Machine Learning Assignment 2

Reshan Faraz

PhD19006

Answer to Question 1:

Preprocessing of Data: I load the data using read_csv() of pandas after that ,After that I create two numpy arrays one for sample and other for label.In the sample array there is column which have nominal values like 'I','F','M' . I convert the M to 1,F to 0 and I to -1. For this question I implemented my MSE(Mean Square Array) function and K-fold split for splitting the data. In k-fold split (in this case k=5) i first take starting 80% data as training data and rest as testing data for second time next 20% data as testing data and rest as training data and so on.
I also defined my predict function in *Regression* Class.

Answer a):
I wrote my own predict function in Regression class , I used fit from sklearn and got the coefficient and intercept of the plane and tried to predict the model based on the training of the model.
Regression.py is attached for the reference.

Answer b): Following is a table containing training and validation mean square (MSE) error for each fold.

Training MSE(sklearn)	Validation MSE(sklearn)	Training MSE(Custom)	Validation MSE(Custom)
5.05621	3.97998	5.05621	3.97998
5.33394	3.02439	5.33394	3.02439
4.63991	5.83068	4.63991	5.83068
5.09884	3.81734	5.09884	3.81734
3.77404	9.88011	3.77404	9.88011

```
Mean of Training MSE (custom) is : 4.780590110876814
Mean of Validation MSE (custom) is 5.306499070829909
Mean of Training MSE (sklearn)is : 4.780590110876816
Mean of Validation MSE(sklearn) is 5.3064990708299105
```

From the table it is clear that the MSE function from sklearn and custom implementation have the same MSE(Mean Square Error).

Answer c): Following is a table containing training and validation mean square (MSE) error for each fold.

K-Fold	Training MSE(sklearn)	Validation MSE(sklearn)	Training MSE(Custom)	Validation MSE(Custom)
1	5.05621	3.97998	5.05621	3.97998
2	5.33394	3.02439	5.33394	3.02439
3	4.63991	5.83068	4.63991	5.83068
4	5.09884	3.81734	5.09884	3.81734
5	3.77404	9.88011	3.77404	9.88011

```
Mean of Training MSE (custom)is : 4.780590110876811
Mean of Validation MSE(custom) is 5.306499070829521
Mean of Training MSE (sklearn)is : 4.780590110876816
Mean of Validation MSE(sklearn) is 5.306499070829519
```

The Validation MSE(Mean Square Error) and Training MSE is the same as in part b.

Answer d):

Following are the results when I used Linear Regression from the sklearn.

The results are same because I implemented my Linear Regression as same as the sklearn even the MSE(Mean Square Error)

Training MSE(sklearn)	Validation MSE(sklearn)	Training MSE(Custom)	Validation MSE(Custom)
5.05621	3.97998	5.05621	3.97998
5.33394	3.02439	5.33394	3.02439
4.63991	5.83068	4.63991	5.83068
5.09884	3.81734	5.09884	3.81734
3.77404	9.88011	3.77404	9.88011

```
Mean of Training MSE (custom) is : 4.780590110876815
Mean of Validation MSE (custom) is 5.306499070829909
Mean of Training MSE (sklearn)is : 4.780590110876816
Mean of Validation MSE(sklearn) is 5.3064990708299105
```

Answer 2 :

Preprocessing of Data :

I load the data using scipy and then separate the data into samples and labels.

After that I used k-fold to split the data used for training of the model. I defined the k-split (k=5) where i first take starting 80% data as training data and rest as testing data for second time next 20% data as testing data and rest as training data and so on.

LogRegression Class:

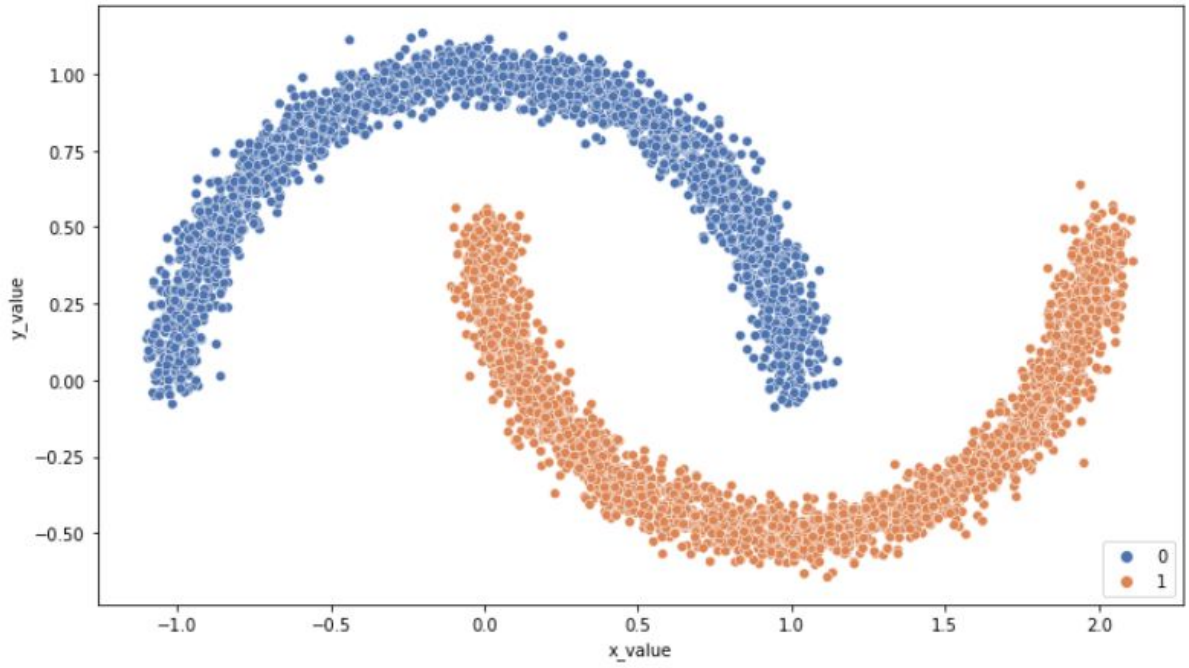
In the LogRegression class I wrote my own fit and predict methods.

