Cameron Lindsay
V00927778
Undergraduate Student

SENG 474 – Assignment 2

1. Dataset Preprocessing, Testing Information, and Issues

The dataset utilized is the popular fashion-MNIST images developed by Zalando Research (https://github.com/zalandoresearch/fashion-mnist). Prior to use with developed models, data was pruned for class 5 and 7 targets and rescaled according to SENG 474 Assignment 2 guidelines. Class labels of the utilized dataset were altered from 5 and 7 to 0 and 1, respectively. All models developed used the provided 12000-image training set to train the model and 2000-image testing set to test the model unless otherwise specified. Learning curves for all models used Scikit Learn's (sklearn) default settings for all optional parameters, of note: this means a 3-fold cross-validation strategy was utilized. While a custom k-fold cross-validation strategy was developed for the purposes of hyperparameter tuning, no discernable way of using this custom k-fold algorithm for the learning curves was known. Given the default cross-validation strategy was not used for the purposes of hyperparameter tuning directly and instead was used for pedagogical purposes of parameter manipulation on the observed models, its presence should not impact the study's overall analysis.

2. Logistic Regression

Logistic Regression models were created using sklearn's 'LogisticRegression' function [1]. All logistic regression models used a Limited-memory Broyden–Fletcher–Goldfarb–Shannon optimization solver and an L2-Regularization bias strategy. When the learning curves where initially performed, convergence errors occurred due to the default iteration settings of sklearn. To resolve this, maximum iterations of the model were increased to 5000 (parameter max_iter = 5000 was used) and was kept altered for the final logistic regression model. Learning curves were shadowed by the standard deviation (STD) for each individual curve as a measure of variance between data. With the regularization parameter (C) being of focus for this analysis, learning curves comparing varying C-values to negative mean squared error (MSE) were utilized in conjunction with k-fold cross-validation hyperparameter tuning techniques to find the optimal C parameter for the final logistic regression model.

2.1. Logistic Regression Regularization Parameter Baseline

Learning curves were initially performed to obtain a broad view of the impact of changing regularization parameters on model MSE. This was done by observing several C-values on a decreasing logarithmic range (though not represented as logarithm). Values of C observed ranged from 0.001 to 100 (Figure 2.1a and b). Following observations of both graphs, the range of observed C-values was narrowed according to minimizing MSE (maximizing negative MSE) as training set size increases for both training and testing curves. Graphically, this would appear as both curves converging toward an MSE that most nears 0. When observing Figure 2.1a, overfitting as a product of too little regularization is apparent relative to other observed curves when C is highest (C=100). A much wider gap in negative MSE between training error and testing error is shown, with training error being much lower relative to testing error. This suggests the model has obtained too much complexity that is relevant only to the training data, and thus fails when predicting classifications of novel test data. The opposite appears true of using the smallest C-value (Figure 2.1b, C=0.001), as both training and testing data have nearly equal MSE but is much worse relative to other observed values. This suggests too much bias has been introduced, and the model is far too general to make accurate predictions from novel data.

