CSE574 Introduction to Machine Learning
Programming Assignment 3
Classification and Regression
(Logistic Regression and Support Vector Machines)

PRIYA MURTHY: 50248887
SHEFALI SHARMA: 50247677

Problem 1.1: Implementation of Logistic Regression

The figure below shows the accuracy for Training set, Validation set and Testing set accuracy obtained for logistic regression.

Logistic Regression

Training set Accuracy	Validation set Accuracy	Testing set Accuracy
84.958	83.73	84.2

Time taken to execute: 959.646082162857 seconds

We implemented Logistic Regression using one vs all strategy, by building 10 binary-classifiers (one for each class) to distinguish a given class from all other classes.

Conclusions:

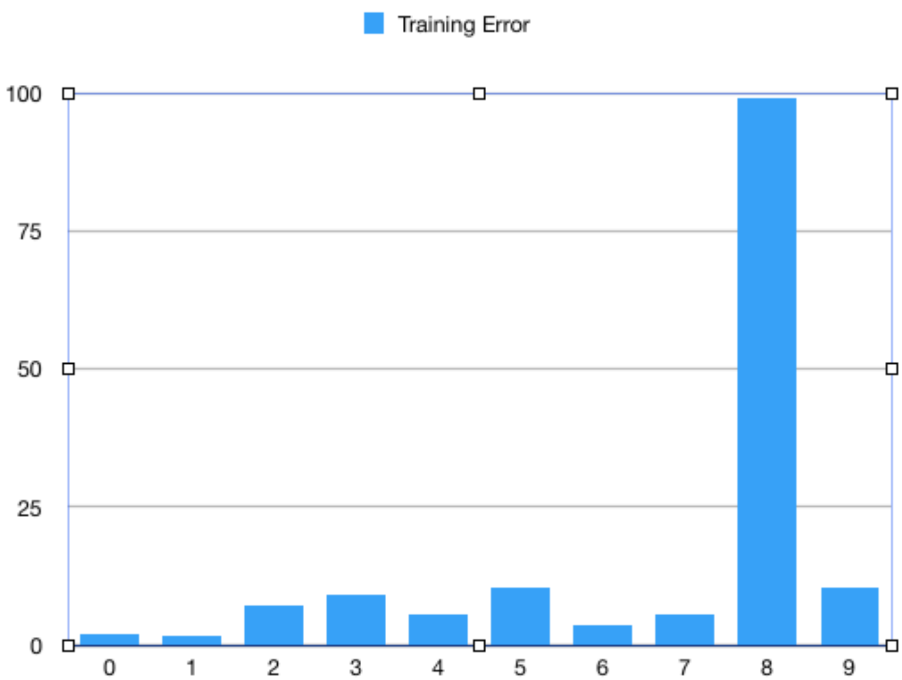
- We can see that the time taken is more because we perform optimization 10 times, once for each digit.
- In case of Logistic Regression, since it is a non-convex function, we may obtain many local optimum, and many local minima mean that gradient descent may not find the global optimum and may get stuck in global minimas.
- This could be one of the reasons due to which the testing set accuracy is low!

Logistic Regression – Category-wise Error
Category wise error has been shown below for each label 0-9 using Confusion Matrix

