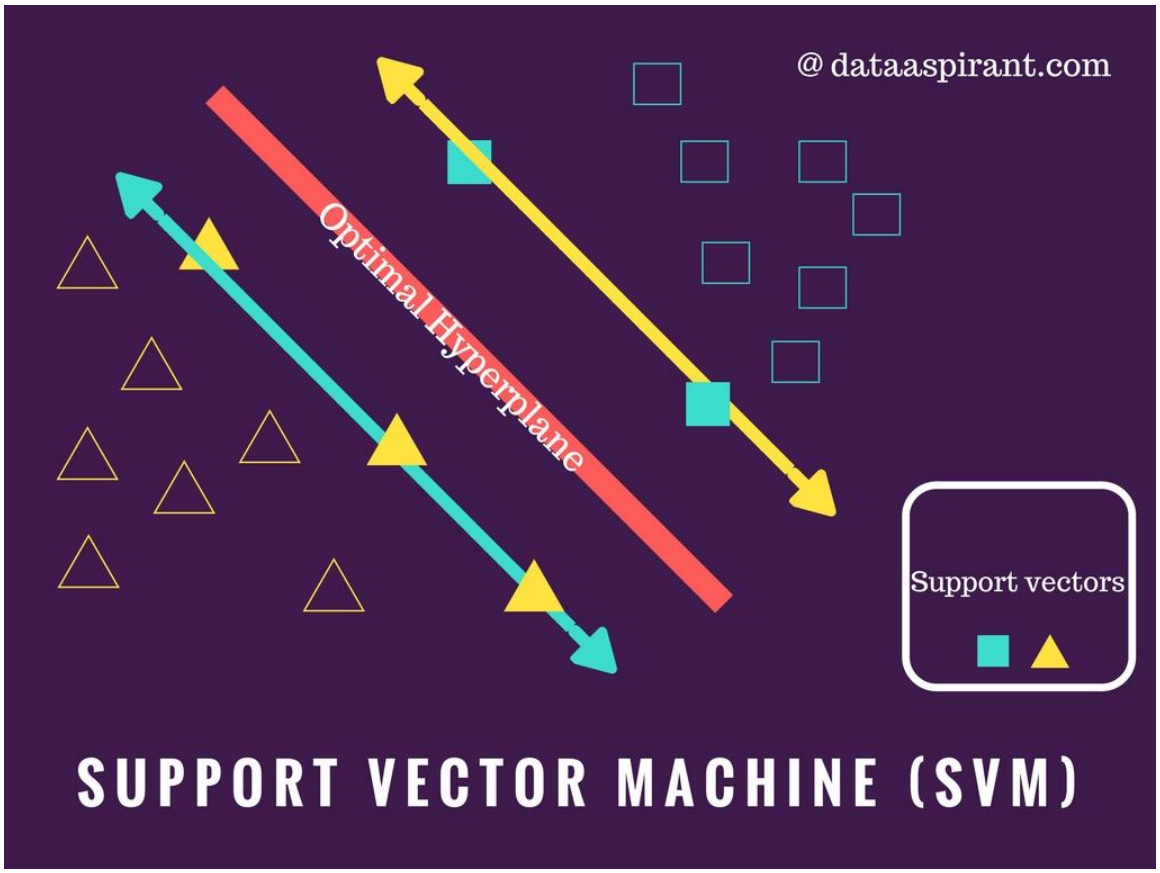


Support vector machine

HW3

109511276 練鈞揚 |Machine Learning | 4/30/2023

Discuss About the SVM

In HW3, I learned a lot about SVM. This was my first time using open-source software to complete my homework. There were different platforms and programming languages available to implement SVM, but I chose to use Python for this homework.

First, c-SVM is a conventional support vector machine classifier that seeks to classify data by finding a hyperplane that maximizes the margin between classes. It balances the tradeoff between training error and model complexity through the regularization parameter C. Larger C values can make the classifier more prone to overfitting the training data, while smaller C values lead to a simpler model.

Second, nu-SVM is a kernel-based support vector machine classifier that uses a parameter called ν to balance the tradeoff between training error and model complexity. Unlike C, ν limits both training error and the number of support vectors. ν can be viewed as an upper bound on training error while limiting the number of support vectors.

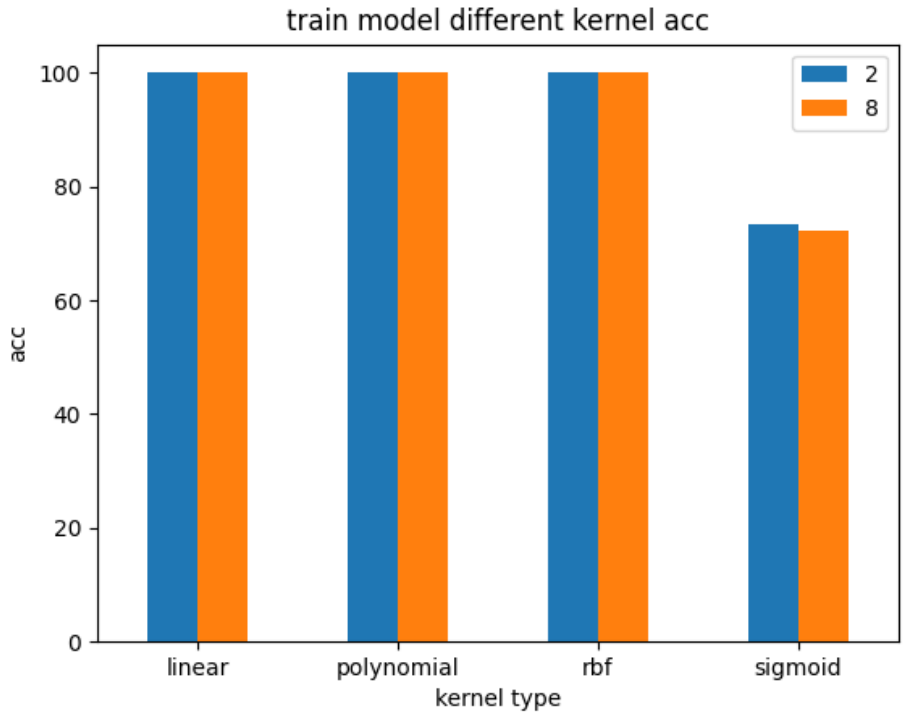
In practice, c-SVM is typically used for binary classification problems, while nu-SVM is more suitable for multi-class problems. Additionally, c-SVM has faster training times for large datasets, while nu-SVM is better suited for smaller datasets as it can reduce the number of support vectors while maintaining an upper bound on training error.

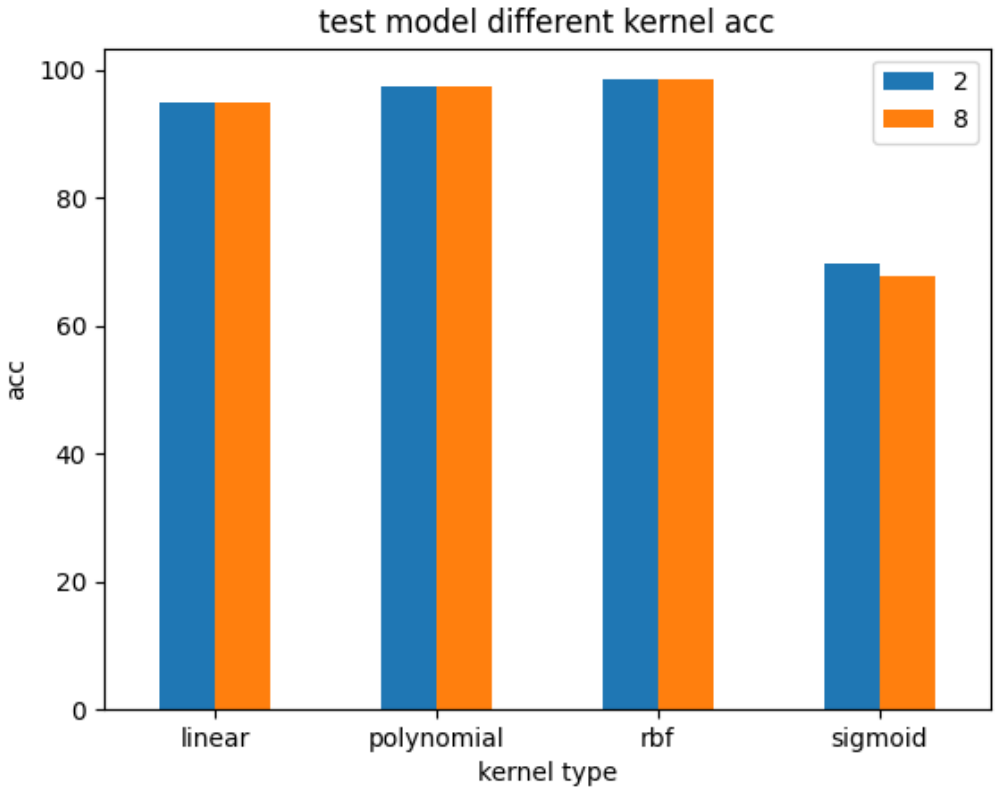
In order to find the best C value for c-SVM, I used easy.py to explore all the parameters of c-SVM. I found that with a gamma value of 0.03125 and C value of 8, the prediction accuracy on the training data was 100%. With a C value of 2, the prediction accuracy was around 98%.

For nu-SVM, I was unsure of the best n value, so I wrote code to automatically train the model and find the best parameter. After training the nu-SVM model, I found that the Python version could not run the command "svm_prediction". I suspected that the Python version had a bug, so I wrote a mini Python function to call the "svm-predict" application to obtain accuracy.

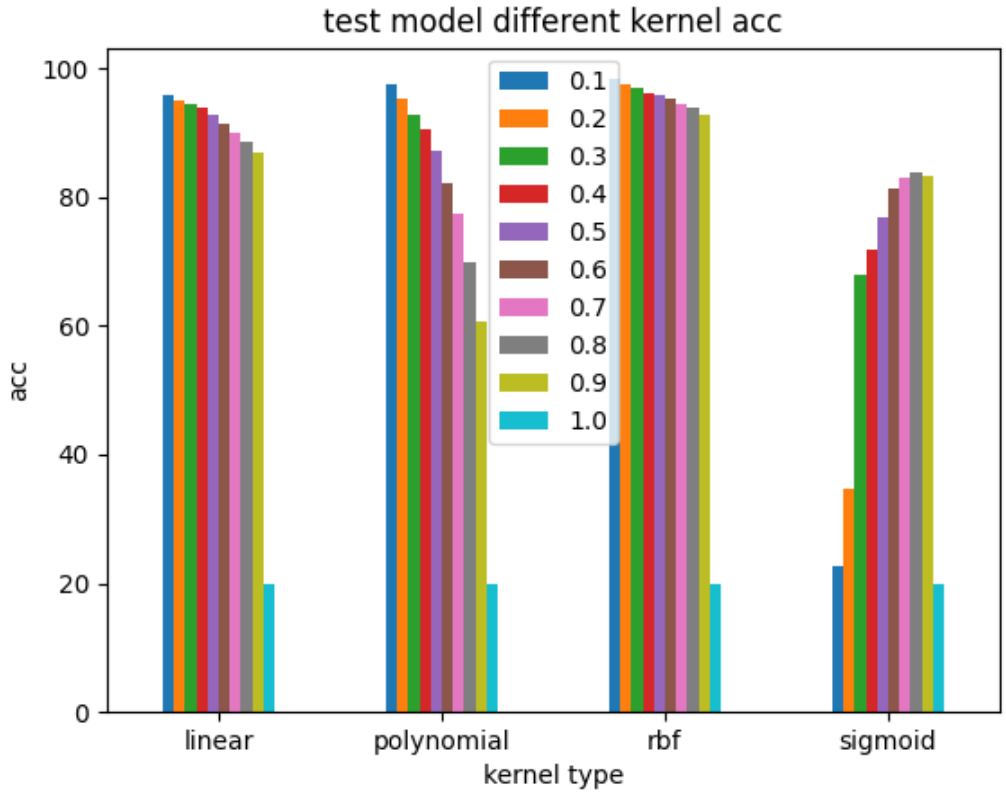
ACCURACY :

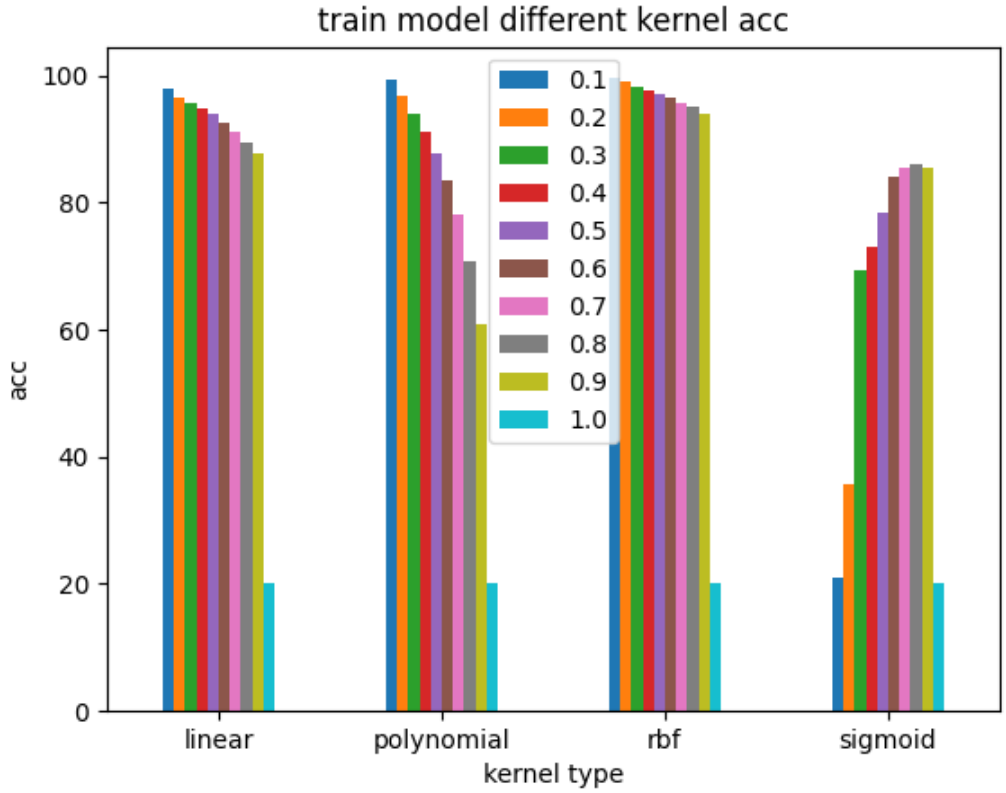
- *c-SVM (C in 2 and 8)(find in easy.py)*





