Titanic ML Program

I wanted to take a basic machine learning approach, as I am new to ML, but I wanted to explore it more I used, numpy arrays and decision trees to get the data I removed some of the data that would be difficult to calculate numerically The program prints out the array of 0s and 1s for the test data set. Not all of the passengers were included in the dataset because the program drops some that don't have the right amount of info. I recieved help from a tutor and used sklearn's website as resources to help with the program.

In [1]:

```
%matplotlib inline
import numpy as np
import pandas as pd
import sklearn

from sklearn import tree
import matplotlib.pyplot as plt
```

In [2]:

```
df = pd.read_csv("train.csv")
```

In [3]:

```
df.head()
```

Out[3]:

	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 [4]:

```
df['Sex']= df['Sex'].map({'male':1, 'female':2})
df['Embarked']= df['Embarked'].map({'S':0, 'C':1,'Q':3})
```

In [5]:

```
predictors = list(df.columns.values)
```

In [6]:

```
predictors.remove('Name')
```

In [7]:

```
predictors.remove('PassengerId')
```

In [8]:

```
predictors.remove('Ticket')
```

In [9]:

```
predictors.remove('Cabin')
df=df.dropna()
target = df.Survived
predictors.remove('Survived')
```

In [10]:

```
train=df[predictors]
```

In [11]:

```
train
```

Out[11]:

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
1	1	2	38.0	1	0	71.2833	1.0
3	1	2	35.0	1	0	53.1000	0.0
6	1	1	54.0	0	0	51.8625	0.0
10	3	2	4.0	1	1	16.7000	0.0
11	1	2	58.0	0	0	26.5500	0.0
21	2	1	34.0	0	0	13.0000	0.0
23	1	1	28.0	0	0	35.5000	0.0
27	1	1	19.0	3	2	263.0000	0.0
52	1	2	49.0	1	0	76.7292	1.0
54	1	1	65.0	0	1	61.9792	1.0
62	1	1	45.0	1	0	83.4750	0.0
66	2	2	29.0	0	0	10.5000	0.0
75	3	1	25.0	0	0	7.6500	0.0
88	1	2	23.0	3	2	263.0000	0.0
92	1	1	46.0	1	0	61.1750	0.0
96	1	1	71.0	0	0	34.6542	1.0
97	1	1	23.0	0	1	63.3583	1.0
102	1	1	21.0	0	1	77.2875	0.0
110	1	1	47.0	0	0	52.0000	0.0
118	1	1	24.0	0	1	247.5208	1.0
123	2	2	32.5	0	0	13.0000	0.0
124	1	1	54.0	0	1	77.2875	0.0

136	1	2	19.0	0	2	26.2833	0.0
137	1	1	37.0	1	0	53.1000	0.0
139	1	1	24.0	0	0	79.2000	1.0
148	2	1	36.5	0	2	26.0000	0.0
151	1	2	22.0	1	0	66.6000	0.0
170	1	1	61.0	0	0	33.5000	0.0
174	1	1	56.0	0	0	30.6958	1.0
177	1	2	50.0	0	0	28.7125	1.0
...	...	...	...	...	...	...	...
737	1	1	35.0	0	0	512.3292	1.0
741	1	1	36.0	1	0	78.8500	0.0
742	1	2	21.0	2	2	262.3750	1.0
745	1	1	70.0	1	1	71.0000	0.0
748	1	1	19.0	1	0	53.1000	0.0
751	3	1	6.0	0	1	12.4750	0.0
759	1	2	33.0	0	0	86.5000	0.0
763	1	2	36.0	1	2	120.0000	0.0
765	1	2	51.0	1	0	77.9583	0.0
772	2	2	57.0	0	0	10.5000	0.0
779	1	2	43.0	0	1	211.3375	0.0
781	1	2	17.0	1	0	57.0000	0.0
782	1	1	29.0	0	0	30.0000	0.0
789	1	1	46.0	0	0	79.2000	1.0

796	1	2	49.0	0	0	25.9292	0.0
802	1	1	11.0	1	2	120.0000	0.0
806	1	1	39.0	0	0	0.0000	0.0
809	1	2	33.0	1	0	53.1000	0.0
820	1	2	52.0	1	1	93.5000	0.0
823	3	2	27.0	0	1	12.4750	0.0
835	1	2	39.0	1	1	83.1583	1.0
853	1	2	16.0	0	1	39.4000	0.0
857	1	1	51.0	0	0	26.5500	0.0
862	1	2	48.0	0	0	25.9292	0.0
867	1	1	31.0	0	0	50.4958	0.0
871	1	2	47.0	1	1	52.5542	0.0
872	1	1	33.0	0	0	5.0000	0.0
879	1	2	56.0	0	1	83.1583	1.0
887	1	2	19.0	0	0	30.0000	0.0
889	1	1	26.0	0	0	30.0000	1.0

183 rows × 7 columns

In [12]:

```
X=train
Y=target
clf = tree.DecisionTreeClassifier()

clf.fit(X, Y)
```

Out[12]:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=None, min_samples_leaf=1,
            min_samples_split=2, min_weight_fraction_leaf=0.0,
            presort=False, random_state=None, splitter='best')
```

In [13]:

```
test_df = pd.read_csv("test.csv")
test_df.describe()
```

```
C:\Users\zgdel\Anaconda3\lib\site-packages\numpy\lib\function_base.py:3834: R
untimeWarning: Invalid value encountered in percentile
  RuntimeWarning)
```

Out[13]:

	PassengerId	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	1100.500000	2.265550	30.272590	0.447368	0.392344	35.627188
std	120.810458	0.841838	14.181209	0.896760	0.981429	55.907576
min	892.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	996.250000	1.000000	NaN	0.000000	0.000000	NaN
50%	1100.500000	3.000000	NaN	0.000000	0.000000	NaN
75%	1204.750000	3.000000	NaN	1.000000	0.000000	NaN
max	1309.000000	3.000000	76.000000	8.000000	9.000000	512.329200

In [14]:

```
predictors2 = list(test_df.columns.values)
predictors2.remove('Name')
predictors2.remove('PassengerId')
predictors2.remove('Ticket')
predictors2.remove('Cabin')
test_df = test_df[predictors2]
test_df.describe()
```

```
C:\Users\zgdel\Anaconda3\lib\site-packages\numpy\lib\function_base.py:3834: R
untimeWarning: Invalid value encountered in percentile
  RuntimeWarning)
```

Out[14]:

	Pclass	Age	SibSp	Parch	Fare
count	418.000000	332.000000	418.000000	418.000000	417.000000
mean	2.265550	30.272590	0.447368	0.392344	35.627188
std	0.841838	14.181209	0.896760	0.981429	55.907576
min	1.000000	0.170000	0.000000	0.000000	0.000000
25%	1.000000	NaN	0.000000	0.000000	NaN
50%	3.000000	NaN	0.000000	0.000000	NaN
75%	3.000000	NaN	1.000000	0.000000	NaN
max	3.000000	76.000000	8.000000	9.000000	512.329200

In [15]:

```
test_df = test_df.dropna()
test_df.describe()
```

Out[15]:

	Pclass	Age	SibSp	Parch	Fare
count	331.000000	331.000000	331.000000	331.000000	331.000000
mean	2.141994	30.181269	0.483384	0.398792	40.982087
std	0.846251	14.104573	0.875004	0.811582	61.228558
min	1.000000	0.170000	0.000000	0.000000	0.000000
25%	1.000000	21.000000	0.000000	0.000000	8.050000
50%	2.000000	27.000000	0.000000	0.000000	16.000000
75%	3.000000	39.000000	1.000000	1.000000	40.633350
max	3.000000	76.000000	8.000000	6.000000	512.329200

In [16]:

```
test_df=test_df.dropna()
test_df['Sex']= test_df['Sex'].map({'male':1, 'female':2})
test_df['Embarked']= test_df['Embarked'].map({'S':0, 'C':1,'Q':3})
test_df.describe()

test = test_df[predictors2]
```

In [17]:

```
print(clf.predict(test))
print("Size of testing dataset: ")
print(len(test))
```

```
[0 0 0 1 1 1 0 1 0 0 0 1 0 1 1 1 0 0 0 0 1 0 1 1 1 0 1 1 1 1 0 0 0 1 0 0 1
 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 0 0 1 0 0 0 1 0 0 0 1 1 1 0 0 0 1 0 1 0 0 1
 1 1 1 0 1 0 0 1 1 1 0 0 1 0 0 0 1 0 1 0 0 0 1 1 1 0 1 0 0 1 1 1 0 1 0 1 0
 1 1 0 0 1 1 0 1 0 1 1 1 0 1 0 1 1 1 1 0 1 1 0 0 0 1 1 1 1 0 1 1 0 1 1 1 0
 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 1 0 1 0 1 1 0 1 0 0 0 0 0 1 1 0
 0 0 0 0 1 1 0 1 0 1 1 1 1 0 0 1 1 1 1 1 0 0 0 1 0 0 1 0 1 1 1 0 0 1 0 1 1
 0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 1 0 1 0 0 0 1 0 0 0 1 1 0 0 1 1 1 0 1 0 0 0
 1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 1 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 0 1
 1 0 0 0 1 1 1 1 0 1 0 1 0 0 1 0 0 1 0 1 0 0 1 0 1 1 1 0 1 0 0 1 0 1 0]
Size of testing dataset: 
331
```

Looking at test dataset

A good deal of samples were dropped using `dropna()`.

In [ ]:

In [ ]:

In [ ]:

In [ ]: