

## C2

```
In [1]: import pandas as pd
        from sklearn.linear_model import LogisticRegression
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import StandardScaler
```

```
In [3]: df=pd.read_csv("C2_test.gender_submission.csv")
        df1=pd.read_csv("C2_train.gender_submission.csv")
```

```
In [4]: df_=df.drop(["Cabin", "Name", "Embarked", "Ticket", "PassengerId", "Sex"],axis=1)
        df1_=df1.drop(["Survived", "Cabin", "Name", "Embarked", "Ticket", "PassengerId"],axis=1)
        print(df_)
        print(df1_)
```

	Pclass	Age	SibSp	Parch	Fare
0	3	34.5	0	0	7.8292
1	3	47.0	1	0	7.0000
2	2	62.0	0	0	9.6875
3	3	27.0	0	0	8.6625
4	3	22.0	1	1	12.2875
..	...	...	...	...	...
413	3	NaN	0	0	8.0500
414	1	39.0	0	0	108.9000
415	3	38.5	0	0	7.2500
416	3	NaN	0	0	8.0500
417	3	NaN	1	1	22.3583

[418 rows x 5 columns]

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

[891 rows x 6 columns]

In [5]: df1.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 [6]: df\_=df\_.dropna()  
df1\_=df1\_.dropna()  
df1\_.info()  
df\_.info()

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 714 entries, 0 to 890
Data columns (total 6 columns):
#   Column  Non-Null Count  Dtype
---  -
0   Pclass  714 non-null    int64
1   Sex     714 non-null    object
2   Age     714 non-null    float64
3   SibSp   714 non-null    int64
4   Parch   714 non-null    int64
5   Fare    714 non-null    float64
dtypes: float64(2), int64(3), object(1)
memory usage: 39.0+ KB
<class 'pandas.core.frame.DataFrame'>
Int64Index: 331 entries, 0 to 415
Data columns (total 5 columns):
#   Column  Non-Null Count  Dtype
---  -
0   Pclass  331 non-null    int64
1   Age     331 non-null    float64
2   SibSp   331 non-null    int64
3   Parch   331 non-null    int64
4   Fare    331 non-null    float64
dtypes: float64(2), int64(3)
memory usage: 15.5 KB
```

```
In [2]: from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
from sklearn.model_selection import GridSearchCV
from sklearn.tree import plot_tree
```

```
In [7]: y=df1_["Sex"]
x=df1_.drop(["Sex"],axis=1)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

```
In [18]: rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

```
Out[18]: RandomForestClassifier()
```

```
In [12]: parameter={'max_depth':[1,2,3,4,5],
                    "min_samples_leaf":[5,10,15,20,25],
                    "n_estimators":[10,20,30,40,50]}
```

```
In [13]: grid_search = GridSearchCV(estimator=rfc,param_grid=parameter,cv=2,scoring="ac
grid_search.fit(x_train,y_train)
```

```
Out[13]: GridSearchCV(cv=2, estimator=RandomForestClassifier(),
                    param_grid={'max_depth': [1, 2, 3, 4, 5],
                                'min_samples_leaf': [5, 10, 15, 20, 25],
                                'n_estimators': [10, 20, 30, 40, 50]},
                    scoring='accuracy')
```

```
In [14]: grid_search.best_score_
```

```
Out[14]: 0.6653654618473896
```

```
In [16]: rfc_best=grid_search.best_estimator_
```

```
In [17]: plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],class_names=['Yes','No'],filled=True)
```

```
Out[17]: [Text(2575.3846153846152, 1902.6000000000001, 'X[4] <= 8.756\ngini = 0.461\ns
amples = 312\nvalue = [180, 319]\nnclass = No'),
Text(1373.5384615384614, 1359.0, 'X[4] <= 7.746\ngini = 0.26\nsamples = 93\n
value = [22, 121]\nnclass = No'),
Text(686.7692307692307, 815.4000000000001, 'X[4] <= 7.227\ngini = 0.053\nsam
ples = 25\nvalue = [1, 36]\nnclass = No'),
Text(343.38461538461536, 271.79999999999995, 'gini = 0.0\nsamples = 15\nvalu
e = [0, 24]\nnclass = No'),
Text(1030.1538461538462, 271.79999999999995, 'gini = 0.142\nsamples = 10\nva
lue = [1, 12]\nnclass = No'),
Text(2060.3076923076924, 815.4000000000001, 'X[4] <= 7.988\ngini = 0.318\nsa
mples = 68\nvalue = [21, 85]\nnclass = No'),
Text(1716.9230769230767, 271.79999999999995, 'gini = 0.408\nsamples = 46\nva
lue = [20, 50]\nnclass = No'),
Text(2403.6923076923076, 271.79999999999995, 'gini = 0.054\nsamples = 22\nva
lue = [1, 35]\nnclass = No'),
Text(3777.230769230769, 1359.0, 'X[2] <= 3.5\ngini = 0.494\nsamples = 219\nv
alue = [158, 198]\nnclass = No'),
Text(3433.8461538461534, 815.4000000000001, 'X[4] <= 82.508\ngini = 0.495\ns
amples = 211\nvalue = [154, 188]\nnclass = No'),
Text(3090.461538461538, 271.79999999999995, 'gini = 0.488\nsamples = 181\nva
lue = [122, 166]\nnclass = No'),
Text(3777.230769230769, 271.79999999999995, 'gini = 0.483\nsamples = 30\nval
ue = [32, 22]\nnclass = Yes'),
Text(4120.615384615385, 815.4000000000001, 'gini = 0.408\nsamples = 8\nvalue
 = [4, 10]\nnclass = No')]
```

