Project Title	Fake News Detection on Reddit Post
Skills take away From This Project	Text Preprocessing, Feature Engineering, Vectorization Techniques, Model Building, Hyperparameter Tuning
Domain	Social Media Analytics

Background:

In the age of digital communication, **social media has become a primary source of news** and information. Platforms like Reddit allow users to post, share, and comment on content instantly, making them powerful tools for both information sharing and opinion shaping.

However, with this power comes the problem of **misinformation and fake news**. False or misleading information spreads quickly, influencing public perception and leading to real-world consequences such as:

- Public panic or confusion
- Political manipulation
- Health-related misinformation
- Financial scams and rumors

Detecting fake news manually is challenging due to the **volume of content**, the **short nature of post titles**, and the **lack of context**. Hence, there's a growing need for **automated systems** that can detect potential fake content in real time.

This project addresses that need by building an AI-powered fake news detection model, focusing specifically on **Reddit post titles**, which are often short, informal, and context-light. The model aims to **identify linguistic patterns and deception cues**, helping platforms and users **flag suspicious content** before it goes viral.

Problem Statement:

Social media platforms like Reddit are widely used for sharing news, opinions, and trending content. However, the **rapid spread of fake news** on these platforms has become a serious concern, leading to misinformation in critical areas like **public health, politics, and finance**.

The challenge lies in **detecting misleading or fake posts** from short, often informal, post titles. These titles may contain **slang, sarcasm, or ambiguous phrases**, making it difficult to classify them accurately without context.

The objective of this project is to develop a **machine learning model** that can analyze **Reddit post titles** and predict whether the content is **fake (1)** or **genuine (0)**. This model will help in **automating fake news detection**, improving platform trust, and supporting fact-checking systems.

The project includes:

- Preprocessing and cleaning short-text data
- Creating text-based features using vectorization
- Building ML models with evaluation using ROC-AUC
- Documenting multiple trials and performance improvements

Business Use Cases

Use Case 1: Reddit's Content Moderation System

Reddit can integrate the fake news detection model into its backend to **automatically score and flag suspicious post titles**. This helps moderators prioritize which posts to review manually, especially in fast-growing or sensitive subreddits (e.g., r/news, r/politics, r/worldnews).

Use Case 2: Real-time User Warnings

When a user attempts to post a title that the model predicts as likely fake, Reddit can **display a warning message** suggesting the user verify their information or cite sources. This nudges users towards responsible posting without blocking freedom of speech.

Use Case 3: Insights for Subreddit Moderators

Subreddit moderators can receive **periodic analytics** on the number and nature of fake vs real posts detected, helping them identify misinformation trends within their communities and adjust subreddit rules or filters accordingly.

Use Case 4: Platform-wide Trend Monitoring

Reddit's data team can use model predictions to **track the spread of fake news trends across subreddits**. This can be valuable for identifying coordinated misinformation campaigns or viral hoaxes early.

Use Case 5: Third-Party Research & API Access

Reddit can provide API access to this model's output (fake/genuine scores) to **approved researchers and fact-checkers**, supporting studies on misinformation dynamics, user behavior, and the impact of fake news interventions.

Approach

To detect fake news from Reddit post titles, this project follows a structured and step-by-step machine learning pipeline:

1. Data Understanding & Exploration

- Load the dataset and understand the distribution of fake (1) vs real (0) labels.
- Perform Exploratory Data Analysis (EDA) to check for:
    - Title length
    - Common words in fake vs real posts
    - Engagement vs label distribution (if provided)

2. Data Preprocessing

- Convert text to lowercase, remove special characters and punctuation.
- Tokenization: Split text into words.
- Remove stop words (e.g., "the", "is", "and").
- Apply stemming or lemmatization to reduce words to root form.
- Handle short text challenges such as sarcasm or vague phrases.

3. Feature Engineering

- Generate features using:
    - **TF-IDF Vectorizer**
    - **Countvectorizer**
    - **Character n-grams and Word n-grams**
- Create custom features like:
    - Title length
    - Number of exclamation marks
    - Use of all-caps words (often used in fake news)

4. Model Building

- Build and evaluate multiple models:

    - **Baseline**: Logistic Regression
    - **Advanced**: Random Forest, XGBoost
    - **Optional**: Deep learning model (LSTM or BERT for short text)
- Use pipelines to automate preprocessing and model training steps.

5. Hyperparameter Tuning

- Use Grid Search or Random Search to improve model performance.
- Tune parameters like:
    - Max depth, n-estimators for tree models
    - C value for Logistic Regression

6. Evaluation

- Evaluate using **ROC-AUC score** as the main metric.
- Also check accuracy, precision, recall, and confusion matrix.
- Use cross-validation to ensure the model generalizes well.

7. Experiment Tracking

- Conduct at least 5 model trials.
- Document each experiment's parameters, results, and next steps.
- Compare models based on performance and interpretability.

8. Final Prediction & Submission

- Generate final predictions as probability scores (0 to 1).

Project Evaluation Metrics

- **ROC-AUC Score** – Main metric to evaluate how well the model separates fake and real news.
- **Accuracy** – Measures the overall correctness of the model.
- **Precision** – Checks how many predicted fake posts are actually fake.
- **Recall** – Checks how many real fake posts were correctly identified.
- **F1-Score** – Balance between precision and recall.
- **Confusion Matrix** – Visual view of correct and incorrect predictions.
- **Cross-Validation** – Ensures the model performs well on different data splits.

Results

- Successfully developed a machine learning model that predicts whether a Reddit post title is fake or genuine.
- Demonstrated that short-text classification is effective using techniques like TF-IDF and models such as XGBoost.
- Gained insights into the language patterns and features commonly found in fake news content.
- Showed that AI can assist in automating fake news detection, helping platforms like Reddit in content moderation.
- Validated that even limited information, like post titles, can be used to detect misinformation with good accuracy.
- The project highlights the practical use of NLP and machine learning in addressing real-world problems related to misinformation.

Technical Tags:

Natural Language Processing (NLP), Text Classification, Fake News Detection, TF-IDF Vectorization, Feature Engineering, XGBoost, Logistic Regression, ROC-AUC Score, Cross-Validation, Python (Scikit-learn, Pandas, NumPy)

Data Set:

Data Set Link: Reddit

Data Set Explanation

The dataset contains Reddit post titles labeled as either **fake news (1)** or **genuine (0)**. Each entry represents a short text headline commonly found on Reddit, covering various topics and writing styles.

Key Components:

- **ID**: A unique identifier for each Reddit post title.
- **Title**: The main text content to be analyzed (short-form, informal, user-generated).
- **Label**: The target variable:
    - 1 → Fake news
    - 0 → Genuine news

Characteristics:

- Text data is **short and unstructured**, requiring advanced NLP techniques.
- No additional metadata is provided, so **all insights are derived from the title text**.
- The dataset is balanced (or may be slightly imbalanced), which needs to be considered during model training.

Project Deliverables

Source Code:

- Python scripts for text preprocessing, feature engineering, and model building.
- Jupyter notebooks showcasing the entire workflow from data exploration to evaluation.
- Scripts for hyperparameter tuning and prediction generation.

ML Model Outputs:

- Trained machine learning models (Logistic Regression, XGBoost, etc.).
- Saved model files using joblib or pickle for future use.
- Final prediction CSV file with ID and predicted probabilities (label).

Evaluation Reports:

- ROC-AUC scores, confusion matrix, and performance metrics for each model.
- Visualizations comparing model accuracy, precision, recall, and F1-score.
- Summary of cross-validation results and experiment tracking.

Dashboard (Optional for Extension):

- A lightweight dashboard (e.g., Streamlit or Power BI) to input Reddit titles and see prediction results in real time.

Documentation:

- Detailed report including:
    - Problem statement and objective
    - Dataset explanation
    - Approach and methodology
    - Results and key findings
    - Challenges faced and how they were solved
- Answers to required theoretical questions (n-grams, tokenization, etc.)

Project Guidelines

1. Text Preprocessing

- Apply best practices such as lowercasing, punctuation removal, stop word removal, tokenization, and stemming/lemmatization.
- Ensure the text is cleaned and ready for vectorization.

2. Feature Extraction

- Use appropriate vectorization methods such as TF-IDF or CountVectorizer.
- Experiment with word-level and character-level n-grams to improve model understanding.

3. Model Development

- Build baseline models like Logistic Regression for quick benchmarking.
- Train advanced models such as Random Forest and XGBoost for improved performance.
- Optionally explore deep learning methods (e.g., LSTM or BERT) for enhanced results.

4. Evaluation and Validation

- Evaluate models using ROC-AUC as the primary metric.
- Support evaluation with accuracy, precision, recall, and F1-score.
- Use cross-validation to ensure model consistency and avoid overfitting.

5. Experimentation

- Conduct and document at least five different model experiments.
- Log changes in preprocessing, feature selection, model choice, and tuning.
- Justify each decision and compare results to determine the best-performing model.

6. Submission Format

- Generate predictions as a CSV file with two columns: ID and label (predicted probability between 0 and 1).

7. Code and Documentation

- Maintain a clean and modular Python codebase or Jupyter Notebook.
- Clearly comment and explain each step in the pipeline.
- Include a detailed report covering methodology, results, and challenges faced.

Approval Workflow:

Created By:	Verified By:	Approved By:
Santhosh N		