**Module 5: Letters Motor Skills**

ALY6020 – Predictive Analytics, Northeastern University, Boston

Professor Justin Grosz

CRN: 20387

03/26/2023

Submitted by: Abhinav Jain

**Introduction**

Fine motor abilities and coordination, which are essential for early childhood to develop, are needed for the capacity for writing numbers. The school was keen on implementing a writing assessment as a way to find pupils who may require early intervention for their motor skills. After analysis decided to develop an algorithm capable of predicting which number has been written to accomplish this. To accomplish this, a dataset contains photos of reference numbers by pupils.

 To begin an investigation with a straightforward method, the K Nearest Neighbors (KNN) algorithm then implemented the Random Forest model, and Neural network to keep the dataset useful. This method depends on the concept that similar things were grouped together in high-dimensional space to function. KNN assigns a specific data point a classification based on most of its neighbors' classifications by identifying the k-nearest neighbors to it.

**Data Cleaning**

There are no missing values in the dataset, as you said. Nonetheless, it was always a good idea to examine the dataset again for any missing values and treat them properly. Int64 was the default data type for all columns. Given that this dataset just contains numerical values, this was reasonable. It was crucial to comprehend the distribution of unique values in each column since this can help us understand the data.

In this instance, knowing that the label column has 10 unique values and that the pixel column has 256 unique values at its highest point and 15 unique values at its lowest point. To find any outliers or unusual numbers, it would be a good idea to look at the distribution of unique values in other columns as well.

**Analysis of the implementation of the Prediction Model**

**K Nearest Neighbor (KNN) :**

In KNN, the class of a data point's k-nearest neighbors in the training set was used to calculate its value. K was a hyperparameter that had to be specified before the model was trained. In this instance, k has been set to one and to eleven. In contrast, when k=1, the model merely considers the new data point's nearest neighbor when making a forecast. Because of the model's high sensitivity to noise and outliers in the training set, overfitting may result. Therefore, the accuracy of the model may not be reliable, and it may not generalize well to new data. On the other hand, when k=11, the model considers the class of the 11 nearest neighbors of the new data point to make the prediction. As a result, the model may generalize to new data more effectively and be less susceptible to noise and outliers in the training data. Therefore, the accuracy of the model was higher when k=11 compared to when k=1. Based on the accuracy values you provided, the KNN model with k=11 has a higher accuracy of 65.83%, which was better than the model with k=1 with an accuracy of 61.34%. This suggests that the KNN model with k=11 may be a better choice for classification tasks, as it provides more reliable and generalized predictions.

One of the biggest challenges in developing a KNN model was determining the appropriate value of k. A high value of k can lead to underfitting, whereas a low value of k can lead to overfitting. The value of k must thus be carefully chosen using techniques like grid search or cross-validation. The curse of dimensionality, which occurs when the number of features rises, might affect KNN, making the distance between data points less significant. As KNN was a distance-based technique, the size of the features might have an impact on the model's performance. As a result, before using KNN, the data must be correctly scaled. This can be accomplished by normalization or standardization.

**Random Forest Model:**

The Random Forest model successfully predicted the understanding of the motor skills category for 55.42% of the test cases, indicating a level of accuracy of 55.42%. The ensemble of 500 decision trees that make up the model has a value of n estimators in the range of 500. The Random Forest approach creates a number of decision trees using random selections of the training data and traits. Each tree provides a forecast, and the combined predictions of all trees result in the final prediction. Several trees are used to reduce overfitting and increase the model's generalizability.

The top five features for this Random Forest model in text categorization are features 36, 37, 33, 38, and 34, with relevance scores ranging from 0.110076 to 0.046358, according to the findings of the feature ranking. The fact that feature significance ratings rely on the dataset and the modeling strategy utilized makes them difficult to understand. Also, it's conceivable that the model was overstating the significance of some characteristics since they were highly associated with other features. The fact that the significance scores simply reflect the relative relevance of the feature in the model rather than the direction or causality of the link between the feature and the target variable presents another difficulty. Using 500 n estimators, the Random Forest model was able to predict the knowledge of motor skills category in text texts with an accuracy of 55.42%. Even if this precision might not be the best for all applications, it might nevertheless offer insightful information about the data and act as a foundation for future developments.

**Neural Network:**

To classify texts, the Multilayer Perceptron (MLP) classifier was employed. Using varying settings, two MLP classifiers were created.

The original MLP classifier featured 20 hidden layers, using the "adam" solver with a learning rate of 0.01 and the "relu" activation function. 20,000 training iterations were performed on the model. 68% of the test set was correctly classified by this MLP classifier. The second MLP classifier utilized the 'lbfgs' solver with a learning rate of 0.01 and the 'relu' activation function, which had 15 hidden layers, and a learning rate of 0.01. 15,000 iterations were used to train the model. For the test set, the accuracy of this MLP classifier was 66.77%. Determining the ideal number of hidden layers and selecting suitable hyperparameters like the solver, learning rate, and activation function may be difficult when employing MLP classifiers. MLP classifier training can also be computationally costly, especially for big datasets or complicated models. For this text classification challenge, the MLP classifiers performed moderately overall, with accuracies ranging from 66.77% to 68%. The performance of the model may be enhanced by further hyperparameter adjustment and optimization.

|  |  |
| --- | --- |
| Prediction Matrix | Accuracy (%) |
| KNN | 65 |
| Random Forest | 56 |
| Neural Network | 68 |

**Conclusion:**

The study used the K-Nearest Neighbor (KNN), Random Forest, and Multilayer Perceptron classification models (MLP). By considering the class of its k-nearest neighbors in the training data, the KNN model predicts the class of a new data point. The forecasts from each decision tree are combined to get the final prediction by the Random Forest model. In order to determine the class of a new data point, the MLP model makes use of an artificial neural network. The MLP classifier with 20 hidden layers comes in second with an accuracy of 68%, while the KNN model with k=11 has the greatest accuracy at 65.83%. The accuracy of the Random Forest model was 55.42%. The accuracy of the MLP classifier with 15 hidden layers was 66.77%.

To further investigate the model's performance, it was recommended to evaluate it on new data and explore other modeling approaches if necessary. Additionally, adding more features to identify the color scheme could lead to more effective results. Taking these actions can help to improve the model's accuracy and reliability in a short time.

**Appendix:**

**Feature Importance**

**Text

Description automatically generated**

**Chart, histogram

Description automatically generated**

**Random Forest Tree**

**Graphical user interface, application

Description automatically generated**