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

In [2]:
```
titanic_data=pd.read_csv('titanic_train.csv')
```

In [3]:
```
len(titanic_data)
```

Out[3]:
```
891
```

In [4]:
```
titanic_data.head()
```

Out[4]:

	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 [5]:
```
titanic_data.index
```

Out[5]:
```
RangeIndex(start=0, stop=891, step=1)
```

In [6]:
```
titanic_data.columns
```

Out[6]:
```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [7]:
```
titanic_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 [8]:
```
titanic_data.dtypes
```

Out[8]:
```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Cabin           object
Embarked        object
dtype: object
```

In [9]:
```
titanic_data.describe()
```

Out[9]:

	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 [10]:

```
titanic_data.isna()
```

Out[10]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	False	False	False	False	False	False	False	False	False	False	True	False
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	True	False
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	True	False
...	...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	True	False
887	False	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	False	False	True	False	False	False	False	True	False
889	False	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	True	False

891 rows × 12 columns

In [11]:

```
titanic_data.isna().sum()
```

Out[11]:

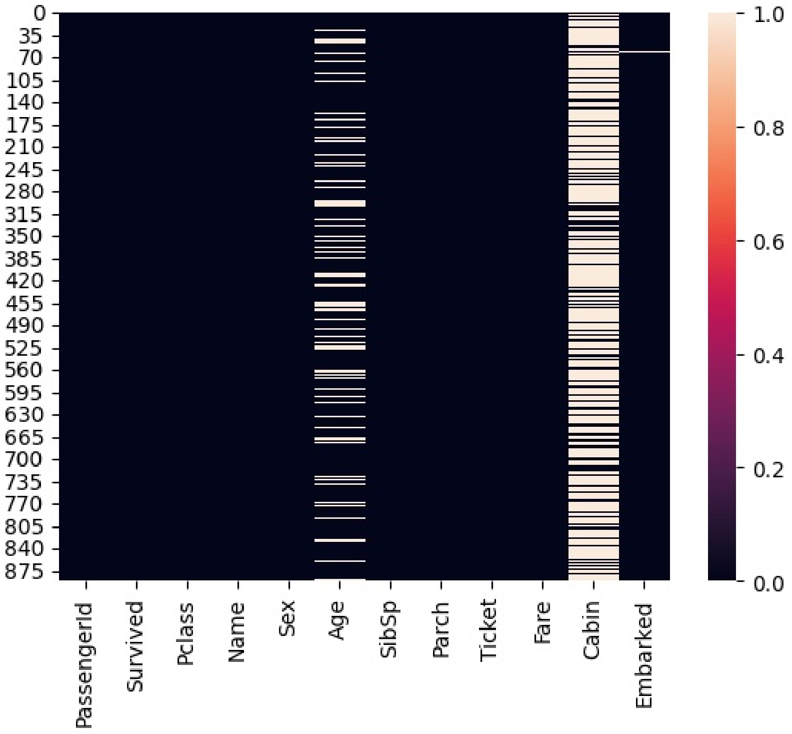
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [12]:

```
sns.heatmap(titanic_data.isna())
```

Out[12]:

```
<Axes: >
```



```
In [13]: (titanic_data['Age'].isna().sum()/len(titanic_data['Age']))*100
```

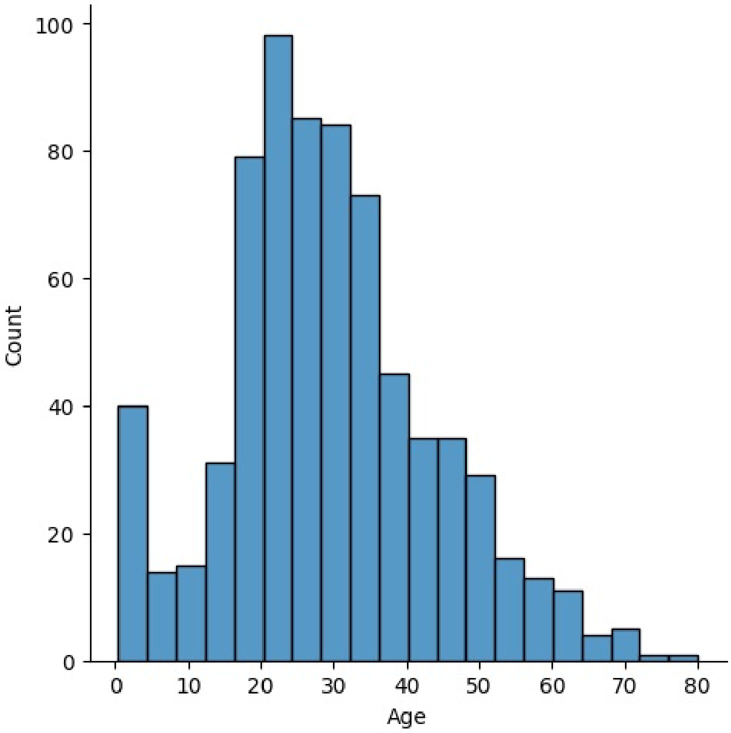
```
Out[13]: 19.865319865319865
```

```
In [14]: (titanic_data['Cabin'].isna().sum()/len(titanic_data['Cabin']))*100
```

```
Out[14]: 77.10437710437711
```

```
In [15]: sns.displot(x='Age',data=titanic_data)
```

```
Out[15]: <seaborn.axisgrid.FacetGrid at 0x2028649e4e0>
```