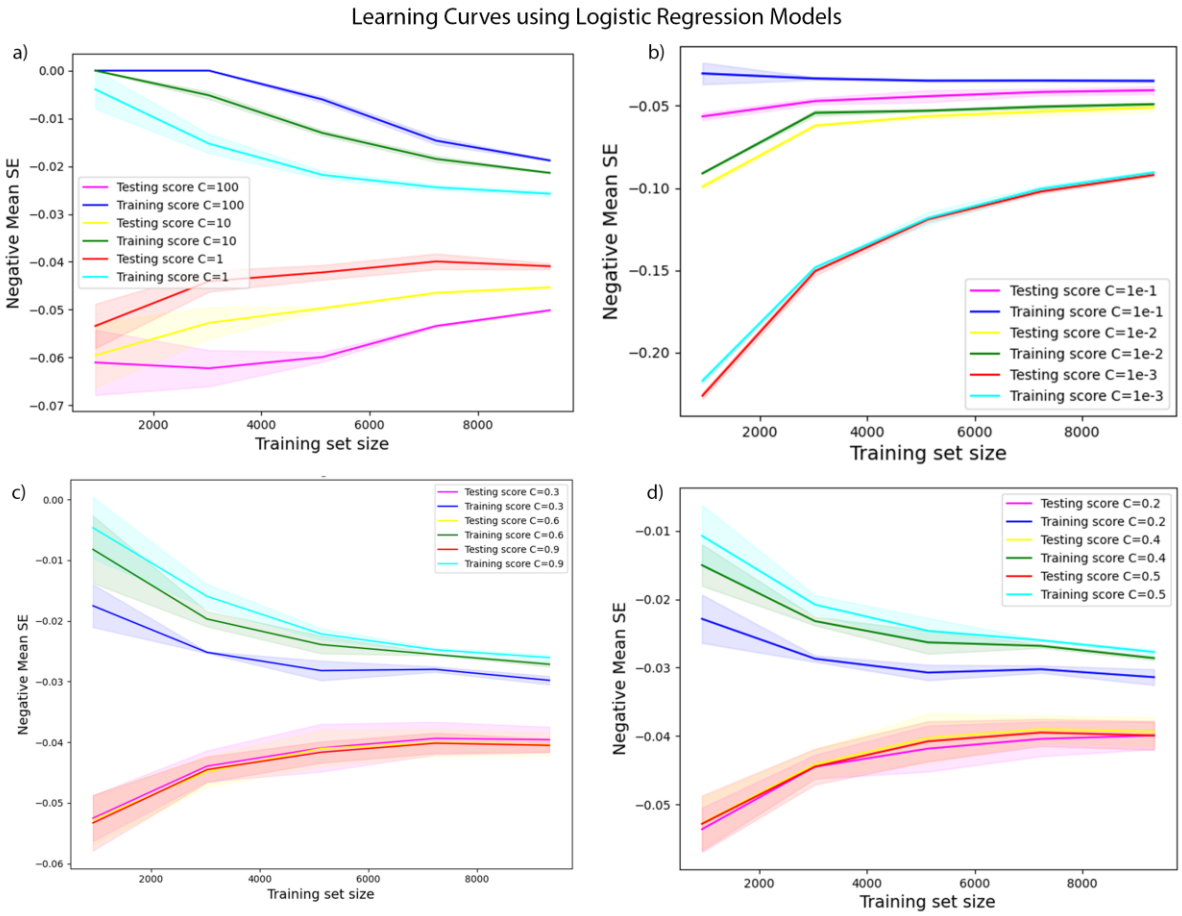

Figure 2.1. C Parameter Impacts on Regularization – Logistic Regression

From Figure 2.1a, the lowest root mean squared error (RMSE) and highest accuracy of the tested model was seen when C = 1 (RMSE = 0.2049, Accuracy = 0.958). From Figure 2.1b, this same trend was seen when C was highest (C=0.1, RMSE = 0.2, Accuracy = 0.96). From this, it was gleaned that the next list of observed C-values should fall between 0.1 and 1.0. Figures 2.1c and d are reflective of this next range of values and changes in RMSE appear relatively stable but is smallest when C = 0.4 (RMSE = 0.1923, Accuracy = 0.963). This would suggest hyperparameter tuning should focus on a range of values that lies between 0.3 and 0.5, using a cross validation scheme to determine the optimally regularized logistic regression model. A table of all observed C-values used for this experiment can be seen in the Appendix's Table 1. Also of note, is the STD of all curves appears relatively minimal, but overall higher for the test data. This is understandable given the much smaller sample size of the test data relative to the training data.

2.2. Logistic Regression Model Hyperparameter Tuning

Using estimates from the previous section (Section 2.1) as a baseline for ranges worthy of observation, logistic regression models were subjected to a 5-fold cross-validation scheme using only training data (12000 samples, 9600 for training and 2400 for testing) and were compared to models using the same 5-fold cross-validation with the complete dataset. Cross validation techniques are used to maximize

available training data and provide a view of the model's ability to generalize to novel data following the modification of various parameters that may impact its performance. K-fold cross validation does this by dividing the dataset provided to it into k divisions (folds). One of the folds is used to validate the model trained with the remaining folds. This is repeated with each fold acting as the validation set and new models using the same parameter. Models are identical based on initial parameters but are not re-trained. The average of the validation measurement observed is taken from each model and returned for final analysis.

Using the 5-fold cross validation scheme, a validation curve was developed observing C-values ranging from 0.3-0.5 and were compared for RMSE against test data utilizing the full dataset (12000 samples for training, 2000 for testing, Figure 2.2.1). Accuracy scores were also observed. While an optimal C-value would likely be selected based on the test data's values, the purpose of this exercise is to evaluate the utility of cross-validation and the regularization of models. Therefore, an optimal C-value was selected according to the RMSE scores from the validation curve.

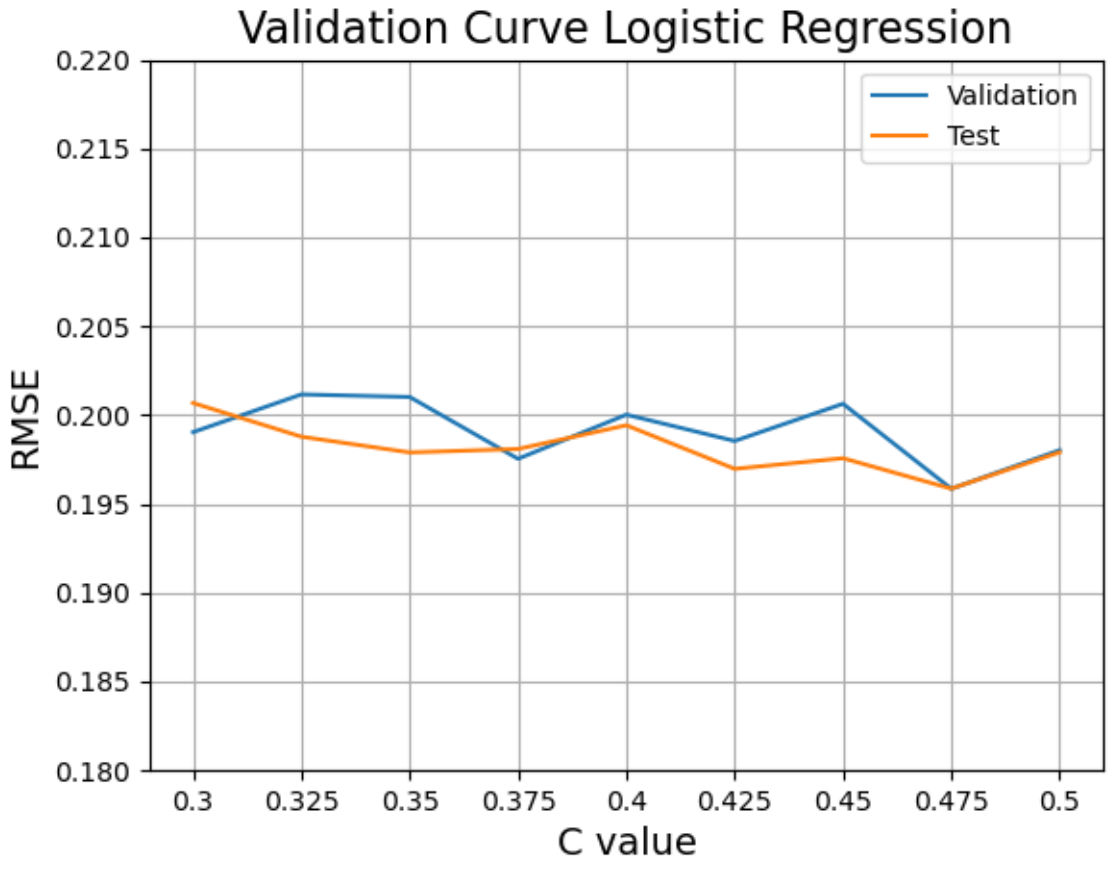

Figure 2.2.1. Validation Curve comparing C-values to RMSE – Logistic Regression

From Figure 2.2.1, the validation RMSE is lowest when using a C-value of 0.475 (RMSE = 0.1958, Accuracy = 0.9615). Interestingly, this is also the same value as compared to the test curve (RMSE = 0.1958, Accuracy = 0.9615). A complete set of RMSE and accuracy scores from validation and test curves can be seen in the Appendix (Table 2). When 5-fold cross-validation was repeated using only C= 0.475,

an F-test using SciPy Stats cdf function [3] with a 95% confidence interval (degrees of freedom assumed to be 1) was performed alongside a boxplot comparison of median values to IQR (Figure 2.2.2). The F-Test found the errors to not be statistically significant in their difference (H_0: error validation = error test, p=0.575) and this is reflected in the boxplot, as the median of the whole data (right) falls within the IQR of the training data alone (left).

Figure 2.2.2. Comparison of median RMSE values of C=0.475 using training alone vs whole dataset Note: Green arrows = mean, yellow line = median, notched area represents 95% confidence interval around the median followed by the interquartile range (IQR). Some instances show an IQR that higher than the CI values, resulting in the graph 'folding in' on itself. Whiskers represent range of data with points representing outliers.

3. Support Vector Machines

Support Vector Machine (SVM) models were created using sklearn's 'SVC' function [2]. Bias strategy utilized was a squared L2 penalty. With only the C parameter and kernel manipulated from default values. Initially, methodology utilized in the logistic regression validation curves (Section 2) was attempted for these models. Unfortunately, running times were extremely long and, rather than modifying the dataset size, a maximum iterations value of 5000 (max_iter = 5000) was placed on the models. This resulted in a failure to converge and an early termination of the solver. Therefore, it is likely that the algorithm did not meet the tolerance cut-off value of 0.0001 (default). However, this is still preferable to lowering the tolerance directly as there was a chance it was met during some iterations of the algorithm. While this may impact overall model accuracy, it likely did not have as much of an impact as removal of half (or more) of the training data values as per SENG 474 Assignment 2 suggestions. As in the previous section (Section 2), learning curves were shadowed by the standard deviation (STD) for each individual curve as a measure of variance between data. Learning curves comparing varying C-values to negative mean squared error (MSE) were utilized in conjunction with k-fold cross-validation hyperparameter tuning techniques to find the optimal C parameter for the final SVM model utilizing a linear kernel. The SVM utilizing a Gaussian kernel was subjected only to k-fold cross validation hyperparameter testing, albeit with a larger range of observed data values.

3.1 Linear Kernel

As previous, learning curves were initially performed to obtain a broad view of the impact of changing regularization parameters on model MSE. This was done by observing several C-values on a decreasing logarithmic scale (though not represented as logarithm). Values of C observed ranged from 0.001 to 100 (Figure 3.1.1a and b). Following observations of both graphs, the range of observed C-values was narrowed according to minimizing MSE (maximizing negative MSE) as training set size increases for both training and testing curves. Interestingly, when observing Figure 3.1.1a overfitting as a product of too little regularization is difficult to interpret, as both training and testing curves converge as training set size increases. The impact of C-value modification also appears to have dramatic impacts that are difficult to predict. Conventional knowledge would suggest that as C increases, the model becomes increasingly complex and vice versa. However, when C=10 the negative MSE plummets and variance dramatically increases for both training and test curves. This is not true of C=100, where we would expect to see even further drops in negative MSE but, the increased STD is also there, suggesting the increased variance is a by-product of including more misclassified data values when training the algorithm. When observing the smaller C-values (Figure 3.1.1b), underfitting shows the same trends as logistic regression, with both training and testing data have nearly equal MSE but is much worse relative to other observed values. Once again, suggesting too much bias has been introduced, and the model is far too general to make accurate predictions from novel data.

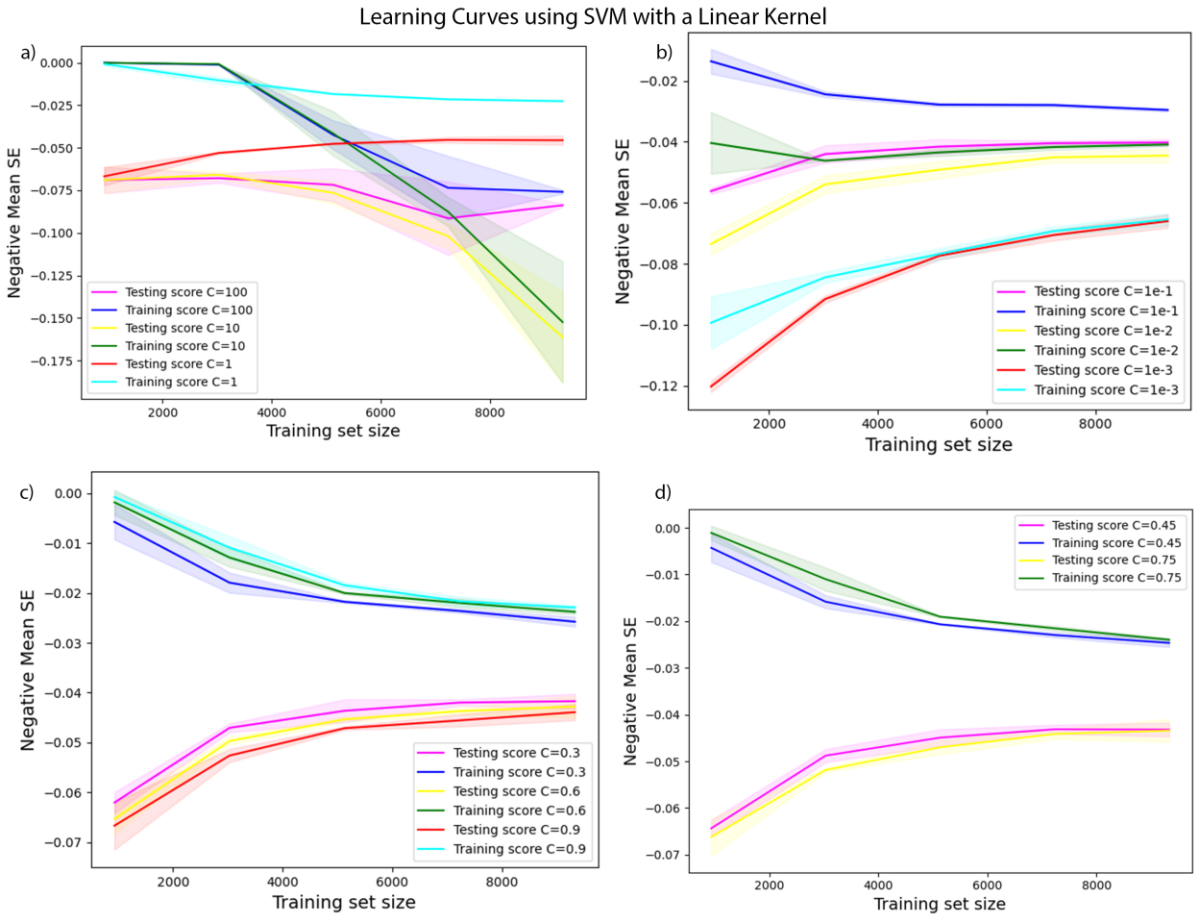

Figure 3.1.1. C Parameter Impacts on Regularization – SVM with Linear Kernel

From Figure 3.1.1a, the lowest root mean squared error (RMSE) and highest accuracy of the tested model was seen when C = 1 (RMSE = 0.2156, Accuracy = 0.9535). From Figure 3.1.1b, this same trend was seen when C was highest (C=0.1, RMSE = 0.1962, Accuracy = 0.9615). From this, it was gleaned that the next list of observed C-values should fall between 0.1 and 1.0. Figures 3.1.1c and d are reflective of this next range of values and changes in RMSE appear relatively stable but is smallest when C = 0.6 (RMSE = 0.1897, Accuracy = 0.964). This would suggest hyperparameter tuning should focus on a range of values that lies between 0.5 and 0.7, using a cross validation scheme to determine the optimally regularized SVM model. A table of all observed C-values used for this experiment can be seen in the Appendix's Table 3.

3.2 Linear Kernel Hyperparameter Tuning

Using estimates from the previous section (Section 3.1) as a baseline for ranges worthy of observation, SVM models were subjected to a 5-fold cross-validation scheme using only training data (12000 samples, 9600 for training and 2400 for testing) and were compared to models using the same 5-fold cross-validation with the complete dataset. A validation curve was developed observing C-values ranging from 0.5-0.7 and were compared for RMSE against test data utilizing the full dataset (Figure 3.2.1). Accuracy scores were also observed. While an optimal C-value would likely be selected based on the test

data's values, the purpose of this exercise is to evaluate the utility of cross-validation and the regularization of models. Therefore, an optimal C-value was selected according to the RMSE scores from the validation curve.

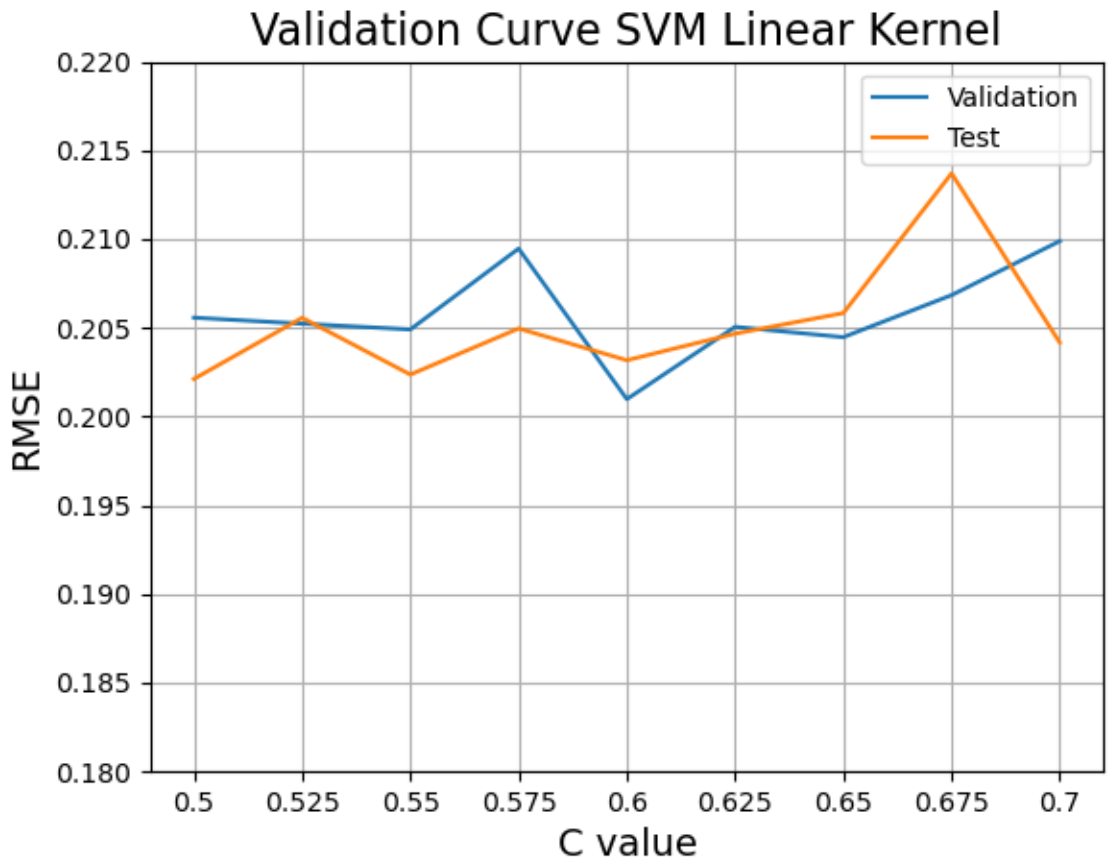

Figure 3.2.1. Validation Curve comparing C-values to RMSE – SVM with Linear Kernel

From Figure 3.2.1, the validation RMSE is lowest when using a C-value of 0.6 (RMSE = 0.2009, Accuracy = 0.9595). Interestingly, this C-value is much higher than the test curve's lowest RMSE (C=0.5, RMSE = 0.2021, Accuracy = 0.9591). A complete set of RMSE and accuracy scores from validation and test curves can be seen in the Appendix (Table 4). When 5-fold cross-validation was repeated using only C= 0.6, an F-test using 95% confidence interval (degrees of freedom assumed to be 1) was performed alongside a boxplot comparison of median values to IQR (Figure 3.2.2). The F-Test found the errors to not be statistically significant in their difference (H_0: error validation = error test, p=0.327) and this is reflected in the boxplot, as the median of the whole data (right) falls within the IQR of the training data alone (left).

Figure 3.2.2. Comparison of median RMSE values of C=0.6 using training alone vs whole dataset
Note: Green arrows = mean, yellow line = median, notched area represents 95% confidence interval around the median followed by the IQR. Some instances show an IQR that lower than the CI values, resulting in the graph 'folding in' on itself. Whiskers represent range of data with points representing outliers.

3.3 Gaussian Kernel

Using 5-fold cross validation methodologies defined in the previous section (Section 3.2), Gamma (γ) values were modified using sklearn's default scaling coefficient for γ ("1 / (n_features * X.var()" [2]) and modification of the C regularization parameter. A logarithmic base range of C-values (though not represented as logarithm) from 0.0001 to 10000 was used to obtain a baseline of ranges worthy of observation (Figure 3.3.1a). As the lowest RMSE of the validation curve was between 1 and 100 (C=10, RMSE=0.1342, Accuracy=0.9819), this process was repeated to further narrow the range of observation (Figure 3.3.1b). From Figure 3.3.1b, the lowest RMSE of the validation curve was found to be at a C-value of 60 (RMSE=0.1310, Accuracy=0.9827) and therefore, the range of observation was further narrowed to C=55-65. From Figure 3.3.1c, the optimal value of C from the validation curve was selected based on lowest RMSE and shown to be C=60 (RMSE=0.1354, Accuracy=0.9815). This was the C-value utilized in

the final model. A table of all observed C-values used for this experiment can be seen in the Appendix's Table 5.

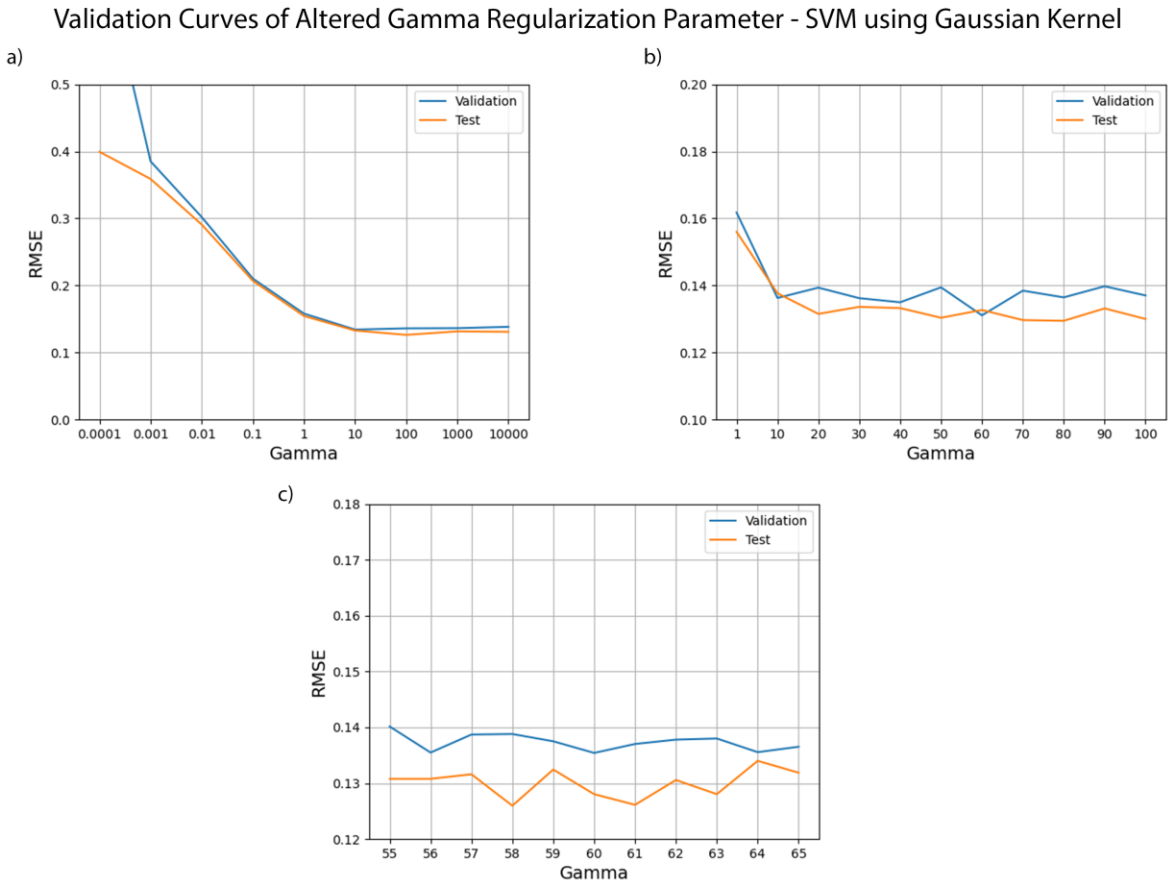

Figure 3.3.1. Validation Curves of SVM using Gaussian Kernel

From Figure 3.3.1a, the C-values of the test curve versus the validation curve optimal C-values seem to differ by at least a factor 10. The lowest RMSE of the test curve is at C=100 (RMSE=0.1294, Accuracy=0.9838). This would suggest training size is a very important factor in determining regularization relative to the other observed models. As C-value increases by a factor of 10 when training size is increased only by 2400 samples (20%). Surprisingly, when 5-fold cross-validation was repeated using only C= 60 and subjected to an F-test using 95% confidence interval (degrees of freedom assumed to be 1), the errors were found to not be statistically significant in their difference (H_0: error validation = error test, p=0.563). A boxplot comparison of median values to IQR (Figure 3.3.2) was also conducted and the median of the whole data (right) falls within the IQR of the training data alone (left).

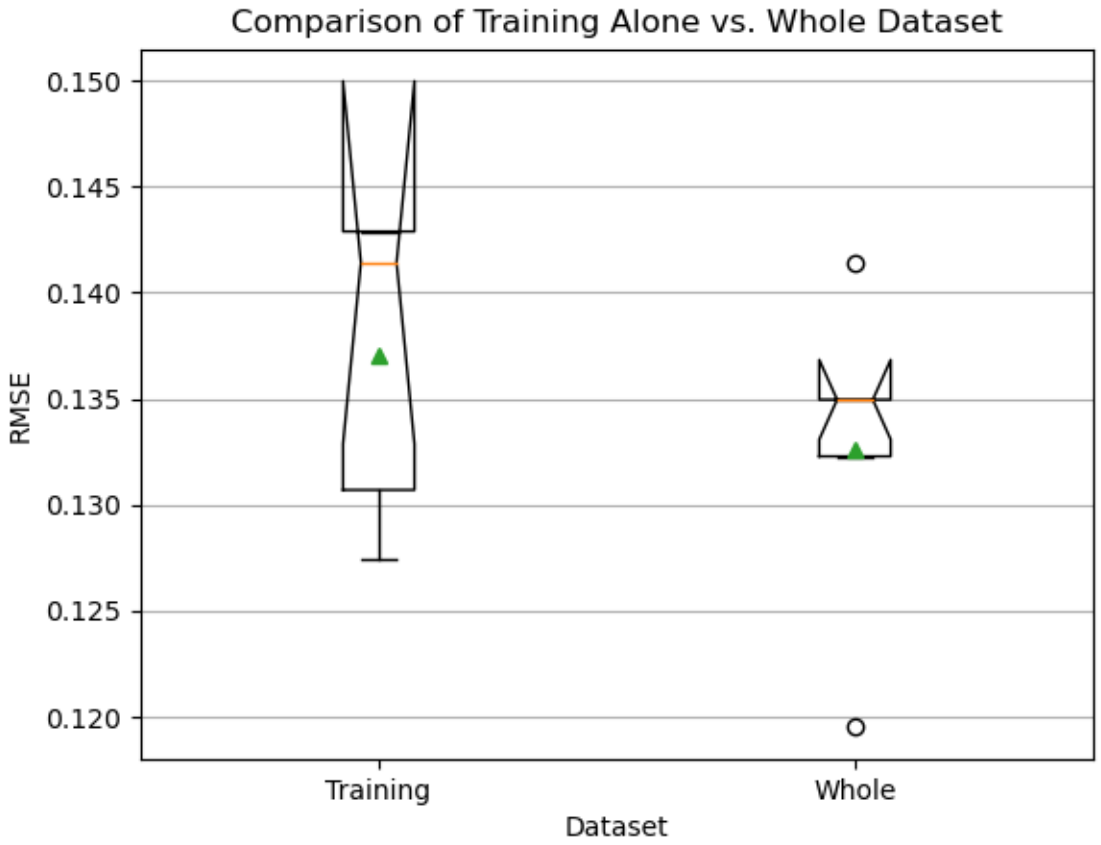

Figure 3.3.2. Comparison of median RMSE values of C=60 using training alone vs whole dataset
Note: Green arrows = mean, yellow line = median, notched area represents 95% confidence interval around the median followed by the IQR. Some instances show an IQR that lower than the CI values, resulting in the graph 'folding in' on itself. Whiskers represent range of data with points representing outliers.