Training set Accuracy:84.958%

Logistic Regression - Training Error

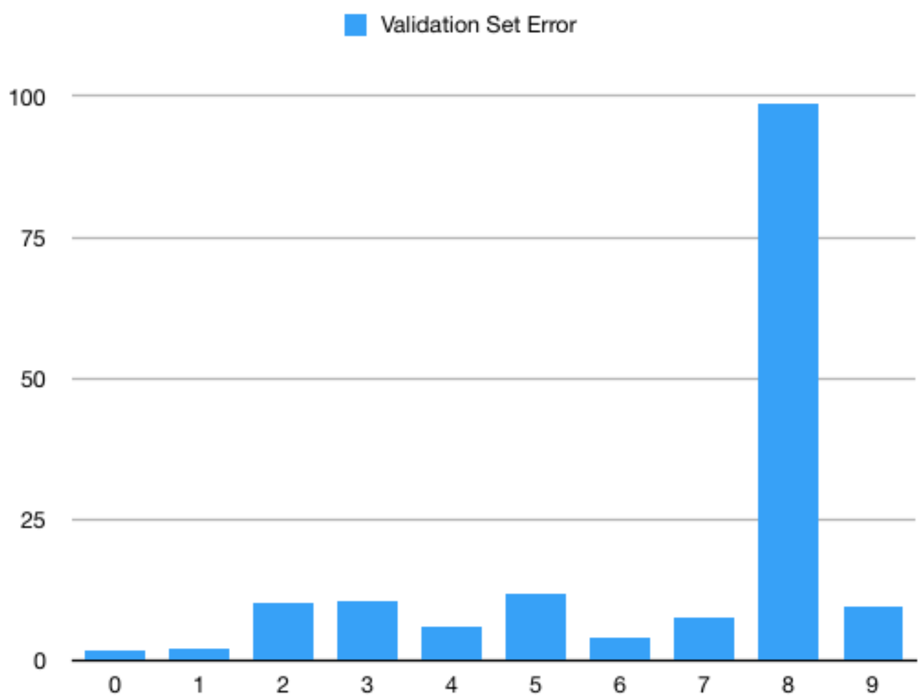
	0	1	2	3	4	5	6	7	8	9	Training Error
0	4830	1	15	9	9	19	29	5	2	4	1.88909
1	2	5654	31	13	3	18	2	10	2	7	1.53257
2	35	30	4607	71	48	30	52	68	1	16	7.07947
3	21	22	132	4663	10	146	21	45	1	70	9.12103
4	8	20	25	5	4567	13	27	15	0	162	5.67947
5	44	17	36	136	50	3961	93	18	8	58	10.4049
6	26	13	31	2	20	69	4748	3	3	3	3.45669
7	11	17	47	11	44	13	4	497	0	145	5.54606
8	137	249	847	1008	193	1319	120	56	44	878	99.093
9	28	19	15	90	162	45	1	157	0	4432	10.4466



Validation set Accuracy:83.73%

Logistic Regression - Validation Set

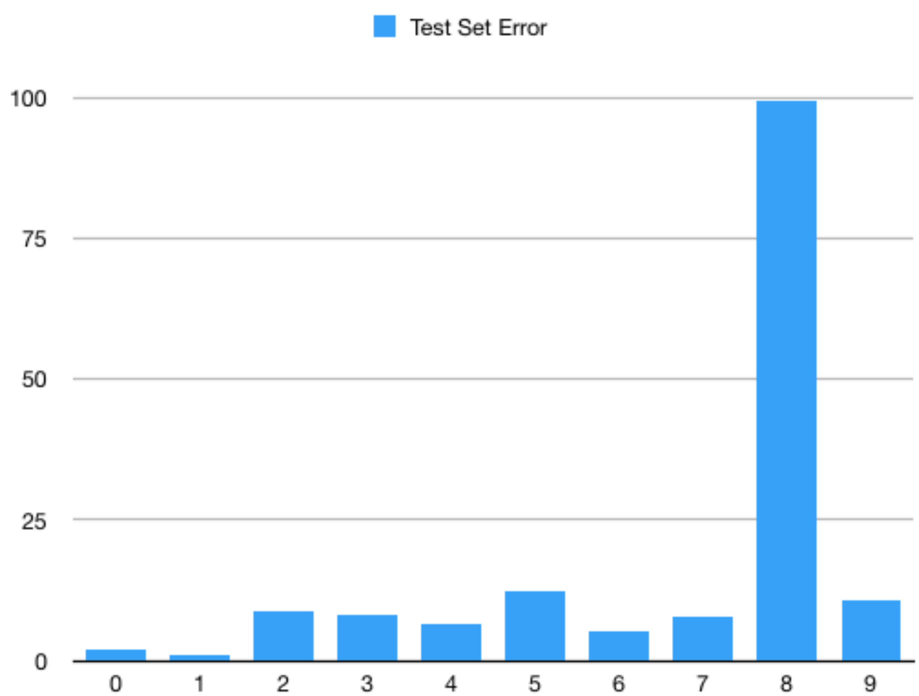
	0	1	2	3	4	5	6	7	8	9	Validation Set Error
0	981	0	1	3	0	6	6	1	0	2	1.9
1	0	978	5	3	2	9	0	1	0	2	2.2
2	11	14	897	20	14	5	14	16	2	7	10.3
3	4	10	34	894	3	24	4	12	0	15	10.6
4	1	7	7	2	941	2	8	0	0	32	5.9
5	8	9	8	43	19	882	19	2	0	10	11.8
6	7	4	6	0	6	14	959	1	1	2	4.1
7	2	5	10	1	15	2	0	924	0	41	7.6
8	32	68	201	223	37	240	41	6	12	140	98.8
9	10	3	5	21	23	5	1	27	0	905	9.5



