Essential Techniques for Categorizing News Headlines and Traditional Semantic Analysis

Team Name: Group 404 Not Found
Team Members: CHEN Chi-Wei(22102091d) LAM Yuen Hei(22074854d) ZHENG Shusen(22103691d)

Introduction

This project aims to adequately classify the headlines in the dataset "A Million Headlines." To achieve this goal, we used extra datasets, labeled news header lines, from Kaggle to train our various models, including LSTM (Long short-term memory) and Logistic Regression, and thus further make predictions to the A Million Headlines dataset. After which, based on the labeled header lines, we further conducted various research such as semantic track and inter-category relationship analysis and found insightful results.

About the given dataset

To commence with, we analyzed the dataset to develop a basic understanding of the date patterns. The dataset contains data on news headlines published over 15 years. Each record in the dataset includes the date of publishment and the headline content, for example: "20030219, act fire witnesses must be aware of defamation" and "20030221, a class action may cost Esso 50m".

It is noticeable that multiple abbreviates exist in the headlines, for instance, "fed opp" and "tas man," which increases the uncertainty for the clustering job as the meaning of the words here is not clearly defined. With an eye on the abovementioned difficulties, we introduced an extra data set to assist in the model training. By doing so, the architecture of this project will be the combination of supervised and unsupervised learning, in which the former is mainly referred to as the training stage, and the latter refers to the prediction stage.

Extra dataset used in this project

We found a suitable date set containing labeled news header lines from Mr. Rishabh Misra at Kaggle (https://www.kaggle.com/datasets/rmisra/news-category-dataset).

This dataset contains around 210k news headlines from 2012 to 2022 from HuffPost. Each record in the dataset consists of its category, headline, authors, link, short description, and publication date. To further match the original dataset, we decided only to use each record's category features and headline. For example: "CRIME, There Were 2 Mass Shootings in Texas Last Week..." and "POLITICS, Booyah: Obama Photographer Hilariously Trolls Trump's…"

With the assistance of introducing this labeled dataset, we could train the model to facilitate the classification of the original one by grouping up the news header lines and further understanding the characteristics and preferences of news underlying the dataset.

Model introduction and data preprocessing

While resolving the problem, several models were considered; specifically, logistic regression and long short-term memory were used. Model-tuning techniques and optimization strategies are also introduced to carry out better models, but only sometimes do the predictions improve. Our strategy included ensemble methods, grid search, enabling multiple predictions, etc.

Due to the tremendous demand for calculation power for the training and predicting of the full-size dataset, we randomly extracted 5% of data from the full-size dataset for the training and predicting using the simple code below. In addition, the random state-valued one is used for better reproducibility.

```
# Calculate 5% of the total number of rows
sample_size = int(len(data) * 0.05)
random_sample = data.sample(n=sample_size, random_state=1)  # random_state for reproducibility
```

Model One: Logistic Regression

A. Import the data:

(the step won't be shown for the other models as they shared the same way for importing the data)

Label Encoding is used to convert the categorical data further, that is, the news category here, into numerical form, making it suitable for the Logistic Regression algorithm.

```
train_data_path = "train.csv"
df = pd.read_csv(train_data_path, names=["index", "category", "headline"], skiprows=1)
df["category"] = df["category"].str.strip()
df["headline"] = df["headline"].str.strip()

label_encoder = LabelEncoder()
df["category_encoded"] = label_encoder.fit_transform(df["category"])

train_texts, val_texts, train_labels, val_labels = train_test_split(
    df["headline"], df["category_encoded"], test_size=0.2, random_state=42
)
```

B. Term Frequency-Inverse Document Frequency Vectorizer

TfidfVectorizer is used to convert the textual part of the data into numerical representations, that is, the header lines here, because of the following benefits: 1. reduces the impact of common words such as "the" and "is." 2. captures the importance of words, which helps in identifying key terms for news category classification. 3. scales well for large text corpora. Expressly, the parameter is set as shown below:

```
vectorizer = TfidfVectorizer(max_features=5000, ngram_range=(1, 2), stop_words="english")
X_train = vectorizer.fit_transform(train_texts)
X_val = vectorizer.transform(val_texts)
```

C. Chunks and Batches

Chunks are used to split news data into smaller chunks to ensure memory efficiency and enable scalability, as only a portion of the data is processed at a time. On the other hand, Batch processing improves speed by handling multiple samples simultaneously, reducing the overhead of repeated function calls. Overall, chunks and batches adopted by this model enable efficient parallel processing and can speed up model training or prediction workflows.

```
def predict_batch(batch):
    headlines = batch.iloc[:, 1]
    X_batch = vectorizer.transform(headlines)
    predictions = model.predict(X_batch)
    return label_encoder.inverse_transform(predictions)

chunk_size = 10000
with pd.read_csv(input_file, chunksize=chunk_size, header=None) as reader:
    for i, chunk in enumerate(reader):
        print(f"Processing chunk {i + 1}...")
        chunk_predictions = predict_batch(chunk)
        chunk["predicted_category"] = chunk_predictions
        chunk.to_csv(output_file, mode="a", index=False, header=False)
```

