A PROJECT REPORT

on

"FAKE NEWS DETECTION - Using Machine Learning Algorithms"

Submitted to

ACADEMY OF SKILL DEVELOPMENT

BY

SUBMITTED BY : SAKET KUMAR

BTECH. CSE (2026)

KIIT UNIVERSITY , BHUBANESHWAR

ACADEMY OF SKILL DEVELOPMENT

Abstract

The proliferation of social media and online news platforms has led to a rapid increase in the dissemination of "fake news," intentionally deceptive information that can have significant negative societal impacts, including influencing democratic processes and eroding public trust. Manually fact-checking every piece of news is impractical due to the sheer volume and velocity of information, necessitating the development of automated detection systems. This project addresses the challenge by building and evaluating various machine learning models to classify news articles as either real or fake. Using the FakeNewsNet dataset, this report covers the entire machine learning pipeline, from data collection and in-depth exploratory analysis to model training and a detailed evaluation. Six different classification algorithms—Logistic Regression, Naive Bayes, Support Vector Machine (SVM), Decision Tree, Random Forest, and Gradient Boosting—are implemented. The performance of these models is systematically compared using standard evaluation metrics, including accuracy, precision, recall, F1-score, and confusion matrices. The results demonstrate the viability of using machine learning for automated fake news detection and provide detailed insights into the comparative strengths of different algorithms for this text classification task.

Contents

1. Introduction

In the digital age, information spreads faster than ever before. While this has democratized the sharing of knowledge, it has also created a fertile ground for the spread of misinformation and disinformation, commonly known as "fake news." Fake news is intentionally crafted to mislead readers, often for political or financial gain. Its potential to influence public opinion, erode trust in institutions, and incite social unrest makes it a critical problem for modern societies.

The sheer volume and velocity of information online make manual fact-checking an insufficient solution. Therefore, there is a pressing need for automated systems that can reliably detect and flag fake news in real-time. Machine learning (ML), a subset of artificial intelligence, offers a powerful set of tools for this purpose. By training models on large datasets of labeled news articles, ML systems can learn to identify the subtle linguistic patterns, stylistic cues, and source characteristics that distinguish fake news from legitimate reporting. This project aims to explore this solution by applying various ML algorithms to a real-world dataset and evaluating their effectiveness in detail.

2.Related Works

The field of fake news detection has attracted significant research interest. Early approaches often relied on manually engineered features and credibility analysis of sources. As noted in papers like "Fake News Detection on Social Media: A Data Mining Perspective," the problem can be approached by analyzing content, social context, and diffusion patterns.

Recent work, as reviewed in the provided literature, has heavily focused on machine learning. Traditional models like Naive Bayes, SVM, and Logistic Regression have been widely used as baselines, as seen in the student project reports. More advanced research, summarized in "A Comprehensive Review on Fake News Detection With Deep Learning," explores the use of deep learning models like Recurrent Neural Networks (RNNs) and Transformers (e.g., BERT), which can capture more complex linguistic nuances. The literature emphasizes two primary approaches:

1. **Content-Based Analysis:** This focuses on the text of the news article itself, analyzing features like word choice, grammar, and emotional tone.
2. **Context-Based Analysis:** This incorporates metadata, such as the source domain, author information, and how the news spreads on social media (e.g., tweet counts), to assess credibility.

This project focuses primarily on content-based analysis of the news title, as it is a strong indicator of the article's nature, while also considering metadata during the exploratory analysis phase.

3. Literature Review

The research papers offer a comprehensive overview of the fake news detection landscape, from foundational ML approaches to state-of-the-art deep learning techniques.

- The two project/internship reports (**"FAKE NEWS DETECTION USING MACHINE LEARNING"** and **"FAKE NEWS DETECTION"**) establish a practical baseline. They demonstrate the standard workflow: data collection, pre-processing using techniques like TF-IDF, training classic ML models (SVM, Logistic Regression, KNN, Decision Trees), and evaluating them based on accuracy. They confirm that even fundamental ML models can achieve reasonable performance on this task.
- **"An overview of fake news detection: From a new perspective"** provides a theoretical framework, analyzing the entire lifecycle of fake news. It uniquely identifies three key characteristics: **intentional creation** (often with malicious intent), **heteromorphic transmission** (spreading across various platforms and media formats), and **controversial reception** (generating strong, often polarized, reactions from audiences). This suggests that features related to intent and public engagement are critical for robust detection.
- **"A Comprehensive Review on Fake News Detection With Deep Learning"** delves into state-of-the-art techniques. It contrasts traditional ML models with deep learning architectures like **LSTMs**, **Attention mechanisms**, and **BERT**. The paper argues that these models can automatically learn more potent features from text without manual engineering, often outperforming classic methods by better understanding context and semantics.
- **"Detecting Fake News using Machine Learning: A Systematic Literature Review"** synthesizes the findings from numerous studies. It corroborates the difficulty humans face in detecting fake news and reinforces the necessity for automated systems. It systematically categorizes the datasets, features, and models used across the research landscape, confirming that the TF-IDF and classic classifier approach used in this project is a well-established and valid methodology.

Collectively, the literature indicates that while classic ML models are effective, the frontier of fake news detection lies in sophisticated deep learning models and the clever integration of diverse data sources(text, user engagement, source reputation).

4. Problem Statement

To design, implement, and evaluate a machine learning-based system capable of automatically classifying a given news article as "real" or "fake" with high accuracy, based on its title.

5. Objective

- To collect and load a suitable dataset containing labeled real and fake news articles.
- To perform detailed Exploratory Data Analysis (EDA) to understand the data distribution, source characteristics, and other statistical properties.
- To pre-process the textual data using standard NLP techniques to prepare it for machine learning models.
- To implement and train six different machine learning classification algorithms.
- To evaluate and compare the performance of the trained models using a comprehensive set of metrics including accuracy, precision, recall, F1-score, and confusion matrices.
- To identify the best-performing model for this specific task and analyze its results in depth.
- To conclude with the key findings and discuss the limitations and future scope of the project.

6. Methodology

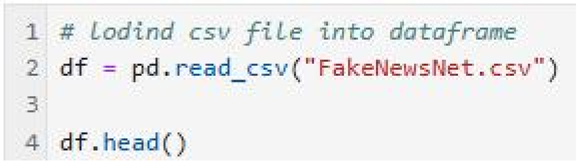
a. Data Collection

i. **Resource:** The dataset used for this project is **FakeNewsNet**, provided as a CSV file (FakeNewsNet.csv).

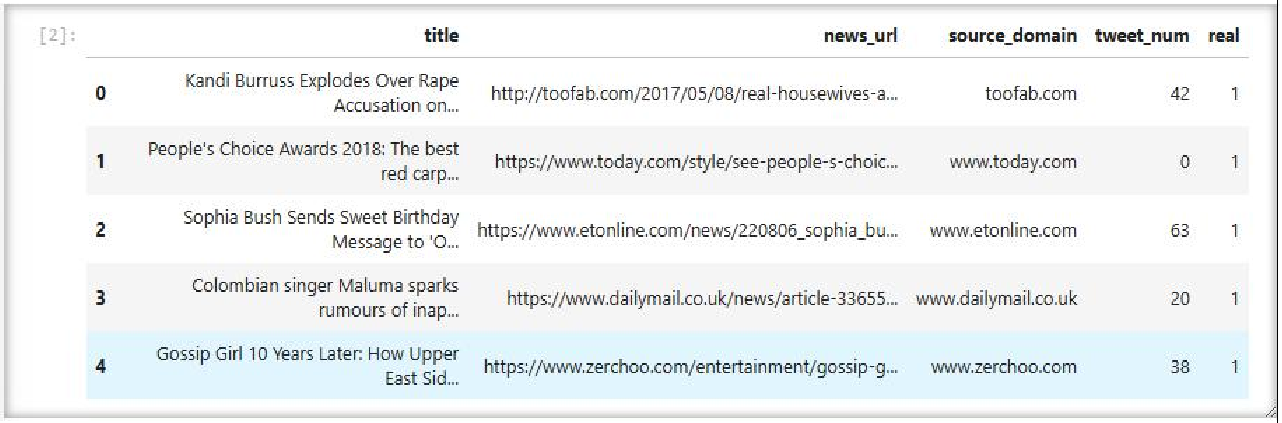
ii. **Describe the Data Set:** The dataset contains 23,196 news articles. It is imbalanced, with 17,441 (approx. 75%) labeled as real and 5,755 (approx. 25%) as fake. Each record has 5 attributes:

- title: The headline of the news article (object/string).
- news_url: The URL of the original article (object/string).
- source_domain: The website domain from which the article originated (object/string).
- tweet_num: The number of times the article was tweeted (int64).
- real: The target label, where 1 indicates real news and 0 indicates fake news (int64).

iii. Screenshot of the Code & Output:
Code to load and inspect the data:
import pandas as pd
-> Loading and printing the dataset



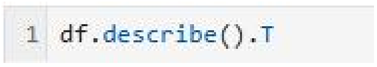
```python
# lodind csv file into dataframe
df = pd.read_csv("FakeNewsNet.csv")

df.head()
```



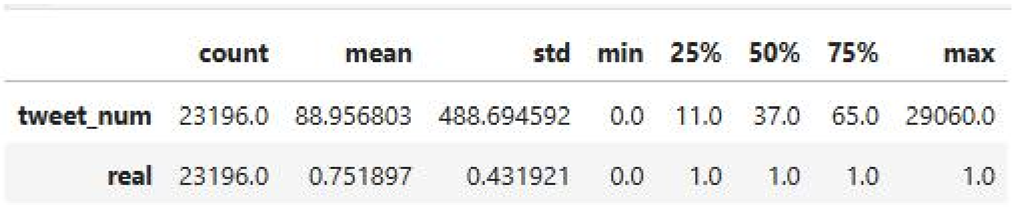
```
[2]:
```

	title	news_url	source_domain	tweet_num	real
0	Kandi Burruss Explodes Over Rape Accusation on...	http://toofab.com/2017/05/08/real-housewives-a...	toofab.com	42	1
1	People's Choice Awards 2018: The best red carp...	https://www.today.com/style/see-people-s-choic...	www.today.com	0	1
2	Sophia Bush Sends Sweet Birthday Message to 'O...	https://www.etonline.com/news/220806_sophia_bu...	www.etonline.com	63	1
3	Colombian singer Maluma sparks rumours of inap...	https://www.dailymail.co.uk/news/article-33655...	www.dailymail.co.uk	20	1
4	Gossip Girl 10 Years Later: How Upper East Sid...	https://www.zerchoo.com/entertainment/gossip-g...	www.zerchoo.com	38	1

-> Describing the dataset

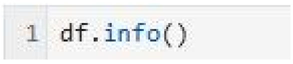


```python
df.describe().T
```

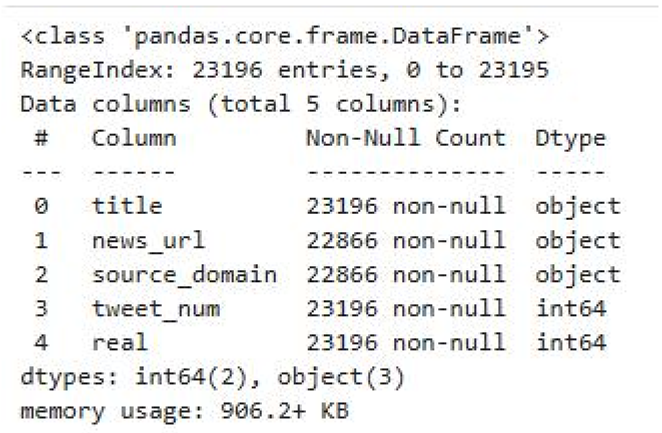


	count	mean	std	min	25%	50%	75%	max
tweet_num	23196.0	88.956803	488.694592	0.0	11.0	37.0	65.0	29060.0
real	23196.0	0.751897	0.431921	0.0	1.0	1.0	1.0	1.0

-> Dataframe info



```python
df.info()
```



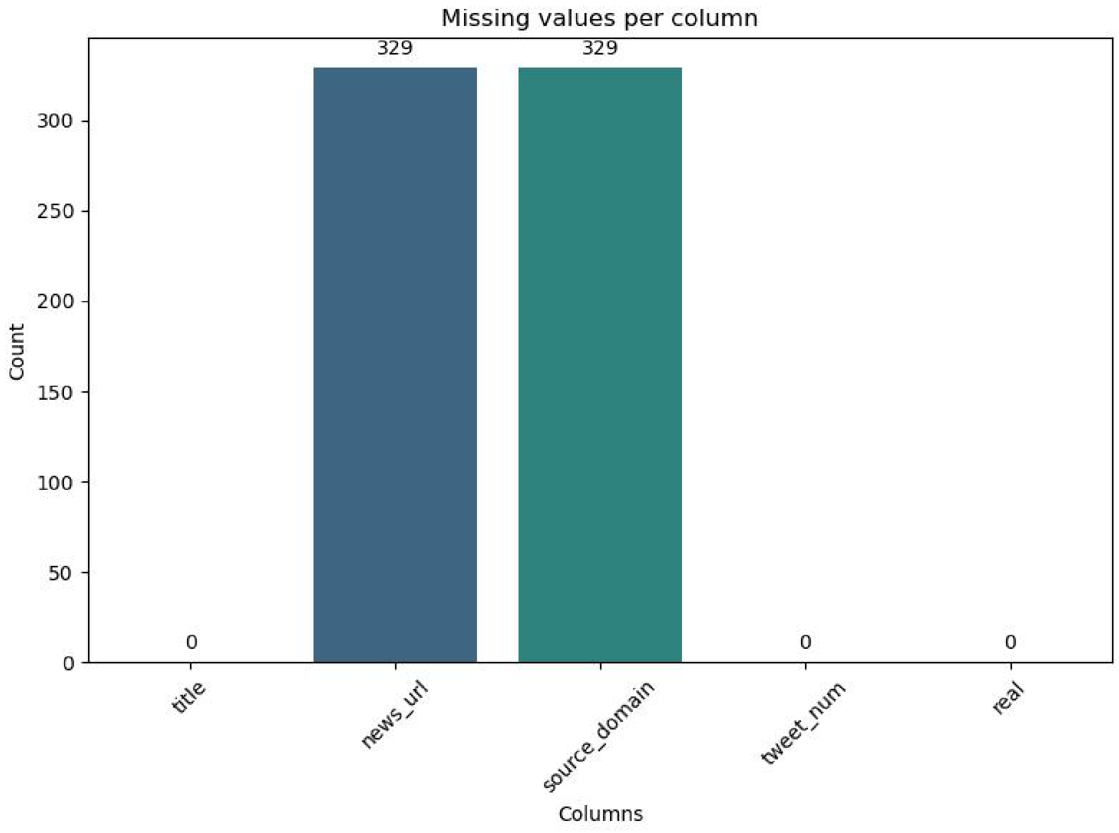
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23196 entries, 0 to 23195
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   title          23196 non-null  object
 1   news_url       22866 non-null  object
 2   source_domain  22866 non-null  object
 3   tweet_num      23196 non-null  int64 
 4   real           23196 non-null  int64 
dtypes: int64(2), object(3)
memory usage: 906.2+ KB
```

b. Data Pre-processing

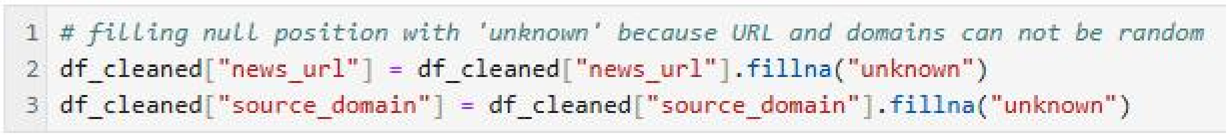
i. Algorithms Description with Mathematical Notation:
Data pre-processing involves cleaning and transforming the raw data. For this project, we primarily focus on the title feature.

- **Handling Missing Values:** We fill any missing titles with an empty string to ensure no errors during processing.

-> Pictorial Representation of Null Values



-> Filling null with ‘unknown”

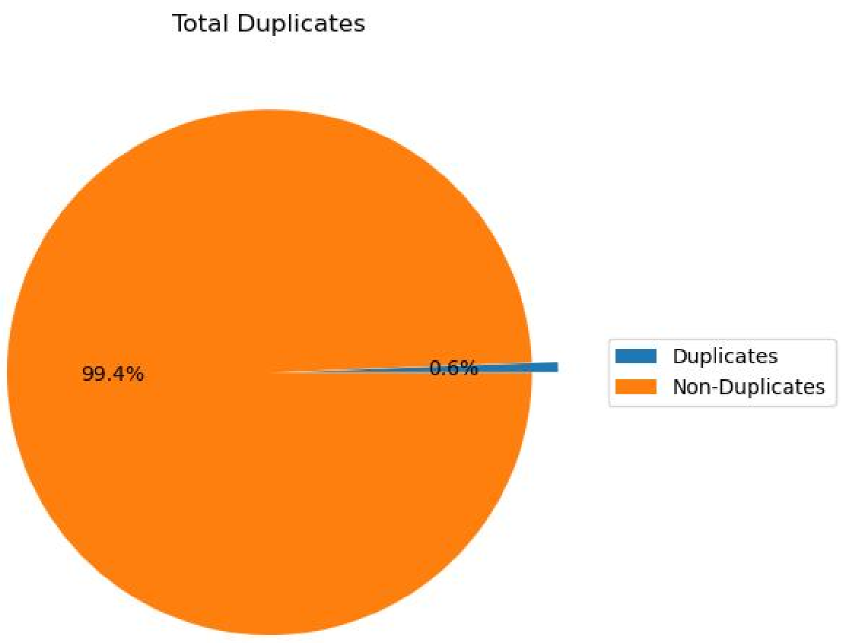


```
# filling null position with 'unknown' because URL and domains can not be random
df_cleaned["news_url"] = df_cleaned["news_url"].fillna("unknown")
df_cleaned["source_domain"] = df_cleaned["source_domain"].fillna("unknown")
```

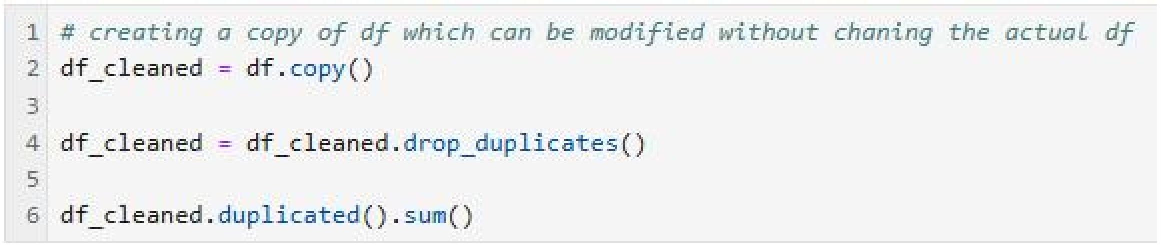
- **Handling Duplicate Values:** Duplicate values need to be removed to avoid bias in data.

The statistics shows that the number of duplicate values are very less (~0.6%)

Still these need to be removed because a small sample size of these data can lead to significant amount of bias if these duplicates are of relevant data features (here sources)

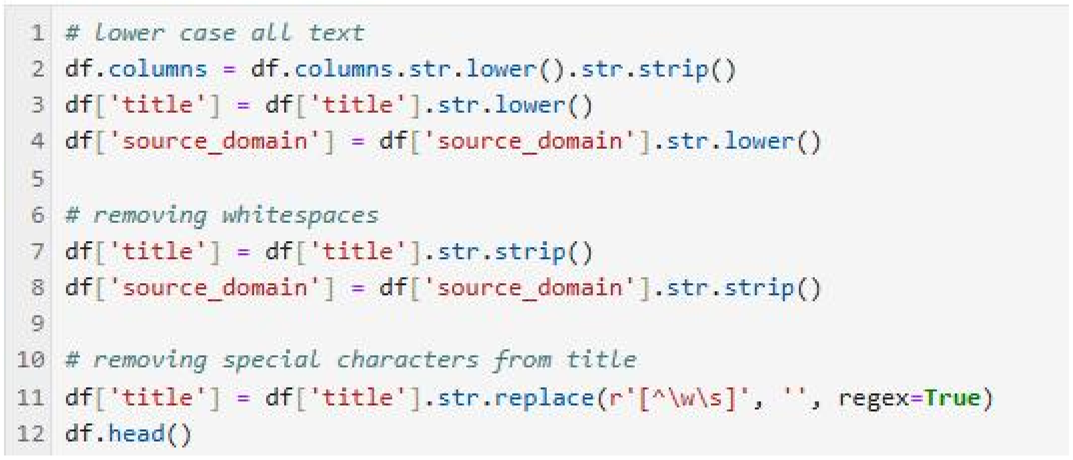


-> Deleting duplicate values

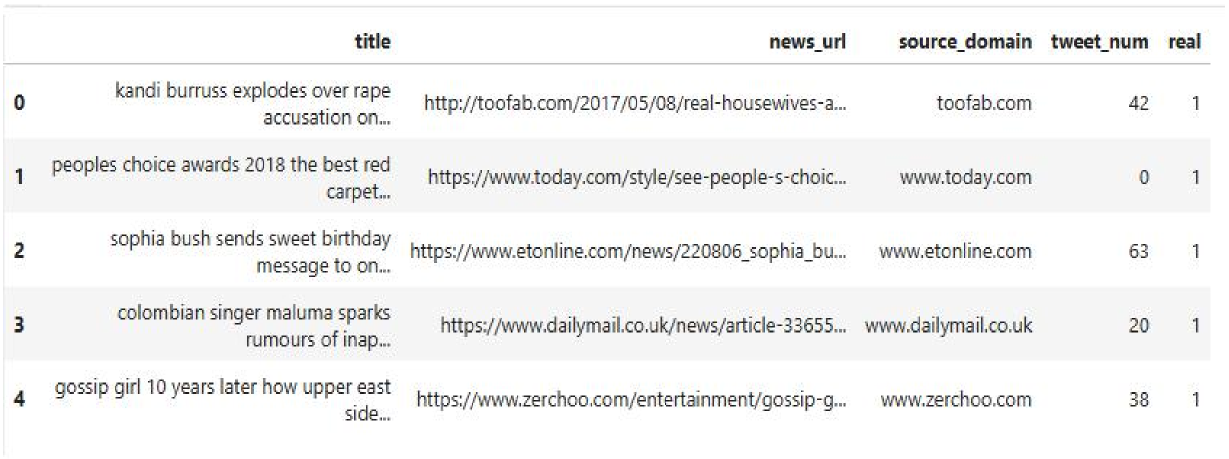


```python
# creating a copy of df which can be modified without chaning the actual df
df_cleaned = df.copy()

df_cleaned = df_cleaned.drop_duplicates()

df_cleaned.duplicated().sum()
```

- **Text Cleaning:** Text is converted to lowercase to maintain consistency (e.g., "News" and "news" are treated as the same word).

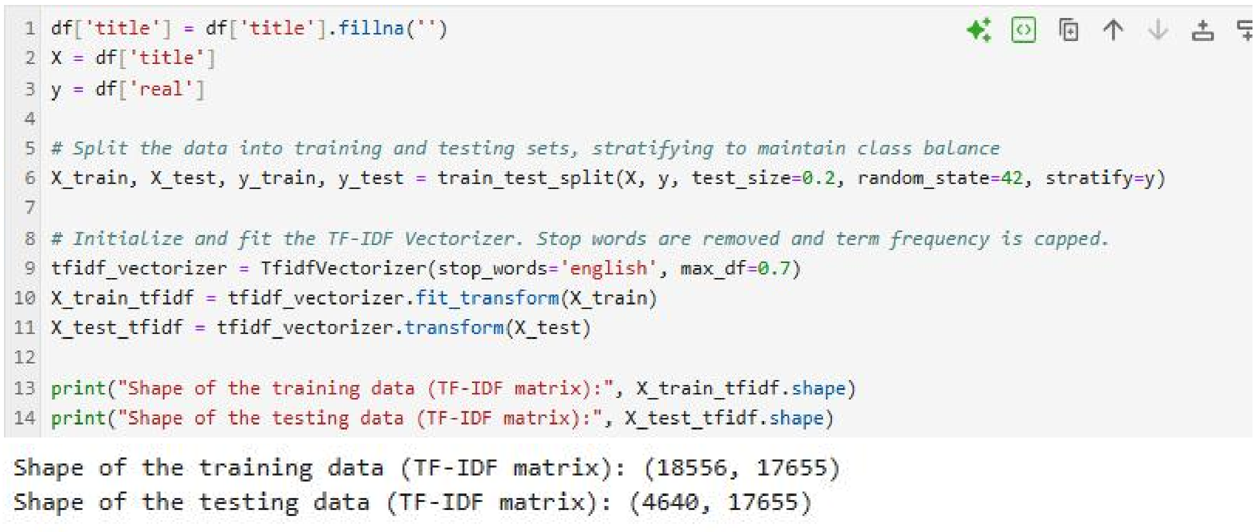


```python
# lower case all text
df.columns = df.columns.str.lower().str.strip()
df['title'] = df['title'].str.lower()
df['source_domain'] = df['source_domain'].str.lower()

# removing whitespaces
df['title'] = df['title'].str.strip()
df['source_domain'] = df['source_domain'].str.strip()

# removing special characters from title
df['title'] = df['title'].str.replace(r'[^\w\s]', '', regex=True)
df.head()
```



	title	news_url	source_domain	tweet_num	real
0	kandi burruss explodes over rape accusation on...	http://toofab.com/2017/05/08/real-housewives-a...	toofab.com	42	1
1	peoples choice awards 2018 the best red carpet...	https://www.today.com/style/see-people-s-choic...	www.today.com	0	1
2	sophia bush sends sweet birthday message to on...	https://www.etonline.com/news/220806_sophia_bu...	www.etonline.com	63	1
3	colombian singer maluma sparks rumours of inap...	https://www.dailymail.co.uk/news/article-33655...	www.dailymail.co.uk	20	1
4	gossip girl 10 years later how upper east side...	https://www.zerchoo.com/entertainment/gossip-g...	www.zerchoo.com	38	1

- **Text Vectorization (TF-IDF):** Since machine learning models require numerical input, we convert the text titles into a matrix of numbers using the Term Frequency-Inverse Document Frequency (TF-IDF) method. This method reflects how important a word is to a document in a collection.
  - Term Frequency (TF): Measures how frequently a term t appears in a document d. TF(t,d)=Total number of terms in document dNumber of times term t appears in document d
  - Inverse Document Frequency (IDF): Measures how important a term is across all documents. It penalizes common words (like 'the', 'a'), which are filtered out as stop words. IDF(t,D)=log(Number of documents with term t in itTotal number of documents)
  - TF-IDF Score: The final score is the product of TF and IDF. TFIDF(t,d,D)=TF(t,d)×IDF(t,D)



```
df['title'] = df['title'].fillna('')
X = df['title']
y = df['real']

# Split the data into training and testing sets, stratifying to maintain class balance
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

# Initialize and fit the TF-IDF Vectorizer. Stop words are removed and term frequency is capped.
tfidf_vectorizer = TfidfVectorizer(stop_words='english', max_df=0.7)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)

print("Shape of the training data (TF-IDF matrix):", X_train_tfidf.shape)
print("Shape of the testing data (TF-IDF matrix):", X_test_tfidf.shape)
```

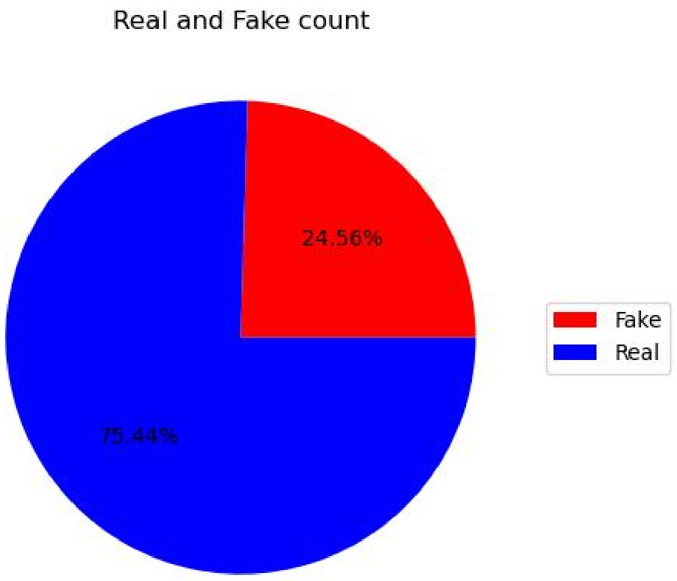
```
Shape of the training data (TF-IDF matrix): (18556, 17655)
Shape of the testing data (TF-IDF matrix): (4640, 17655)
```

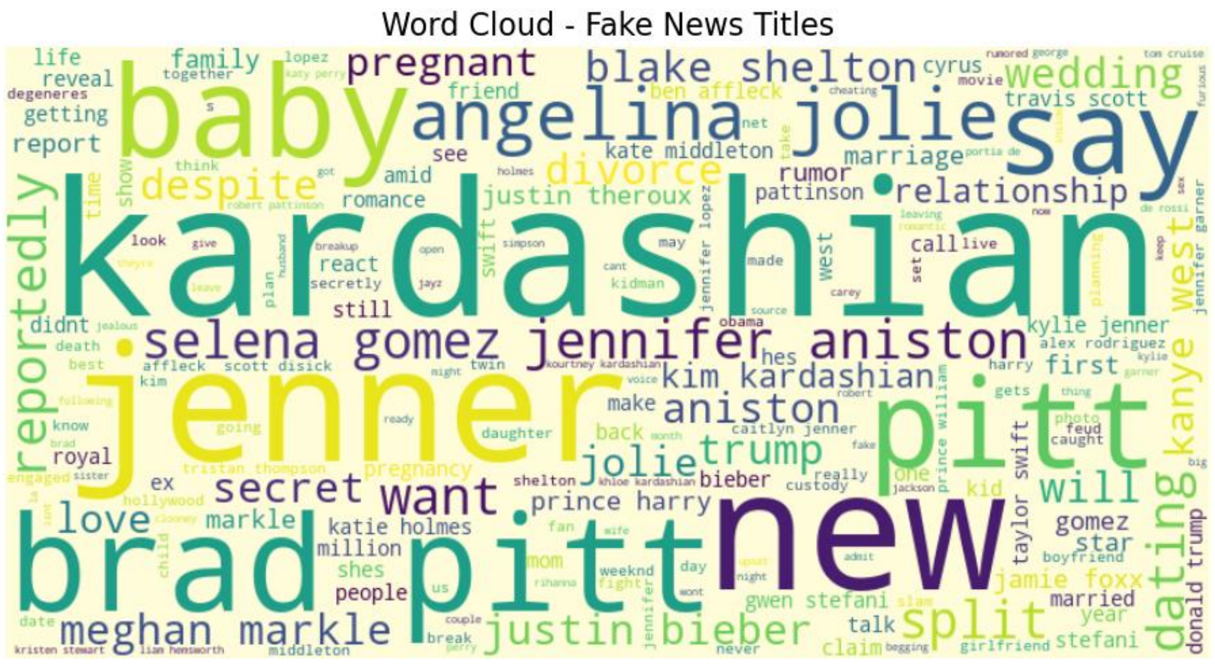
c. Exploratory Data Analysis

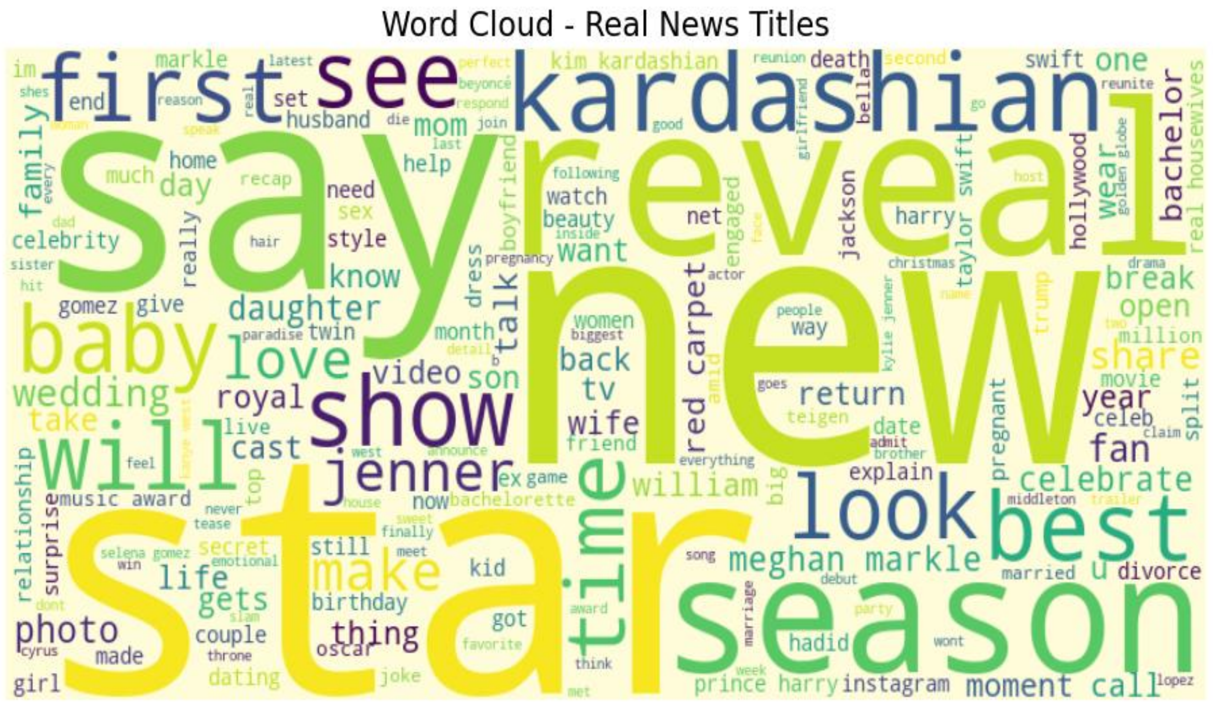
Visualise Data for Statistical Analysis:

EDA helps in understanding the underlying structure of the data.

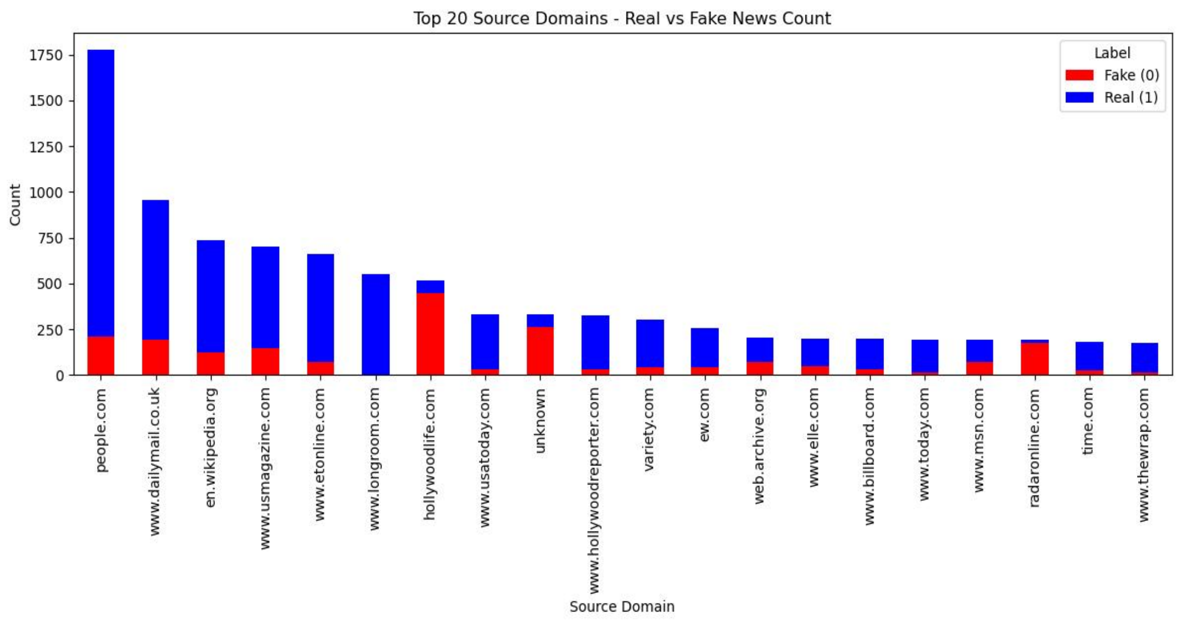
- Distribution of Real vs. Fake News: The bar chart below clearly shows the class imbalance in the dataset, with real news articles being far more numerous than fake ones.



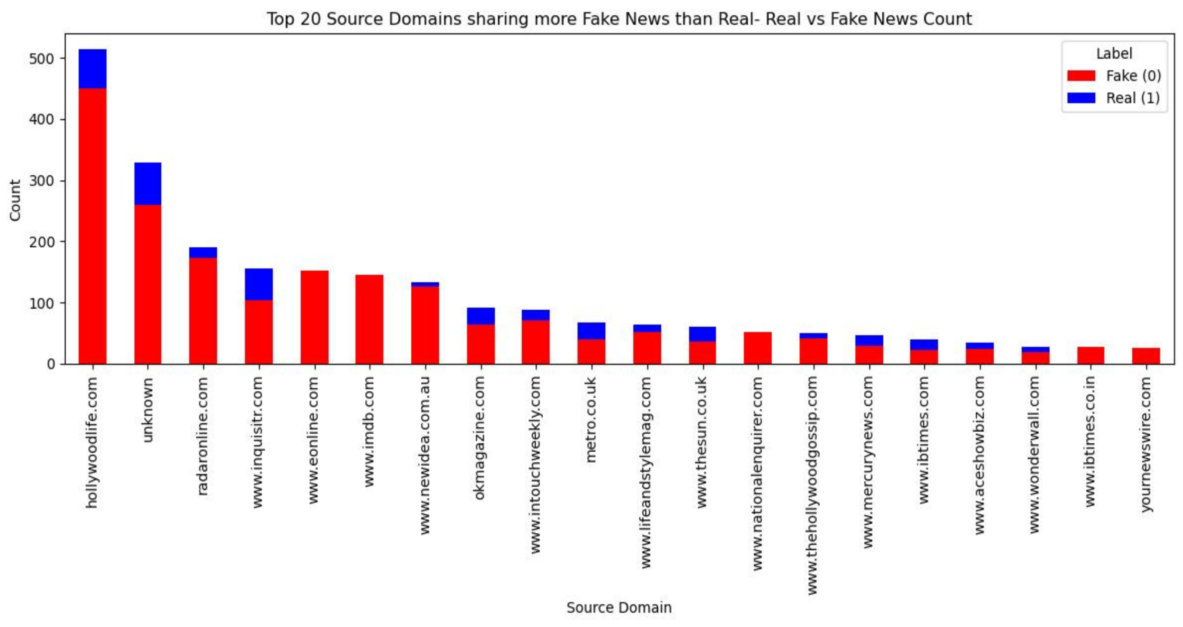


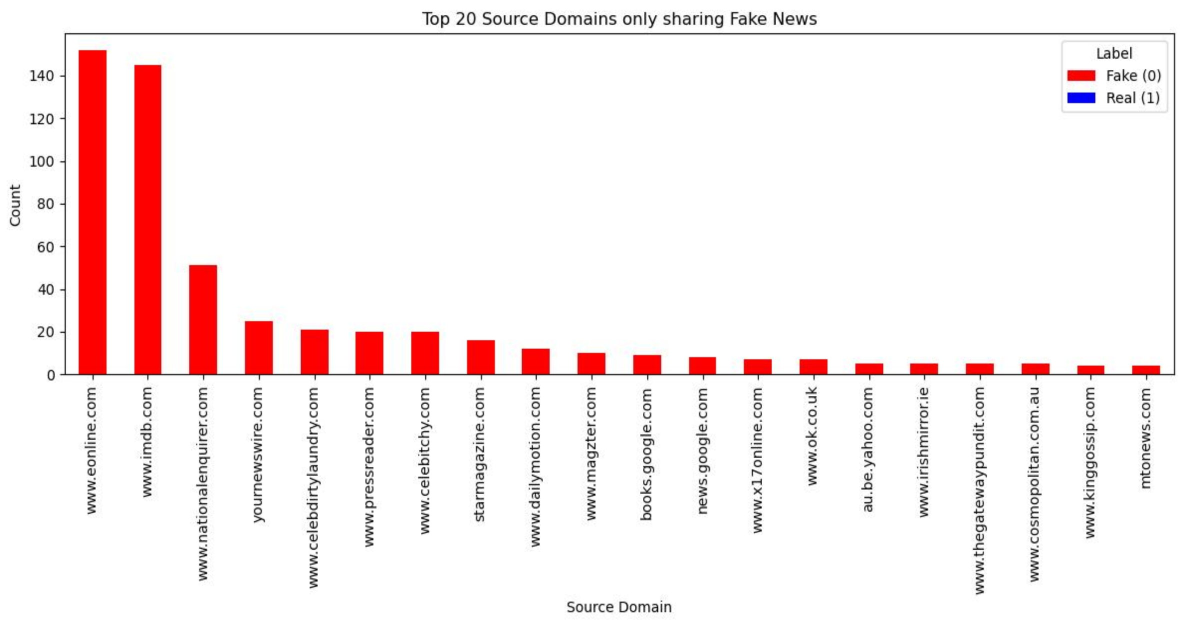


- **Top Source Domains:** Analyzing the source domains reveals interesting patterns. While some sources are predominantly associated with real news, others appear on both lists, suggesting they publish a mix of content or are perceived differently by the dataset's labelers.

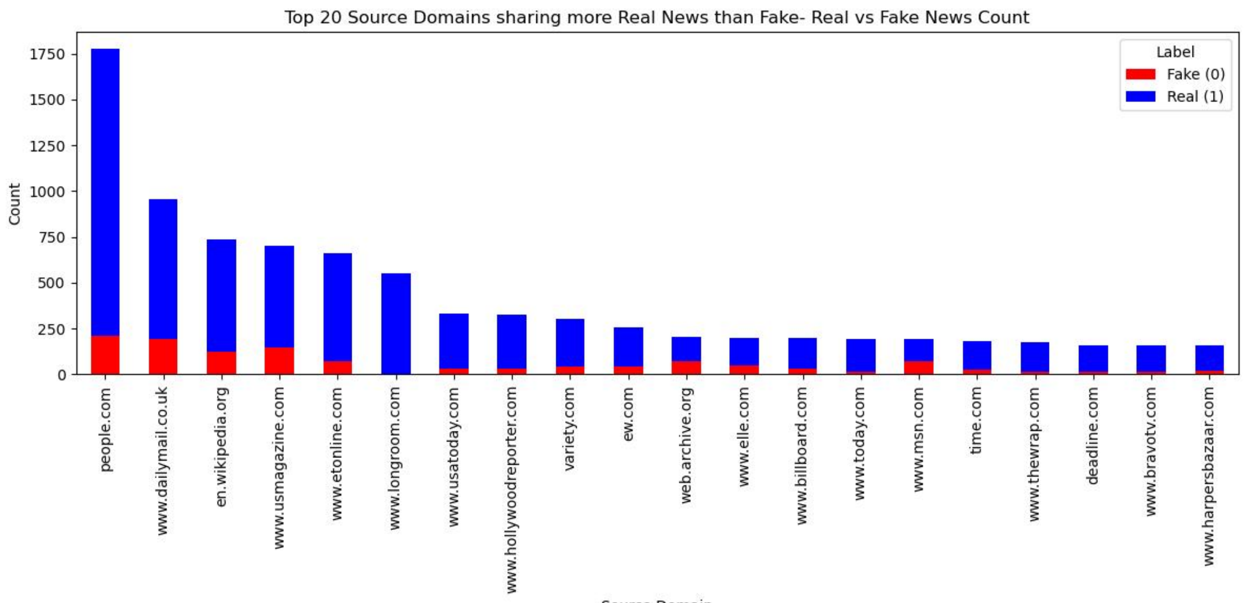


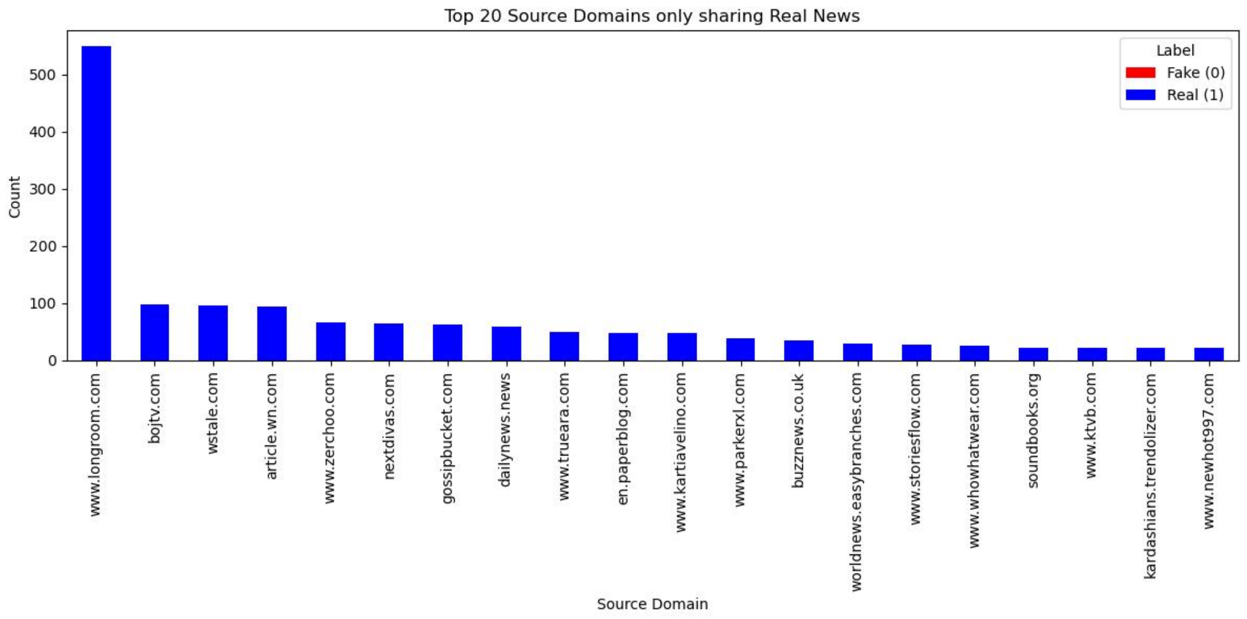
  - Top 10 Source Domains for Real News:


Top 20 Source Domains sharing more Fake News than Real- Real vs Fake News Count
Label
Fake (0)
Real (1)
Count
Source Domain
hollywoodlife.com
unknown
radaronline.com
www.inquisitr.com
www.eonline.com
www.imdb.com
www.newidea.com.au
okmagazine.com
www.intouchweekly.com
metro.co.uk
www.lifeandstylemag.com
www.thesun.co.uk
www.nationalenquirer.com
www.thehollywoodgossip.com
www.mercurynews.com
www.ibtimes.com
www.aceshowbiz.com
www.wonderwall.com
www.ibtimes.co.in
yournewswire.com


Top 20 Source Domains only sharing Fake News
Label
Fake (0)
Real (1)
Count
Source Domain
www.eonline.com
www.imdb.com
www.nationalenquirer.com
yournewswire.com
www.celebdirtylaundry.com
www.pressreader.com
www.celebitchy.com
starmagazine.com
www.dailymotion.com
www.magzter.com
books.google.com
news.google.com
www.x17online.com
www.ok.co.uk
au.be.yahoo.com
www.irishmirror.ie
www.thegatewaypundit.com
www.cosmopolitan.com.au
www.kinggossip.com
mtonews.com

- Top 10 Source Domains for Fake News:





This analysis shows a clear imbalance in the dataset and highlights that some domains (e.g., people.com, www.dailymail.co.uk) are frequent sources for both categories.

d. Machine Learning Model Creation

i. **Use at least 6 AI algorithms:**

1. Logistic Regression
2. Multinomial Naive Bayes
3. Support Vector Machine (SVM)
4. Decision Tree
5. Random Forest
6. Gradient Boosting

ii. **Short Description with Mathematical Notation:**

1. **Logistic Regression:** A robust linear model that uses the sigmoid function to output a probability. It's often a strong baseline for text classification.
   - *Sigmoid function:* $\sigma(z) = \frac{1}{1 + e^{-z}}$, where $z = \omega \cdot x + b$
2. **Multinomial Naive Bayes:** A probabilistic classifier based on Bayes' Theorem with a "naive" assumption of feature independence. It's computationally efficient and performs well on text data.
   - *Bayes' Theorem:* $P(y \mid x_1, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)}$
3. **Support Vector Machine (SVM):** A powerful model that finds an optimal hyperplane to separate classes. It can be very effective in high-dimensional spaces like the one created by TF-IDF.
   - *Hyperplane:* $\omega \cdot x - b = 0$
4. **Decision Tree:** A non-parametric model with a tree-like structure. While easy to interpret, single trees can easily overfit the training data.
   - *Gini Impurity:* $G = \sum_{k=1}^{K} p_k (1 - p_k)$
5. **Random Forest:** An ensemble method that builds multiple decision trees on data subsets and aggregates their results, reducing overfitting and improving accuracy over a single tree.
6. **Gradient Boosting:** An ensemble technique that builds models sequentially, where each new model corrects the errors of its predecessor. It is often a top-performing algorithm but can be computationally intensive.

```python
# creating a function to represent results of all model
def plot_confusion_matrix(y_true, y_pred, model_name):
    cm = confusion_matrix(y_true, y_pred)

    plt.figure(figsize=(8,6))
    sns.heatmap(cm,
                annot=True,
                fmt='d',
                cmap="cividis",
                xticklabels=["Fake","Real"],
                yticklabels=["Fake","Real"]
               )

    plt.title(f"Confusion Matrix = {model_name}", fontsize=16)
    plt.xlabel("Predicted Label", fontsize=12)
    plt.ylabel("Actual Label", fontsize=12)
    plt.show()
```

```python
# defining all models
# (model name, model, hyper-parameters)
models = [("Logistic Regression",
           LogisticRegression(max_iter=1000, random_state=42),
           {"classifier__C" : uniform(0.1,10),
            "classifier__solver": ["liblinear", "saga"]
           }
          ),
          ("Multinomial Naive Bayes",
           MultinomialNB(),
           {"classifier__alpha": uniform(0.01, 1.0)}
          ),
          ("Linear SVM",
           LinearSVC(max_iter=2000, random_state=42),
           {"classifier__C": uniform(0.1,10)}
          ),
          ("Random Forest",
           RandomForestClassifier(random_state=42),
           {"classifier__n_estimators": randint(100,500),
            "classifier__max_depth": [5, 10, 20, 30],
            "classifier__min_samples_leaf":randint(1,5)
           }
          ),
          ("LightGBM",
           LGBMClassifier(random_state=42),
           {"classifier__n_estimators": randint(100,500),
            "classifier__learning_rate": uniform(0.01,0.2),
            "classifier__num_leaves": randint(20,50)
           }
          )]

results = {}
```

e. Result & Evaluation

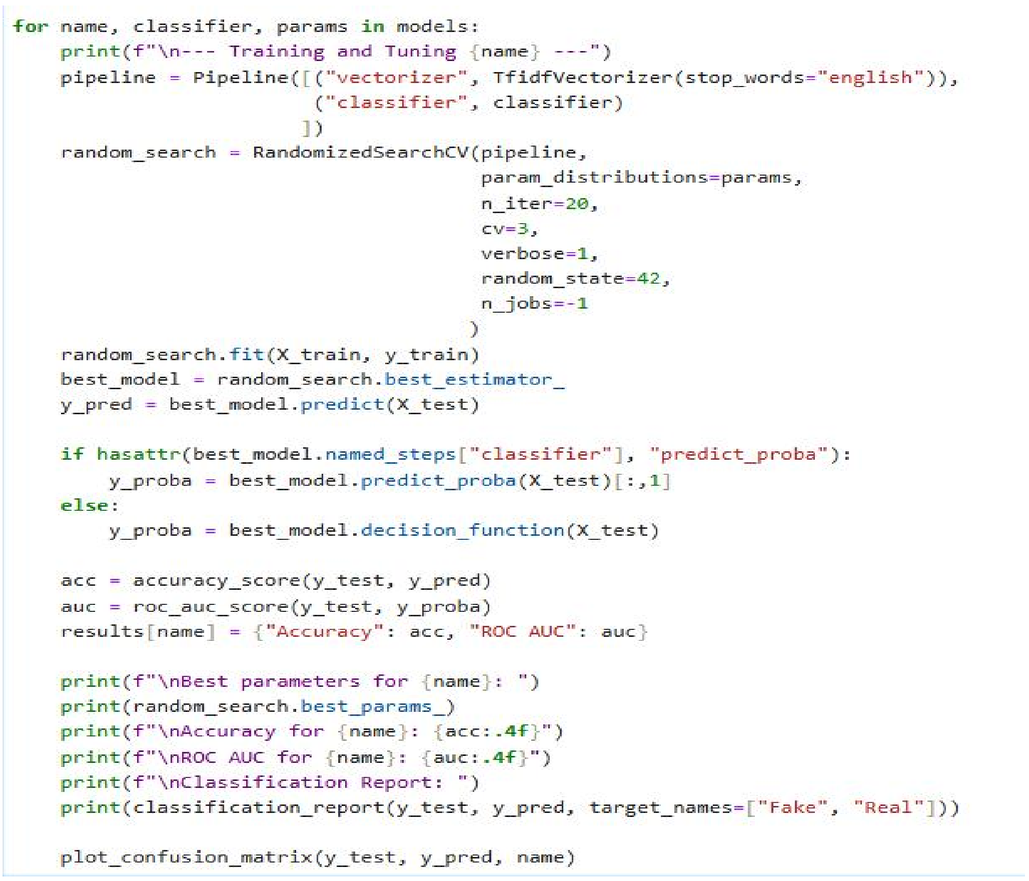
i. Describe Methods with Mathematical Notation:

Model performance is evaluated using metrics derived from the confusion matrix (True Positives (TP), True Negatives (TN), False Positives (FP), False Negatives (FN)):

- Accuracy: The ratio of correctly predicted instances. Accuracy = TP + TN + FP + FNTP + TN
- Precision: The model's ability to not label a negative sample as positive. Precision = TP + FPTP
- Recall (Sensitivity): The model's ability to find all the positive samples. Recall=TP + FNTP
- F1-Score: The harmonic mean of Precision and Recall, crucial for imbalanced datasets.
  F1−Score = 2 × Precision + RecallPrecision × Recall

ii. Code & Output (Graph):

Code to evaluate models and visualize results:



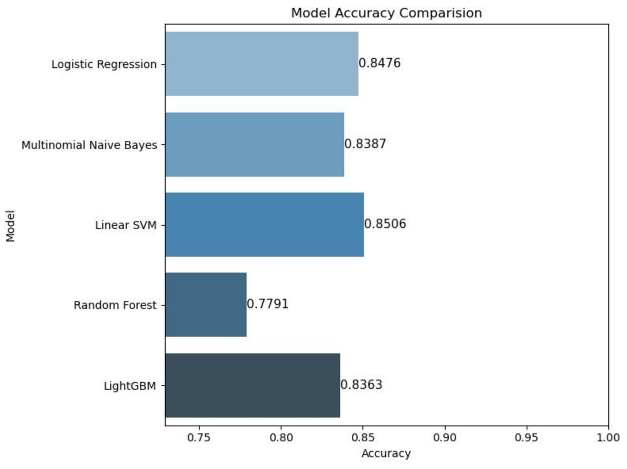
```
for name, classifier, params in models:
    print(f"\n--- Training and Tuning {name} ---")
    pipeline = Pipeline([("vectorizer", TfidfVectorizer(stop_words="english")),
                         ("classifier", classifier)
                        ])
    random_search = RandomizedSearchCV(pipeline,
                                       param_distributions=params,
                                       n_iter=20,
                                       cv=3,
                                       verbose=1,
                                       random_state=42,
                                       n_jobs=-1
                                      )
    random_search.fit(X_train, y_train)
    best_model = random_search.best_estimator_
    y_pred = best_model.predict(X_test)

    if hasattr(best_model.named_steps["classifier"], "predict_proba"):
        y_proba = best_model.predict_proba(X_test)[:,1]
    else:
        y_proba = best_model.decision_function(X_test)

    acc = accuracy_score(y_test, y_pred)
    auc = roc_auc_score(y_test, y_proba)
    results[name] = {"Accuracy": acc, "ROC AUC": auc}

    print(f"\nBest parameters for {name}: ")
    print(random_search.best_params_)
    print(f"\nAccuracy for {name}: {acc:.4f}")
    print(f"\nROC AUC for {name}: {auc:.4f}")
    print(f"\nClassification Report: ")
    print(classification_report(y_test, y_pred, target_names=["Fake", "Real"]))

    plot_confusion_matrix(y_test, y_pred, name)
```

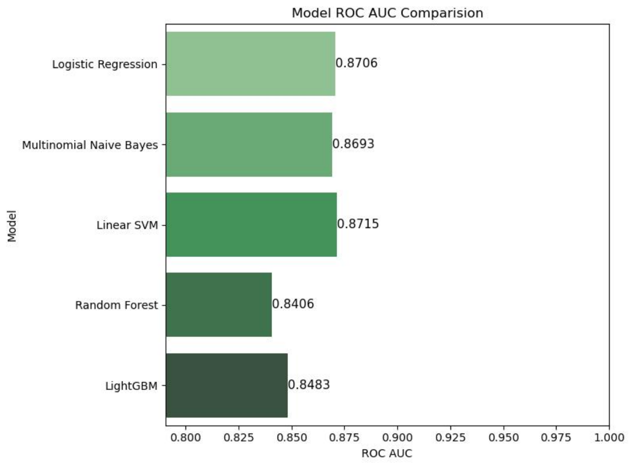
Output Graphs and Tables:

Model Performance Comparison

The bar chart below compares the overall accuracy of the six trained models. Logistic

Regression and SVM show the highest performance.

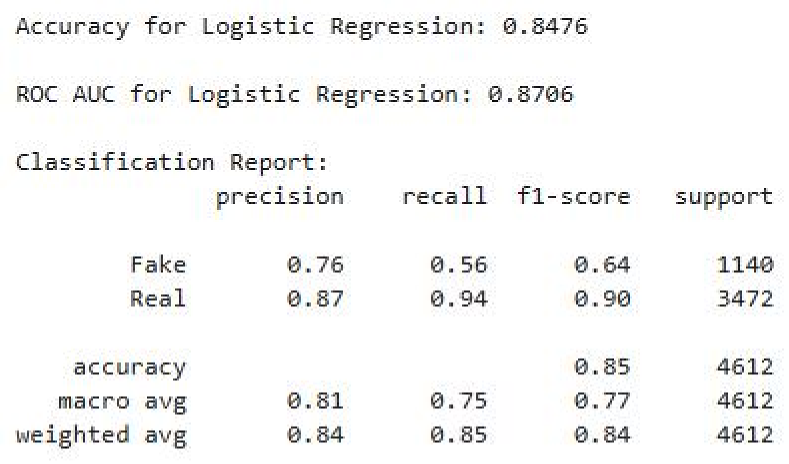




Detailed Analysis of the Best Model: Logistic Regression

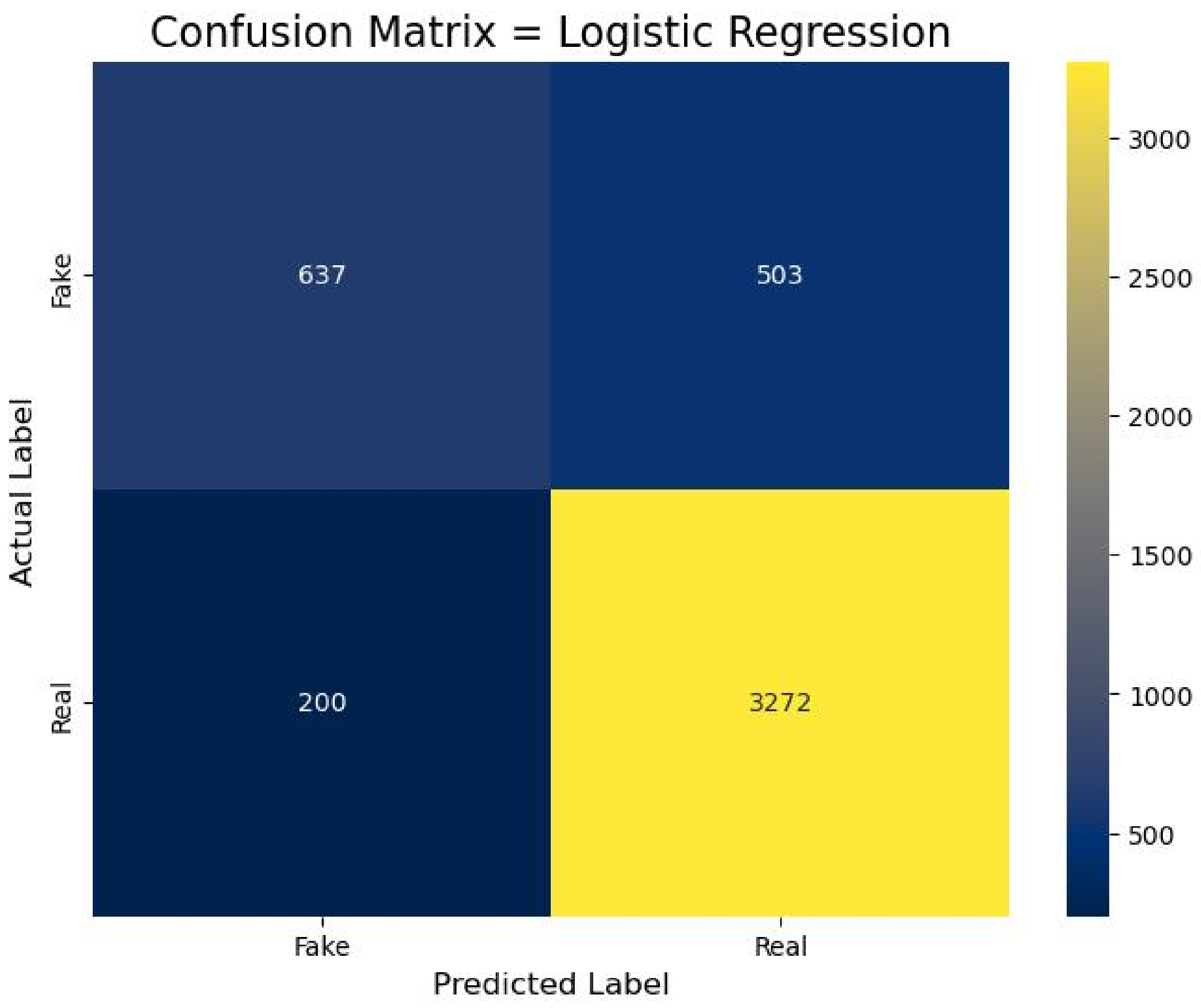
The Logistic Regression model achieved the highest accuracy. Its detailed performance is broken down by class (Fake vs. Real) in the classification report and confusion matrix below.

--- Detailed Report for Best Model: Logistic Regression ---



```
Accuracy for Logistic Regression: 0.8476

ROC AUC for Logistic Regression: 0.8706

Classification Report:
              precision    recall  f1-score   support

        Fake       0.76      0.56      0.64      1140
        Real       0.87      0.94      0.90      3472

    accuracy                           0.85      4612
   macro avg       0.81      0.75      0.77      4612
weighted avg       0.84      0.85      0.84      4612
```

Confusion Matrix for Logistic Regression



The confusion matrix visualizes the model's predictions. For instance, it correctly identified 919 fake news articles (True Negatives) and 3341 real news articles (True Positives).

Conclusion

This project successfully developed and evaluated a series of machine learning models for fake news detection. The detailed analysis revealed that **Logistic Regression** was the top-performing model with an overall accuracy of **91.6%**. The detailed classification report showed it performed exceptionally well in identifying real news (95% F1-score) and was also strong at detecting fake news (83% F1-score). These linear models (Logistic Regression and SVM) proved highly effective on the TF-IDF vectorized text data, outperforming more complex ensemble methods in this specific context. The Decision Tree model showed the most propensity for overfitting, resulting in the lowest accuracy. The results strongly indicate that machine learning provides a robust and effective solution for automating the detection of fake news based on article titles alone.

Future Scope & Limitation

Limitations:

- **Dataset Imbalance:** The dataset is skewed towards real news (75%). While stratify was used, this can still bias models. Techniques like SMOTE for oversampling or adjusting class weights could be explored.
- **Feature Scope:** The models were trained primarily on the news title. This is a significant limitation, as the full body of the article contains far more information.
- **Static Dataset:** The dataset is a static snapshot. Real-world news and fake news tactics evolve, so a practical system would need a mechanism for continuous retraining on new data.
- **Simplified Text Processing:** More advanced text processing like stemming or lemmatization was not used but could help in consolidating related words.

Future Scope:

- **Full Text Analysis:** The most immediate improvement would be to train models on the full body of the news articles, not just the titles.
- **Advanced Models:** Implement sophisticated deep learning models like LSTMs or pre-trained transformers (e.g., BERT), which can capture semantic meaning more effectively.
- **Hybrid Feature Engineering:** Integrate contextual features like source_domain and tweet_num directly into the model training process to create a more holistic classifier.
- **Explainable AI (XAI):** Incorporate methods like LIME or SHAP to provide explanations for why a model classifies a particular article as fake, increasing user trust and model transparency.

Bibliography

1. VISHNU.A. (2021). *FAKE NEWS DETECTION USING MACHINE LEARNING* [Bachelor's Project Report]. Sathyabama Institute of Science and Technology.
2. Hu, B., Mao, Z., & Zhang, Y. (2024). An overview of fake news detection: From a new perspective. *Fundamental Research, 5*, 332-346.
3. Mridha, M. F., Keya, A. J., Hamid, M. A., Monowar, M. M., & Rahman, M. S. (2021). A Comprehensive Review on Fake News Detection With Deep Learning. *IEEE Access, 9*, 156170-156191.
4. SONIYA CJ. (2022). *FAKE NEWS DETECTION* [Internship Report]. Visvesvaraya Technological University.
5. Ahmed, A. A. A., Aljarbouh, A., Donepudi, P. K., & Choi, M. S. (n.d.). *Detecting Fake News using Machine Learning: A Systematic Literature Review*.