League of Legends Result Prediction

Final Write-Up

Jay Kang, Amanda Lin

INTRODUCTION

The goal of this machine learning (ML) application is to predict the result of a League of Legends game given some data. A Kaggle data set titled “(LoL) League of Legends Ranked Games” was used for the ML application. The data set contains a collection of over 50,000 ranked EUW League of Legends games. There are a total of 71 features. To simplify the ML application and to decrease the time needed to learn each ML algorithm, only 18 features were used, namely:

- 'gameDuration' – The length of the game in seconds
- 'firstBlood' – The team that got first blood (1 for team 1, 2 for team 2)
- 'firstTower' – The team that got first tower (1 for team 1, 2 for team 2)
- 'firstInhibitor' – The team that got first inhibitor (1 for team 1, 2 for team 2)
- 'firstBaron' – The team that got first baron (1 for team 1, 2 for team 2)
- 'firstDragon' – The team that got first dragon (1 for team 1, 2 for team 2)
- 'firstRiftHerald' – The team that got first rift herald (1 for team 1, 2 for team 2)
- 't1_towerKills' – The number of tower kills team 1 had
- 't1_inhibitorKills' – The number of inhibitor kills team 1 had
- 't1_baronKills' – The number of baron kills team 1 had
- 't1_dragonKills' – The number of dragon kills team 1 had

- 't1_riftHeraldKills' – The number of rift herald kills team 1 had
- 't2_towerKills' – The number of tower kills team 2 had
- 't2_inhibitorKills' – The number of inhibitor kills team 2 had
- 't2_baronKills' – The number of baron kills team 2 had
- 't2_dragonKills' – The number of dragon kills team 2 had
- 't2_riftHeraldKills' – The number of rift herald kills team 2 had’
- ‘winner’ – The winner of the game (0 for team 1, 1 for team 2)

Supervised analysis was used to train the ML application. The three learning models implemented were Logistic Regression, SVM, and Neural Networks. For each model, different feature transformations and regularization techniques were used.

SUPERVISED ANALYSIS

Logistic Regression

- Gradient ascent with sigmoid function
- L1 regularization
- L2 regularization
- Polynomial (degree 2) feature transformation with L1 regularization
- Polynomial (degree 2) feature transformation with L2 regularization
- Random state 30 and L1 regularization

SVM

- SVM linear kernel
- SVM radial kernel
- SVM polynomial kernel

Neural Networks

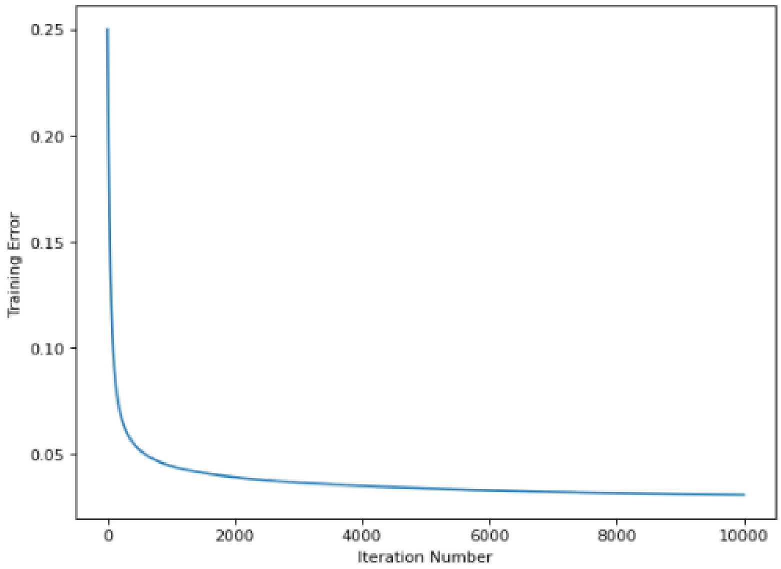
- Sigmoid activation function
- ReLU activation function
- Leaky ReLU activation function

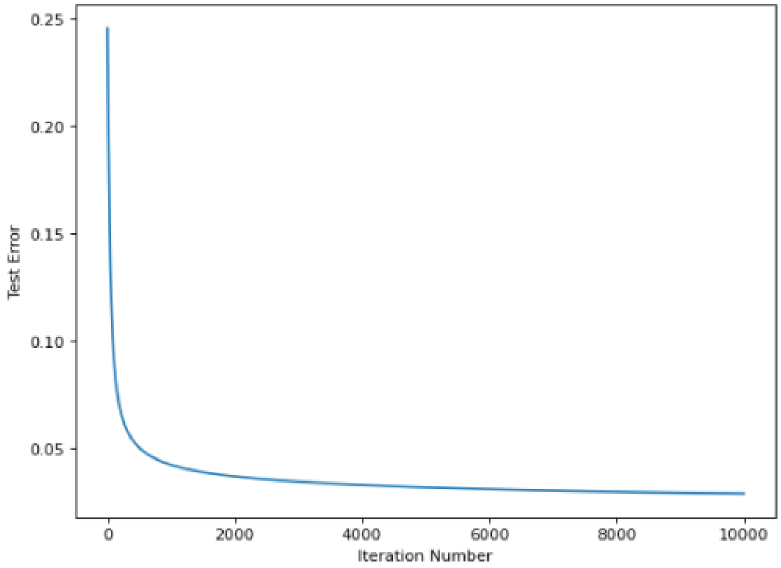
SUPERVISED ANALYSIS RESULT TABLES AND GRAPHS

Logistic Regression

- Gradient ascent and sigmoid function

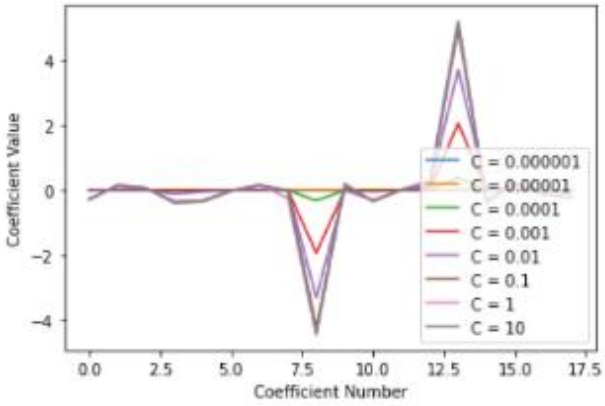
Learning Rate	Iterations	Training Accuracy	Testing Accuracy
0.01	**10,000**	**96.042%**	**96.119%**

