In [7]:
```
pwd
```

Out[7]:
```
'C:\\Users\\anshu'
```

In [10]:
```
cd C:/Users/anshu/Desktop/UNCC Courses/Data Science/datasets/
```

```
C:\Users\anshu\Desktop\UNCC Courses\Data Science\datasets
```

In [11]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import math
```

In [12]:
```
titanic = pd.read_csv("titanic.csv")
titanic.head()
```

Out[12]:

	Survived	Pclass	Name	Sex	Age	Siblings/Spouses Aboard	Parents/Children Aboard	Fare
0	0	3	Mr. Owen Harris Braund	male	22.0	1	0	7.2500
1	1	1	Mrs. John Bradley (Florence Briggs Thayer) Cum...	female	38.0	1	0	71.2833
2	1	3	Miss. Laina Heikkinen	female	26.0	0	0	7.9250
3	1	1	Mrs. Jacques Heath (Lily May Peel) Futrelle	female	35.0	1	0	53.1000
4	0	3	Mr. William Henry Allen	male	35.0	0	0	8.0500

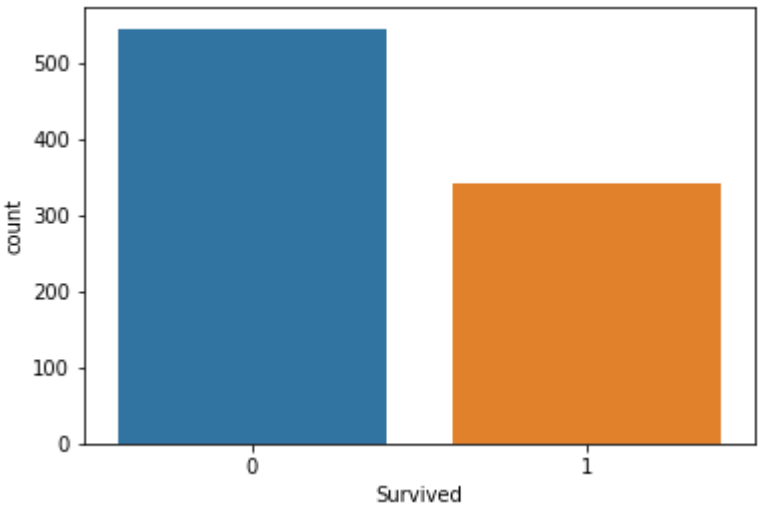
In [13]:
```
print("# of passengers in the dataset:"+str(len(titanic.index)))
```

```
# of passengers in the dataset:887
```

Analysis of Dataset

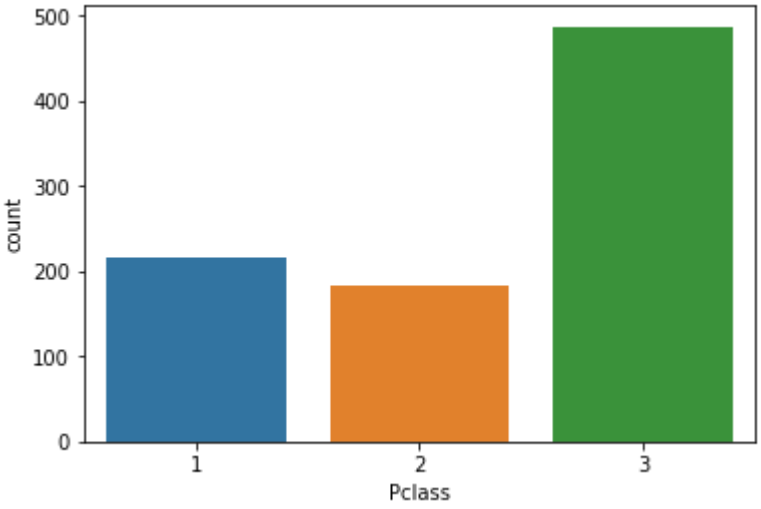
In [14]:
```
sns.countplot(x="Survived", data=titanic)
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x27983495eb8>



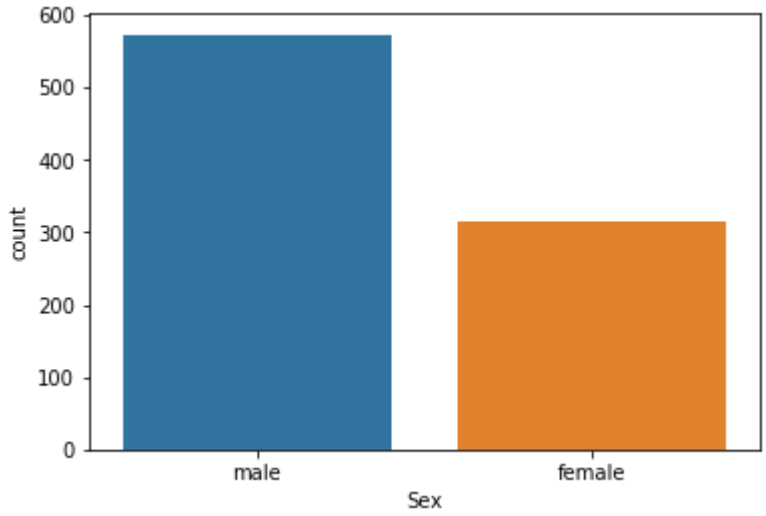
In [15]:
```
sns.countplot(x="Pclass", data=titanic)
```

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x279847a2e48>



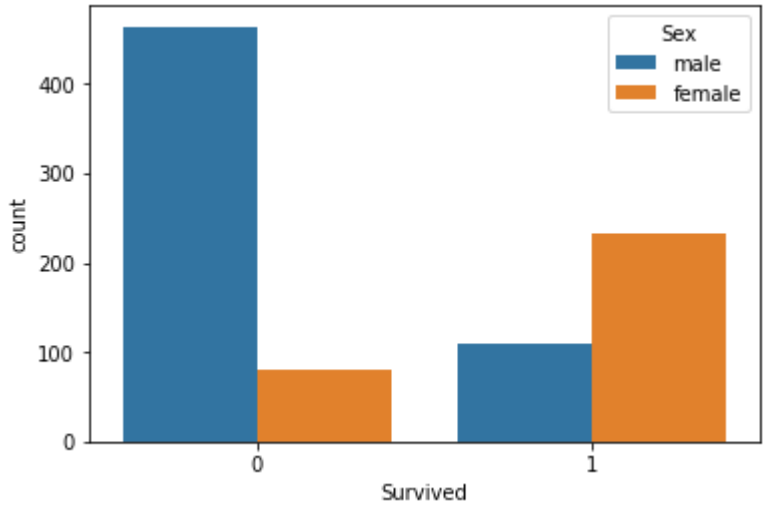
In [16]:
```
sns.countplot(x="Sex",data= titanic)
```

Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x279848175f8>



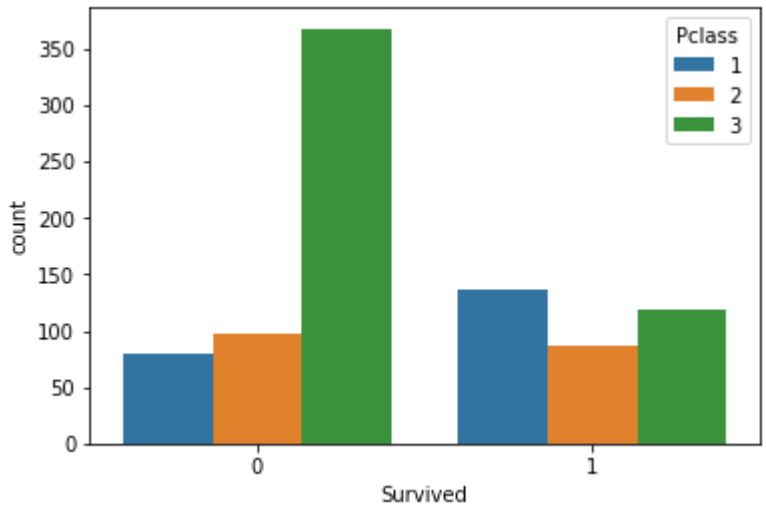
In [17]:
```
sns.countplot(x="Survived", hue="Sex", data = titanic)
```

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x2798486c668>



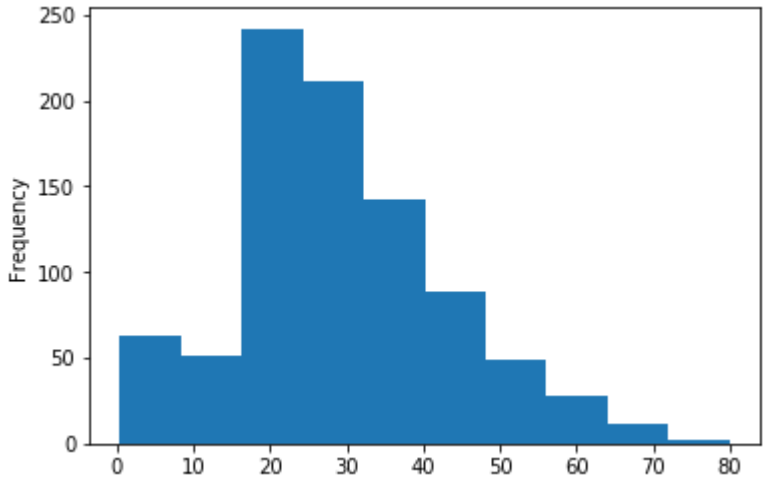
In [18]:
```
sns.countplot(x="Survived", hue="Pclass",data=titanic)
```

Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x279848ba6d8>



In [19]:
```
titanic["Age"].plot.hist()
```

Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x2798491bbe0>



Data Cleaning

In [20]:
```
titanic.isnull()
```

Out[20]:

	Survived	Pclass	Name	Sex	Age	Siblings/Spouses Aboard	Parents/Children Aboard	Fare
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False
5	False	False	False	False	False	False	False	False
6	False	False	False	False	False	False	False	False
7	False	False	False	False	False	False	False	False
8	False	False	False	False	False	False	False	False
9	False	False	False	False	False	False	False	False
10	False	False	False	False	False	False	False	False
11	False	False	False	False	False	False	False	False
12	False	False	False	False	False	False	False	False
13	False	False	False	False	False	False	False	False
14	False	False	False	False	False	False	False	False
15	False	False	False	False	False	False	False	False
16	False	False	False	False	False	False	False	False
17	False	False	False	False	False	False	False	False
18	False	False	False	False	False	False	False	False
19	False	False	False	False	False	False	False	False
20	False	False	False	False	False	False	False	False
21	False	False	False	False	False	False	False	False
22	False	False	False	False	False	False	False	False
23	False	False	False	False	False	False	False	False
24	False	False	False	False	False	False	False	False
25	False	False	False	False	False	False	False	False
26	False	False	False	False	False	False	False	False
27	False	False	False	False	False	False	False	False
28	False	False	False	False	False	False	False	False
29	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...
857	False	False	False	False	False	False	False	False
858	False	False	False	False	False	False	False	False

	Survived	Pclass	Name	Sex	Age	Siblings/Spouses Aboard	Parents/Children Aboard	Fare
859	False	False	False	False	False	False	False	False
860	False	False	False	False	False	False	False	False
861	False	False	False	False	False	False	False	False
862	False	False	False	False	False	False	False	False
863	False	False	False	False	False	False	False	False
864	False	False	False	False	False	False	False	False
865	False	False	False	False	False	False	False	False
866	False	False	False	False	False	False	False	False
867	False	False	False	False	False	False	False	False
868	False	False	False	False	False	False	False	False
869	False	False	False	False	False	False	False	False
870	False	False	False	False	False	False	False	False
871	False	False	False	False	False	False	False	False
872	False	False	False	False	False	False	False	False
873	False	False	False	False	False	False	False	False
874	False	False	False	False	False	False	False	False
875	False	False	False	False	False	False	False	False
876	False	False	False	False	False	False	False	False
877	False	False	False	False	False	False	False	False
878	False	False	False	False	False	False	False	False
879	False	False	False	False	False	False	False	False
880	False	False	False	False	False	False	False	False
881	False	False	False	False	False	False	False	False
882	False	False	False	False	False	False	False	False
883	False	False	False	False	False	False	False	False
884	False	False	False	False	False	False	False	False
885	False	False	False	False	False	False	False	False
886	False	False	False	False	False	False	False	False

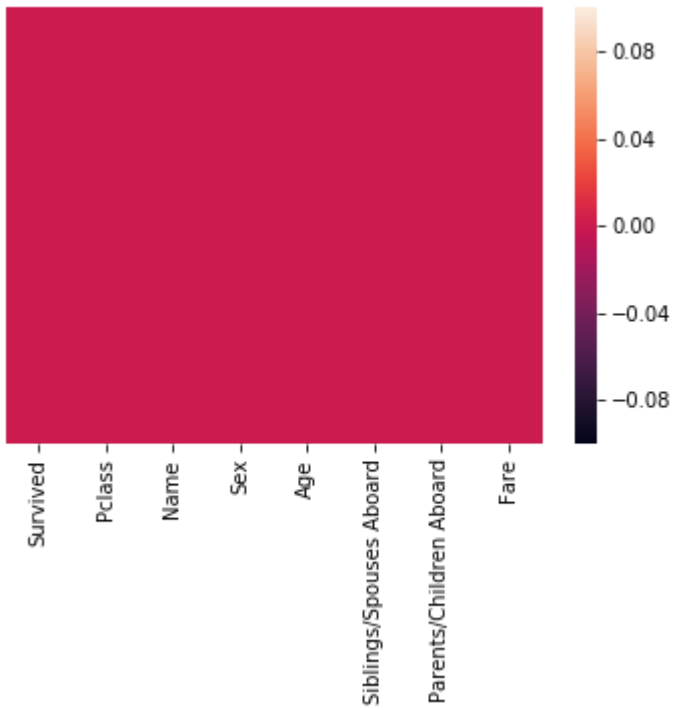
887 rows × 8 columns

In [21]:
```
titanic.isnull().sum()
```

Out[21]:
```
Survived                    0
Pclass                      0
Name                        0
Sex                         0
Age                         0
Siblings/Spouses Aboard     0
Parents/Children Aboard     0
Fare                        0
dtype: int64
```

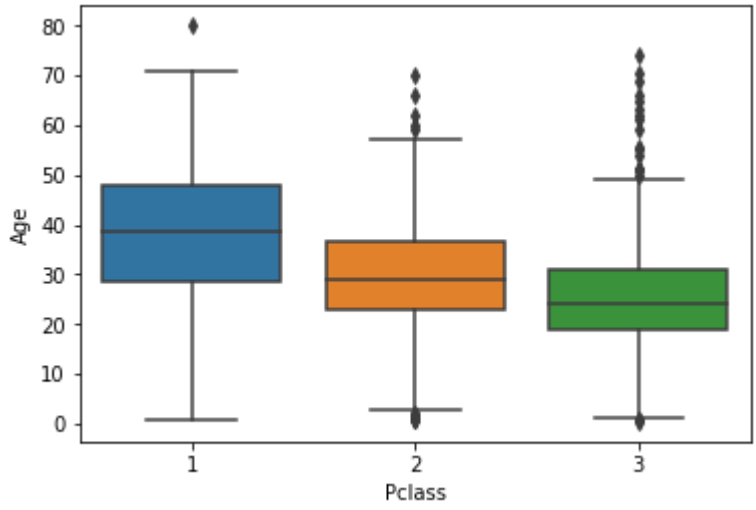
In [22]:
```
#plotting a heat map just to show the other way of finding the null value
sns.heatmap(titanic.isnull(),yticklabels=False)
```

Out[22]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x27984a04f28>
```



In [23]:
```
#checking the box plot to see the outliers
sns.boxplot(x="Pclass", y="Age", data=titanic)
```

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x27984a99da0>



In [24]:
```
#if nan present in dataset, then simply droping it
titanic.dropna(inplace=True)
```

In [25]:
```
#creating dummies variable as it is not categorical data
Sex = pd.get_dummies(titanic['Sex'])
Sex.head()
```

Out[25]:

	female	male
0	0	1
1	1	0
2	1	0
3	1	0
4	0	1

In [26]:
```
#droping the female column as we can interpret the above data using only the
Sex = pd.get_dummies(titanic['Sex'],drop_first = True)
Sex.head()
```

Out[26]:

	male
0	1
1	0
2	0
3	0
4	1

In [27]:
```
Pclass = pd.get_dummies(titanic['Pclass'],drop_first = True)
Pclass.head()
```

Out[27]:

	2	3
0	0	1
1	0	0
2	0	1
3	0	0
4	0	1

In [28]:
```
titanic = pd.concat([titanic,Pclass,Sex],axis=1)
titanic.head()
```

Out[28]:

	Survived	Pclass	Name	Sex	Age	Siblings/Spouses Aboard	Parents/Children Aboard	Fare	2	3
0	0	3	Mr. Owen Harris Braund	male	22.0	1	0	7.2500	0	1
1	1	1	Mrs. John Bradley (Florence Briggs Thayer) Cum...	female	38.0	1	0	71.2833	0	0
2	1	3	Miss. Laina Heikkinen	female	26.0	0	0	7.9250	0	1
3	1	1	Mrs. Jacques Heath (Lily May Peel) Futrelle	female	35.0	1	0	53.1000	0	0
4	0	3	Mr. William Henry Allen	male	35.0	0	0	8.0500	0	1

In [29]:
```
#renaming the dummies column
titanic.rename(columns={2:"2nd Class",3:"3rd Class"},inplace=True)
```

In [30]:
```
titanic.head()
```

Out[30]:

	Survived	Pclass	Name	Sex	Age	Siblings/Spouses Aboard	Parents/Children Aboard	Fare	2n Clas
0	0	3	Mr. Owen Harris Braund	male	22.0	1	0	7.2500	
1	1	1	Mrs. John Bradley (Florence Briggs Thayer) Cum...	female	38.0	1	0	71.2833	
2	1	3	Miss. Laina Heikkinen	female	26.0	0	0	7.9250	
3	1	1	Mrs. Jacques Heath (Lily May Peel) Futrelle	female	35.0	1	0	53.1000	
4	0	3	Mr. William Henry Allen	male	35.0	0	0	8.0500	

In [31]:
```
titanic.drop(['Pclass','Sex','Name'],axis=1,inplace=True)
titanic.head()
```

Out[31]:

	Survived	Age	Siblings/Spouses Aboard	Parents/Children Aboard	Fare	2nd Class	3rd Class	male
0	0	22.0	1	0	7.2500	0	1	1
1	1	38.0	1	0	71.2833	0	0	0
2	1	26.0	0	0	7.9250	0	1	0
3	1	35.0	1	0	53.1000	0	0	0
4	0	35.0	0	0	8.0500	0	1	1

In [32]:
```
#To check the details of the dataset and its impact on the model
#Before this, install pandas_profiling from anaconda cmd using "pip install p
import pandas_profiling
pandas_profiling.ProfileReport(titanic)
```

Out[32]:

Overview

Dataset info

Number of variables	8
Number of observations	887
Total Missing (%)	0.0%
Total size in memory	44.2 KiB
Average record size in memory	51.0 B

Variables types

Numeric	4
Categorical	0
Boolean	4

Training and Testing my Dataset

In [33]:
```
#Defining my independent and dependent variable
X=titanic.drop('Survived',axis=1)
Y=titanic["Survived"]
```

In [34]:
```
#Spliting Data into Training and Testing subset
from sklearn.model_selection import train_test_split
```

In [35]:
```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.3)
```

In [36]:
```
#Building my Logistic Regression Model
from sklearn.linear_model import LogisticRegression
```

In [37]:
```
logmodel = LogisticRegression()
logmodel.fit(X_train,Y_train)
```

```
C:\Users\anshu\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)
```

Out[37]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=Tru
e,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
```

In [38]:
```
#predicting the model
predictions = logmodel.predict(X_test)
```

In [39]:
```
#Calculating the accuracy and classification Report
from sklearn.metrics  import classification_report
classification_report(Y_test,predictions)
```

Out[39]:
```
'              precision    recall  f1-score   support\n\n           0
0.82      0.86      0.84       175\n           1       0.71      0.64
0.67        92\n\n   micro avg       0.79      0.79      0.79       267\n
macro avg       0.77      0.75      0.76       267\nweighted avg       0.78
0.79      0.78       267\n'
```

In [40]:
```
#printing confusion matrix
from sklearn.metrics import confusion_matrix
confusion_matrix (Y_test,predictions)
```

Out[40]:
```
array([[151,  24],
       [ 33,  59]], dtype=int64)
```

In [41]:
```
#Accuracy for the precition Model
from sklearn.metrics import accuracy_score
accuracy_score(Y_test,predictions)
```

Out[41]:
```
0.7865168539325843
```

In [42]:
```
#Now, lets check the P-value of Variables and see if any variable has to be d
import statsmodels.discrete.discrete_model as sm
logit=sm.Logit(Y_train,X_train)
logit.fit().summary()
```

```
Optimization terminated successfully.
         Current function value: 0.491351
         Iterations 6
```

Out[42]:

Logit Regression Results

Dep. Variable:	Survived	**No. Observations:**	620
Model:	Logit	**Df Residuals:**	613
Method:	MLE	**Df Model:**	6
Date:	Mon, 24 Jun 2019	**Pseudo R-squ.:**	0.2713
Time:	19:52:33	**Log-Likelihood:**	-304.64
converged:	True	**LL-Null:**	-418.06
		LLR p-value:	3.591e-46

	coef	std err	z	P>|z|	[0.025	0.975]
Age	0.0050	0.006	0.838	0.402	-0.007	0.017
Siblings/Spouses Aboard	-0.2774	0.113	-2.451	0.014	-0.499	-0.056
Parents/Children Aboard	-0.1823	0.133	-1.371	0.170	-0.443	0.078
Fare	0.0197	0.004	5.328	0.000	0.012	0.027
2nd Class	0.8162	0.257	3.170	0.002	0.312	1.321
3rd Class	0.1987	0.212	0.939	0.348	-0.216	0.614
male	-2.2861	0.213	-10.738	0.000	-2.703	-1.869

In [43]:
```
##droping 3rd class and parent/children column as its p value is quite high.
titanic.drop(["3rd Class"],axis=1,inplace=True)
```

In [44]:
```
titanic.drop(["Parents/Children Aboard"],axis=1,inplace=True)
```

In [45]:
```
titanic.head()
```

Out[45]:

	Survived	Age	Siblings/Spouses Aboard	Fare	2nd Class	male
0	0	22.0	1	7.2500	0	1
1	1	38.0	1	71.2833	0	0
2	1	26.0	0	7.9250	0	0
3	1	35.0	1	53.1000	0	0
4	0	35.0	0	8.0500	0	1

In [46]:
```
X=titanic.drop('Survived',axis=1)
Y=titanic["Survived"]
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.3)
logmodel = LogisticRegression()
logmodel.fit(X_train,Y_train)
predictions = logmodel.predict(X_test)
classification_report(Y_test,predictions)
confusion_matrix (Y_test,predictions)
accuracy_score(Y_test,predictions)
logit=sm.Logit(Y_train,X_train)
logit.fit().summary()
```

```
Optimization terminated successfully.
         Current function value: 0.484274
         Iterations 6

C:\Users\anshu\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)
```

Out[46]:

Logit Regression Results

Dep. Variable:	Survived	**No. Observations:**	620
Model:	Logit	**Df Residuals:**	615
Method:	MLE	**Df Model:**	4
Date:	Mon, 24 Jun 2019	**Pseudo R-squ.:**	0.2752
Time:	19:52:33	**Log-Likelihood:**	-300.25
converged:	True	**LL-Null:**	-414.26
		LLR p-value:	3.509e-48

	coef	std err	z	P>|z|	[0.025	0.975]
Age	0.0070	0.005	1.304	0.192	-0.004	0.018
Siblings/Spouses Aboard	-0.2529	0.099	-2.550	0.011	-0.447	-0.059
Fare	0.0159	0.003	5.167	0.000	0.010	0.022
2nd Class	0.7580	0.232	3.274	0.001	0.304	1.212
male	-2.2508	0.202	-11.152	0.000	-2.646	-1.855

In [47]:
```
titanic.drop(['Age'],axis=1,inplace=True)
```

In [48]:
```
X=titanic.drop('Survived',axis=1)#independent variable
Y=titanic["Survived"]#dependent variable
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.3)#creating
logmodel = LogisticRegression()
logmodel.fit(X_train,Y_train)
predictions = logmodel.predict(X_test)#predicting testing model
classification_report(Y_test,predictions)
confusion_matrix (Y_test,predictions)#confusion matrix
accuracy_score(Y_test,predictions)#accuracy score
```

```
C:\Users\anshu\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.p
y:433: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Sp
ecify a solver to silence this warning.
  FutureWarning)
```

Out[48]:
```
0.7940074906367042
```

In [49]:
```
logit=sm.Logit(Y_train,X_train)#find p-value for variables
logit.fit().summary()
```

```
Optimization terminated successfully.
         Current function value: 0.493736
         Iterations 6
```

Out[49]:

Logit Regression Results

Dep. Variable:	Survived	**No. Observations:**	620
Model:	Logit	**Df Residuals:**	616
Method:	MLE	**Df Model:**	3
Date:	Mon, 24 Jun 2019	**Pseudo R-squ.:**	0.2611
Time:	19:52:33	**Log-Likelihood:**	-306.12
converged:	True	**LL-Null:**	-414.26
		LLR p-value:	1.270e-46

	coef	std err	z	P>|z|	[0.025	0.975]
Siblings/Spouses Aboard	-0.2413	0.095	-2.537	0.011	-0.428	-0.055
Fare	0.0181	0.003	6.119	0.000	0.012	0.024
2nd Class	0.8905	0.217	4.104	0.000	0.465	1.316
male	-2.0406	0.162	-12.597	0.000	-2.358	-1.723

In [58]:
```
predictions = logmodel.predict(X_test)
```

In [60]:
```
from sklearn.metrics import classification_report
classification_report(Y_test,predictions)
```

Out[60]:
```
'              precision    recall  f1-score   support\n\n           0
0.82      0.86      0.84       166\n           1       0.74      0.69
0.72       101\n\n   micro avg       0.79      0.79      0.79       267\n
macro avg       0.78      0.77      0.78       267\nweighted avg       0.79
0.79      0.79       267\n'
```

In [61]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(Y_test,predictions)
```

Out[61]:
```
array([[142,  24],
       [ 31,  70]], dtype=int64)
```

In [62]:
```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
```

In [63]:
```
model.fit(X_train,Y_train)
```

```
C:\Users\anshu\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246:
FutureWarning: The default value of n_estimators will change from 10 in ver
sion 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

Out[63]:
```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

In [64]:
```
from sklearn.metrics import roc_curve
from sklearn.metrics import roc_auc_score
probs = model.predict_proba(X_test)
```

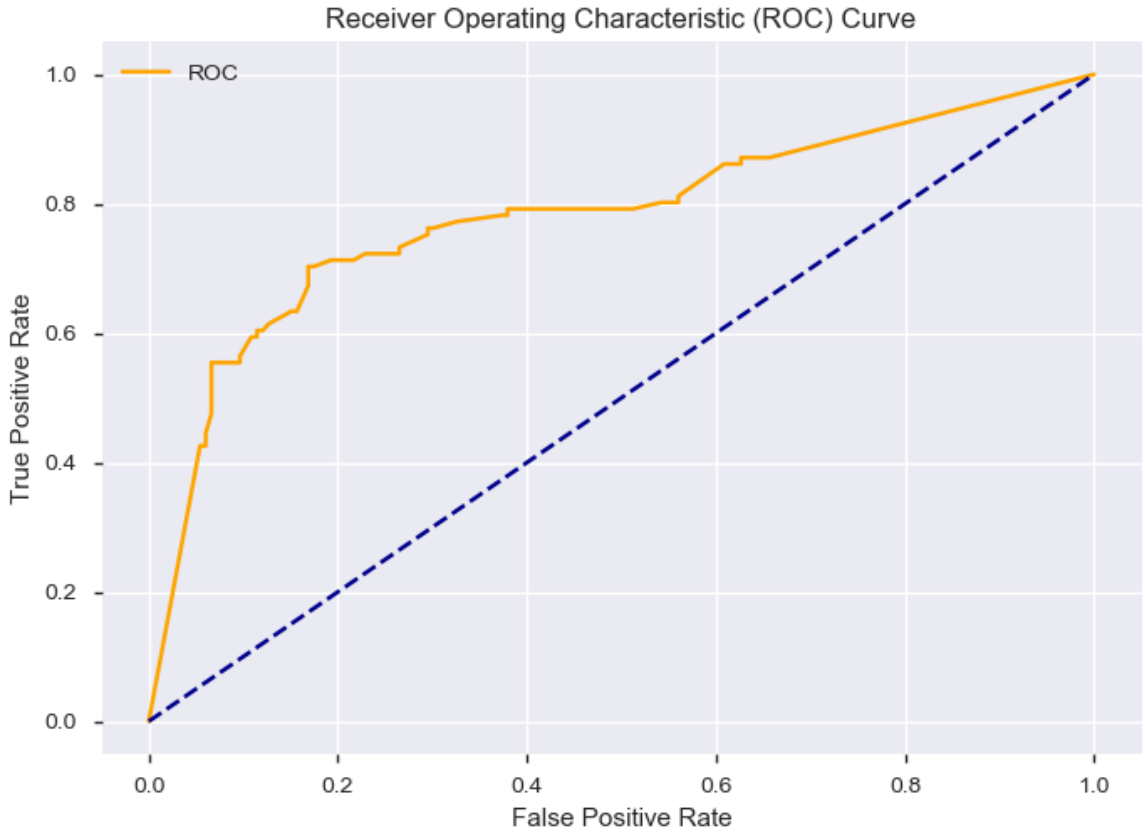
In [67]:
```
auc = roc_auc_score(Y_test,probs)
print('AUC: %.2f' % auc)
```

```
AUC: 0.78
```

In [69]:
```
fpr,tpr,threshold = roc_curve(Y_test, probs)
```

In [70]:
```
def plot_roc_curve(fpr, tpr):
    plt.plot(fpr, tpr, color='orange', label='ROC')
    plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver Operating Characteristic (ROC) Curve')
    plt.legend()
    plt.show()
```

In [71]:
```
from sklearn.datasets import make_classification
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt
plot_roc_curve(fpr,tpr)
```

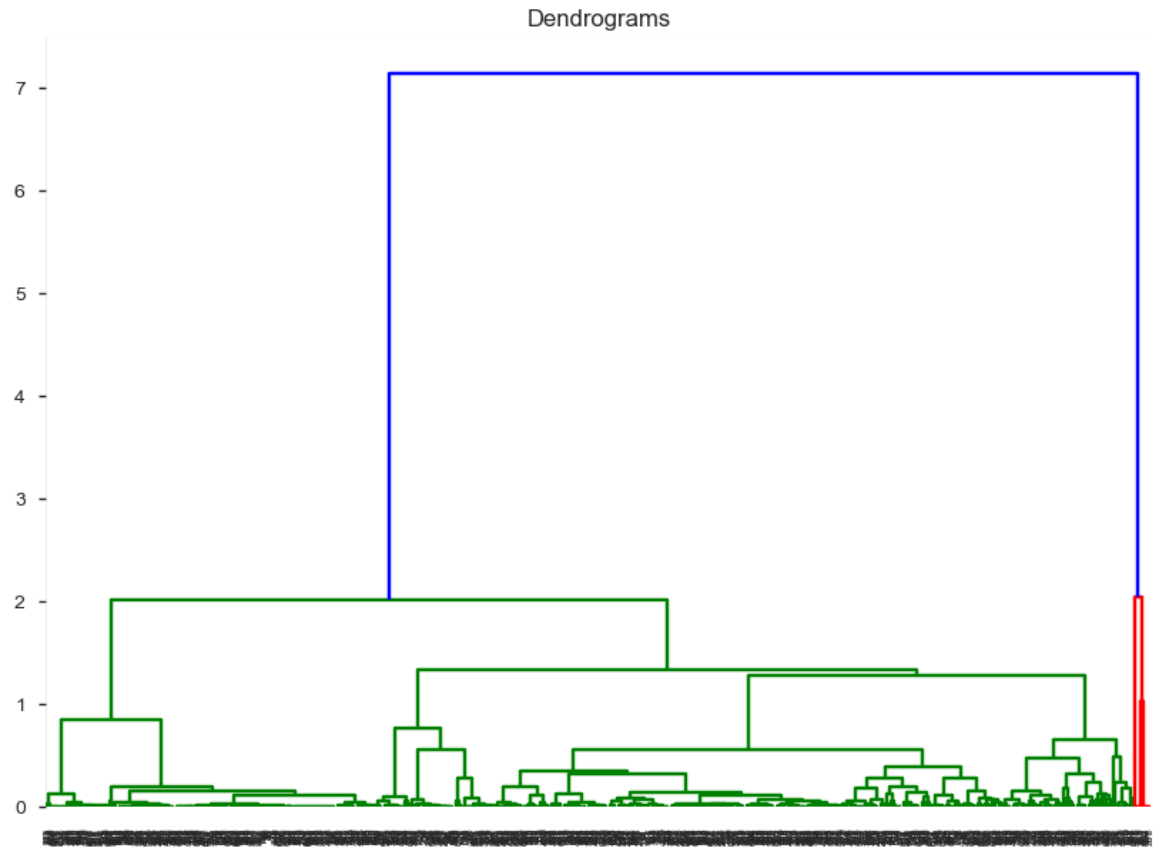


In [50]:
```
from sklearn.preprocessing import normalize
data_scaled = normalize(titanic)
data_scaled = pd.DataFrame(data_scaled, columns=titanic.columns)
data_scaled.head()
```

Out[50]:

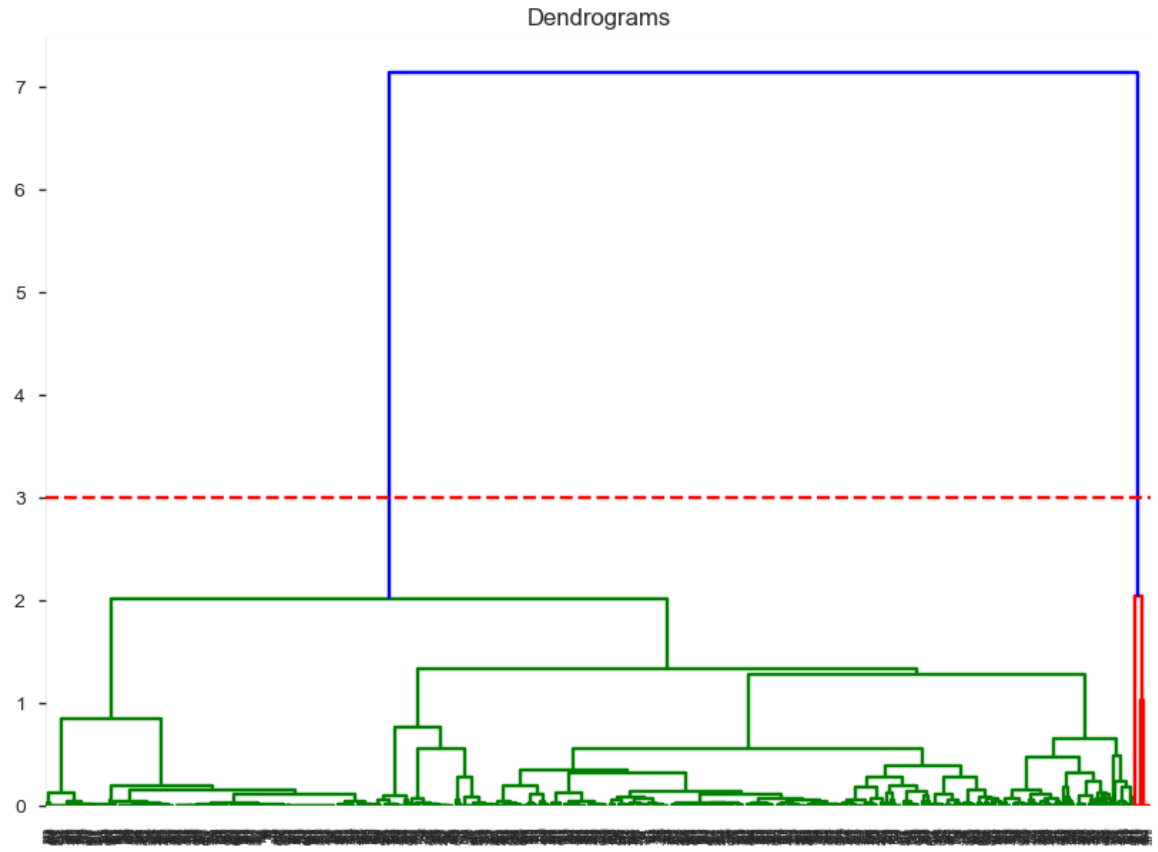
	Survived	Siblings/Spouses Aboard	Fare	2nd Class	male
0	0.000000	0.135379	0.981501	0.0	0.135379
1	0.014026	0.014026	0.999803	0.0	0.000000
2	0.125190	0.000000	0.992133	0.0	0.000000
3	0.018826	0.018826	0.999646	0.0	0.000000
4	0.000000	0.000000	0.992372	0.0	0.123276

In [51]:
```
import scipy.cluster.hierarchy as shc
plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method='ward'))
```



In [72]:
```
#just trying different methods ( this one is for clustering and finding out t
plt.figure(figsize=(10, 7))
plt.title("Dendrograms")
dend = shc.dendrogram(shc.linkage(data_scaled, method='ward'))
plt.axhline(y=3, color='r', linestyle='--')
```

Out[72]: <matplotlib.lines.Line2D at 0x27987378780>



In [73]:

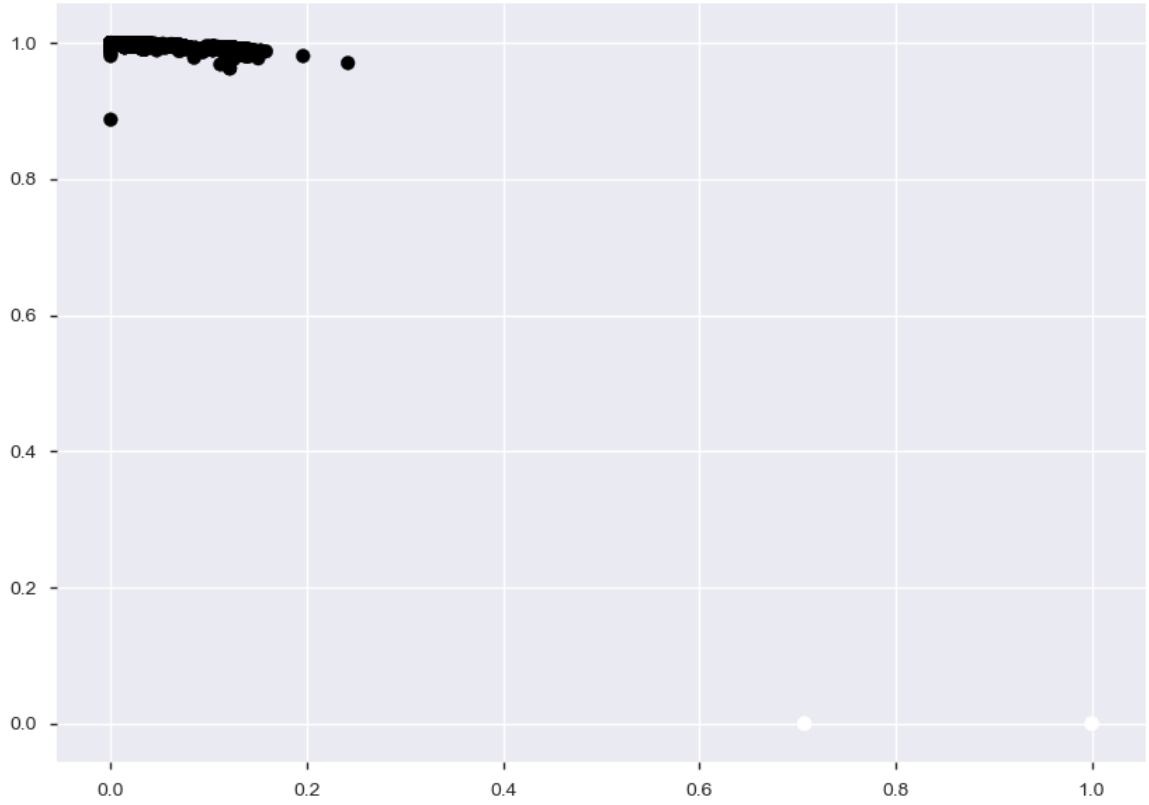
```python
#if taking 2 clusters suggested bt dendogram above
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=2, affinity='euclidean', linkage
cluster.fit_predict(data_scaled)
```

Out[73]:

```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1], dtype=int64)
```

In [74]:
```
plt.figure(figsize=(10, 7))
plt.scatter(data_scaled['male'], data_scaled['Fare'], c=cluster.labels_)
```

Out[74]: <matplotlib.collections.PathCollection at 0x27989079a90>



In [75]:
```
#if changing no of clusters from 2 to 4
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=4, affinity='euclidean', linkage
cluster.fit_predict(data_scaled)
```

Out[75]:
```
array([0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 3, 0, 0, 3, 0, 0, 0, 3, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0,
       3, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 3, 0, 0, 3, 0, 0, 0, 0, 0, 0,
       3, 0, 0, 3, 0, 0, 0, 0, 3, 3, 3, 0, 0, 3, 0, 0, 0, 0, 0, 0, 3, 0,
       3, 3, 3, 0, 0, 3, 3, 0, 0, 0, 0, 0, 3, 0, 3, 0, 3, 0, 0, 3, 0, 0,
       0, 3, 0, 0, 3, 3, 0, 0, 0, 0, 3, 0, 0, 0, 0, 3, 0, 0, 3, 3, 3, 0,
       0, 3, 3, 0, 0, 3, 0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 3, 3, 0, 3, 0, 3,
       0, 0, 3, 3, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0,
       0, 3, 1, 0, 3, 0, 0, 0, 0, 0, 0, 0, 3, 0, 3, 0, 0, 0, 0, 3, 3, 0,
       0, 3, 0, 3, 3, 0, 0, 0, 0, 0, 0, 3, 0, 3, 3, 0, 0, 0, 0, 0, 3, 0,
       3, 3, 3, 0, 3, 0, 3, 3, 0, 0, 3, 3, 0, 3, 0, 0, 0, 3, 3, 0, 0, 3,
       3, 3, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 3,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 2, 0, 0, 3, 3, 3, 0, 0, 3, 0, 3,
       0, 0, 0, 0, 3, 0, 3, 0, 3, 0, 0, 0, 0, 0, 1, 0, 3, 0, 0, 0, 0, 0,
       0, 0, 0, 3, 0, 0, 0, 3, 0, 0, 3, 3, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0,
       0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 3, 3, 3, 0, 0, 0, 0, 3, 3, 0, 0, 0,
       3, 3, 0, 0, 0, 0, 0, 0, 0, 3, 0, 3, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0,
       0, 0, 3, 3, 0, 0, 3, 0, 3, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 3, 0, 0,
       3, 0, 0, 3, 0, 0, 0, 0, 3, 0, 3, 0, 3, 3, 0, 2, 0, 0, 0, 0, 3, 0,
       3, 3, 0, 0, 3, 0, 0, 3, 0, 0, 0, 0, 3, 0, 0, 0, 0, 0, 3, 0, 3, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0, 0, 3, 0, 3, 0, 3, 3,
       3, 2, 0, 3, 0, 3, 3, 0, 0, 0, 0, 0, 0, 3, 0, 0, 2, 3, 0, 0, 0, 0,
       0, 3, 0, 0, 3, 0, 0, 3, 0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 3, 0, 0, 3, 0, 0, 0, 0, 3, 0, 3, 3, 0, 3, 3, 0, 0, 3, 0, 0,
       3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3,
       0, 0, 0, 0, 0, 0, 0, 3, 3, 3, 3, 0, 0, 3, 0, 3, 0, 0, 0, 0, 0, 3,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 3, 0, 3, 3, 3, 0, 3, 0, 0, 0, 0,
       1, 3, 0, 0, 3, 0, 3, 0, 0, 3, 0, 0, 0, 0, 3, 0, 3, 3, 0, 0, 0, 0,
       3, 0, 0, 0, 3, 0, 0, 3, 0, 3, 3, 0, 3, 0, 1, 0, 0, 3, 0, 0, 0, 3,
       0, 0, 0, 0, 0, 3, 0, 3, 0, 3, 0, 3, 0, 0, 0, 3, 0, 3, 0, 0, 3, 0,
       3, 0, 0, 3, 3, 3, 0, 0, 0, 3, 0, 2, 3, 3, 0, 0, 0, 0, 0, 3, 0, 0,
       0, 0, 3, 3, 0, 0, 0, 0, 3, 0, 3, 3, 0, 0, 3, 0, 0, 0, 3, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 3, 3, 3, 0, 0, 3, 0, 0, 3, 3, 0, 3, 0, 0, 0, 0,
       0, 0, 2, 3, 3, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0,
       3, 3, 0, 0, 3, 3, 3, 0, 0, 3, 0, 0, 3, 0, 0, 0, 0, 3, 3, 3, 0, 3,
       0, 3, 3, 0, 3, 0, 0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 3, 0, 0, 0, 3, 3,
       0, 0, 3, 0, 3, 0, 0, 0, 0, 3, 1, 0, 3, 0, 3, 0, 3, 0, 3, 1, 0, 0,
       3, 0, 0, 0, 1, 0, 0, 3, 0, 0, 0, 0, 0, 0, 3, 3, 3, 0, 3, 3, 0, 0,
       3, 3, 0, 3, 3, 3, 0, 3, 0, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0,
       0, 0, 3, 0, 0, 0, 3, 0, 3, 0, 3, 3, 0, 0, 3, 3, 3, 0, 0, 3, 0, 3,
       3, 0, 3, 0, 0, 0, 3], dtype=int64)
```

In [76]:
```
plt.figure(figsize=(10, 7))
plt.scatter(data_scaled['male'], data_scaled['Fare'], c=cluster.labels_)
```

Out[76]: <matplotlib.collections.PathCollection at 0x2798935b780>