D. Train the model

```
model = LogisticRegression(max_iter=1000, random_state=42)
model.fit(X_train, train_labels)
```

This naïve model achieved 0.58 overall accuracy on the testing dataset. Specifically, it performed the best in classifying news related to divorce (0.83) and weddings (0.79). However, these two topics don't hold the most significant volume of the training data. Therefore, it is believed that the biased distribution among the training data set won't primarily influence the prediction, as it has not been proven that the more training data on a topic, the better prediction results are conducted. More details are shown in Appendix 1.

Improvement on Model One

A. Grid Search

```
# Define parameter grid for Logistic Regression
param_grid = {
    'C': [10.0 ** i for i in range(-5, 6)],
    'penalty': ['l1', 'l2'],
    'class_weight': ['balanced']
}
model = LogisticRegression(solver='liblinear', max_iter=1000, random_state=42)
grid_search = GridSearchCV(estimator=model, param_grid=param_grid, scoring='accuracy', cv=5)
grid_search.fit(X_train, train_labels)
```

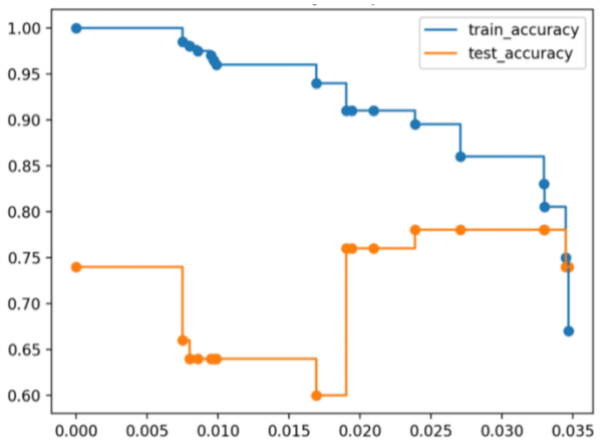
Grid search is used for testing and finding the best hyperparameters and the best combination of the hyperparameters, including C, penalty, and class weight. However, this method takes more than 1 hour to complete the grid search due to its nature and the enormous volume of the training dataset. So, instead of throwing the entire data set into the grid search function, a more realistic way is randomly selecting a subset of data for it. The average of this improved model reached 0.63 accuracy, slightly improved from the default hyperparameters.

```
# Use a subset of training data for GridSearchCV
subset_size = 10000  # Define the number of samples to use
subset_texts, _, subset_labels, _ = train_test_split(
    train_texts, train_labels, train_size=subset_size, stratify=train_labels, random_state=42
)

# Transform the subset of data
X_subset = vectorizer.transform(subset_texts)
```

B. Bootstrap Aggregation (Bagging)

Plotting the perdition accuracy regarding the hyperparameter C shows a rational range of possible values of C, which all perform well. So, instead of only selecting a single best C, grouping multiple best Cs and using a majority vote for the final prediction is more reasonable. Specifically, this method selects and aggregates the models with an accuracy higher than 0.4. The related code is shown below:




```
for C in candidate_C_values:
    # Train a model with the specific C value
    model = LogisticRegression(C=C, penalty='l2', class_weight='balanced', solver='liblinear', max_iter=1000, random_state=42)
    model.fit(X_subset, subset_labels)
    # Predict on the training set
    predictions = model.predict(X_val_subset)
final_train_predictions = []
for i in range(train_predictions.shape[1]):  # Iterate over all training samples
    sample_predictions = train_predictions[:, i]
    most_common = Counter(sample_predictions).most_common(1)[0][0]
    final_train_predictions.append(most_common)
```

This method generates prediction with an accuracy of 0.40, which is way worse than the naïve method. It is analyzed that the reason behind this is that the number of candidate C values is limited and often falls into the range between 2 and 4, which is inappropriate for using a majority vote.

Therefore, with the inspiration from the actual news application, single news can be labeled with multiple categories; thus, instead of using a majority vote to generate the best prediction, it may be more proper to keep all generated categories, and as long as one of which matches with the correct label, it is considered as a successful prediction.

```
# Store predictions per headline
for idx, pred in enumerate(predictions):
    all_predictions[chunk_idx * chunk_size + idx].append(pred)
```

This method significantly increases the accuracy from 0.4 to 0.615. However, for most headlines, all labels generated by different models are the same, which is an immature result due to their natures.

```
0,1,predicted_categories
20030219,aba decides against community broadcasting licence,"GOOD NEWS, GOOD NEWS, GOOD NEWS"
20181111,sunday november 11 full program,"POLITICS, POLITICS, RELIGION"
20040420,tas man sentenced in us court over baby battery,"CRIME, CRIME, CRIME"
20040813,mixed response to water plan,"POLITICS, POLITICS, POLITICS"
20060827,party to decide future of clp senator,"POLITICS, POLITICS, POLITICS"
```

