Essential Techniques for Categorizing News Headlines and Traditional Semantic Analysis

By Group 404 Not Found
LAM Yuen Hei 22074854d
CHEN Chi-wei 22102091d
ZHENG Shusen 22103691d

Agenda

Introduction

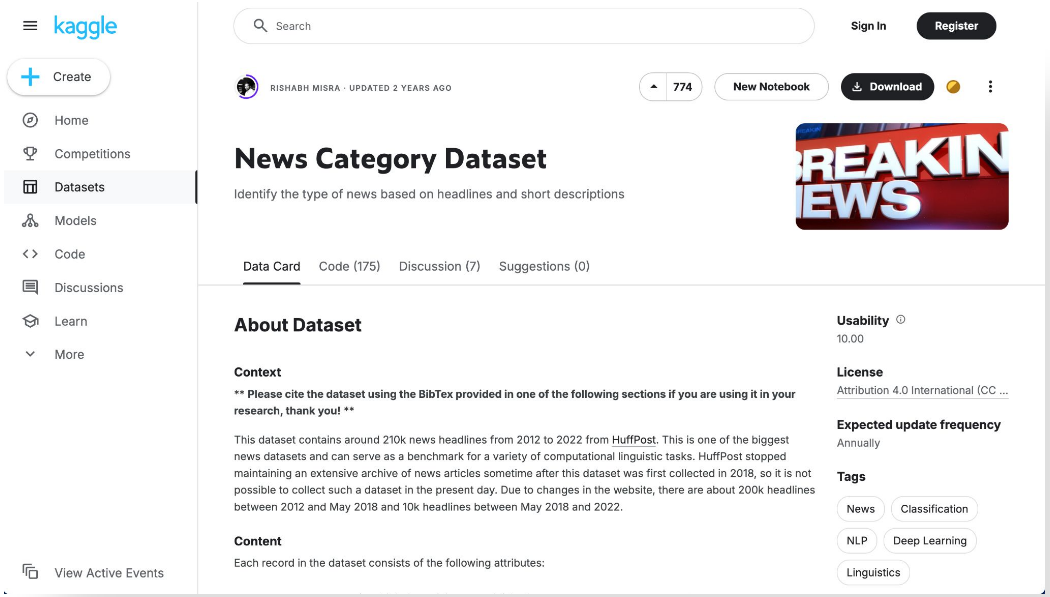
Our group has conducted analysis on both relationship of each category and semantic analysis.

The data used by relationship analysis base on the categories are labeled with Long Short Term Memory model.

The data used by semantic analysis are labeled with Logistic Regression model. Apart from that, we introduced VADER and Textblob for the analysis of semantic underlying in the headlines.

About the dataset

Extra dataset



RISHABH MISRA · UPDATED 2 YEARS AGO

News Category Dataset

Identify the type of news based on headlines and short descriptions

Data Card | Code (175) | Discussion (7) | Suggestions (0)

About Dataset

Context

**** Please cite the dataset using the BibTex provided in one of the following sections if you are using it in your research, thank you! ****

This dataset contains around 210k news headlines from 2012 to 2022 from HuffPost. This is one of the biggest news datasets and can serve as a benchmark for a variety of computational linguistic tasks. HuffPost stopped maintaining an extensive archive of news articles sometime after this dataset was first collected in 2018, so it is not possible to collect such a dataset in the present day. Due to changes in the website, there are about 200k headlines between 2012 and May 2018 and 10k headlines between May 2018 and 2022.

Content

Each record in the dataset consists of the following attributes:

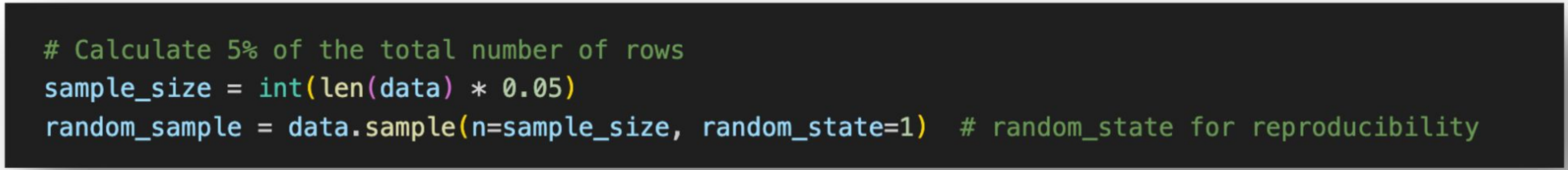
Usability
10.00

License
Attribution 4.0 International (CC ...