Following are function definition for fit() and predict()

def fit(self,features, labels, lr, epochs,val_features, val_labels,l2_penalty):
Here, **self** the reference to the class,
features - training features,
labels - training labels ,
lr - Learning rate,
epochs - Number of Iteration,
val_features - validation features,
val_labels - validation Labels,
l2_penalty - L2 regularization (L2 Penalty)

When we take reference of the class by calling its constructor we must pass the type of classification as *binary,OVO,OVR* as *LogResgression('binary')* for Binary Classification.

Answer a) Following are the scatter graph for the visualization. The labels are 0 and 1 which are mentioned in the graph.

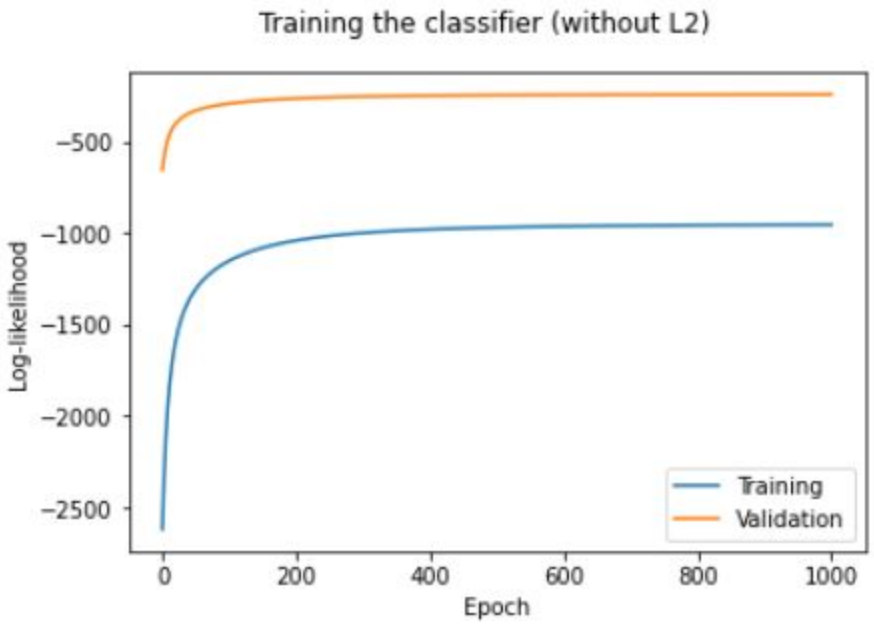


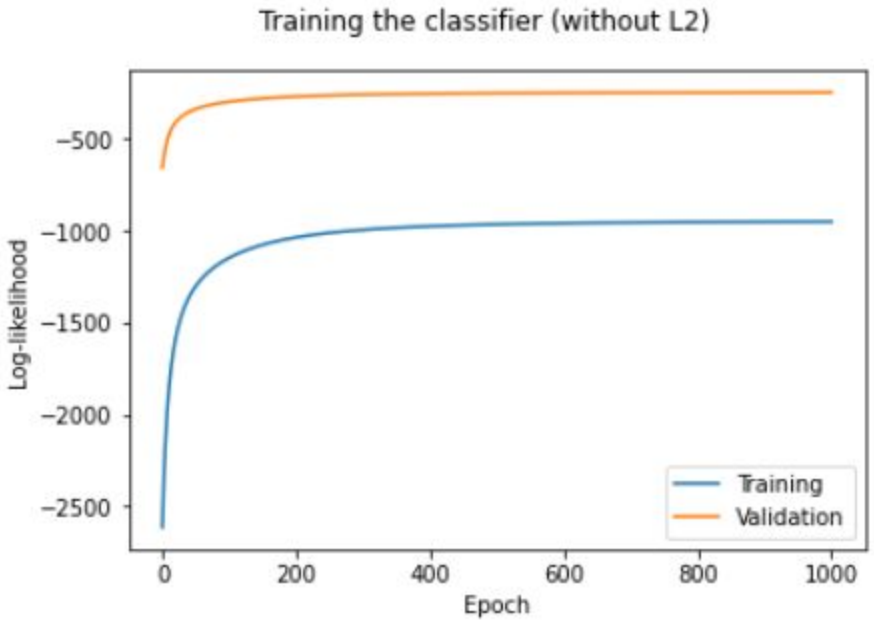
Answer b:
I wrote my own Log Regression class ,I defined the predict and fit function. using the log likelihood and other methods like gradient ascent as I tried to maximize the log likelihood.I also defined some helper function for the same. I also write my own accuracy function and compare with the sklearn.

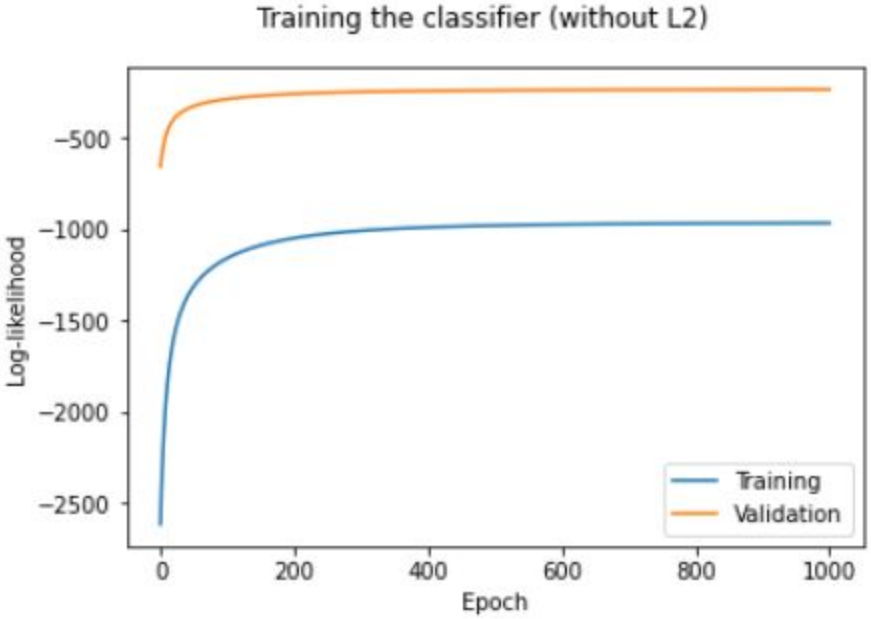
Answer c:

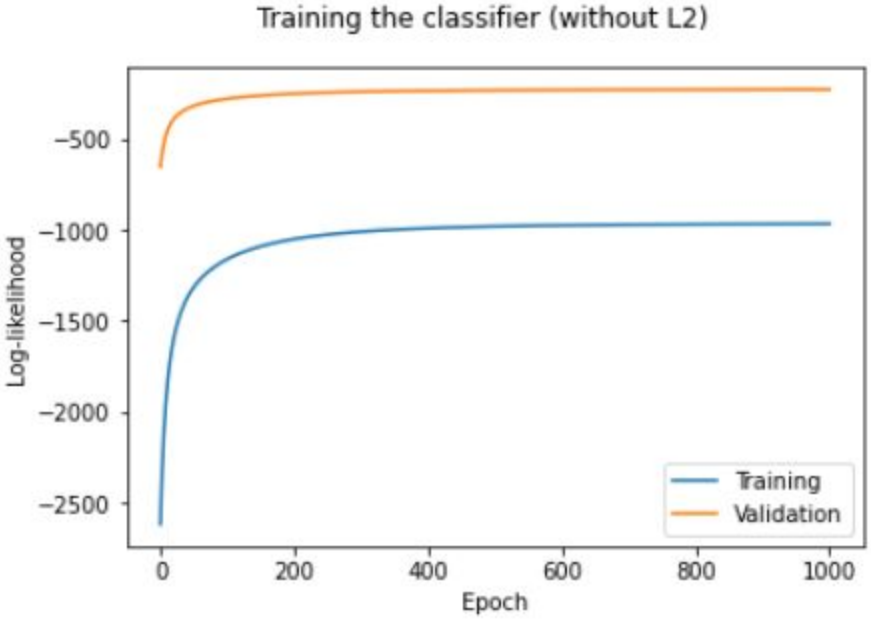
Following are the graph and table for each fold respectively from 1 to 5.

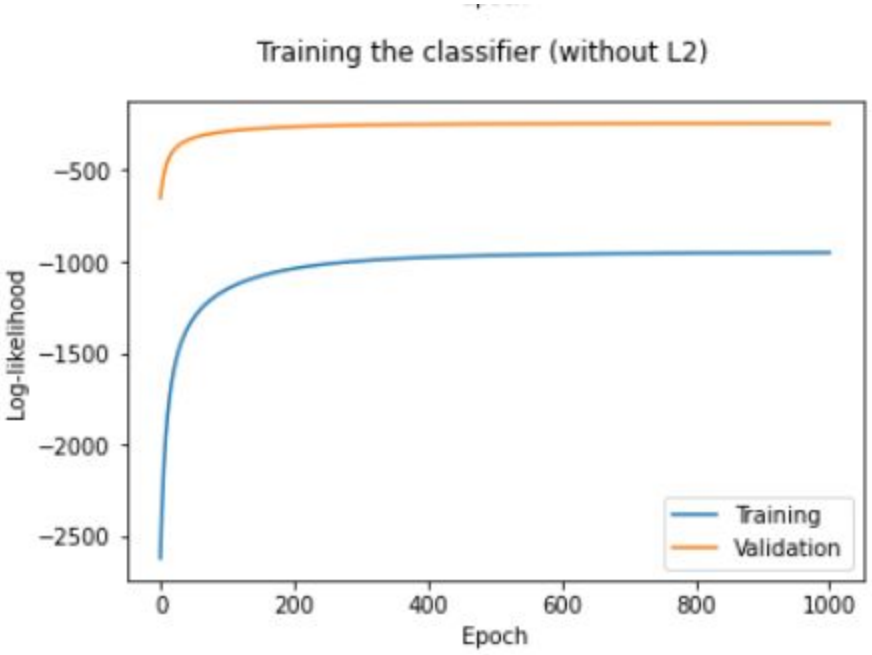
The training and validation learning statistics are given in single graph for each fold as mentioned in the question






Training the classifier (without L2)
−500
−1000
−1500
−2000
−2500
Log-likelihood
0
200
400
600
800
1000
Epoch
Training
Validation


Training the classifier (without L2)
−500
−1000
−1500
−2000
−2500
Log-likelihood
0
200
400
600
800
1000
Epoch
Training
Validation



Here is table for User Defined Accuracy and from sklearn with Mean Accuracy

K-Fold	Training Accuracy(sklearn)	Validation Accuracy(sklearn)	Training Accracy(Custom)	Validation Accuracy(Custom)
1	0.8915	0.887	0.8915	0.887
2	0.89	0.891	0.89	0.891
3	0.8895	0.892	0.8895	0.892
4	0.889	0.895	0.889	0.895
5	0.892	0.889	0.892	0.889

Mean of Training Accuracy is : 0.8904
Mean of Validation Accuracy is 0.8907999999999999

Following are table for zero one loss for each fold:

K-Fold	Training Loss	Validation Loss
1	0.1085	0.113
2	0.11	0.109
3	0.1105	0.108
4	0.111	0.105
5	0.108	0.111

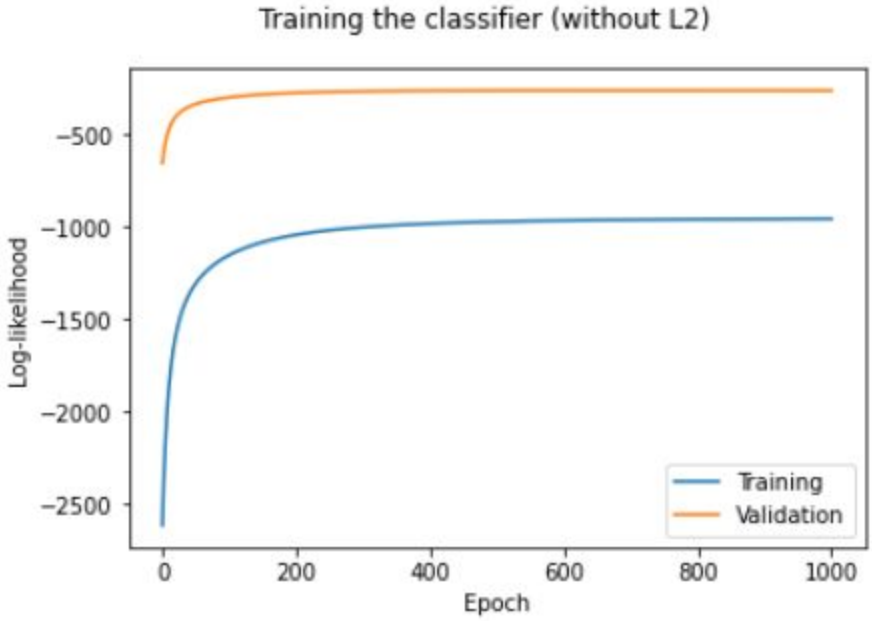
Mean of Training Loss is : 0.1096
Mean of Validation Loss is 0.10919999999999999