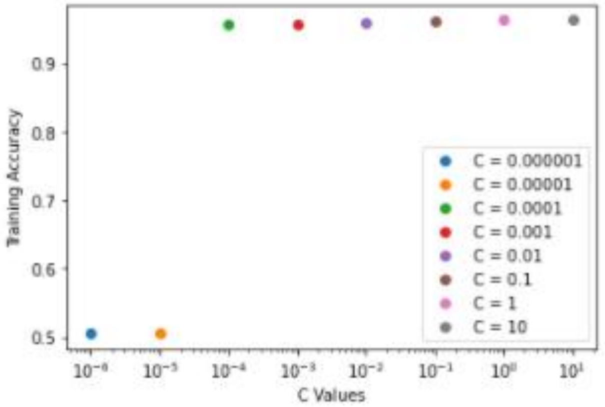


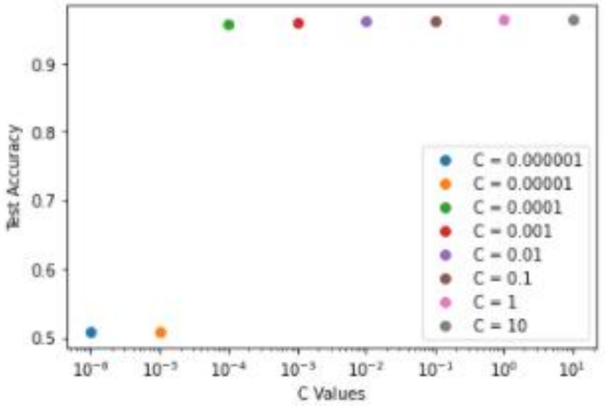


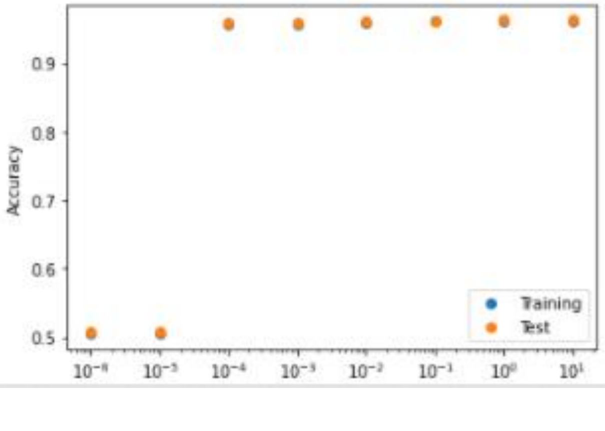
- L1 regularization

cValue	Training Accuracy	Testing Accuracy
0.000001	50.600%	50.781%
0.00001	50.600%	50.781%
0.0001	95.717%	95.852%
0.001	95.720%	95.891%
0.01	95.888%	96.248%
0.1	96.082%	96.256%
1	96.302%	96.326%
10	96.302%	96.318%


Coefficient Value
4
2
0
−2
−4
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Coefficient Number
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

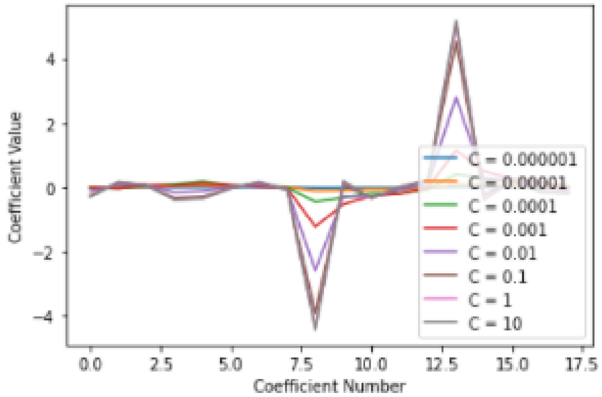

Training Accuracy
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

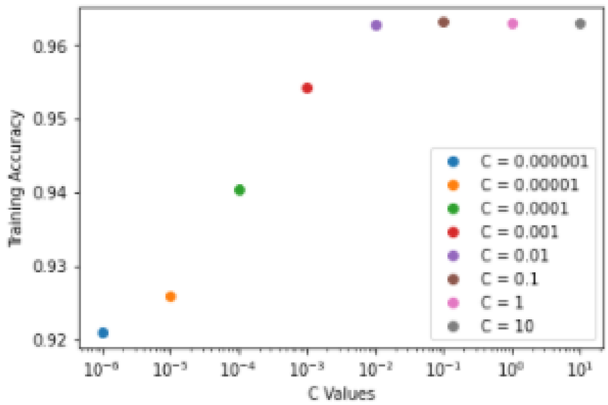

Test Accuracy
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

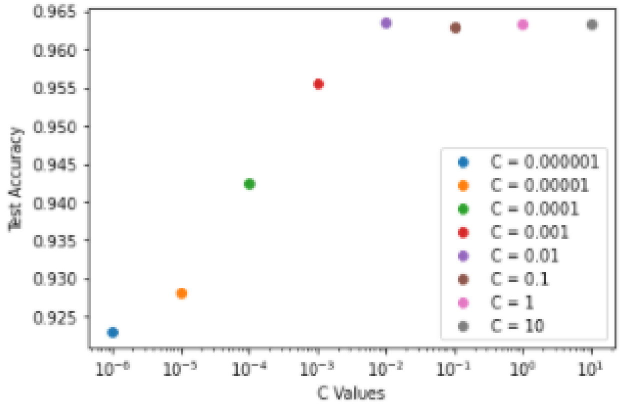

Accuracy
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
Training
Test

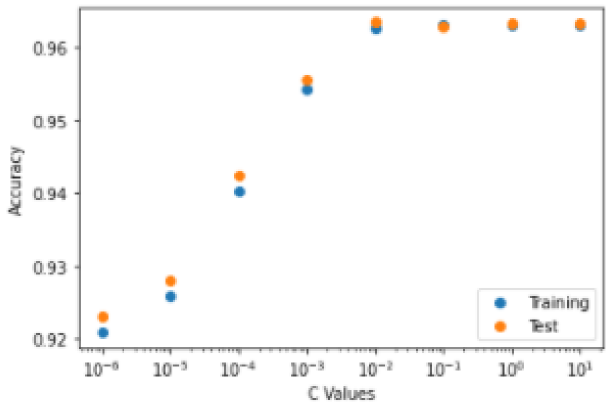
- L2 regularization

cValue	Training Accuracy	Testing Accuracy
0.000001	92.094%	92.302%
0.00001	92.584%	92.807%
0.0001	94.031%	94.236%
0.001	95.419%	95.557%
0.01	96.269%	96.341%
0.1	96.307%	96.295%
1	96.302%	96.326%
10	96.300%	96.333%


Coefficient Value
4
2
0
−2
−4
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Coefficient Number
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

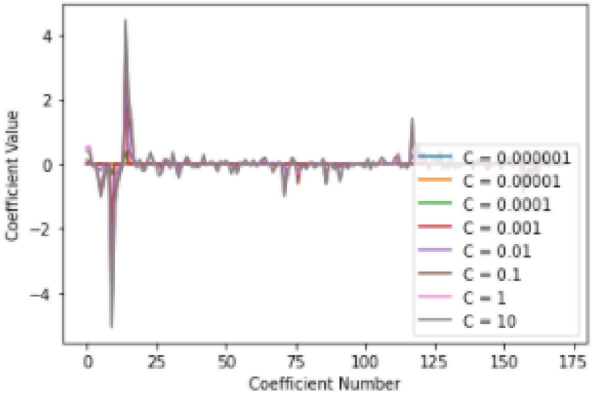

Training Accuracy
0.96
0.95
0.94
0.93
0.92
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

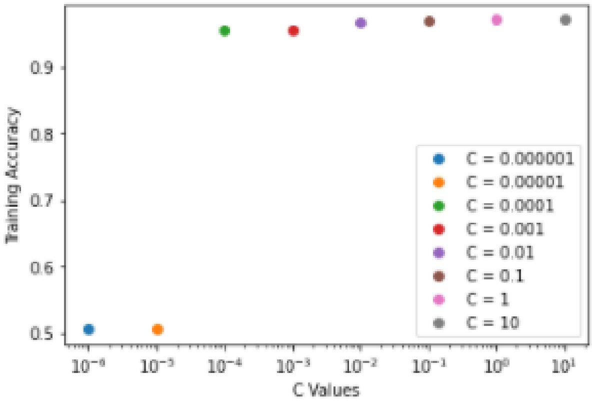

Test Accuracy
0.965
0.960
0.955
0.950
0.945
0.940
0.935
0.930
0.925
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

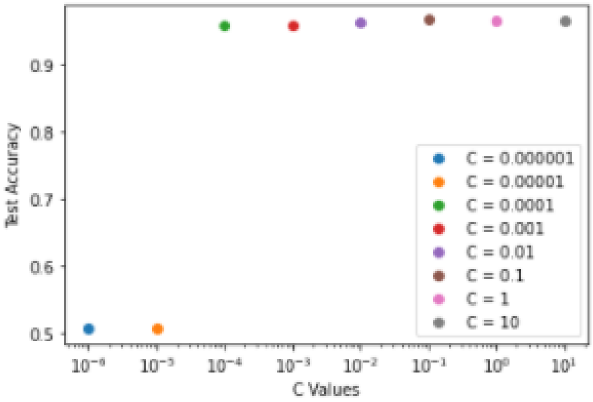

Accuracy
0.96
0.95
0.94
0.93
0.92
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
Training
Test

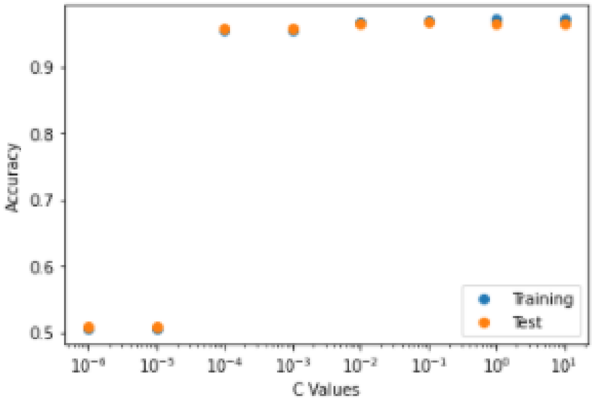
- Polynomial (degree 2) feature transformation with L1 regularization

cValue	Training Accuracy	Testing Accuracy
0.000001	50.600%	50.781%
0.00001	50.600%	50.781%
0.0001	95.717%	95.852%
0.001	95.720%	95.891%
0.01	96.742%	96.450%
0.1	97.069%	96.722%
1	97.149%	96.636%
10	97.157%	96.582%


Coefficient Value
4
2
0
−2
−4
0
25
50
75
100
125
150
175
Coefficient Number
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

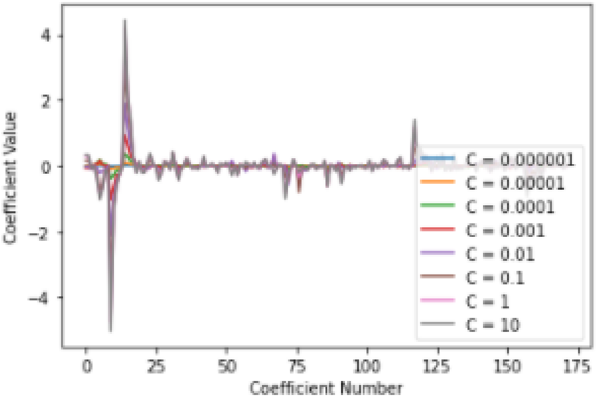

Training Accuracy
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10


Test Accuracy
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

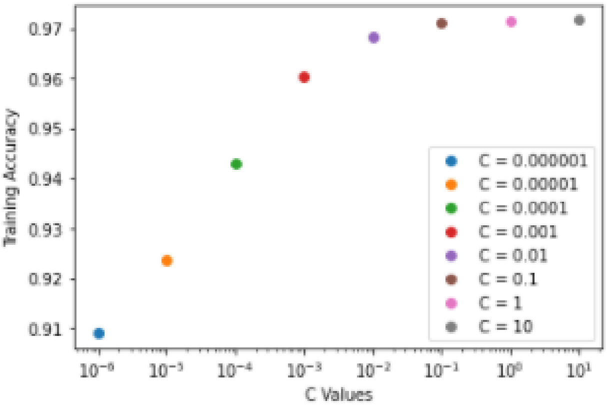

Accuracy
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
Training
Test

- Polynomial (degree 2) feature transformation with L2 regularization

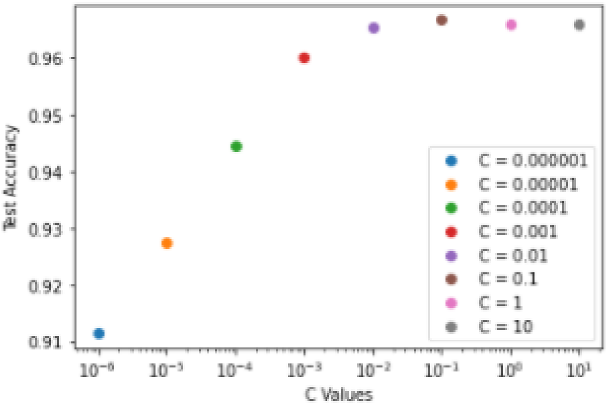
cValue	Training Accuracy	Testing Accuracy
0.000001	90.911%	91.160%
0.00001	92.382%	92.760%
0.0001	94.306%	94.446%
0.001	96.051%	95.999%
0.01	96.823%	96.535%
0.1	97.110%	96.660%
1	97.152%	96.582%
10	97.162%	96.582%



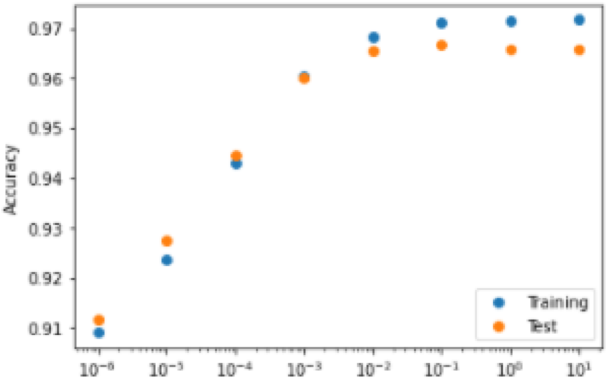
Coefficient Value
4
2
0
−2
−4
0
25
50
75
100
125
150
175
Coefficient Number
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10



Training Accuracy
0.97
0.96
0.95
0.94
0.93
0.92
0.91
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10



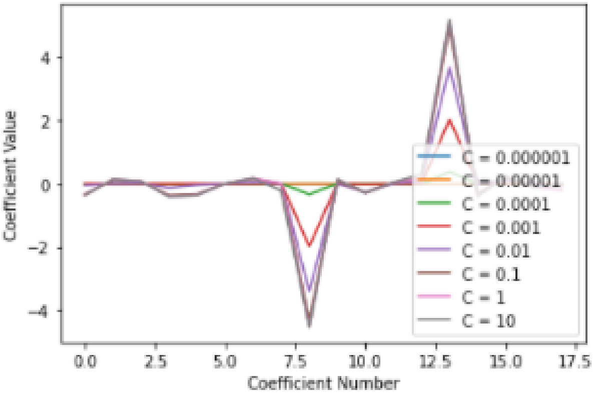
Test Accuracy
0.96
0.95
0.94
0.93
0.92
0.91
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C Values
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

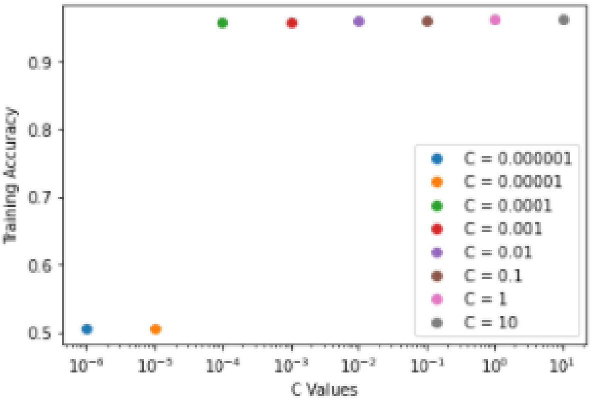


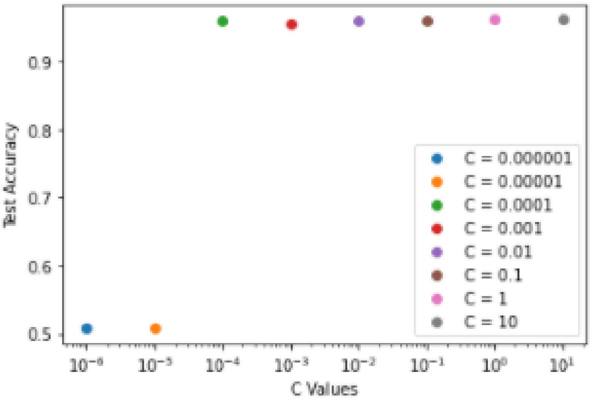
Accuracy
0.97
0.96
0.95
0.94
0.93
0.92
0.91
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
Training
Test