4. Model Comparisons and Concluding Remarks

All three models were subjected to a oneway ANOVA using SciPy's F_oneway function [4] (F-test, alpha=0.05, H_0: error logistic regression = error SVM linear kernel = error SVM Gaussian kernel) to determine differences in variance between all three error values of optimal models subjected to 5-fold cross-validation. This was accompanied by a boxplot comparison of median RMSE values to IQR (Figure 4.1).

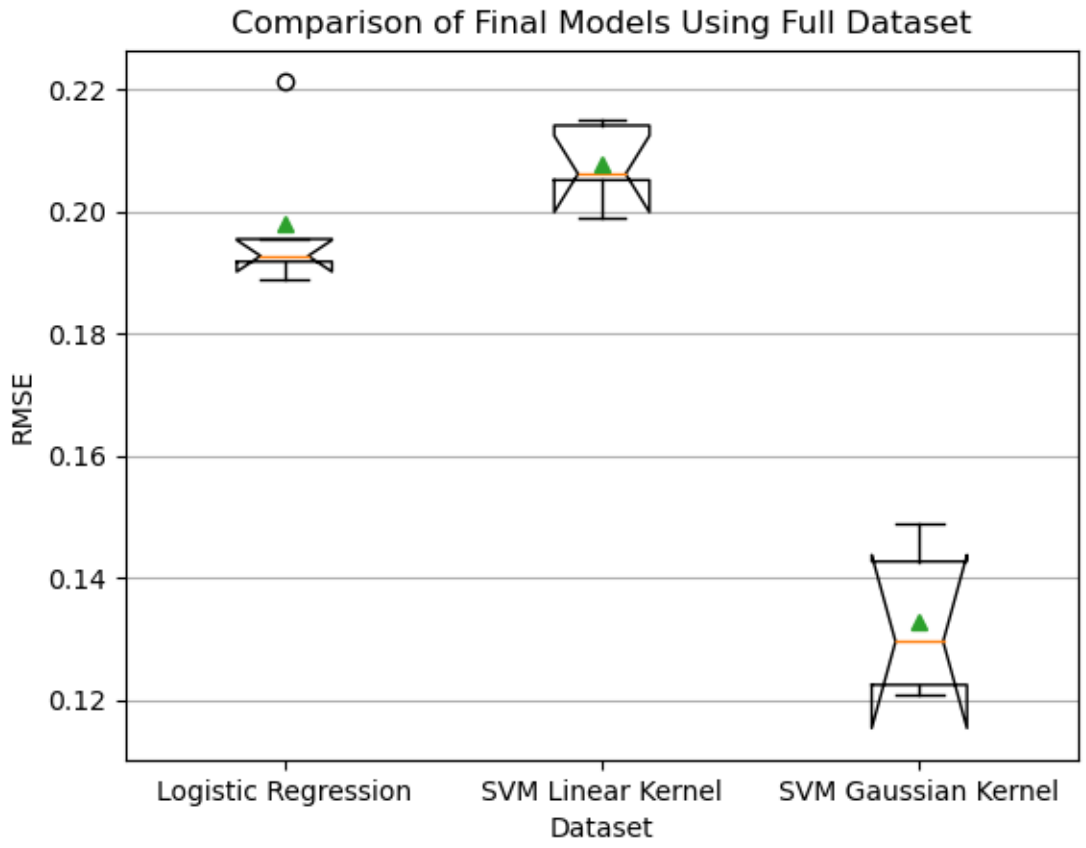

Figure 4.1. Comparison of median RMSE values of each model with optimal parameters
Note: Green arrows = mean, yellow line = median, notched area represents 95% confidence interval around the median followed by the IQR. Some instances show an IQR that higher than the CI values, resulting in the graph 'folding in' on itself. Whiskers represent range of data with points representing outliers.

What was found was the failure of H_0 and the errors of each model were significantly different from one another ($p<0.05$). This is reflected in the boxplot as well as no median values overlapped IQRs or confidence intervals. Based on these results, the SVM model utilizing a gaussian kernel is the best model to use for the dataset and classification task at hand. From previous tests observing RMSE using optimal regularization parameters, the SVM using a Gaussian kernel had a lowest mean RMSE of 0.1280. This

was followed by the logistic regression model at 0.1958 and finally the SVM using a linear kernel at 0.2031.

Several issues arose during the conduction of this study. The most prevalent error would be the maximum iterations limitation placed on the SVM models as an effort to reduce runtime. As mentioned previously, this limitation likely had an impact on the accuracy of these models. In particular, the SVM utilizing the linear kernel threw the most convergence errors and this was likely the reason behind its relatively poor performance and odd overfitting behaviors. When researching into overfitting behaviors of SVM using learning curves, no examples presented trends I had encountered in Section 3.1 with higher C-values. This was most likely an artifact of applying a max iterations quantifier on the models observed and is not representative of overfitting. Most likely, the curves would have behaved comparably to its counterpart when C=100, albeit with a less wide margin between the testing and training curves.

Fortunately, SVM using a Gaussian kernel displayed quite low RMSE even despite this limitation on max iterations. Although, as seen in Section 3.3, the regularization parameter appears to be highly sensitive to the amount of data used to train the algorithm. The optimal model used likely could have been better using a regularization parameter closer to 100.

