Report

Q1.

Approach:

1. Converted categorical features into one hot encoding
2. In MSE, function y_actual is converted to numpy and then mean square error is calculated
3. In split-n fold function, for splitting data frame(df) ceil value is considered to divide it in n folds
4. For the question 5 folds is used
5. In every folds loop copy of data is made to do computations. So, that for a new fold main data is not manipulated
6. Predict() function is written from scratch in Regression class

b.

MSE Calculated from Function Written from Scratch

Folds	Training MSE	Validation MSE
1	3.762134947344344	9.773657764544327
2	5.310142665521644	3.021925857762971
3	4.625581743045682	5.847498259475927
4	5.074301542974596	3.836889823658044
5	5.03173857328424	3.9336617459483794

Mean Training MSE: 4.760779894434101
Mean Validation MSE: 5.28272669027793

MSE Calculated From Sklearn MSE Function

Folds	Training Sklearn MSE	Validation Sklearn MSE
1	3.762134947344344	9.773657764544327
2	5.310142665521644	3.021925857762971
3	4.625581743045682	5.847498259475927
4	5.074301542974596	3.836889823658044
5	5.03173857328424	3.9336617459483794

Mean Training Sklearn MSE : 4.760779894434101
Mean Validation Sklearn MSE: 5.28272669027793

There is no difference in the results of mse and sklearn mse results on different folds. Both are giving same answers foldwise.

c. Normal Equation1

In this normal equation1 is used to make predictions and then mse calculated on training and validation set

Folds	Training MSE	Validation MSE
1	3.762028052124054	9.77494268547752
2	5.307209308009528	2.988593177546498
3	4.6140454527084085	5.806497356009884
4	5.064279618471935	3.8227890471290147
5	5.0294375742481945	3.9534433229164714

Mean Training MSE: 4.755400001112425
Mean Validation MSE: 5.269253117815877

There is a slight deviation in mse values of training and validation from a part this because coefficients are learnt using normal equation instead of gradient descent

d.
Sklearn linear regression class is used for fitting the model and mse calculations on training and validation set

MSE Calculation foldwise:

Folds	Training MSE	Validation MSE
1	3.7614291207371484	9.791404322574014
2	5.311082069833134	2.9840286473908493
3	4.651528827446872	5.87013704582835
4	5.078056310329991	3.8447499252392343
5	5.03635660198887	3.9812899769282715

Mean Training MSE: 4.767690586067203
Mean Validation MSE: 5.294321983592143

There is a slight deviation in the performance of three approaches, as shown in table below

Approach	Mean Training MSE	Mean Validation MSE
b)Using MSE Function built from scratch	4.760779894434101	5.28272669027793
b)SKlearn MSE Function but fitting is using Regression class	4.760779894434101	5.28272669027793
c)Using normal equation1	4.755400001112425	5.269253117815877
d)Using Sklearn Linear Regression class for fitting and MSE calculation	4.767690586067203	5.294321983592143

Q2.

1.Issues Faced:

1. Due to the nature of loss function of logistic regression as the sigmoid values goes completely 0 or completely 1 the loss function becomes infinite. So, to handle this we did gradient clipping. Through gradient clipping we set the lower and upper bound on values so that sigmoid value will not be 0 or 1.

2. Another problem we faced is fixing the value of learning rate . If the learning rate was too low then convergence was slow and number of iterations were insufficient to minimize the loss function. But if the learning rate was too high then the loss function was fluctuating and overshooting. So, it was necessary to keep the learning rate not too low and not too high

2. Approach:

LogRegression class is used to perform binary or multiclass logistic regression with or without regularization.

beta- It is used to activate the L2 regularization. By default its value is 0(Not active L2 regularization)
multiclassification_type- It is used to tell if multi class classification then what to be followed OneVsOne or OneVsRest. By default its value is ovr but ovo is also possible

learning_rate: It tells which learning rate to be followed. Its default value is 0. 0000005

a. Analysis and visualization of data set

1. Target value distribution is as follows:
    a. 0 500
    b. 1 268
    Name: Outcome, dtype: int64

2. Data frame information is as follows
    a. RangeIndex: 768 entries, 0 to 767
    b. Data columns (total 9 columns)

#	Column	Non-Null Count	Dtype
0	Pregnancies	768 non-null	int64
1	Glucose	768 non-null	int64
2	BloodPressure	768 non-null	int64
3	SkinThickness	768 non-null	int64
4	Insulin	768 non-null	int64
5	BMI	768 non-null	float64
6	DiabetesPedigreeFunction	768 non-null	float64
7	Age	768 non-null	int64
8	Outcome	768 non-null	int64

3. Checked no null value in the data frame:
    a. Pregnancies 0
    b. Glucose 0
    c. BloodPressure 0
    d. SkinThickness 0
    e. Insulin 0
    f. BMI 0
    g. DiabetesPedigreeFunction 0
    h. Age 0
    i. Outcome 0

4. Checked no nan value in the data frame:
    a. Pregnancies 0
    b. Glucose 0
    c. BloodPressure 0
    d. SkinThickness 0
    e. Insulin 0
    f. BMI 0
    g. DiabetesPedigreeFunction 0
    h. Age 0
    i. Outcome 0

5. Description of all features in data frame

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

c.
Accuracy performance on 5-folds

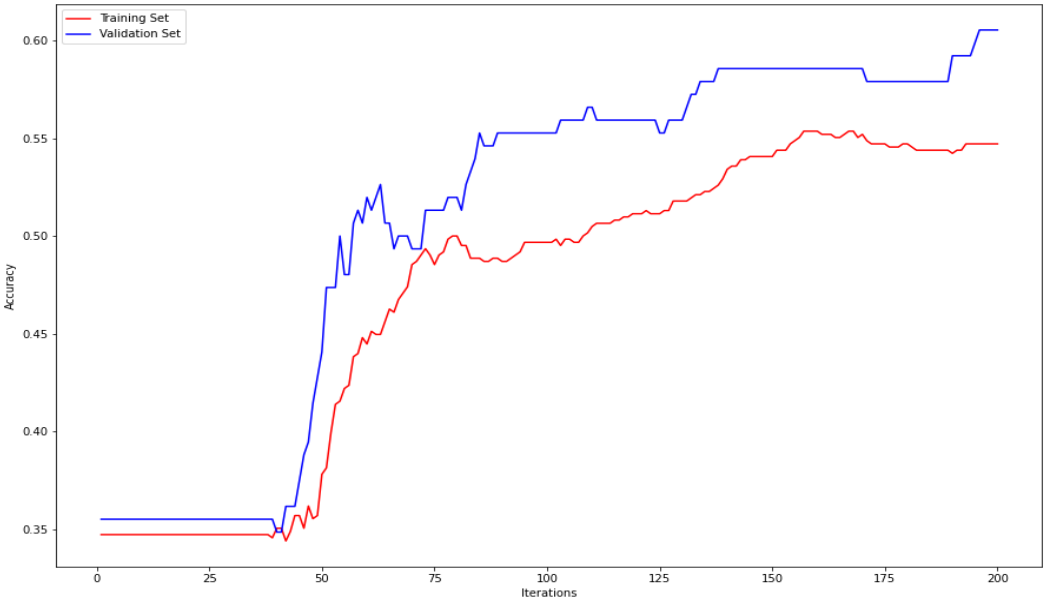
Folds	Training	Validation
1	0.5602605863192183	0.5909090909090909
2	0.5537459283387622	0.5194805194805194
3	0.6351791530944625	0.6558441558441559
4	0.5472312703583062	0.5194805194805194
5	0.547077922077922	0.6052631578947368

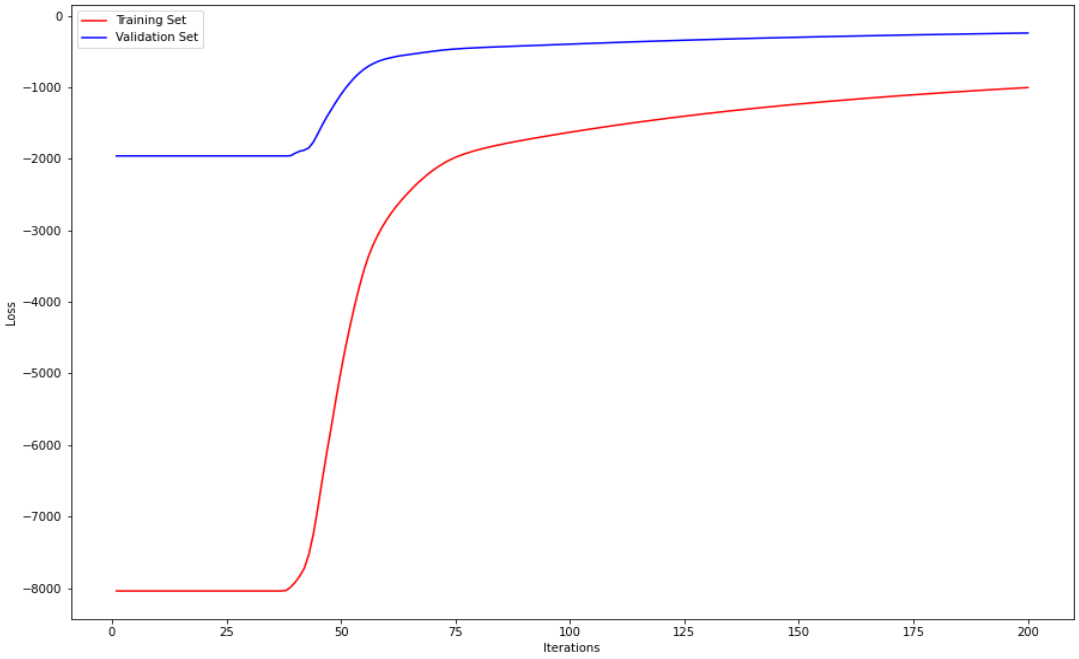
Mean Training Accuracy: 0.5686989720377342