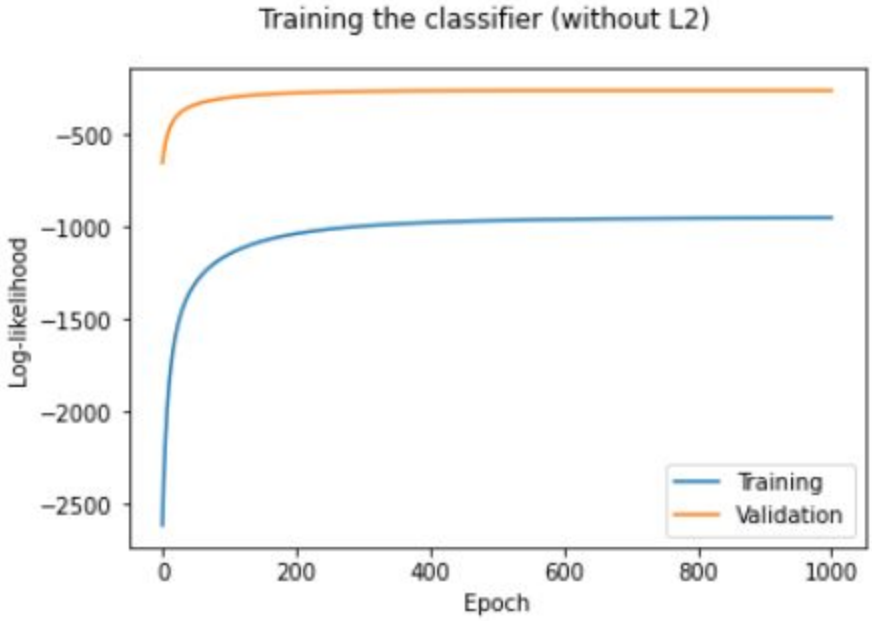
Answer d:)

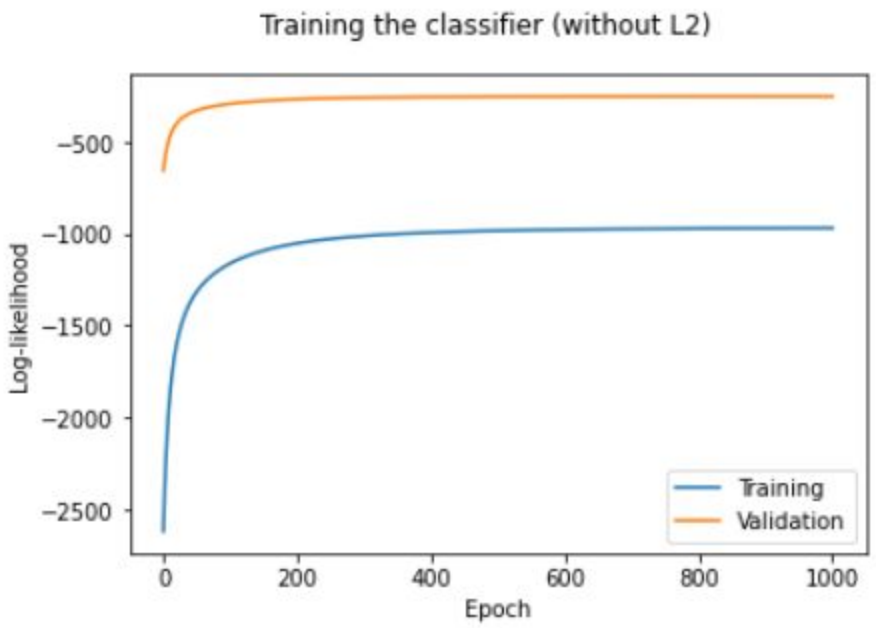
Using L2 regularization with the following parameters :

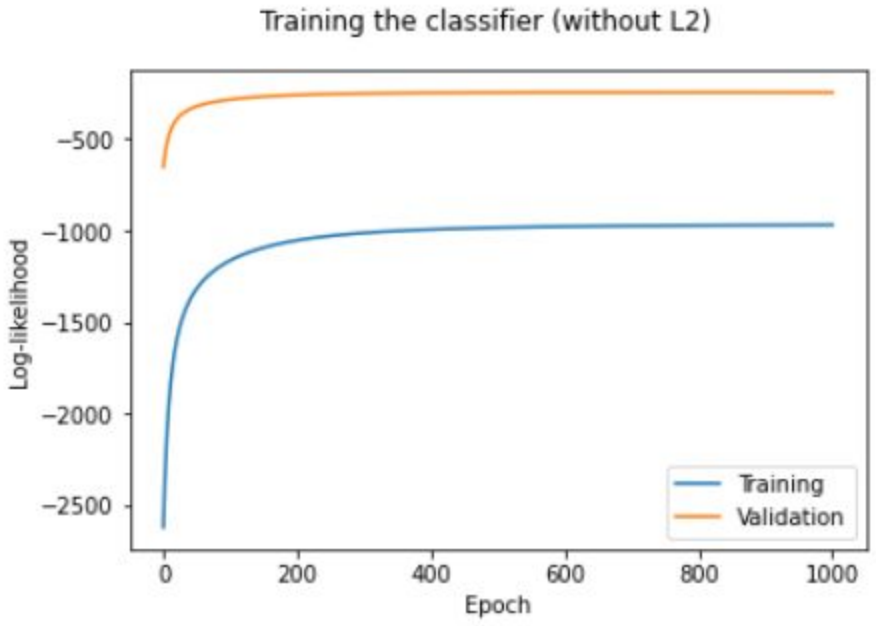
Maximum iteration = 1000
Learning rate = 0.0001

