**Madelyn Caufield**

**ECON425**

**Homework Assignment 4**

**PART I**

Google co-lab link: <https://colab.research.google.com/drive/1qdzIUMamL4zjqbWOCDNy-Y0_SCQykfHW?usp=sharing>

After randomly splitting the data, three different FNN models were trained over the training samples and tested over the validation samples. The first set of hyperparameters consisted of two hidden layers (10 and 5 neural units, both ReLU), a sigmoid activation, and an ‘adam’ optimizer. This model provided an R^2 of -0.7936011701393288. The second model consisted of three hidden layers (15, 10, and 5 neural units, sigmoid, ReLU, and ReLU activation), a sigmoid output activation, and an ‘adam’ optimizer. This model provided an R^2 of -0.7936011714813744. The final model consisted of two hidden layers (100 and 50 neural units, sigmoid and ReLU activation), a sigmoid output activation, and an ‘adam’ optimizer. This model provided an R^2 of -0.7936011746369931. Based on the R^2, model 1 performed the best. We applied this top ranked model over the testing samples and achieved an R^2 of -0.8651321275075976.

Ten testing samples which received largest absolute error:

Graphical user interface, application, table, Excel

Description automatically generated

When comparing the observations associated with the highest absolute values to the 10,186 other observations, we see that population block population is lower than the mean value of all 10,196 observations. But for the most part the feature values seem standard.

**PART II**

After randomly splitting the data, we plot the validation errors using different values of C.

A picture containing shape

Description automatically generated

From the above graph, we see can see that the lowest error with the smallest C-value is C=1. Next, we plot the validation errors using linear, RBF, and polynomial kernel types with C=1.

Chart, line chart

Description automatically generated

From the above graph, we see that the linear kernel type provided us the smallest error. We trained an SVM model with a linear kernel type and C-value of 1 and applied it over the testing set to obtain the following results:

-------------------------------------------------------

Confusion Matrix:

[[48 0]

[ 8 44]]

Average Accuracy: 0.92

Per-Class Precision: [0.85714286 1. ]

Per-Class Recall: [1. 0.84615385]

-------------------------------------------------------

Correctly Classified Samples:

-------------------------------------------------------

array([[ 1. , 14.3, 11.6, 31.3, 35.5, 12.7],

[ 0. , 13.7, 11. , 27.5, 30.5, 12.2],

[ 1. , 16.4, 13. , 35.7, 41.8, 15.2],

[ 1. , 11.2, 10. , 22.8, 26.9, 9.4],

[ 0. , 21.6, 15.4, 45.7, 49.7, 20.6]])

-------------------------------------------------------

Incorrectly Classified Samples:

-------------------------------------------------------

array([[ 0. , 12.5, 9.4, 23.2, 26. , 10.8],

[ 1. , 12.3, 11. , 26.8, 31.5, 11.4],

[ 1. , 11.8, 10.5, 25.2, 29.3, 10.3],

[ 1. , 11.1, 9.9, 23.8, 27.1, 9.8],

[ 0. , 13.2, 11. , 27.1, 30.4, 12.2]])

-------------------------------------------------------