{}
0.7958333333333333
```