Student ID:210033137
Module: CS5014 Machine Learning

Part 1: Data Processing

A) How did you load and clean the data, and why?

Firstly, I load the dataset and add column names for all the features ('ID' 'Age' etc). Then extract the columns needed for model prediction i.e. (personality attributes 2-13 and Nicotine) as these are the ones required for our model.

Then I explore the data set as this tells us information about the checking shape, unique values, and general information of dataset, null values, target distribution etc.

B) How did you split the data into test/train set and why?

I split the data into 80% for training and 20% for testing as this is common practise and works well with large datasets. The reason we split the data set is to ensure we prevent both underfitting and overfitting. As a good accuracy level in the training dataset doesn't equate to success of your model on unseen data. Hence testing set should be unseen data.

C) How did you process the data including encoding, conversion, and scaling, and why?

In terms of the encoding after we explore the dataset, we one hot encode all the categorical variables i.e. (Gender, Education, Country, Ethnicity, Impulsive, SS and concatenate them into a new data frame).

In terms of the conversion, we create a function that returns the input features and output feature for model

In terms of the scaling aspect, we normalise the x train and x test to ensure it is 0 &1s.

D) How did you ensure that there is no data leakage?

In terms of preventing data leakage, we must ensure there are no duplicates, and the data set is cleaned properly. We ensure we then split the dataset into Training, validation and test groups and we normalise after splitting but before cross validation.

Part 2: Training

A) Train using penalty='none' and class weight=None. What is the best and worst classification accuracy you can expect and why?

When set at penalty none the accuracy is 36% we can expect higher accuracy levels as this could be biased as CL0 and CL6 are very high values and the others are all very low

B) Explain each of the following parameters used by LogisticRegression in your own words: penalty, tol, max iter.

Penalty= This is a way to avoid overfitting. As it reduces parameters and simplifies the model. Thus is a form of regularisation.

Tol= Is the tolerance for the stopping criteria. This tells scikit to stop searching for a minimum (or maximum) once some tolerance is achieved, i.e., once you're close enough. Tol will change depending on the objective function being minimized and the algorithm they use to find the minimum, and thus will depend on the model you are fitting

Max iter= This stands for the maximum number of iterations taken for the solvers to converge.

C) Train using balanced class weights (setting class weight='balanced'). What does this do and why is it useful?

   To balance the classes, we can inform the estimator to adjust how it calculates the loss. Using weights, we can enforce an estimator to learn based on greater or less importance (weight) given to a particular class given there often is an imbalance.

   Weights scale the loss function. Therefore, as the model trains on each point, the error will be multiplied by the weight of the point. The estimator will try to minimise error on the more heavily weighted classes, because they will have a larger effect on error, sending a stronger signal. Without weights set, the model treats each point as equally important. Hence this is a crucial step.

D) LogisticRegression provides three functions to obtain classification results. Explain the relationship between the vectors returned by predict(), decision function(), and predict proba().

Predict()= Predict class labels for samples. This is the final label which constitutes a 1 or 0. (scikit-learn.org)

Predict_proba()= Used to predict the class probability estimates. The greater the probability the greater the confidence score. Hence, the further you are away from the sample point from the hyperplane the higher the confidence.

Decision function()= Predict confidence scores for samples. k. This method basically returns a Numpy array. Whereby each element represents whether a predicted sample for x_test by the classifier lies to the right or left side of the Hyperplane and also how far from the HyperPlane.

Also, tells us that how confidently each value predicted for x_test by the classifier is Positive (large-magnitude Positive value) or Negative (large-magnitude Negative value). (geeksforgeeks.org).

Student ID:210033137
Module: CS5014 Machine Learning

Part 3: Evaluation

A) What is the classification accuracy of your model and how is it calculated? Give the formula.

The classification accuracy of my model for training and testing for unbalanced is: 36%

$$\text{Classification Accuracy} = \frac{C}{N}$$

Where C is = Number of correct predictions

Where N = Total number of predictions

This is calculated through the SkiKit-Learn function accuracy_score.

B) What is the balanced accuracy of your model and how is it calculated? Give the formula

$$\text{Accuracy} = \frac{1}{M}\sum_{m=1}^{M}\frac{Cm}{Nm}$$

Where M= Number of classes

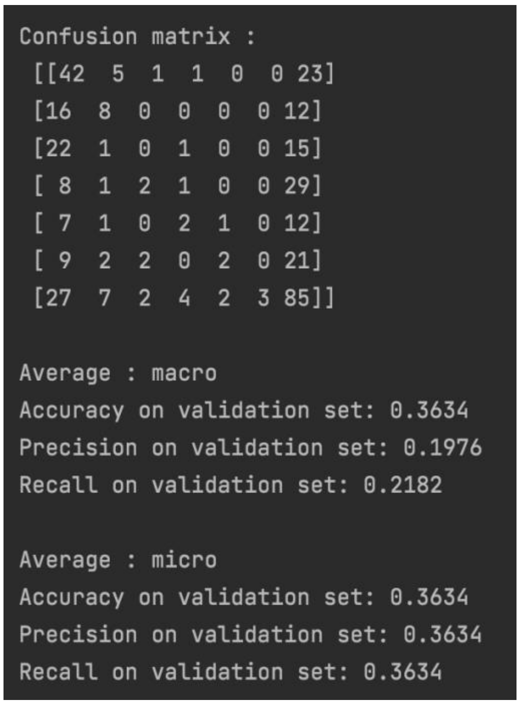
Where Cm= Number of correctly predicted labels belonging to class m

Where Nm= The total number of labels belonging to class m

The balanced accuracy of my model for training and testing for balanced is: 23%

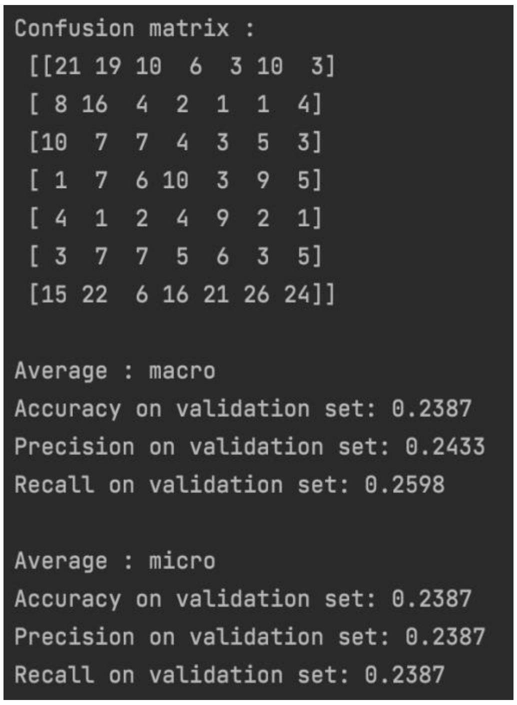
Student ID:210033137
Module: CS5014 Machine Learning

C) Show the confusion matrix for your classifier for both unbalanced (2a) and balanced (2c) cases. Discuss any differences.



```
Confusion matrix :
 [[42  5  1  1  0  0 23]
 [16  8  0  0  0  0 12]
 [22  1  0  1  0  0 15]
 [ 8  1  2  1  0  0 29]
 [ 7  1  0  2  1  0 12]
 [ 9  2  2  0  2  0 21]
 [27  7  2  4  2  3 85]]

Average : macro
Accuracy on validation set: 0.3634
Precision on validation set: 0.1976
Recall on validation set: 0.2182

Average : micro
Accuracy on validation set: 0.3634
Precision on validation set: 0.3634
Recall on validation set: 0.3634
```

Figure 1- Unbalanced Confusion Matrix

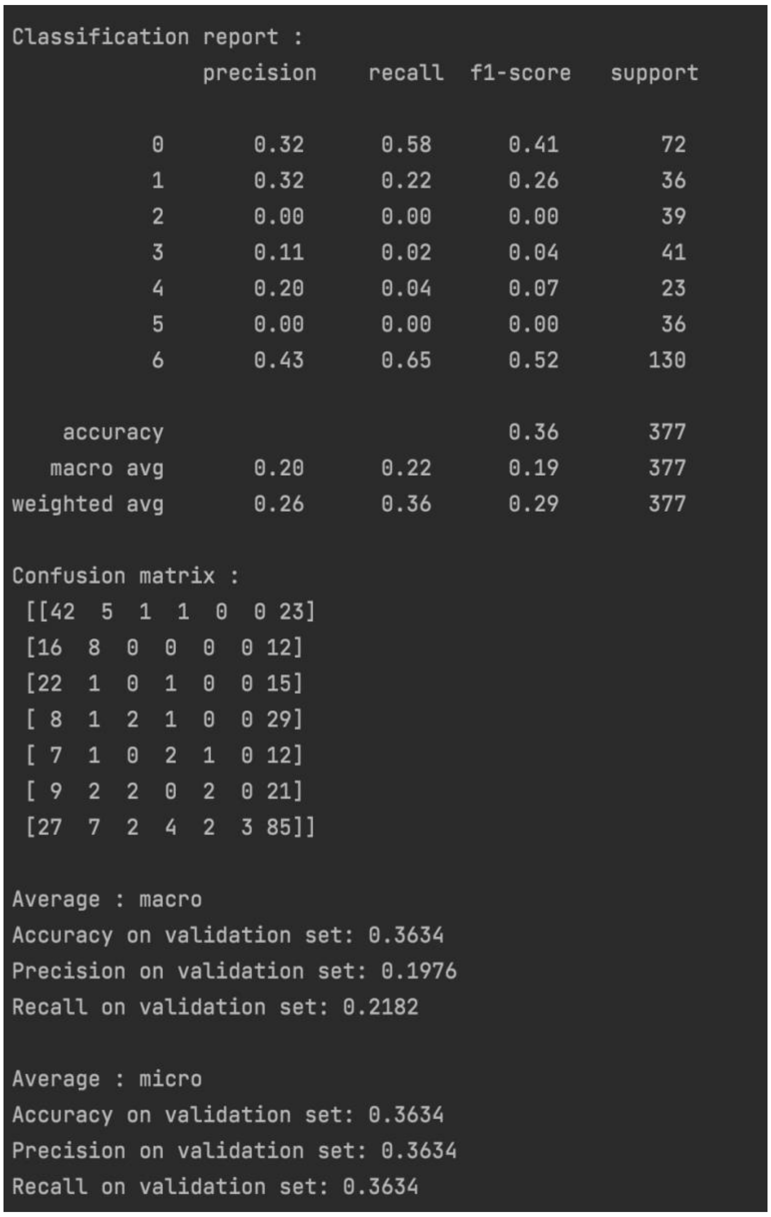


```
Confusion matrix :
 [[21 19 10  6  3 10  3]
 [ 8 16  4  2  1  1  4]
 [10  7  7  4  3  5  3]
 [ 1  7  6 10  3  9  5]
 [ 4  1  2  4  9  2  1]
 [ 3  7  7  5  6  3  5]
 [15 22  6 16 21 26 24]]

Average : macro
Accuracy on validation set: 0.2387
Precision on validation set: 0.2433
Recall on validation set: 0.2598

Average : micro
Accuracy on validation set: 0.2387
Precision on validation set: 0.2387
Recall on validation set: 0.2387
```

Figure 2- Balanced Confusion Matrix

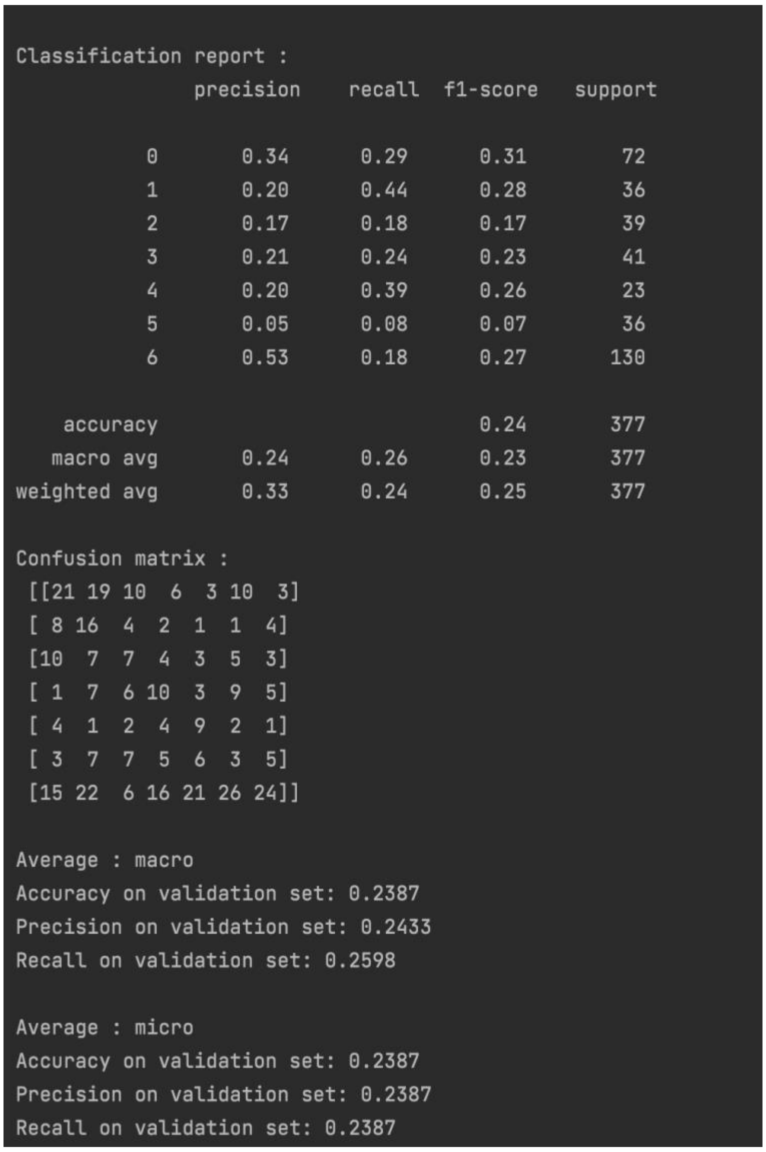
One difference we can see in figure 2 is along the true positive diagonal there is a better fit to predict the minority classes.

D) Show the precision and recall of your algorithm for each class, as well as the micro and macro averages. Explain the difference between the micro and macro averages.



```
Classification report :
              precision    recall  f1-score   support

           0       0.32      0.58      0.41        72
           1       0.32      0.22      0.26        36
           2       0.00      0.00      0.00        39
           3       0.11      0.02      0.04        41
           4       0.20      0.04      0.07        23
           5       0.00      0.00      0.00        36
           6       0.43      0.65      0.52       130

    accuracy                           0.36       377
   macro avg       0.20      0.22      0.19       377
weighted avg       0.26      0.36      0.29       377

Confusion matrix :
 [[42  5  1  1  0  0 23]
 [16  8  0  0  0  0 12]
 [22  1  0  1  0  0 15]
 [ 8  1  2  1  0  0 29]
 [ 7  1  0  2  1  0 12]
 [ 9  2  2  0  2  0 21]
 [27  7  2  4  2  3 85]]

Average : macro
Accuracy on validation set: 0.3634
Precision on validation set: 0.1976
Recall on validation set: 0.2182

Average : micro
Accuracy on validation set: 0.3634
Precision on validation set: 0.3634
Recall on validation set: 0.3634
```

Figure 3- Unbalanced- Classification report for Precision, Recall & Micro & Macro Averages



```
Classification report :
               precision    recall  f1-score   support

           0       0.34      0.29      0.31        72
           1       0.20      0.44      0.28        36
           2       0.17      0.18      0.17        39
           3       0.21      0.24      0.23        41
           4       0.20      0.39      0.26        23
           5       0.05      0.08      0.07        36
           6       0.53      0.18      0.27       130

    accuracy                           0.24       377
   macro avg       0.24      0.26      0.23       377
weighted avg       0.33      0.24      0.25       377

Confusion matrix :
 [[21 19 10  6  3 10  3]
 [ 8 16  4  2  1  1  4]
 [10  7  7  4  3  5  3]
 [ 1  7  6 10  3  9  5]
 [ 4  1  2  4  9  2  1]
 [ 3  7  7  5  6  3  5]
 [15 22  6 16 21 26 24]]

Average : macro
Accuracy on validation set: 0.2387
Precision on validation set: 0.2433
Recall on validation set: 0.2598

Average : micro
Accuracy on validation set: 0.2387
Precision on validation set: 0.2387
Recall on validation set: 0.2387
```

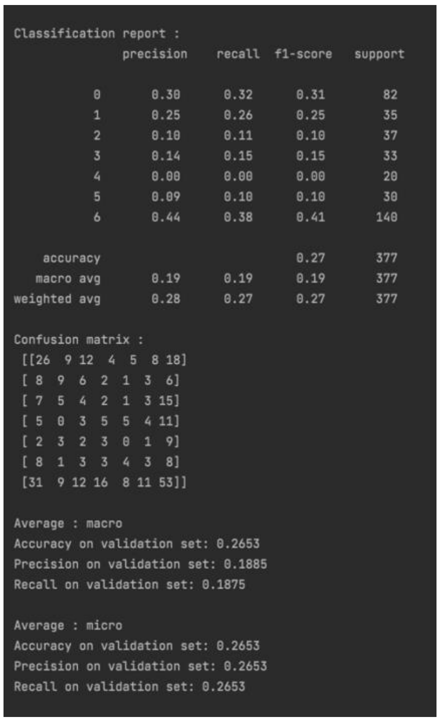
Figure 4- Balanced- Classification report for Precision, Recall & Micro & Macro Averages

A macro-average will compute the metric independently for each class and then take the average (hence treating all classes equally), whereas a micro-average will aggregate the contributions of all classes to compute the average metric. In a multi-class classification setup, micro-average is preferable if you suspect there might be class imbalance (i.e., you may have many more examples of one class than of other classes) (Datascience.stackexchange).

Student ID:210033137
Module: CS5014 Machine Learning

Part 4: Advanced tasks

B) Implement a 2nd degree polynomial expansion on the dataset. Explain how many dimensions this produces and why.



```
Classification report :
              precision    recall  f1-score   support

           0       0.30      0.32      0.31        82
           1       0.25      0.26      0.25        35
           2       0.10      0.11      0.10        37
           3       0.14      0.15      0.15        33
           4       0.00      0.00      0.00        20
           5       0.09      0.10      0.10        30
           6       0.44      0.38      0.41       140

    accuracy                           0.27       377
   macro avg       0.19      0.19      0.19       377
weighted avg       0.28      0.27      0.27       377

Confusion matrix :
 [[26  9 12  4  5  8 18]
 [ 8  9  6  2  1  3  6]
 [ 7  5  4  2  1  3 15]
 [ 5  0  3  5  5  4 11]
 [ 2  3  2  3  0  1  9]
 [ 8  1  3  3  4  3  8]
 [31  9 12 16  8 11 53]]

Average : macro
Accuracy on validation set: 0.2653
Precision on validation set: 0.1885
Recall on validation set: 0.1875

Average : micro
Accuracy on validation set: 0.2653
Precision on validation set: 0.2653
Recall on validation set: 0.2653
```

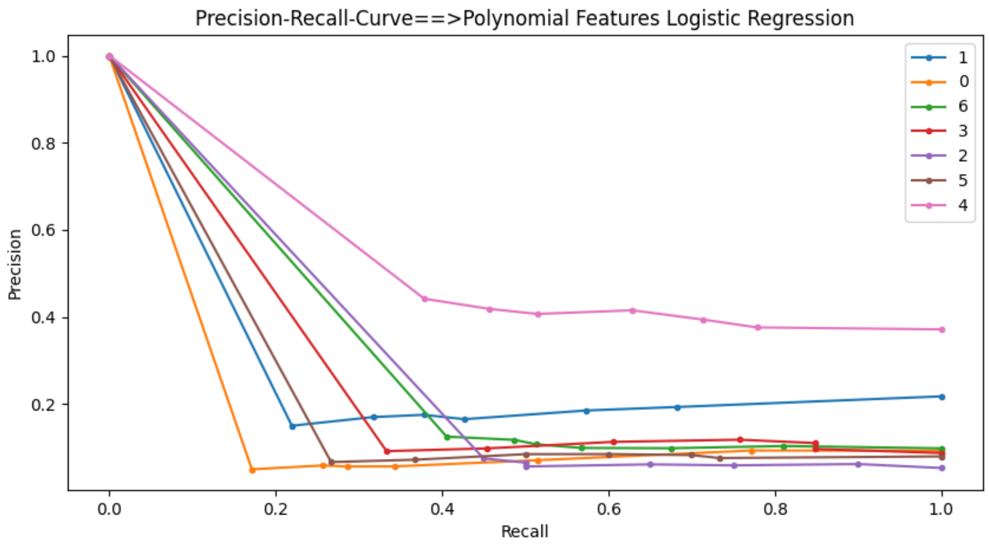
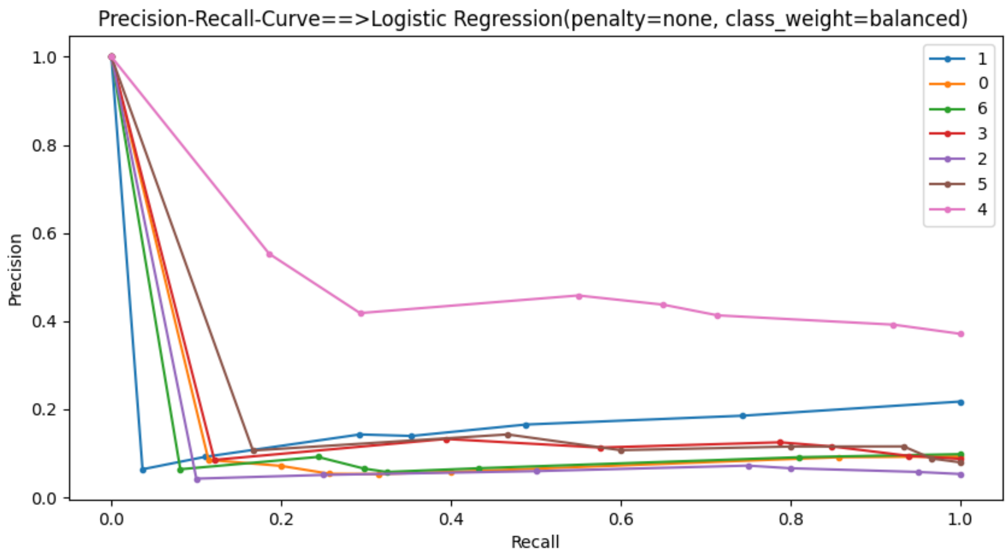
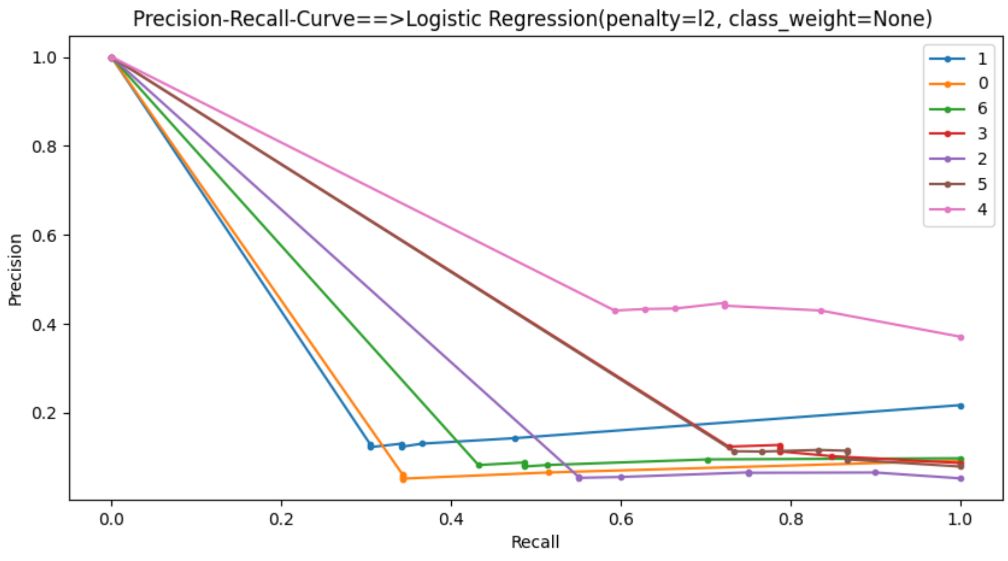


