Logo

Description automatically generated

ALY6020 Module 5 Project

**Authored by: Prateek Singh**

Course: ALY6020 – Predictive Analytics

Submission Date: 05/21/2022

**Instructor’s name: Prof. Kristen Drobnis**

# Introduction

Text categorization has become one of the most significant challenges for web searching, datamining, spam filtering, sentiment analysis, decision making, and other applications as the amount of data gathered and stored has increased. The primary goal of text mining or natural language processing (NLP) is to categorize information into classes. It is required for good textual data management. Medical text classification is one use of text classification where health practitioners spend a significant amount of time reading notes to detect critical concerns and comprehend the patients' state. Many automatic text categorization approaches exist, including Nave Bayes, SVM, KNN, decision trees, and artificial neural networks (ANN).

A dataset is used in this project to identify letters/text using two different models. Text classification is accomplished using KNN and neural network models.

Following the collection of the necessary data, each document is pre-processed, and the preprocessing stage includes document conversion, stemming, and indexing. The dataset contains 42 variables and 42000 observations. The data is made up of variables that define the pixel position of the text, and the label represents the text itself, which is identified using the pixel position. The problem statement here is to assist in identifying the text written by students to assist in identifying students who require assistance with motor skills. The model's goal is to identify a number. The model's goal is to identify a number based on pixel information, which will aid in the resolution of the business problem.

# Analysis

## Data Profiling & Cleaning

This section discusses about the dataset, data preprocessing, data cleaning and implementation and results of the KNN and Neural network classification models.

The dataset consists of text labels (determined in numbers) and the corresponding pixel positions information. We want to classify the text labels according to the pixel positional values presented in the dataset. We will be creating and testing couple of classification models to predict the same. The dataset has 42 variables and 42000 observations.

A screenshot of a computer

Description automatically generated with low confidence

Fig 1: Dataset Structure (First 5 Rows)

There is no need of the dataset cleaning as there are no missing values present and all the variables are equally important as they are the text identifiers and represents pixel position of the text labels. Label variable in our dataset is our Target variable and rest all the variables are independent variables.

Table

Description automatically generated

Fig 2: Number of Null Values

Because of the fact that the pixel variables are the positional representation of the text labels, there is no need of checking the normalization condition of the columns. However, we can see that there are few duplicate rows present which can be possible and no treatment for the same is required.



Fig 3: Duplicate rows in the dataset

Before proceeding to build the model, we created test and train set for training our train dataset and testing it on test dataset to find how accurate our models are. I have set random seed value as 42 so that every time the code is run same set of samples are created and hence the results are same throughout multiple runs. Random state value is also set at 30.

Because we will be building Neural network model it is very critical to scale the data. We have used StandardScaler to normalize the values around mean. It is very important for our dataset to not have any outliers or extraneous values when building a neural network model.

## Correlation Matrix Heat Map

According to the correlation Matrix Heat Map, it was observed that all the pixel values are strongly correlated with neighboring values of the pixel variable. This assure us that all the pixel values are highly important and together only can accurately determine the text labels.

A picture containing text, light, traffic, lit

Description automatically generated

Fig 4: Heat Map for the Correlation Matrix

## K-Nearest Neighbor Model

Before starting to fit the model, it is very vital to define the best value of K which will be used to fit the model. We do not want to use very low value of K as it will create underfitting in the model and might not reflect the true picture. Low value of K leads to model predicting sometimes based on the outlier and not paint the true picture. However, we also do not want to use very high value of K as well as it will create a model which will be true for almost all the testing scenarios and hence won’t reflect the true neighbors and be able to classify unknown datapoints.

I ran the GridSearchCV to find the optimum value for K which came out to be 12 for our training dataset. We should point that it usually takes huge amount of time to get the results from the GridSearchCV as it looks for the optimum value from the range of 1 to 50 which we have set and give us the output with the best K-value.



Fig 5: Best Value for K

Upon getting the best value for K, we went ahead to fit our model which presented us the accuracy of 65.17% and took 8 seconds and 457665 microseconds to run the model. The confusion matrix obtained is 10 x 10 which gives us the precision and recall of the model. We can see that the model precision declines as the text label increases with few fluctuations for some text labels, hence our model is better predicting the lower labels than the higher labels. The precision of the model comes out to be 0.84 to 0.47 and the Mean Absolute Error Value as 1.15. Our model best predicted the text label 6 at 84% accuracy and the least for label 9 at 47% accuracy.

|  |  |
| --- | --- |
| Accuracy | 65.17% |
| Mean Absolute Error | 1.15 |
| Precision range | 0.84 - 0.47 |
| Performance | 8 seconds and 457665 microseconds |

Table

Description automatically generated

Fig 6: Confusion Matrix & Classification Report of KNN Model

## Neural Network Model

As we all know Neural Networks are complex models and try to self-learn along the process. Upon fitting the model, the accuracy comes out to be 68.61% with a mean absolute error as 1.03. We observed in NN model as well the most accurately predicted text label was 6 and the least was 9.

|  |  |
| --- | --- |
| Accuracy | 68.61% |
| Mean Absolute Error | 1.03 |
| Precision range | 0.89 - 0.44 |
| Performance | 99 seconds and 849075 microseconds |

Table

Description automatically generated

Fig 7: Confusion Matrix & Classification Report of ANN Model

## Comparison of Models

|  |  |  |
| --- | --- | --- |
|  | KNN | ANN |
| Accuracy | 65.17% | 68.61% |
| Mean Absolute Error | 1.15 | 1.03 |
| Precision range | 0.84 - 0.47 | 0.89 - 0.44 |
| Performance | 8 seconds and 457665 microseconds | 99 seconds and 849075 microseconds |

We got the better accuracy for Neural Network Models when compared with KNN model. The mean absolute error value also got reduced in Neural Network model with better precision range in comparison to KNN. Only thing was the performance of the model where KNN performed much better than ANN, however processing time to calculate the best value of K was much higher. Hence overall we can conclude that Neural networks classifies and predicts the text labels more accurately and precisely and yields the better result than KNN model. The text classification problem can be best solved by Neural Network models.

# Conclusion

1. Neural Network model is more accurate, precise, and faster than KNN model.
2. Neural Network model has lower MAE value of 1.03
3. Neural Network yields better results than KNN model
4. Neural Network is better for text classifications problems

# Reference

1. Nikles Donges(2021), Guide on Random Forest Algorithm. Retrieved August 15, 2021 from <https://builtin.com/data-science/random-forest-algorithm>
2. Onel Harriison (2018) , Machine learning Basics with K-Nearest neighbor Algorithm Retrieved August 15, 2021 from <https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761>
3. Baati, K., & Mohsil, M. (2020). Real-Time Prediction of Online Shoppers’ Purchasing Intention Using Random Forest. Artificial Intelligence Applications and Innovations: 16th IFIP WG 12.5 International Conference, AIAI 2020, Neos Marmaras, Greece, June 5–7, 2020, Proceedings, Part I, 583, 43–51. <https://doi.org/10.1007/978-3-030-49161-1_4>
4. Korstanje, J. (2021, April 10). The K-nearest neighbors (knn) algorithm in Python. Real Python. Retrieved May 21, 2022, from <https://realpython.com/knn-python/>
5. Sklearn.neural\_network.MLPClassifier. scikit. (n.d.). Retrieved May 21, 2022, from <https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html>

# Appendix

Note: Code is attached separately.