Samuel Reade

Professor Hosseinmardi

Communications 188C

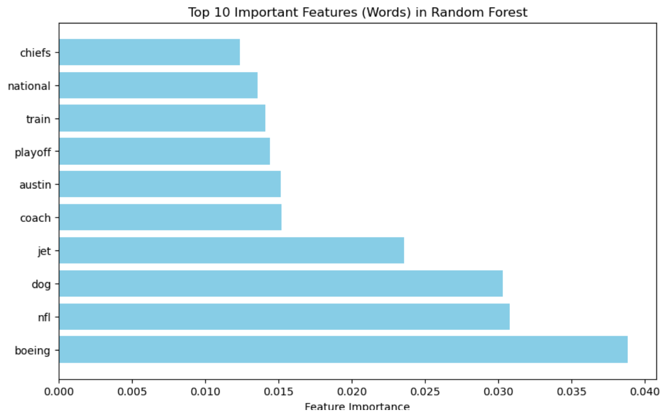
May 22nd, 2025

Homework 3 Report

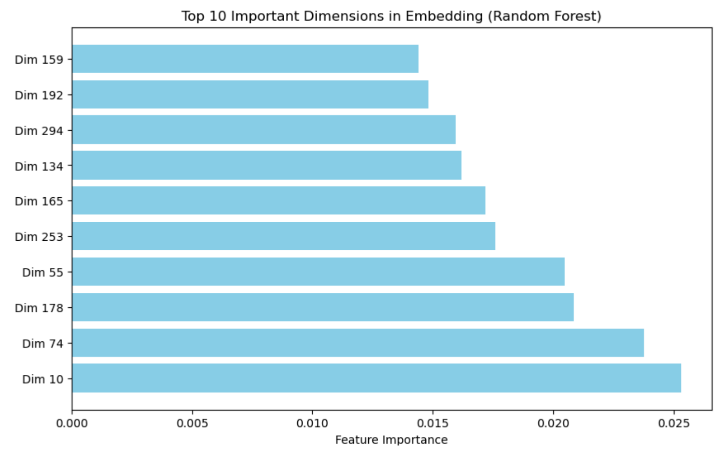
After creating a virtual environment, downloading and calling all the libraries that were needed to complete the assignment, reading in the 'news_AP.csv' file was the next step. This was then converted into a pandas dataframe. Once the data was in a pandas data frame the code performs text clustering and evaluation on a dataset of news titles. First, it defines a function 'preprocess_texts' that takes a list of text strings and applies several preprocessing steps. This includes converting all letters to lowercase, removing punctuation, tokenizing the text, removing stop words, and filtering out rare words, words that show up only once. The cleaned and filtered tokens are then joined back into strings. This function is applied to the 'title' column of the 'news' data frame enabling more reliable string data, which would produce better clustering results.

Next, the code vectorizes the preprocessed titles using TF-IDF, 'TfidfVectorizer', turning the text data into a numerical matrix which was initialized as 'titles_tfidf'. In the matrix each row represents a title and each column corresponds to a unique term weighted by its TF-IDF score. The resulting shape and array are printed for inspection.

After the titles were preprocessed the code applied text classification using a Random Forest model trained on the TF-IDF vectors. First, it sets the target variable to the 'topic' column from the 'news' dataset. It then splits the data into training and test sets using an 80/20 split. A grid of hyperparameters is defined for tuning the Random Forest, 'n_estimators' and ‘max_depth’. Using ‘GridSearchCV’ with 5-fold cross-validation, the model selects the best combination of hyperparameters based on accuracy. After training, the best model is used to predict on the test set, and a classification report and overall accuracy are printed to evaluate performance. Finally, the feature importances from the trained model are extracted and used to identify and visualize the 10 most influential words in predicting the topics with a bar chart.



The next step was to repeat the same process but with sentence embeddings. The code uses embeddings as input features to train the second Random Forest classifier. It begins by loading a pre-trained SentenceTransformer model, ‘all-MiniLM-L6-v2’, to convert the news titles into vector representations, ‘titles_embed’. The data is split into training and test sets using an 80/20 ratio, and a hyperparameter grid is defined to tune the Random Forest using ‘GridSearchCV’ with 5-fold cross-validation. After identifying the best-performing model, predictions are made on the test set and evaluated using precision, recall, F1-score, and overall accuracy. The code then extracts and plots feature importances. It analyzes and visualizes which dimensions of the sentence embeddings were most important for the Random Forest classifier. Since sentence embeddings are dense vectors without interpretable word features, each feature corresponds to a numerical dimension like ‘Dim 1’ or ‘Dim 2’. The code gets the feature importances from the trained Random Forest model and sorts them in descending order to identify the top 10 most influential dimensions. These top dimensions are then visualized in a horizontal bar chart. This provides insight into which parts of the embedding space contributed most to the model's predictions.



The TF-IDF based model outperformed the embedding based model across nearly all evaluation metrics. It achieved an overall accuracy of 95%, compared to 92.5% for the embedding model. The TF-IDF model also had a higher macro-averaged precision, 0.9583 vs. 0.9444, recall, 0.9427 vs. 0.9083, and F1-score, 0.9465 vs. 0.9201. At the class level, the TF-IDF model performed particularly well on the "Transportation" and "Sports - Football" categories, achieving perfect or near-perfect scores. The embeddings still produced good results and may generalize better in more complex contexts. The sparse TF-IDF representation proved more effective on this particular dataset of short news headlines. The supervised approach was far more effective than the unsupervised approach. The average accuracy for the supervised approaches was 93.75%, while the average accuracy for the unsupervised approach was only 37.94%. Overall, the supervised Random Forest approach was much more effective than the unsupervised k-means clustering approach.