Modeling Textual Bias on Wikipedia Articles

By “Totally Bias”
(Eli Adams, Ken Klabnik, & Randy Overbeek)

Meet the Team

Who are we? What were our contributions to the project?

Eli Adams
Data Science
Graduate

Ken Klabnik
Data Science
Graduate

Randy Overbeek
Data Science
Graduate

Task Explanation

1. **How can we collect and prepare Wikipedia data for bias analysis?**
2. **What does exploratory data analysis reveal about bias, sentiment, or linguistic patterns in Wikipedia text?**
3. **Can we train a model to accurately detect bias at the sentence level?**
4. **How can we predict the overall bias of a new Wikipedia article based on sentence-level predictions?**
5. **Bonus: What types of articles tend to be more biased, and what characteristics do they share?**

Notebook Overview

- This project explores the detection of textual bias in news articles using a combination of expert-labeled datasets and natural language processing techniques.

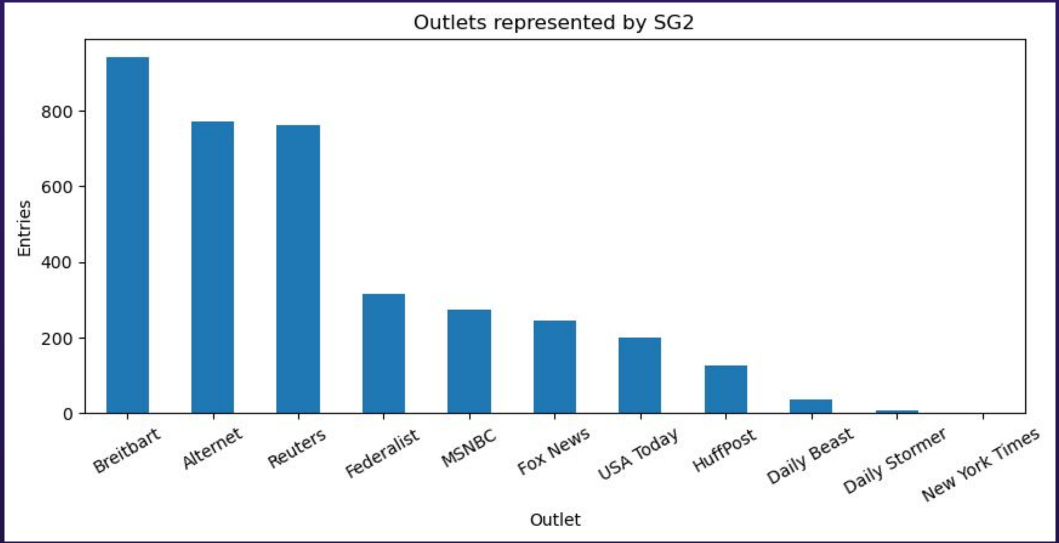
- Process
    - Loading SG1, SG2, MBIC, and bias word lexicon data files
    - Data preprocessing
    - Exploratory data analysis
    - Model building
    - Acquiring Wikipedia articles
    - Evaluating bias

Data Preprocessing

- Recode bias labels to binary for modeling: Biased = 1, Non-biased = 0
- Compute (and scale) lexicon match count by counting how many words from each sentence match words in the provided bias word lexicon
- Append article titles to their text, when available
- Vectorize text using TF-IDF
    - Term Frequency-Inverse Document Frequency, is a numerical summary reflecting how unique each word is to a text within a corpus of texts

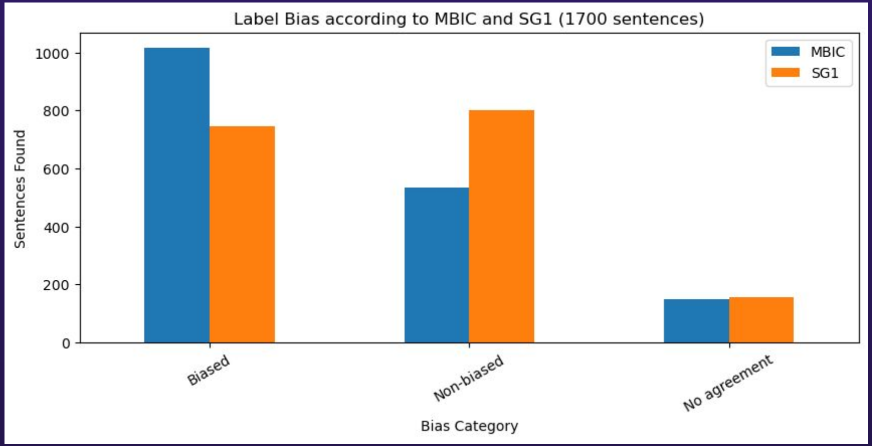
Training Data - Exploratory Analysis

- BABE (“Bias Annotated By Experts”)
- SG2 (“SubGroup 2”)
  - Contains 3,673 sentences
  - 11 outlets represented
  - Labeled by panel of five experts
  - Also contains:
    - Outlet
    - Link to source
    - List of words indicating bias
    - Topic



Training Data - Exploratory Analysis

- SG1 (BABE "Subgroup 1")
  - 1700 sentences, subset of SG2
  - Labeled by panel of eight experts
    - Same five as SG2, plus three
- Media Bias Including Characteristics
  - ("MBIC")
  - Same 1700 sentences as SG1
  - Labels crowdsourced
  - More sensitive to bias than SG1



Creating Models

- Why logistic regression?
  - In previous work, logistic regression seems to outperform tree-based models for text
  - Early testing indicated this held true for our data as well
- ROC-AUC scores:
  - SG2: 0.809
  - SG1: 0.751
  - MBIC: 0.725
- Confidence threshold calculated for bias (47%)

```python
vectorizer = TfidfVectorizer(stop_words='english')
classifier = LogisticRegression(max_iter=1000, solver='liblinear')

preprocessor = ColumnTransformer(transformers=[
    ('text', vectorizer, text_feature),
    ('num', StandardScaler(), numeric_features)
])

pipeline = Pipeline(steps=[
    ('preprocessing', preprocessor),
    ('classifier', classifier)
])

param_grid = {
    'preprocessing__text__max_features': [5000, 10000, 15000, 20000],
    'preprocessing__text__ngram_range': [(1, 1), (1, 2), (1, 3), (1, 4), (1, 5)],
    'classifier__C': [0.01, 0.1, 1.0, 10.0, 100.0]
}

X = combined_df[[text_feature] + numeric_features]
y = combined_df['label']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y, random_state=42)

grid_search = GridSearchCV(pipeline, param_grid, scoring='roc_auc', cv=3, n_jobs=-1)
grid_search.fit(X_train, y_train)

print("Best AUC:", grid_search.best_score_)
print("Best Params:", grid_search.best_params_)

y_proba = grid_search.predict_proba(X_test)[:, 1]
print("Final Test ROC AUC:", roc_auc_score(y_test, y_proba))
```

Acquiring Wikipedia Articles for Testing

```python
def fetch_article(title):
    page = wiki.page(title)
    if page.exists():
        return page.text
    else:
        raise ValueError(f"Article '{title}' not found.")
```

1. We use the wikipediaapi library to fetch full article text using the article title.
2. The text is then split into individual sentences using nltk's sent_tokenize function.

```python
def normalize_text(text):
    return re.sub(r"\s+", " ", text).strip()


def predict_bias_from_article(title, model):
    article_text = fetch_article(title)
    sentences = sent_tokenize(normalize_text(article_text))

    temp_df = pd.DataFrame({"combined_text": sentences})
    temp_df["lexicon_match_count"] = temp_df["combined_text"].apply(
        lambda x: sum(word in bias_words_set for word in str(x).lower().split())
    )

    preds = model.predict(temp_df)
    proba = model.predict_proba(temp_df)[:, 1]
    preds = (proba > best_threshold).astype(int)
    bias_score = proba.mean()

    return {
        "bias_score": round(bias_score, 3),
        "biased_sentences": int(preds.sum()),
        "total_sentences": len(sentences),
        "sentences": sentences,
        "predictions": preds,
        "probabilities": proba,
    }
```

1. Text Normalization and Sentence Splitting: The article text is cleaned to remove excess whitespace and then split into individual sentences for analysis.
2. Feature Engineering and Prediction: Each sentence is transformed into a feature vector including TF-IDF and a count of matched bias words. A logistic regression model predicts the probability of each sentence being biased.
3. Bias Scoring and Output: Sentences with probabilities above a threshold are marked as biased, and the average bias probability becomes the article's overall bias score. The function returns sentence-level results and summary statistics.

Testing a random article

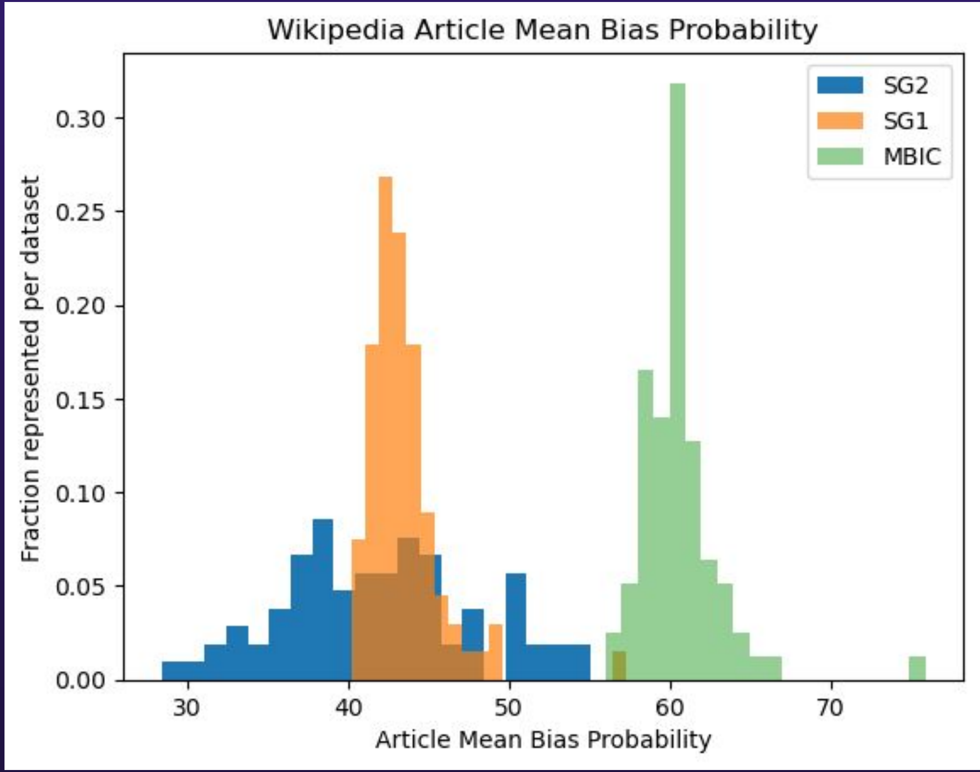
```
results = predict_bias_from_article("Donald Trump", pipeline)

print(
    f"Bias Score: {results['bias_score']} ({results['biased_sentences']} of {results['total_sentences']} sentences)"
)
```

```
Bias Score: 0.447 (255 of 563 sentences)
```

```
for sent, prob in sorted(
    zip(results["sentences"], results["probabilities"]),
    key=lambda x: x[1],
    reverse=True,
):
    if prob > best_threshold:
        print(f"⚠ {round(prob, 3)}: {sent}")
```

```
⚠ 0.977: Trump is the central figure of Trumpism, and his faction is dominant within the Republican Party.
⚠ 0.974: Racist and Islamophobic attitudes are strong indicators of support for Trump.
⚠ 0.973: Relations between the U.S. and its European allies were strained under Trump.
⚠ 0.972: He used harsher, more dehumanizing anti-immigrant rhetoric than during his presidency.
⚠ 0.972: Political practice and rhetoric Beginning with his 2016 campaign, Trump's politics and rhetoric led to the creation of a political movement known as Trumpism.
⚠ 0.969: Trump has also used anti-communist sentiment in his rhetoric, regularly calling his opponents "communists" and "Marxists".
⚠ 0.967: External links Archive of Donald Trump's tweets Appearances on C-SPAN Donald Trump at IMDb Donald Trump on the Internet Archive
```

Test Results and Observations

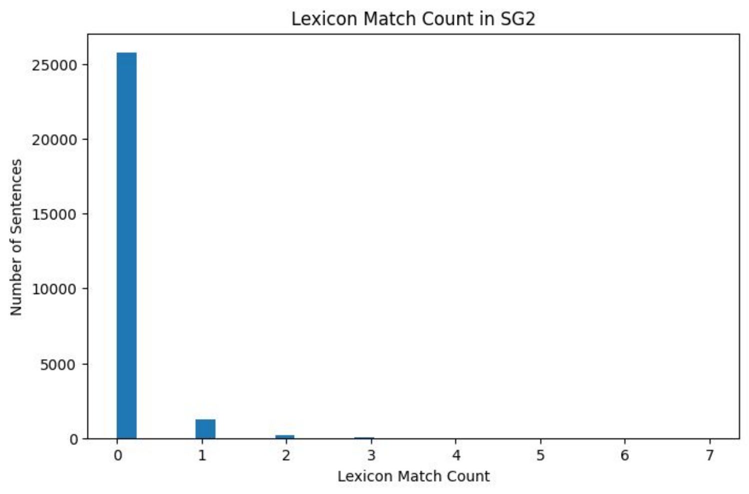


Test Results and Observations

- SG1:
  - Most biased article: “Islamophobia” (57.3% Mean Bias Probability)
  - Least biased article: “Baseball” (40.2% MBP)
    - Range: 17.1%
  - 11% of sentences flagged for bias
  - Overall MBP: 43.4%
- MBIC:
  - Most biased article: “Islamophobia” (75.8% MBP)
  - Least biased article: “Russian invasion of Ukraine” (56.0% MBP)
    - Range: 19.8%
  - 96% of sentences flagged for bias
  - Overall MBP: 60%

Test Results and Observations

- SG2:
  - Most biased article: “Gender identity” (55.1% MBP)
  - Least biased article: “Russian invasion of Ukraine” (28.4%)
    - Range: 26.7%
  - 32.6% of sentences flagged for bias
  - Overall MBP: 40.8%
- Lexicon match count:
  - Most sentences contained 0 words from lexicon
    - Lexicon match insufficient for detection



```python
sg1_bottom10_articles = (
    sg1_wiki_sentence_dataset.groupby("article_title")["bias_probability"]
    .mean()
    .sort_values(ascending=True)
    .head(10)
)
sg1_bottom10_articles
```

```
article_title
Baseball                       0.402314
Russian invasion of Ukraine    0.402762
NATO                           0.405319
Water cycle                    0.407212
Gun control                    0.408276
Planet Earth                   0.411004
Jogging                        0.411257
Nintendo                       0.411488
Milan                          0.415084
Climate change                 0.415346
Name: bias_probability, dtype: float64
```

SG1

```python
mbic_bottom10_articles = (
    mbic_wiki_sentence_dataset.groupby("article_title")["bias_probability"]
    .mean()
    .sort_values(ascending=True)
    .head(10)
)
mbic_bottom10_articles
```

```
article_title
Russian invasion of Ukraine    0.559731
LGBT adoption                  0.566437
Milan                          0.576255
Nintendo                       0.576369
NATO                           0.576744
COVID-19 pandemic              0.579492
Gun control                    0.581251
Same-sex marriage              0.582639
Norway                         0.583179
Pro-choice                     0.583352
Name: bias_probability, dtype: float64
```

MBIC

SG2

	title	bias_score	biased_sentences	total_sentences	percent_biased
18	Gender identity	0.551	146	223	0.655
24	Breitbart News	0.539	146	256	0.570
21	Fox News	0.534	363	604	0.601
33	Islamophobia	0.530	195	342	0.570
67	Caterpillar	0.522	84	137	0.613
19	Critical race theory	0.512	157	278	0.565
76	Euclid	0.508	74	131	0.565
25	The New York Times	0.508	227	412	0.551
77	Photosynthesis	0.506	180	310	0.581
75	Symbiosis	0.504	78	143	0.545
8	Pro-life	0.503	43	74	0.581
41	Creationism	0.498	160	291	0.550
68	Seahorse	0.474	105	199	0.528
34	Christian nationalism	0.473	68	117	0.581
69	Quartz	0.471	87	172	0.506
32	Evangelicalism	0.471	239	490	0.488
7	QAnon	0.470	361	688	0.525
6	Tea Party movement	0.468	194	389	0.499
22	MSNBC	0.455	153	310	0.494
28	Ukraine war	0.454	45	77	0.584
73	Snowman	0.454	36	79	0.456
13	Immigration to the United States	0.451	272	549	0.495
0	Donald Trump	0.447	255	563	0.453
23	CNN	0.446	97	199	0.487
65	Origami	0.446	87	189	0.460

	title	bias_score	biased_sentences	total_sentences	percent_biased
55	Library	0.384	70	214	0.327
38	Vaccine hesitancy	0.383	206	621	0.332
5	Bernie Sanders	0.382	240	647	0.371
59	Miocene	0.382	54	183	0.295
60	Milan	0.381	196	557	0.352
3	Barack Obama	0.375	168	518	0.324
52	Mount Everest	0.375	239	753	0.317
45	War on drugs	0.374	197	617	0.319
42	Police brutality	0.372	45	136	0.331
47	Baseball	0.371	133	417	0.319
27	Hamas	0.370	183	604	0.303
1	Joe Biden	0.367	203	681	0.298
56	Train station	0.362	55	179	0.307
51	Water cycle	0.360	29	141	0.206
2	Kamala Harris	0.358	92	312	0.295
50	Nintendo	0.357	153	491	0.312
20	Affirmative action	0.350	107	353	0.303
26	Israeli-Palestinian conflict	0.349	129	521	0.248
16	LGBT adoption	0.329	34	142	0.239
11	Gun control	0.329	39	185	0.211
4	Ron DeSantis	0.328	78	302	0.258
43	Black Lives Matter	0.320	169	665	0.254
30	NATO	0.319	47	245	0.192
37	COVID-19 pandemic	0.306	139	619	0.225
29	Russian invasion of Ukraine	0.284	132	721	0.183

Conclusion

1. **Collected bias-labeled data** from the MBIC and SG1 and SG2 datasets and scraped Wikipedia articles to test real-world predictions.

2. **Conducted EDA** to compare label distributions, outlet biases, and political leanings, revealing differences in labeling strategies.

3. **Trained three supervised models** using logistic regression with TF-IDF and lexicon features; SG2 and SG1 favored non-bias, MBIC flagged more sentences as biased.

4. **Built a prediction function** that scores entire Wikipedia articles based on sentence-level bias, enabling article-level comparisons.

5. **Identified patterns in biased articles**, with MBIC labeling politically sensitive topics more harshly, while SG1 and SG2 showed more nuance and restraint.

Extending the project

What new questions might be raised by our results?

- Can these techniques be used to detect bias in other formats?
  - Audio transcripts? Video footage? Microblogging posts?
- How can an article be changed to be perceived as less biased?
  - (Or, for that matter, more biased?)
- Do model predictions of bias by outlet match public sentiment regarding whether those outlets are prone to bias?
- Is LLM output biased? Which ones are most/least biased?
- Is there general agreement on what bias means?
  - Public vs experts

Thank you!

Any questions?