Daniel Thang, Reed Balentine, Khiran Kumar

Data 606 Capstone Project – Final Report

Introduction – Purpose

The purpose of our project is to build a machine learning model that can reliably predict who the speaker is for lines of dialogue from NPR Radio Transcripts. It is a text classification project that focuses on utilizing NLP methods. It is estimated that around 80% of all information is unstructured, with text being one of the most common types of unstructured data (MonkeyLearn, n.d.). Due to the messy nature of text, “analyzing, understanding, organizing, and sorting through text data is hard and time-consuming, so most companies fail to use it to its full potential.” (MonkeyLearn, n.d.). According to a 2019 survey by Deloitte, only 18% of organizations were taking advantage of their organization’s unstructured data, which displays that there is a huge untapped resource with the potential to create a competitive advantage for companies that figure out how to use it. (Harbert, 2021). So, using text classifiers and NLP techniques, companies can automatically structure all manner of relevant text, from emails to social media to, in our case, NPR media transcripts, in a fast and cost-effective way. This allows companies to save time “analyzing text data, automate business processes, and make data-driven business decisions” (MonkeyLearn, n.d.).

Additionally, the benefit of using machine learning techniques for text classification include:

- Scalability – since manually analyzing and organizing is slow and can result in less accurate output, ML will allow you to automate the process.
- Real time-analysis – which allows you to take immediate actions during critical situations
- Consistent criteria – which allows you to eliminate human errors and subjectivities

Lastly, NLP methods will also be vital to our project since it helps resolve ambiguity in language and adds useful structure to the data for many downstream applications, like text classification. (MonkeyLearn, n.d.). Being able to automate the process of identification through the patterns of text data is a difficult task but it is a great way to analyze text data to its full potential since the methods used can be applied to other text data that is probably more structured than media transcripts, so their accuracy will be much higher.

Dataset Explained

Data source:
https://www.kaggle.com/datasets/shuyangli94/interview-npr-media-dialog-transcripts

The goal of this project is to build a machine learning model that can reliably predict who the speaker is for a given line or lines of dialogue. The dataset will have a speaker column, which consists of radio hosts, and the utterance column, which are the transcripts or lines spoken by the radio host. Originally, there are a total of 3.2 million spoken lines pulled from 105,000 Episodes that aired from 1999-2019. However, after cleaning and applying preprocessing methods, the total rows will decrease.

We chose this dataset as a challenge because due to the large size along with the extremely messy nature of the text data which would make it difficult to clean, process, and get a high model accuracy. The model will be predicting the speaker/host from the utterance/texts, which is much harder to achieve as there isn't a similar line of topic that each host stick to. So, getting a decent model accuracy with a dataset like this will prove two points:

1. Prove the power and usefulness of machine learning techniques in efficiently automating the process of text classification and analysis with minimal time and effort.
2. Prove that if such a messy and unstructured dataset can be processed using these simple classification models, we can firmly state that level of accuracy and efficiency that can be achieved with text datasets that are relatively less messy such as emails or social media will be much higher.

EDA and Data Cleaning

- All string lowercase
- Map with host-file (file that has NPR host name and host_id)
- Only keep rows with more than or equal to 100 characters in the utterance/line.
- Only keep the top 5 host/speaker
- Drop any rows if the string in the utterance column starts with “undefined”.

- Use SampleBy() method to limit the counts of host "Neal Conan" due to it being too high compared to other hosts. (Balance dataset)
- Drop all NaNs
- Drop all columns except the speaker and utterance column

Note: This project will consist of two notebooks with different preprocessing, feature extraction, and transformation methods. They will also have different ML models with varying model accuracy rates. However, the cleaning methods will be applied to both notebooks. The two notebooks serve the purpose to display the various NLP methods and classification models that can be utilized in different arrangements.

Notebook I: Databricks Pyspark

Preprocessing, Feature Extractors, and Transformers Pipelines

- For preprocessing, an open-source python library called Gensim is used to perform basic preprocessing methods such as stripping tags, punctuation, white spaces, and numbers. It also removes stop words like "the, or, a, what", so it will focus on the important words and information.
- For feature transformers, regex tokenizer is used to splits a text into words or sub-words, and string indexer is used to create labels from the speakers that we will focus on.
- For feature extractors, count vectorizer is used to vectorize text and perform basic preprocessing like lemmatization, stemming, and vectorization.

After applying those pipelines to the data frame, the data is split for training and testing data that will be used for the actual classification models. To note, there are about 120,000 rows in the training set and around 30,000 in the test dataset.

Models and Evaluation

- Logistic regression:

  Model Accuracy: 65.53%

  Precision and Recall: 67%

  F1: 66%
- Naive Bayes Classifier

Model Accuracy: 64.65%

Precision and Recall: 65%

F1: 64%

- Random Forest Classifier

  Model Accuracy: 33.08%

  Precision: 62%

  Recall: 33%

  F1: 26%

- Logistic regression with cross validation

  Model Accuracy: 66.96%

  Precision, Recall, and F1: 67%

- ❖ Model Evaluators that were used
  - o Spark Ml: MulticlassClassificationEvaluator
  - o Scikit-learn: classification_report

Notebook II: Python

Feature Extractors and Transformers Pipelines

- ❖ For feature transformers, the open-source NLP python library, NLTK, is mainly used. The methods include regex tokenizer, which basically splits text into words or sub-words, English stop word remover, and wordnetlemmatizer, which provides semantic relationships between its words.
- ❖ For feature extractors, TFIDF vectorizer is used to give numerical representations of words and provide many other preprocessing methods that we might have missed.

Models and Evaluation

- MultinomialNB

  Model Accuracy: Around 90-98% (Suspicion of overfitting)

  Precision and Recall: 51%

  F1: 49%

- Best Parameters for MultinomialNB: TFID grid search

  alpha: 0.1

fit_prior: true

Tfid_ngram_range: [1, 2]

- Other Models: (Through Randomized Search CV)

  GaussianNB: 21.1%

  ComplementNB: 29.6%

  BernoulliNB: 32%

  Hist Gradient Boosting Classifier: 35.3% (took a long time and did not complete the search)

  Random Forest Classifier: 37% (also did not finish the search)
- Model Evaluators that were used

  Classification report

Failed Methods

Feature Extractors: For vectors, we tried TF-IDF and Word2Vec, but they got a relatively low model accuracy rate compared to CountVectorizer, so they were removed.

Feature Transformers: Normalizer and standard scaler were used but they did not have much impact as in one run, they decreased the model accuracy by around 0.5%, so they were removed.

Classification Models:

- BERT: 14-15%

  One notebook was dedicated to the BERT model, as the processing methods are much different fromother models. BERT itself already has a lot of the inbuilt cleaning and preprocessing methods, so we did not have to do much cleaning other than balancing the dataset as it is a necessary component of running a BERT model. We used a pre-trained BERT model imported from sckit-learn, and it took many hours to run in a local host, but only gave around 14-15% accuracy rate. So, we decided to not use the notebook but added it to our github repo to display our project's progress.
- Multilayer Perceptron Model: 20%

  The multilayer perceptron model did have the potential to reach a decent accuracy model rate as it is a Neural Network model. However, this was the last model to be used, so due to time constraints, it was not smart to invest time as the accuracy started out quite low. It would be more beneficial to focus on increasing the model accuracy of our main models that displayed great accuracy from the start.

Overall, we think it will be extremely difficult to increase it above 67%, especially when working with such a messy data and trying to classify it with 5 labels. Honestly, how likely do you think an average person can predict the speaker based on a given text? It will most likely be lower than 67% that's for sure. So, we are satisfied with our model accuracy despite the many failed methods and errors that occurred.

Conclusion

To conclude, we hope you can see the usefulness of applying ML and NLP methods in the big data space. Since we had a decent model accuracy with a dataset like this, you can imagine how much better the model accuracy will be using datasets with a bit more structures like medical reports, emails, online reviews, news reports, or work documents. Additionally, think about projects where you need to classify only two or three labels like an email spam detection project or running a sentiment analysis to understand the overall attitude and emotional tone of a text. So, not only is the model allowing the business to run large-scale text analysis on big and diverse datasets but run with a more objective and accurate lens that is void of human error.

Another big advantage of this project is the automation aspect that will allow you to streamline processes and reduce the time, effort, and cost that your business might face when classifying or organizing big data. For example, anytime there is new data, you can use the model to automatically classify that new data and streamline it to automatically organize that data to some sort of system or directory based on the model's output category. Another example allows you to use the model to analyze and sort social media data or customer feedback through various categories like by topics or sentiments which will allow our business to understand how customers use our product or service and why they may be satisfied or unsatisfied, which will impact our marketing strategy or simply improve customer satisfaction rate.

All in all, this project works to prove that there is this untapped potential for companies to create competitive advantages from the fresh analysis that can be made based on the classified dataset and automate business processes. Thank you.

Works Cited

Harbert, T. (2021, February 1). *Tapping the power of unstructured data*. MIT Sloan. Retrieved December 10, 2022, from https://mitsloan.mit.edu/ideas-made-to-matter/tapping-power-unstructured-data

Text classification - importance, use cases, and process. Shaip. (2022, November 18). Retrieved December 10, 2022, from https://www.shaip.com/blog/text-classification-importance-use-cases-and-process/

Text classification: What it is and why it matters. MonkeyLearn. (n.d.). Retrieved December 10, 2022, from https://monkeylearn.com/text-classification/