Testing set Accuracy:84.2%

Logistic Regression - Test Set

	0	1	2	3	4	5	6	7	8	9	Test Set Error
0	961	0	1	3	1	5	6	2	0	1	1.93878
1	0	1124	4	1	0	1	4	1	0	0	0.969166
2	9	14	943	19	12	4	14	13	0	4	8.62403
3	4	0	20	929	3	23	4	14	0	13	8.0198
4	1	2	5	3	918	0	11	2	1	39	6.51731
5	11	3	4	41	11	782	22	9	0	9	12.3318
6	8	4	8	2	4	22	909	1	0	0	5.11482
7	2	10	22	7	8	2	1	948	0	28	7.7821
8	35	34	146	201	55	257	44	20	4	178	99.5893
9	10	8	1	17	35	14	1	21	0	902	10.6046



Problem 1.2: Multi-class Logistic Regression

The figure below shows the accuracy for Training set, Validation set and Testing set accuracy obtained for multi-class logistic regression.

Multi-class Logistic Regression

	Training set Accuracy	Validation set Accuracy	Testing set Accuracy
	93.448	92.48	92.55

Time taken to execute: 60.527652740478516 seconds

We implemented multi-class logistic regression. In this case, unlike in Logistic regression (one vs all strategy) we now only built 1 classifier that can classify 10 classes at the same time.

Conclusions:

- As we can see from above results, multi-class Logistic regression takes less time to execute than Logistic regression and also gives a better accuracy for all the three sets.
- Why is there a difference between training and test error? One of the possible reasons for this difference could be because we have not performed regularization. Though there is not a large difference between the training set and test set accuracy, but since we have not performed regularization, the testing set accuracy is a little low.

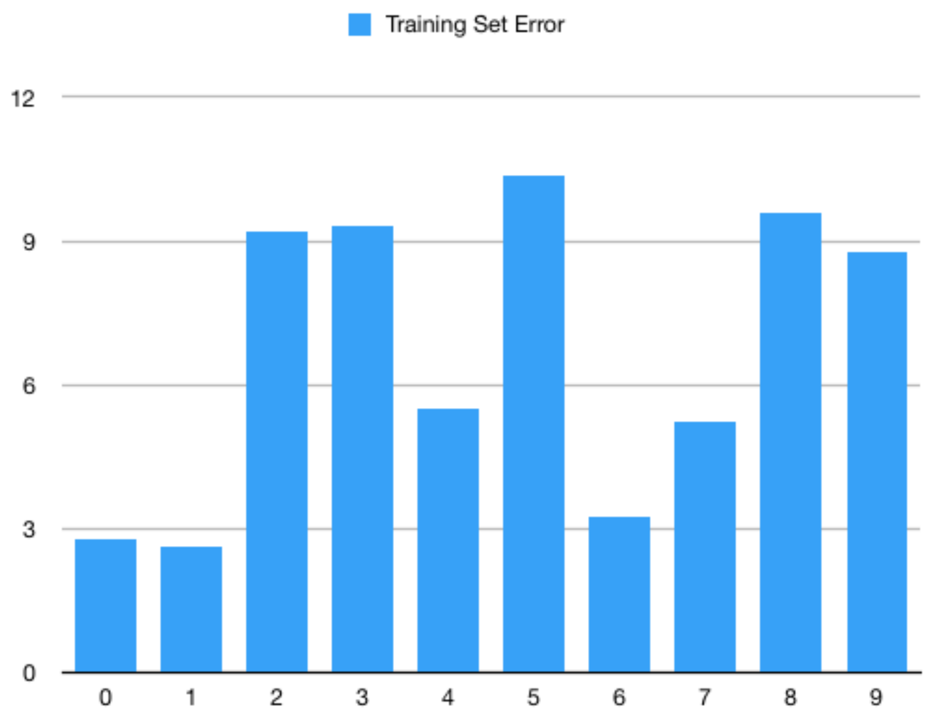
Multiclass Logistic Regression – Category-wise Error

