**MACHINE LEARNING FROM DATA**

**Report: Lab Session 6– Support Vector Machines**

**Names: Abdullah Qureshi, S M Rakib Hasan and Niina Hietamäki**

**Instructions**

* Answer the questions
* Save the report and upload the file and the notebook in a zip file

**Questions**

**Q1: Complete a table with the training, validation and test errors for the linear and Gaussian SVMs. Which are the values of C in each case and the value of h for the Gaussian SVM?**

Summary of Training and Testing Errors:

C = 0.1: Train Error = 0.1625, Test Error = 0.1000

C = 1: Train Error = 0.1625, Test Error = 0.1500

C = 10: Train Error = 0.1625, Test Error = 0.1500

C = 100: Train Error = 0.1750, Test Error = 0.1000

C = 500: Train Error = 0.1750, Test Error = 0.1000



|  | Linear SVM | |
| --- | --- | --- |
| C\_value | Train Error | Test Error |
| 0.1 | 0.1625 | 0.1000 |
| 1 | 0.1625 | 0.1500 |
| 10 | 0.1625 | 0.1500 |
| 100 | 0.1750 | 0.1000 |
| 500 | 0.1750 | 0.1000 |

The unoptimized classifier with C=1 has a train error of 0.1625 and testing error of 0.1500

Summary of Training and Testing Errors:

C = 1, gamma = 1.0000e-04: Train Error = 0.1500, Test Error = 0.2000

C = 1, gamma = 1.0000e-03: Train Error = 0.1500, Test Error = 0.2000

C = 1, gamma = 1.0000e-02: Train Error = 0.1375, Test Error = 0.1500

C = 1, gamma = 1.0000e-01: Train Error = 0.1500, Test Error = 0.1000

C = 10, gamma = 1.0000e-04: Train Error = 0.1500, Test Error = 0.2000

C = 10, gamma = 1.0000e-03: Train Error = 0.1375, Test Error = 0.1500

C = 10, gamma = 1.0000e-02: Train Error = 0.1625, Test Error = 0.1500

C = 10, gamma = 1.0000e-01: Train Error = 0.1375, Test Error = 0.1000

C = 100, gamma = 1.0000e-04: Train Error = 0.1375, Test Error = 0.1500

C = 100, gamma = 1.0000e-03: Train Error = 0.1625, Test Error = 0.1500

C = 100, gamma = 1.0000e-02: Train Error = 0.1500, Test Error = 0.1000

C = 100, gamma = 1.0000e-01: Train Error = 0.1375, Test Error = 0.1000

C = 200, gamma = 1.0000e-04: Train Error = 0.1750, Test Error = 0.1500

C = 200, gamma = 1.0000e-03: Train Error = 0.1625, Test Error = 0.1000

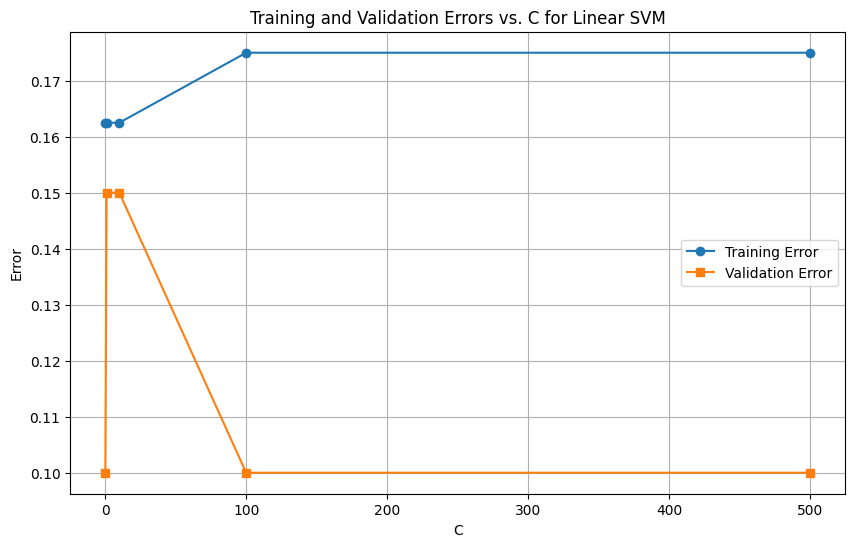
C = 200, gamma = 1.0000e-02: Train Error = 0.1500, Test Error = 0.1000

