Identifying the Speaker from the Script of The Office: a Machine Learning & Text Mining Use-case

Student: Patrick Walsh
Professor: Dr. Ami Gates
School: Syracuse University
Date: 3/21/2024

INTRODUCTION

The Office, an American sitcom, has gained immense popularity for its unique humor and relatable characters. In this text-mining analysis, we delve into the vast dialogue dataset of The Office to uncover patterns and insights that shed light on the dynamics between characters and the underlying themes of the show.

This project will utilize The Office dataset from Kaggle, consisting of every line of dialogue from every character in the TV show The Office. The goal will be to text-mine this dataset, focusing on the speaker and the line of dialogue columns to train models that can read a line of dialogue and predict who the speaker is.

ANALYSIS

Dataset

The dataset consists of dialogue lines from The Office TV series, along with the corresponding speaker's name. It was initially loaded from a CSV file and then cleaned and preprocessed for analysis.

The raw dataset can be found here: https://www.kaggle.com/datasets/nasirkhalid24/the-office-us-complete-dialoguetranscript

```python
import pandas as pd

data = "C:/Users/Patrick/Syracuse_courses/IST_736/Final_Project/The-Office-Lines-V4.csv"
raw_df = pd.read_csv(data)
display(raw_df)
```

	season	episode	title	scene	speaker	line	Unnamed: 6
0	1	1	Pilot	1	Michael	All right Jim. Your quarterlies look very good...	NaN
1	1	1	Pilot	1	Jim	Oh, I told you. I couldn't close it. So...	NaN
2	1	1	Pilot	1	Michael	So you've come to the master for guidance? Is ...	NaN
3	1	1	Pilot	1	Jim	Actually, you called me in here, but yeah.	NaN
4	1	1	Pilot	1	Michael	All right. Well, let me show you how it's done.	NaN
...	...	...	...	...	...	...	...
54621	9	24	Finale	8153	Creed	It all seems so very arbitrary. I applied for ...	NaN
54622	9	24	Finale	8154	Meredith	I just feel lucky that I got a chance to share...	NaN
54623	9	24	Finale	8155	Phyllis	I'm happy that this was all filmed so I can re...	NaN
54624	9	24	Finale	8156	Jim	I sold paper at this company for 12 years. My ...	NaN
54625	9	24	Finale	8157	Pam	I thought it was weird when you picked us to m...	NaN

54626 rows × 7 columns

Data Preparation

Before diving into the riches of dialogue, we took care to clean and polish our dataset. We stripped away extraneous details, focusing solely on the 'speaker' and 'line' columns. By gauging the frequency of each speaker, we identified the leading voices that echo throughout the series. Filtering out sparse dialogues and short lines, we ensured our analysis revolves around substantial conversations, brimming with insights and laughter.

The data preparation and data cleaning process consisted of the following steps:

- Loaded the raw dataset from a CSV file.
- Kept only the 'speaker' and 'line' columns.
- Calculated the frequency of each speaker to identify prominent speakers.
- Filtered the dataset to keep only rows where the speaker appears at least 1000 times.
- Removed rows with very short lines (3 words or less) to focus on meaningful text data.
- Saved the cleaned dataset to a new CSV file for further processing.

	speaker	line
0	Michael	All right Jim. Your quarterlies look very good...
1	Jim	Oh, I told you. I couldn't close it. So...
2	Michael	So you've come to the master for guidance? Is ...
3	Jim	Actually, you called me in here, but yeah.
4	Michael	All right. Well, let me show you how it's done.
...	...	...
28939	Kevin	No, but maybe the reason...
28940	Erin	How did you do it? How did you capture what it...
28941	Darryl	Everyday when I came into work, all I wanted t...
28942	Jim	I sold paper at this company for 12 years. My ...
28943	Pam	I thought it was weird when you picked us to m...

28944 rows × 2 columns

	LABEL	00	000	001	00983	01	03	05	10	100	...	zombie	zombies	zone	zoo	zoom	zoppity	zoran	zuckerberg	zuckerberged	zwarte
0	Michael	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	Jim	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	Michael	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	Jim	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	Michael	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
28939	Kevin	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
28940	Erin	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
28941	Darryl	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
28942	Jim	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
28943	Pam	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

28944 rows × 15998 columns

Models

Our arsenal of models mirrors the diversity of The Office's characters, each bringing its unique perspective to the table:

Naive Bayes Model: An intuitive approach that captures the essence of dialogue patterns and speaker quirks.