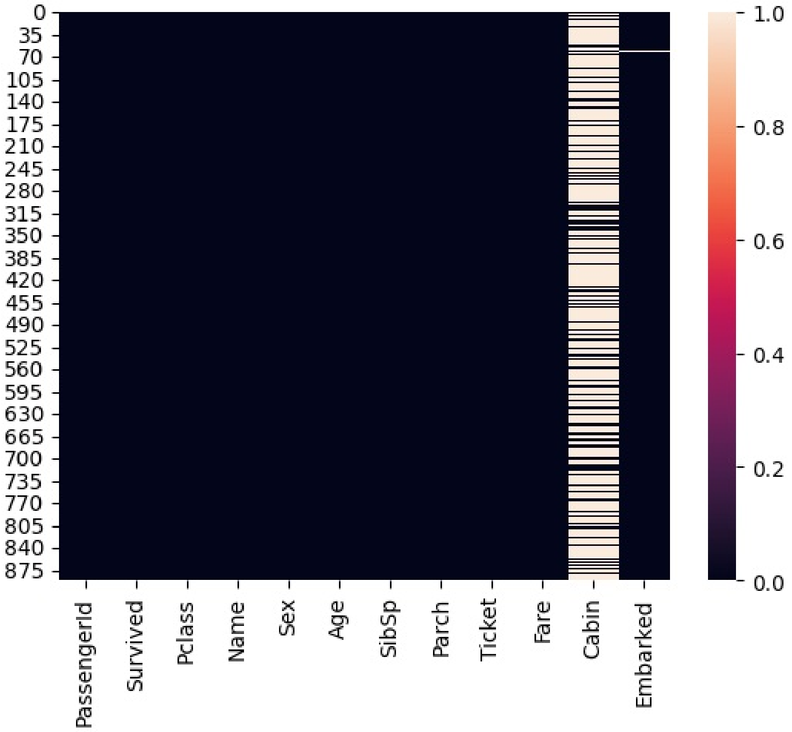


```
In [17]: titanic_data['Age'].isna().sum()

Out[17]: 0

In [18]: sns.heatmap(titanic_data.isna())

Out[18]: <Axes: >
```



```
In [19]: titanic_data.drop('Cabin',axis=1,inplace=True)
```

```
In [20]: titanic_data.head()
```

Out[20]:

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

```
In [21]: titanic_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 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          891 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  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(4)
memory usage: 76.7+ KB
```

In [22]:

```
titanic_data.dtypes
```

Out[22]:

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Embarked        object
dtype: object
```

In [23]:

```
gender=pd.get_dummies(titanic_data['Sex'],drop_first=True)
```

In [24]:

```
titanic_data['Gender']=gender
```

In [25]:

```
titanic_data.head()
```

Out[25]:

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

In [26]:

```
titanic_data.drop(['Name','Sex','Ticket','Embarked'],axis=1,inplace=True)
```

In [27]:

```
titanic_data.head()
```

Out[27]:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	Gender
0	1	0	3	22.0	1	0	7.2500	True
1	2	1	1	38.0	1	0	71.2833	False
2	3	1	3	26.0	0	0	7.9250	False
3	4	1	1	35.0	1	0	53.1000	False
4	5	0	3	35.0	0	0	8.0500	True

In [28]:

```
x=titanic_data[['PassengerId','Pclass','Age','SibSp','Parch','Fare','Gender']]
y=titanic_data['Survived']
```

In [29]:

```
y
```

Out[29]:
```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 891, dtype: int64
```

In [31]:
```
titanic_data.isnull().sum()
```

Out[31]:
```
PassengerId    0
Survived       0
Pclass         0
Age            0
SibSp          0
Parch          0
Fare           0
Gender         0
dtype: int64
```

In [32]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [34]:
```
features_matrix = titanic_data.iloc[:, 0:34]
```

In [35]:
```
target_vector = titanic_data.iloc[:, -1]
```

In [37]:
```
print('The Features Matrix Has %d Rows And %d Column(s)'%(features_matrix.shape))
print('The Target Matrix Has %d Rows And %d Column(s)'%(np.array(target_vector).reshape(-1, 1).shape))
```

```
The Features Matrix Has 891 Rows And 8 Column(s)
The Target Matrix Has 891 Rows And 1 Column(s)
```

In [38]:
```
features_matrix_standardized = StandardScaler().fit_transform(features_matrix)
```

In [60]:
```
Logistic_Regression_Model = algorithm.fit(features_matrix_standardized, target_vector)
```

In [ ]:
```
observation = [[1, 0, 0.99539, -0.05889, 0.8524299999999999, 0.02306,
0.8339799999999999, -0.37708, 1.0, 0.0376,
0.8524299999999999, -0.17755, 0.59755, -0.44945, 0.60536,
-0.38223, 0.8435600000000001, -0.38542, 0.58212, -0.32192,
0.56971, -0.29674, 0.36946, -0.47357, 0.56811, -0.51171,
0.41078000000000003, -0.46168000000000003, 0.21266, -0.3409,
0.42267, -0.54487, 0.18641, -0.453]]
```

In [40]:
```
from sklearn.linear_model import LogisticRegression

algorithm = LogisticRegression(penalty=None, dual=False, tol=1e-4, C=1.0, fit_intercept=True, l1_ratio=None)
```

In [43]:
```
print("""The Model Says The Probability Of The Observation We Passed Belonging To Class ['g'] Is %s""")
```

```
The Model Says The Probability Of The Observation We Passed Belonging To Class ['g'] Is %s
```

In [48]:
```
from sklearn.model_selection import train_test_split
```

In [49]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state=42)
```

In [50]:
```
from sklearn.linear_model import LogisticRegression
```

In [51]:
```
lr=LogisticRegression()
```

In [52]:
```
lr.fit(x_train,y_train)
```

```
C:\Users\lekha\AppData\Roaming\Python\Python312\site-packages\sklearn\linear_model\_logistic.py:469: Convergence
Warning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[52]:
```
▾ LogisticRegression
LogisticRegression()
```

In [53]:
```
predict=lr.predict(x_test)
```

In [54]:
```
from sklearn.metrics import confusion_matrix
```

In [55]:
```
pd.DataFrame(confusion_matrix(y_test,predict),columns=['Predicted No','Predicted Yes'],index=['Actual No','Actu
```

Out[55]:

	Predicted No	Predicted Yes
Actual No	151	24
Actual Yes	38	82

In [56]:
```
from sklearn.metrics import classification_report
```

In [57]:
```
print(classification_report(y_test,predict))
```

```
              precision    recall  f1-score   support

           0       0.80      0.86      0.83       175
           1       0.77      0.68      0.73       120

    accuracy                           0.79       295
   macro avg       0.79      0.77      0.78       295
weighted avg       0.79      0.79      0.79       295
```

In [ ]:
```
x = titanic_data.drop('g',axis=1)
y = titanic_data['g']
```

In [ ]:
```
df['g'].value_counts()
```

In [64]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.7, random_state=42)
x_train.shape, x_test.shape
```

Out[64]:
```
((623, 7), (268, 7))
```

In [73]:
```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[73]:
```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [74]:
```
rf = RandomForestClassifier()
```

In [75]:
```
params = {'max_depth': [2,3,5,10,20],
 'min_samples_leaf': [5,10,20,50,100,200],
 'n_estimators': [10,25,30,50,100,200]}
```

In [76]:
```
from sklearn.model_selection import GridSearchCV
grid_search = GridSearchCV(estimator=rf,param_grid=params,cv = 2, scoring="accuracy")
grid_search.fit(x_train, y_train)
```

Out[76]:
```
▸         GridSearchCV
▸ estimator: RandomForestClassifier
    ▸ RandomForestClassifier
```

In [77]:
```
grid_search.best_score_
```

Out[77]:
```
0.8218113612004287
```

In [78]:
```
rf_best = grid_search.best_estimator_
print(rf_best)
```

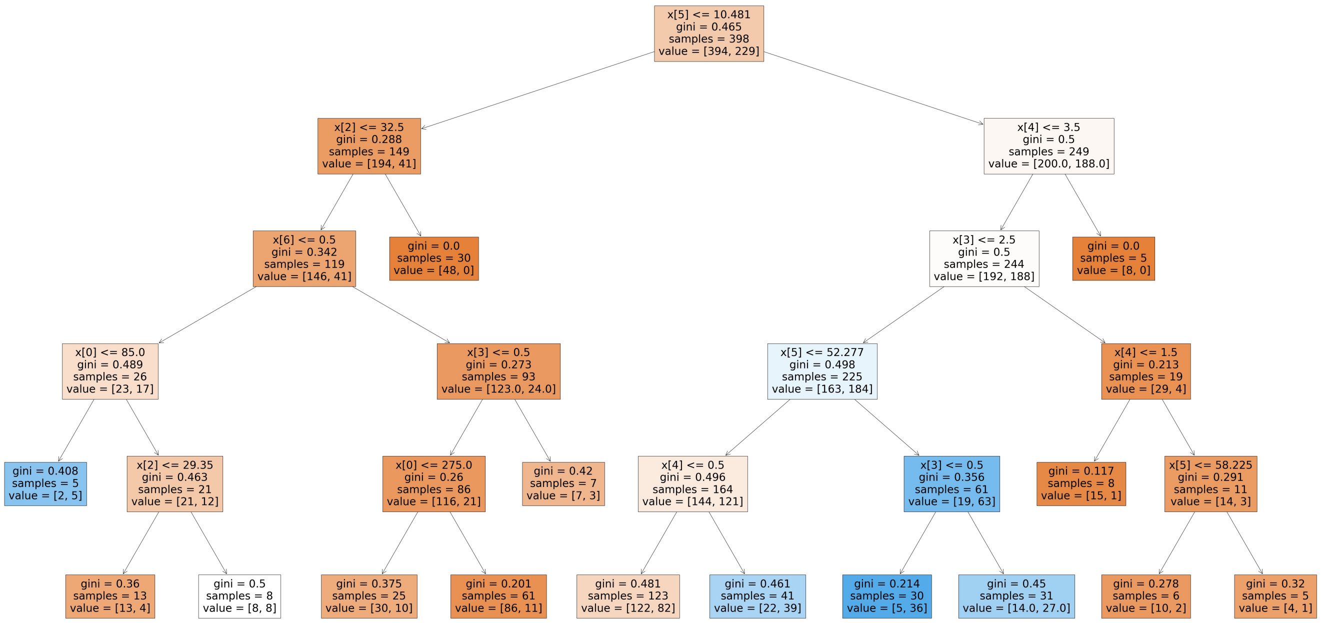
```
RandomForestClassifier(max_depth=5, min_samples_leaf=5, n_estimators=10)
```

In [79]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
```

```
plot_tree(rf_best.estimators_[5], filled=True)
```

Out[79]:

```
[Text(0.5357142857142857, 0.9166666666666666, 'x[5] <= 10.481\ngini = 0.465\nsamples = 398\nvalue = [394, 229]'
),
 Text(0.2857142857142857, 0.75, 'x[2] <= 32.5\ngini = 0.288\nsamples = 149\nvalue = [194, 41]'),
 Text(0.23809523809523808, 0.5833333333333334, 'x[6] <= 0.5\ngini = 0.342\nsamples = 119\nvalue = [146, 41]'),
 Text(0.09523809523809523, 0.4166666666666667, 'x[0] <= 85.0\ngini = 0.489\nsamples = 26\nvalue = [23, 17]'),
 Text(0.047619047619047616, 0.25, 'gini = 0.408\nsamples = 5\nvalue = [2, 5]'),
 Text(0.14285714285714285, 0.25, 'x[2] <= 29.35\ngini = 0.463\nsamples = 21\nvalue = [21, 12]'),
 Text(0.09523809523809523, 0.08333333333333333, 'gini = 0.36\nsamples = 13\nvalue = [13, 4]'),
 Text(0.19047619047619047, 0.08333333333333333, 'gini = 0.5\nsamples = 8\nvalue = [8, 8]'),
 Text(0.38095238095238093, 0.4166666666666667, 'x[3] <= 0.5\ngini = 0.273\nsamples = 93\nvalue = [123.0, 24.0]'
),
 Text(0.3333333333333333, 0.25, 'x[0] <= 275.0\ngini = 0.26\nsamples = 86\nvalue = [116, 21]'),
 Text(0.2857142857142857, 0.08333333333333333, 'gini = 0.375\nsamples = 25\nvalue = [30, 10]'),
 Text(0.38095238095238093, 0.08333333333333333, 'gini = 0.201\nsamples = 61\nvalue = [86, 11]'),
 Text(0.42857142857142855, 0.25, 'gini = 0.42\nsamples = 7\nvalue = [7, 3]'),
 Text(0.3333333333333333, 0.5833333333333334, 'gini = 0.0\nsamples = 30\nvalue = [48, 0]'),
 Text(0.7857142857142857, 0.75, 'x[4] <= 3.5\ngini = 0.5\nsamples = 249\nvalue = [200.0, 188.0]'),
 Text(0.7380952380952381, 0.5833333333333334, 'x[3] <= 2.5\ngini = 0.5\nsamples = 244\nvalue = [192, 188]'),
 Text(0.6190476190476191, 0.4166666666666667, 'x[5] <= 52.277\ngini = 0.498\nsamples = 225\nvalue = [163, 184]'
),
 Text(0.5238095238095238, 0.25, 'x[4] <= 0.5\ngini = 0.496\nsamples = 164\nvalue = [144, 121]'),
 Text(0.47619047619047616, 0.08333333333333333, 'gini = 0.481\nsamples = 123\nvalue = [122, 82]'),
 Text(0.5714285714285714, 0.08333333333333333, 'gini = 0.461\nsamples = 41\nvalue = [22, 39]'),
 Text(0.7142857142857143, 0.25, 'x[3] <= 0.5\ngini = 0.356\nsamples = 61\nvalue = [19, 63]'),
 Text(0.6666666666666666, 0.08333333333333333, 'gini = 0.214\nsamples = 30\nvalue = [5, 36]'),
 Text(0.7619047619047619, 0.08333333333333333, 'gini = 0.45\nsamples = 31\nvalue = [14.0, 27.0]'),
 Text(0.8571428571428571, 0.4166666666666667, 'x[4] <= 1.5\ngini = 0.213\nsamples = 19\nvalue = [29, 4]'),
 Text(0.8095238095238095, 0.25, 'gini = 0.117\nsamples = 8\nvalue = [15, 1]'),
 Text(0.9047619047619048, 0.25, 'x[5] <= 58.225\ngini = 0.291\nsamples = 11\nvalue = [14, 3]'),
 Text(0.8571428571428571, 0.08333333333333333, 'gini = 0.278\nsamples = 6\nvalue = [10, 2]'),
 Text(0.9523809523809523, 0.08333333333333333, 'gini = 0.32\nsamples = 5\nvalue = [4, 1]'),
 Text(0.8333333333333334, 0.5833333333333334, 'gini = 0.0\nsamples = 5\nvalue = [8, 0]')]
```

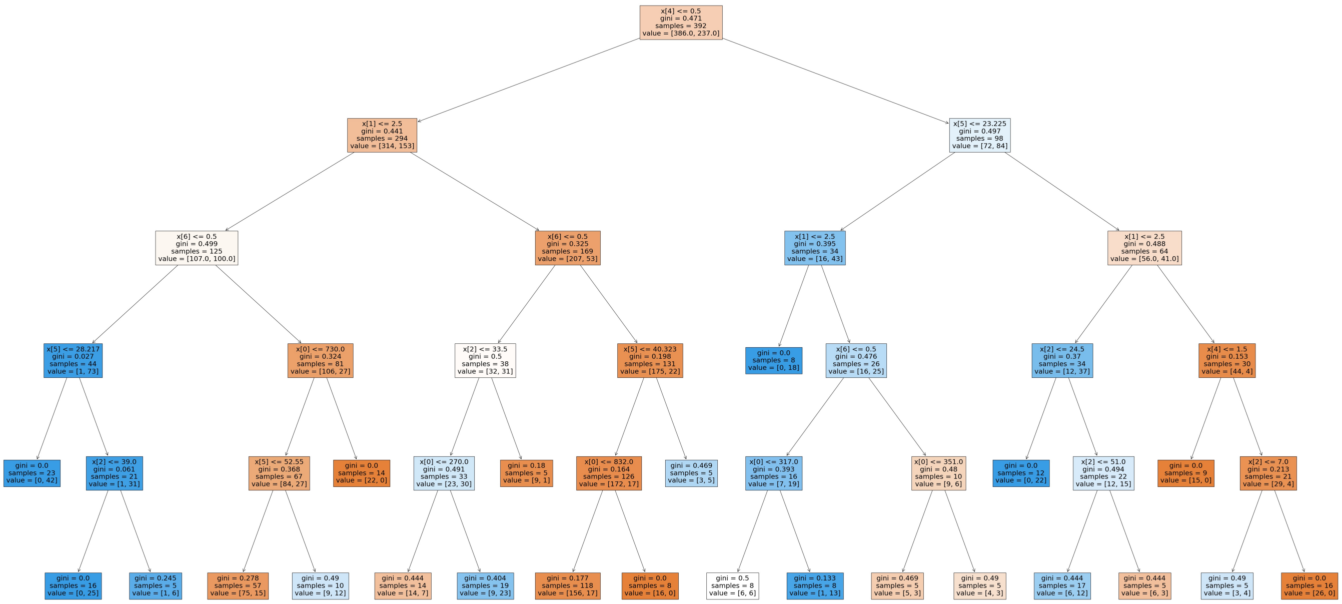

x[5] <= 10.481 gini = 0.465 samples = 398 value = [394, 229]
x[2] <= 32.5 gini = 0.288 samples = 149 value = [194, 41]
x[4] <= 3.5 gini = 0.5 samples = 249 value = [200.0, 188.0]
x[6] <= 0.5 gini = 0.342 samples = 119 value = [146, 41]
gini = 0.0 samples = 30 value = [48, 0]
x[3] <= 2.5 gini = 0.5 samples = 244 value = [192, 188]
gini = 0.0 samples = 5 value = [8, 0]
x[0] <= 85.0 gini = 0.489 samples = 26 value = [23, 17]
x[3] <= 0.5 gini = 0.273 samples = 93 value = [123.0, 24.0]
x[5] <= 52.277 gini = 0.498 samples = 225 value = [163, 184]
x[4] <= 1.5 gini = 0.213 samples = 19 value = [29, 4]
gini = 0.408 samples = 5 value = [2, 5]
x[2] <= 29.35 gini = 0.463 samples = 21 value = [21, 12]
x[0] <= 275.0 gini = 0.26 samples = 86 value = [116, 21]
gini = 0.42 samples = 7 value = [7, 3]
x[4] <= 0.5 gini = 0.496 samples = 164 value = [144, 121]
x[3] <= 0.5 gini = 0.356 samples = 61 value = [19, 63]
gini = 0.117 samples = 8 value = [15, 1]
x[5] <= 58.225 gini = 0.291 samples = 11 value = [14, 3]
gini = 0.36 samples = 13 value = [13, 4]
gini = 0.5 samples = 8 value = [8, 8]
gini = 0.375 samples = 25 value = [30, 10]
gini = 0.201 samples = 61 value = [86, 11]
gini = 0.481 samples = 123 value = [122, 82]
gini = 0.461 samples = 41 value = [22, 39]
gini = 0.214 samples = 30 value = [5, 36]
gini = 0.45 samples = 31 value = [14.0, 27.0]
gini = 0.278 samples = 6 value = [10, 2]
gini = 0.32 samples = 5 value = [4, 1]

In [80]:

```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[7],filled=True)
```

Out[80]:

```
[Text(0.5075757575757576, 0.9166666666666666, 'x[4] <= 0.5\ngini = 0.471\nsamples = 392\nvalue = [386.0, 237.0]
'),
 Text(0.2878787878787879, 0.75, 'x[1] <= 2.5\ngini = 0.441\nsamples = 294\nvalue = [314, 153]'),
 Text(0.15151515151515152, 0.5833333333333334, 'x[6] <= 0.5\ngini = 0.499\nsamples = 125\nvalue = [107.0, 100.0
]'),
 Text(0.06060606060606061, 0.4166666666666667, 'x[5] <= 28.217\ngini = 0.027\nsamples = 44\nvalue = [1, 73]'),
 Text(0.030303030303030304, 0.25, 'gini = 0.0\nsamples = 23\nvalue = [0, 42]'),
 Text(0.09090909090909091, 0.25, 'x[2] <= 39.0\ngini = 0.061\nsamples = 21\nvalue = [1, 31]'),
 Text(0.06060606060606061, 0.08333333333333333, 'gini = 0.0\nsamples = 16\nvalue = [0, 25]'),
 Text(0.12121212121212122, 0.08333333333333333, 'gini = 0.245\nsamples = 5\nvalue = [1, 6]'),
 Text(0.24242424242424243, 0.4166666666666667, 'x[0] <= 730.0\ngini = 0.324\nsamples = 81\nvalue = [106, 27]'),
 Text(0.21212121212121213, 0.25, 'x[5] <= 52.55\ngini = 0.368\nsamples = 67\nvalue = [84, 27]'),
 Text(0.18181818181818182, 0.08333333333333333, 'gini = 0.278\nsamples = 57\nvalue = [75, 15]'),
 Text(0.24242424242424243, 0.08333333333333333, 'gini = 0.49\nsamples = 10\nvalue = [9, 12]'),
 Text(0.2727272727272727, 0.25, 'gini = 0.0\nsamples = 14\nvalue = [22, 0]'),
 Text(0.42424242424242425, 0.5833333333333334, 'x[6] <= 0.5\ngini = 0.325\nsamples = 169\nvalue = [207, 53]'),
 Text(0.36363636363636365, 0.4166666666666667, 'x[2] <= 33.5\ngini = 0.5\nsamples = 38\nvalue = [32, 31]'),
 Text(0.3333333333333333, 0.25, 'x[0] <= 270.0\ngini = 0.491\nsamples = 33\nvalue = [23, 30]'),
 Text(0.30303030303030304, 0.08333333333333333, 'gini = 0.444\nsamples = 14\nvalue = [14, 7]'),
 Text(0.36363636363636365, 0.08333333333333333, 'gini = 0.404\nsamples = 19\nvalue = [9, 23]'),
 Text(0.3939393939393939, 0.25, 'gini = 0.18\nsamples = 5\nvalue = [9, 1]'),
 Text(0.48484848484848486, 0.4166666666666667, 'x[5] <= 40.323\ngini = 0.198\nsamples = 131\nvalue = [175, 22]'
),
 Text(0.45454545454545453, 0.25, 'x[0] <= 832.0\ngini = 0.164\nsamples = 126\nvalue = [172, 17]'),
 Text(0.42424242424242425, 0.08333333333333333, 'gini = 0.177\nsamples = 118\nvalue = [156, 17]'),
 Text(0.48484848484848486, 0.08333333333333333, 'gini = 0.0\nsamples = 8\nvalue = [16, 0]'),
 Text(0.5151515151515151, 0.25, 'gini = 0.469\nsamples = 5\nvalue = [3, 5]'),
 Text(0.7272727272727273, 0.75, 'x[5] <= 23.225\ngini = 0.497\nsamples = 98\nvalue = [72, 84]'),
 Text(0.6060606060606061, 0.5833333333333334, 'x[1] <= 2.5\ngini = 0.395\nsamples = 34\nvalue = [16, 43]'),
 Text(0.5757575757575758, 0.4166666666666667, 'gini = 0.0\nsamples = 8\nvalue = [0, 18]'),
 Text(0.6363636363636364, 0.4166666666666667, 'x[6] <= 0.5\ngini = 0.476\nsamples = 26\nvalue = [16, 25]'),
 Text(0.5757575757575758, 0.25, 'x[0] <= 317.0\ngini = 0.393\nsamples = 16\nvalue = [7, 19]'),
 Text(0.5454545454545454, 0.08333333333333333, 'gini = 0.5\nsamples = 8\nvalue = [6, 6]'),
 Text(0.6060606060606061, 0.08333333333333333, 'gini = 0.133\nsamples = 8\nvalue = [1, 13]'),
 Text(0.696969696969697, 0.25, 'x[0] <= 351.0\ngini = 0.48\nsamples = 10\nvalue = [9, 6]'),
 Text(0.6666666666666666, 0.08333333333333333, 'gini = 0.469\nsamples = 5\nvalue = [5, 3]'),
 Text(0.7272727272727273, 0.08333333333333333, 'gini = 0.49\nsamples = 5\nvalue = [4, 3]'),
 Text(0.8484848484848485, 0.5833333333333334, 'x[1] <= 2.5\ngini = 0.488\nsamples = 64\nvalue = [56.0, 41.0]'),