Category wise error has been shown below for each label 0-9 using Confusion Matrix

Training set Accuracy:93.448%

Multi Logistic Regression - Training Set

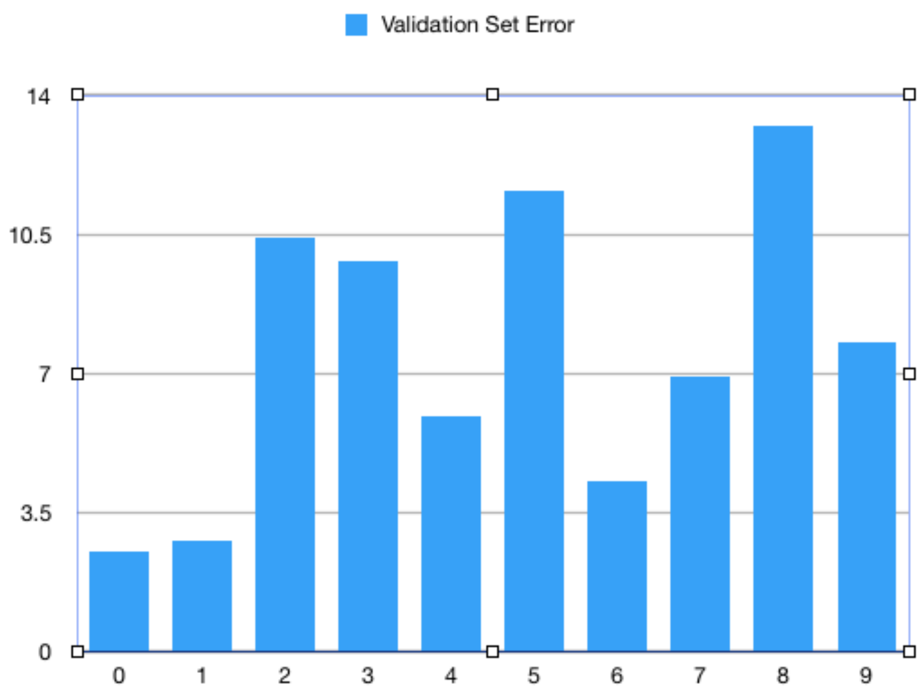
	0	1	2	3	4	5	6	7	8	9	Training Set Error
0	4786	1	12	7	11	33	30	7	32	4	2.78286
1	1	5592	26	17	6	19	2	13	58	8	2.61233
2	23	45	4503	72	58	24	59	53	108	13	9.17709
3	14	18	95	4654	4	148	15	39	105	39	9.29644
4	8	20	21	7	4576	6	42	13	24	125	5.4936
5	39	13	36	117	34	3963	68	18	102	31	10.3596
6	23	11	29	1	24	52	4758	2	16	2	3.25335
7	8	16	49	18	34	9	4	498	14	124	5.24216
8	22	75	51	103	16	113	23	16	4387	45	9.56504
9	17	18	9	55	126	30	2	134	42	4516	8.74924