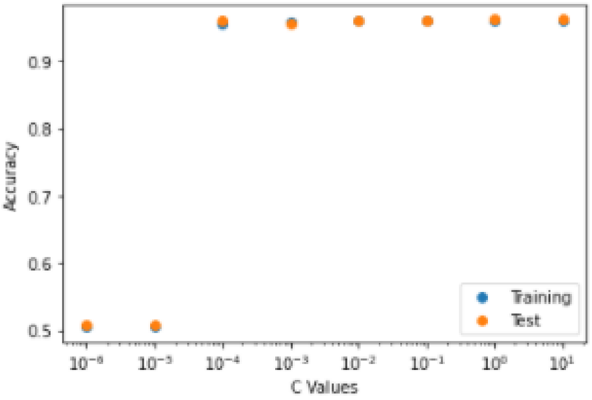
- Random state 30 and L1 regularization

cValue	Training Accuracy	Testing Accuracy
0.000001	50.590%	50.812%
0.00001	50.590%	50.812%
0.0001	95.670%	96.023%
0.001	95.701%	95.673%
0.01	95.968%	96.007%
0.1	96.064%	96.085%
1	96.116%	96.163%
10	96.116%	96.163%


Coefficient Value
Coefficient Number
4
2
0
−2
−4
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10


Training Accuracy
C Values
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

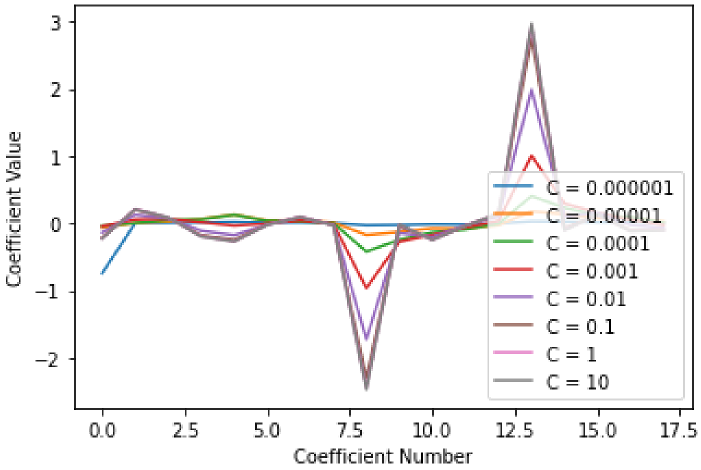

Test Accuracy
C Values
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10


Accuracy
C Values
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
Training
Test

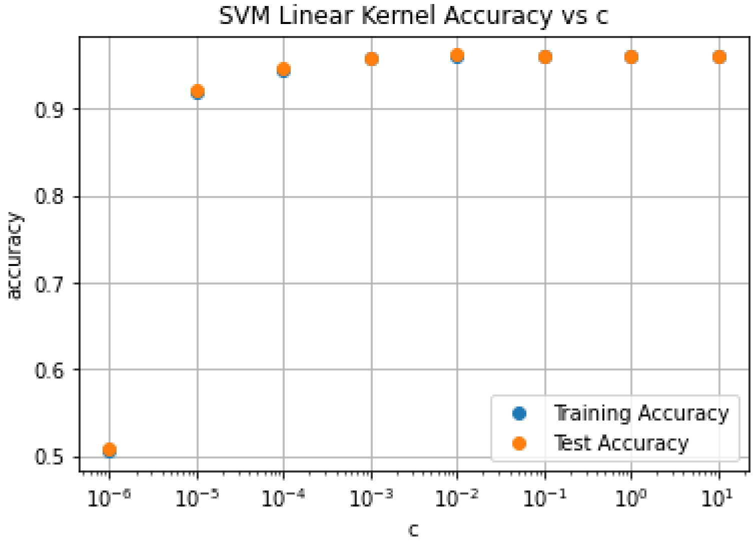
SVM

- SVM linear kernel

cValue	Training Accuracy	Testing Accuracy
0.000001	50.600%	50.781%
0.00001	91.8171%	92.1541%
0.0001	94.4299%	94.6632%
0.001	95.8697%	95.9450%
0.01	96.1131%	96.2091%
0.1	96.0872%	96.1470%
1	96.1002%	96.1159%
10	96.1002%	96.1159%



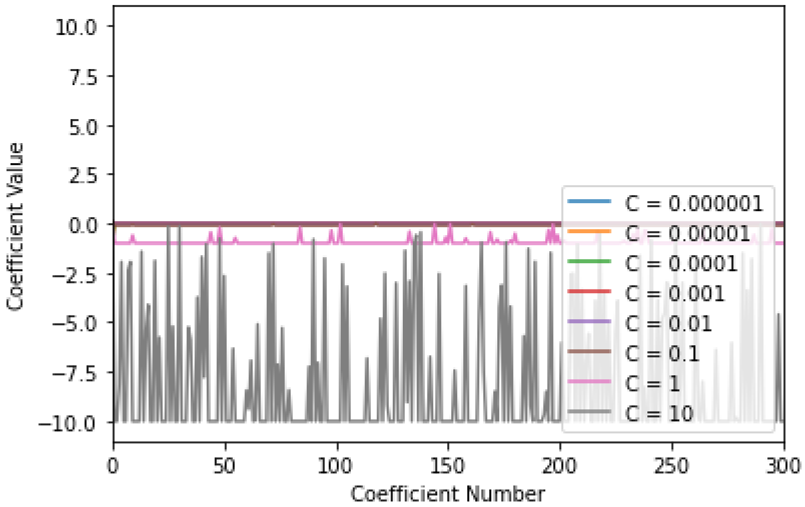
3
2
1
0
−1
−2
Coefficient Value
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Coefficient Number
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

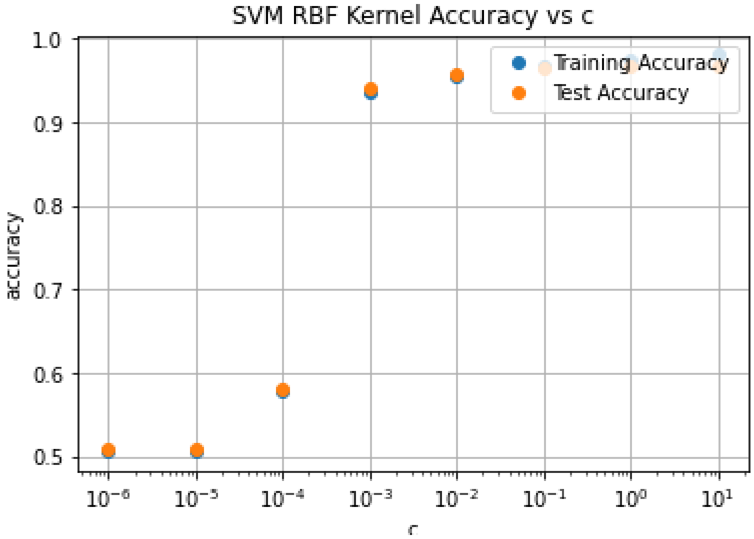


SVM Linear Kernel Accuracy vs c
0.9
0.8
0.7
0.6
0.5
accuracy
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
c
Training Accuracy
Test Accuracy

- SVM radial kernel

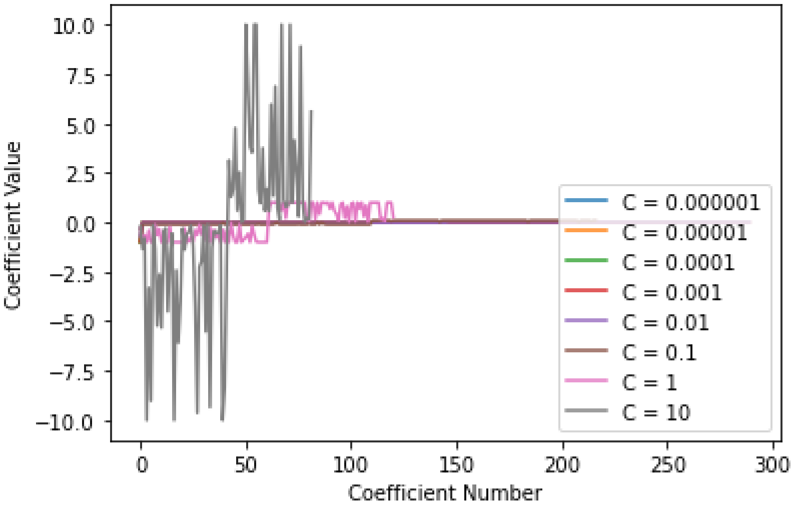
cValue	Training Accuracy	Testing Accuracy
0.000001	**50.5995%**	**50.7807%**
0.00001	**50.5995%**	**50.7807%**
0.0001	**57.8761%**	**58.0828%**
0.001	**93.6531%**	**93.9952%**
0.01	**95.4994%**	**95.6653%**
0.1	**96.8408%**	**96.5898%**
1	**97.4648%**	**96.7374%**
10	**98.0760%**	**96.7451%**

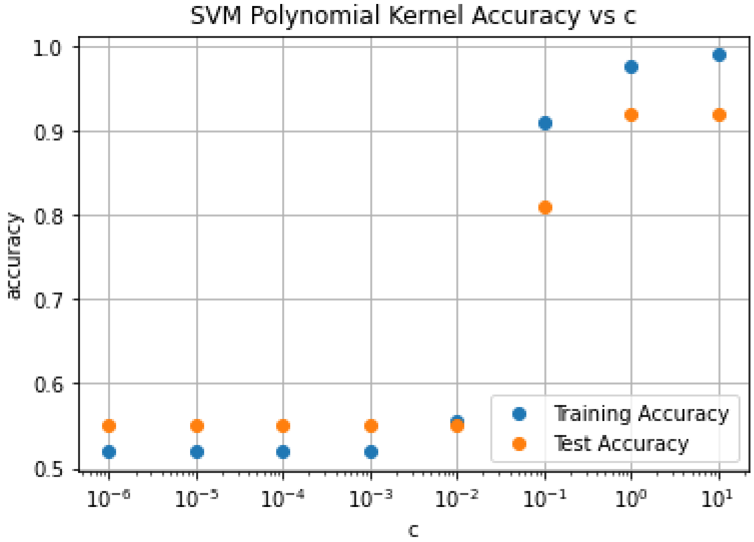



SVM RBF Kernel Accuracy vs c

- SVM polynomial kernel

cValue	Training Accuracy	Testing Accuracy
0.000001	**52.0000%**	**55.0000%**
0.00001	**52.0000%**	**55.0000%**
0.0001	**52.0000%**	**55.0000%**
0.001	**52.0000%**	**55.0000%**
0.01	**55.6667%**	**55.0000%**
0.1	**91.0000%**	**81.0000%**
1	**97.6667%**	**92.0000%**
10	**99.0000%**	**92.0000%**

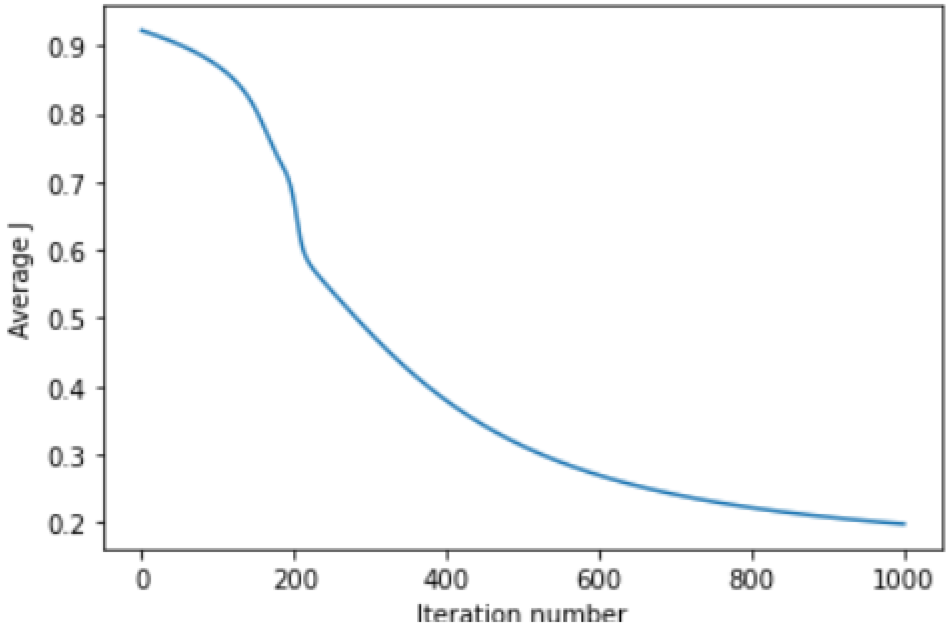

Coefficient Value
Coefficient Number
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
0
50
100
150
200
250
300
C = 0.000001
C = 0.00001
C = 0.0001
C = 0.001
C = 0.01
C = 0.1
C = 1
C = 10

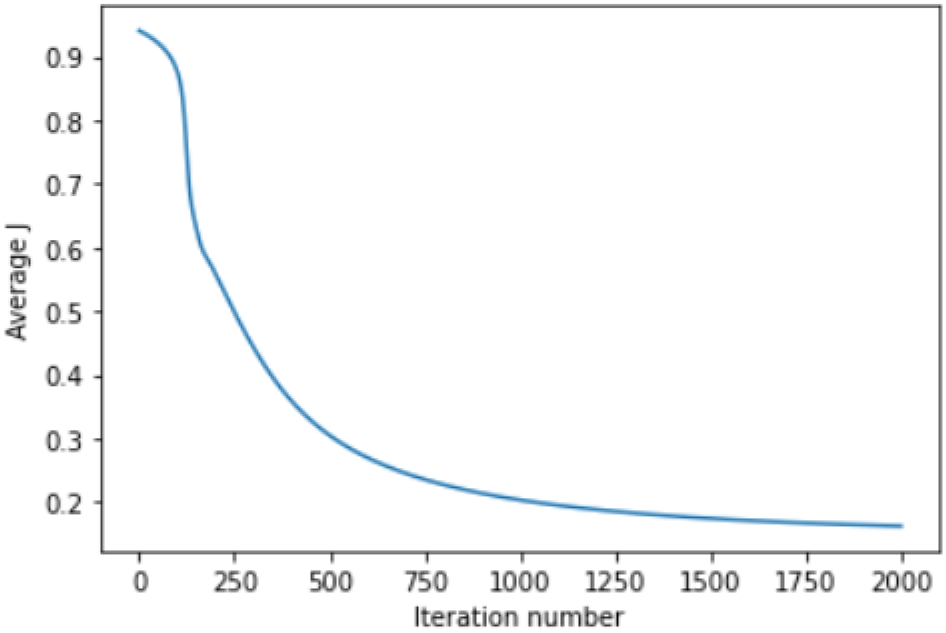

SVM Polynomial Kernel Accuracy vs c
accuracy
c
1.0
0.9
0.8
0.7
0.6
0.5
10^{-6}
10^{-5}
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
Training Accuracy
Test Accuracy