 Text(0.7878787878787878, 0.4166666666666667, 'x[2] <= 24.5\ngini = 0.37\nsamples = 34\nvalue = [12, 37]'),
 Text(0.7575757575757576, 0.25, 'gini = 0.0\nsamples = 12\nvalue = [0, 22]'),
 Text(0.8181818181818182, 0.25, 'x[2] <= 51.0\ngini = 0.494\nsamples = 22\nvalue = [12, 15]'),
 Text(0.7878787878787878, 0.08333333333333333, 'gini = 0.444\nsamples = 17\nvalue = [6, 12]'),
 Text(0.8484848484848485, 0.08333333333333333, 'gini = 0.444\nsamples = 5\nvalue = [6, 3]'),
 Text(0.9090909090909091, 0.4166666666666667, 'x[4] <= 1.5\ngini = 0.153\nsamples = 30\nvalue = [44, 4]'),
 Text(0.8787878787878788, 0.25, 'gini = 0.0\nsamples = 9\nvalue = [15, 0]'),
 Text(0.9393939393939394, 0.25, 'x[2] <= 7.0\ngini = 0.213\nsamples = 21\nvalue = [29, 4]'),
 Text(0.9090909090909091, 0.08333333333333333, 'gini = 0.49\nsamples = 5\nvalue = [3, 4]'),
 Text(0.9696969696969697, 0.08333333333333333, 'gini = 0.0\nsamples = 16\nvalue = [26, 0]')]
```


x[4] <= 0.5 gini = 0.471 samples = 392 value = [386.0, 237.0]
x[1] <= 2.5 gini = 0.441 samples = 294 value = [314, 153]
x[5] <= 23.225 gini = 0.497 samples = 98 value = [72, 84]
x[6] <= 0.5 gini = 0.499 samples = 125 value = [107.0, 100.0]
x[6] <= 0.5 gini = 0.325 samples = 169 value = [207, 53]
x[1] <= 2.5 gini = 0.395 samples = 34 value = [16, 43]
x[1] <= 2.5 gini = 0.488 samples = 64 value = [56.0, 41.0]
x[5] <= 28.217 gini = 0.027 samples = 44 value = [1, 73]
x[0] <= 730.0 gini = 0.324 samples = 81 value = [106, 27]
x[2] <= 33.5 gini = 0.5 samples = 38 value = [32, 31]
x[5] <= 40.323 gini = 0.198 samples = 131 value = [175, 22]
gini = 0.0 samples = 8 value = [0, 18]
x[6] <= 0.5 gini = 0.476 samples = 26 value = [16, 25]
x[2] <= 24.5 gini = 0.37 samples = 34 value = [12, 37]
x[4] <= 1.5 gini = 0.153 samples = 30 value = [44, 4]
gini = 0.0 samples = 23 value = [0, 42]
x[2] <= 39.0 gini = 0.061 samples = 21 value = [1, 31]
x[5] <= 52.55 gini = 0.368 samples = 67 value = [84, 27]
gini = 0.0 samples = 14 value = [22, 0]
x[0] <= 270.0 gini = 0.491 samples = 33 value = [23, 30]
gini = 0.18 samples = 5 value = [9, 1]
x[0] <= 832.0 gini = 0.164 samples = 126 value = [172, 17]
gini = 0.469 samples = 5 value = [3, 5]
x[0] <= 317.0 gini = 0.393 samples = 16 value = [7, 19]
x[0] <= 351.0 gini = 0.48 samples = 10 value = [9, 6]
gini = 0.0 samples = 12 value = [0, 22]
x[2] <= 51.0 gini = 0.494 samples = 22 value = [12, 15]
gini = 0.0 samples = 9 value = [15, 0]
x[2] <= 7.0 gini = 0.213 samples = 21 value = [29, 4]
gini = 0.0 samples = 16 value = [0, 25]
gini = 0.245 samples = 5 value = [1, 6]
gini = 0.278 samples = 57 value = [75, 15]
gini = 0.49 samples = 10 value = [9, 12]
gini = 0.444 samples = 14 value = [14, 7]
gini = 0.404 samples = 19 value = [9, 23]
gini = 0.177 samples = 118 value = [156, 17]
gini = 0.0 samples = 8 value = [16, 0]
gini = 0.5 samples = 8 value = [6, 6]
gini = 0.133 samples = 8 value = [1, 13]
gini = 0.469 samples = 5 value = [5, 3]
gini = 0.49 samples = 5 value = [4, 3]
gini = 0.444 samples = 17 value = [6, 12]
gini = 0.444 samples = 5 value = [6, 3]
gini = 0.49 samples = 5 value = [3, 4]
gini = 0.0 samples = 16 value = [26, 0]

In [82]:

```
imp_df = pd.DataFrame({"Varname": x_train.columns,"Imp": rf_best.feature_importances_})
imp_df.sort_values(by="Imp", ascending=False)
```

Out[82]:

	Varname	Imp
6	Gender	0.364141
5	Fare	0.229148
1	Pclass	0.112789
2	Age	0.111807
3	SibSp	0.076368
0	PassengerId	0.062734
4	Parch	0.043012

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js