Support Vector Machines (SVM): Harnessing the power of mathematical boundaries to distinguish between character voices with precision.

Random Forest Classifier: Like a forest of decision trees, picking up nuances and subtleties in dialogue delivery.

Neural Network (MLPClassifier): Mimicking the complexity of human thought processes, diving deep into the intricacies of dialogue analysis.

LSTM Model (Word Embeddings): Delving into the soul of words, creating embeddings that capture the essence of each line and speaker.

Naive Bayes Model

- Trained a Multinomial Naive Bayes model on the cleaned text data.
- Evaluated the model's performance using accuracy and confusion matrix.

Support Vector Machines (SVM)

- Scaled the training and testing sets using MinMaxScaler for SVMs.
- Created an SVM classifier with a linear kernel.
- Fit the SVM model with sample weights calculated based on class imbalance.
- Predicted speaker labels using the trained SVM model.

Random Forest Classifier

- Trained a Random Forest Classifier on the text data.
- Made predictions using the trained Random Forest model.

Neural Network (MLPClassifier)

- Implemented a Multi-Layer Perceptron (MLP) classifier with specified parameters.
- Trained the MLP classifier on the text data and evaluated its predictions.

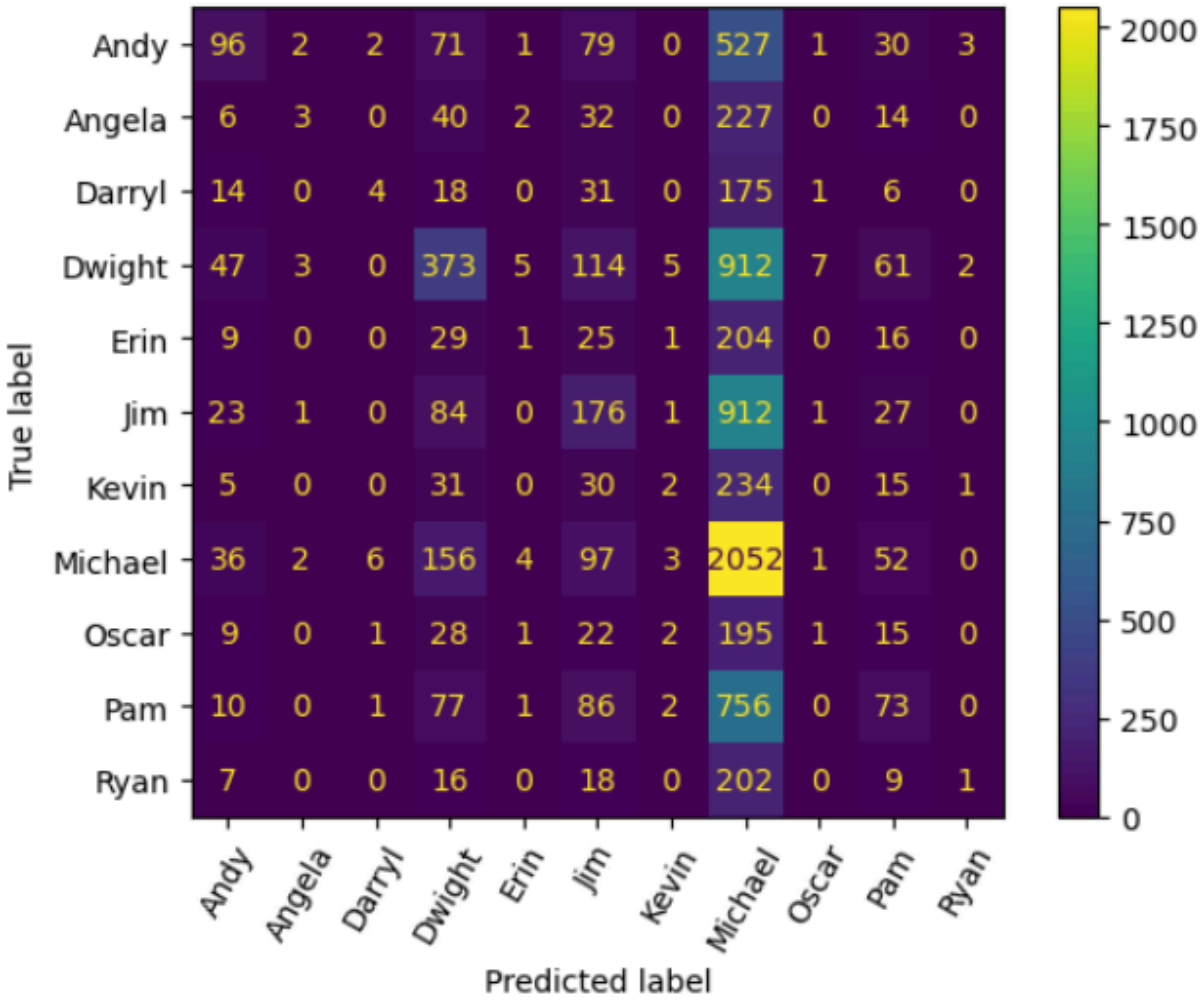
LSTM Model (Word Embeddings)