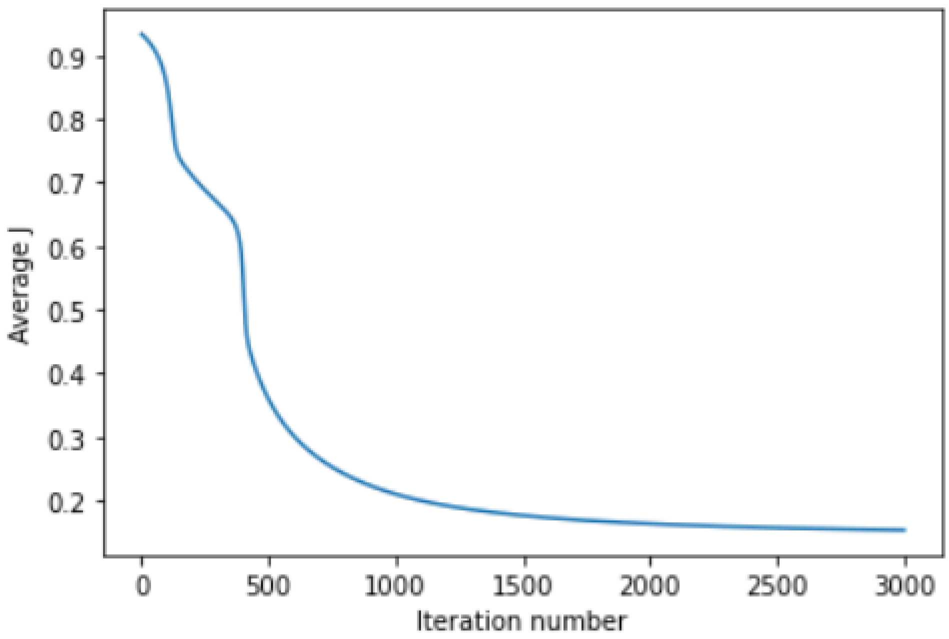
Neural Networks

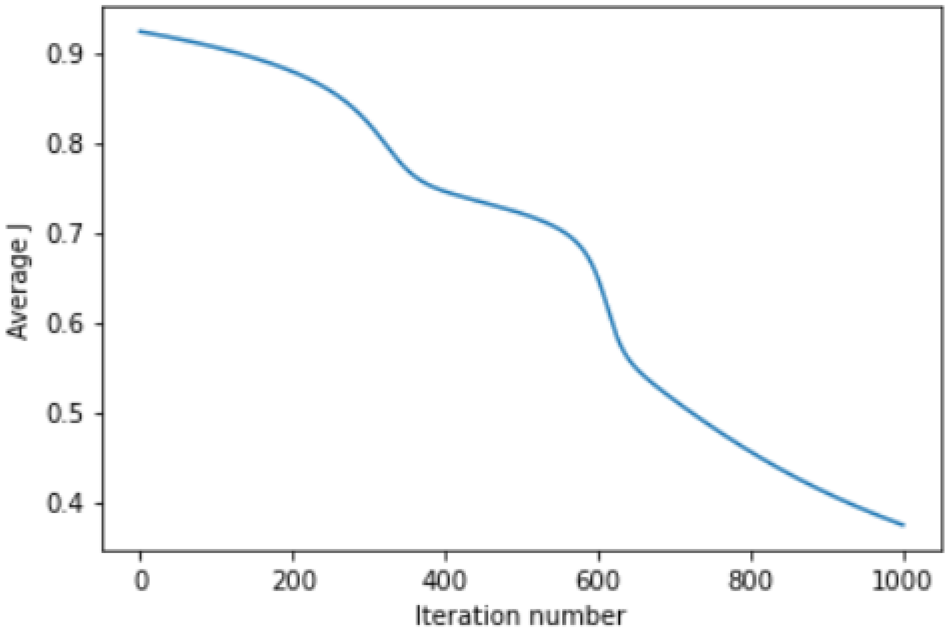
- Sigmoid activation function

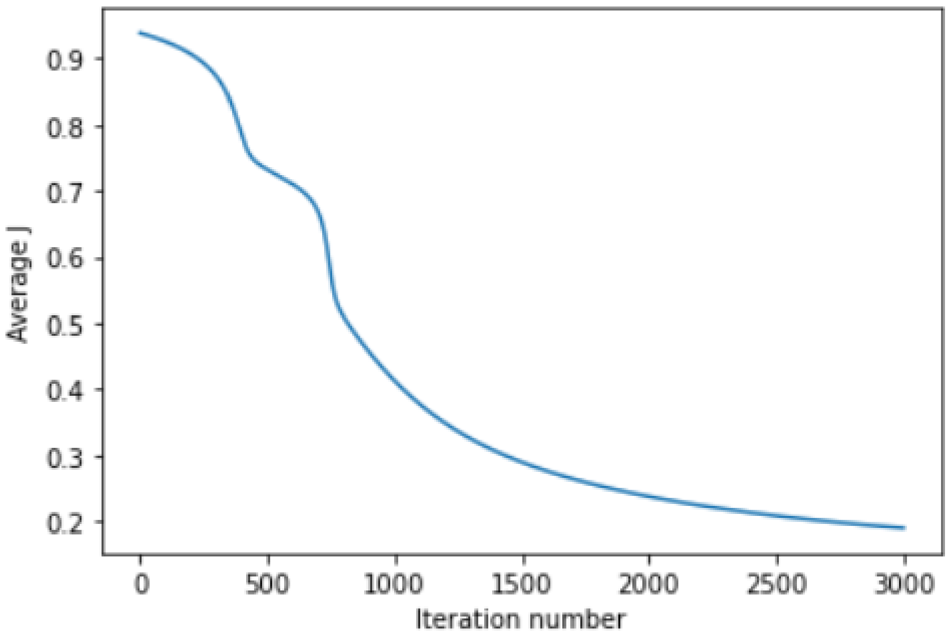
Iterations	Alpha	Lambda	Training Accuracy	Testing Accuracy
1,000	0.25	0.001	95.176%	94.824%
2,000	0.25	0.001	95.851%	95.518%
3,000	0.25	0.001	96.108%	95.914%
1,000	0.1	0.001	89.885%	89.691%
3,000	0.1	0.001	94.958%	94.521%


Average J
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
200
400
600
800
1000
Iteration number


Average J
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
250
500
750
1000
1250
1500
1750
2000
Iteration number

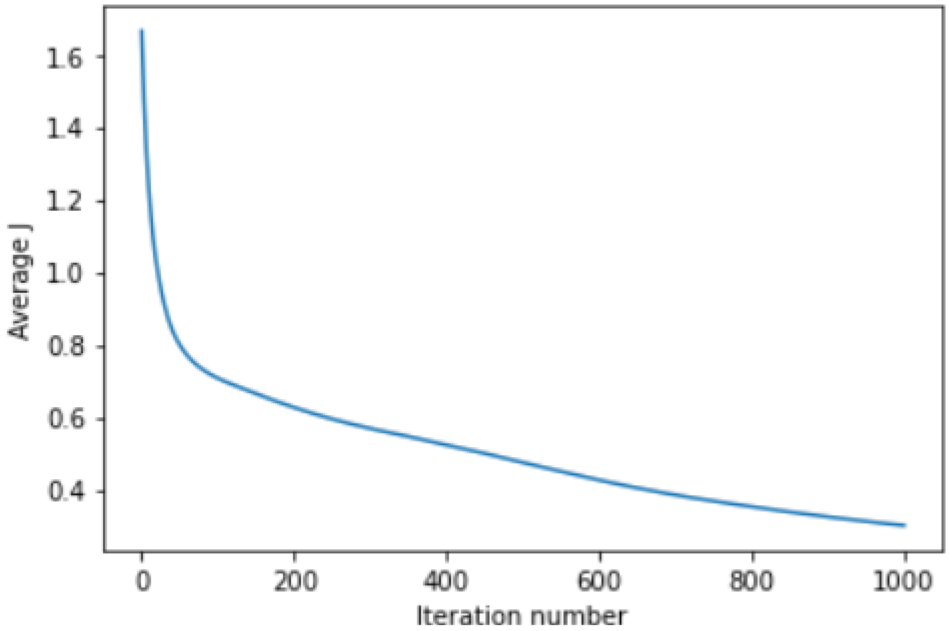

Average J
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
500
1000
1500
2000
2500
3000
Iteration number

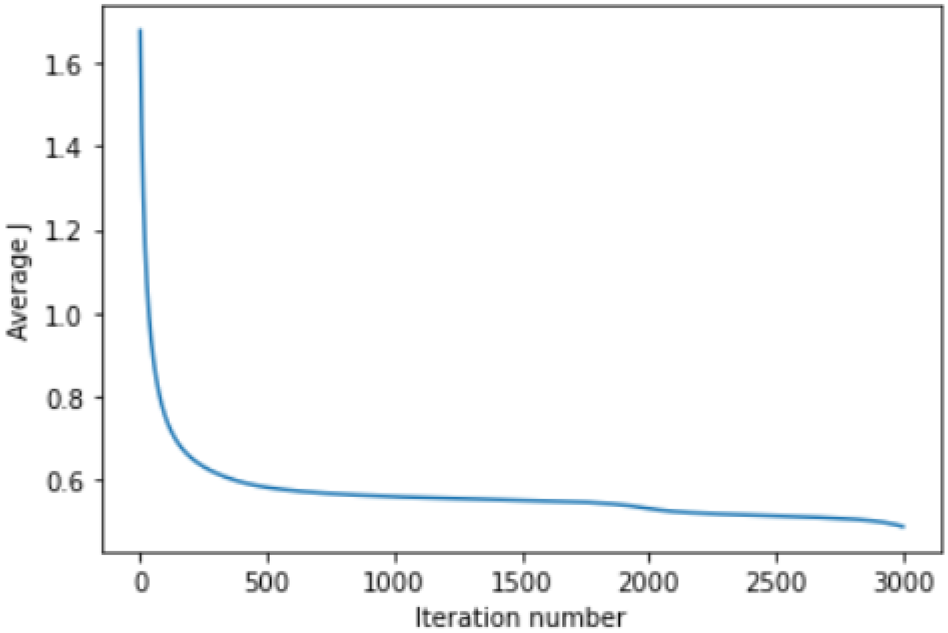

Average J
0.9
0.8
0.7
0.6
0.5
0.4
0
200
400
600
800
1000
Iteration number

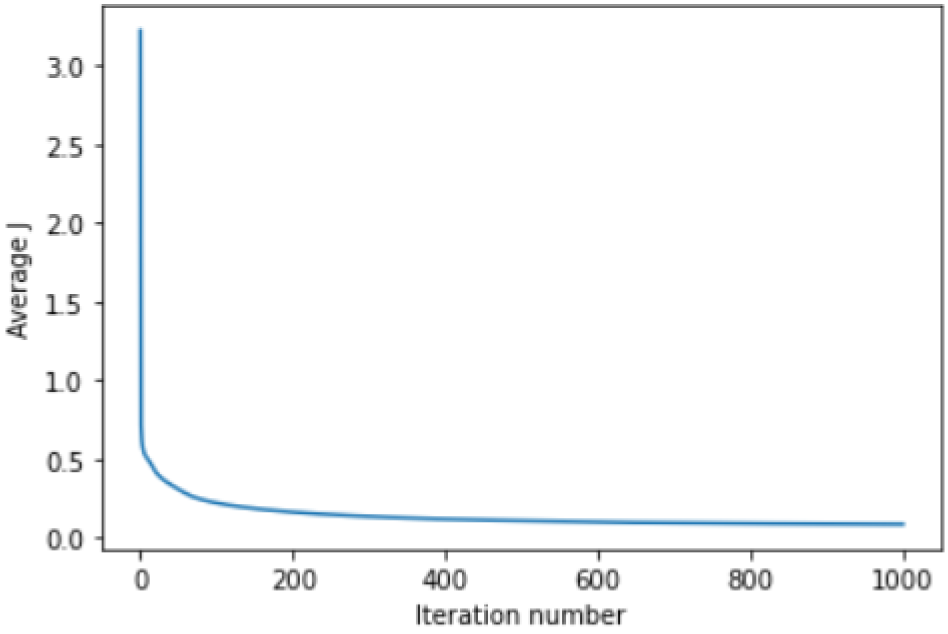

Average J
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
500
1000
1500
2000
2500
3000
Iteration number

- ReLU activation function

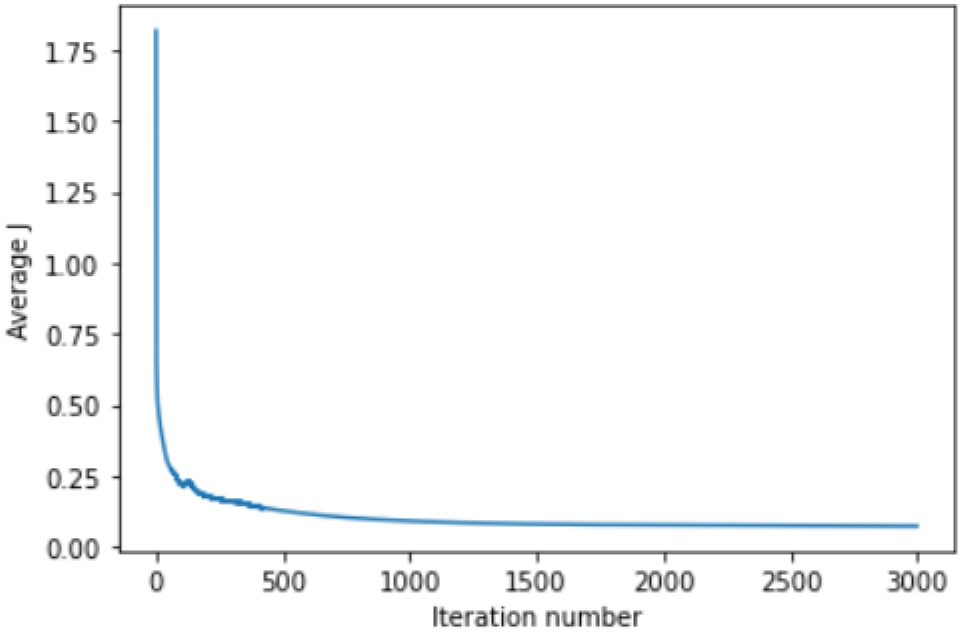
Iterations	**Alpha**	**Lambda**	**Training Accuracy**	**Testing Accuracy**
1,000	**0.01**	**0.001**	**71.644%**	**71.653%**
3,000	**0.01**	**0.001**	**54.748%**	**54.311%**
1,000	**0.25**	**0.001**	**71.131%**	**70.736%**
3,000	**0.25**	**0.001**	**67.550%**	**67.471%**


1.6
1.4
1.2
1.0
0.8
0.6
0.4
Average J
0
200
400
600
800
1000
Iteration number


1.6
1.4
1.2
1.0
0.8
0.6
Average J
0
500
1000
1500
2000
2500
3000
Iteration number



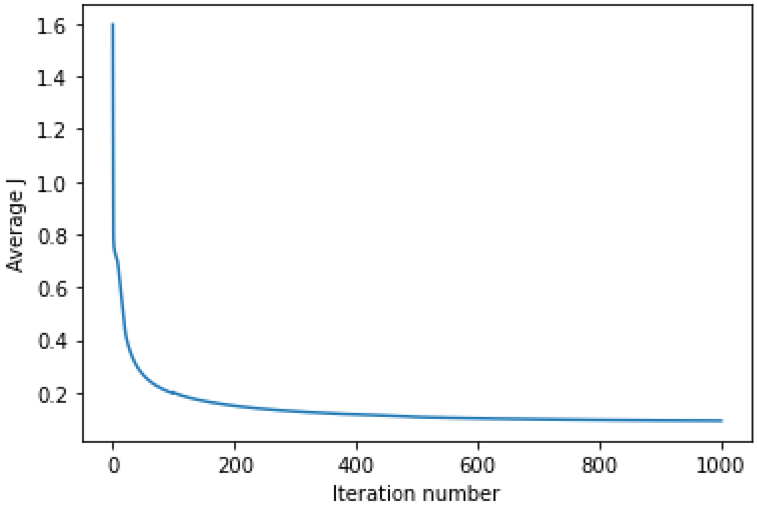
Average J
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
200
400
600
800
1000
Iteration number