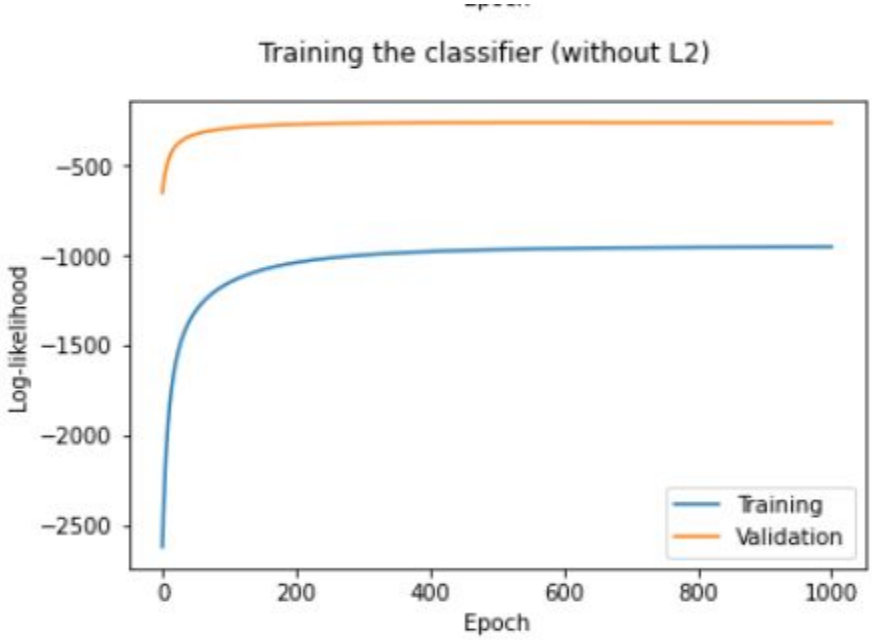
With using grid search to find out the optimal value of hyperparameter which is **-1.2625**
Which can be found over 500 iterations
I also saved the model using joblib which are also attached with submission.
Following are the graphs for each fold from 1 to 5 . The training and validation learning statistics are given in a single graph for each fold as mentioned in the question.






Training the classifier (without L2)
−500
−1000
−1500
−2000
−2500
Log-likelihood
0
200
400
600
800
1000
Epoch
Training
Validation


Training the classifier (without L2)
−500
−1000
−1500
−2000
−2500
Log-likelihood
0
200
400
600
800
1000
Epoch
Training
Validation



Here is table for User Defined Accuracy and from sklearn with Mean Accuracy

K-Fold	Training Accuracy(sklearn)	Validation Accuracy(sklearn)	Training Accracy(Custom)	Validation Accuracy(Custom)
1	0.891	0.888	0.891	0.888
2	0.89075	0.89	0.89075	0.89
3	0.89	0.892	0.89	0.892
4	0.889	0.896	0.889	0.896
5	0.89275	0.889	0.89275	0.889

Mean of Training Accuracy is : 0.8907000000000002
Mean of Validation Accuracy is 0.891

From the table it can be seen user Implementation and sklearn implementation gives the same result for Training and Validation Accuracy
Following are table for zero one loss for each fold :

K-Fold	Training Loss	Validation Loss
1	0.109	0.112
2	0.10925	0.11
3	0.11	0.108
4	0.111	0.104
5	0.10725	0.111

Mean of Training Loss is : 0.10929999999999998
Mean of Validation Loss is 0.10899999999999999

Answer e:

In this question I run the Logistic Regression from sklearn and compare the result with the result obtained for c and d. I used the same parameter as used in 2-c and 2-d part. For not using regularization while training my model for comparison with 2-c,I set ‘C’(a parameter in Logistic Regression which is the inverse of the regularization strength so setting C high we can train our model with regularization).Following are the result in tabular form.

With L2 regularization as in part d

With L2 Regularization as in part d

K-Fold	Training Accuracy(sklearn)	Validation Accuracy(sklearn)	Training Accracy(Custom)	Validation Accuracy(Custom)
1	0.8925	0.888	0.8925	0.888
2	0.88975	0.891	0.88975	0.891
3	0.88975	0.893	0.88975	0.893
4	0.88875	0.893	0.88875	0.893
5	0.89225	0.89	0.89225	0.89

Mean of Training Accuracy is : 0.8905999999999998
Mean of Validation Accuracy is 0.8909999999999998

Following are the zero one loss table:

K-Fold	Training Loss	Validation Loss
1	0.1075	0.112
2	0.11025	0.109
3	0.11025	0.107
4	0.11125	0.107
5	0.10775	0.11

Mean of Training Loss is : 0.10939999999999998
Mean of Validation Loss is 0.10899999999999999

Without L2 regularization as in part c

Without l2 Regularization as in c

K-Fold	Training Accuracy(sklearn)	Validation Accuracy(sklearn)	Training Accracy(Custom)	Validation Accuracy(Custom)
1	0.8915	0.887	0.8915	0.887
2	0.8895	0.891	0.8895	0.891
3	0.8895	0.893	0.8895	0.893
4	0.89025	0.892	0.89025	0.892
5	0.89125	0.888	0.89125	0.888

Mean of Training Accuracy is : 0.8904
Mean of Validation Accuracy is 0.8902000000000001

Following are the zero one loss table

K-Fold	Training Loss	Validation Loss
1	0.1085	0.113
2	0.1105	0.109
3	0.1105	0.107
4	0.10975	0.108
5	0.10875	0.112

Mean of Training Loss is : 0.10960000000000003
Mean of Validation Loss is 0.10979999999999998

From the tables it is clear that penalty (regularization) will not play a significant role in the development of a better model as there is slightly difference between the validation accuracy in using regularization and without regularization.FRom the table it is also clear that there is hardly difference between the Logistics Regression from the sklearn and from the our implementation.
We have achieved the same Validation accuracy as scikit-learn's implementation.