- Loaded the cleaned lines dataset.
- Tokenized the text and trained a Word2Vec model to create word embeddings.
- Preprocessed the text data for the LSTM model by tokenizing, padding sequences, and encoding labels.
- Built an LSTM model using Keras with an embedding layer, LSTM layer, and dense layer.
- Compiled and trained the LSTM model on the training data.

RESULTS

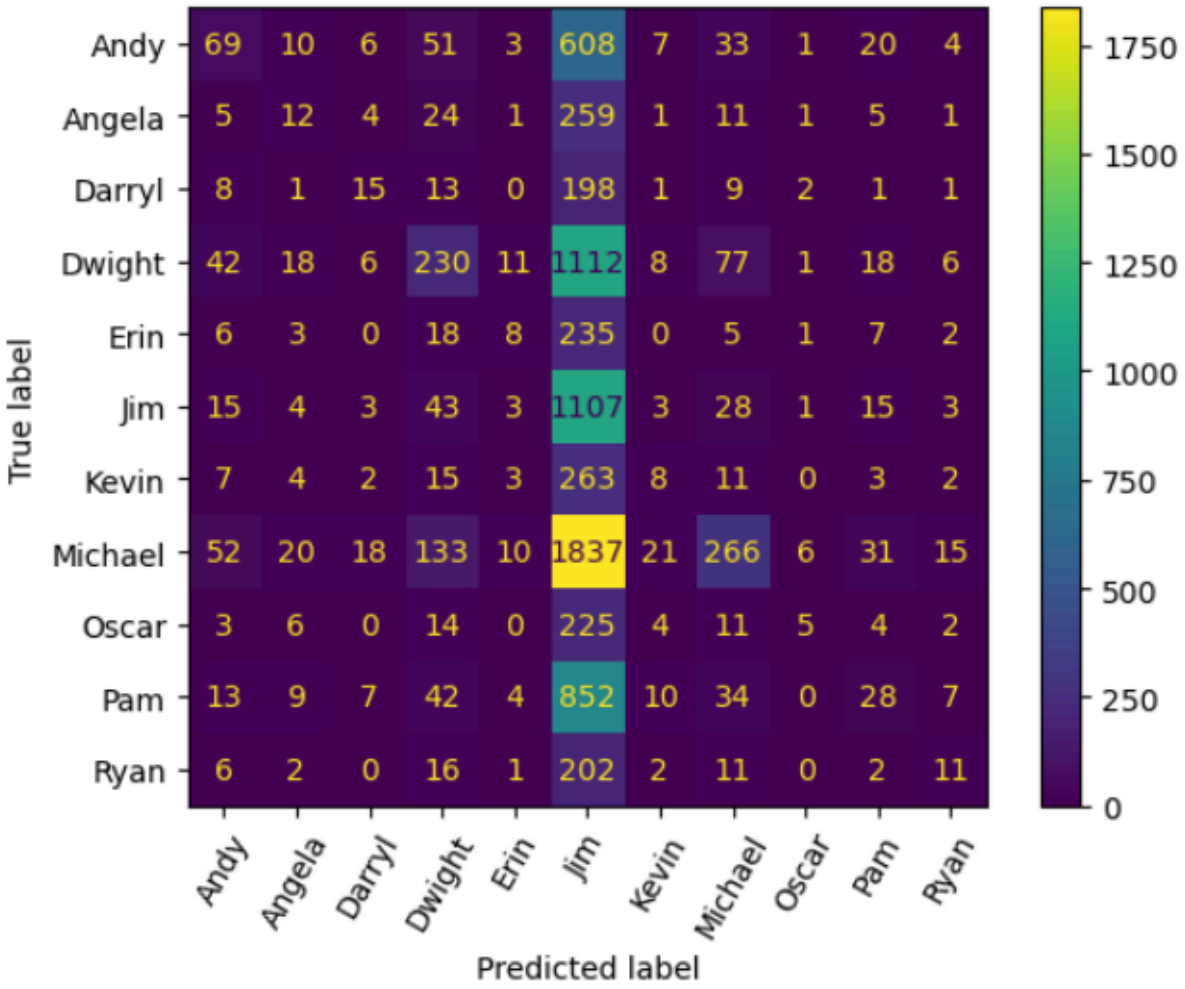
The results of each model, including accuracy scores, confusion matrices, and predictions, were analyzed to assess the performance of speaker classification on The Office dataset.

The Model (Multinomial Naive Bayes) confusion matrix is:



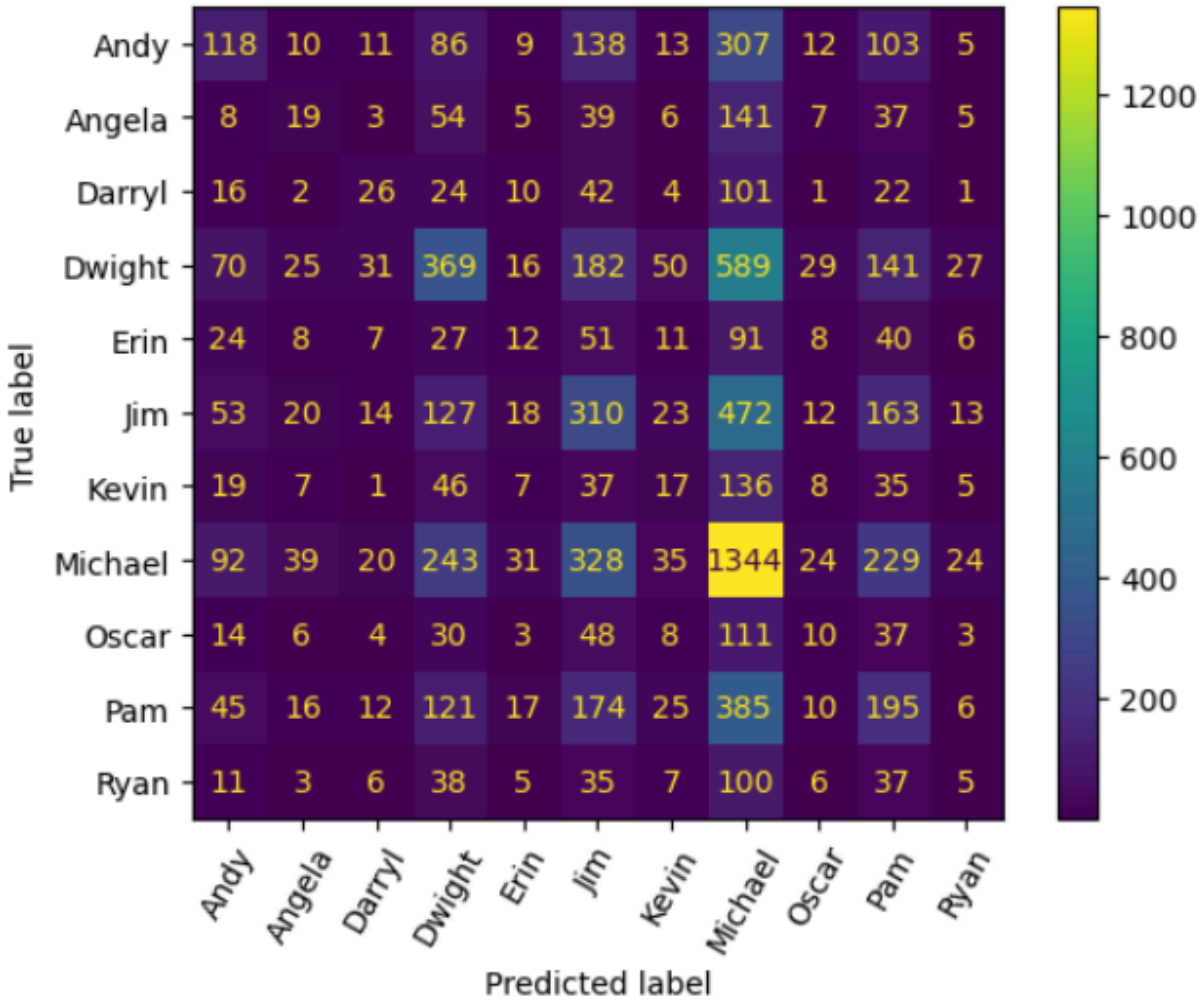
	precision	recall	f1-score	support
Andy	0.37	0.12	0.18	812
Angela	0.27	0.01	0.02	324
Darryl	0.29	0.02	0.03	249
Dwight	0.40	0.24	0.30	1529
Erin	0.07	0.00	0.01	285
Jim	0.25	0.14	0.18	1225
Kevin	0.12	0.01	0.01	318
Michael	0.32	0.85	0.47	2409
Oscar	0.08	0.00	0.01	274
Pam	0.23	0.07	0.11	1006
Ryan	0.14	0.00	0.01	253
accuracy			0.32	8684
macro avg	0.23	0.13	0.12	8684
weighted avg	0.29	0.32	0.24	8684