Appendix

Table 1. Learning Curve Score Measurements – Logistic Regression

C-Value	*RMSE*	*F1-Score*	*Accuracy Score*
Figure 2.1a			
100	0.2236	0.95	0.95
10	0.2156	0.95	0.9535
1	0.2049	0.96	0.958
Figure 2.1b			
0.1	0.2	0.96	0.96
0.01	0.2269	0.95	0.9485
0.001	0.2819	0.92	0.9205
Figure 2.1c			
0.3	0.1974	0.96	0.961
0.6	0.1974	0.96	0.961
0.9	0.2037	0.96	0.9585
Figure 2.1d			
0.2	0.1974	0.96	0.961
0.4	0.1923	0.96	0.963
0.5	0.1949	0.96	0.962

Table 2. Validation Curve Score Measurements – Logistic Regression

C-Value	*RMSE: Validation Curve*	*RMSE: Test Curve*	*Accuracy: Validation Curve*	*Accuracy: Test Curve*
0.3	0.1990	0.2006	0.9603	0.9597
0.325	0.2011	0.1987	0.9594	0.9604
0.35	0.2010	0.1978	0.9595	0.9607
0.375	0.1975	0.1980	0.9609	0.9607

0.4	0.2000	0.1994	0.9599	0.9601
0.425	0.1985	0.1969	0.9605	0.9611
0.45	0.2006	0.1975	0.9596	0.9609
0.475	0.1958 (min)	0.1958 (min)	0.9615 (max)	0.9615 (max)
0.5	0.1980	0.1979	0.9606	0.9607

Table 3. Learning Curve Score Measurements – SVM with Linear Kernel

C-Value	*RMSE*	*F1-Score*	*Accuracy Score*
Figure 3.1a			
100	0.4785	0.77	0.771
10	0.2991	0.91	0.9105
1	0.2156	0.95	0.9535
Figure 3.1b			
0.1	0.1962	0.96	0.9615
0.01	0.2190	0.95	0.952
0.001	0.2489	0.94	0.938
Figure 3.1c			
0.3	0.1949	0.96	0.962
0.6	0.1897	0.96	0.964
0.9	0.2121	0.95	0.955
Figure 3.1d			
0.45	0.2024	0.96	0.959
0.75	0.2133	0.95	0.9545

Table 4. Validation Curve Score Measurements – SVM with Linear Kernel

C-Value	*RMSE: Validation Curve*	*RMSE: Test Curve*	*Accuracy: Validation Curve*	*Accuracy: Test Curve*
0.5	0.2055	0.2021 (min)	0.9576	0.9591 (max)

0.525	0.2052	0.2055	0.9577	0.9577
0.55	0.2049	0.2023	0.958	0.9588
0.575	0.2094	0.2049	0.9560	0.9579
0.6	0.2009 (min)	0.2031	0.9595 (max)	0.9585
0.625	0.2050	0.2046	0.9579	0.958
0.65	0.2044	0.2058	0.958	0.9574
0.675	0.2068	0.2137	0.9571	0.9542
0.7	0.2098	0.2041	0.9559	0.9582

Table 5. Validation Curve Score Measurements – SVM with Gaussian Kernel

C-Value	*RMSE: Validation Curve*	*RMSE: Test Curve*	*Accuracy: Validation Curve*	*Accuracy: Test Curve*
Figure 3.3.1a				
0.0001	0.7129	0.3993	0.4916	0.8403
0.001	0.3851	0.3588	0.8515	0.8711
0.01	0.3020	0.2909	0.9087	0.9153
0.1	0.2103	0.2070	0.9556	0.9570
1	0.1581	0.1545	0.9748	0.9760
10	0.1342 (min)	0.1328	0.9819 (max)	0.9822
100	0.1360	0.1263 (min)	0.9814	0.9838 (max)
1000	0.1363	0.1316	0.9813	0.9826
10000	0.1383	0.1309	0.9807	0.9827
Figure 3.3.1b				
1	0.1617	0.1560	0.9736	0.9756
10	0.1362	0.1376	0.9814	0.9810
20	0.1393	0.1315	0.9804	0.9826
30	0.1362	0.1336	0.9814	0.9820
40	0.1349	0.1332	0.9815	0.9822

50	0.1394	0.1303	0.9804	0.9828
60	0.1310 (min)	0.1326	0.9827	0.9823
70	0.1384	0.1296	0.9807	0.9830
80	0.1364	0.1294 (min)	0.9813 (max)	0.9832 (max)
90	0.1397	0.1331	0.9804	0.9821
100	0.1370	0.1300	0.9811	0.9830
Figure 3.3.1c				
55	0.1401	0.1307	0.9801	0.9828
56	0.1354	0.1307	0.9814	0.9827
57	0.1387	0.1316	0.9806	0.9826
58	0.1388	0.1259 (min)	0.9805	0.9840 (max)
59	0.1375	0.1324	0.9810	0.9824
60	0.1354 (min)	0.1280	0.9814	0.9834
61	0.1370	0.1261	0.9811	0.984
62	0.1378	0.1305	0.9810	0.9829
63	0.1380	0.1280	0.9809	0.9834
64	0.1355	0.1340	0.9815 (max)	0.9819
65	0.1365	0.1318	0.9812	0.9825

References

[1] “Sklearn. linear_model.LogisticRegression.” Scikit, scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html. Accessed July 10, 2020.

[2] “Sklearn.svm.SVC.” Scikit, scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html. Accessed July 10, 2020.

[3] Unknown, Z. (2020, July 14). How to Perform an F-Test in Python. Retrieved July 15, 2020, from https://www.statology.org/f-test-python/

[4] Scipy.stats.f_oneway¶. (n.d.). Retrieved July 16, 2020, from https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.f_oneway.html