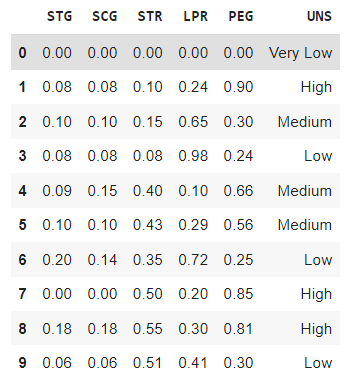
**ELG5255 Applied Machine Learning**

REPORT of: Group Assignment 1 (Group-18)

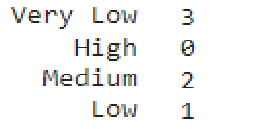
**Problem 1:**

1. **load and understand the dataset**
2. DUMD It is the real dataset about the students' knowledge status about the subject of Electrical DC Machines. The data is cleaned and divided into training and testing. the data belong to classes (0,1,2,3) ((Very Low, High, Medium, and Low)).
3. We loaded the DUMD dataset.





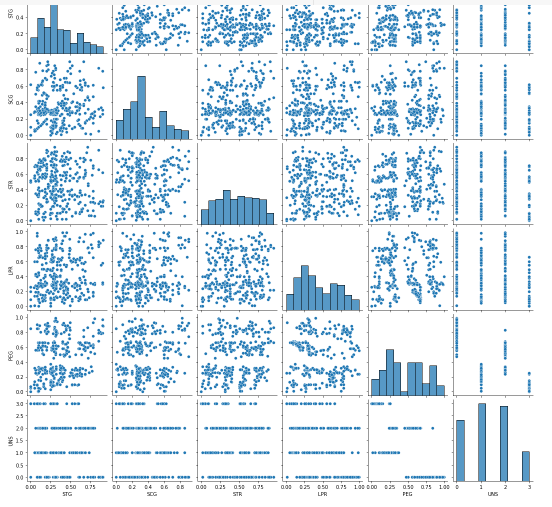
1. **Label Encoding**
2. We converted categorical class labels (UNS) to numerical values by using the LabelEncoder.

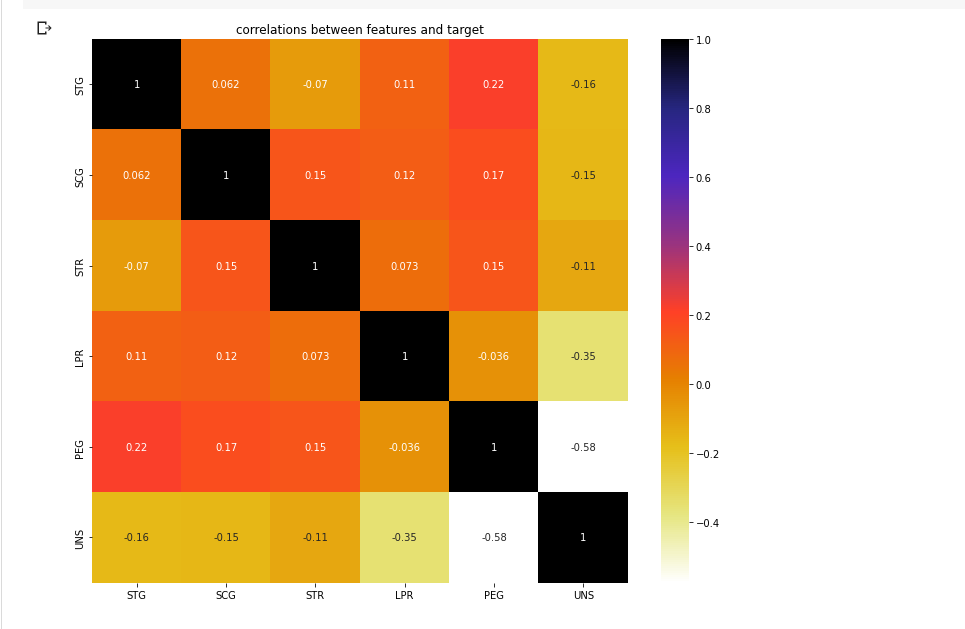


1. **Features selection**
2. we used a sklearn function to choose the best 2 features using the correlation between features and the target and the result was

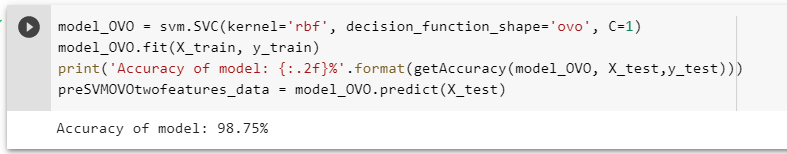
[LPR, PEG] features.

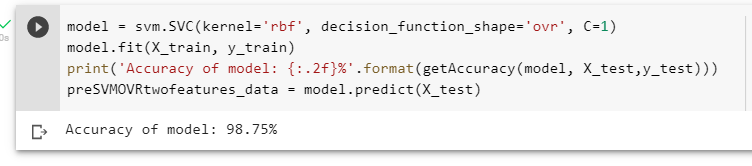
**plot pairwise relationships in a dataset**

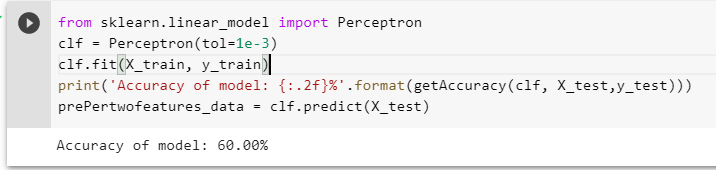


****

1. **build OVO-SVM, OVR-SVM classifiers, and perceptron classifiers**
2. we used python libraries to build three classifiers:
   * + 1. **OVO-SVM 2. OVR-SVM 3. perceptron classifier**



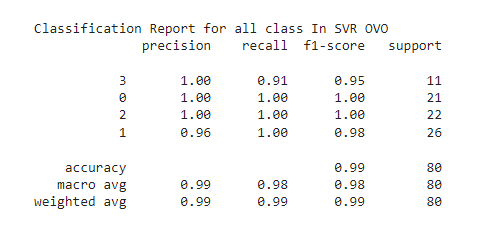


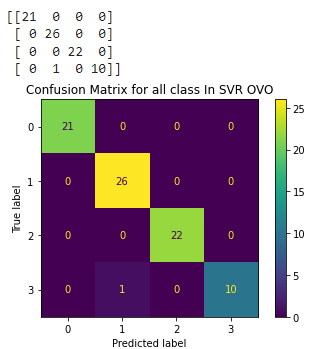


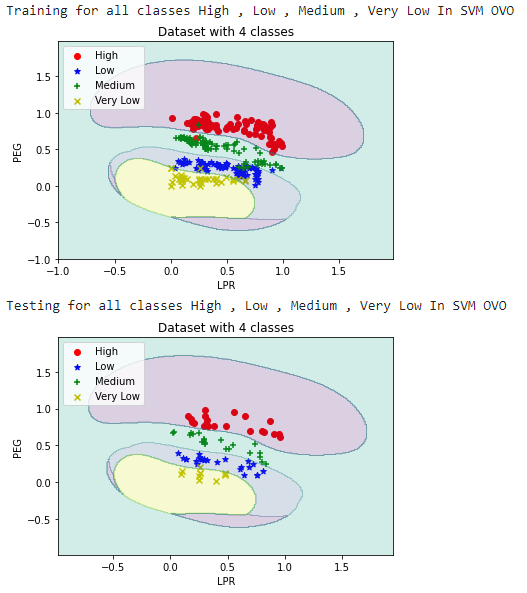
1. We built functions to get the best accuracy, confusion matrix, decision boundaries, and plot correct and wrong prediction points.

**OVO-SVM using sklearn Confusion Matrix and decision boundary:**

**The accuracy of the model: 98.75%**

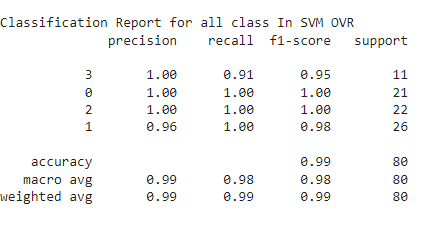


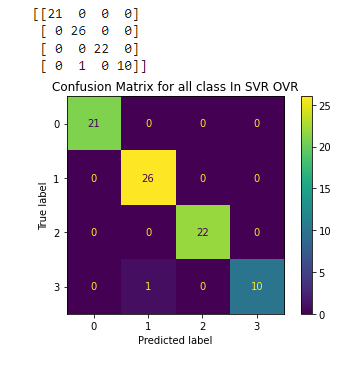


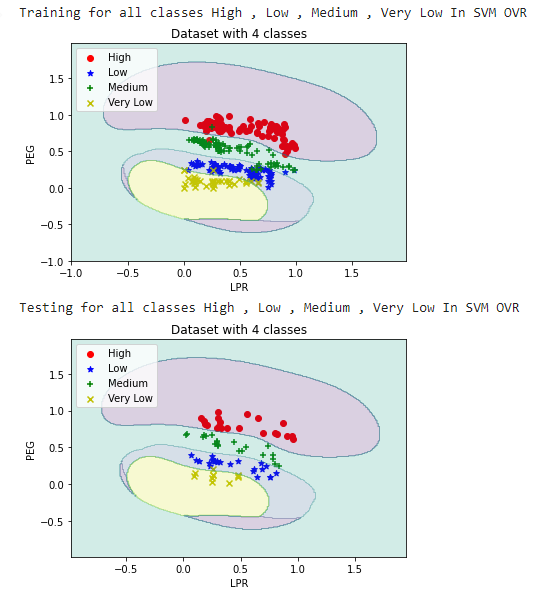


**OVR-SVM using sklearn Confusion Matrix and decision boundary:**

**The accuracy of the model: 98.75%**

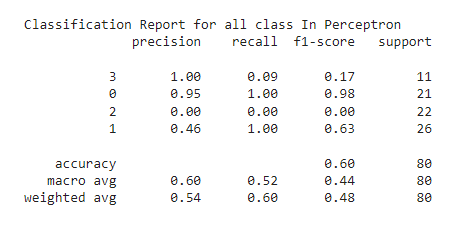


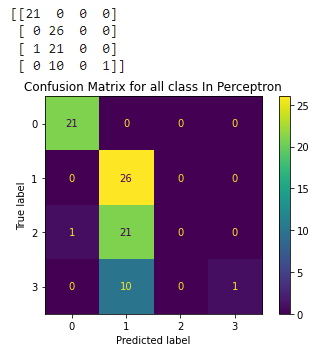


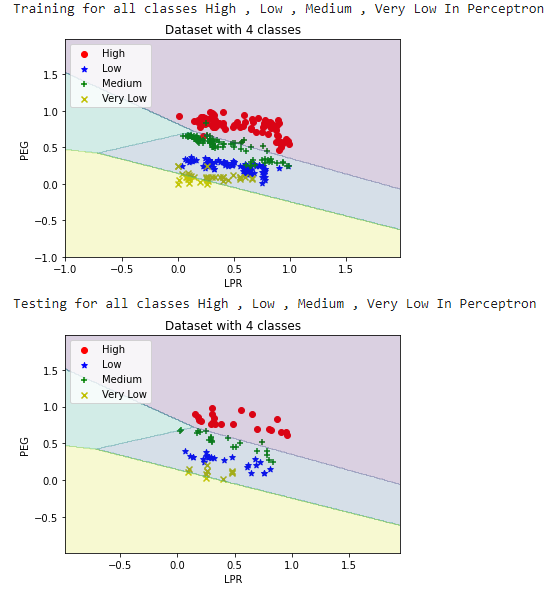


**Perceptron using sklearn Confusion Matrix and decision boundary:**

**The accuracy of the model: 60%**



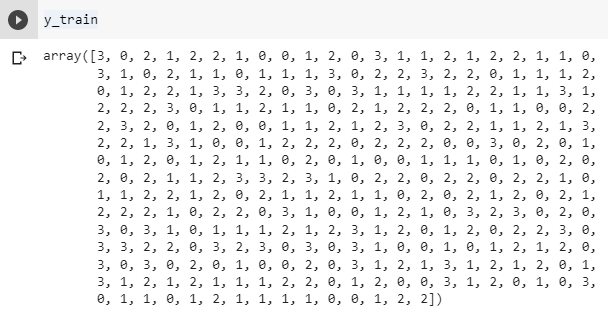




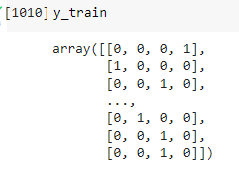
**Problem 2:** Build OVR-SVM

**2.1 obtain the binarized labels**

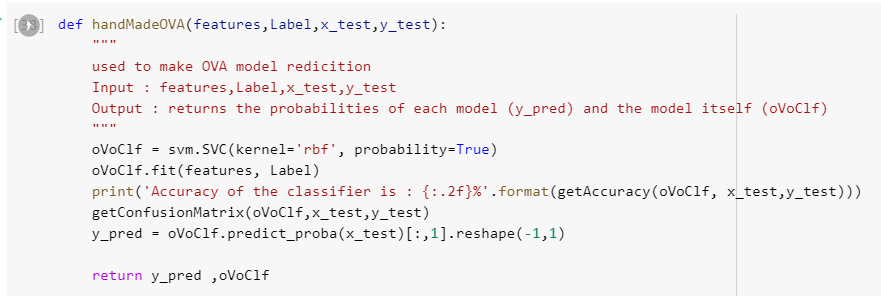
**Before binarization**

****

**After Binarization**

****

**2.2 build the OVR (OVA) and obtain the Accuracy**



**The accuracy**

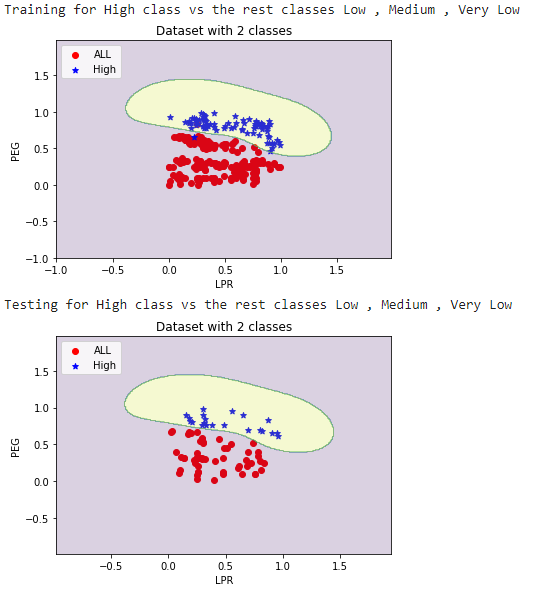
**(All, High)** (100.00%), **(All, Low)** (97.50%), **(All, Median)** (100.00%), **(All, Very Low)** (98.75%).

|  |  |
| --- | --- |
|  |  |
|  |  |

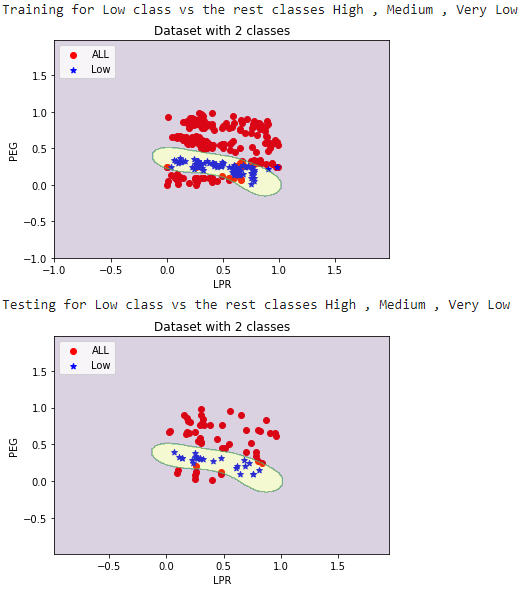
**2.3 plots and Boundaries** for both training and testing data

**Decision Boundary**: (All, High), (All, Low), (All, Median), (All, Very Low)

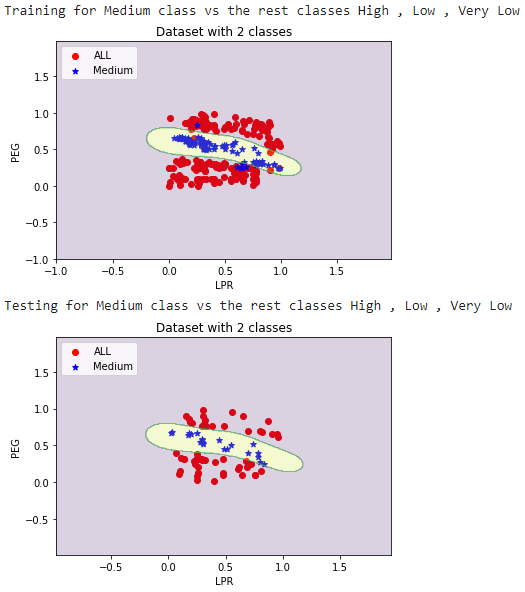
**(All, High)**



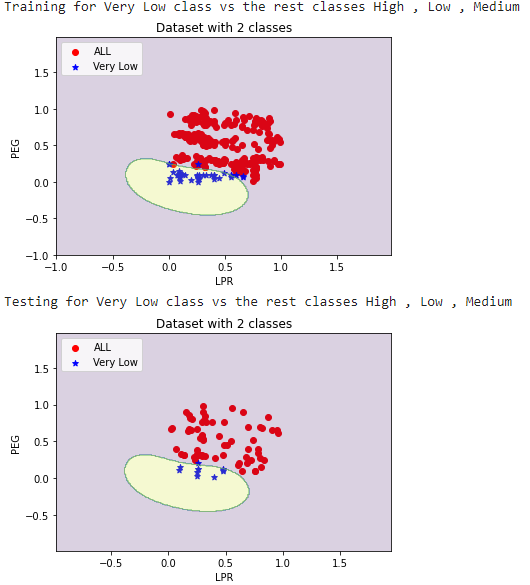
**(All, Low)**



**(All, Median)**



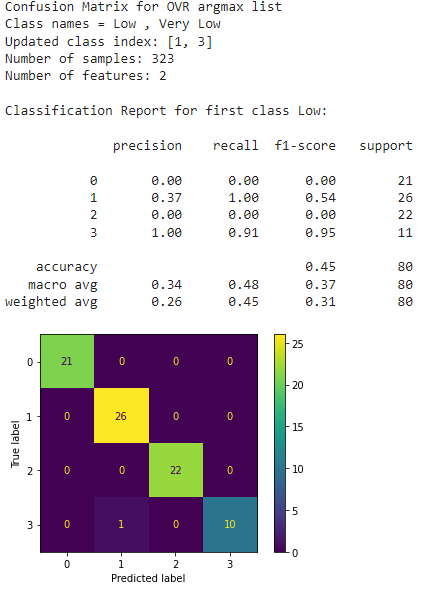
**(All, Very Low)**



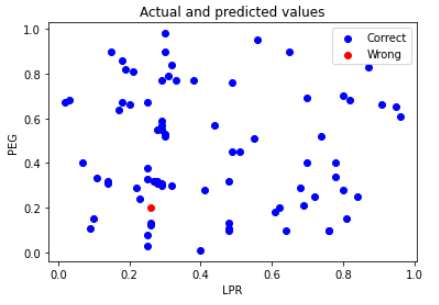
### (2.B) using argmax to obtain the final predicted labels

### 

### 



### 

****

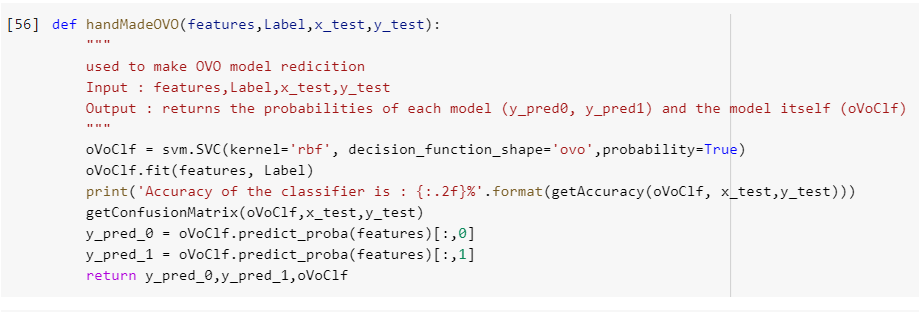
**Accuracy comment: the accuracy of the model is nice, maybe because we have a small number of data.**

**Problem 3: Build OVO-SVM:**

**3.1 obtain the binarized labels:** already explained in problem 2

**3.2 build the OVR (OVA) and obtain the Accuracy**

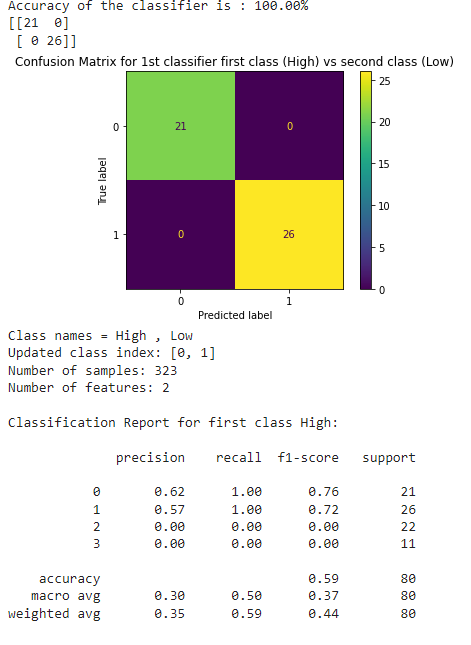
We built 6 model because the target contains 4 classes so **(n\*(n-1)/2 = 6)**



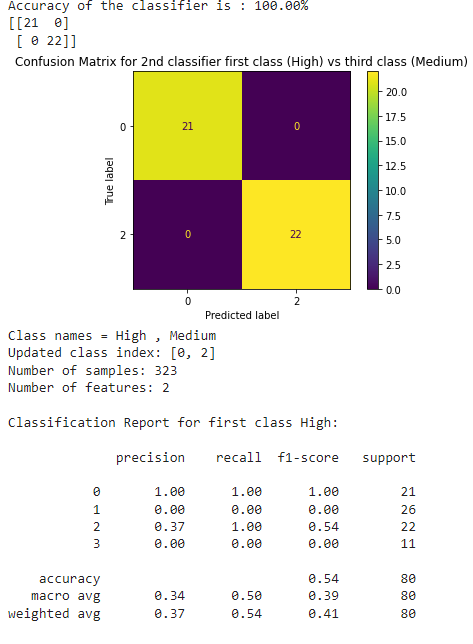
**The accuracy**: **(0, 1)** is (100%), **(0, 2)** is (100%) , **(0,3)** is (100%), **(1, 2)** is (100%), **(1, 3)**  is (97.30%) and **(2,3)** is (100%)

**Confusion Matrix**: (0, 1), (0, 2), (0,3), (1, 2), (1, 3) and (2, 3)

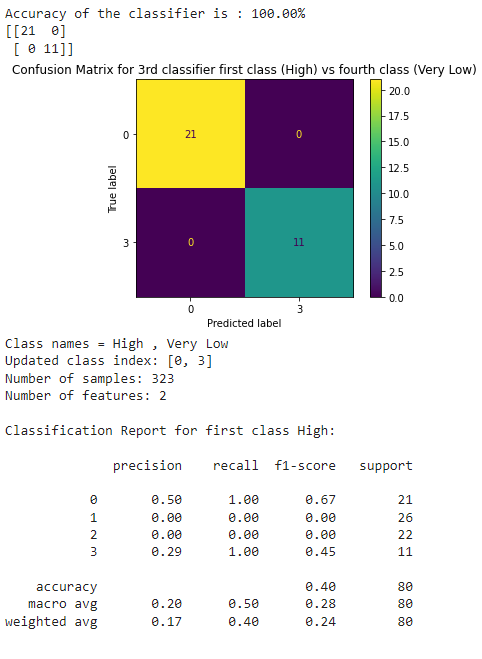
**(HIGH, LOW)**



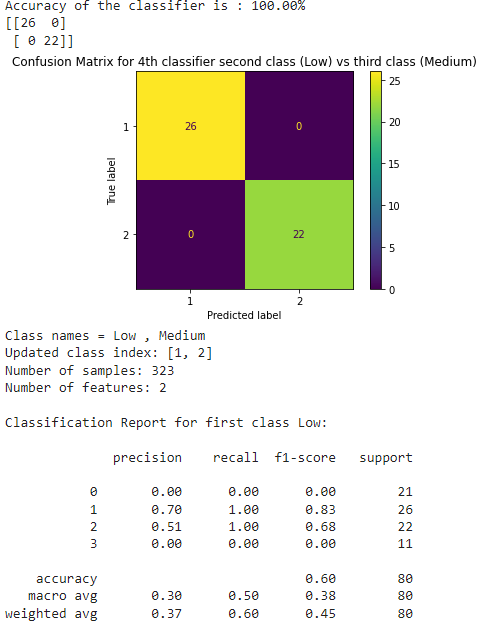
**(HIGH, MEDIUM)**



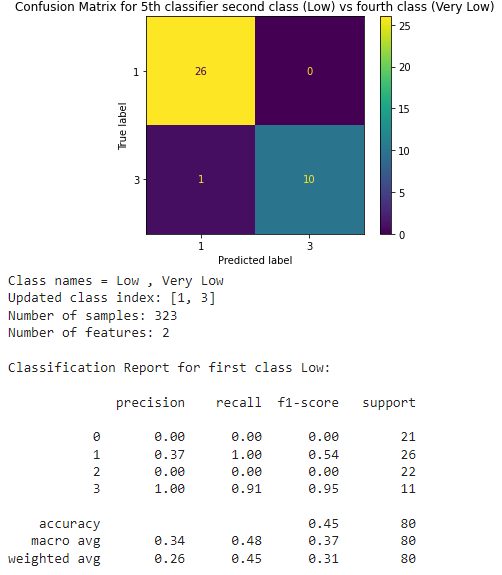
**(HIGH, VERY LOW)**



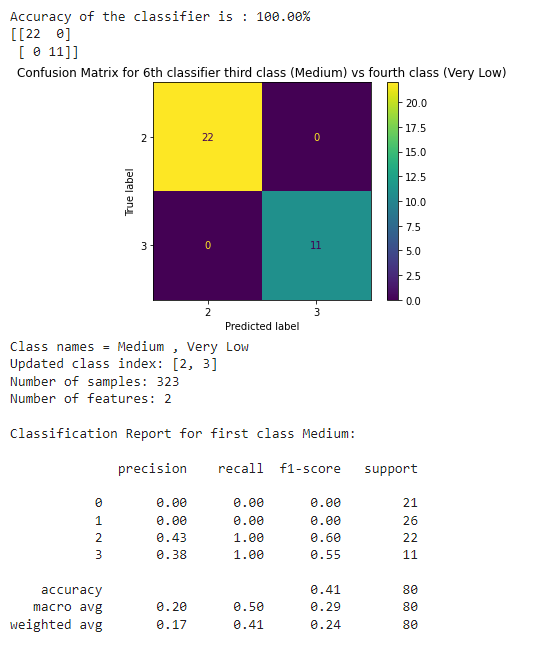
**(LOW, MEDIUM)**



**(LOW, VERY LOW)**

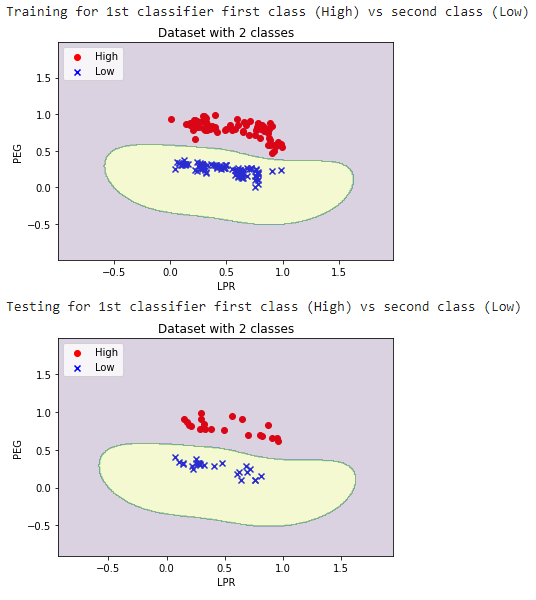


**(MEDIUM, VERY LOW)**

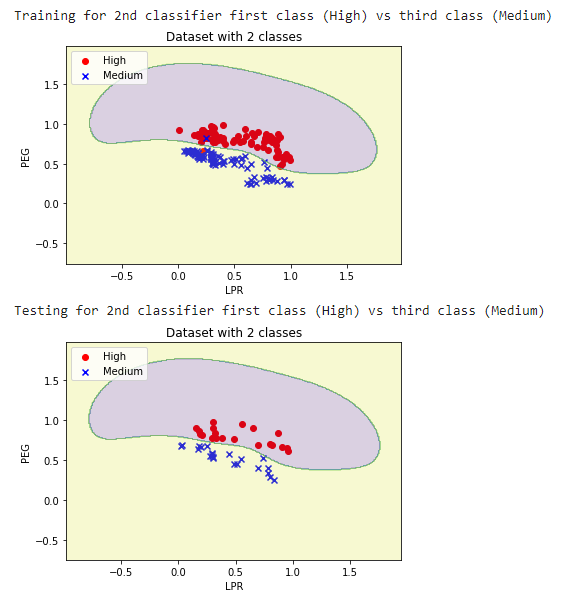


**3.3 plots and Boundaries** for both training and testing data

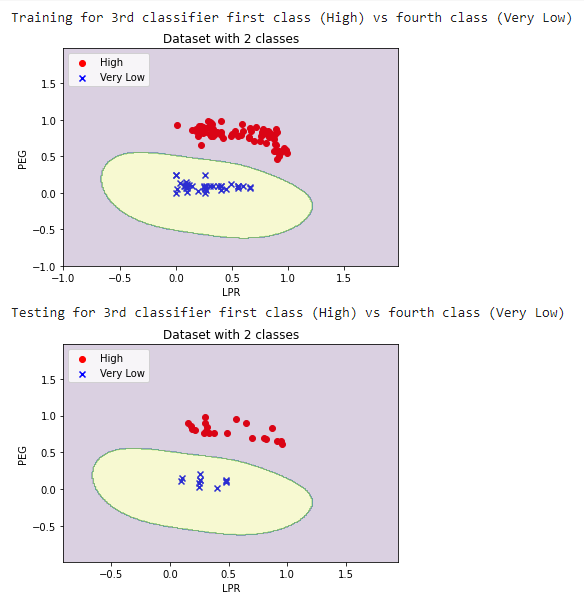
**(HIGH, LOW)**



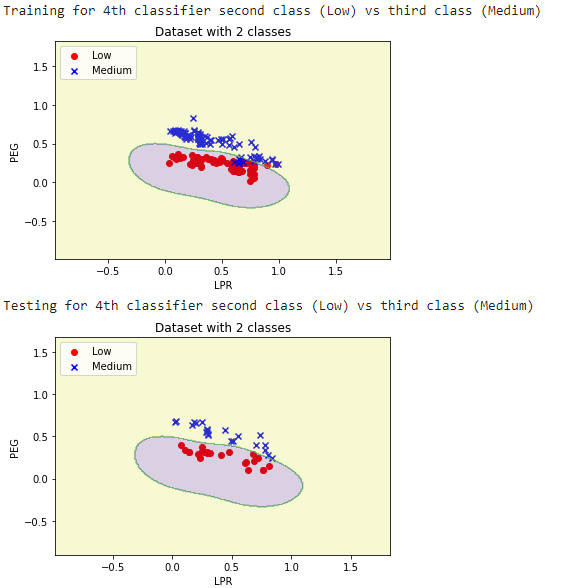
**(HIGH, MEDIUM)**



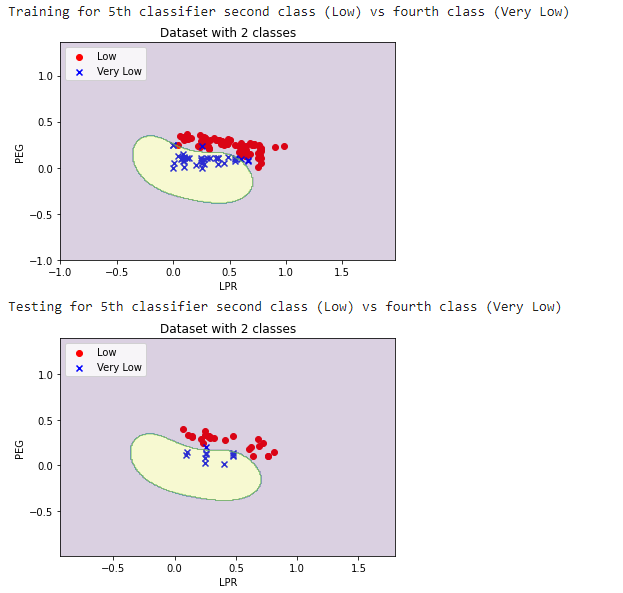
**(HIGH, VERY LOW)**



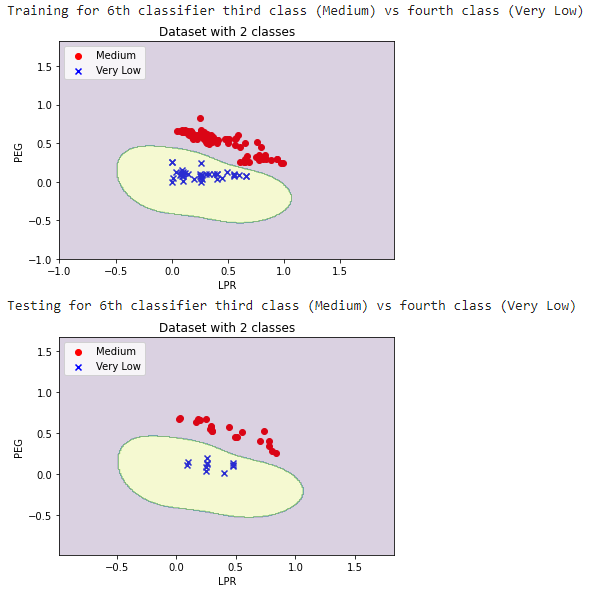
**(LOW, MEDIUM)**



**(LOW, VERY LOW)**



**(MEDIUM, VERY LOW)**



### 3.3 Use argmax to aggregate confidence scores and obtain the final label and obtain the performance

**comment:** because we are using one versus one technique that made us face a problem which is how we obtain the opinion of each model in a specific sample. that is because each model will say for example it’s 0 or 1, but each time 0 and 1 have a different meaning depending on the model itself, so we took two approaches to solve this problem and get all the probability of a sample to belong to specific class from all classifiers:

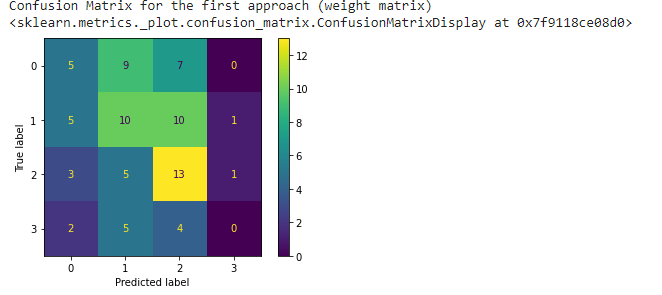
**The first approach**: we built a weight matrix, and the mission was to fill this matrix with the summation of the probability of each class, that’s why we called it a weight matrix because the values not located between 0 and 1 like probabilities

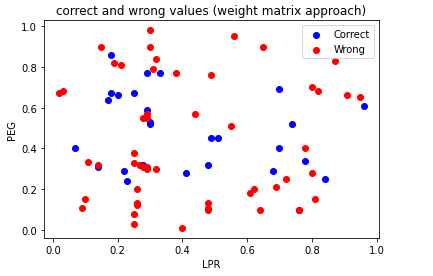
**The second approach**: we built a voting matrix, and the mission was to fill this matrix with the summation of the raised hands “how many classifiers said this sample belongs to that class”

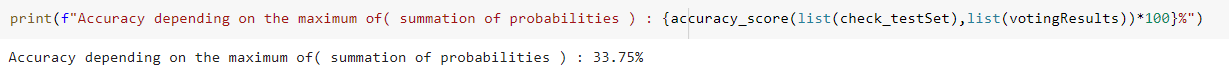
**the accuracy of the first approach was so bad, so we used the second approach.**

**the first approach**

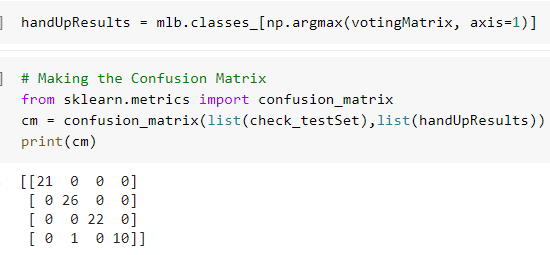
### 

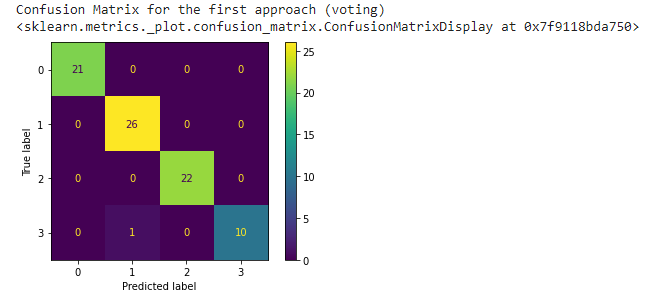


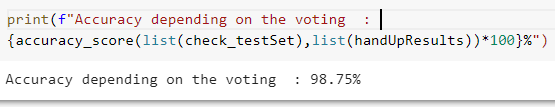


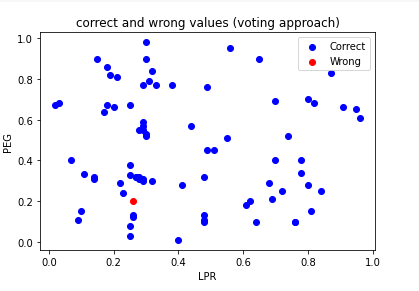
****

**the second approach**

****







**Conclusion:**

we loaded the dataset and understood it carefully, we checked the correlation between the features and the label and we selected two features with the highest correlation ratio with the target.

in problem 1 we built three models and trained them with the selected features, some models gave good predictions such as SVM and others gave fewer predictions such as Perceptron. then we used the MultiLabelBinarizer library to binarize the encoded target then we built an OVR-SVM and OVO-SVM (in problems 2 and 3).

we learned how to use OVO-SVM and OVR-SVM, and we also learned the difference between both of them and the mechanism that each one of them uses to predict the class of a sample.

some of the problems that we faced while working on this assignment:

1. in problem 2 some of the decision boundaries were a bit weird but when we used the kernel trick

(“rbf” kernel) it got much better. So, we tried to use the kernel trick also in problem 1 and we got a higher accuracy than the accuracy with the linear kernel

2. in problem 3 the weight method to collect the probabilities of each sample to be in a specific class the method didn’t give us a good accuracy so we used the second approach which was to get the voting results.