C. [illegible]

As modified in the final section of the former part, it is considered a semi-success by applying muti-category prediction. However, due to its high duplication rate, this part will further discuss the improvement methodologies based on its naïve idea. The method gives the probability of each class for a headline and further selects the top two categories with the highest probabilities. Thus, a prediction is considered valid if at least one of the two labels matches the ground truth.

```
model = LogisticRegression(max_iter=1000, random_state=42)
model.fit(X_train, train_labels)
val_probs = model.predict_proba(X_val)
val_top2 = [label_encoder.inverse_transform(probs.argsort()[-2:][::-1]) for probs in val_probs]
```

```
probs = model.predict_proba(X_batch)
top2_categories = [
    label_encoder.inverse_transform(probs[i].argsort()[-2:][::-1]) for i in range(len(probs))
]
```

This method produces the highest accuracy so far, which is 0.74. By continuing to increase the number of final predicted labels, the accuracy will slowly grow. For example, if we increase it to three labels, the final prediction reaches 0.81. However, from a practical perspective, the meaning of having more than three labels for single news could be more visible. So, we decided to stop at this stage.

Other Approaches with Different Models

Besides the use of a logistic regression model to predict category labels for the headlines. We have applied different models to discover potential improvements. Notably, the Support Vector Machine (SVM) and Long Short-term Memory model are used in our discoveries.

Support Vector Machine (SVM)

SVM is a robust supervised machine learning algorithm commonly used for classification tasks. It works by finding the optimal hyperplane that best separates different data classes in a high-dimensional feature space. This hyperplane is chosen to maximize the margin, which is the distance between the hyperplane and the nearest data points from each class, known as support vectors. This model is chosen due to its effectiveness in high-dimensional spaces. As mentioned earlier, the TF-IDF transforms words into vectors, often resulting in a large, sparse matrix. Therefore, with SVM's inherent strengths in high-dimensional sparse data analysis and linear kernel efficiency, we could potentially enhance the accuracy of news headline categorization.

After the training with the naïve SVM model, the result of classifications was not appetible. Hence, a grid search was applied to improve the adjustment of the hyperparameter with the top two labels for each news headline. The result improved substantially to 0.66, but still lower than the earlier logistic regression model. After that, we applied several fine-tuning methods for data regularization and model generalization. The improvement is not significant or even lowers its accuracy; it implies that the accuracy of this model has reached a bottleneck. As a result, we decided to look for other models that may better handle the complexity of the data. We noticed that the Long Short-Term Memories model has been frequently mentioned in the Kaggle discussion forum, and it is widely considered adequate for this scenario. Given that, we switched our model to the LSTM model to improve the classification accuracy.

Long Short-term Memory (LSTM)

LSTM is a specialized recurrent neural network (RNN) designed to effectively capture and learn from long-range dependencies in sequential data. This architecture addresses the vanishing gradient problem that traditional RNNs face by utilizing a memory cell and various gating mechanisms (input, forget, and output gates) to control the flow of information. In the news headline categories prediction task, LSTM offers significant advantages by capturing the sequential nature of language, which is crucial for understanding and contextualizing the flow of a headline. Unlike models that treat each word independently, LSTMs can learn from the context provided by the order of words, thereby understanding word importance and sentiment over a sequence. Furthermore, its ability to retain information over longer spans makes it suitable for capturing the essential relationships between words necessary to categorize headlines accurately.

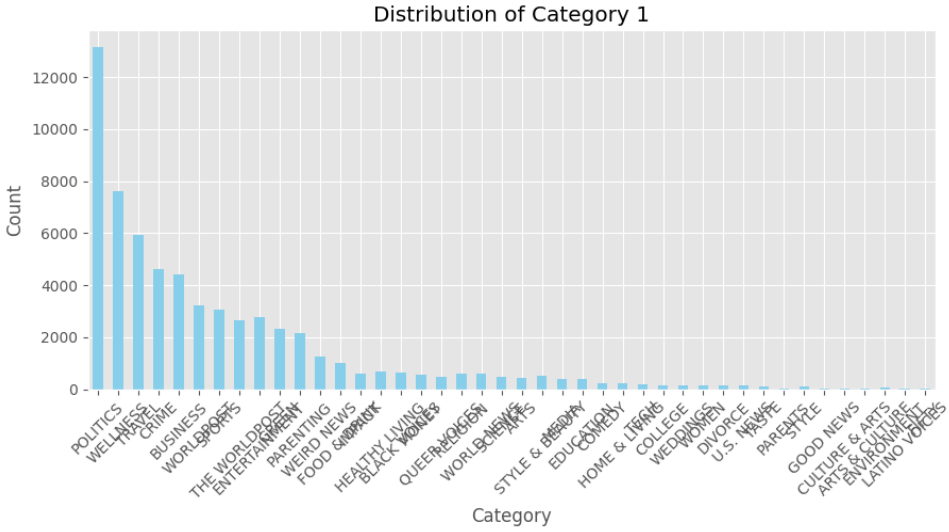
```
# Build the BiLSTM model
model = Sequential([
    (import) Bidirectional: Any
    Bidirectional(LSTM(32)),
    Dense(256, activation='relu'),
    Dropout(0.4),
    Dense(y_categorical.shape[1], activation='softmax')
])

# Compile the model
model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])
```

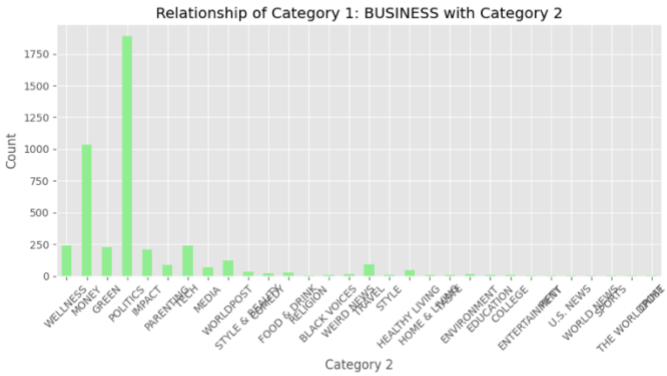
Similarly to the improvement of model one, we applied grid search to fine-tune LSTM's hyperparameters and predict the top two predicted labels instead of one. These adjustments significantly improved classification accuracy.

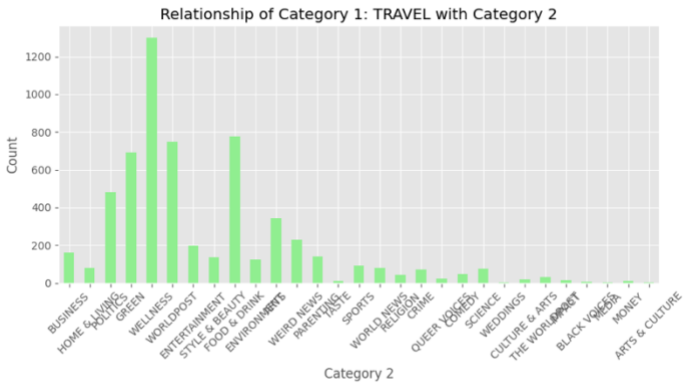
In addition, we further revised our labeling policy by grouping related headline labels to better reflect topic relationships and similarities. The final version of the LSTM model involved minor adjustments to the dropout rate and the maximum word count, further enhancing performance. Ultimately, the LSTM model achieved an accuracy of 0.70 on the test dataset.

The distribution of the news with grouping is shown on the next page. Category 1 represents the label the model considers has the highest similarity to the headline content; Category 2 is the second highest similarity label. Furthermore, the distribution reflects that lifestyle and wellness news will most likely appear in the headlines. In contrast, science and technology news are the least likely to be the headlines. Shown below:



The relationship between the categories is also worthy of analysis, which implies the relationship and closeness of the topics, given that the cosine distance in the words' space determines the category's similarity. For instance, travel and tourism are closely related to the wellness category; business news is closely related to political news (monetary or economic policy). It shows that the content of headlines can overlap with several topics, which conform to our common sense. More details are shown below:



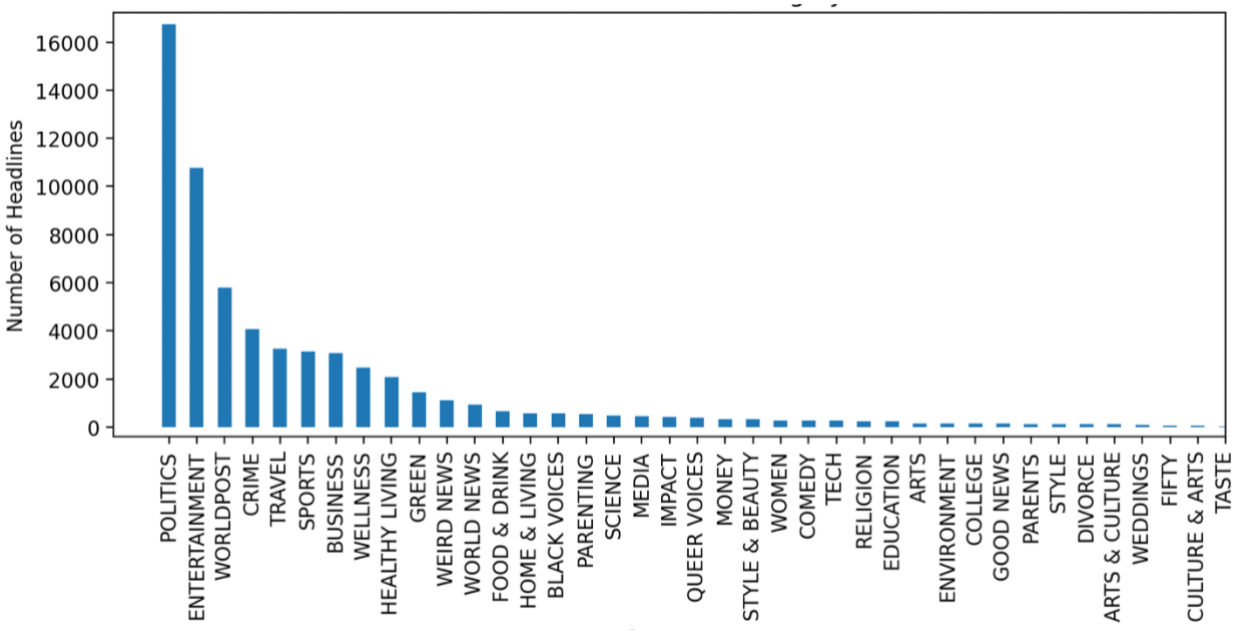


Semantic Analysis

News typically conveys objective information about events happening worldwide and the underlying emotional sentiments of reporters and interviewees. Therefore, conducting further semantic analysis on labeled data is worthwhile in tracing human emotions' evolution over time. Additional libraries like Text Blob and VADER were utilized to achieve more accurate and professional results.

Text Blob:

Before showing the result, it is worth explaining the nature of the dataset for a better understanding. The distribution among different categories is different, so the number of news items in each category varies greatly, as shown below. Therefore, semantic analysis is greatly influenced by this natural factor of the data set.



As shown in Appendix 2, some news categories appear to have abnormal results. For example, for the category "culture and art," due to the limited number of corresponding attributes (47), many years of its polarity value appear as 0.00 or NaN. Therefore, conducting semantic research only on the top 5 categories is more appropriate for achieving meaningful results.

As shown in Appendix 3, the result suggests that **Travel** consistently shows a tiny positive sentiment trend, staying close to 0.0 or slightly above. **Entertainment** fluctuates near neutral

sentiment but generally remains somewhat positive. Conversely, **Politics** hovers around the negative sentiment, while **World Post** also trends negatively, though less steeply. **Crime** has the most negative sentiment score, consistently below -0.3, showing a downward trend over the years.

On the other hand, subjectivity trends vary more significantly than sentiment trends. Over the years, **Entertainment** and **Travel** display a gradual increase in subjectivity, with noticeable spikes post-2015. **Politics** and the **World Post** follow similar patterns but remain less subjective overall. **Crime**, while fluctuating, also shows an upward subjectivity trend. The graph highlights increasing variability in subjectivity in recent years, with multiple sharp peaks and dips.

Vader

As another rule-based sentiment analysis tool, it shows consistent trends in sentiment across news categories. **Travel** and **Entertainment** maintain the most positive sentiment, while **Crime** consistently exhibits highly negative sentiment. These findings highlight a shift in news headlines toward more emotional and opinionated reporting, especially in positive, lifestyle-oriented categories. Details are shown in Appendix 4.

Conclusion

To conclude, the research labeled over five million headlines with text mining. With the labeled headlines, the research team calculated the frequency of labels subscript to the content of headlines; the statistics show that politics and lifestyle content are more likely to appear in the headlines of ABC. Moreover, the research above reflects the relationship and connectivity between different categories of headlines; we can conclude that the content of headlines may be related to multiple topics. Apart from that, we provided an analysis of polarity, subjectivity, and semantics in the content of headlines through logistic regression and text mining; it is found that news headlines in travel and entertainment maintain more positive sentiment and higher subjectivity, while crime remains the most negative and increasingly subjective over time.

Finally, it is believed that there is no myth of machine learning and that the only way to find a better model is to navigate a landscape of trade-offs, as different models may excel under varying conditions. Hyperparameter tuning is a crucial aspect of model optimization, including cross-validation, grid search, and Bayesian optimization.

Appendix

1. prediction reports on the naïve logistic regression model:

Classification Report:

	precision	recall	f1-score	support
ARTS	0.49	0.24	0.32	312
ARTS & CULTURE	0.32	0.11	0.17	288
BLACK VOICES	0.50	0.33	0.40	873
BUSINESS	0.45	0.43	0.44	1119
COLLEGE	0.43	0.27	0.33	214
COMEDY	0.60	0.39	0.47	1066
CRIME	0.53	0.55	0.54	641
CULTURE & ARTS	0.57	0.22	0.32	209
DIVORCE	0.83	0.64	0.72	729
EDUCATION	0.41	0.28	0.33	189
ENTERTAINMENT	0.54	0.73	0.62	3246
ENVIRONMENT	0.54	0.16	0.24	274
FIFTY	0.39	0.10	0.16	260
FOOD & DRINK	0.58	0.69	0.63	1231
GOOD NEWS	0.45	0.12	0.19	269
GREEN	0.41	0.32	0.36	529
HEALTHY LIVING	0.38	0.19	0.26	1303
HOME & LIVING	0.68	0.67	0.68	800
IMPACT	0.44	0.25	0.32	695
LATINO VOICES	0.70	0.21	0.33	217
MEDIA	0.52	0.31	0.39	537
MONEY	0.52	0.32	0.40	356
PARENTING	0.50	0.64	0.56	1730
PARENTS	0.44	0.20	0.27	787
POLITICS	0.66	0.84	0.74	6613
QUEER VOICES	0.76	0.64	0.70	1234
RELIGION	0.57	0.37	0.45	490
SCIENCE	0.58	0.38	0.46	431
SPORTS	0.63	0.59	0.61	972
STYLE	0.53	0.21	0.30	448
STYLE & BEAUTY	0.73	0.80	0.76	1980
TASTE	0.46	0.11	0.18	410
TECH	0.53	0.37	0.44	417
TRAVEL	0.65	0.77	0.71	1959
WEDDINGS	0.79	0.72	0.75	740
WEIRD NEWS	0.36	0.22	0.27	509
WELLNESS	0.53	0.80	0.63	3584
WOMEN	0.38	0.31	0.34	693
WORLD NEWS	0.37	0.11	0.17	413
WORLDPOST	0.55	0.54	0.55	1216
accuracy			0.58	39983
macro avg	0.53	0.40	0.44	39983
weighted avg	0.57	0.58	0.56	39983

2. Text Blob

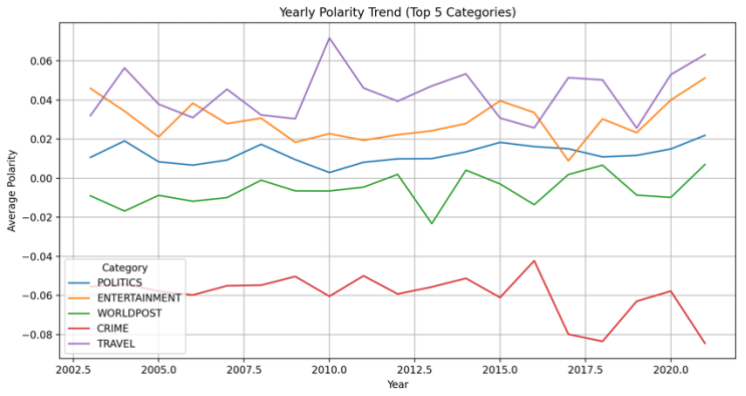
=== Average Polarity Per Year (By Category) ===

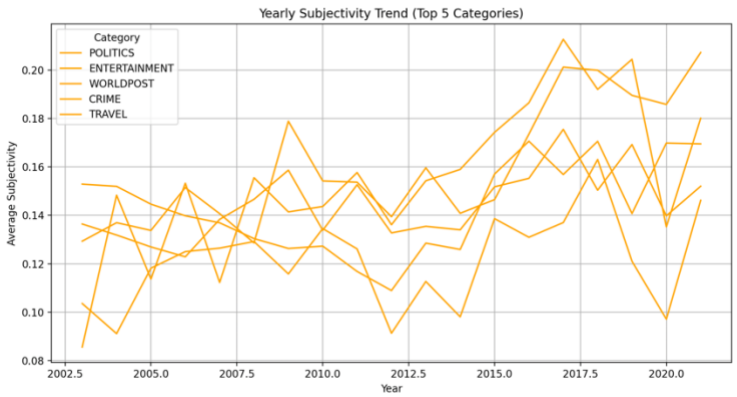
Category	ARTS	ARTS & CULTURE	BLACK VOICES	BUSINESS	COLLEGE	COMEDY	CRIME	CULTURE & ARTS	DIVORCE	EDUCATION	ENTERTAINMENT	ENVIRONMENT	FIFTY	FOOD & DRINK	GOOD NEWS
Year															
2003	0.041667	0.250000	-0.010573	0.018368	0.096212	-0.037302	-0.055490	0.000000	-0.034286	0.120966	0.045927	-0.051852	0.000000	0.089478	-0.031111
2004	-0.065000	0.028571	0.036293	0.020471	0.035000	0.040625	-0.054137	0.000000	0.000000	-0.000303	0.034292	-0.020000	0.000000	0.033667	0.150000
2005	0.086111	0.045455	-0.016919	0.009276	0.000000	-0.030909	-0.057808	0.250000	0.104545	0.021164	0.021141	-0.075926	0.000000	0.064463	0.260000
2006	0.059444	0.000000	-0.047178	0.002975	0.081818	0.050000	-0.059771	0.000000	-0.067321	0.000000	0.038347	0.066667	0.000000	0.070370	-0.040000
2007	0.015385	0.095238	-0.022385	0.006337	-0.009091	-0.144444	-0.055042	0.450000	-0.092857	-0.001389	0.027892	-0.013889	0.025000	0.097333	-0.055556
2008	0.030769	0.015057	0.029915	0.031582	-0.140000	-0.026471	-0.054759	0.000000	0.000000	0.010277	0.030697	0.041667	0.100000	0.094815	-0.059184
2009	0.033333	-0.333333	-0.001948	0.023427	0.050000	-0.062500	-0.050285	0.000000	0.000000	-0.005952	0.018369	0.109740	0.158333	0.136806	-0.033333
2010	0.102381	0.060000	-0.027020	0.030758	0.062500	-0.112092	-0.060424	0.000000	-0.077143	0.007500	0.022748	0.022917	-0.016667	0.103378	-0.107500
2011	0.200000	0.000000	0.022377	0.009455	0.041667	0.093706	-0.049965	0.100000	-0.002381	0.026842	0.019401	0.084524	0.050000	0.101004	0.007692
2012	-0.026190	0.020000	-0.012626	0.029911	0.011111	-0.094156	-0.059223	0.000000	0.000000	0.010714	0.022201	-0.083333	0.016667	0.055611	0.015385
2013	-0.004167	0.062500	-0.031032	0.027071	0.010000	0.002083	-0.055656	0.208333	0.024793	-0.013333	0.024233	0.057407	0.063636	0.069478	0.050000
2014	0.023077	0.106061	0.002835	0.046636	0.017562	-0.039286	-0.051244	NaN	0.042857	0.033434	0.027942	0.085085	0.000000	-0.005590	0.050000
2015	0.054966	0.013636	-0.009091	0.014462	0.091905	0.047697	-0.061044	0.147273	0.000000	-0.013182	0.039547	0.018750	0.000000	0.047179	0.108889
2016	0.053147	0.042857	-0.040817	0.036555	0.006122	0.013815	-0.042233	0.125000	-0.038889	0.074500	0.033590	0.135892	0.002500	0.002558	0.035000
2017	0.020833	0.009091	0.018056	0.032757	0.027778	-0.034762	-0.079830	0.000000	0.042500	0.232000	0.008868	-0.091667	0.011905	0.121648	0.033333
2018	0.043333	-0.080000	-0.025479	0.006027	0.166667	-0.213384	-0.083500	0.000000	-0.037500	0.035556	0.030296	-0.089286	-0.008333	0.021494	0.229286
2019	0.208333	-0.053472	-0.006875	0.016380	0.038333	-0.031194	-0.062967	0.000000	0.112121	-0.078571	0.023279	-0.083333	0.054167	0.021528	0.110000
2020	0.016667	0.000000	-0.071875	0.026886	0.004938	0.010882	-0.057763	NaN	0.000000	0.000000	0.039949	0.045455	0.104167	0.016250	-0.158333
2021	0.050000	0.066667	-0.483333	0.050777	0.000000	0.166667	-0.084478	NaN	0.000000	0.080000	0.051226	0.000000	NaN	0.228571	0.250000

=== Average Subjectivity Per Year (By Category) ===

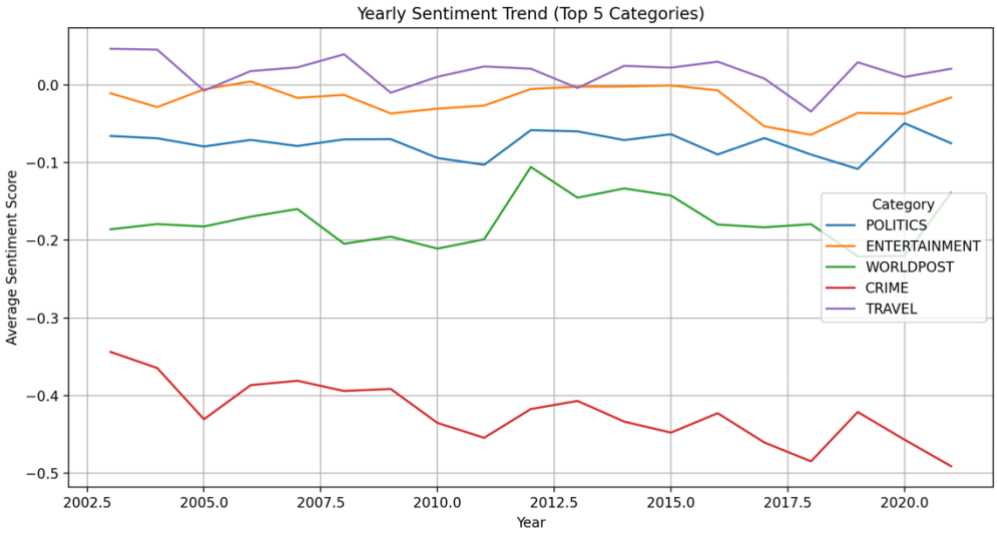
Category	ARTS	ARTS & CULTURE	BLACK VOICES	BUSINESS	COLLEGE	COMEDY	CRIME	CULTURE & ARTS	DIVORCE	EDUCATION	ENTERTAINMENT	ENVIRONMENT	FIFTY	FOOD & DRINK	GOOD NEWS
Year															
2003	0.083333	0.250000	0.194631	0.200056	0.181818	0.282540	0.085564	0.000000	0.322857	0.196693	0.152905	0.096296	0.000000	0.203662	0.057778
2004	0.104000	0.150000	0.162382	0.161656	0.090000	0.206250	0.148327	0.000000	0.000000	0.170379	0.151968	0.130000	0.000000	0.179667	0.162500
2005	0.197222	0.151515	0.182576	0.159786	0.083333	0.220303	0.113665	0.500000	0.131818	0.082011	0.144607	0.214815	0.000000	0.124441	0.320000
2006	0.143333	0.000000	0.173448	0.125792	0.154209	0.116667	0.153283	0.000000	0.247381	0.000000	0.139809	0.100000	0.000000	0.114815	0.070000
2007	0.096154	0.178571	0.142118	0.123215	0.009091	0.376587	0.112264	0.600000	0.273571	0.212037	0.136904	0.172222	0.056667	0.159000	0.348889
2008	0.123077	0.185606	0.149145	0.140099	0.133333	0.105882	0.155548	0.000000	0.000000	0.161430	0.130377	0.187500	0.200000	0.188519	0.236735
2009	0.316667	0.333333	0.136750	0.131748	0.162500	0.163636	0.141407	0.000000	0.000000	0.192857	0.126300	0.225325	0.408333	0.156250	0.044444
2010	0.271429	0.000000	0.234577	0.133796	0.100000	0.205229	0.143648	0.000000	0.121429	0.060313	0.127274	0.308333	0.066667	0.154279	0.138333
2011	0.300000	0.000000	0.163373	0.154869	0.111111	0.179021	0.157693	0.175000	0.290476	0.053421	0.116807	0.161905	0.135714	0.268076	0.053846
2012	0.302381	0.080000	0.121212	0.132389	0.000000	0.302020	0.136110	0.000000	0.000000	0.116071	0.108923	0.088889	0.066667	0.219512	0.180769
2013	0.195833	0.093750	0.177460	0.170252	0.060833	0.097917	0.154288	0.105556	0.098140	0.089259	0.128509	0.104012	0.112121	0.184665	0.016667
2014	0.073077	0.159091	0.116439	0.164463	0.063085	0.221429	0.158977	NaN	0.071429	0.118485	0.125899	0.347159	0.000000	0.206414	0.083333
2015	0.158838	0.025000	0.157576	0.163112	0.174762	0.102632	0.174302	0.240909	0.000000	0.132121	0.157067	0.131250	0.000000	0.215308	0.307778
2016	0.254079	0.257143	0.193571	0.146264	0.138265	0.132353	0.186510	0.275000	0.072222	0.228833	0.170632	0.325135	0.285000	0.171579	0.103333
2017	0.020833	0.013636	0.133056	0.188740	0.148148	0.209127	0.212733	0.000000	0.290833	0.321333	0.156871	0.300000	0.068452	0.257184	0.133333
2018	0.060000	0.260000	0.116038	0.154996	0.166667	0.423990	0.192043	0.000000	0.025000	0.144537	0.170585	0.257143	0.033333	0.228555	0.477857
2019	0.275000	0.351389	0.200000	0.126516	0.083889	0.061765	0.204466	0.000000	0.366667	0.142857	0.140722	0.108333	0.308333	0.195139	0.210000
2020	0.066667	0.000000	0.176875	0.121625	0.037037	0.069697	0.135240	NaN	0.000000	0.009524	0.169863	0.151515	0.458333	0.136250	0.150000
2021	0.100000	0.033333	0.583333	0.174677	0.000000	0.166667	0.180135	NaN	0.000000	0.270000	0.169540	0.000000	NaN	0.228571	0.400000

3. Text Blob 2





4. Vader


Yearly Sentiment Trend (Top 5 Categories)
Average Sentiment Score
0.0
−0.1
−0.2
−0.3
−0.4
−0.5
2002.5
2005.0
2007.5
2010.0
2012.5
2015.0
2017.5
2020.0
Year
Category
POLITICS
ENTERTAINMENT
WORLDPOST
CRIME
TRAVEL