

```
In [68]: import pandas as pd
import warnings
warnings.filterwarnings("ignore")
```

```
In [69]: data=pd.read_csv("/home/placenet/Downloads/Titanic Dataset.csv")
```

```
In [70]: data.describe()
```

```
Out[70]:
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [71]: data.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
#   Column          Non-Null Count  Dtype
---  -
0   PassengerId      891 non-null    int64
1   Survived         891 non-null    int64
2   Pclass           891 non-null    int64
3   Name             891 non-null    object
4   Sex              891 non-null    object
5   Age              714 non-null    float64
6   SibSp            891 non-null    int64
7   Parch            891 non-null    int64
8   Ticket           891 non-null    object
9   Fare             891 non-null    float64
10  Cabin            204 non-null    object
11  Embarked         889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

In [72]: data.head()

Out[72]:

	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 [73]: data.isna().sum()
```

```
Out[73]: PassengerId      0
Survived      0
Pclass        0
Name          0
Sex           0
Age          177
SibSp         0
Parch         0
Ticket        0
Fare          0
Cabin        687
Embarked      2
dtype: int64
```

```
In [74]: data1=data.drop(['PassengerId', 'Name', 'Ticket', 'Cabin', 'SibSp', 'Parch'],axis=1)
data1
```

```
Out[74]:
```

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	male	22.0	7.2500	S
1	1	1	female	38.0	71.2833	C
2	1	3	female	26.0	7.9250	S
3	1	1	female	35.0	53.1000	S
4	0	3	male	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	male	27.0	13.0000	S
887	1	1	female	19.0	30.0000	S
888	0	3	female	NaN	23.4500	S
889	1	1	male	26.0	30.0000	C
890	0	3	male	32.0	7.7500	Q

891 rows × 6 columns

```
In [75]: data2=data1.fillna(data.median())  
data2
```

```
Out[75]:
```

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	male	22.0	7.2500	S
1	1	1	female	38.0	71.2833	C
2	1	3	female	26.0	7.9250	S
3	1	1	female	35.0	53.1000	S
4	0	3	male	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	male	27.0	13.0000	S
887	1	1	female	19.0	30.0000	S
888	0	3	female	28.0	23.4500	S
889	1	1	male	26.0	30.0000	C
890	0	3	male	32.0	7.7500	Q

891 rows × 6 columns

```
In [76]: data2.isna().sum()
```

```
Out[76]: Survived    0  
Pclass    0  
Sex        0  
Age        0  
Fare        0  
Embarked    2  
dtype: int64
```

```
In [77]: data2.fillna(35,inplace=True)
data2
```

```
Out[77]:
```

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	male	22.0	7.2500	S
1	1	1	female	38.0	71.2833	C
2	1	3	female	26.0	7.9250	S
3	1	1	female	35.0	53.1000	S
4	0	3	male	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	male	27.0	13.0000	S
887	1	1	female	19.0	30.0000	S
888	0	3	female	28.0	23.4500	S
889	1	1	male	26.0	30.0000	C
890	0	3	male	32.0	7.7500	Q

891 rows × 6 columns

```
In [78]: data2.isna().sum()
```

```
Out[78]: Survived    0
Pclass    0
Sex        0
Age        0
Fare       0
Embarked   0
dtype: int64
```

```
In [79]: data2.dtypes
```

```
Out[79]: Survived      int64  
Pclass        int64  
Sex           object  
Age          float64  
Fare         float64  
Embarked      object  
dtype: object
```

```
In [80]: data2['Sex']=data2['Sex'].map({'male':0,'female':1})  
data2
```

```
Out[80]:
```

	Survived	Pclass	Sex	Age	Fare	Embarked
0	0	3	0	22.0	7.2500	S
1	1	1	1	38.0	71.2833	C
2	1	3	1	26.0	7.9250	S
3	1	1	1	35.0	53.1000	S
4	0	3	0	35.0	8.0500	S
...	...	...	...	...	...	...
886	0	2	0	27.0	13.0000	S
887	1	1	1	19.0	30.0000	S
888	0	3	1	28.0	23.4500	S
889	1	1	0	26.0	30.0000	C
890	0	3	0	32.0	7.7500	Q

891 rows × 6 columns

```
In [81]: x=data2.drop('Survived',axis=1)  
y=data2['Survived']
```

```
In [82]: x=pd.get_dummies(x)
x
```

```
Out[82]:
```

	Pclass	Sex	Age	Fare	Embarked_35	Embarked_C	Embarked_Q	Embarked_S
0	3	0	22.0	7.2500	0	0	0	1
1	1	1	38.0	71.2833	0	1	0	0
2	3	1	26.0	7.9250	0	0	0	1
3	1	1	35.0	53.1000	0	0	0	1
4	3	0	35.0	8.0500	0	0	0	1
...	...	...	...	...	...	...	...	...
886	2	0	27.0	13.0000	0	0	0	1
887	1	1	19.0	30.0000	0	0	0	1
888	3	1	28.0	23.4500	0	0	0	1
889	1	0	26.0	30.0000	0	1	0	0
890	3	0	32.0	7.7500	0	0	1	0

891 rows × 8 columns

```
In [83]: #divide the data into testing & training
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.33,random_state=42)
```

```
In [84]: #importing Random Forest Classifier from sklearn.ensemble
from sklearn.model_selection import GridSearchCV #GridSearchCV is for parameter tuning
from sklearn.ensemble import RandomForestClassifier
cls=RandomForestClassifier()
n_estimators=[25,50,75,100,125,150,175,200] #number of decision trees in the forest, default = 100
criterion=['gini', 'entropy'] #criteria for choosing nodes default = 'gini'
max_depth=[3,5,10] #maximum number of nodes in a tree default = None (it will go till all possible nodes)
parameters={'n_estimators': n_estimators, 'criterion':criterion, 'max_depth':max_depth} #this will undergo 8*2
RFC_cls = GridSearchCV(cls, parameters)
RFC_cls.fit(x_train,y_train)
```

```
Out[84]: GridSearchCV(estimator=RandomForestClassifier(),
                      param_grid={'criterion': ['gini', 'entropy'],
                                   'max_depth': [3, 5, 10],
                                   'n_estimators': [25, 50, 75, 100, 125, 150, 175, 200]})
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.  
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

```
In [85]: RFC_cls.best_params_
```

```
Out[85]: {'criterion': 'entropy', 'max_depth': 5, 'n_estimators': 200}
```

```
In [86]: cls=RandomForestClassifier(n_estimators=175,criterion='entropy',max_depth=10)
```

```
In [87]: cls.fit(x_train,y_train)
```

```
Out[87]: RandomForestClassifier(criterion='entropy', max_depth=10, n_estimators=175)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.  
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.



```
In [88]: rfy_pred=cls.predict(x_test)
rfy_pred
```

```
Out[88]: array([0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1,
                0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1,
                0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1,
                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
                1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0,
                0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1,
                0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0,
                0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0,
                1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0,
                0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1,
                0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0,
                0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0,
                1, 0, 1, 1, 0, 0, 1, 1, 0])
```

```
In [89]: from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,rfy_pred)
```

```
Out[89]: array([[148,  27],
                [ 34,  86]])
```

```
In [90]: from sklearn.metrics import accuracy_score
accuracy_score(y_test,rfy_pred)
```

```
Out[90]: 0.7932203389830509
```

```
In [ ]:
```