CP 218: Theory and Applications of Bayesian Learning

Project 1 Report:Team MAVERICK

KUNAL WASNIK 22393 | SUBHASIS PANDA 22992 | PRAVEEN PRASAD HANDIGOL 23062 |

1 Problem Statement and Objectives

Social networks like Twitter have become a significant part of modern communication, with users following one another to stay updated with their activities. In this study, we explore the task of predicting the location of the users using machine learning models. We are provided with a dataset containing historical information about 6,000 users with User ID, number of tweets, timestamp of the tweets and a tweet posted on the platform by the user. Our goal is to leverage this data to train various machine learning models and then apply them to predict the location of the users for a test dataset of 2,000 users.

2 Data Description

The training dataset comprises 6,000 unique User IDs and their location in the form of 'Latitude' and 'Longitude'. The dataset includes information about one tweet posted by the user, number of tweets posted and the timestamps of posting of the tweet.The test data includes same information as the training dataset except the location information, for 2000 different unique users.

3 Methodology

Task 1:Data Visualisation:The makers on the map were plotted using the latitude and longitude information given in the training dataset and it was observed that most of the tweets were from the US region.

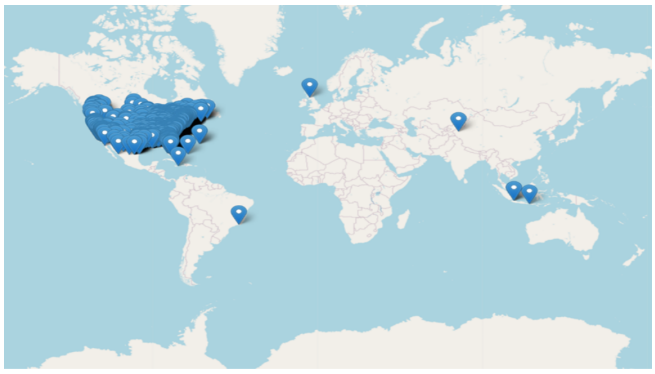

Figure 1: Locations of training data

Task 2: **Data Preprocessing**:As we are aware that the tweet content(tweets by users)can be anything,so processing it was necessary .At first ,stop words(those local words that are location-irrelevant) were removed.Other irrelevant subjects like punctuation,mentioned usernames,outliers,hashtags, were also removed.Spelling correction was done using TextBlob library.It is crucial to understand the meaning of a user's messages,for which lemmatization was done.Short forms and abbreviations were preserved to get the local language information.

Task 3: **Feature Extraction**:To train the dataset ,some relevant features are needed.Some are given below:
1- **Timestamp feature extraction**:. In the dataset, all posting times are recorded in UTS. For this, we explored a method to augment temporal information for model training. This method involved converting each timestamp into seconds and subsequently summing the seconds across all tweets posted by a user. This process effectively

consolidated the temporal activity of each user into a single cumulative timestamp representation.
A better method to facilitate temporal analysis is to discretize the timestamps into bins of 30-minute intervals.After binning a UTS day into time slots of equal length, users are viewed as distributions of tweet posting times.

2- **Vectorization of tweets**:Most machine learning algorithms operate on numerical data. Tweets, being text data, need to be converted into numerical format for the algorithms to process them.Vectorization allows for the extraction of meaningful features from text data. In the case of tweets, features could include words, phrases, or other linguistic patterns that may be indicative of location.For tweet content, we employed various techniques such as stemming (using Porter and Snowball algorithms) and lemmatization (Spacy and WordNet) to derive root words from different words in the tweets. Subsequently,We used TF-IDF (Term Frequency-Inverse Document Frequency) technique which represent the text in a lower-dimensional space while retaining important semantic information. We experimented with an alternative approach tailored specifically for LSTM models.Utilizing the embedding method available in the TensorFlow Keras library, we employed one-hot encoding to represent each unique word in the dataset. Each tweet was then converted into a fixed-length vector of dimension 40, where each word within the tweet was represented by a vector of dimension 10. This technique enabled us to capture the semantic relationships between words within the tweets, enhancing the LSTM model's ability to understand and interpret textual data.

Task 4:**Model Training**:We adopted a comprehensive approach to model training, utilizing a diverse array of machine learning algorithms including lasso regression, decision tree regressor, random forest regressor, XGBoost regressor, artificial neural networks (ANN), and long short-term memory (LSTM) networks. Before feeding the data into the models, feature scaling was performed using MinMaxScaler to ensure consistent scaling across features.

4 Results and Discussions

All the models were run for various hyper parameters and the ones which gave the best performance were chosen. A brief description of the models are given below.
1.Decision Tree Regressor: Max Depth = 5
2.Lasso Regression: alpha = 0.02
3.XGBoost Regressor: Max Depth = 3, n_estimators = 100, learning rate = 0.1
4.Random Forest Regressor: Max Depth = 10
5.ANN: 4 hidden layers with decreasing number of neurons (128,64,32,16)

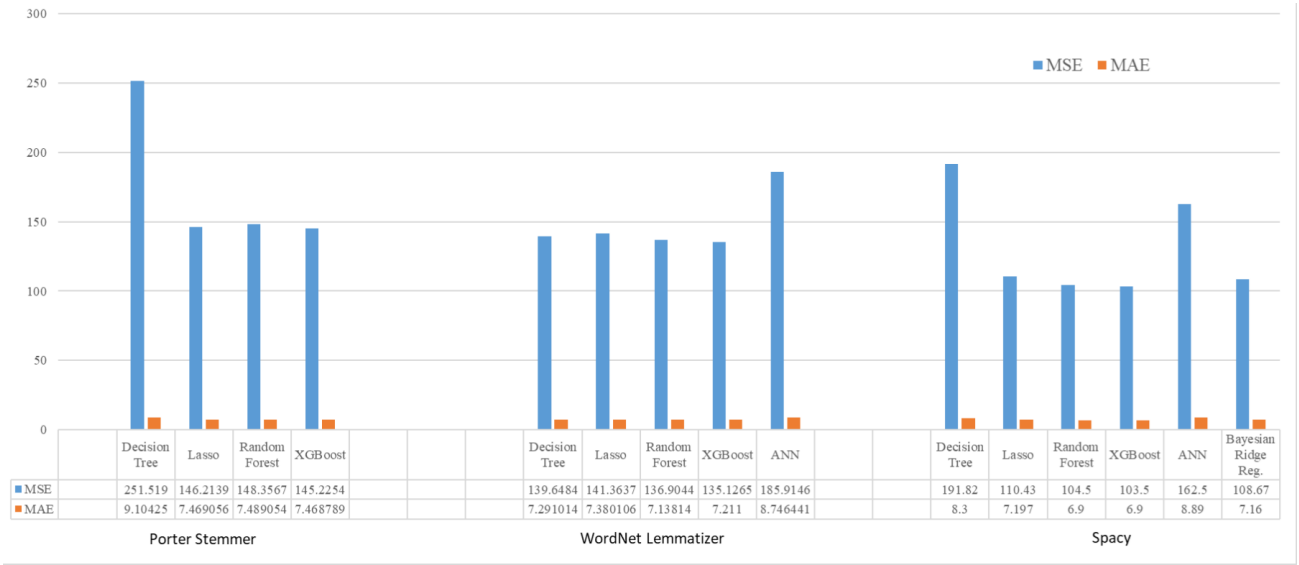

Figure 2: Accuracy of different text processing methods

The above figure shows MSE and the MAE values for various models trained considering various stemmers and lemmatizers. The time bins method was used to compare the validation accuracy between the models. It is seen that models which were trained using Porter Stemmer had lower validation accuracy compared to WordNet and Spacy Lemmatizers. There is not much difference in the performance between Spacy and WordNet. Amongst all the models, XGBoost Regressor was the best performing model. This is because XGBoost operates on the principle of gradient boosting, which involves sequentially adding decision trees to the ensemble. It minimizes the loss function by adding decision trees that minimize the residuals from the previous step. It also combines the strength of the gradient-based optimization techniques using second-order Taylors Series and regularization to produce highly accurate and robust regression models. Random Forest Regressor was the next best performing model. Random Forest Regressor is known for its simplicity, scalability, and ability to handle high-dimensional datasets. During prediction, each decision tree in the Random Forest independently produces a prediction. For regression tasks, the final prediction is computed by averaging the predictions of all trees in the ensemble. This averaging process helps to reduce overfitting and improve the stability of the predictions.

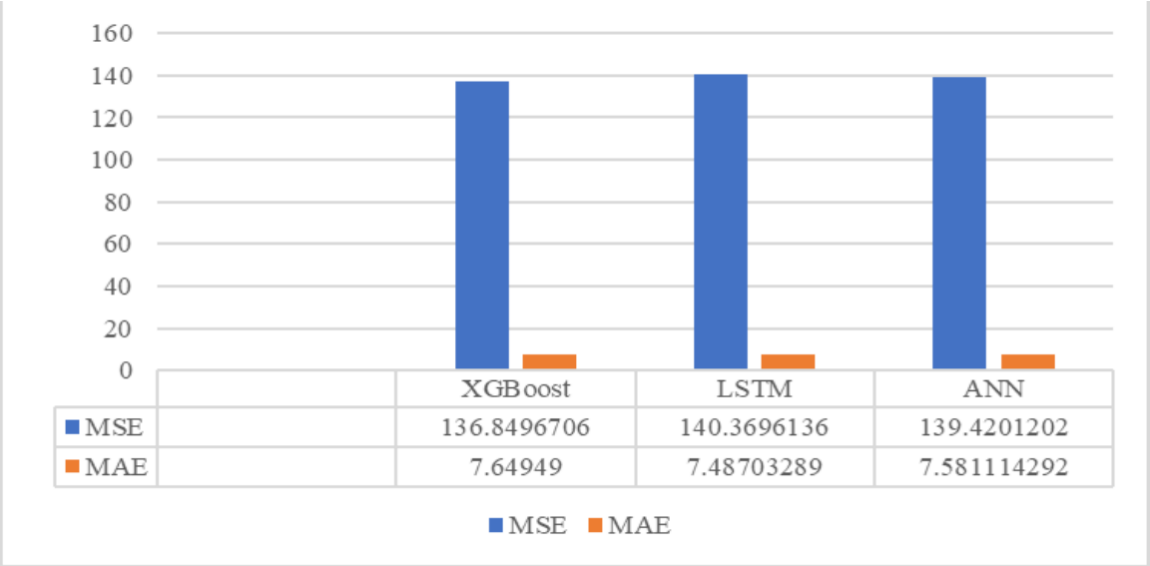

Figure 3: Accuracy with word Embedding

A different approach of Embeddings was tried out for specifically training LSTM model. The results were compared with the best performing XGBoost model and ANN. It can be seen that the performance of the nueral networks were improved on introducing the embeddings approach. This is because each unique word is represented with a unique number by using one hot encoder and by introducing dimensions to the words, it helps the model to understand the pattern in the data and thereby help in predicting the location of the user based on the tweet.

5 References

- Took common abbrevations data from `https://github.com/grrrrnt/notion-emotion-twitter/blob/main/abbreviations.csv`
- X. Zheng, J. Han and A. Sun, "A Survey of Location Prediction on Twitter," in IEEE Transactions on Knowledge and Data Engineering, vol. 30, no. 9, pp. 1652-1671, 1 Sept. 2018, doi: 10.1109/TKDE.2018.2807840.
- Ajao, Oluwaseun Hong, Jun Liu, Weiru. (2015). A Survey of Location Inference Techniques on Twitter. Journal of Information Science. 41. 855-864. 10.1177/0165551515602847.