Expected update frequency
Annually

Tags
News, Classification, NLP, Deep Learning, Linguistics

Pre-processing



```
# 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: SVM and LSTM

TD-IDF

Term Frequency X Inverse Document Frequency

$$w_{x,y} = tf_{x,y} \times log(\frac{N}{df_x})$$

Text1: Basic Linux Commands for Data Science

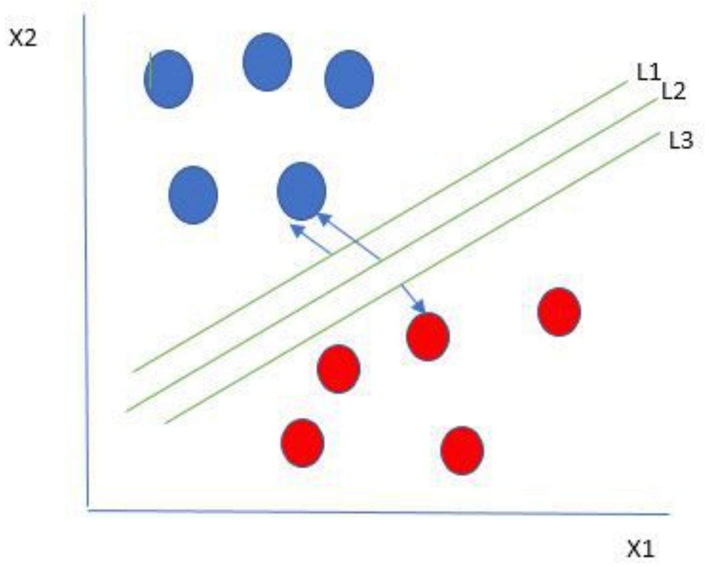
Text2: Essential DVC Commands for Data Science

	basic	commands	data	dvc	essential	for	linux	science
Text 1	0.5	0.35	0.35	0.0	0.0	0.35	0.5	0.35
Text 2	0.0	0.35	0.35	0.5	0.5	0.35	0.0	0.35

Support Vector Machine

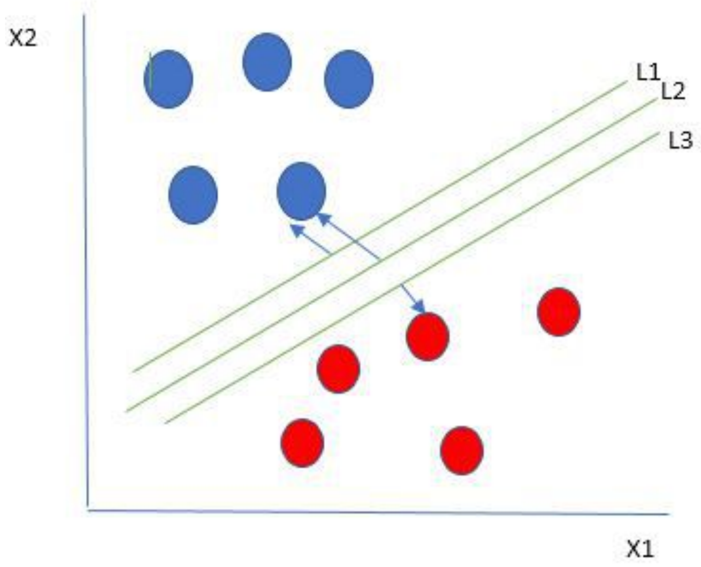
Support Vector Machine (SVM) is a supervised learning algorithm that finds the optimal hyperplane to separate data points into different classes by maximizing the margin between them.

- High Dimensionality Handling
- Sparse Data Efficiency



Support Vector Machine

- Result: Low accuracy (0.66)
- Potential Improvements:
  - Hyperparameter tunning
  - Multiple labels
  - Complex model

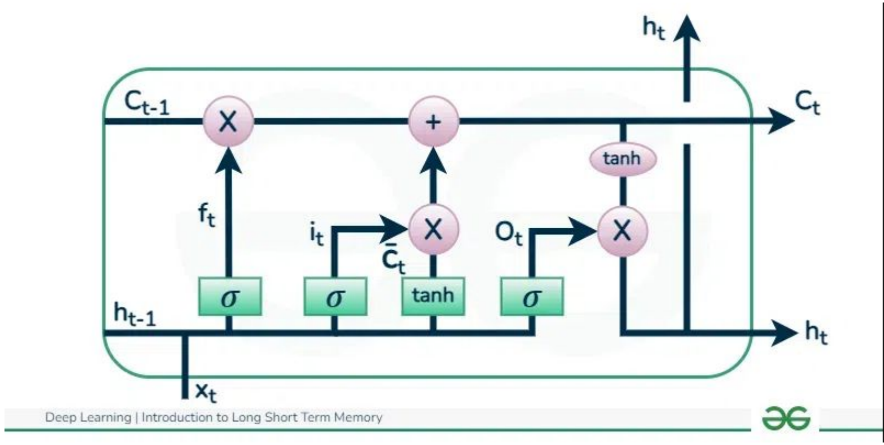


Long short-term memory

Long Short-Term Memory, is a type of recurrent neural network (RNN) architecture specifically designed to overcome the vanishing gradient problem that hinders standard RNNs from learning long-range dependencies in sequential data.

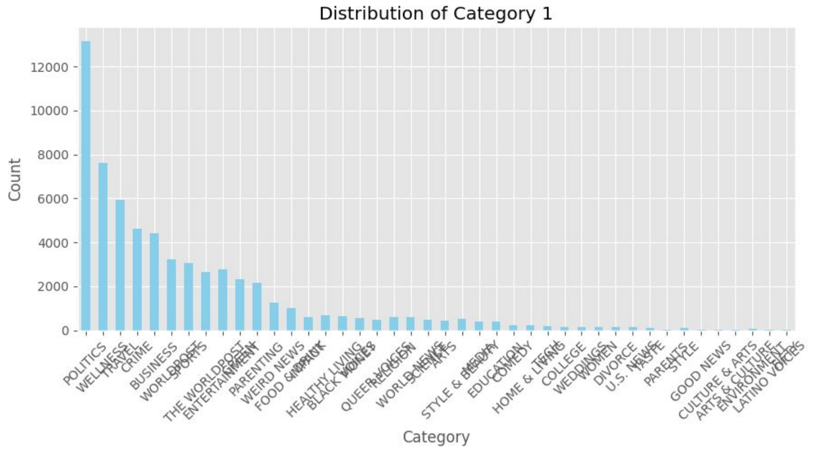
- Capturing Long-Range Dependencies
- Handling Sequential Nature of Text

Accuracy: 0.8 (with two labels)



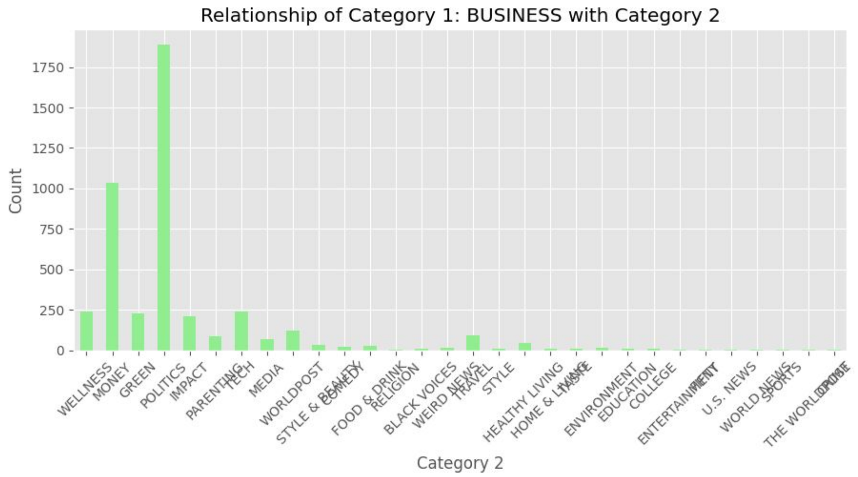
Discoveries based on
Model One: SVM and LSTM

Distribution of Headlines



As what we can observe from the graph, politics news appear the most in the dataset.

Relationship of Categories



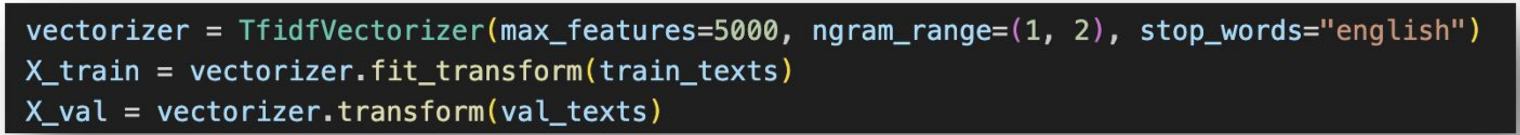
Category 1 is the label with highest similarity with content, Category 2 is the label with second highest similarity

The result shows that news in business topic are regard as similar with politics topics

Model Two:

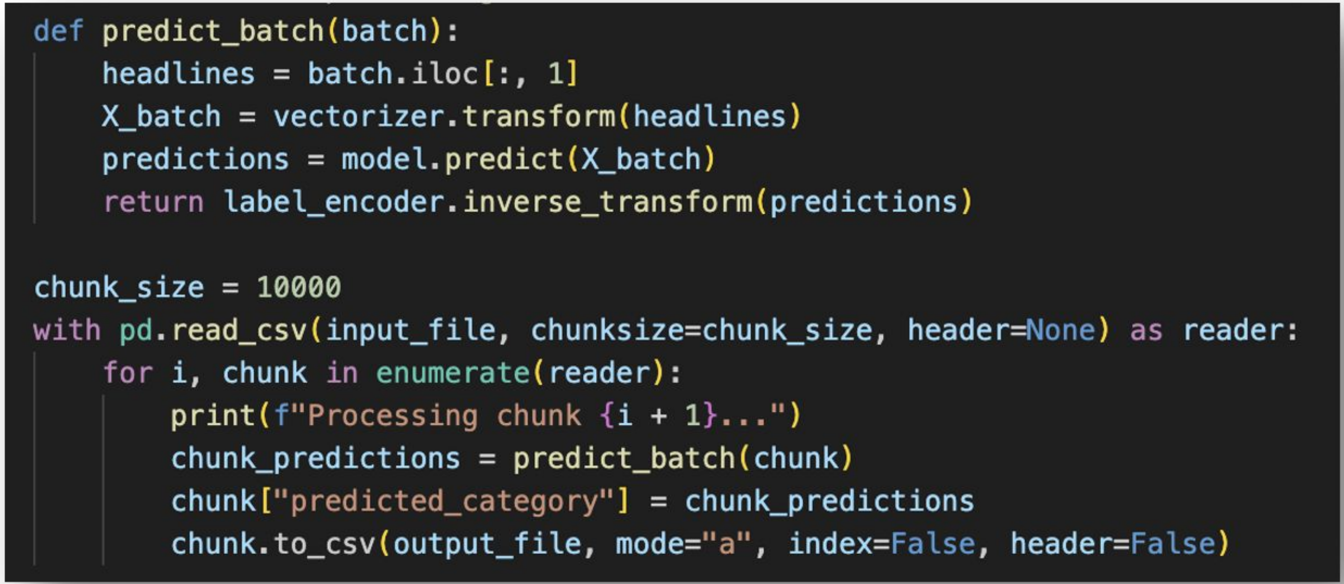
Logistic Regression

TfidfVectorizer



```
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)
```

Chunks and Batches

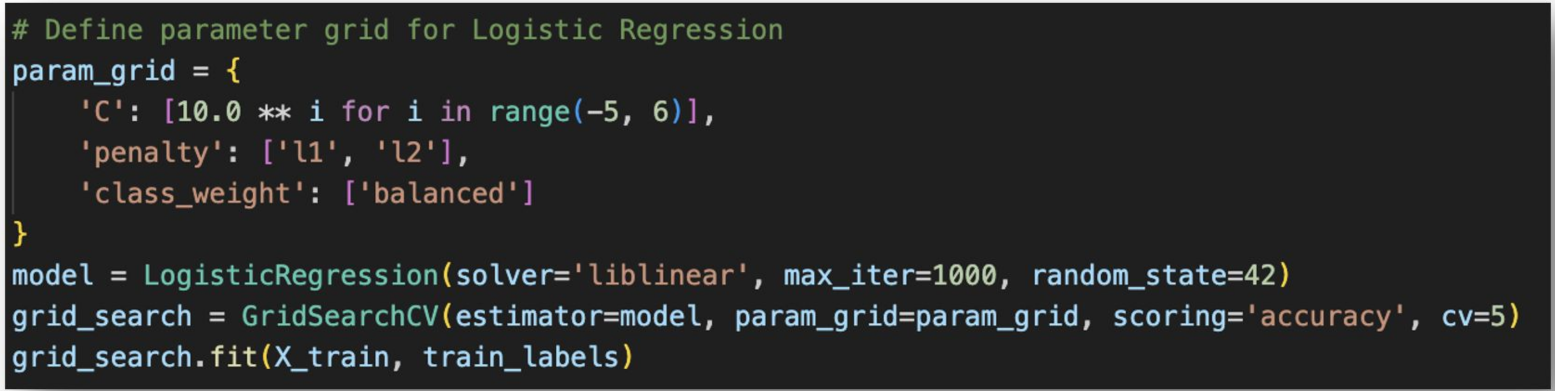

```python
def predict_batch(batch):
    headlines = batch.iloc[:, 1]
    X_batch = vectorizer.transform(headlines)
    predictions = model.predict(X_batch)
    return label_encoder.inverse_transform(predictions)

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)
```

Logsitic Rgressiom:

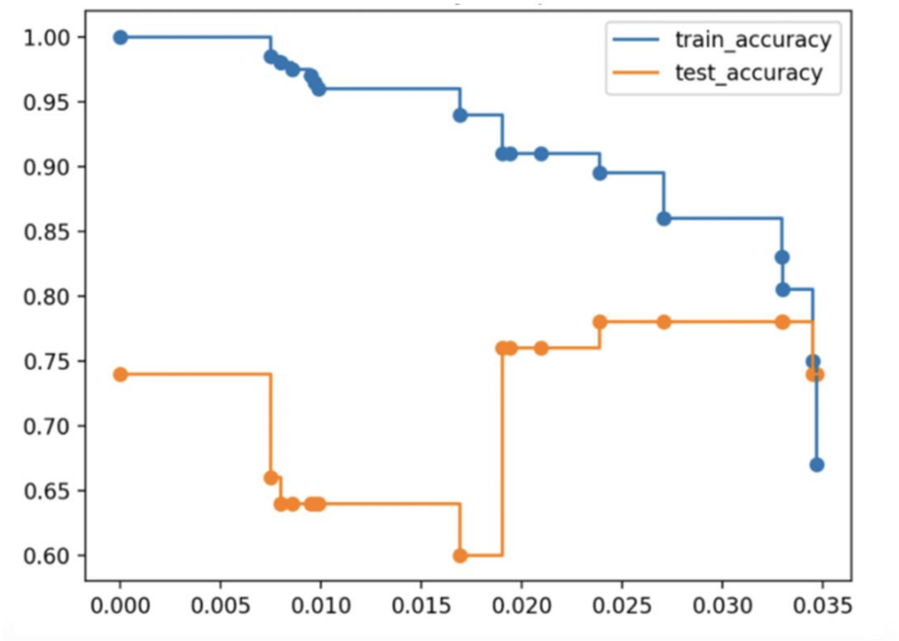
Improvemnt

Grid Search

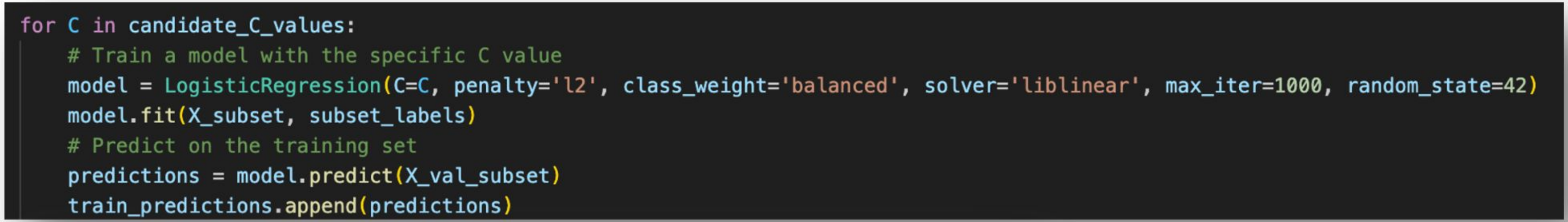


```
# Define parameter grid for Logistic Regression
param_grid = {
    'C': [10.0 ** i for i in range(-5, 6)],
    'penalty': ['l1', 'l2'],
    'class_weight': ['balanced']
}
model = LogisticRegression(solver='liblinear', max_iter=1000, random_state=42)
grid_search = GridSearchCV(estimator=model, param_grid=param_grid,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

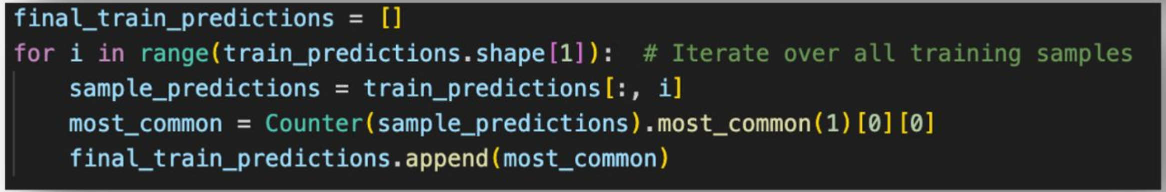
Bootstrap Aggregation (Bagging)



Bootstrap Aggregation (Bagging)

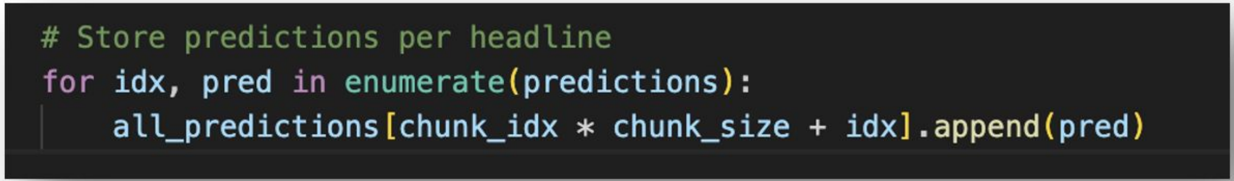


```
for C in candidate_C_values:
    # Train a model with the specific C value
    model = LogisticRegression(C=C, penalty='l2', class_weight='balanced', solver='liblinear', max_iter=1000, random_state=42)
    model.fit(X_subset, subset_labels)
    # Predict on the training set
    predictions = model.predict(X_val_subset)
    train_predictions.append(predictions)
```

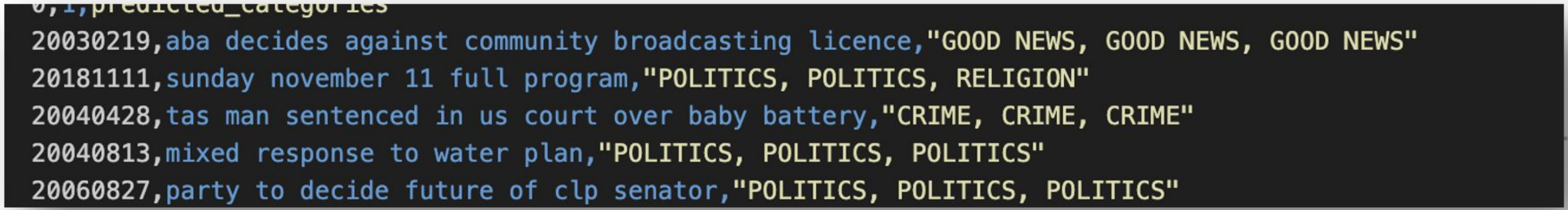


```
final_train_predictions = []
for i in range(train_predictions.shape[1]):  # Iterate over all training samples
    sample_predictions = train_predictions[:, i]
    most_common = Counter(sample_predictions).most_common(1)[0][0]
    final_train_predictions.append(most_common)
```

Labled with multiples

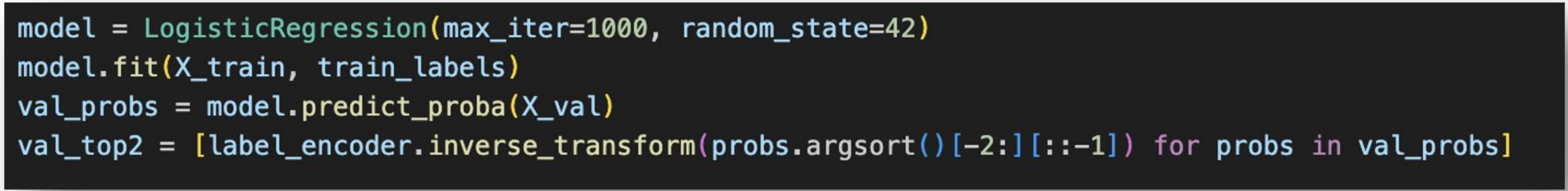


```
# Store predictions per headline
for idx, pred in enumerate(predictions):
    all_predictions[chunk_idx * chunk_size + idx].append(pred)
```

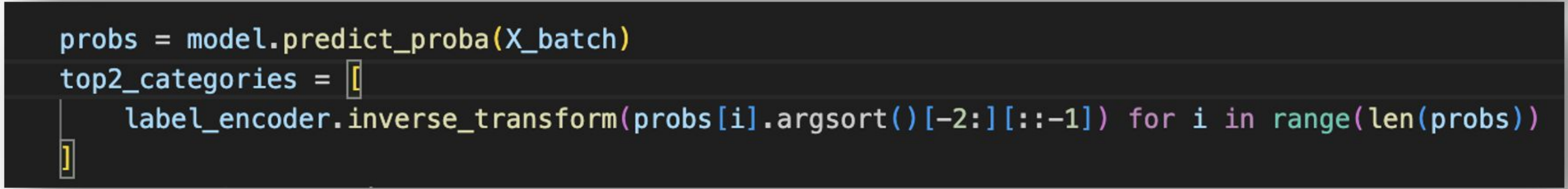


```
0,1,predicted_categories
20030219,aba decides against community broadcasting licence,"GOOD NEWS, GOOD NEWS, GOOD NEWS"
20181111,sunday november 11 full program,"POLITICS, POLITICS, RELIGION"
20040428,tas man sentenced in us court over baby battery,"CRIME, CRIME, CRIME"
20040813,mixed response to water plan,"POLITICS, POLITICS, POLITICS"
20060827,party to decide future of clp senator,"POLITICS, POLITICS, POLITICS"
```

Logistic Regression Probabilities



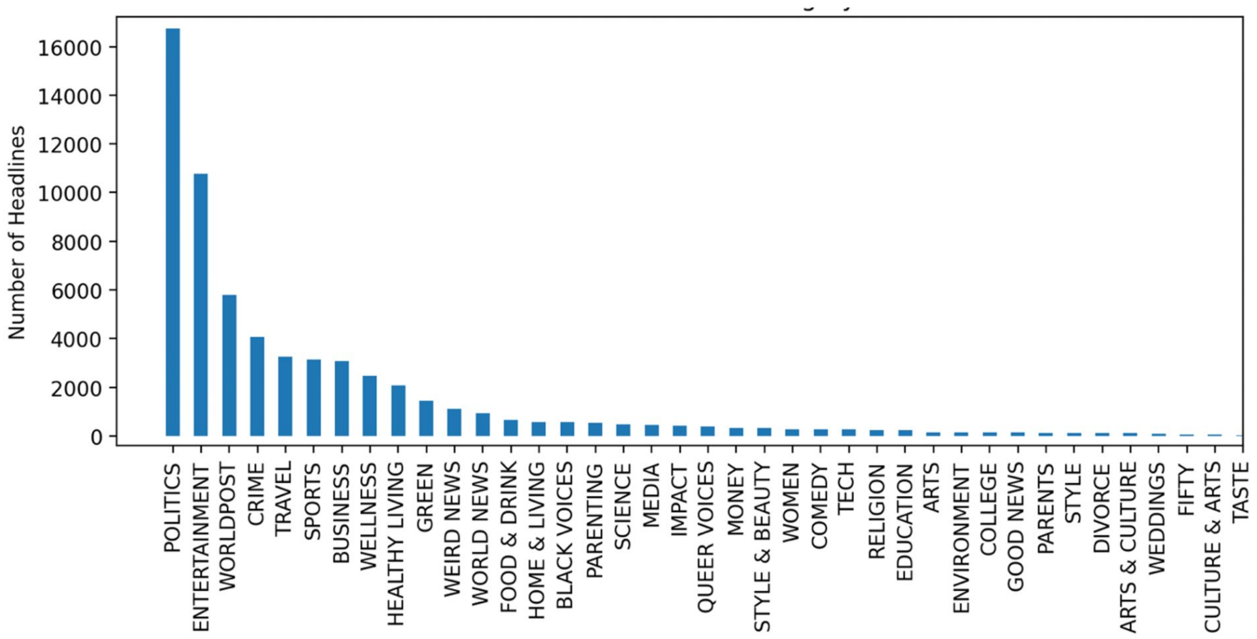
```
model = LogisticRegression(max_iter=1000, random_state=42)
model.fit(X_train, train_labels)
val_probs = model.predict_proba(X_val)
val_top2 = [label_encoder.inverse_transform(probs.argsort()[-2:][::-1]) for probs in val_probs]
```



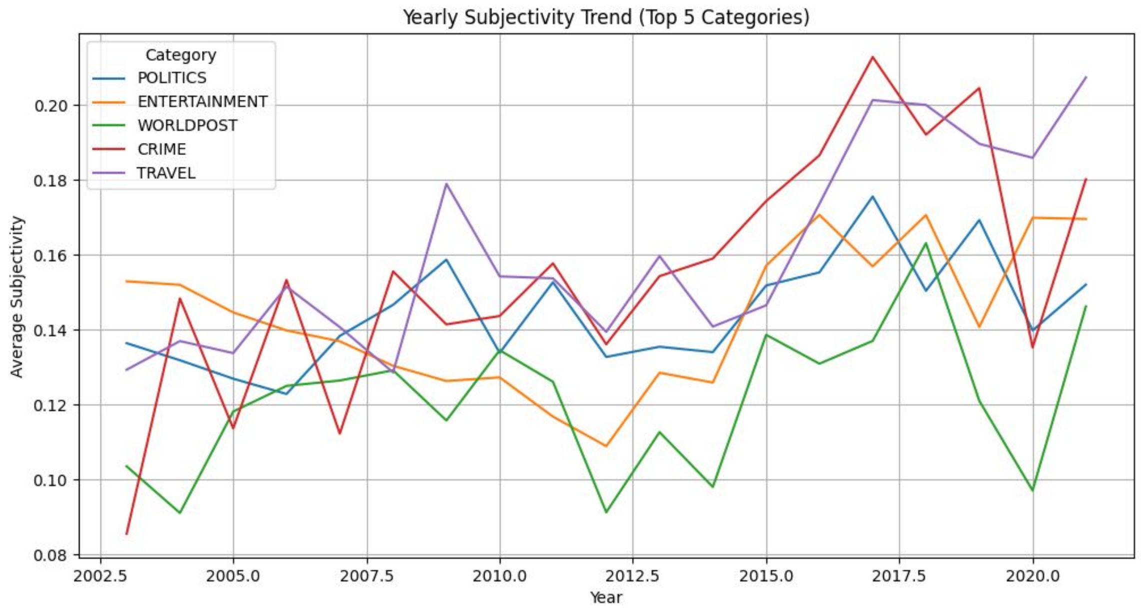
```
probs = model.predict_proba(X_batch)
top2_categories = [
    label_encoder.inverse_transform(probs[i].argsort()[-2:][::-1]) for i in range(len(probs))
]
```

Semantic Analysis based on Model two: Logistic Regression

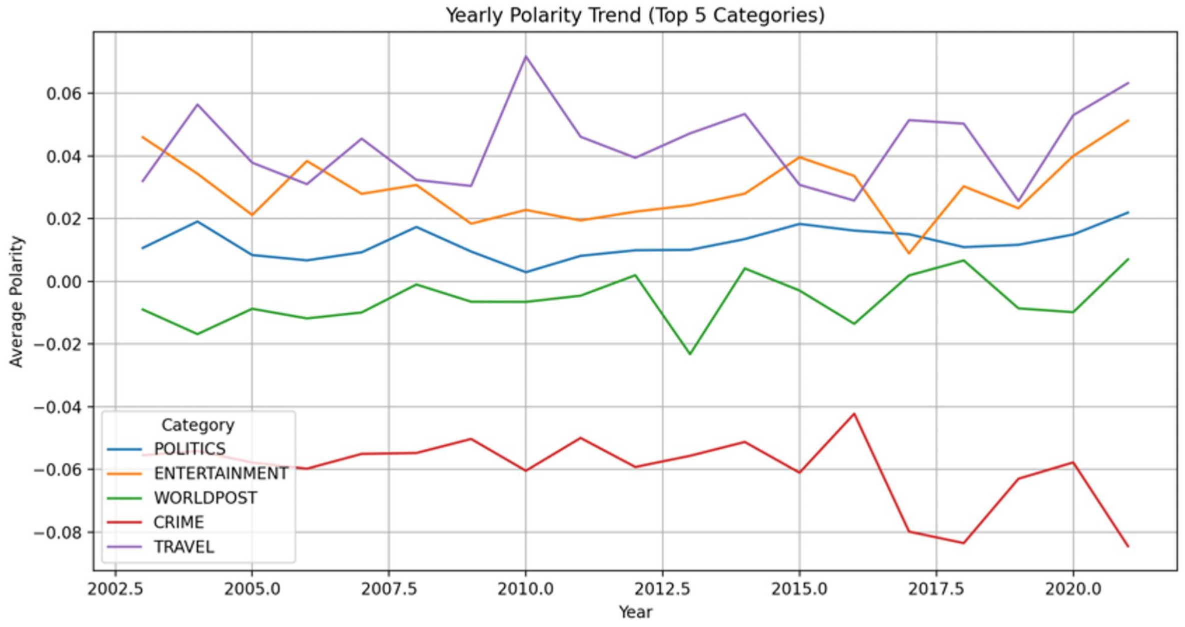
Data representation



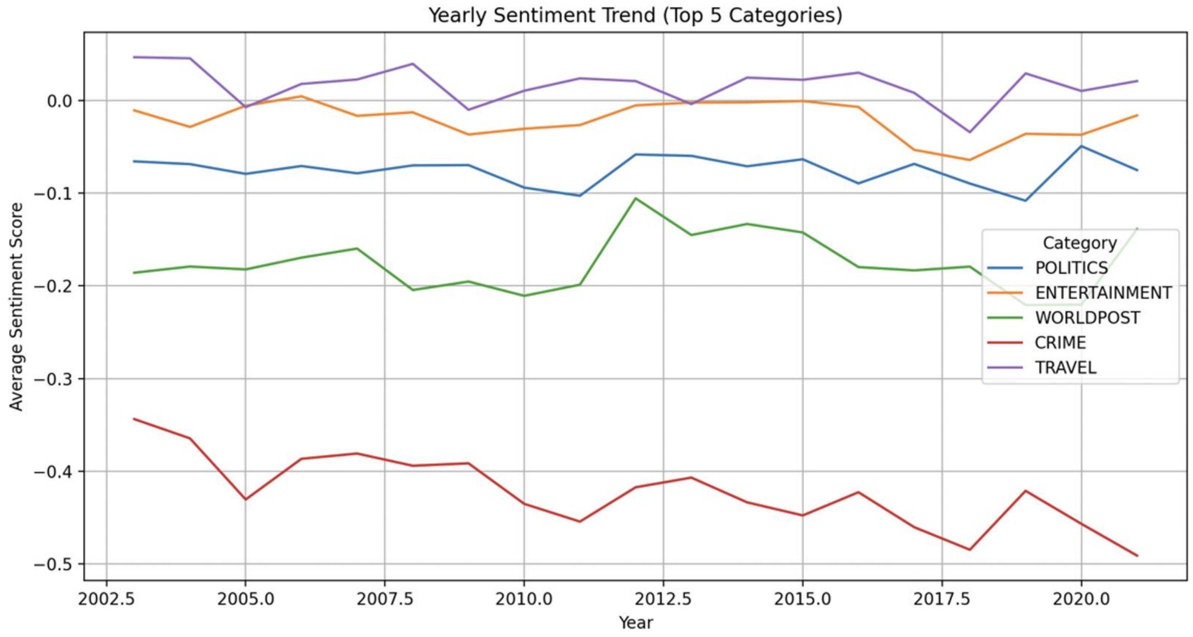
Subjectivity Analysis - Text Blob



Polarity Analysis - Text Blob



Sentiment Analysis - Vader



Conclusion

About this project

About machine learning

Thank you !