Sean McLean

ALY 6020

Module 5 Project

Text Classification

Introduction

The module assignment focuses on the usage of a writing test to examine the motor skills of children and determine if more improvement needs to be made. The test consists of writing numbers that can be predicted using KNN modeling and neural networks modeling taught in the week 5 module. The modeling used can provide predictions that could help understand whether a student needs more help with their motor skills. By using two different models, a comparison and contrast is conducted at the end of the project to see which model provides better accuracy from the dataset being used.

Analysis

The necessary packages and libraries are installed before the Excel dataset is imported into the Python notebook. The first five rows are displayed using the head function, showing that there are 46 variables and 42,000 rows in the dataset. The first variable is titled ‘label’ which is the number that the child is attempting to write, ranging from the numbers zero to nine. The other 45 variables pertain to the pixel number contained within the number drawn by the child. The values in each row are the color or intensity of the number's drawing, ranging from zero to 255 as the maximum number. An example of this from the first five rows would be in cell one, where the child is attempting to write the number zero and there are values of 137 in the columns ‘pixel124’ and ‘pixel125’, indicating the intensity or color of the writing on that pixel.

Summary statistics were executed to get a look at some potential relationships and patterns in the dataset. The describe function was used that showed the consistency of each column with the range of pixel values. The frequency of each value in the label column was employed which displayed how many times each number was written by the children in the writing test. The

number written the most was number one with 4,684 attempts, and the number least written was number five with 3,795 attempts. A correlation matrix was built to show the attribute relationships but showed very little correlation in most of the variables, so it is deemed to be not especially useful.

The last step before modeling was data cleaning and preparation with the building of training and testing sets for teaching the machine learning model and then evaluating the model's performance after it has been trained. The target variable that was used is the 'label' variable which is what we are looking at for predicting what number is being written. The dataset is then checked for any missing values like blank cells and non-numerical values, with no missing values found in the entire dataset. The data is also scaled using the StandardScaler function for the purpose of standardizing or normalizing features so that they have similar ranges and distributions (Anaconda, 2024).

Part 1

After importing the necessary libraries and packages, a KNN model is built using the training and testing sets and the 'KNeighborsClassifier' function. The predict function of the scaled x_test data and the accuracy score function of the y_test and y_pred data were then executed that provided an accuracy score of 0.6460. This score was verified by building a confusion matrix and classification report that displayed similar accuracy scores overall. The confusion matrix provided is a 10 by 10 matrix, which indicates a multi-class classification problem with 10 classes, ranging from zero to nine from the label variable. Each row in the confusion matrix represented the actual class, while each column represents the predicted class. The diagonal values represent the number of correct predictions for each class with an example being the value 762 in the (0,0) position, indicating that 762 instances of the digit 0 were correctly predicted as

zero. Values off the diagonal represent misclassifications with an example being an instance where 49 in the (2,0) position indicates that 49 instances of digit 2 were incorrectly predicted as 0 (Anaconda, 2024).

The classification report was built which gives the precision, recall, f1, and support scores for each number in the dependent variable. A few numbers had strong accuracy scores, with the numbers zero, one, and six all having much higher accuracy scores than the average. The other numbers were around the overall accuracy score with a couple numbers coming in below the average. The highest f1 scores were the numbers zero and one at 0.85 and the lowest score was the number nine at 0.43. This could suggest that some numbers are easier to draw than others and that some numbers are difficult to draw for children.

The low accuracy score overall provides a challenge in that there were many predictions that were not correct as indicated by the confusion matrix and classification report. This was adjusted and fine-tuned by changing the optimal number of neighbors for a K-Nearest Neighbors (KNN) classifier which can be crucial for balancing the model's performance. Another issue was the inconsistency with the f1 scores, where some numbers were high, and the others had mediocre scores. This could be an indication that this model only works on particular numbers but is not effective overall. Despite revising the number, the accuracy score did not substantially increase, so there is evidence that this model should not be used for this dataset.

Part 2

Using the MLPClassifer package from the sklearn library and the training and testing sets, the neural networks are created. The MLPClassifier is configured with one hidden layer containing 100 neurons. The stochastic gradient descent (SGD) is used as the optimization

algorithm to adjust the weights of the network. The learning rate is set to 0.01, which controls the step size during training. The model is then set to perform a maximum of 10,000 iterations during the training process. This setup aims to train the neural network to learn patterns from the data and improve its classification accuracy by updating the weights based on the training data. The accuracy score at 10,000 iterations was 0.6943, so the number was revised to 15,000 iterations which only improved the accuracy score slightly to 0.6989 (Anaconda, 2024).

The results of the neural networks were validated by building a confusion matrix and classification report with the model set at 10,000 iterations. The number of correct predictions for each class looked to be slightly higher than the matrix from the scores in the KNN model. The classification report shows a significant increase from the KNN model in overall f1 scores of the numbers from the label column. While the numbers zero, one, and six also have the highest f1 scores, all the other numbers show increases in performance with only a couple numbers still lagging with subpar scores.

While neural networks, especially with multiple hidden layers and appropriate tuning, often achieve higher accuracy compared to simpler models like KNN, the accuracy score overall is still relatively low. Using a different number of iterations seemed to not have much of an impact on improving the score, so other factors should be considered. This could be achieved during the data cleaning process by incorporating other methods, while effective preprocessing and feature extraction can improve model performance. Proper tuning of hyperparameters could also be a way to significantly impact accuracy (Anaconda, 2024).

Part 3

Both models are compared and contrasted with the training and testing sets that provide the accuracy scores overall and the scores from the classification report. The neural networks model scores per benchmarking metric on average fared better than the KNN model by around five or six points. The results for each model provide a comprehensive view of how well each performs across different evaluation criteria, allowing for an informed comparison of their effectiveness. The f1 scores for the numbers zero, one, and six in each model are very high, suggesting that children perform better writing those numbers regardless of the model being used. The other numbers are all slightly higher in the neural networks model than the KNN model, all ranging between about five and nine points in the difference between models. The number nine had the worst score in each model, indicating that children have the toughest time writing this number or that the model's proportion of correct predictions is not accurate.

Based on the evaluation of KNN and neural network models, it is recommended to use a neural network (MLPClassifier) because it offers greater accuracy potential, handles complex patterns effectively, and scales well with large datasets. Neural networks are particularly effective for tasks like handwriting recognition, where understanding detailed relationships between pixel intensities and digit shapes is essential. However, if the neural network only achieves 69% accuracy, further enhancements are needed. These improvements should involve refining data quality, experimenting with different model architectures and hyperparameters, adjusting training parameters, and applying cross-validation. If these adjustments still do not lead to better results, it may be worthwhile to revisit simpler models such as KNN or explore other potential solutions.

Conclusion

The incorporation of KNN and neural network models are a powerful method in predicting numbers drawn by children as evident in the models built for this project. While both models were both very not very high in their overall accuracy, they offered a detailed look at how each model can be utilized when working with a large dataset. The recommendations would hopefully determine which model is more suitable for higher prediction values and lead to more resources in children needing improvements with their motor skills.

References

Anaconda Distribution. (2024). Anaconda (Version 4.12.3). Retrieved August 10th, 2024, from https://www.anaconda.com