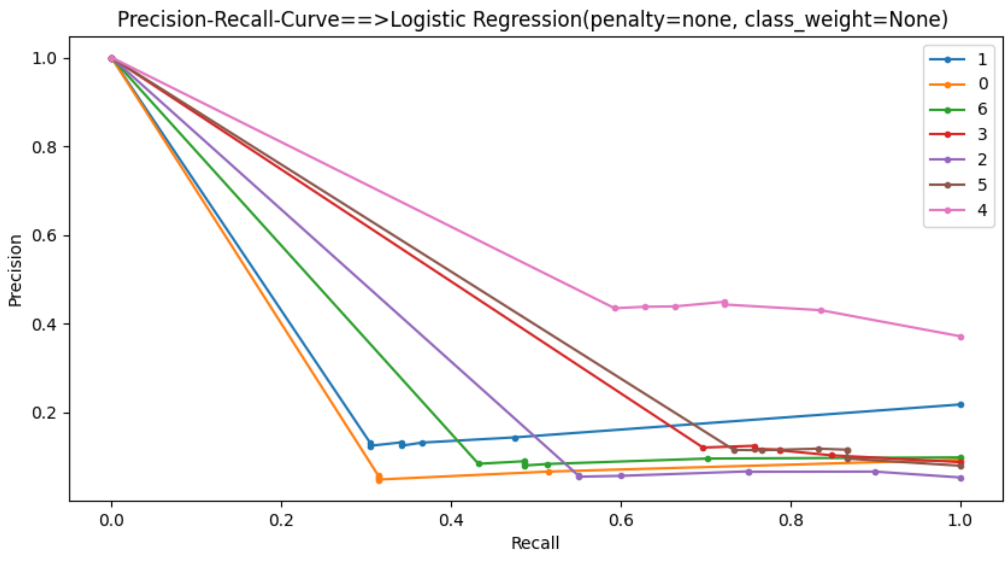
Figure 5- Polynomial expansion

D) Extend your solution to provide Precision-Recall plots for each class of nicotine consumption. You may need to independently explore advanced scikit-learn functionality for this.

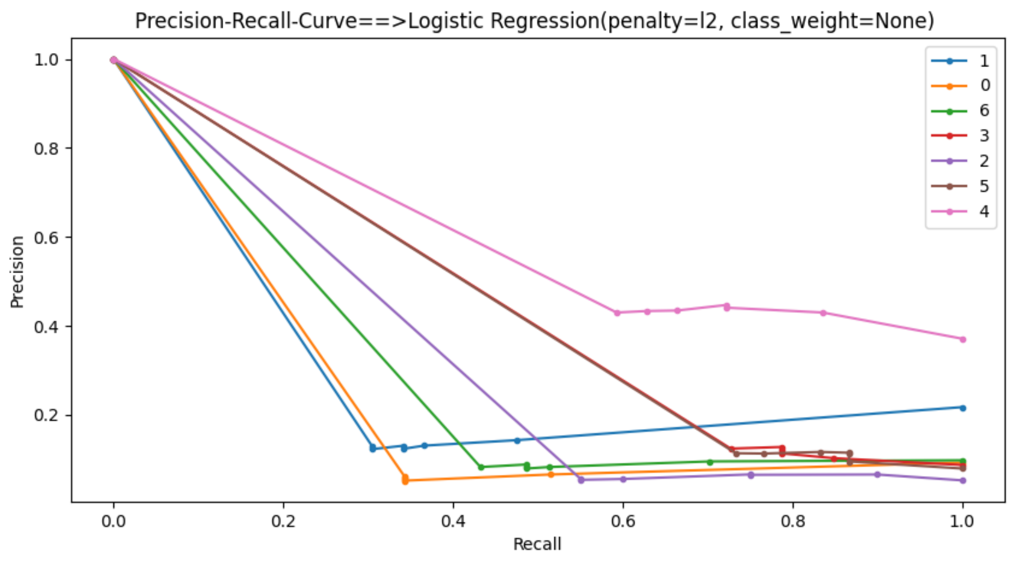


Student ID:210033137
Module: CS5014 Machine Learning





Student ID:210033137
Module: CS5014 Machine Learning



References

https://www.geeksforgeeks.org/ml-decision-function/

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html