Sarcasm Detection System - Project Summary

Overview:

This project builds a machine learning-based sarcasm detection system using a labeled dataset of tweets. The main objective is to classify whether a tweet is sarcastic or not.

Key Steps:

1. Data Loading & Initial Exploration

- Loaded a sarcasm dataset containing tweets and binary labels (sarcastic or not).
- Checked and removed rows with missing values.

2. Data Cleaning & Preprocessing

- Applied advanced text cleaning including:
  - Expanding contractions (e.g., "can't" -> "cannot")
  - Lowercasing
  - Removing emails, URLs, HTML tags, emojis, and punctuation
  - Tokenization
  - Stopwords removal (including custom stopwords/slangs)
  - Lemmatization to reduce words to their base forms

3. Data Balancing

- Handled class imbalance using Random Oversampling to equalize sarcastic and non-sarcastic tweet counts.

4. Feature Extraction

- Converted cleaned tweets into numerical features using TF-IDF vectorization.

5. Train-Test Split & Scaling

- Split the dataset into training (80%) and testing (20%) sets.
- Applied standard scaling suitable for sparse matrices.

6. Model Training & Evaluation

- Trained multiple classifiers including:
  - Support Vector Machine (SVM)
  - Random Forest Classifier
  - Logistic Regression
  - Gradient Boosting Classifier
  - Multinomial Naive Bayes
- Evaluated models using accuracy, precision, recall, F1-score, and confusion matrices.
- Random Forest and SVM achieved the best accuracies (~91%), with Random Forest slightly better.

7. Sarcasm Detection Function

- Created a reusable function `detect_sarcasm` that takes a new sentence, applies the same cleaning and vectorization pipeline, and predicts sarcasm using the trained Random Forest model.

Results:

- Best Model: Random Forest Classifier
- Accuracy: ~91% on test data
- Confusion Matrix: Balanced true positives and true negatives, showing robust sarcasm detection.

Technologies and Libraries:

- Python
- Pandas, Numpy
- NLTK (text preprocessing)
- scikit-learn (TF-IDF, classifiers, evaluation)
- imbalanced-learn (RandomOverSampler)
- contractions (for expanding contractions)

Usage:

- Load the trained model and vectorizer
- Use the `detect_sarcasm` function to classify new text input

Sarcasm Detection System - Project Summary

Future Work:

You can easily extend this project by experimenting with other advanced NLP techniques such as word embeddings, deep learning models, or transformer-based classifiers.