Mean Validation Accuracy: 0.5781954887218045

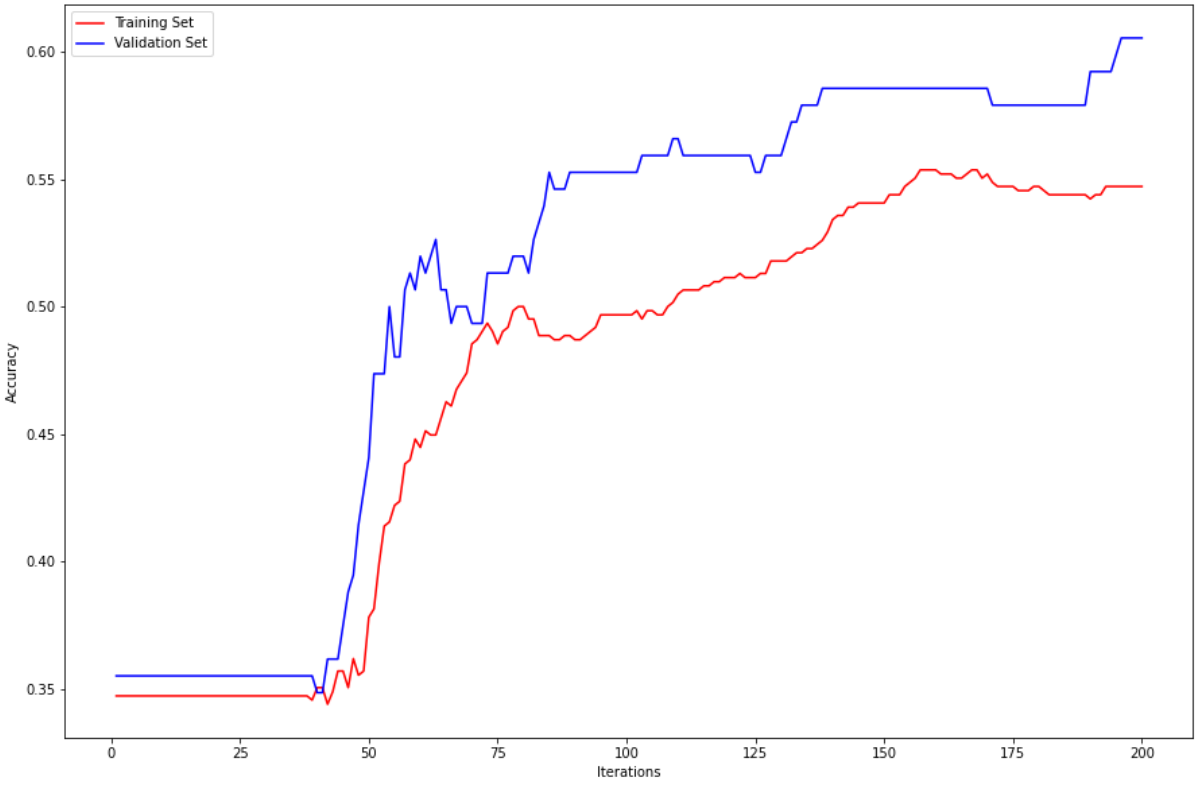
Plotting on loss vs iterations and accuracy vs iterations for each fold is as follows:

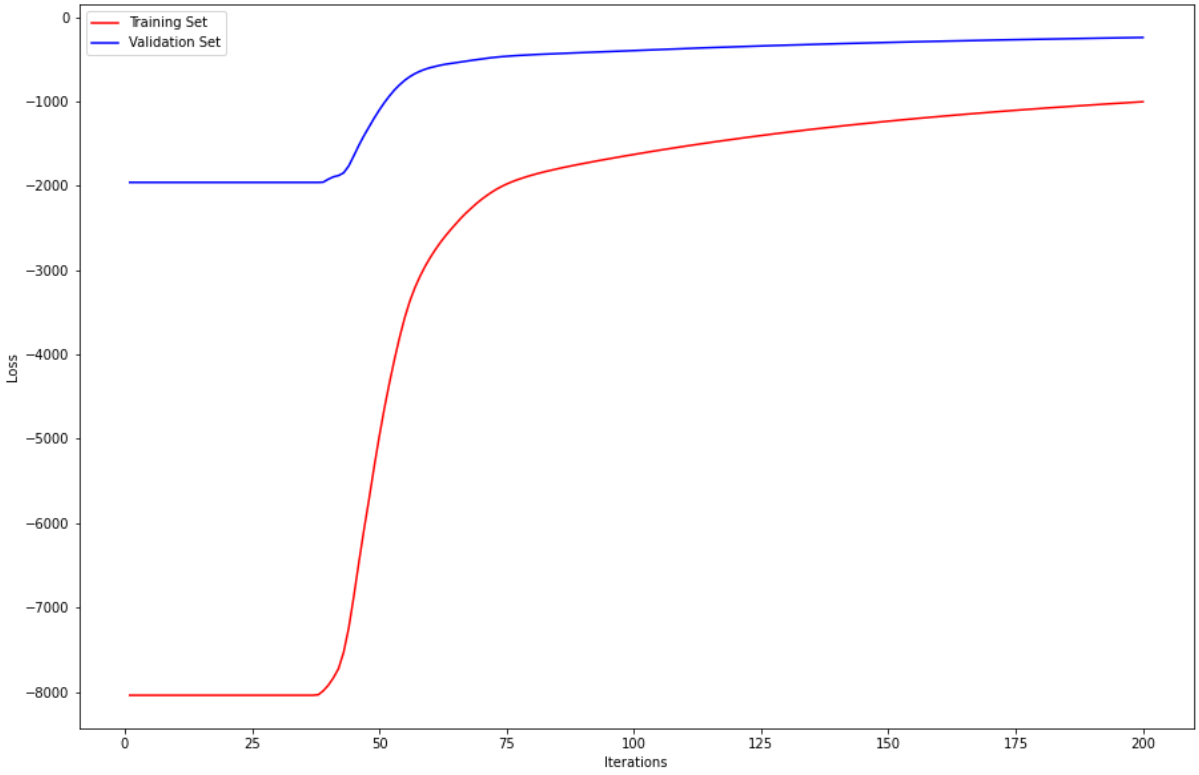
Fold 1:


Training Set
Validation Set
Accuracy
Iterations

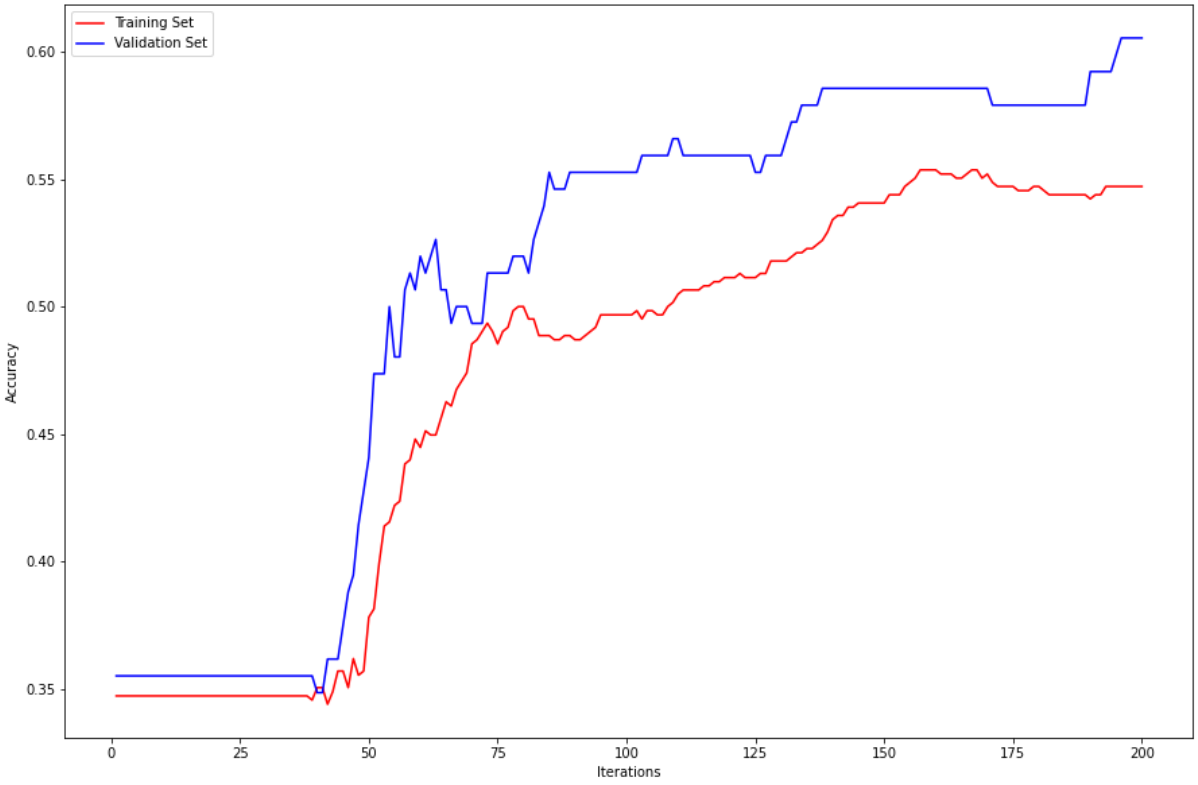

Training Set
Validation Set
Loss
Iterations

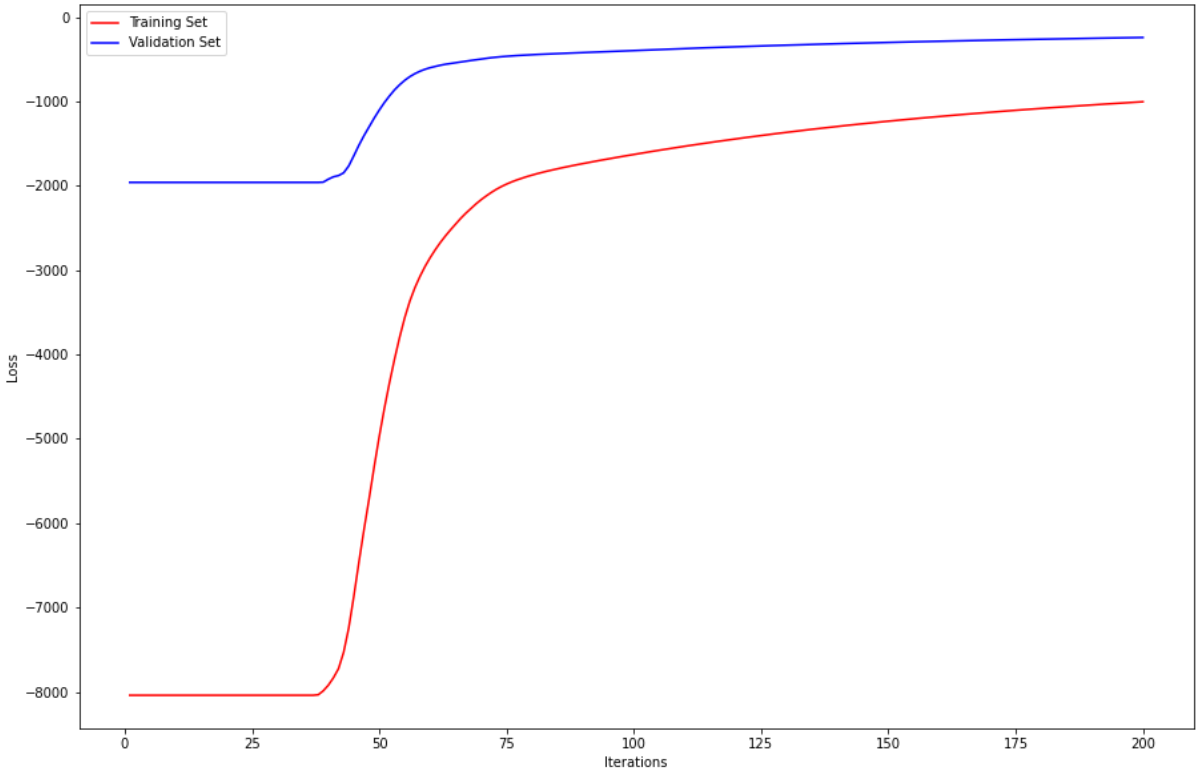
Fold 2


Training Set
Validation Set
Accuracy
Iterations
0.60
0.55
0.50
0.45
0.40
0.35
0
25
50
75
100
125
150
175
200

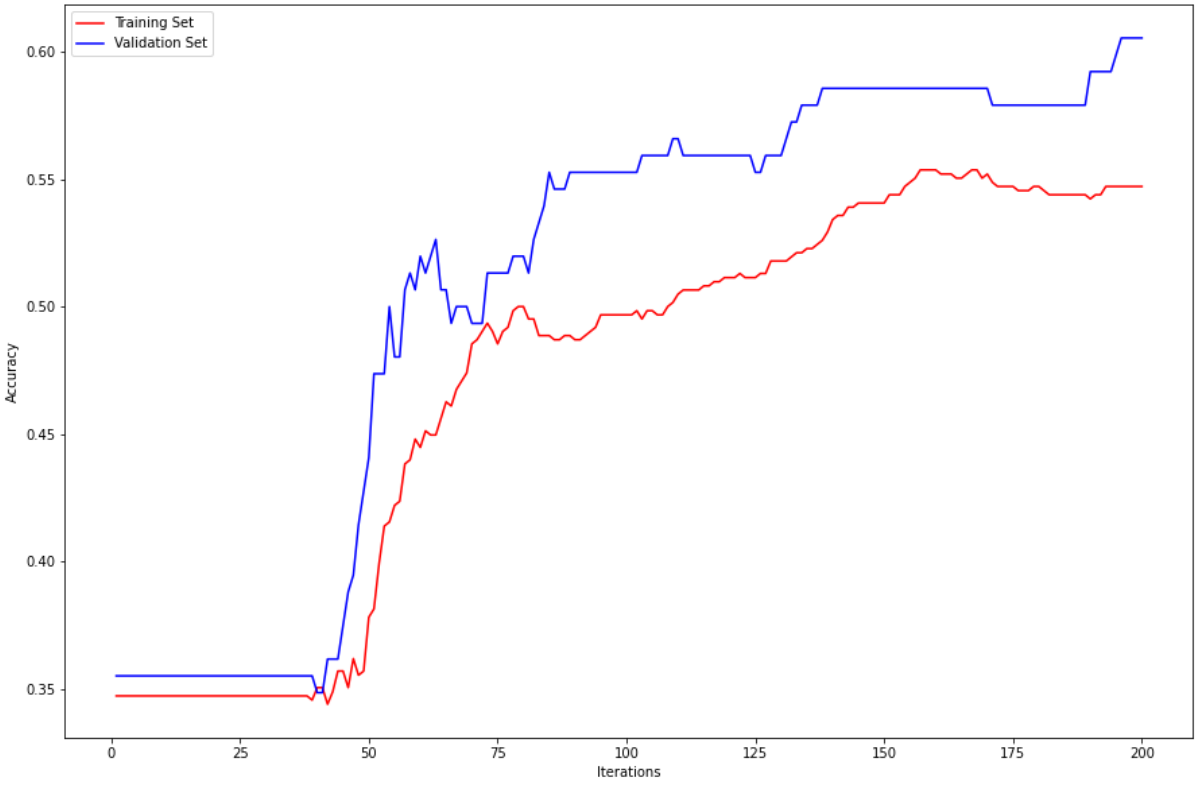

Training Set
Validation Set
Loss
Iterations
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
−8000
0
25
50
75
100
125
150
175
200

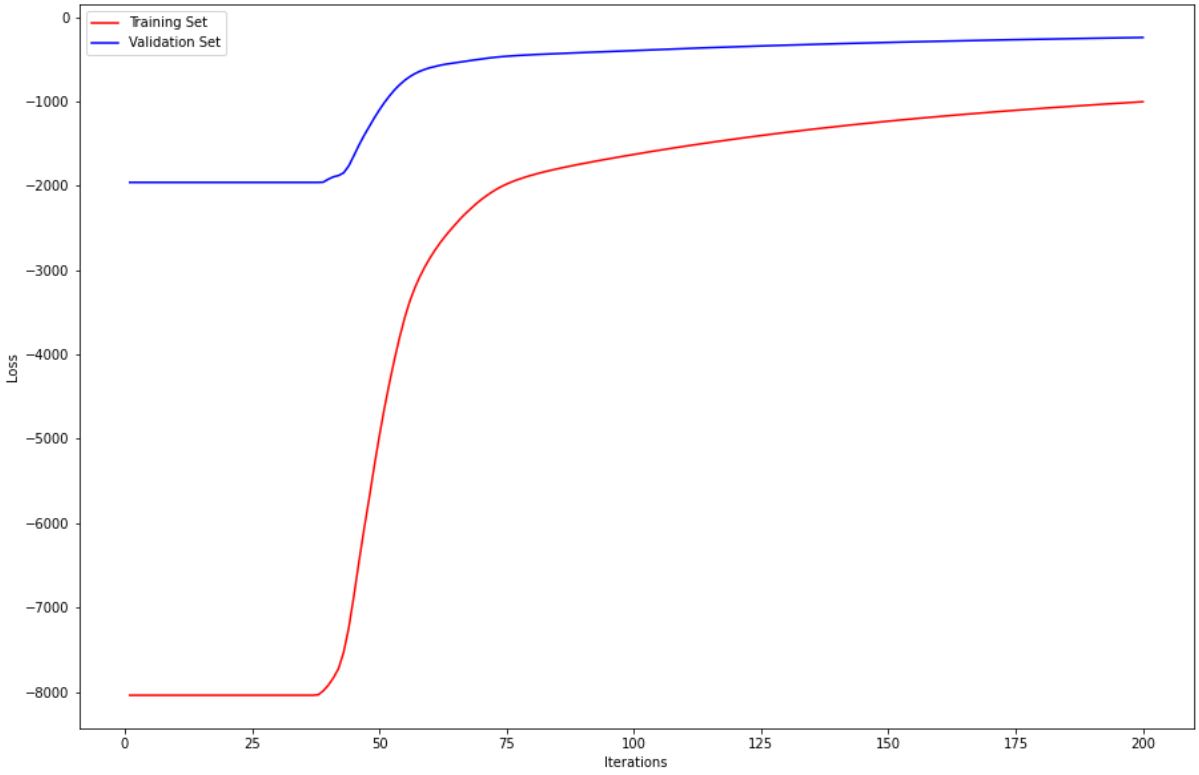
Fold 3


Training Set
Validation Set
Accuracy
Iterations
0.60
0.55
0.50
0.45
0.40
0.35
0
25
50
75
100
125
150
175
200

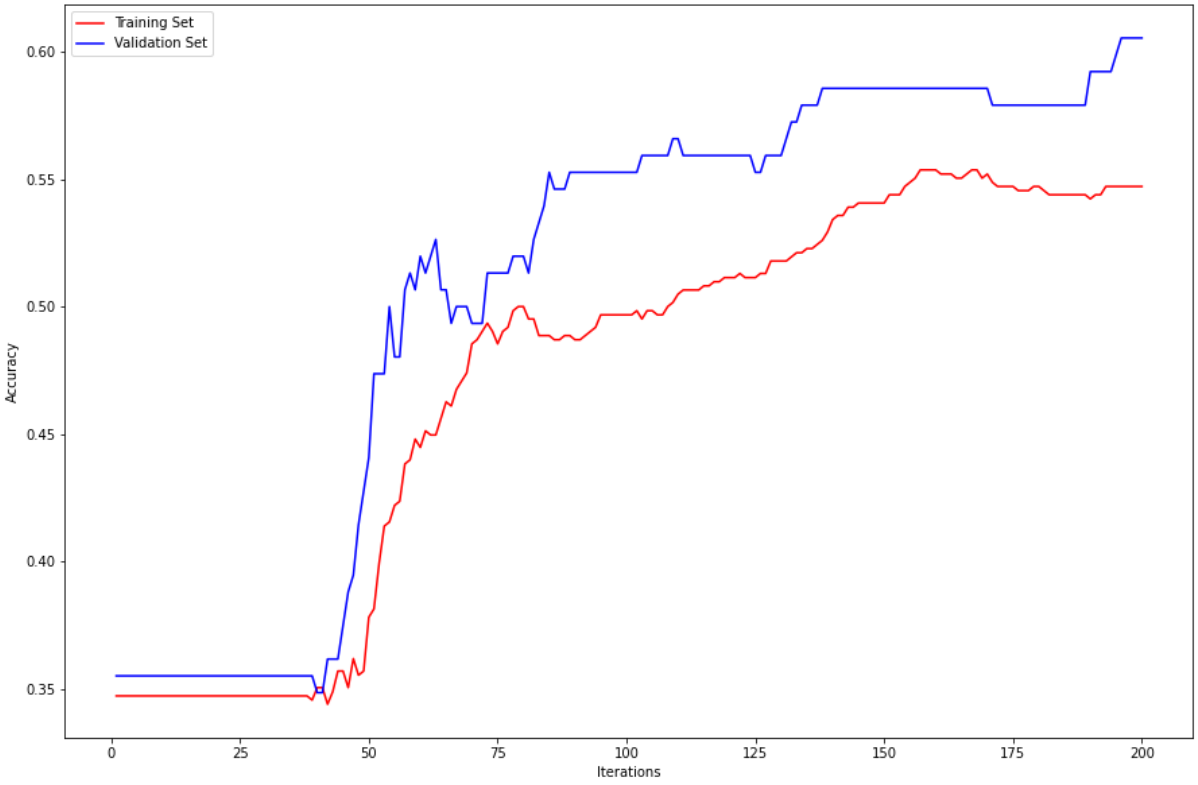

Training Set
Validation Set
Loss
Iterations
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
−8000
0
25
50
75
100
125
150
175
200

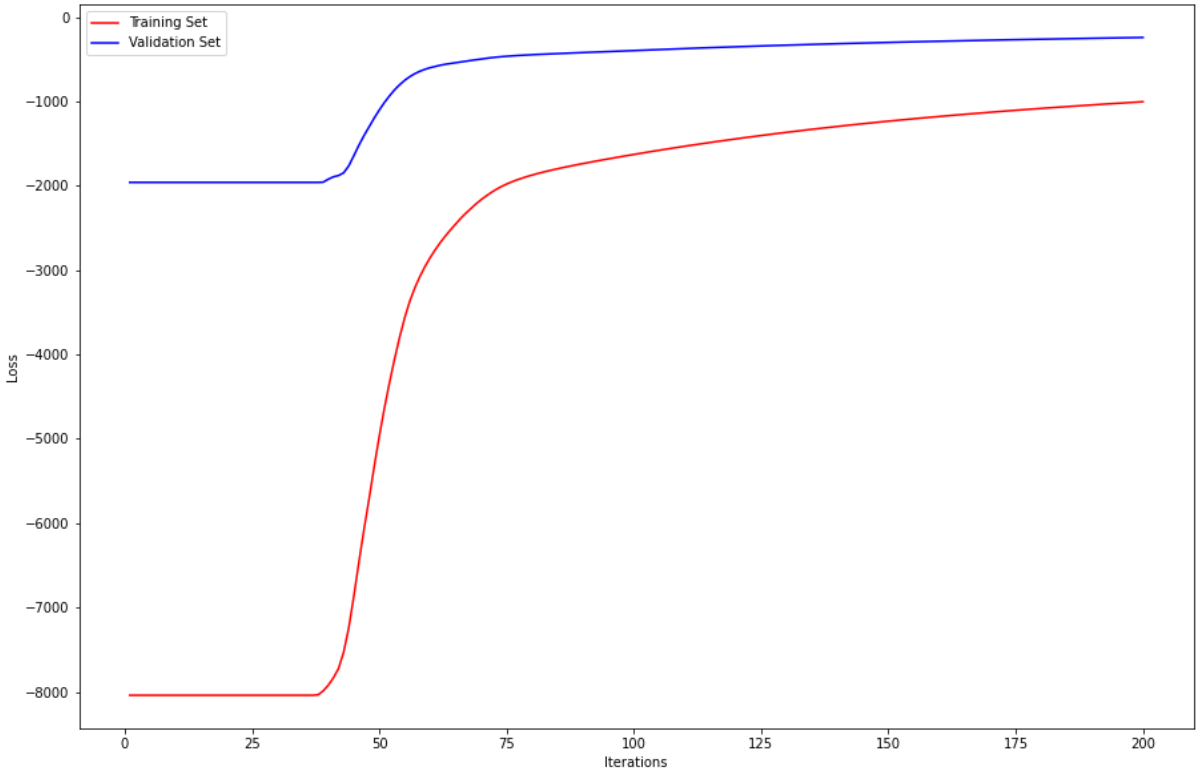
Fold 4


Training Set
Validation Set
Accuracy
Iterations
0.60
0.55
0.50
0.45
0.40
0.35
0
25
50
75
100
125
150
175
200


Training Set
Validation Set
Loss
Iterations
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
−8000
0
25
50
75
100
125
150
175
200

Fold 5





d.

Grid Search on the following over the regularization constant (λ=Beta in code) to obtain its optimal value

Beta	0.001	Loss Value-362.8546333118403
Beta	0.002	Loss Value-466.08277977706837
Beta	0.003	Loss Value-321.46195364357106

Beta 0.004 Loss Value-423.38150330964
Beta 0.005 Loss Value-418.455766354297
Beta 0.006 Loss Value-326.480782424211
Beta 0.007 Loss Value-327.86649365861786
Beta 0.008 Loss Value-564.6115533686244
Beta 0.009000000000000001 Loss Value-387.542811600011

Beta- 0.003 is chosen for regularization

Accuracy Performance is as follows:

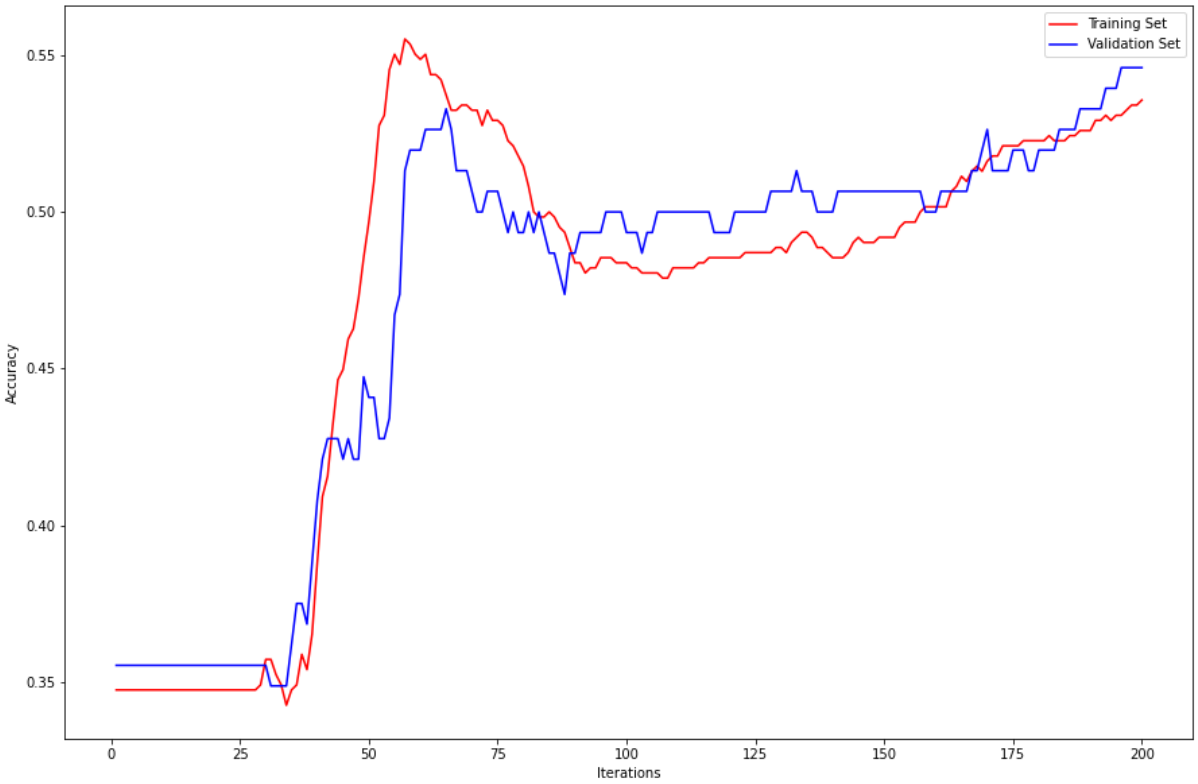
Folds	Training Accuracy	Validation Accuracy
1	0.5602605863192183	0.6038961038961039
2	0.5260586319218241	0.538961038961039
3	0.5586319218241043	0.5194805194805194
4	0.5977198697068404	0.5844155844155844
5	0.5357142857142857	0.5460526315789473

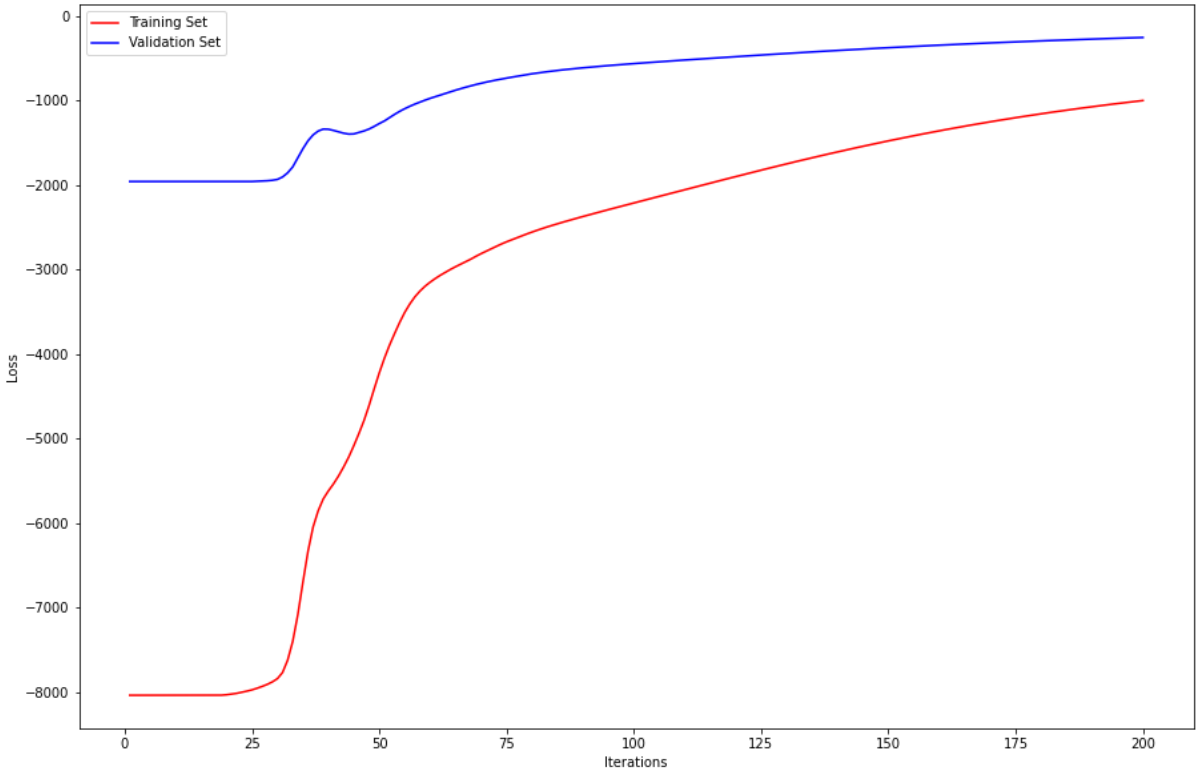
Mean Training Accuracy: 0.5556770590972546

Mean Validation Accuracy: 0.5585611756664388

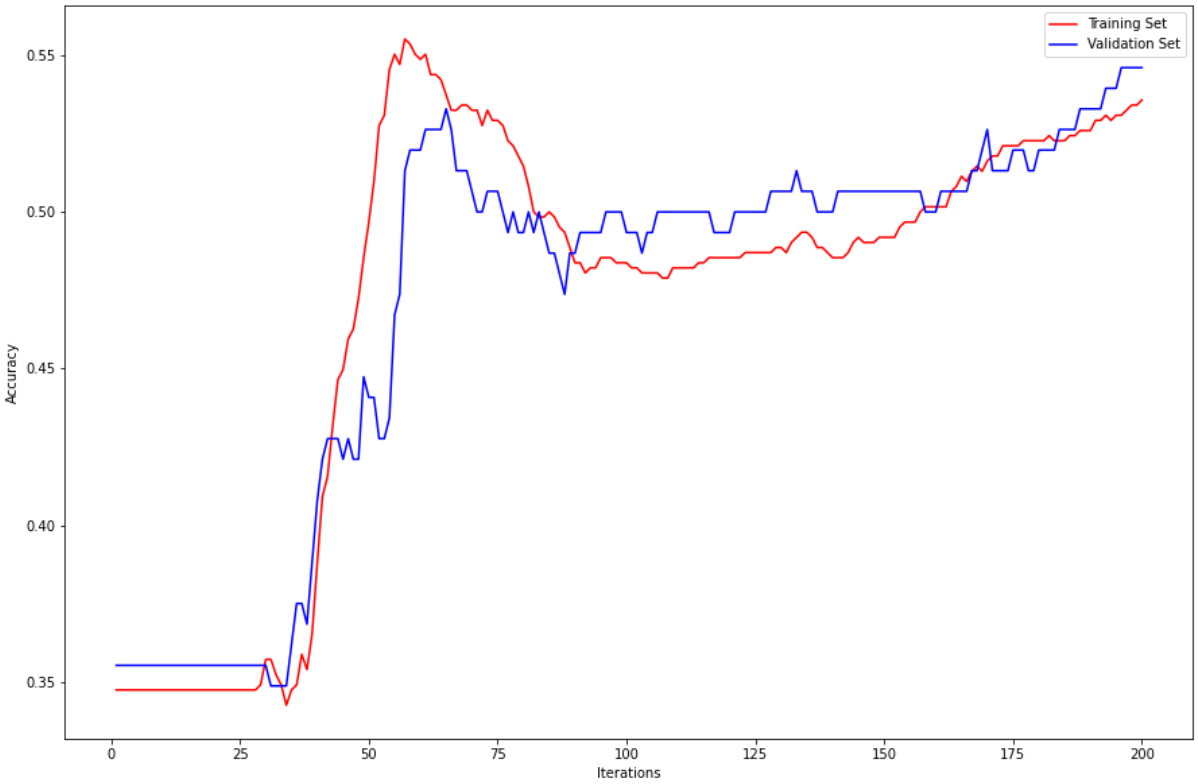
Plots of Accuracy vs iteration and loss vs iterations for each folds using regularization:

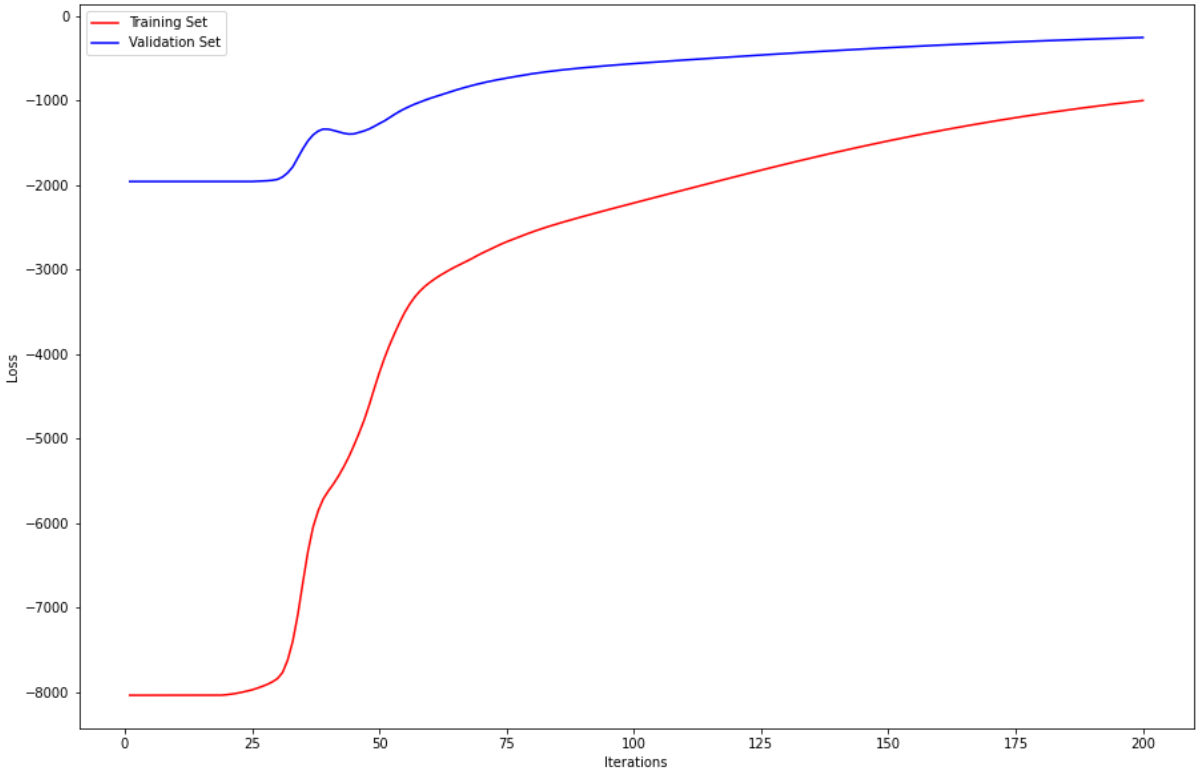
Fold 1



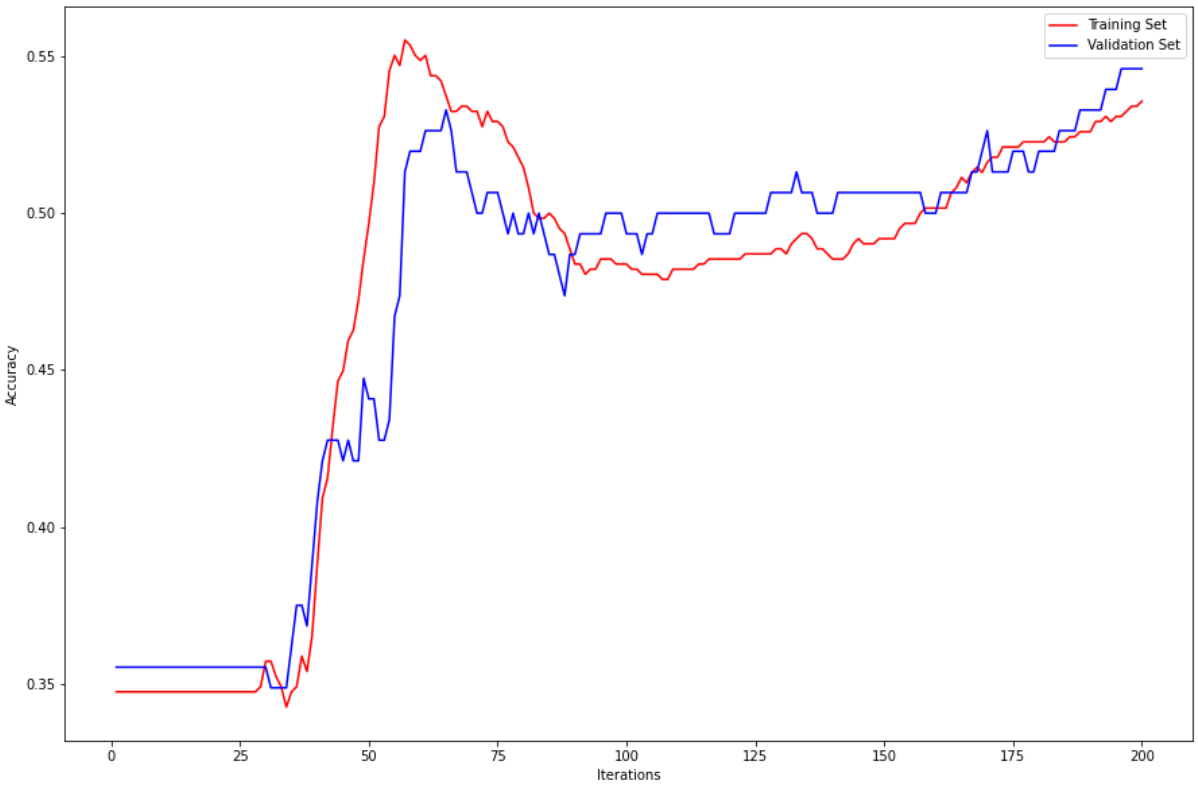

Training Set
Validation Set
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
−8000
Loss
0
25
50
75
100
125
150
175
200
Iterations

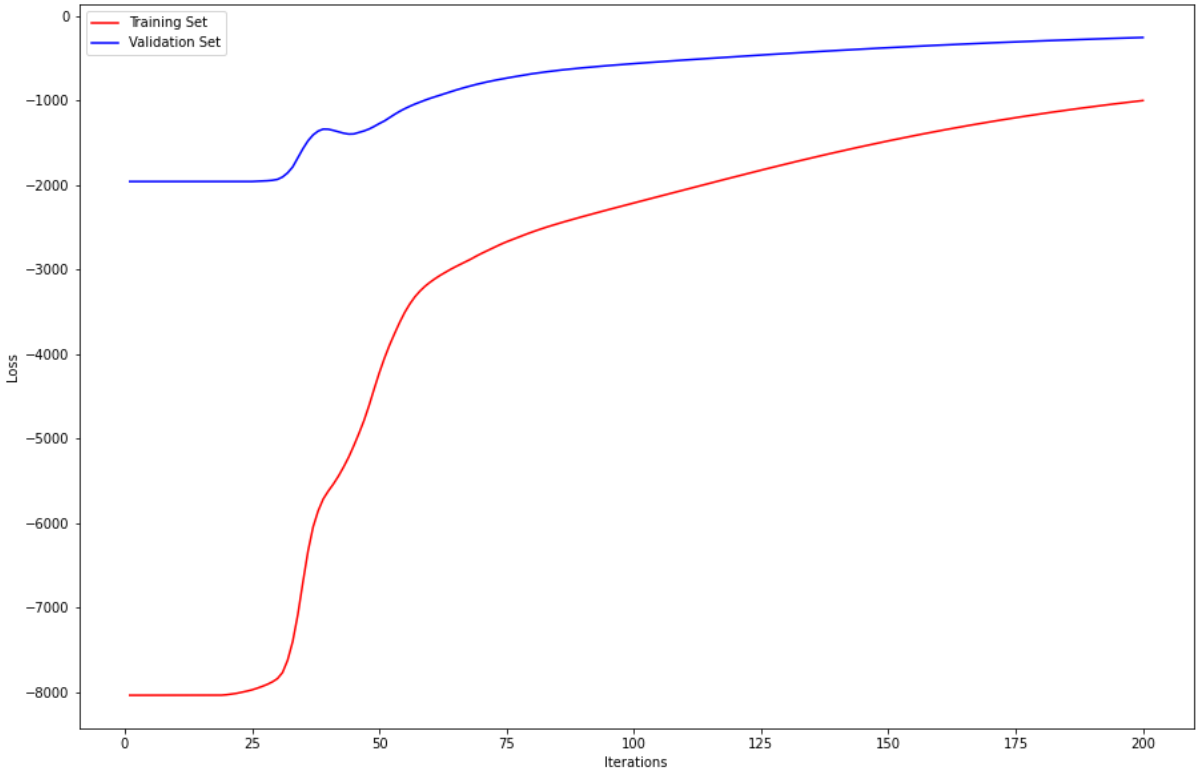
Fold 2


Training Set
Validation Set
0.55
0.50
0.45
0.40
0.35
Accuracy
0
25
50
75
100
125
150
175
200
Iterations

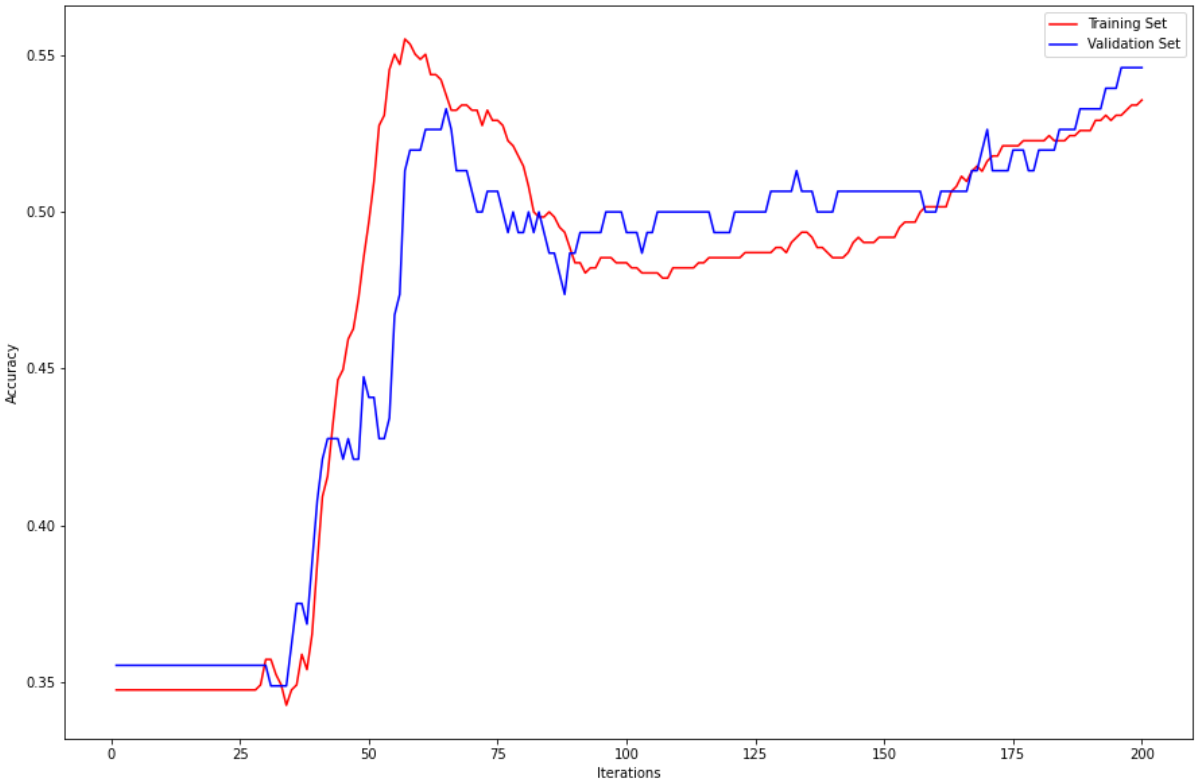

Training Set
Validation Set
Loss
Iterations
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
−8000
0
25
50
75
100
125
150
175
200

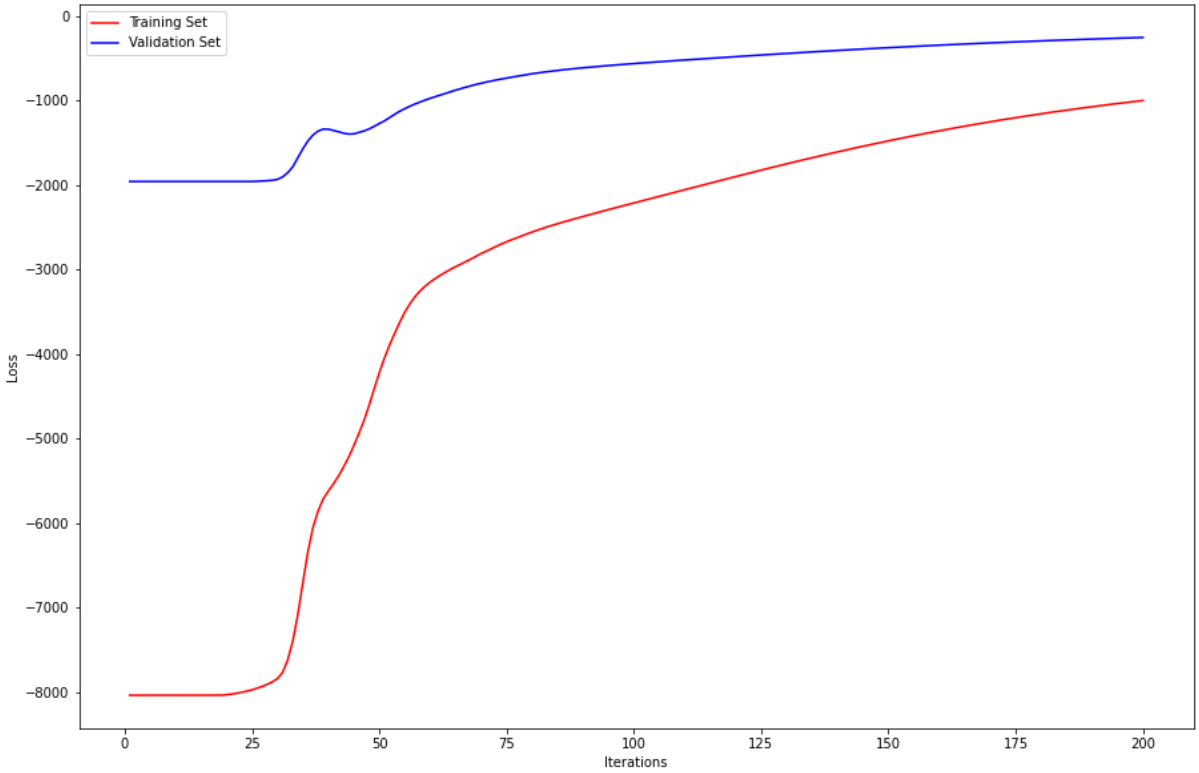
Fold 3


Training Set
Validation Set
Accuracy
Iterations
0.55
0.50
0.45
0.40
0.35
0
25
50
75
100
125
150
175
200