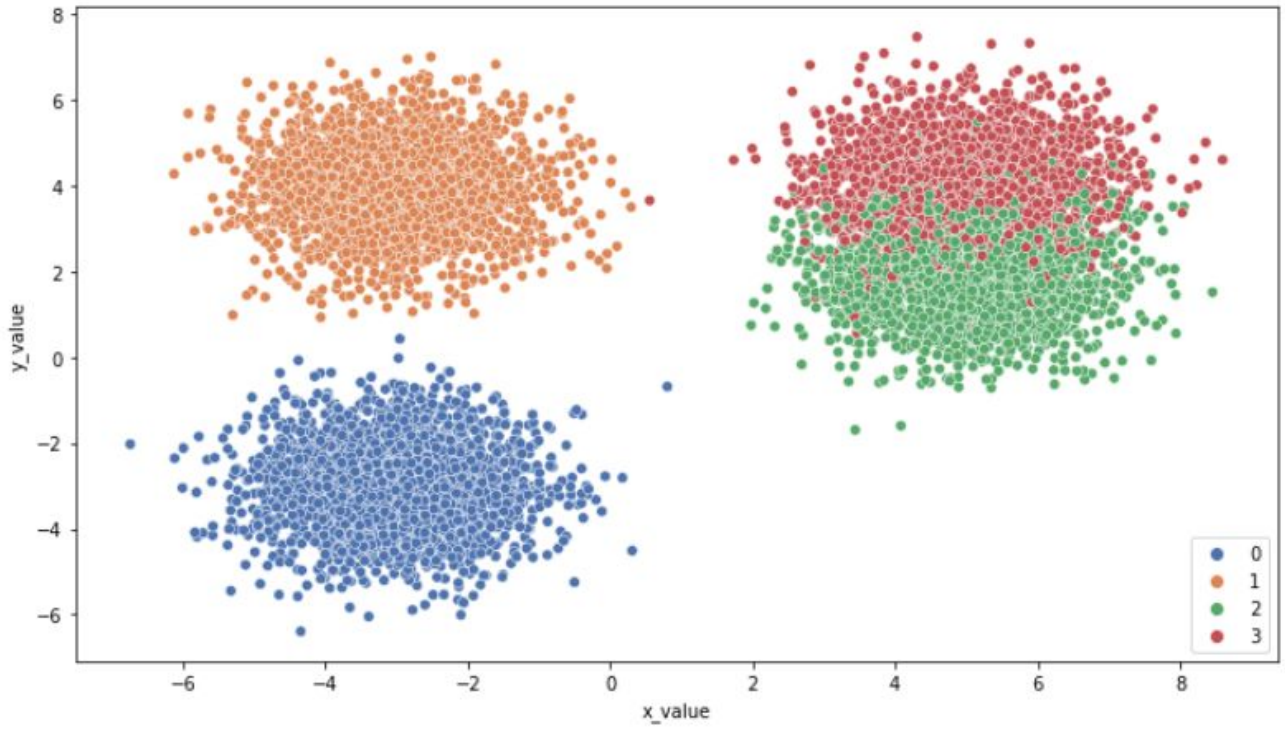
Answer 3 :

Preprocessing of Data :I load the data using scipy and then separate the data into samples and labels. I used k-fold to split the data used for training of the model. I defined the k-split (k=5) where i first take starting 80% data as training data and rest as testing data for second time next 20% data as testing data and rest as training data and so on.

LogRegression Class:
In the LogRegression class I wrote my own fit and predict methods.
Following are function definition for fit() and predict()
def fit(self,features, labels, lr, epochs,val_features, val_labels,l2_penalty):
Here, **self** the reference to the class,
features - training features,
labels - training labels ,
lr - Learning rate,
epochs - Number of Iteration,
val_features - validation features,
val_labels - validation Labels,
l2_penalty - L2 regularization (L2 Penalty)

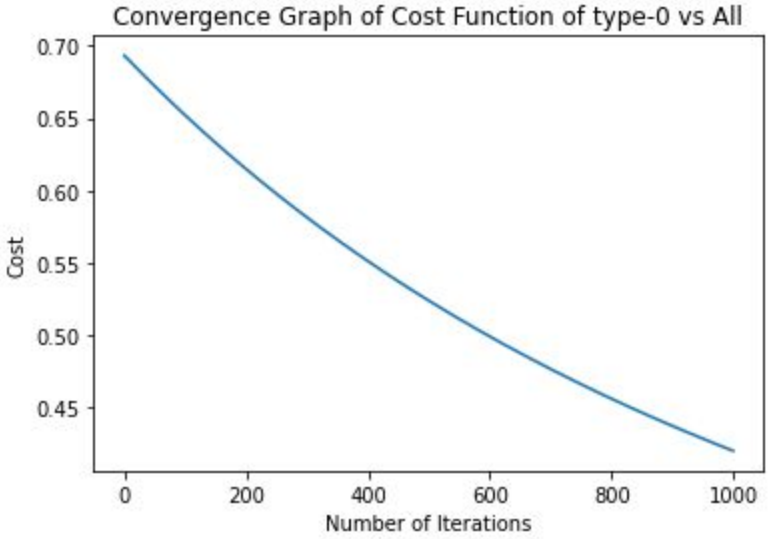
When we take reference of the class by calling its constructor we must pass the type of classification as *binary,OVO,OVR* as *LogResgression('binary')* for Binary Classification.

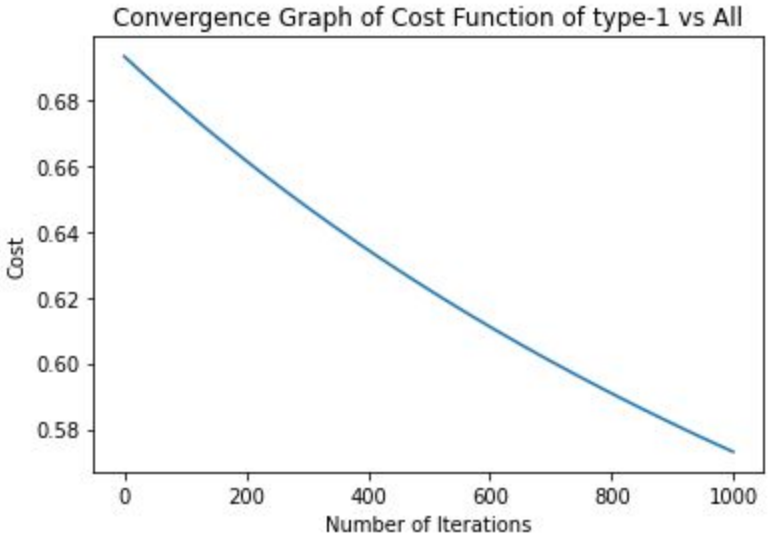
Answer a) Following are the scatter graph for the visualization. The labels are 0, 1,2,3 which are mentioned in the graph. From the graph it is clear that it is multinomial classification.