C = 200, gamma = 1.0000e-01: Train Error = 0.1250, Test Error = 0.1000



|  |  | Gaussian SVM | |
| --- | --- | --- | --- |
| C\_value | h\_value | Train Error | Test Error |
| 1 | 1.00e-04 | 0.1500 | 0.2000 |
| 1 | 1.00e-03 | 0.1500 | 0.2000 |
| 1 | 1.00e-02 | 0.1375 | 0.1500 |
| 1 | 1.00e-01 | 0.1500 | 0.1000 |
| 1 | 1.00e-04 | 0.1500 | 0.2000 |
| 1 | 1.00e-03 | 0.1375 | 0.1500 |
| 1 | 1.00e-02 | 0.1625 | 0.1500 |
| 1 | 1.00e-01 | 0.1375 | 0.1000 |
| 1 | 1.00e-04 | 0.1375 | 0.1500 |
| 1 | 1.00e-03 | 0.1625 | 0.1500 |
| 1 | 1.00e-02 | 0.1500 | 0.1000 |
| 1 | 1.00e-01 | 0.1375 | 0.1500 |
| 1 | 1.00e-04 | 0.1750 | 0.1000 |
| 1 | 1.00e-03 | 0.1625 | 0.1000 |
| 1 | 1.00e-02 | 0.1500 | 0.1000 |
| 1 | 1.00e-01 | 0.1250 | 0.1000 |

**Q2: Plot the training and the validation scores, find the optimal value of C**



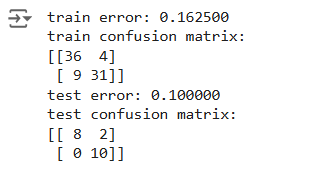
Training Error: Shows minimal variation as Cincreases.

Validation Error: Increases for higher C values due to overfitting

The optimal value of C is around 0.1 which minimizes the validation error.

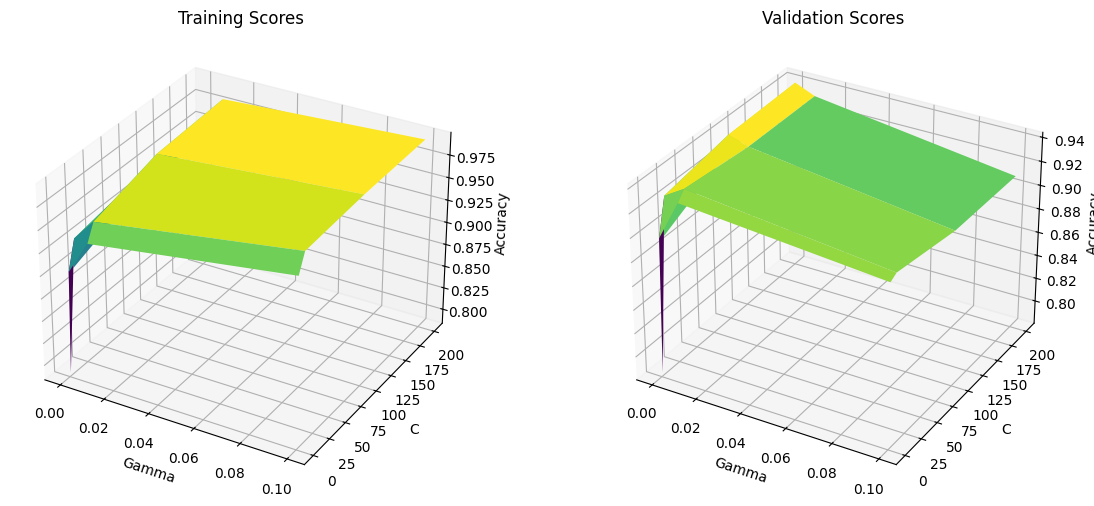
**Q3: For the best classifier found in the previous step, compute classification error on the test set, compare with the error obtained for the non-optimized linear classifier (Q1).**

Using the value of 0.1 of C with the Linear SVM the test set error of 0.1000



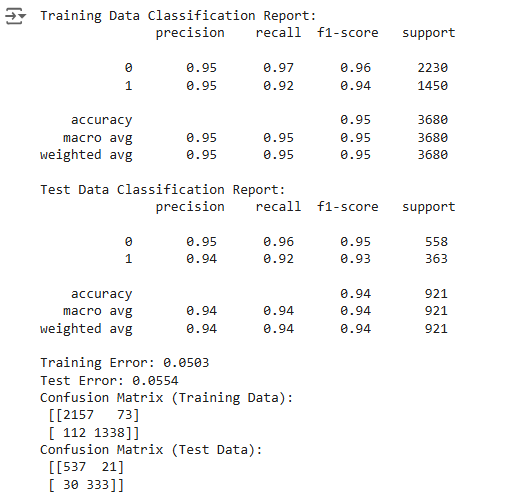
With the non-optimized linear classifier we got the same test error of around 0.1500 with C=1. Optimization of CCC significantly improves model performance, as observed in the test error reduction for the Linear SVM.

**Q4: Plot the training and validation scores (two 3D plots). Find the optimal values of C and h**



The optimal values of C=200 and γ=0.001 were determined using grid search cross-validation (GridSearchCV). These parameters achieved the highest validation accuracy, as shown in the 3D plots of training and validation scores.

**Q5: For the best classifier found in the previous step, compute classification error on the test set, compare with the error obtained for the non-optimized Gaussian classifier (Q1) and for the results of the optimized linear SVM (Q3).**



Non-Optimized Gaussian SVM: Training Error = 0.1625, Test Error = 0.1500

Optimized Linear SVM: Training Error = 0.1625, Test Error = 0.1000

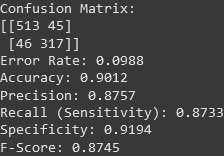
Optimized Gaussian SVM: Training Error = 0.0503, Test Error = 0.0554

The optimized Gaussian SVM performed the best out of all the models, with the lowest training and test errors. It showed a big improvement compared to both the non-optimized Gaussian SVM and the optimized linear SVM, making it the most accurate and reliable option.

**Q6: Compute the confusion matrix for the test set. Compute the six metrics (*error*, *accuracy*, *precision*, *recall*, *specificity* and *f-score*).**

From the previous case, the best c=200 and best gamma=0.001, which we get from the table that shows the c and gamma values with their corresponding F1 scores.

So, for this part, we choose these parameters for our SVM model. After training and testing, we get the metrics as follows:



**Q7: Explain why *precision*, *recall*, *specificity* and *f-score* are more appropriate than *accuracy* and *error* for evaluating the classifier performance.**

**Answer:** Accuracy and error are simple metrics that show how many predictions were correct or incorrect. However, they can be misleading, especially when the dataset is imbalanced (one class has far more samples than the other). For example, if 90% of emails are NOT SPAM, a model that always predicts "NOT SPAM" will have high accuracy but will fail to detect SPAM emails.

In contrast:

* Precision: Focuses on how many of the predicted SPAM emails are actually SPAM. This is important when the cost of false positives (e.g., wrongly marking important emails as SPAM) is high.
* Recall (Sensitivity): Measures how many of the actual SPAM emails were correctly identified. This is crucial when we don’t want to miss any SPAM emails.
* Specificity: Look at how well the model correctly identifies non-SPAM emails. This ensures regular emails are not wrongly flagged as SPAM.
* F-Score: Balances precision and recall. It’s useful when both false positives and false negatives matter.

These metrics give a deeper understanding of the classifier's performance, especially when class distributions are uneven or the cost of errors is different for each class.