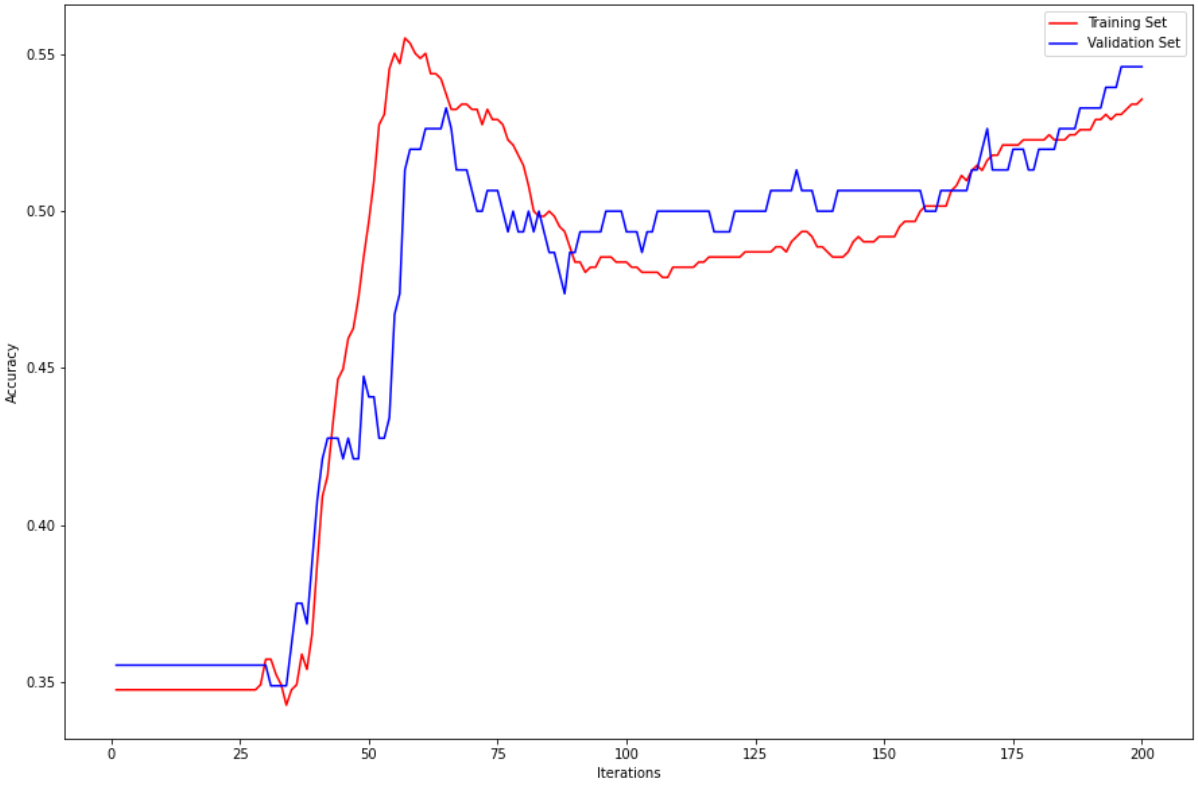

Training Set
Validation Set
Loss
Iterations
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
−8000
0
25
50
75
100
125
150
175
200

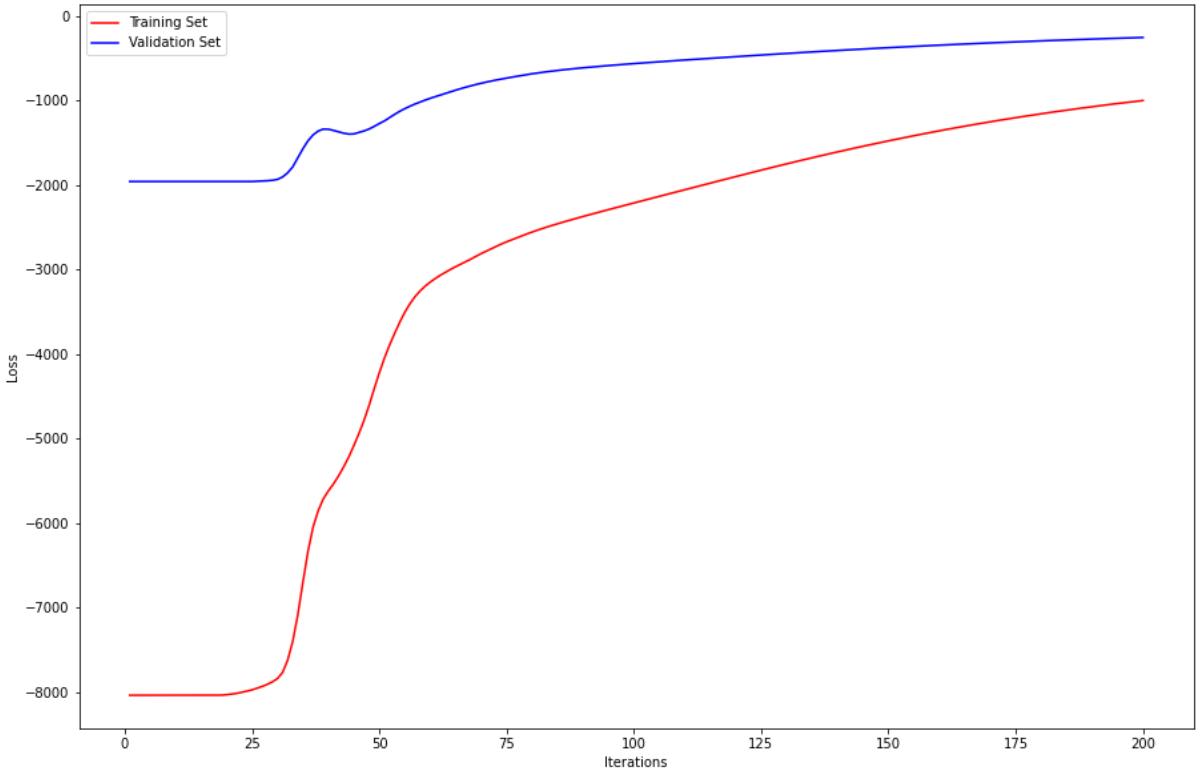
Fold 4


Training Set
Validation Set
Accuracy
Iterations
0.55
0.50
0.45
0.40
0.35
0
25
50
75
100
125
150
175
200


Training Set
Validation Set
Loss
Iterations
0
−1000
−2000
−3000
−4000
−5000
−6000
−7000
−8000
0
25
50
75
100
125
150
175
200

Fold 5


Training Set
Validation Set
Accuracy
Iterations
0.55
0.50
0.45
0.40
0.35
0
25
50
75
100
125
150
175
200



There is deviation in the performance without regularization is giving slight better accuracy than regularized

e.
logistic regression from the sklearn

Performance Of Accuracy is as:

Folds	Training Accuracy	Validation Accuracy
1	0.7817589576547231	0.7727272727272727
2	0.7899022801302932	0.7207792207792207
3	0.7768729641693811	0.7662337662337663
4	0.760586319218241	0.8311688311688312
5	0.7873376623376623	0.7697368421052632

Mean Training Accuracy: 0.7792916367020603

Mean Validation Accuracy: 0.7721291866028708

Accuracy Performance using regularization on training and validation data using sklearn logistic regression

Folds	Training Accuracy	Validation Accuracy
1	0.7736156351791531	0.7402597402597403
2	0.7882736156351792	0.7337662337662337
3	0.7703583061889251	0.7792207792207793

4	0.755700325732899	0.8181818181818182
5	0.7711038961038961	0.7631578947368421

Mean Training Accuracy: 0.7718103557680105

Mean Validation Accuracy: 0.7669172932330828

Sklearn logistic has different performance than c and d. It is giving better results compared to them.

Q3

b. One Vs One

Accuracy Performance of folds:

Folds	Training Accuracy	Validation Accuracy
1	0.9358214285714286	0.9268571428571428
2	0.93875	0.9262857142857143
3	0.9368214285714286	0.9222857142857143
4	0.9412142857142857	0.9234285714285714
5	0.9353035714285715	0.9352142857142857

Mean Training Accuracy: 0.9375821428571429

Mean Validation Accuracy: 0.9268142857142857

Classwise Accuracy Performance of OVO:

Folds	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9
1	0.97651	0.984	0.930	0.894	0.9539	0.8792	0.9590	0.9413	0.8334	0.8990
2	0.965	0.973	0.9104	0.9181	0.9344	0.86	0.9543	0.9514	0.8930	0.8905
3	0.969	0.963	0.885	0.9220	0.9337	0.8434	0.9424	0.9347	0.9185	0.8946
4	0.982	0.975	0.9278	0.8952	0.9473	0.8683	0.9535	0.9243	0.8846	0.8784
5	0.9708	0.982	0.9356	0.943	0.945	0.876	0.965	0.9318	0.884	0.9077

c. One Vs Rest

Accuracy Performance Foldwise:

Folds	Training Accuracy	Validation Accuracy
1	0.8581964285714285	0.8525714285714285
2	0.7128392857142857	0.7048571428571428
3	0.7888571428571428	0.7725
4	0.6734464285714286	0.6630714285714285
5	0.8216428571428571	0.8372142857142857

Mean Training Accuracy: 0.7709964285714286

Mean Validation Accuracy: 0.7660428571428571

Accuracy Performance For OneVsRest Classwise:

Folds	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9
1	0.9864	0.9446	0.8639	0.7730	0.9189	0.8319	0.8621	0.8759	0.7516	0.7007
2	0.9940	0.9606	0.8554	0.8438	0.8813	0.4184	0.8768	0.8804	0.2397	0.0314
3	0.9949	0.9543	0.8946	0.8120	0.8805	0.4626	0.8727	0.8792	0.1948	0.7074
4	0.9963	0.9681	0.8669	0.8095	0.8851	0.0532	0.8717	0.8614	0.2035	0.0177
5	0.9905	0.9674	0.8909	0.8966	0.9062	0.4253	0.8872	0.8705	0.8783	0.6032

d.
Yes, there is performance difference when we use sklearn logistic regression as it performs better than c and b implementations in terms of accuracy
OneVsOne performs better than OneVsRest

Below table give the analysis of mean accuracy for b,c and d for ovo and ovr:

Approach	Training Accuracy Mean	Validation Accuracy Mean
OneVsOne **(b part)**	0.9375821428571429	0.9268142857142857
OneVsRest. **(c part)**	0.7709964285714286	0.7660428571428571
Sklearn OneVsRest Logistic Regression. **(d part)**	0.9274357142857141	0.9133714285714285
Sklearn Logistic Regression OneVsOneClassifier. **(d part)**	0.9759	0.9224