The Model 2 (SVM with Linear SVC) confusion matrix is:



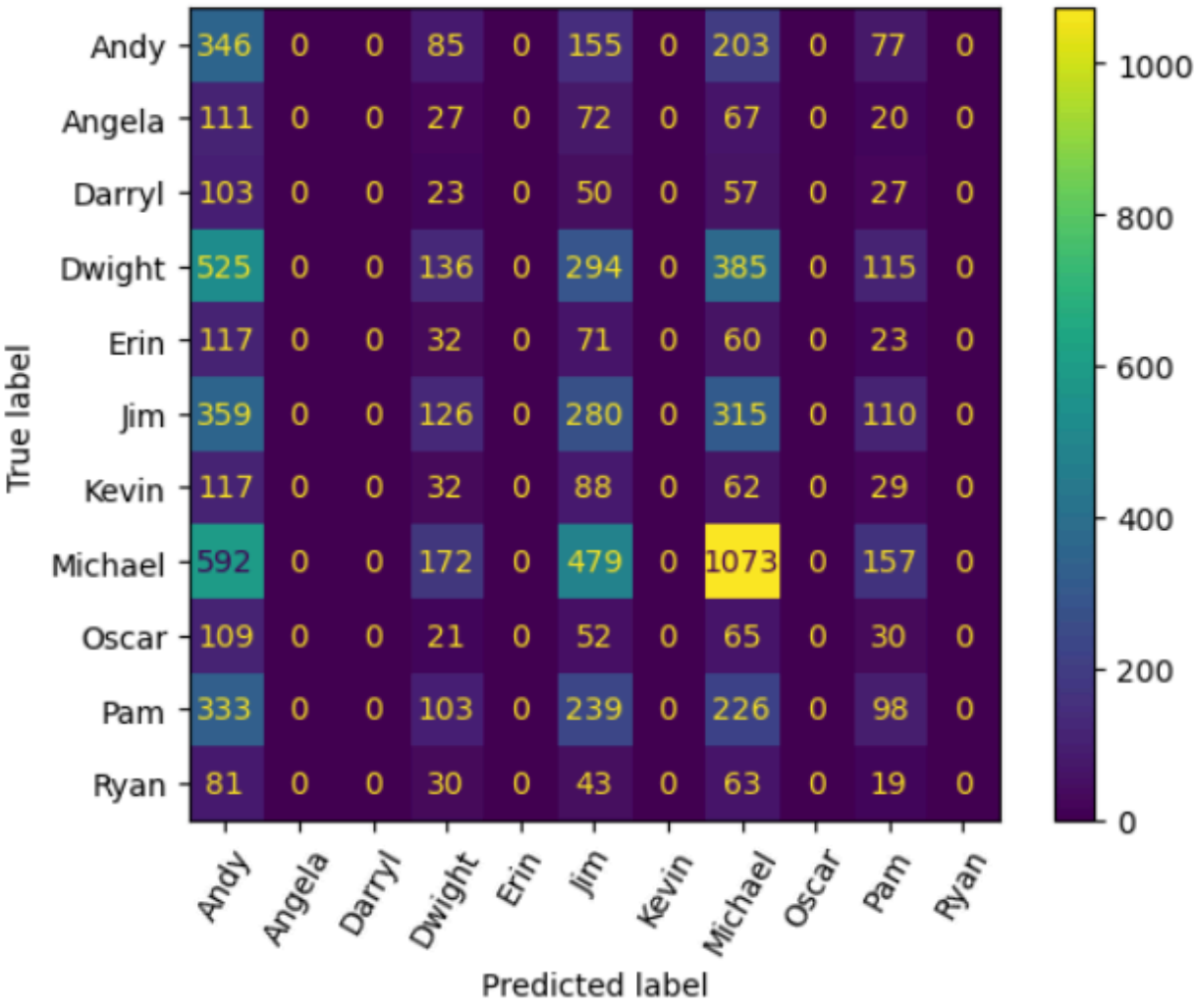
	precision	recall	f1-score	support
Andy	0.31	0.08	0.13	812
Angela	0.13	0.04	0.06	324
Darryl	0.25	0.06	0.10	249
Dwight	0.38	0.15	0.22	1529
Erin	0.18	0.03	0.05	285
Jim	0.16	0.90	0.27	1225
Kevin	0.12	0.03	0.04	318
Michael	0.54	0.11	0.18	2409
Oscar	0.28	0.02	0.03	274
Pam	0.21	0.03	0.05	1006
Ryan	0.20	0.04	0.07	253
accuracy			0.20	8684
macro avg	0.25	0.14	0.11	8684
weighted avg	0.33	0.20	0.16	8684

The confusion matrix (Random Forest) is:



	precision	recall	f1-score	support
Andy	0.25	0.15	0.18	812
Angela	0.12	0.06	0.08	324
Darryl	0.19	0.10	0.14	249
Dwight	0.32	0.24	0.27	1529
Erin	0.09	0.04	0.06	285
Jim	0.22	0.25	0.24	1225
Kevin	0.09	0.05	0.07	318
Michael	0.36	0.56	0.43	2409
Oscar	0.08	0.04	0.05	274
Pam	0.19	0.19	0.19	1006
Ryan	0.05	0.02	0.03	253
accuracy			0.28	8684
macro avg	0.18	0.16	0.16	8684
weighted avg	0.25	0.28	0.26	8684

confusion matrix for NN



	precision	recall	f1-score	support
Andy	0.12	0.40	0.19	866
Angela	0.00	0.00	0.00	297
Darryl	0.00	0.00	0.00	260
Dwight	0.17	0.09	0.12	1455
Erin	0.00	0.00	0.00	303
Jim	0.15	0.24	0.19	1190
Kevin	0.00	0.00	0.00	328
Michael	0.42	0.43	0.43	2473
Oscar	0.00	0.00	0.00	277
Pam	0.14	0.10	0.12	999
Ryan	0.00	0.00	0.00	236
accuracy			0.22	8684
macro avg	0.09	0.11	0.09	8684
weighted avg	0.20	0.22	0.20	8684

LSTM using Word2Vec

```
%pip install keras
```

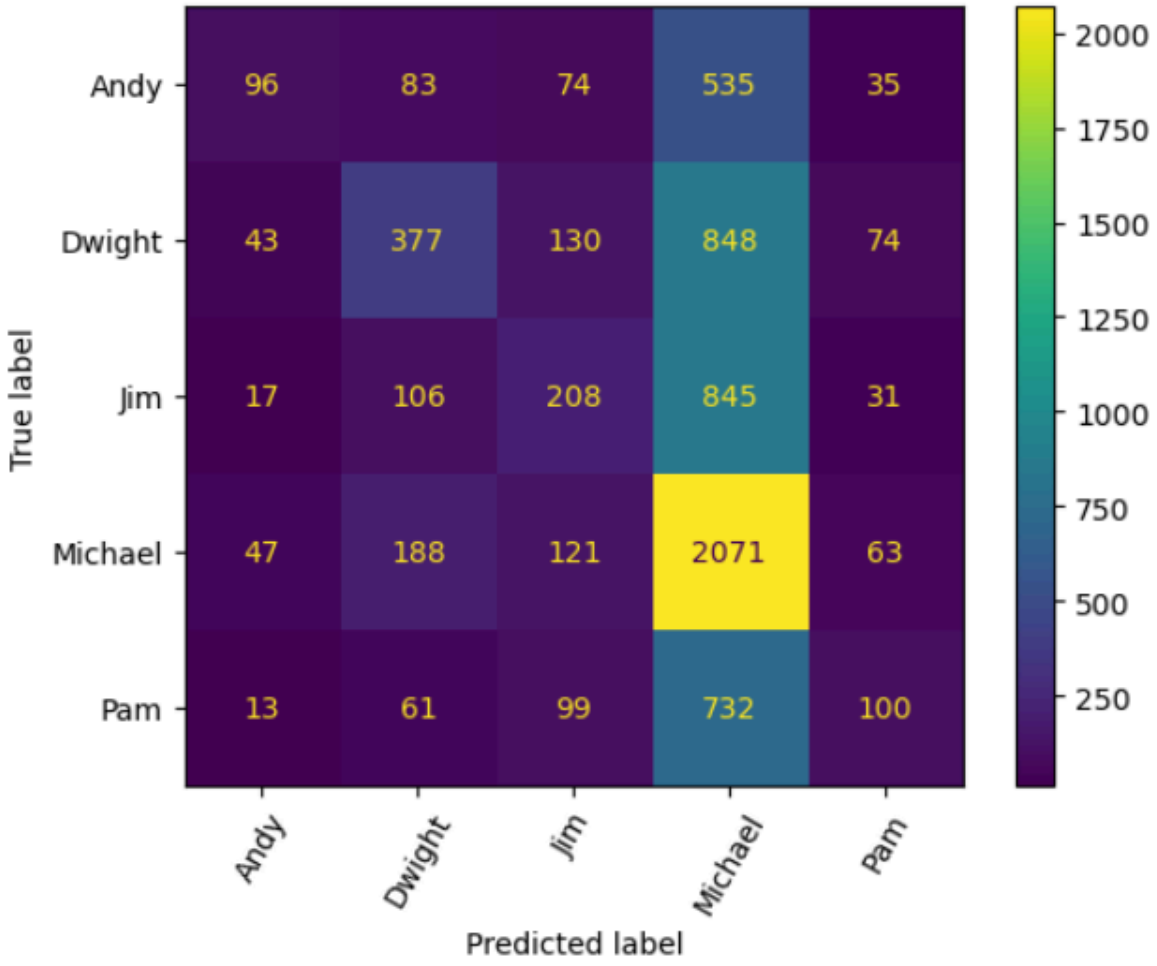
```
# Evaluate the model
loss, accuracy = model.evaluate(X_test, y_test)
print(f'Test Accuracy: {accuracy}')
```

```
181/181 ━━━━━━━━━━━━━━━━━━━━ 4s 23ms/step - accuracy: 0.0358 - loss: -38890.9570
Test Accuracy: 0.034548282623291016
```

Naive Bayes with fewer classes

```
# Load the CSV data into a pandas DataFrame
df = pd.read_csv('C:/Users/Patrick/Syracuse_courses/IST_736/Final_Project/cleaned_lines.csv')

# Calculate the frequency of each speaker
speaker_counts = df['speaker'].value_counts()

# Filter the DataFrame to keep only rows where the speaker appears at least 100 times
df = df[df['speaker'].isin(speaker_counts[speaker_counts >= 2000].index)].reset_index(drop=True)
```

The Model (Multinomial Naive Bayes) confusion matrix is:



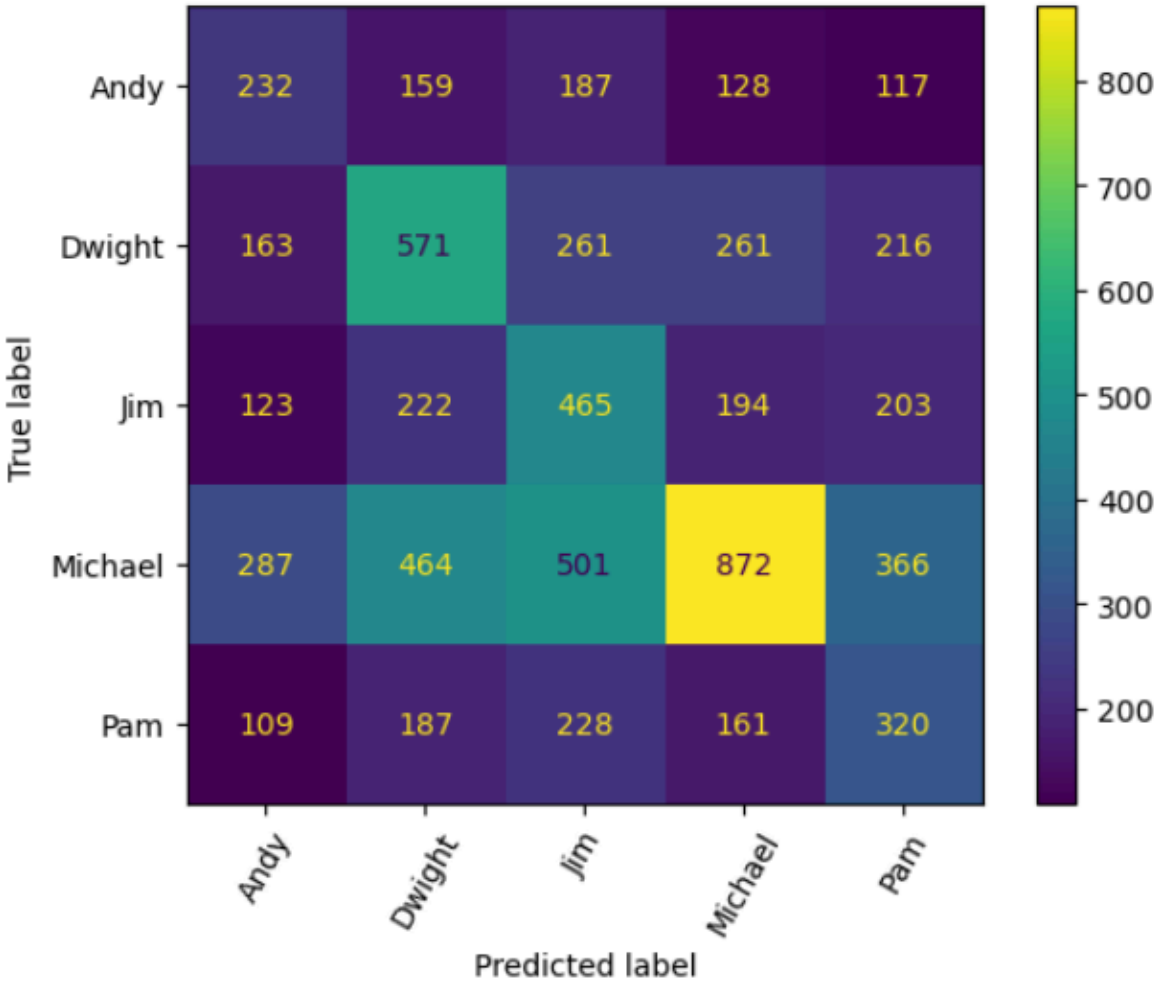
	precision	recall	f1-score	support
Andy	0.44	0.12	0.18	823
Dwight	0.46	0.26	0.33	1472
Jim	0.33	0.17	0.23	1207
Michael	0.41	0.83	0.55	2490
Pam	0.33	0.10	0.15	1005
accuracy			0.41	6997
macro avg	0.40	0.30	0.29	6997
weighted avg	0.40	0.41	0.35	6997

Class weighting to account for inbalanced dataset

```python
from sklearn.utils import class_weight
from sklearn.svm import SVC

# Calculate class weights
classes = np.unique(TrainLabels)
class_weights = class_weight.compute_class_weight('balanced', classes=classes, y=TrainLabels)

# Print the computed class weights
print("Class Weights:", class_weights)

# Create a dictionary mapping class labels to their weights
class_weights_dict = dict(zip(classes, class_weights))

# Map class labels to their corresponding weights using NumPy indexing
sample_weights = np.array([class_weights_dict[label] for label in TrainLabels])

# Create an SVM classifier
svm_model = SVC(kernel='linear')

# Fit the SVM model with sample weights
svm_trained = svm_model.fit(TrainSet, TrainLabels, sample_weight=sample_weights)
```

```
Class Weights: [1.69786791 0.93821839 1.12160769 0.57391457 1.40611542]
```

The Model (Linear SVC) confusion matrix is:



	precision	recall	f1-score	support
Andy	0.25	0.28	0.27	823
Dwight	0.36	0.39	0.37	1472
Jim	0.28	0.39	0.33	1207
Michael	0.54	0.35	0.42	2490
Pam	0.26	0.32	0.29	1005
accuracy			0.35	6997
macro avg	0.34	0.34	0.34	6997
weighted avg	0.38	0.35	0.36	6997

CONCLUSIONS

The text mining analysis of The Office dataset has provided valuable insights into various aspects of the show's dialogue and character interactions. One key finding is the distinct dialogue styles of each character, which contribute to their unique personalities and roles within the show.

However, it proved very difficult to develop robust classification methods for identifying the speaker by dialogue alone. It may be the case that more holistic methods that take into consideration other pieces of data may be necessary. For example, methods that utilize other information such as the season and episode number may yield better results in the task of speaker identification.

Overall, this analysis highlights the richness of The Office's dialogue and its contribution to the show's success. It showcases how text mining techniques can be used to extract valuable information from large datasets, offering new perspectives and enhancing our appreciation of beloved TV shows like The Office.