Validation set Accuracy:92.48%

Multi Logistic Regression - Validation Set

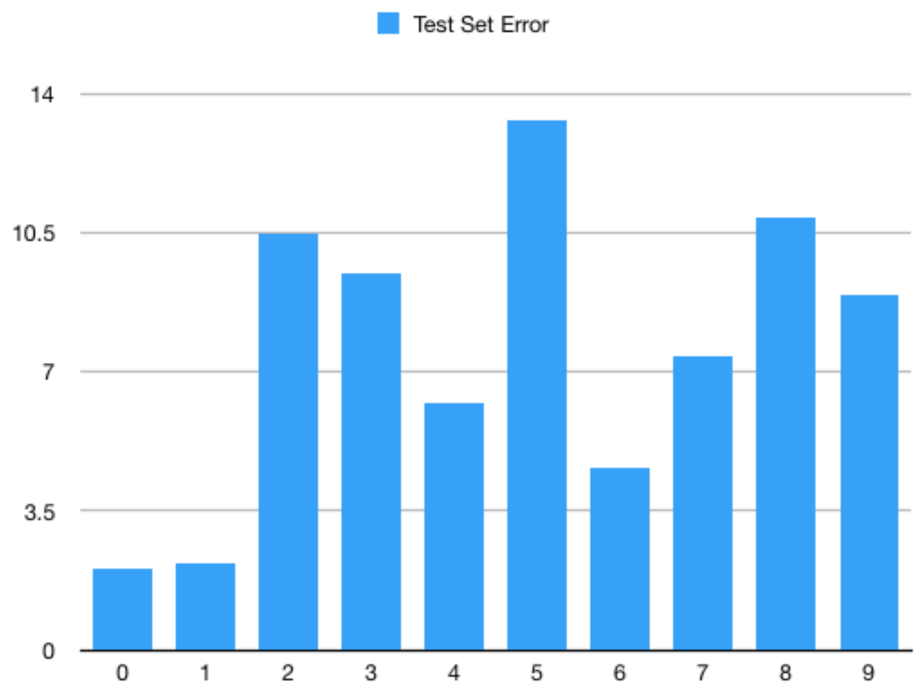
	0	1	2	3	4	5	6	7	8	9	Validation Set Error
0	975	0	1	3	2	7	3	2	6	1	2.5
1	0	972	3	2	1	5	0	2	13	2	2.8
2	10	13	896	22	13	4	11	9	18	4	10.4
3	1	7	23	902	3	28	2	12	13	9	9.8
4	1	4	8	3	941	1	10	2	7	23	5.9
5	9	4	6	37	17	884	14	2	22	5	11.6
6	9	2	4	1	7	12	957	1	6	1	4.3
7	2	3	9	0	9	1	0	931	3	42	6.9
8	13	17	19	27	9	20	19	2	868	6	13.2
9	4	3	5	14	19	4	1	24	4	922	7.8



Testing set Accuracy:92.55%

Multi Logistic Regression - Test Set

	0	1	2	3	4	5	6	7	8	9	Test Set Error
0	960	0	0	3	0	6	6	4	1	0	2.04082
1	0	1110	3	2	0	2	4	2	12	0	2.20264
2	6	8	924	16	10	3	14	8	39	4	10.4651
3	4	1	20	914	0	25	3	10	26	7	9.50495
4	1	1	6	2	921	0	9	4	9	29	6.21181
5	10	2	2	37	10	773	15	6	30	7	13.3408
6	9	3	4	2	7	15	914	3	1	0	4.5929
7	1	9	19	6	6	2	0	952	2	31	7.393
8	9	8	6	26	9	23	10	8	868	7	10.883
9	11	8	0	10	28	5	0	20	8	919	8.91972



Comparison of one-vs-all strategy with multi-class strategy:

COMPARISON PARAMETER	LOGISTIC REGRESSION	MULTI-CLASS LOGISTIC REGRESSION
Performance	Logistic regression is used for binary classification, thus in this case we used one-vs-all strategy to classify 10 digits. Multi class logistic regression performs better than logistic regression.	The performance of multi-class logistic regression is better since global minimum can be obtained in this case.
Time	This takes more time, because the optimization is performed 10 times	In this case, we have built one classifier that can classify 10 classes at the same time. Thus, the time taken to execute Multi-class logistic regression is less as compared to Logistic regression.

Problem 1.3: Support Vector Machines

1.3.1 Linear Kernel (all other parameters kept default):

SVM - Using Linear Kernel

Training set Accuracy	Validation set Accuracy	Testing set Accuracy
97.286	93.64	93.78

Time taken to execute: 754.1166579723358 seconds.

Conclusions:

- Linear SVM is used for solving multiclass classification problems with large data sets and high dimensional data.
- From the observations for the Training set, Validation set and Test set accuracy, we can see that the high accuracy has been obtained, showing that the classes of data are almost linearly separable.

1.3.2 Radial basis function with value of gamma setting to 1 (all other parameters are kept default):

SVM - Radial Basis Function - Gamma = 1

Training set Accuracy	Validation set Accuracy	Testing set Accuracy
100.0	15.48	17.14

Time taken to execute: 72710.8259062767 seconds.

Conclusions:

- We know that the Radial Basis Function kernel expresses similarity between two vector by measuring the distance between the vectors and the Gamma parameter represents the width of the curve.
- In this case, when Gamma = 1, we can see from the accuracies obtained that Gamma was too large.
- Over fitting has occurred due to which the Validation and Test set accuracy is very low, i.e the prediction accuracy is low.

1.3.3 Radial basis function with value of gamma setting to default (all other parameters are kept default):

SVM - Radial Basis Function - Gamma = default

Training set Accuracy	Validation set Accuracy	Testing set Accuracy
94.294	94.02	94.42

Time taken to execute: 1378.1317870616913 seconds.

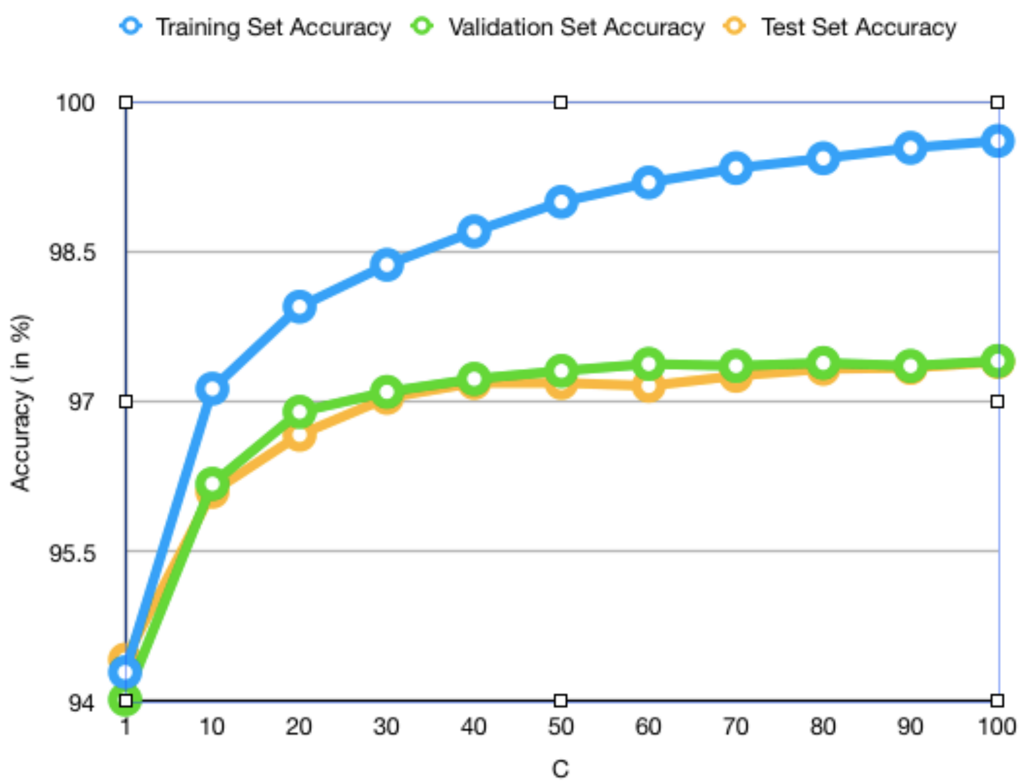
Conclusions:

- In this case, Gamma was set to 0 (default), and the model predicts better i.e the model is not overfit in this case.

1.3.4 Radial basis function with value of gamma setting to default and varying value of C (1; 10; 20; 30; _ _ _ ; 100)

Radial basis function with value of gamma setting to default and varying

C	Training Set Accuracy	Validation Set Accuracy	Test Set Accuracy
1	94.294	94.02	94.42
10	97.132	96.18	96.1
20	97.952	96.9	96.67
30	98.372	97.1	97.04
40	98.706	97.23	97.19
50	99.002	97.31	97.19
60	99.196	97.38	97.16
70	99.34	97.36	97.26
80	99.438	97.39	97.33
90	99.542	97.36	97.34
100	99.612	97.41	97.4



Conclusions:

- In this case, the Gamma is set to zero and the regularization parameter was varied.
- The C parameter is used to avoid misclassification of SVM training sample.
- Smaller value of C, makes the SVM optimizer choose larger margin separating hyperplanes even if there is misclassification.
- Inversely, larger value of C, will ensure the optimizer chooses a smaller hyperplane if more training data points are classified correctly and greater accuracy is obtained.
- From the above observations, we can conclude that with increase in C, the accuracy increases and reaches almost 100 for C = 100.
- Accuracy is least for C = 1.