Machine Learning Experiments with Tic Tac Toe

Huan KUANG, huan2015@ufl.edu

I. IMPLEMENTATION

In this project, there are three parts, namely, binary classification, multi-class classification, and multi-label regressor. The models were implemented using the scikit-learn library. Ten-fold cross validation was adopted to validate the stability of the model and to evaluate the performance of the models.

For binary and multi-class classification problems, linear Support Vector Machines (SVM), k-nearest neighbors (KNN), and multilayer perceptron (MLP) were trained and evaluated. The linear kernel was used for binary SVM, and the radial basis function (RBF) kernel was used for multi-class SVM. Regarding KNN, different k values were evaluated via for loops across the numbers in the range from 1 to 10 (10 values in total) for the binary classification problem, and across the odd numbers in the range from 1 to 19 (10 values in total). Three-layer MLP was implemented for binary and multi-class classification tasks. I used rectified linear unit (ReLU) as the activation function for the hidden layer, and adam as the solver for weight optimization. The number of neurons in the three hidden layers was 40, 10, 5 for the binary task and 60, 45, 30 for the multi-class task. The maximum number of iterations was 200 and 500 for binary and multi-class classification separately.

Linear regression, KNN, and MLP were trained and evaluated for the multi-label task. I produced nine linear regression models, where each model regresses a given single output based on the nine-vector input. I compare the nine continuous outputs of the linear models and decide the move that has the highest output as the optimal move. For the multi-label task, the k value was set as 1 in the KNN. For this study, the best k is 1, and the accuracy decreases when k increases. Like before, Three-layer MLP was implemented with ReLU and adam. The number of neurons in the three hidden layers was 500, 100, 50, and the maximum number of iterations was 500.

II. RESULT

A. *Binary Classification*

Overall, all three models performed well on the binary classification task. The results of stability testing with cross-validation showed that MLP worked best. Specifically, MLP achieved an average model accuracy of 100% with a standard deviation (SD) equal to 0 on the testing set with the ten-fold cross-validation scheme. The linear SVM achieved an average model accuracy of 98.3% with SD equal to 0.01 and KNN (k = 1) achieved an average model accuracy of 99.3% with SD equal to 0.01.

See the normalized confusion matrix of the three models based on the last fold of the Ten-fold cross-validation in Fig. 1. It showed that the trained models could accurately classify when X_Player won the game but misclassified some O_player won cases as X_Player won. The reason that all models perform quite well is that this task is simple.

```
This is the 10 th fold
SVM Accuracy: 0.968421052631579
Here is the confusion matrices for SVM based on testing set:
              Predicted_O_Player_Won  Predicted_X_Player_Won
O_Player_Won                   0.925                   0.075
X_Player_Won                   0.000                   1.000
MLP Accuracy: 0.9789473684210527
Here is the confusion matrices for MLP based on testing set:
              Predicted_O_Player_Won  Predicted_X_Player_Won
O_Player_Won                    0.95                    0.05
X_Player_Won                    0.00                    1.00
KNN (k=1) Accuracy : 1.0
Here is the confusion matrices for KNN based on testing set:
              Predicted_O_Player_Won  Predicted_X_Player_Won
O_Player_Won                     1.0                     0.0
X_Player_Won                     0.0                     1.0
```

Fig. 1. Confusion Matrix for Binary Classification Task

For KNN, when the k is bigger than 8, the accuracy decreases when k increases (see Fig. 2.). When k is smaller than 8, the choice of k did not influence the performance of the model.

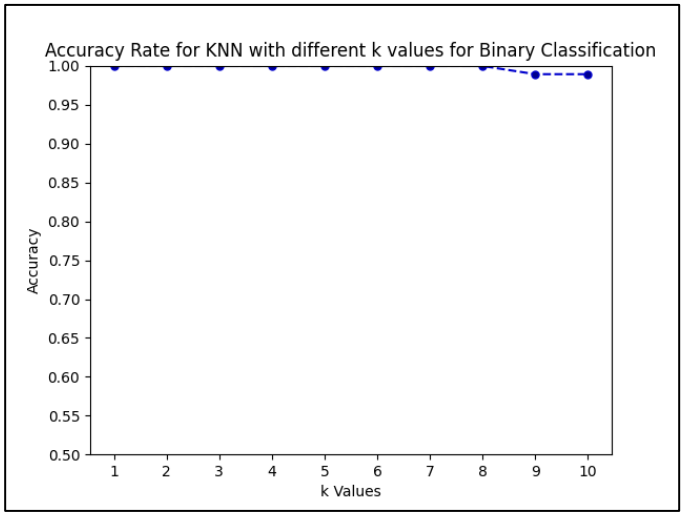

Fig. 2. Accuracy for KNN with different k values for Binary Classification

B. Multi-class Classification

Overall, all three models performed acceptably on the multi-class classification task. The results of stability testing with cross-validation showed that MLP has higher prediction accuracy than the other two models. The reason why MLP worked best maybe that MLP could hold the critical features for classification and filter out the noise from the data by having the latent layers. Moreover, the non-linear active function ReLU helped with deriving the features' non-linear properties. However, it is well seen for this multi-class dataset, SVM and KNN are usually much quicker.