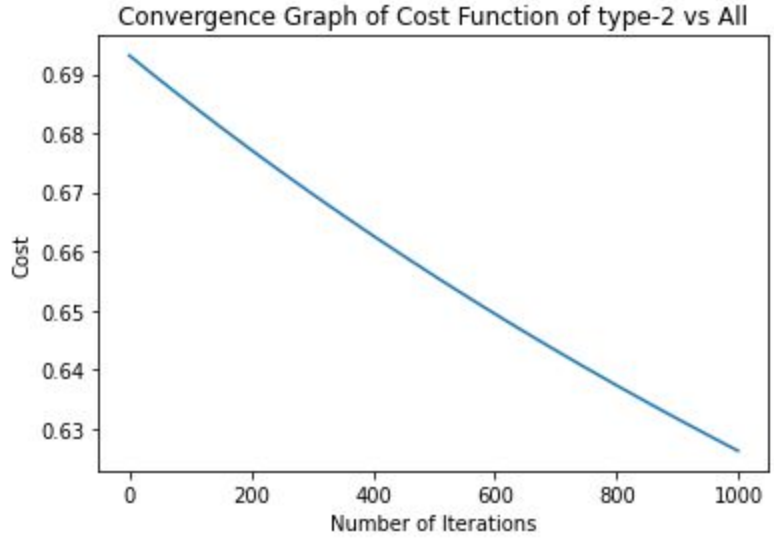


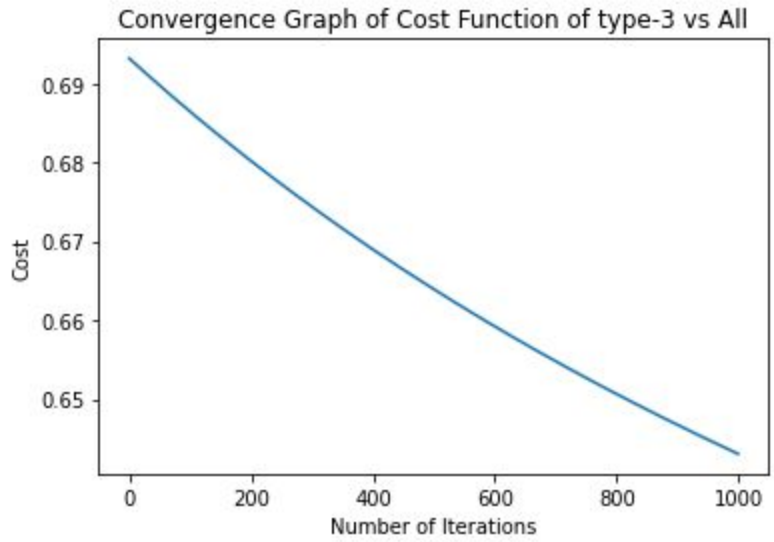
Answer 3-b): I implemented the OVR and OVO in LogRegression Class . We can use the class to find out the Accuracy and all other matrices which are shown below.

Below is the graph between the cost and the iteration to train the model for OVO.
The Learning rate is **0.0001** and the number of iteration is **1000** .


Convergence Graph of Cost Function of type-0 vs All
Cost
0.70
0.65
0.60
0.55
0.50
0.45
0
200
400
600
800
1000
Number of Iterations


Convergence Graph of Cost Function of type-1 vs All
Cost
0.68
0.66
0.64
0.62
0.60
0.58
0
200
400
600
800
1000
Number of Iterations





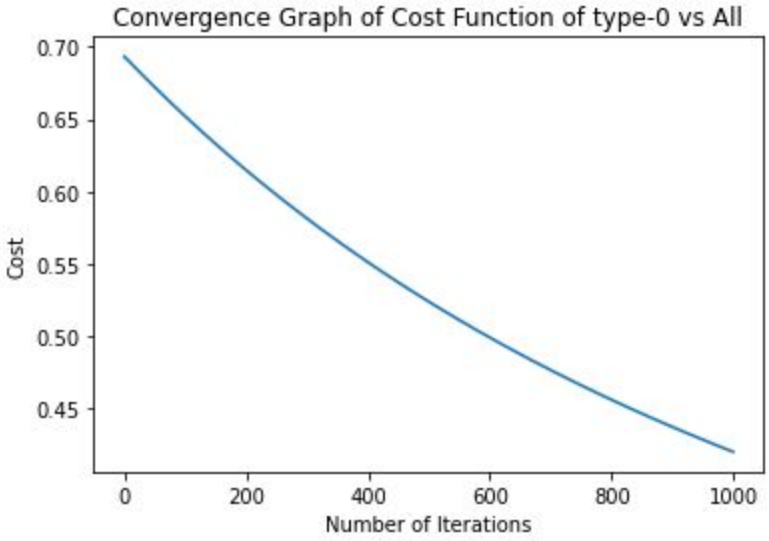
Following is the Accuracy Table for which I used LogRegession class to train the model and predict the values. After that I compare my implementation of the accuracy to the sklearn the result are shown below.