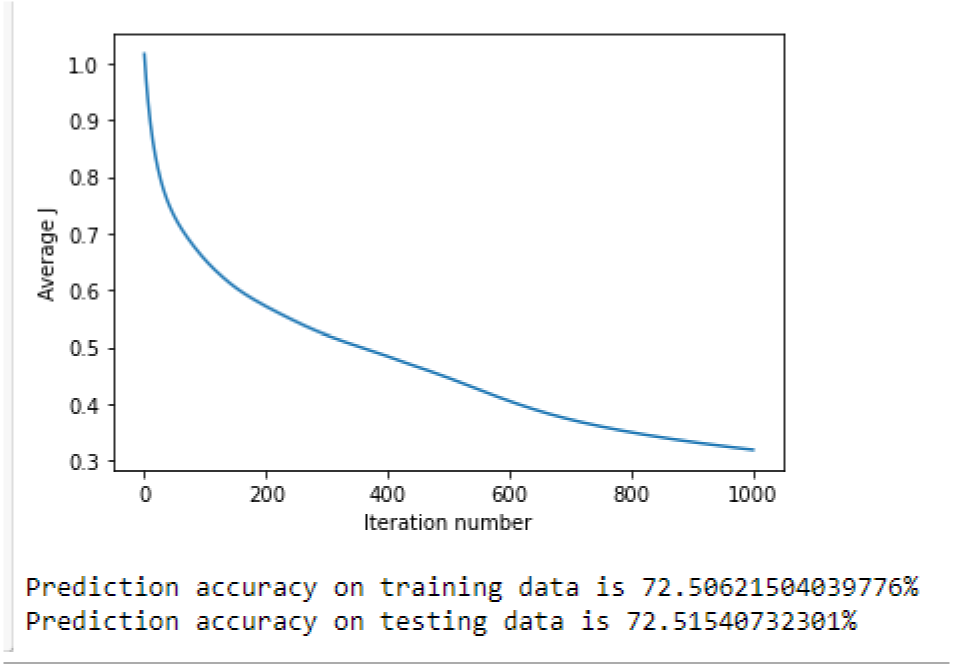


Average J
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
0
500
1000
1500
2000
2500
3000
Iteration number

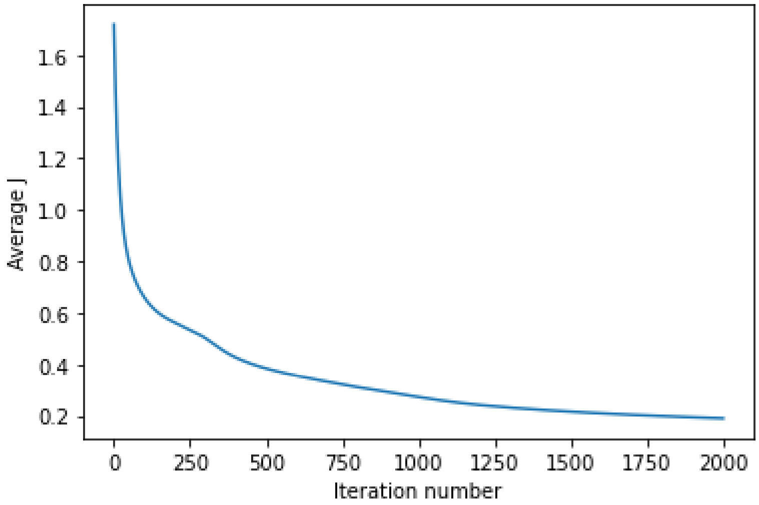
- Leaky ReLU activation function

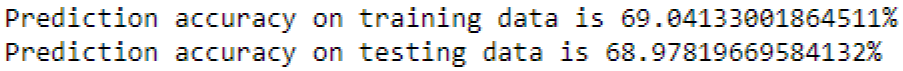
Iterations	Alpha	Lambda	Training Accuracy	Testing Accuracy
1,000	0.25	0.001	22.040%	22.288%
2,000	0.25	0.001	50.800%	51.064%
1000	0.01	0.001	72.553%	72.984%
2000	0.01	0.001	69.041%	68.978%





```
Prediction accuracy on training data is 72.50621504039776%
Prediction accuracy on testing data is 72.51540732301%
```





```
Prediction accuracy on training data is 69.04133001864511%
Prediction accuracy on testing data is 68.97819669584132%
```

CONCLUSION

Logistic Regression

In implementing logistic regression with L1 regularization, as the C hyperparameter value increases, so does the training and testing accuracy of the model. The highest testing accuracy was acquired from using C=1. The training accuracy and testing accuracy is very close to each other, indicating a good fitting and balance between bias and variance.

In implementing logistic regression with L2 regularization, as the C hyperparameter value increases, so does the training and testing accuracy of the model. The highest testing accuracy was acquired from using C=0.01. In comparing the results of logistic regression with L1 regularization and the results of logistic regression with L2 regularization, the training and testing accuracy for the logistic regression with L2 regularization was generally higher. The training accuracy and testing accuracy is very close to each other, indicating a good fitting and balance between bias and variance.

In implementing logistic regression with polynomial feature transformation of degree 2 and L1 regularization, as the C hyperparameter value increases, so does the training and testing accuracy of the model. The highest testing accuracy was acquired from using C=0.1. The resulting training and testing accuracies are different from each other by about 0.5% when C >= 0.1. This is not a significant number; however, it should be noted since in previous models without polynomial feature transformation, the training and testing accuracies were about the same.

In implementing logistic regression with polynomial feature transformation of degree 2 and L2 regularization, as the C hyperparameter value increases, so does the training and testing accuracy of the model. The highest testing accuracy was acquired from using C=0.1. The resulting training and testing accuracies do different from each other by about 0.6% when C >= 0.1. The training accuracy and testing accuracy is very close to each other, indicating a good fitting and balance between bias and variance.

In implementing logistic regression with L1 regularization but with a different randomized training and testing data set, as the C hyperparameter value increases, so does the training and testing accuracy of the model. The highest testing accuracy was acquired from using C=1 and C=10. In comparing the results of logistic regression with L1 regularization with the original training and testing data set with this training and testing data set, the testing accuracies are very similar and very high. This indicates the data set has a good fitting and balance between bias and variance.

SVM

In implementing SVM with linear kernel, the training and testing accuracy of the model increases as the C value (Regularization parameter. Strength of regularization is inversely proportional to C value) rises until 0.01, and from then the accuracy fluctuates very slightly. Overall, the highest testing accuracy is obtained when the C value is 0.01. The training accuracy and testing accuracy is very close to each other, indicating a good fitting and balance between bias and variance.

In implementing SVM with a radial kernel, the training and testing accuracy of the model increases as the C value rises until C value of 10, which is the largest C value we tested. Compared to the linear kernel, higher C values were required to start achieving high accuracy. The highest testing accuracy it achieved was higher than the linear kernel. The difference between training accuracy and testing accuracy showed more differences compared to the linear kernel, as the training accuracy went up, which may indicate overfitting.

In implementing SVM with a polynomial kernel. Similar to the two models before, accuracy rises as the C value increases, although it took higher C values to achieve high accuracy compared to the two models before. At C value of 10, it achieved the greatest accuracy. Although out of the 3 models, the polynomial kernel has the greatest training accuracy, it had the lowest highest testing accuracy. Which may suggest a greater degree of overfitting compared to the other two models.

Neural Network

In implementing neural networks with a sigmoid activation function, two different alpha values were tested with different numbers of iterations. As the number of iterations increased for both alpha values tested, so does the training and testing accuracy of the model. The highest testing accuracy was acquired from using alpha=0.25 and 3,000 iterations.

In implementing neural networks with a ReLU activation function, two different alpha values were tested with different numbers of iterations. As the number of iterations increased for both alpha values tested, the training and testing accuracy of the model decreased. This may be due to overshooting the maximum or getting stuck in shallow local maximums. The training and testing accuracy for both alpha values at 1,000 iterations were similar but for both alpha values at 3,000 iterations, alpha=0.25 had a higher training and testing accuracy. The highest testing accuracy was acquired from using alpha=0.01 and 1,000 iterations.

In implementing neural networks with a Leaky ReLu activation function, two different alpha values were tested with 2 different numbers of iterations. Accuracy tended to decrease as iteration numbers increased, indicating possible problems with overshooting and getting stuck. A smaller alpha value of 0.01 compared to 0.25 showed significantly better accuracy. Overall, the highest testing accuracy was achieved by using parameters of alpha of 0.01 and 1000 iterations.

Overall

For each of the three learning models, the resulting training and testing accuracies never differ from each other by a significant amount. This indicates that there is no overfitting nor underfitting occurring.

Overall, the highest testing accuracy was obtained from using the SVM model. The highest testing accuracy within the logistic regression models was 96.722% obtained from using a polynomial feature transformation of degree 2 and L1 regularization. The highest testing accuracy within the SVM models was 96.745% obtained from using the radial kernel and C=10. The highest testing accuracy within the neural network models was 95.914% obtained from using the sigmoid function, 3,000 iterations, and alpha=0.25.