For this multi-class problem, MLP achieved an average model accuracy of 94.4% with a standard deviation (SD) equal to 0.01 on the testing set with the ten-fold cross-validation scheme. The SVM achieved an average model accuracy of 82.4% with SD equal to 0.02 and KNN (k = 1) achieved an average model accuracy of 88.4% with SD equal to 0.02.

See the normalized confusion matrix of the three models based on the last fold of the Ten-fold cross-validation in Fig. 3. It seemed that there is no clear pattern among the predicted optimal places. Sometimes the trained models could accurately classify the optimal places for the next movement if the places were far from each other, for example, upper right and bottom left.

This is the 10 th fold

SVM Accuracy: 0.8519083969465648

Here is the confusion matrices for SVM based on testing set:

	PUL	PUM	PUR	PML	PCenter	PMR	PBL	PBM	PBR
Upper_Left	0.967	0.000	0.000	0.000	0.026	0.007	0.000	0.000	0.000
Upper_Middle	0.086	0.800	0.029	0.000	0.029	0.000	0.029	0.000	0.029
Upper_Right	0.085	0.038	0.830	0.019	0.019	0.000	0.009	0.000	0.000
Middle_Left	0.043	0.064	0.021	0.766	0.085	0.000	0.021	0.000	0.000
Center	0.059	0.008	0.017	0.008	0.882	0.008	0.017	0.000	0.000
Middle_Right	0.075	0.025	0.025	0.000	0.025	0.800	0.025	0.025	0.000
Bottom_Left	0.158	0.035	0.000	0.000	0.018	0.000	0.789	0.000	0.000
Bottom_Middle	0.222	0.167	0.000	0.000	0.000	0.000	0.000	0.611	0.000
Bottom_Right	0.111	0.022	0.022	0.000	0.000	0.000	0.022	0.000	0.822

MLP Accuracy: 0.9526717557251908

Here is the confusion matrices for MLP based on testing set:

	PUL	PUM	PUR	PML	PCenter	PMR	PBL	PBM	PBR
Upper_Left	0.974	0.000	0.000	0.000	0.000	0.007	0.013	0.000	0.007
Upper_Middle	0.000	0.971	0.000	0.000	0.000	0.000	0.014	0.000	0.014
Upper_Right	0.009	0.028	0.943	0.000	0.019	0.000	0.000	0.000	0.000
Middle_Left	0.043	0.021	0.000	0.936	0.000	0.000	0.000	0.000	0.000
Center	0.000	0.000	0.008	0.008	0.958	0.025	0.000	0.000	0.000
Middle_Right	0.000	0.000	0.000	0.025	0.000	0.925	0.000	0.050	0.000
Bottom_Left	0.070	0.035	0.000	0.000	0.000	0.000	0.895	0.000	0.000
Bottom_Middle	0.000	0.056	0.000	0.000	0.000	0.000	0.000	0.944	0.000
Bottom_Right	0.022	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.978

KNN (k=1) Accuracy : 0.8916030534351145

Here is the confusion matrices for KNN based on testing set:

	PUL	PUM	PUR	PML	PCenter	PMR	PBL	PBM	PBR
Upper_Left	0.922	0.000	0.020	0.007	0.020	0.000	0.007	0.007	0.020
Upper_Middle	0.000	0.829	0.043	0.014	0.000	0.029	0.014	0.029	0.043
Upper_Right	0.038	0.028	0.840	0.019	0.019	0.000	0.019	0.019	0.019
Middle_Left	0.043	0.043	0.043	0.851	0.000	0.000	0.000	0.000	0.021
Center	0.000	0.025	0.034	0.017	0.916	0.000	0.008	0.000	0.000
Middle_Right	0.025	0.000	0.000	0.000	0.025	0.950	0.000	0.000	0.000
Bottom_Left	0.088	0.000	0.000	0.000	0.000	0.000	0.895	0.018	0.000
Bottom_Middle	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.889	0.111
Bottom_Right	0.022	0.000	0.000	0.000	0.022	0.000	0.000	0.022	0.933

Fig. 3. Confusion Matrix for Multi-class Classification Task.
Note. "PUL" refers to predicted upper left, same for the rest abbreviation.

The best k value is 1 for the multi-class classification problem (see Fig. 4.). The accuracy decreases when k increases. It may be that the bigger neighborhood includes too many data points from other classes when k increase. The KNN algorithm would be more sensitive to outliers with a small k-value. There is no outliner in this dataset, such that the small k-value worked perfected.

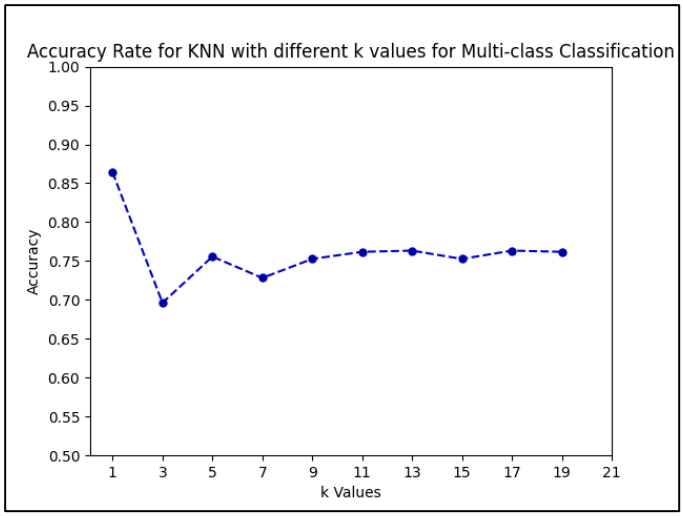

Fig. 4. Accuracy for KNN with different k for Multi-class Classification

C. Multi-label Classification

Overall, all three models performed acceptably on the multi-label task. The results of stability testing with cross-validation showed that MLP has the highest prediction accuracy. Likewise, it may be that MLP could hold the critical features for classification and filter out the noise from the data by having the latent layers. The reason why linear regression performed under expectation could be the multi-class output was non-linear separable. Moreover, the output was unbalanced since there are more 0s than 1s.

For this multi-label problem, MLP achieved an average model accuracy of 89.5% with a standard deviation (SD) equal to 0.02 on the testing set with the ten-fold cross-validation scheme. The linear regression achieved an average model accuracy of 60.5% with SD equal to 0.02 and KNN (k = 1) achieved an average model accuracy of 81.8% with SD equal to 0.02. See the accuracy rate of the three models based on all folds in the Ten-fold cross-validation in Fig. 5.

D. Performance on the 1/10 of the data

To examine what happens to the accuracy of the classifiers if the models were trained on 1/10 as much data, I selected the multi-class classification task as an example to investigate. If train the model on 10% and test it on 90% of the data, MLP achieved an accuracy of 69.4% on the testing set, SVM achieved an accuracy of 67.8% and KNN (k = 1) achieved an accuracy of 54.64%. It will be better to train the models with a larger dataset because 10% of the data could not fully present the possible scenarios. The sample size was extremely crucial for KNN because the dimension of each data point in KNN with

Euclidean distance suffers from the curse of dimensionality. 10% of the data may not be large enough to capture the potential density of the data.

III. INSTRUCTIONS ON RUNNING THE PROGRAMS

• Please run “classifiers_regressors.py”, to examine the models for the Binary Classification Task, Multi-class Classification Task, or Multi-label Regressors. This python script contains the needed models. It will automatically print the required accuracy and or confusion matrix.

• Please run “play_tictactoe.py”, to play the command-line Tic Tac Toe game. You need to install the pickle library to load the trained MLP model to play this game.

• Trained_MLP_mmodel.sav, this python object file contains the multilayer perceptron trained on the tictac_multi.txt dataset.

IV. DIFFICULTIES

The key challenge that I had was to improve the performers of linear regression. I tried different penalty terms, but none of them worked. I also tried logistic regression with the weights to correct the effect of the unbalanced group. However, it did not improve the accuracy. I am wondering is there any specific improvement that I can make towards training linear regression for this task.

Moreover, I found that Gaussian Naive Bayes, Random Forest, and Logistic Regression with classifier chains were more often used in literature. I am wondering why we choose to use linear regression for this multi-label task.

```
This is the 1 th fold
LR Accuracy: 0.6670710594512196
KNN Accuracy: 0.8185975609756098
MLP Accuracy: 0.8948170731707317
This is the 2 th fold
LR Accuracy: 0.6432071984732824
KNN Accuracy: 0.7893129770992366
MLP Accuracy: 0.8748091603053435
This is the 3 th fold
LR Accuracy: 0.6743100610687023
KNN Accuracy: 0.8274809160305343
MLP Accuracy: 0.9038167938931297
This is the 4 th fold
LR Accuracy: 0.6817747480916031
KNN Accuracy: 0.8366412213740458
MLP Accuracy: 0.9206106870229007
This is the 5 th fold
LR Accuracy: 0.6805306335877863
KNN Accuracy: 0.8351145038167939
MLP Accuracy: 0.8931297709923665
This is the 6 th fold
LR Accuracy: 0.6767982900763359
KNN Accuracy: 0.8305343511450382
MLP Accuracy: 0.9206106870229007
This is the 7 th fold
LR Accuracy: 0.6419630839694657
KNN Accuracy: 0.7877862595419848
MLP Accuracy: 0.867175572519084
This is the 8 th fold
LR Accuracy: 0.6680894885496184
KNN Accuracy: 0.8198473282442749
MLP Accuracy: 0.8946564885496183
This is the 9 th fold
LR Accuracy: 0.6743100610687023
KNN Accuracy: 0.8274809160305343
MLP Accuracy: 0.8946564885496183
This is the 10 th fold
LR Accuracy: 0.6606248015267175
KNN Accuracy: 0.8106870229007633
MLP Accuracy: 0.8885496183206106
```

Fig. 5. Accuracy for Multi-label Classification