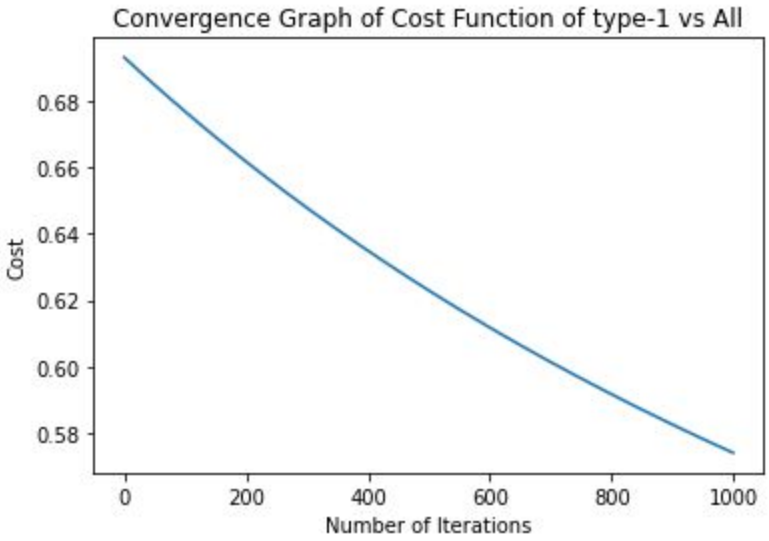
K-Fold	Training Accuracy(sklearn)	Validation Accuracy(sklearn)	Training Accracy(Custom)	Validation Accuracy(Custom)
1	0.776625	0.7875	0.776625	0.7875
2	0.761625	0.7645	0.761625	0.7645
3	0.765625	0.7495	0.765625	0.7495
4	0.765125	0.7575	0.765125	0.7575
5	0.764875	0.7755	0.764875	0.7755

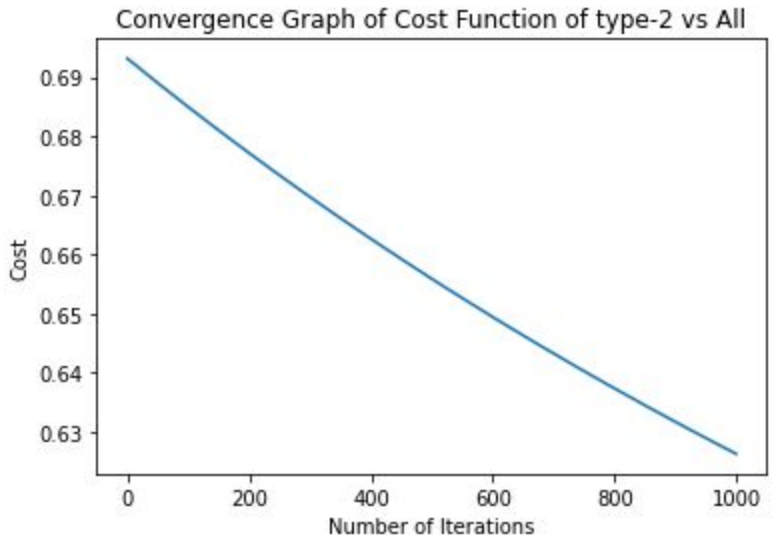
Mean of Training Accuracy is : 0.766775
Mean of Validation Accuracy is 0.7668999999999999

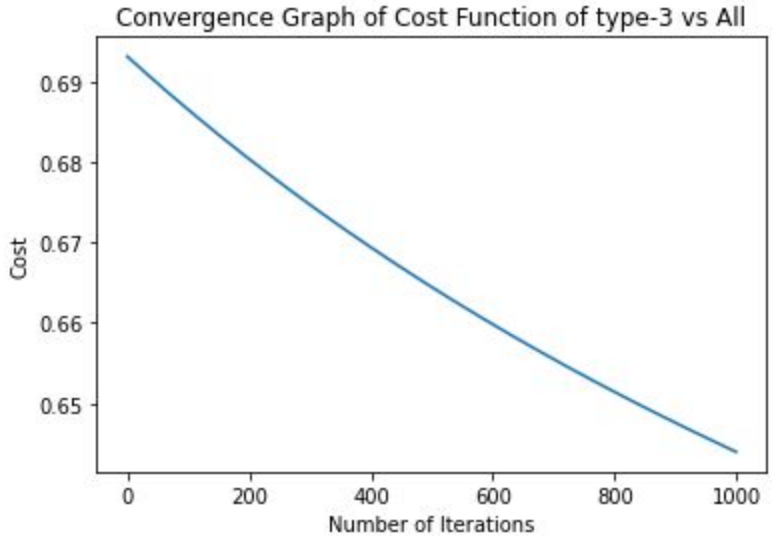
Answer 3-c):

I extend the LogRegression class to implement the OVR(One vs Rest) the Log Regression can handle both the L2 regularization and without L2 regularization.









K-Fold	Training Accuracy(sklearn)	Validation Accuracy(sklearn)	Training Accracy(Custom)	Validation Accuracy(Custom)
1	0.7635	0.7165	0.7635	0.7165
2	0.77025	0.766	0.77025	0.766
3	0.78	0.8085	0.78	0.8085
4	0.7635	0.7655	0.7635	0.7655
5	0.7715	0.785	0.7715	0.785

Mean of Training Accuracy is : 0.7697499999999999
Mean of Validation Accuracy is 0.7683

Answer 3-d:)

Following are the results obtained while using sklearn Logistic Classification as OVR multinomial classification from sklearn. The results are quite different .this may the fact

that they uses some other cost function to train the model or might be some other hypothesis for the model.

ONE VS ONE Using Sklearn

K-Fold	Training Accuracy(sklearn)	Validation Accuracy(sklearn)	Training Accracy(Custom)	Validation Accuracy(Custom)
1	0.92625	0.9185	0.92625	0.9185
2	0.922625	0.934	0.922625	0.934
3	0.926375	0.919	0.926375	0.919
4	0.92325	0.93	0.92325	0.93
5	0.92475	0.9235	0.92475	0.9235

Mean of Training Accuracy is : 0.92465
Mean of Validation Accuracy is 0.925

ONE VS Rest Using Sklearn

K-Fold	Training Accuracy(sklearn)	Validation Accuracy(sklearn)	Training Accracy(Custom)	Validation Accuracy(Custom)
1	0.92575	0.9145	0.92575	0.9145
2	0.9215	0.9325	0.9215	0.9325
3	0.924625	0.919	0.924625	0.919
4	0.921375	0.9295	0.921375	0.9295
5	0.924	0.9215	0.924	0.9215

Mean of Training Accuracy is : 0.9234500000000001
Mean of Validation Accuracy is 0.9234