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

In [2]:
```
from sklearn.linear_model import LogisticRegression
```

In [3]:
```
df_train=pd.read_csv("train_gender.csv")
df_test=pd.read_csv("gender.csv")
df_train
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	C
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	
...	...	...	...	...	...	...	...	...	...	...	
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	

891 rows × 12 columns

In [4]:
```
df_test
```

Out[4]:

	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	female	22.0	1	1	3101298	12.2875	NaN

In [5]:
```
df1=df_train.dropna()
```

In [6]:
```
df2=df_test.dropna()
```

In [7]:
```
feature_matrix=df1[['PassengerId', 'Pclass','Age', 'SibSp',
       'Parch','Fare']]
target_vector=df1[['Embarked']]
```

In [8]:
```
feature_matrix.shape
```

Out[8]:
```
(183, 6)
```

In [9]:
```
target_vector.shape
```

Out[9]:
```
(183, 1)
```

In [10]:
```
from sklearn.preprocessing import StandardScaler
```

In [11]:
```
fs=StandardScaler().fit_transform(feature_matrix)
```

In [12]:
```
logr=LogisticRegression()
logr.fit(fs,target_vector)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:63: Da
taConversionWarning: A column-vector y was passed when a 1d array was expecte
d. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[12]:
```
LogisticRegression()
```

In [13]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 87 entries, 12 to 414
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   PassengerId  87 non-null     int64  
 1   Pclass       87 non-null     int64  
 2   Name         87 non-null     object 
 3   Sex          87 non-null     object 
 4   Age          87 non-null     float64
 5   SibSp        87 non-null     int64  
 6   Parch        87 non-null     int64  
 7   Ticket       87 non-null     object 
 8   Fare         87 non-null     float64
 9   Cabin        87 non-null     object 
 10  Embarked     87 non-null     object 
dtypes: float64(2), int64(4), object(5)
memory usage: 8.2+ KB
```

In [14]:
```
df2.columns
```

Out[14]:
```
Index(['PassengerId', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch',
       'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [15]:
```
observation=df2[['PassengerId', 'Pclass','Age', 'SibSp', 'Parch',
        'Fare']]
```

In [16]:
```
prediction=logr.predict(observation)
print(prediction)
```

```
['S' 'S' 'C' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'C' 'S' 'C' 'C' 'S' 'C' 'S' 'S'
 'S' 'S' 'C' 'S' 'S' 'S' 'S' 'S' 'C' 'S' 'S' 'S' 'S' 'C' 'S' 'S' 'S' 'S'
 'S' 'S' 'S' 'S' 'S' 'C' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S'
 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'C' 'S' 'S' 'S'
 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S' 'S']
```

In [17]:
```
logr.classes_
```

Out[17]:
```
array(['C', 'Q', 'S'], dtype=object)
```

In [18]:
```
logr.predict_proba(observation)
```

Out[18]:
```
array([[6.08971905e-040, 1.00337329e-144, 1.00000000e+000],
       [1.23227402e-046, 1.21293028e-146, 1.00000000e+000],
       [1.00000000e+000, 2.64311964e-144, 2.62330648e-020],
       [8.74684689e-048, 9.59255505e-151, 1.00000000e+000],
       [7.79904257e-058, 1.27363136e-153, 1.00000000e+000],
       [1.47737287e-049, 3.16517530e-151, 1.00000000e+000],
       [9.87655203e-052, 6.61406838e-153, 1.00000000e+000],
       [2.41467012e-059, 1.34463863e-156, 1.00000000e+000],
       [9.48918099e-044, 2.05555026e-150, 1.00000000e+000],
       [4.27537892e-050, 6.30946278e-154, 1.00000000e+000],
       [1.00000000e+000, 3.64533680e-146, 5.79600005e-018],
       [2.12722398e-068, 1.45384757e-162, 1.00000000e+000],
       [1.00000000e+000, 1.95684213e-147, 3.44533462e-018],
       [1.00000000e+000, 4.39546971e-148, 8.71913651e-017],
       [1.32481073e-061, 4.20734452e-161, 1.00000000e+000],
       [1.00000000e+000, 7.58602331e-149, 2.93658715e-017],
       [2.74061829e-062, 3.64139105e-162, 1.00000000e+000],
       [2.25687919e-001, 4.58245775e-139, 7.74312081e-001],
       [1.80986273e-001, 2.53591052e-139, 8.19013727e-001],
       [5.45095546e-063, 5.94350402e-161, 1.00000000e+000],
       [9.98575373e-001, 8.70056035e-140, 1.42462732e-003],
       [3.85799338e-056, 4.60564832e-163, 1.00000000e+000],
       [3.10314517e-046, 4.83527839e-157, 1.00000000e+000],
       [6.48473904e-055, 3.89846152e-162, 1.00000000e+000],
       [2.81368515e-070, 8.67176835e-171, 1.00000000e+000],
       [1.08280308e-063, 5.96125394e-168, 1.00000000e+000],
       [6.92497504e-001, 3.12646809e-143, 3.07502496e-001],
       [1.51715598e-070, 9.07550280e-173, 1.00000000e+000],
       [1.11930587e-049, 1.69989736e-163, 1.00000000e+000],
       [6.53910240e-056, 6.17042047e-166, 1.00000000e+000],
       [8.07049911e-066, 8.14357943e-171, 1.00000000e+000],
       [1.00000000e+000, 2.39992618e-155, 6.61726374e-012],
       [7.98318000e-050, 4.41670143e-169, 1.00000000e+000],
       [5.67530213e-004, 7.59115023e-152, 9.99432470e-001],
       [1.18824306e-068, 3.91524494e-176, 1.00000000e+000],
       [3.52097421e-061, 3.24873391e-174, 1.00000000e+000],
       [2.38027644e-060, 1.03615038e-174, 1.00000000e+000],
       [6.24227366e-067, 3.71362583e-180, 1.00000000e+000],
       [2.30654066e-051, 9.67574039e-173, 1.00000000e+000],
       [5.96967620e-052, 3.09530437e-173, 1.00000000e+000],
       [3.60562527e-062, 1.84107354e-177, 1.00000000e+000],
       [9.98753360e-001, 1.91239301e-156, 1.24664028e-003],
       [1.15583025e-036, 9.14931704e-172, 1.00000000e+000],
       [3.23983533e-005, 1.20566080e-157, 9.99967602e-001],
       [4.27283617e-072, 7.12984244e-185, 1.00000000e+000],
       [1.55106048e-067, 8.39451329e-184, 1.00000000e+000],
       [7.81404051e-012, 8.27024590e-163, 1.00000000e+000],
       [2.54585276e-079, 4.02254454e-190, 1.00000000e+000],
       [1.49481230e-059, 4.60453558e-183, 1.00000000e+000],
       [6.51411592e-058, 8.68155622e-182, 1.00000000e+000],
       [4.33826449e-048, 4.28426682e-179, 1.00000000e+000],
       [3.10340679e-039, 8.85281708e-177, 1.00000000e+000],
       [9.16967301e-067, 2.79103457e-187, 1.00000000e+000],
       [2.30031083e-039, 2.00341270e-178, 1.00000000e+000],
       [1.25754688e-060, 7.63824033e-189, 1.00000000e+000],
       [7.54340191e-041, 7.29736878e-182, 1.00000000e+000],
       [4.50487411e-060, 2.39522055e-191, 1.00000000e+000],
```

```
       [2.74644477e-060, 1.08586460e-191, 1.00000000e+000],
       [2.59943554e-079, 3.48017728e-200, 1.00000000e+000],
       [9.84602149e-039, 9.76481273e-187, 1.00000000e+000],
       [1.73773295e-057, 1.10233307e-192, 1.00000000e+000],
       [1.20940443e-043, 2.54641319e-188, 1.00000000e+000],
       [2.93938669e-040, 1.01111168e-186, 1.00000000e+000],
       [6.99002669e-088, 2.03960384e-207, 1.00000000e+000],
       [9.62309262e-086, 1.34121069e-206, 1.00000000e+000],
       [4.30328484e-078, 3.04175748e-204, 1.00000000e+000],
       [2.44186417e-080, 3.13990447e-205, 1.00000000e+000],
       [4.98766646e-082, 8.94005298e-207, 1.00000000e+000],
       [1.00000000e+000, 3.64074212e-226, 1.52370836e-080],
       [1.23456527e-070, 1.58625020e-204, 1.00000000e+000],
       [4.05689755e-083, 2.82183444e-209, 1.00000000e+000],
       [8.36304637e-075, 7.01537655e-205, 1.00000000e+000],
       [1.34903426e-074, 9.70432827e-208, 1.00000000e+000],
       [1.20000763e-048, 3.30387186e-199, 1.00000000e+000],
       [5.21188318e-093, 1.79611310e-215, 1.00000000e+000],
       [3.33084545e-066, 2.16245191e-205, 1.00000000e+000],
       [9.55155476e-077, 6.11194150e-210, 1.00000000e+000],
       [8.56917344e-064, 6.11231434e-208, 1.00000000e+000],
       [1.78129683e-081, 3.16634703e-214, 1.00000000e+000],
       [1.73596071e-075, 9.69307656e-213, 1.00000000e+000],
       [6.95928293e-069, 6.76039538e-210, 1.00000000e+000],
       [1.02608940e-040, 2.33597670e-200, 1.00000000e+000],
       [2.18536793e-086, 3.31125898e-217, 1.00000000e+000],
       [1.18912368e-091, 7.63907138e-221, 1.00000000e+000],
       [1.10798951e-025, 6.99386766e-195, 1.00000000e+000],
       [2.38853088e-066, 7.74530338e-211, 1.00000000e+000],
       [3.84344053e-060, 2.30602005e-209, 1.00000000e+000]])
```

In [19]:
```
logr.predict_proba(observation)[0][0]
```

Out[19]:
```
6.0897190477931514e-40
```

In [20]:
```
df2['Embarked'].value_counts()
```

Out[20]:
```
C    47
S    39
Q     1
Name: Embarked, dtype: int64
```

In [21]:
```
x=df2[['PassengerId', 'Pclass','Age', 'SibSp', 'Parch',
       'Fare']]
y=df2['Embarked']
```

In [22]:
```
g1={"Embarked":{"C":1,"S":2,"Q":3}}
df2=df2.replace(g1)
df2
```

Out[22]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
12	904	1	Snyder, Mrs. John Pillsbury (Nelle Stevenson)	female	23.0	1	0	21228	82.2667	B45
14	906	1	Chaffee, Mrs. Herbert Fuller (Carrie Constance...	female	47.0	1	0	W.E.P. 5734	61.1750	E31
24	916	1	Ryerson, Mrs. Arthur Larned (Emily Maria Borie)	female	48.0	1	3	PC 17608	262.3750	B57 B59 B63 B66
26	918	1	Ostby, Miss. Helene Ragnhild	female	22.0	0	1	113509	61.9792	B36
28	920	1	Brady, Mr. John Bertram	male	41.0	0	0	113054	30.5000	A21
...	...	...	...	...	...	...	...	...	...	...
404	1296	1	Frauenthal, Mr. Isaac Gerald	male	43.0	1	0	17765	27.7208	D40
405	1297	2	Nourney, Mr. Alfred (Baron von Drachstedt")"	male	20.0	0	0	SC/PARIS 2166	13.8625	D38
407	1299	1	Widener, Mr. George Dunton	male	50.0	1	1	113503	211.5000	C80
411	1303	1	Minahan, Mrs. William Edward (Lillian E Thorpe)	female	37.0	1	0	19928	90.0000	C78
414	1306	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105

87 rows × 11 columns

In [23]:
```
from sklearn.model_selection import train_test_split
```

In [24]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

In [25]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [26]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[26]:
```
RandomForestClassifier()
```

In [27]:
```
parameters={'max_depth':[1,2,3,4,5],
            'min_samples_leaf':[5,10,15,20,25],
            'n_estimators':[10,20,30,40,50]
}
```

In [28]:
```
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[28]:
```
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 [29]:
```
grid_search.best_score_
```

Out[29]:
```
0.6333333333333333
```

In [30]:
```
rfc_best=grid_search.best_estimator_
```

In [33]:
```
from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.columns,class_names=['Yes','N
```

Out[33]:
```
[Text(2232.0, 1630.8000000000002, 'PassengerId <= 1213.5\ngini = 0.464\nsam
ples = 41\nvalue = [38, 22]\nclass = Yes'),
 Text(1116.0, 543.5999999999999, 'gini = 0.391\nsamples = 29\nvalue = [33,
12]\nclass = Yes'),
 Text(3348.0, 543.5999999999999, 'gini = 0.444\nsamples = 12\nvalue = [5, 1
0]\nclass = No')]
```