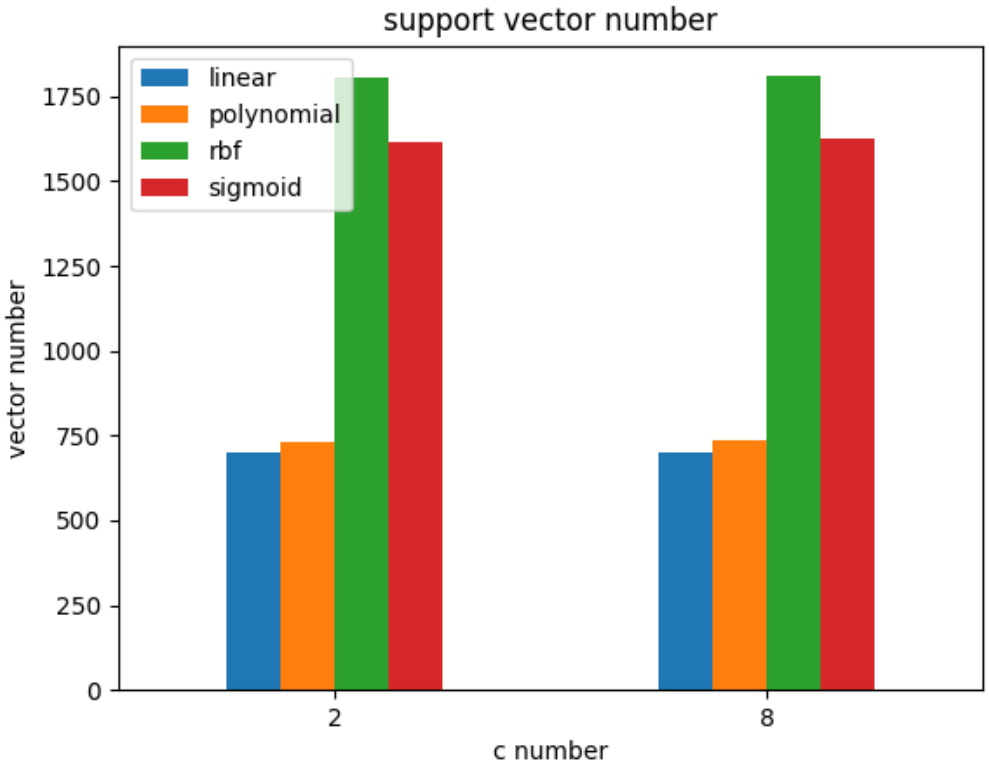
– ***nu-SVM (n number range in 0.1 to 1.0)***



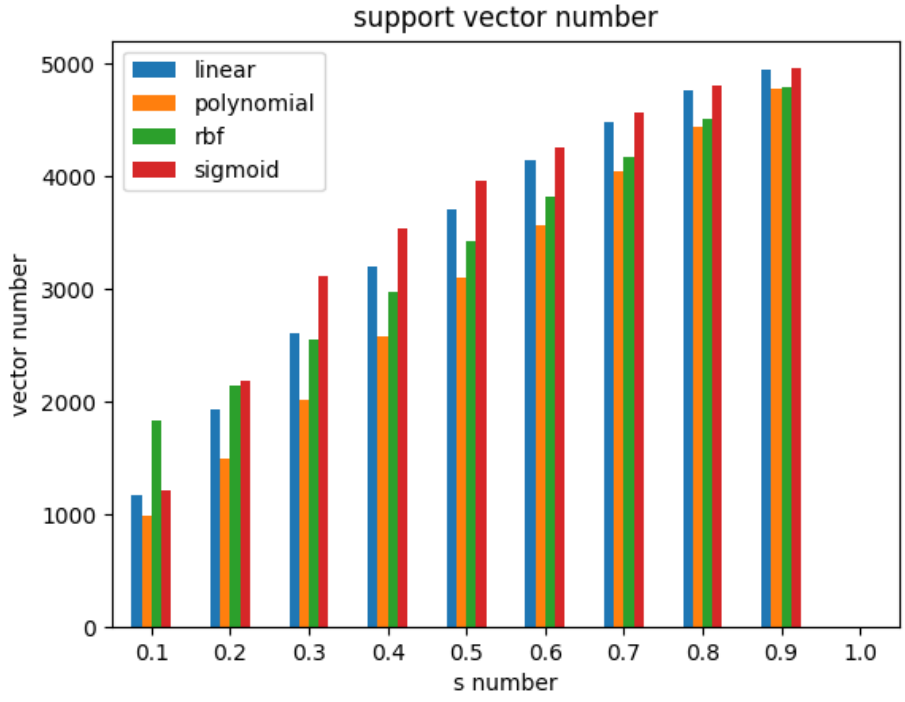


NUMBER OF USE SUPPER VECTOR:

- ***c-SVM:***



- ***nu-SVM:***



Best Model suggestion:

- ***c-SVM***

- c=2
- gammer = 0.03125
- kernel function use polynomial , when need less support vector, else using RBF.

- ***nu-SVM***

- n=0.1
- gammer = 0.03125
- kernel function use polynomial , when need less support vector, else using linear.

This suggest is find in the graph, about the support vector, I save in “svm_c_auto_train/support_vector” and “svm_s_auto_train/support_vector”, file name is “svm_c2_polynomial_support_vector.txt” and “svm_no.1_polynomial_support_vector.txt”