

```
In [34]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [15]: df=pd.read_csv(r"C:\Users\user\Downloads\C2_test.gender_submission - C2_test.g
df
```

```
Out[15]:
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN
...	...	...	...	...	...	...	...	...	...	...
413	1305	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN
414	1306	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105
415	1307	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN
416	1308	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN
417	1309	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN

418 rows × 11 columns



```
In [18]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype  
---  -
 0   PassengerId     418 non-null   int64  
 1   Pclass          418 non-null   int64  
 2   Name            418 non-null   object  
 3   Sex             418 non-null   object  
 4   Age            332 non-null   float64 
 5   SibSp          418 non-null   int64  
 6   Parch          418 non-null   int64  
 7   Ticket          418 non-null   object  
 8   Fare           417 non-null   float64 
 9   Cabin          91 non-null    object  
10   Embarked       418 non-null   object  
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

```
In [21]: df['Embarked'].value_counts()
```

```
Out[21]: S    270
         C    102
         Q     46
         Name: Embarked, dtype: int64
```

```
In [22]: df1=df[['Embarked', 'PassengerId', 'SibSp', 'Parch', 'Pclass']]
```

```
In [23]: x=df1.drop('Embarked',axis=1)
         y=df['Embarked']
```

```
In [24]: g1={"S":{'C':1,"Q":2}}
df=df.replace(g1)
print(df)
```

	PassengerId	Pclass	Name \
0	892	3	Kelly, Mr. James
1	893	3	Wilkes, Mrs. James (Ellen Needs)
2	894	2	Myles, Mr. Thomas Francis
3	895	3	Wirz, Mr. Albert
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)
..	...	...	...
413	1305	3	Spector, Mr. Woolf
414	1306	1	Oliva y Ocana, Dona. Fermina
415	1307	3	Saether, Mr. Simon Sivertsen
416	1308	3	Ware, Mr. Frederick
417	1309	3	Peter, Master. Michael J

  

	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	male	34.5	0	0	330911	7.8292	NaN	Q
1	female	47.0	1	0	363272	7.0000	NaN	S
2	male	62.0	0	0	240276	9.6875	NaN	Q
3	male	27.0	0	0	315154	8.6625	NaN	S
4	female	22.0	1	1	3101298	12.2875	NaN	S
..	...	...	...	...	...	...	...	...
413	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	female	39.0	0	0	PC 17758	108.9000	C105	C
415	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	male	NaN	0	0	359309	8.0500	NaN	S
417	male	NaN	1	1	2668	22.3583	NaN	C

[418 rows x 11 columns]

```
In [31]: from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30)
```

```
In [32]: from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[32]: RandomForestClassifier()

```
In [33]: parameters={'max_depth':[1,2,3,4,5],
                    'min_samples_leaf':[5,10,15,20,25],
                    'n_estimators':[10,20,30,40,50]}
```

```
In [34]: from sklearn.model_selection import GridSearchCV  
grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring='acc  
grid_search.fit(x_train,y_train)
```

```
Out[34]: 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 [35]: grid_search.best_score_
```

```
Out[35]: 0.6506849315068493
```

```
In [36]: rfc_best=grid_search.best_estimator_
```

```
In [37]: from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes', 'No'])
```

```

-----
IndexError                                Traceback (most recent call last)
<ipython-input-37-77bd6a24336f> in <module>
      2
      3 plt.figure(figsize=(80,40))
----> 4 plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names
=['Yes','No'],filled=True)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inner_f(*args, **kwargs)
     61         extra_args = len(args) - len(all_args)
     62         if extra_args <= 0:
--> 63             return f(*args, **kwargs)
     64
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in plot_tree(decision_tree, max_depth, feature_names, class_names, label, filled, impurity, node_ids, proportion, rotate, rounded, precision, ax, fontsize)
    192         proportion=proportion, rotate=rotate, rounded=rounded,
    193         precision=precision, fontsize=fontsize)
--> 194     return exporter.export(decision_tree, ax=ax)
    195
    196

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in export(self, decision_tree, ax)
    582         ax.clear()
    583         ax.set_axis_off()
--> 584         my_tree = self._make_tree(0, decision_tree.tree_,
    585                                 decision_tree.criterion)
    586         draw_tree = buchheim(my_tree)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in _make_tree(self, node_id, et, criterion, depth)
    563         # traverses _tree.Tree recursively, builds intermediate
    564         # "_reingold_tilford.Tree" object
--> 565         name = self.node_to_str(et, node_id, criterion=criterion)
    566         if (et.children_left[node_id] != _tree.TREE_LEAF
    567             and (self.max_depth is None or depth <= self.max_dept
h)):

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\tree\_export.py in node_to_str(self, tree, node_id, criterion)
    353         node_string += 'class = '
    354         if self.class_names is not True:
--> 355             class_name = self.class_names[np.argmax(value)]
    356         else:
    357             class_name = "y%s%s%s" % (characters[1],
IndexError: list index out of range

```

In [ ]: