Logestic_regression_and_synthetic_data

September 30, 2020

[3]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import math

titanic=pd.read_csv(r"D:\msc3\machine learning\lab6\titanictrain.csv")
titanic.head(10)
```

[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				
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0				
5	6	0	3	Moran, Mr. James	male	NaN	0	0				
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0				
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	0				
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	0				
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0				

```
3       0            113803  53.1000  C123        S
4       0            373450   8.0500   NaN        S
5       0            330877   8.4583   NaN        Q
6       0             17463  51.8625   E46        S
7       1            349909  21.0750   NaN        S
8       2            347742  11.1333   NaN        S
9       0            237736  30.0708   NaN        C
```

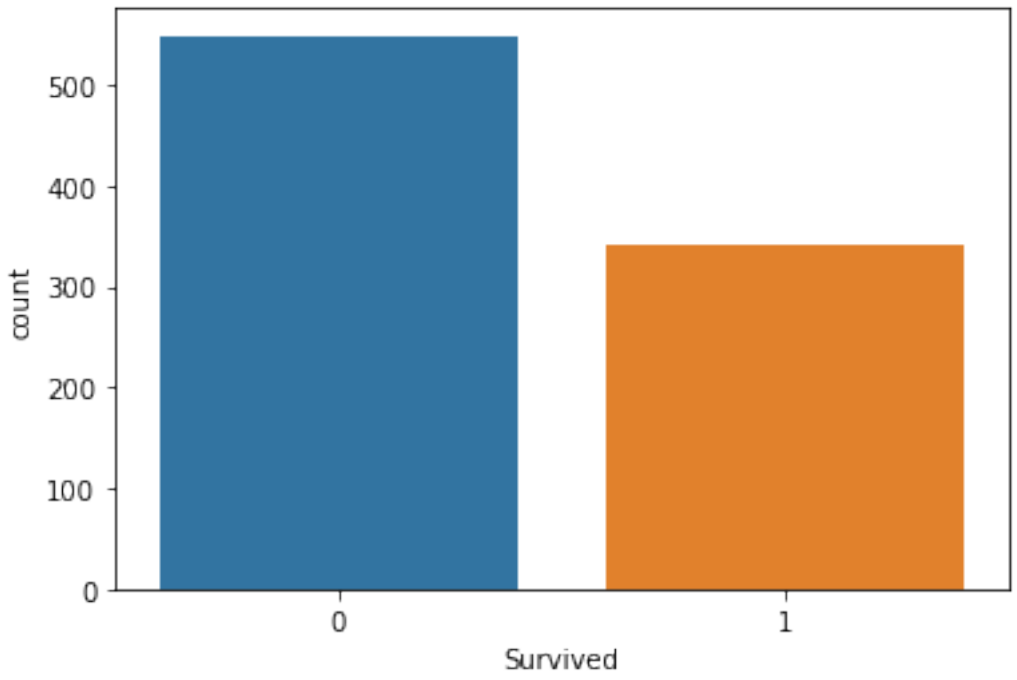
[4]:
```
print("Number of passangers:"+str(len(titanic.index)))
```

```
Number of passangers:891
```

0.1 Analyzing data

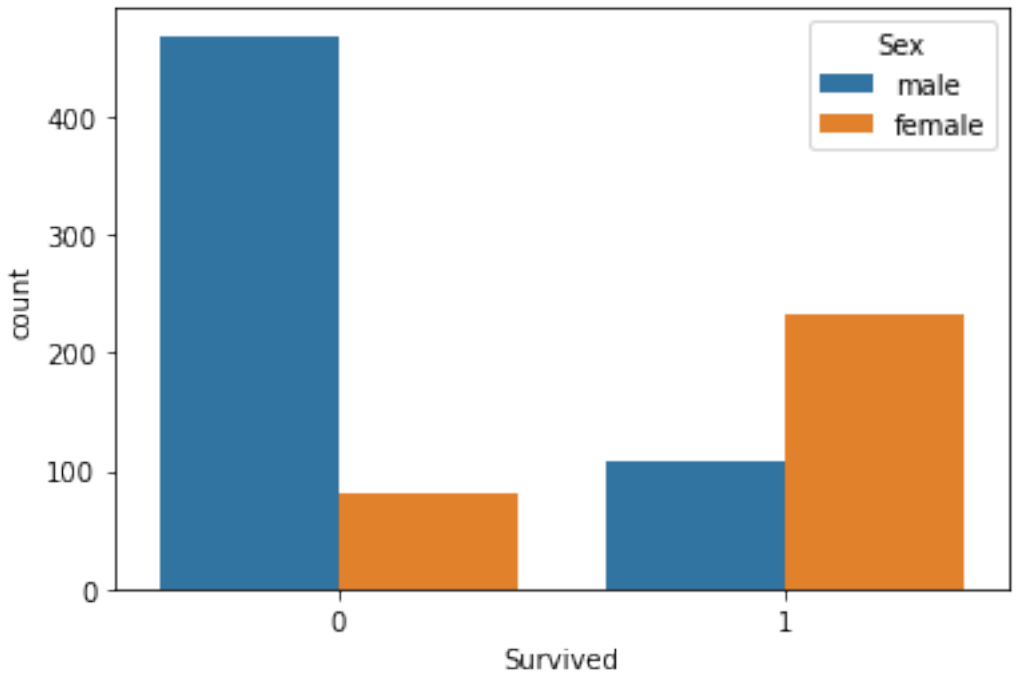
[5]:
```
sns.countplot(x="Survived", data=titanic)
```

[5]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x194cb065188>
```



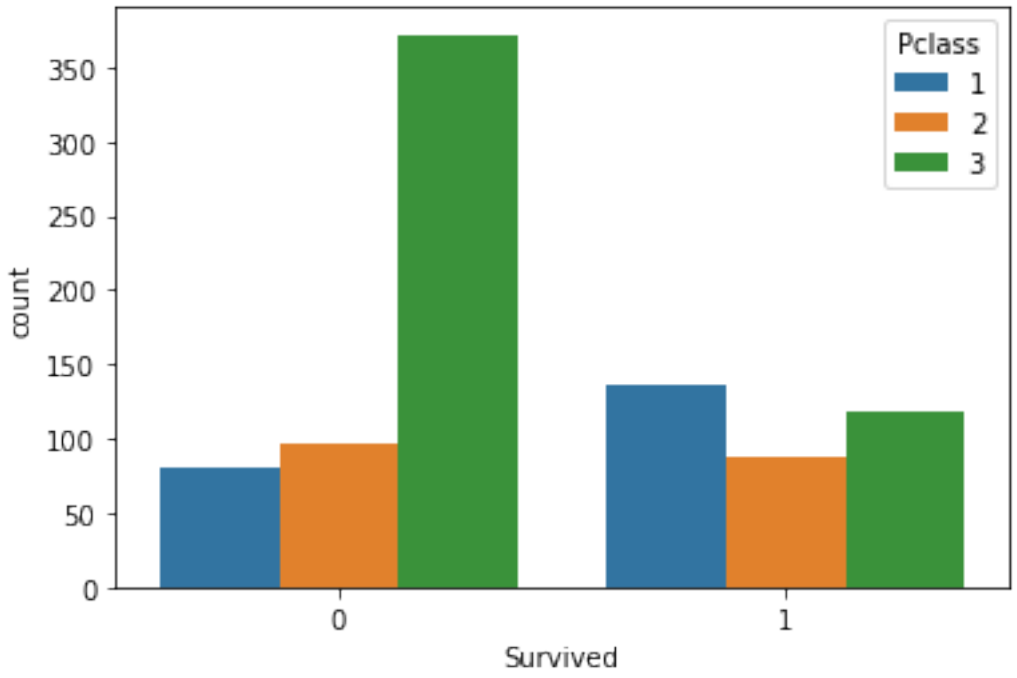
[6]:
```
sns.countplot(x="Survived", hue="Sex",data=titanic)
```

[6]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x194d64839c8>
```



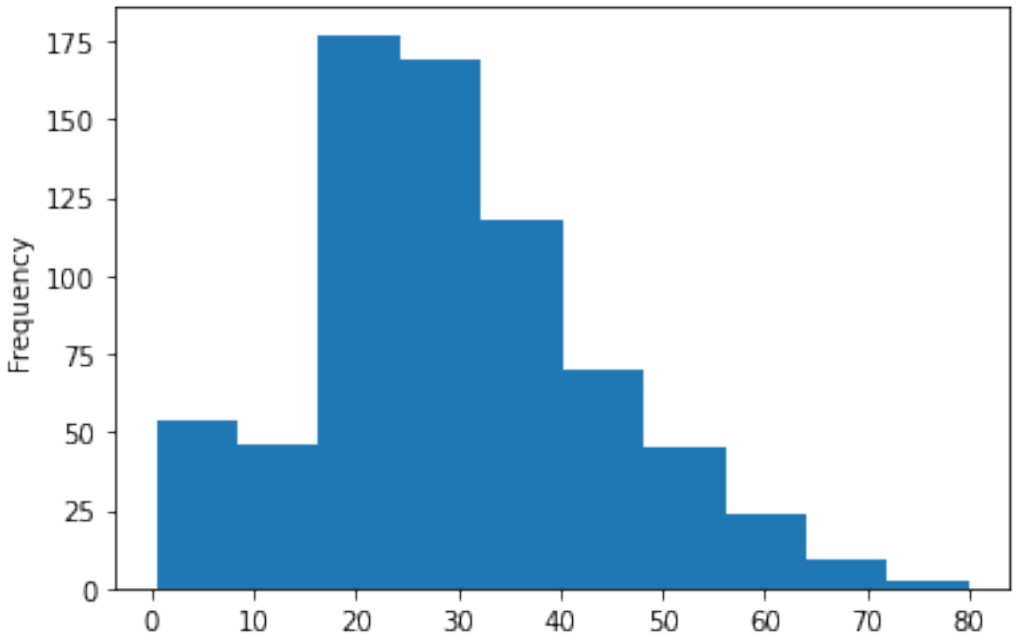
[7]:
```
sns.countplot(x="Survived", hue="Pclass",data=titanic)
```

[7]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x194d65102c8>
```



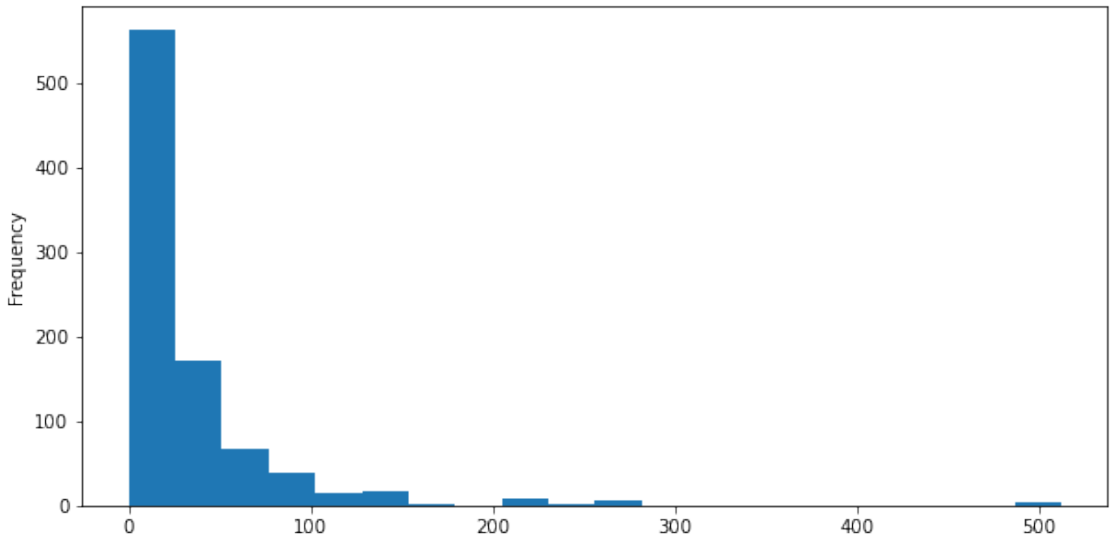
[8]:
```
titanic["Age"].plot.hist()
```

[8]: <matplotlib.axes._subplots.AxesSubplot at 0x194d65916c8>



[9]:
```
titanic["Fare"].plot.hist(bins=20,figsize=(10,5))
```

[9]: <matplotlib.axes._subplots.AxesSubplot at 0x194d6626488>

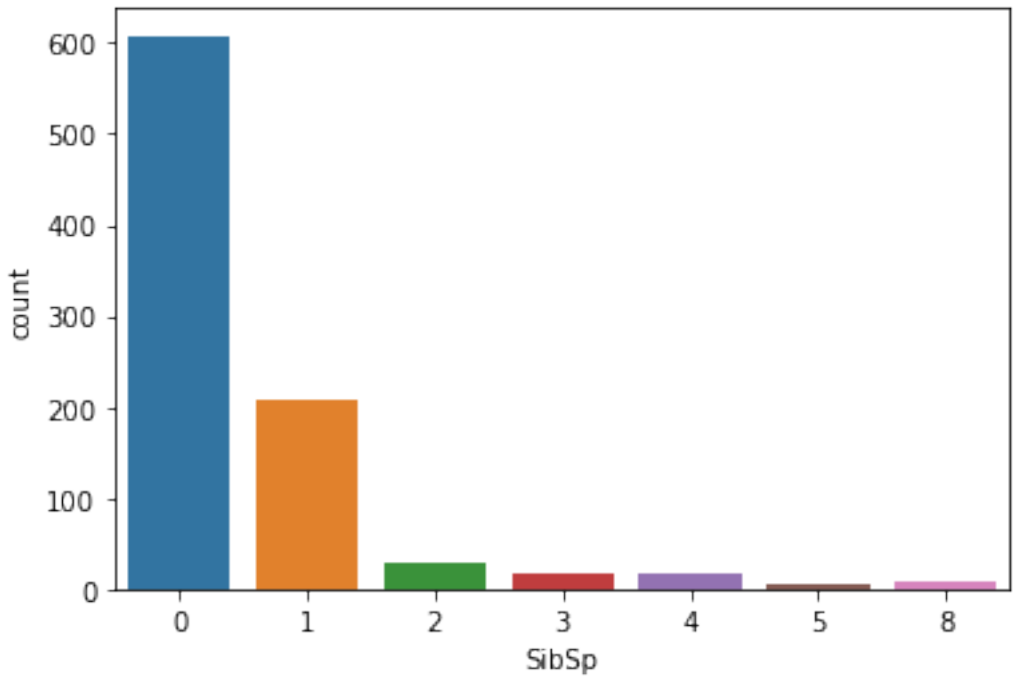


[10]:
```
titanic.info()
```

```
<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
```

[11]:
```
sns.countplot(x="SibSp", data=titanic)
```

[11]: <matplotlib.axes._subplots.AxesSubplot at 0x194d6711288>



0.2 Data Wrangling

[12]:
```
titanic.isnull()
```

[12]:
```
     PassengerId  Survived  Pclass   Name    Sex    Age  SibSp  Parch  Ticket  \
0          False     False   False  False  False  False  False  False   False
1          False     False   False  False  False  False  False  False   False
2          False     False   False  False  False  False  False  False   False
3          False     False   False  False  False  False  False  False   False
4          False     False   False  False  False  False  False  False   False
..           …         …       …      …      …      …      …      …
886        False     False   False  False  False  False  False  False   False
887        False     False   False  False  False  False  False  False   False
888        False     False   False  False  False   True  False  False   False
889        False     False   False  False  False  False  False  False   False
890        False     False   False  False  False  False  False  False   False

      Fare  Cabin  Embarked
0    False   True     False
1    False  False     False
2    False   True     False
3    False  False     False
4    False   True     False
..     …    …       …
886  False   True     False
887  False  False     False
888  False   True     False
889  False  False     False
890  False   True     False

[891 rows x 12 columns]
```

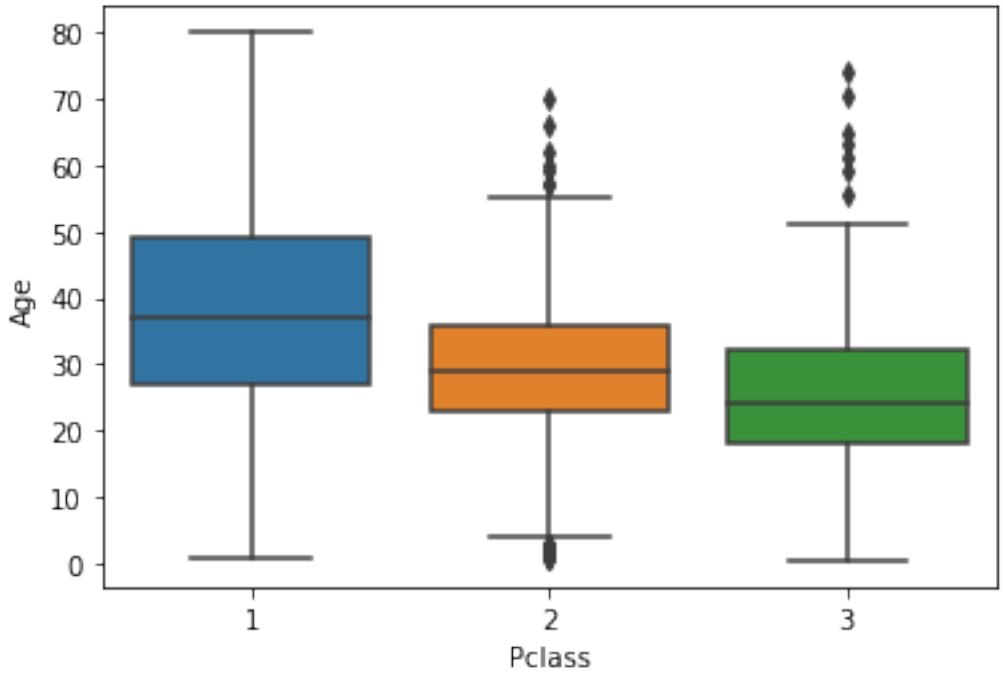
[13]:
```
titanic.isnull().sum()
```

[13]:
```
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
```

[14]:
```
sns.boxplot(x="Pclass", y="Age", data=titanic)
```

[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x194d6780108>
```



[15]:
```
titanic.head(5)
```

[15]:
```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
```

```
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

[16]:
```
titanic.drop("Cabin", axis=1, inplace=True)
```

[17]:
```
titanic.head()
```

[17]:
```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Embarked
0      0         A/5 21171   7.2500        S
1      0          PC 17599  71.2833        C
2      0  STON/O2. 3101282   7.9250        S
3      0            113803  53.1000        S
4      0            373450   8.0500        S
```

[18]:
```
titanic.dropna(inplace=True)
```

[19]:
```
titanic.isnull().sum()
```

[19]:
```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

[20]:
```
sex=pd.get_dummies(titanic['Sex'], drop_first=True)
sex.head()
```

[20]:

	male
0	1
1	0
2	0
3	0
4	1

[21]:
```
embark=pd.get_dummies(titanic['Embarked'], drop_first=True)
embark.head()
```

[21]:

	Q	S
0	0	1
1	0	0
2	0	1
3	0	1
4	0	1

[22]:
```
titanic=pd.concat((titanic,sex,embark),axis=1)
```

[23]:
```
titanic.head()
```

[23]:

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

[24]:
```
titanic.
↪drop(['Sex','Name','PassengerId','Ticket','Embarked'],axis=1,inplace=True)
```

[25]:
```
titanic.head()
```

[25]:

	Survived	Pclass	Age	SibSp	Parch	Fare	male	Q	S
0	0	3	22.0	1	0	7.2500	1	0	1
1	1	1	38.0	1	0	71.2833	0	0	0
2	1	3	26.0	0	0	7.9250	0	0	1
3	1	1	35.0	1	0	53.1000	0	0	1
4	0	3	35.0	0	0	8.0500	1	0	1

0.3 Train and Test

[26]:
```
X=titanic.drop("Survived",axis=1)
y=titanic["Survived"]
```

[27]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.3,␣
 ↪random_state=2)
```

[28]:
```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression(C=1e5)
logreg.fit(X_train,y_train)
predictions=logreg.predict(X_test)
```

```
C:\Users\blr0a\Anaconda3\lib\site-
packages\sklearn\linear_model\_logistic.py:764: ConvergenceWarning: 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
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

[29]:
```
from sklearn.metrics import classification_report
classification_report(y_test,predictions)
```

[29]:
```
'              precision    recall  f1-score   support\n\n           0
0.82      0.79      0.81       130\n           1       0.70      0.74      0.72
84\n\n    accuracy                           0.77       214\n   macro avg
0.76      0.77      0.76       214\nweighted avg       0.77      0.77      0.77
214\n'
```

[30]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,predictions)
```

[30]:
```
array([[103,  27],
       [ 22,  62]], dtype=int64)
```

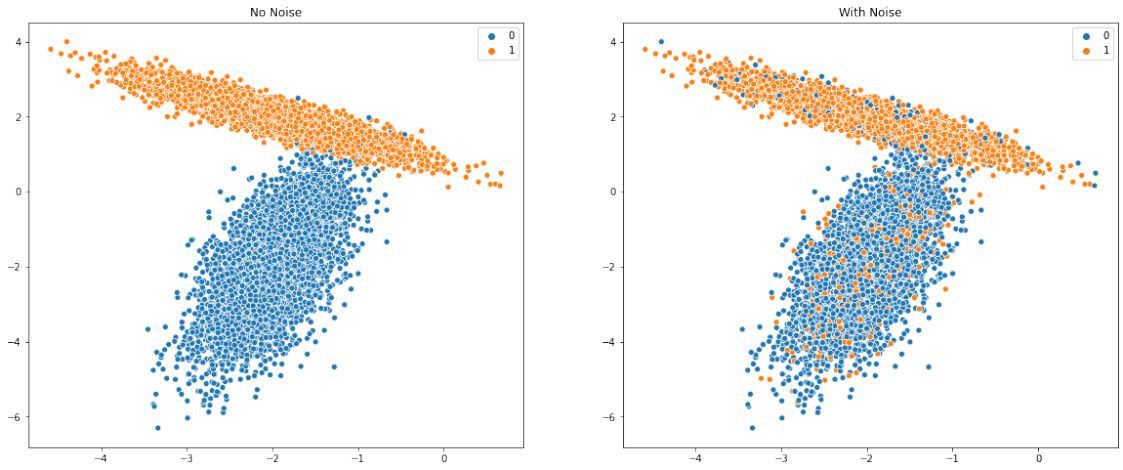
[31]:
```
from sklearn.metrics import accuracy_score
print("train accuracy:")
print(format(logreg.score(X_train,y_train)*100.0))
print("test accuracy:")
accuracy_score(y_test,predictions)
```

```
train accuracy:
80.72289156626506
test accuracy:
```

[31]:
```
0.7710280373831776
```

0.4 USING make_classification

[32]:
```
from sklearn.datasets import make_classification as mc
# Generate Clean data
X,y = mc(n_samples=10000, n_features=2, n_informative=2,n_redundant=0,␣
 ↪n_repeated=0, n_classes=2,␣
 ↪n_clusters_per_class=1,class_sep=2,flip_y=0,weights=[0.5,0.5],␣
 ↪random_state=17)
f, (ax1,ax2) = plt.subplots(nrows=1, ncols=2,figsize=(20,8))
sns.scatterplot(X[:,0],X[:,1],hue=y,ax=ax1);
ax1.set_title("No Noise");
# Generate noisy Data
X,y = mc(n_samples=10000, n_features=2, n_informative=2, n_redundant=0,␣
 ↪n_repeated=0, n_classes=2, n_clusters_per_class=1,class_sep=2,flip_y=0.
 ↪2,weights=[0.5,0.5], random_state=17)
sns.scatterplot(X[:,0],X[:,1],hue=y,ax=ax2);
ax2.set_title("With Noise");
plt.show();
```



0.5 applying logestic regression on make_classification(Noiseless data)

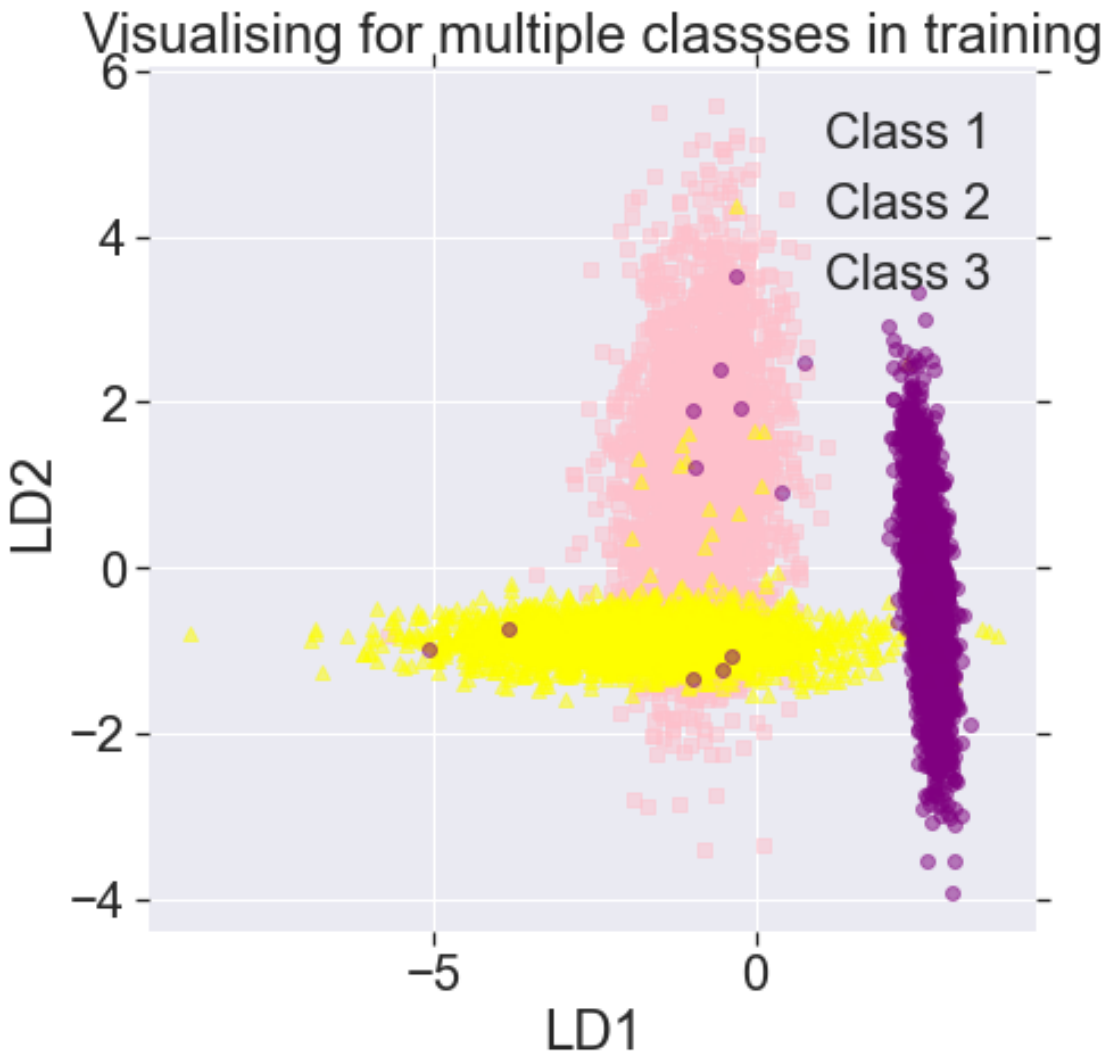
[33]:
```
x,y=mc(n_samples=10000, n_features=8,n_redundant=0, n_repeated=0, n_classes=3,␣
↪n_clusters_per_class=1, random_state=17)
```

[100]:
```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
lda1=LDA(n_components=2)
x_lda=lda1.fit_transform(x,y)
```

[101]:
```
from sklearn.linear_model import LogisticRegression as LR
lr=LR()
x_train,x_test,y_train,y_test=train_test_split(x_lda,y,random_state=0)
l=lr.fit(x_train,y_train)
coef=l.coef_
intercept=l.intercept_
```

[102]:
```
label_dict = {0:'Class 1', 1: 'Class 2', 2: 'Class 3', 3:'Class 4'}
plt.figure(figsize=(7,7))

def plot_scikit_lda(X,y, title):
      ax = plt.subplot(111)
      for label,marker,color in zip(
          [i for i in range(4)],('s','^','o'),('pink','yellow','purple')):

          plt.scatter(x=X[:,0][y == label],
                      y=X[:,1][y == label],
                      marker=marker,
                      color=color,
                      alpha=0.5,
                      label=label_dict[label])

      plt.xlabel('LD1')
      plt.ylabel('LD2')

      leg = plt.legend(loc='upper right', fancybox=True)
      leg.get_frame().set_alpha(0)
      plt.title(title)

      # hide axis ticks
      plt.tick_params(axis="both", which="both", bottom="off", top="off",
              labelbottom="on", left="off", right="off", labelleft="on")

      # remove axis spines
      ax.spines["top"].set_visible(False)
```

```
        ax.spines["right"].set_visible(False)
        ax.spines["bottom"].set_visible(False)
        ax.spines["left"].set_visible(False)
        plt.tight_layout
        plt.show()

plot_scikit_lda(x_train, y_train, title='Visualising for multiple classses in␣
 ↪training')
```

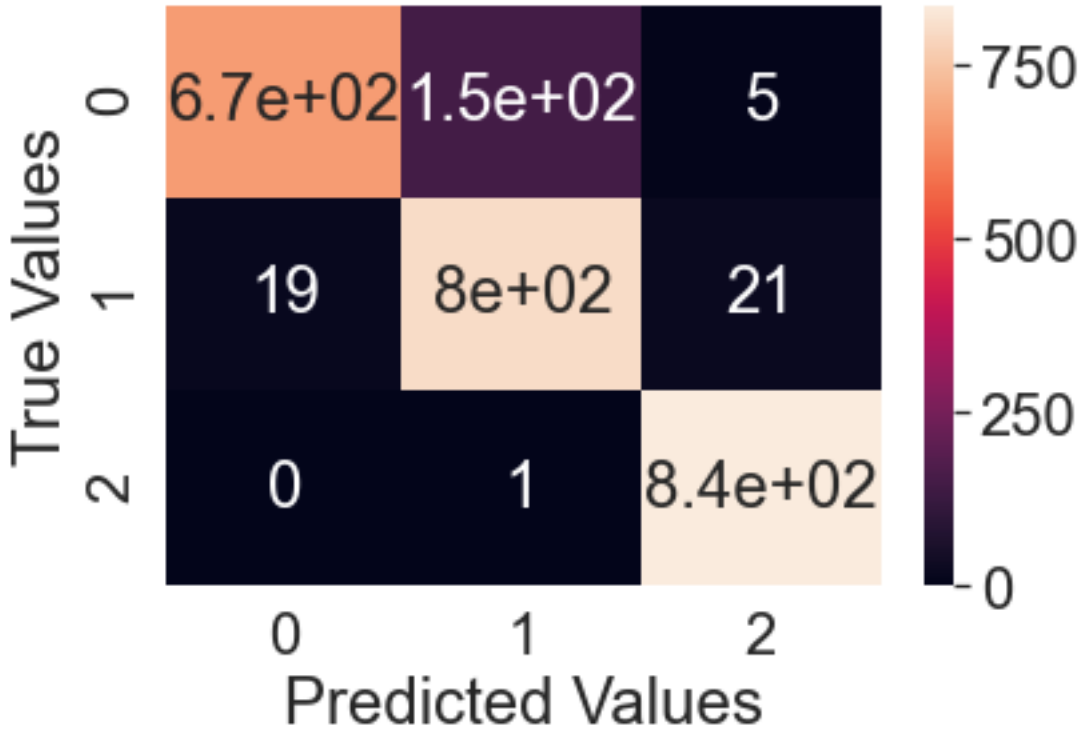


[103]:
```
y_pred=lr.predict(x_test)
print(classification_report(y_true=y_test,y_pred=y_pred))
```

```
              precision    recall  f1-score   support
```

```
              0       0.97      0.81      0.88       822
              1       0.84      0.95      0.89       841
              2       0.97      1.00      0.98       837

    accuracy                            0.92      2500
   macro avg        0.93      0.92      0.92      2500
weighted avg        0.93      0.92      0.92      2500
```

[104]:
```
sns.heatmap(confusion_matrix(y_test,y_pred),annot=True)
plt.xlabel('Predicted Values')
plt.ylabel('True Values')
plt.show()
```



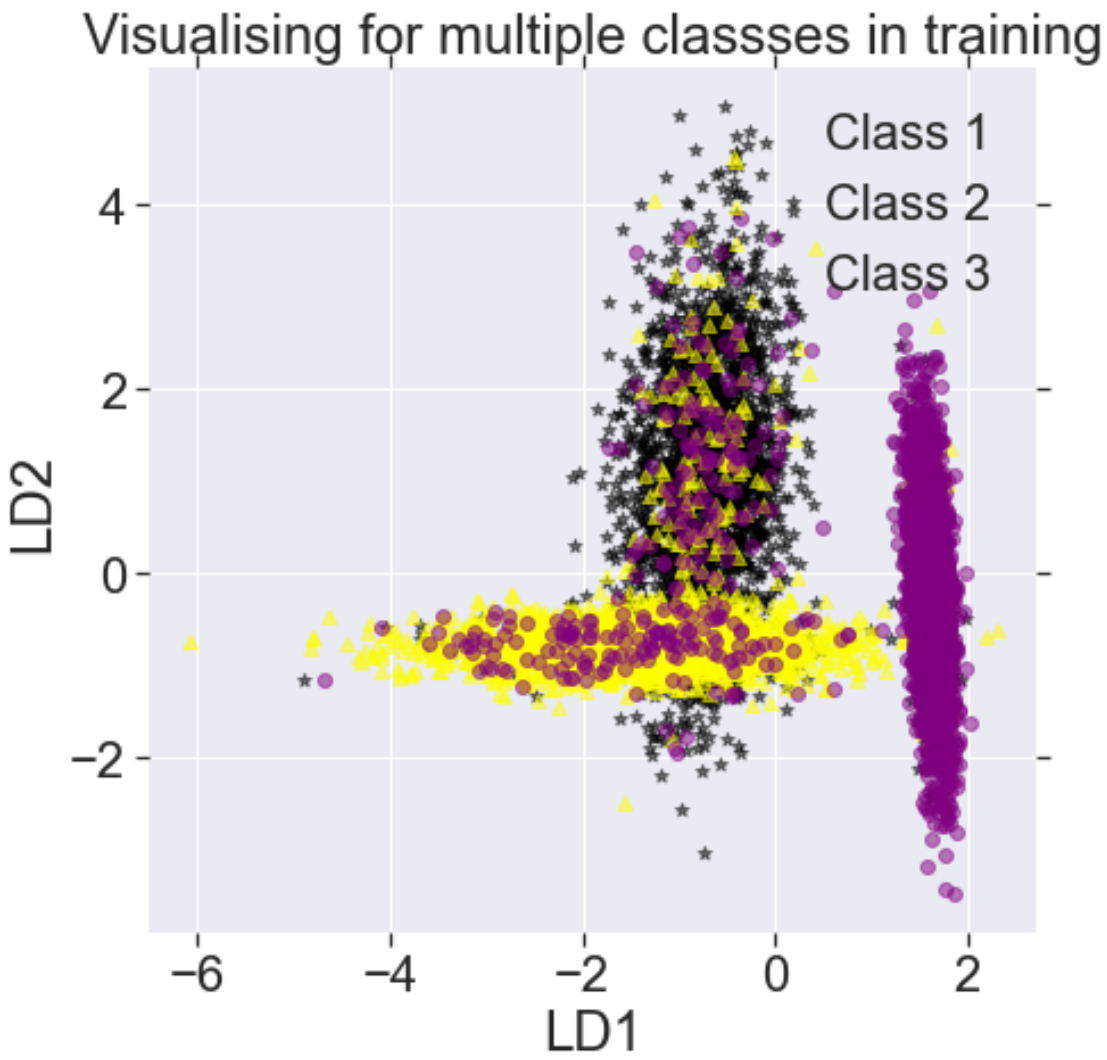
0.6 applying logestic regression on make_classification (noisy data)

[105]:
```
x,y=mc(n_samples=10000, n_features=8,n_redundant=0, n_repeated=0, n_classes=3,␣
↪n_clusters_per_class=1, random_state=17, weights=[0.3,0.3,0.4],flip_y=0.2)
```

[106]:
```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis as LDA
lda1=LDA(n_components=2)
x_lda=lda1.fit_transform(x,y)
```

[107]:
```
lr=LR()
x_train,x_test,y_train,y_test=train_test_split(x_lda,y,random_state=0)
l=lr.fit(x_train,y_train)
coef=l.coef_
intercept=l.intercept_
```

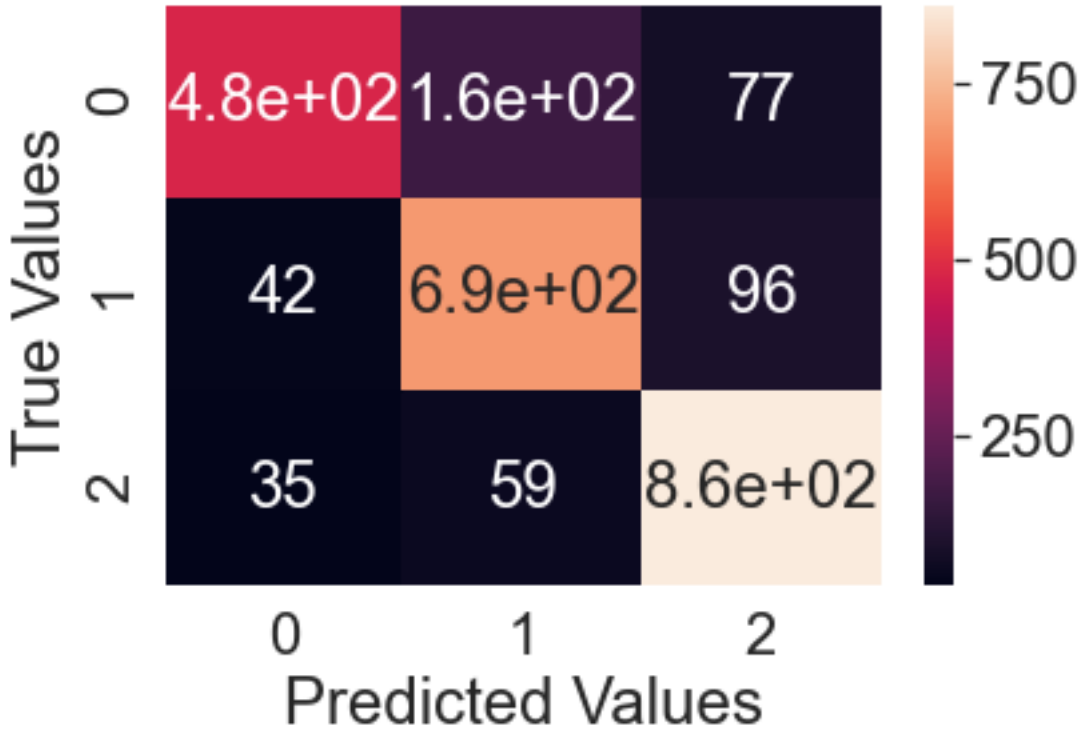
[108]:
```
label_dict = {0:'Class 1', 1: 'Class 2', 2: 'Class 3', 3:'Class 4'}
plt.figure(figsize=(7,7))

def plot_scikit_lda(X,y, title):
      ax = plt.subplot(111)
      for label,marker,color in zip(
          [i for i in range(4)],('*','^','o'),('black','yellow','purple')):

          plt.scatter(x=X[:,0][y == label],
                      y=X[:,1][y == label],
                      marker=marker,
                      color=color,
                      alpha=0.5,
                      label=label_dict[label])

      plt.xlabel('LD1')
      plt.ylabel('LD2')

      leg = plt.legend(loc='upper right', fancybox=True)
      leg.get_frame().set_alpha(0)
      plt.title(title)

      # hide axis ticks
      plt.tick_params(axis="both", which="both", bottom="off", top="off",
              labelbottom="on", left="off", right="off", labelleft="on")

      # remove axis spines
      ax.spines["top"].set_visible(False)
      ax.spines["right"].set_visible(False)
      ax.spines["bottom"].set_visible(False)
      ax.spines["left"].set_visible(False)
      plt.tight_layout
      plt.show()

plot_scikit_lda(x_train, y_train, title='Visualising for multiple classses in␣
 ↪training')
```



[109]:
```
y_pred=lr.predict(x_test)
print(classification_report(y_true=y_test,y_pred=y_pred))
```

```
              precision    recall  f1-score   support

           0       0.86      0.66      0.75       720
           1       0.75      0.83      0.79       826
           2       0.83      0.90      0.87       954

    accuracy                           0.81      2500
   macro avg       0.82      0.80      0.80      2500
weighted avg       0.81      0.81      0.81      2500
```

[110]:
```
sns.heatmap(confusion_matrix(y_test,y_pred),annot=True)
plt.xlabel('Predicted Values')
plt.ylabel('True Values')
plt.show()
```

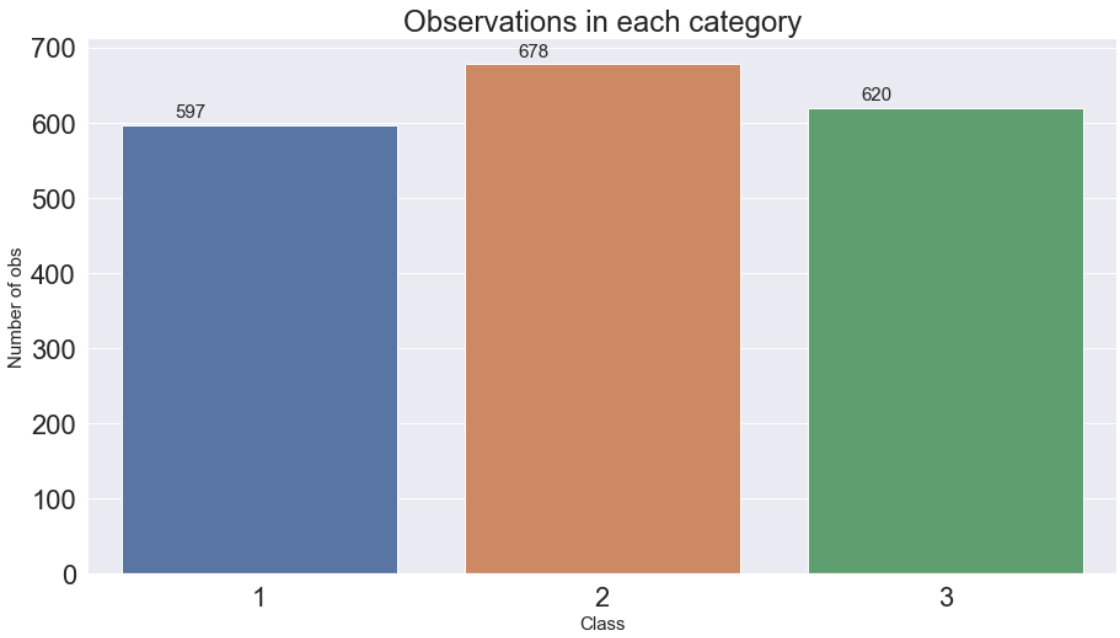


0.7 applying logestic regression on make_multilabel_classification

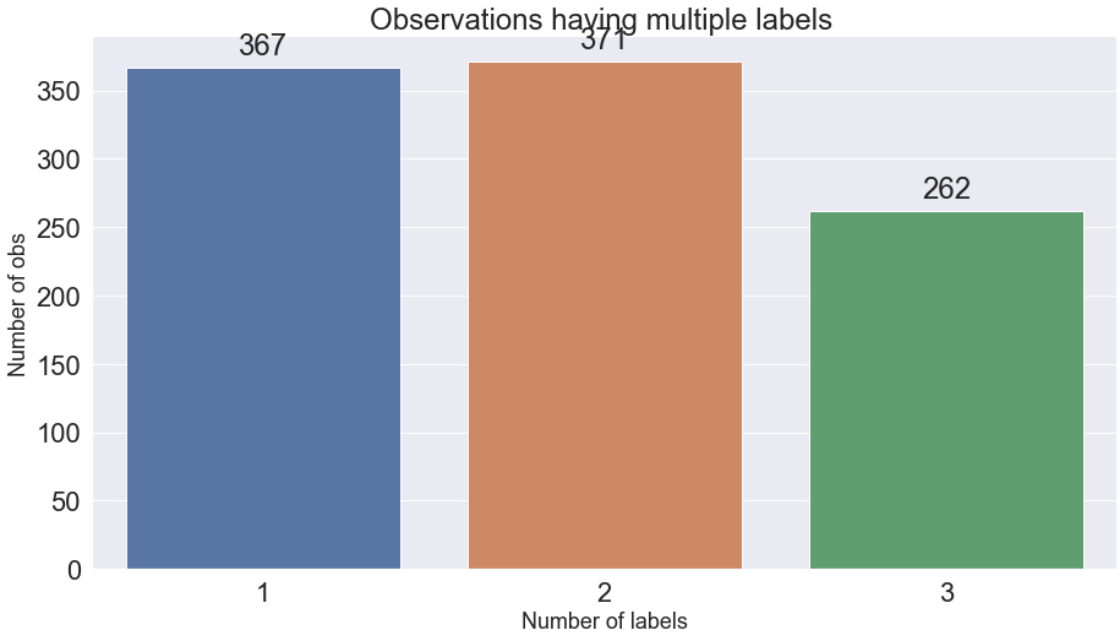
[111]:
```
from sklearn.datasets import make_multilabel_classification as mmc
X,Y=mmc(n_samples=1000,n_features=8,n_classes=3,n_labels=2,length=30,random_state=0,allow_unla
```

[112]:
```
categories = [i for i in range(1,4)]
sns.set(font_scale = 2)
plt.figure(figsize=(15,8))
ax= sns.barplot(categories, np.sum(Y[:,:],axis=0))
plt.title("Observations in each category", fontsize=24)
plt.ylabel('Number of obs', fontsize=15)
plt.xlabel('Class', fontsize=15)
#adding the text labels
rects = ax.patches
labels = np.sum(Y[:,:],axis=0)
for rect, label in zip(rects, labels):
    height = rect.get_height()
```

```
    ax.text(rect.get_x() + rect.get_width()/4, height + 5, label, ha='center',␣
 ↪va='bottom', fontsize=15)
plt.show()
```



[113]:
```
rowSums = np.sum(Y[:,:],axis=1)
multiLabel_counts = np.array([np.count_nonzero(rowSums==i) for i in range(1,4)])
sns.set(font_scale = 2)
plt.figure(figsize=(15,8))
ax = sns.barplot([i for i in range(1,4)], multiLabel_counts)
plt.title("Observations having multiple labels ")
plt.ylabel('Number of obs', fontsize=18)
plt.xlabel('Number of labels', fontsize=18)
#adding the text labels
rects = ax.patches
labels = multiLabel_counts
for rect, label in zip(rects, labels):
    height = rect.get_height()
    ax.text(rect.get_x() + rect.get_width()/2, height + 5, label, ha='center',␣
 ↪va='bottom')
plt.show()
```



[193]:
```
label_dict = {0:'Class 1', 1: 'Class 2', 2: 'Class 3'}
plt.figure(figsize=(15,15))

def plot_scikit_lda(X, title):
    for i in range(3):
      lda=LDA()
      X_lda=lda.fit_transform(X,Y[:,i])
      #print(X_lda.shape)
      ax = plt.subplot(2,2,i+1)
      for label,marker,color in zip(
          [0,1],('o','*','s'),('yellow','blue','red')):

          plt.scatter(x=X_lda[:,0][Y[:,i] == label],
                      y=np.ones(X_lda[Y[:,i] == label].shape[0]),#X[:,1][Y[:,i]␣
 ↪== label],
                      marker=marker,
                      color=color,
                      alpha=0.5,
                      label=("rest",label_dict[i])[int(label==1)])

      plt.xlabel('LD1')
      plt.ylabel('LD2')

      leg = plt.legend(loc='upper right', fancybox=True)
      leg.get_frame().set_alpha(0)
      plt.title(title+label_dict[i])
```

```
    # hide axis ticks
    plt.tick_params(axis="both", which="both", bottom="off", top="off",
            labelbottom="on", left="off", right="off", labelleft="on")

    # remove axis spines
    ax.spines["top"].set_visible(False)
    ax.spines["right"].set_visible(False)
    ax.spines["bottom"].set_visible(False)
    ax.spines["left"].set_visible(False)
    plt.tight_layout(pad=3.0)
  plt.show()

plot_scikit_lda(X, title='Visualising for ')
```

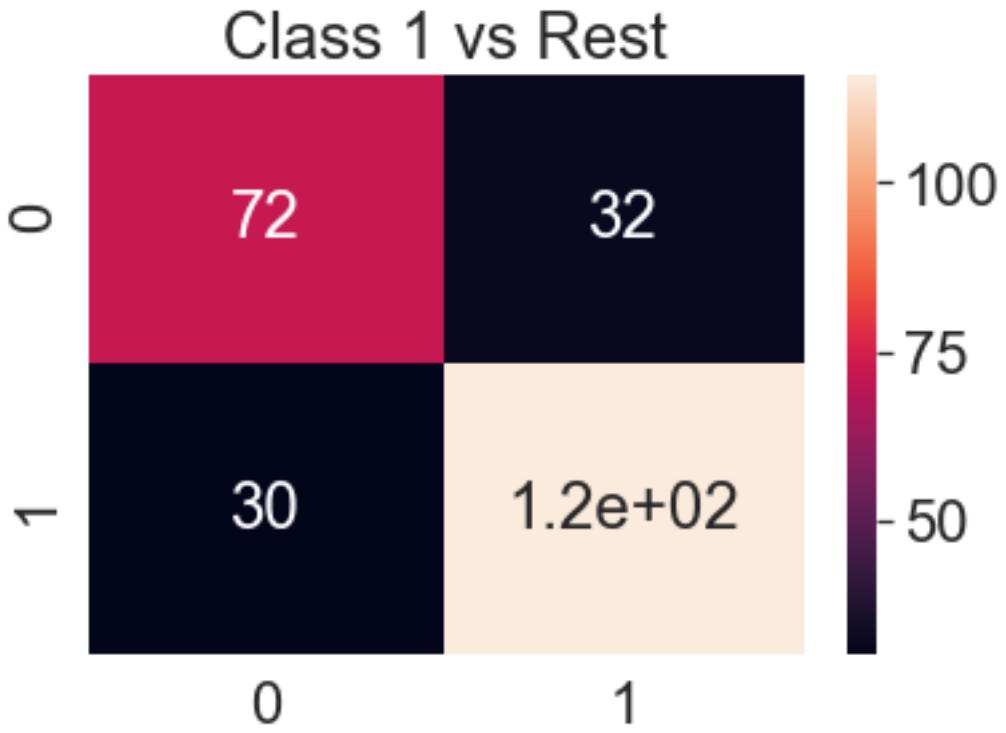
```
    ␣
↪---------------------------------------------------------------------------

     ValueError                                Traceback (most recent call␣
↪last)

     <ipython-input-193-e569071f7855> in <module>
      37     plt.show()
      38
---> 39 plot_scikit_lda(X, title='Visualising for ')


     <ipython-input-193-e569071f7855> in plot_scikit_lda(X, title)
       5     for i in range(3):
       6       lda=LDA()
----> 7       X_lda=lda.fit_transform(X,Y[:,i])
       8       #print(X_lda.shape)
       9       ax = plt.subplot(2,2,i+1)


     ~\Anaconda3\lib\site-packages\sklearn\base.py in fit_transform(self, X,␣
↪y, **fit_params)
     691         else:
     692             # fit method of arity 2 (supervised transformation)
--> 693             return self.fit(X, y, **fit_params).transform(X)
     694
     695


     ~\Anaconda3\lib\site-packages\sklearn\discriminant_analysis.py in␣
↪fit(self, X, y)
```

```
       423         """
       424         X, y = self._validate_data(X, y, ensure_min_samples=2,␣
↪estimator=self,
   --> 425                                    dtype=[np.float64, np.float32])
       426         self.classes_ = unique_labels(y)
       427         n_samples, _ = X.shape


       ~\Anaconda3\lib\site-packages\sklearn\base.py in _validate_data(self, X,␣
↪y, reset, validate_separately, **check_params)
       430                 y = check_array(y, **check_y_params)
       431             else:
   --> 432                 X, y = check_X_y(X, y, **check_params)
       433             out = X, y
       434


       ~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in␣
↪inner_f(*args, **kwargs)
        71                             FutureWarning)
        72         kwargs.update({k: arg for k, arg in zip(sig.parameters,␣
↪args)})
   ---> 73         return f(**kwargs)
        74     return inner_f
        75


       ~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in␣
↪check_X_y(X, y, accept_sparse, accept_large_sparse, dtype, order, copy,␣
↪force_all_finite, ensure_2d, allow_nd, multi_output, ensure_min_samples,␣
↪ensure_min_features, y_numeric, estimator)
       811         y = y.astype(np.float64)
       812
   --> 813     check_consistent_length(X, y)
       814
       815     return X, y


       ~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in␣
↪check_consistent_length(*arrays)
       255     if len(uniques) > 1:
       256         raise ValueError("Found input variables with inconsistent␣
↪numbers of"
   --> 257                          " samples: %r" % [int(l) for l in lengths])
       258
       259
```

```
        ValueError: Found input variables with inconsistent numbers of samples:␣
↪[712, 1000]
```

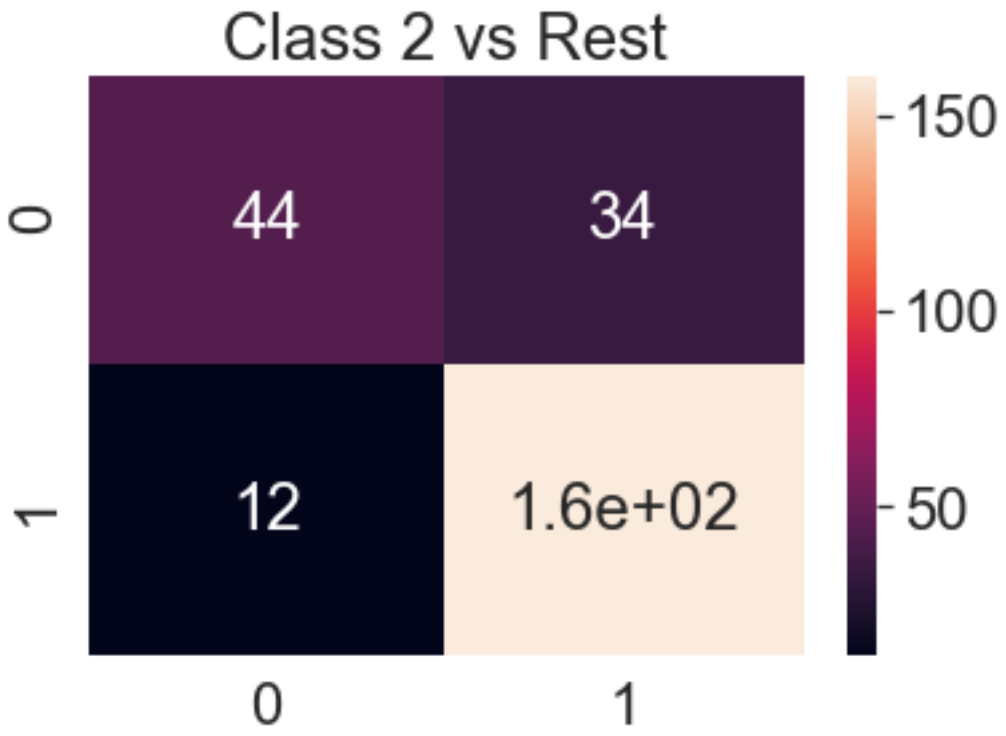
```
<Figure size 1080x1080 with 0 Axes>
```

[115]:
```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,random_state=0)
pred=np.ndarray(Y_test.shape)
for i in range(Y_test.shape[1]):
  lr.fit(X_train,Y_train[:,i])
  pred[:,i]=lr.predict(X_test)
  sns.heatmap(confusion_matrix(Y_test[:,i],pred[:,i]),annot=True)
  plt.title("Class "+str(i+1)+" vs Rest")
  plt.show()
  print(classification_report(Y_test[:,i],pred[:,i]))
```

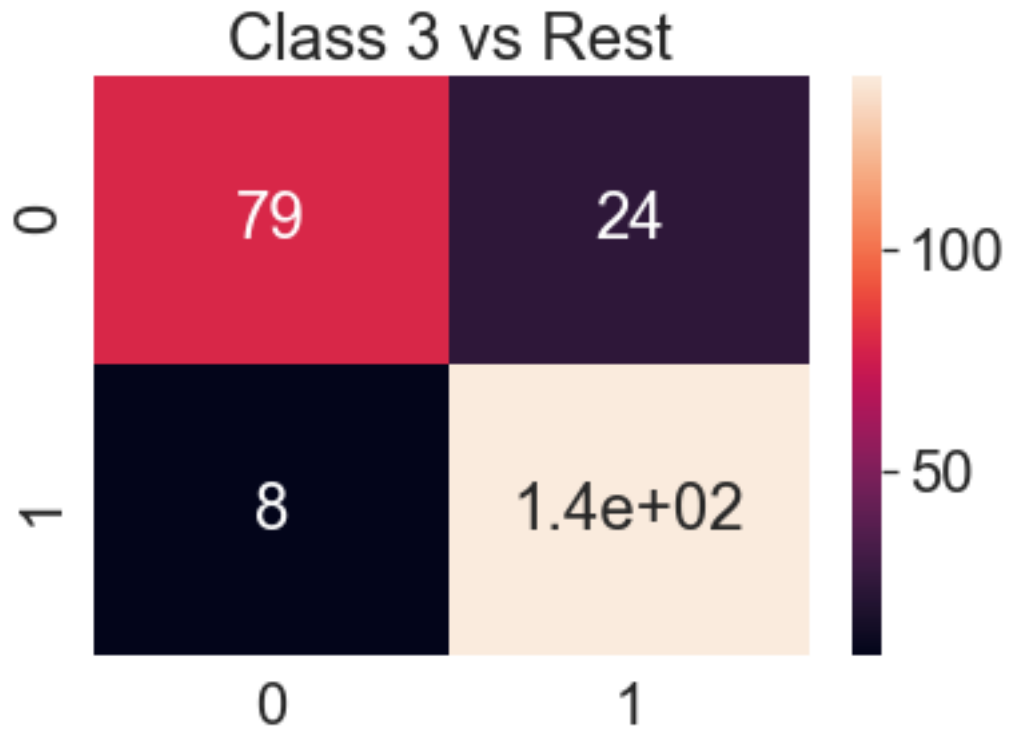



```
              precision    recall  f1-score   support

           0       0.71      0.69      0.70       104
           1       0.78      0.79      0.79       146

    accuracy                           0.75       250
```

macro avg	0.74	0.74	0.74	250
weighted avg	0.75	0.75	0.75	250



	precision	recall	f1-score	support
0	0.79	0.56	0.66	78
1	0.82	0.93	0.87	172
accuracy			0.82	250
macro avg	0.81	0.75	0.77	250
weighted avg	0.81	0.82	0.81	250



	precision	recall	f1-score	support
0	0.91	0.77	0.83	103
1	0.85	0.95	0.90	147
accuracy			0.87	250
macro avg	0.88	0.86	0.86	250
weighted avg	0.88	0.87	0.87	250

[ ]:

[ ]: