BUILDING A SMARTER AI-POWERED SPAM CLASSIFIER

TEAM MEMBER

510521104306: LAVANYA J

PHASE-1: DOCUMENT SUBMISSION

OBJECTIVES:

The problem is to build an AI-powered spam classifier that can accurately distinguish between spam and non-spam messages in emails or text messages. The goal is to reduce the number of false positives (classifying legitimate messages as spam) and false negatives (missing actual spam messages) while achieving a high level of accuracy.

PHASE-1: Problem Definition and Design Thinking

In this part you will need to understand the problem statement and create a document on what have you understood and how will you proceed ahead with solving the problem. Please think on a design and present in form of a document.

DATASET LINK:

https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset

	Go until jurong point, crazy.. Available only in bugis n great world la e buffet...
ham	Cine there got amore wat...
ham	Ok lar... Joking wif u oni...
spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's
ham	U dun say so early hor... U c already then say...
ham	Nah I don't think he goes to usf, he lives around here though
spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, å£1.50 to rcv
ham	Even my brother is not like to speak with me. They treat me like aids patent.
ham	As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. Press *9 to copy your friends Callertune
spam	WINNER!! As a valued network customer you have been selected to receivea å£900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only.
spam	Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030
ham	I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.
spam	SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to 87575. Cost 150p/day, 6days, 16+ TsandCs apply Reply HL 4 info
spam	URGENT! You have won a 1 week FREE membership in our å£100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net LCCLTD POBOX 4403LDNW1A7RW18
ham	I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promise. You have been wonderful and a blessing at all times.
ham	I HAVE A DATE ON SUNDAY WITH WILL!!
spam	XXXMobileMovieClub: To use your credit, click the WAP link in the next txt message or click here>> http://wap. xxxmobilemovieclub.com?n=QJKGIGHJJGCBL
ham	Oh k...i'm watching here:)
ham	Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.
ham	Fine if thatåÕs the way u feel. ThatåÕs the way its gota b
spam	England v Macedonia - dont miss the goals/team news. Txt ur national team to 87077 eg ENGLAND to 87077 Try:WALES, SCOTLAND 4txt/Ì¼1.20 POBOXox36504W45WQ 16+
ham	Is that seriously how you spell his name?

ham	I‰Û÷m going to try for 2 months ha ha only joking
ham	So Ì_ pay first lar... Then when is da stock comin...
ham	Aft i finish my lunch then i go str down lor. Ard 3 smth lor. U finish ur lunch already?
ham	Ffffffffff. Alright no way I can meet up with you sooner?
ham	Just forced myself to eat a slice. I'm really not hungry tho. This sucks. Mark is getting worried. He knows I'm sick when I turn down pizza. Lol
ham	Lol your always so convincing.
ham	Did you catch the bus ? Are you frying an egg ? Did you make a tea? Are you eating your mom's left over dinner ? Do you feel my Love ?
ham	I'm back & we're packing the car now, I'll let you know if there's room
ham	Ahhh. Work. I vaguely remember that! What does it feel like? Lol
ham	Wait that's still not all that clear, were you not sure about me being sarcastic or that that's why x doesn't want to live with us
ham	Yeah he got in at 2 and was v apologetic. n had fallen out and she was actin like spoilt child and he got caught up in that. Till 2! But we won't go there! Not doing too badly cheers. You?
ham	K tell me anything about you.
ham	For fear of fainting with the of all that housework you just did? Quick have a cuppa
spam	Thanks for your subscription to Ringtone UK your mobile will be charged å£5/month Please confirm by replying YES or NO. If you reply NO you will not be charged
ham	Yup... Ok i go home look at the timings then i msg Ì_ again... Xuhui going to learn on 2nd may too but her lesson is at 8am
ham	Oops, I'll let you know when my roommate's done
ham	I see the letter B on my car
ham	Anything lor... U decide...
ham	Hello! How's you and how did saturday go? I was just texting to see if you'd decided to do anything tomo. Not that i'm trying to invite myself or anything!
ham	Pls go ahead with watts. I just wanted to be sure. Do have a great weekend. Abiola
ham	Did I forget to tell you ? I want you , I need you, I crave you ... But most of all ... I love you my sweet Arabian steed ... Mmmmmm ... Yummy

DATA PROCESSING:

Data preprocessing is a crucial step in natural language processing (NLP) and text analysis tasks. It helps in cleaning and transforming raw text data into a format that is suitable for analysis and machine learning models. Here are the steps you mentioned for text data preprocessing:

1. **Removing Special Characters**: Special characters like punctuation marks, symbols, and numbers may not be relevant for many text analysis tasks. Removing them can make the text data cleaner and reduce noise. You can use regular expressions or simple string operations to remove special characters. Here's a Python example using regular expressions:

   python

```
import re

def remove_special_characters(text):
    # Remove non-alphanumeric characters and whitespace
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    return text
```

- **Converting Text to Lowercase**: Consistency in letter casing (e.g., lowercase) is important to ensure that words are treated the same regardless of their original case. This step helps in reducing the dimensionality of the data by merging words with different cases. In Python, you can convert text to lowercase using the lower() method:

python

```
def convert_to_lowercase(text):
    return text.lower()
```

 Tokenization: Tokenization is the process of splitting text into individual words or tokens. It is a fundamental step for most NLP tasks. You can use various libraries and tools for tokenization. In Python, the Natural Language Toolkit (NLTK) and the spaCy library are popular choices. Here's an example using NLTK:

python

```
3. import nltk
4. from nltk.tokenize import word_tokenize
5.
6. def tokenize_text(text):
7.     # Tokenize the text into words
8.     tokens = word_tokenize(text)
9.     return tokens
10.
```

After applying these preprocessing steps, you'll have text data that is cleaned, converted to lowercase, and tokenized into individual words. This processed data can then be used for tasks like text classification, sentiment analysis, information retrieval, and more.

Remember that the specific preprocessing steps you need may vary depending on your NLP task and the characteristics of your text data. Additionally, you may want to consider other preprocessing steps such as stop word removal, stemming, or lemmatization based on the requirements of your project.

FEATURE EXTRACTION:

Feature extraction is a crucial step in text analysis and natural language processing (NLP). It involves converting text data into numerical features that machine learning models can understand. One common technique for feature extraction is TF-IDF (Term Frequency-Inverse Document Frequency). Here's how you can use TF-IDF for feature extraction:

1. **TF-IDF Introduction**:
    - **Term Frequency (TF)**: This measures the frequency of a term (word) in a document. It's calculated as the number of times a term appears in a document divided by the total number of terms in that document.
    - **Inverse Document Frequency (IDF)**: This measures the importance of a term in a collection of documents. It's calculated as the logarithm of the total number of documents divided by the number of documents containing the term.
2. **TF-IDF Calculation**:
    - Calculate the TF for each term in each document.
    - Calculate the IDF for each term in the entire corpus (collection of documents).
    - Multiply TF and IDF to get the TF-IDF score for each term in each document.

Here's how to perform TF-IDF feature extraction in Python using the TfidfVectorizer from scikit-learn, assuming you have a list of tokenized documents:

python
```
from sklearn.feature_extraction.text import TfidfVectorizer

# Example tokenized documents
```

```
tokenized_documents = [
    ["this", "is", "document", "1"],
    ["this", "is", "document", "2"],
    ["another", "document", "is", "here"],
]

# Create a TF-IDF vectorizer
tfidf_vectorizer = TfidfVectorizer()

# Fit and transform the tokenized documents
tfidf_matrix = tfidf_vectorizer.fit_transform([' '.join(doc) for
doc in tokenized_documents])

# Get the TF-IDF features
tfidf_features = tfidf_matrix.toarray()

# The tfidf_features array contains TF-IDF values for each
term in each document
print(tfidf_features)
```

The tfidf_features array will contain the numerical features for your text data, where each row corresponds to a document, and each column corresponds to a unique term in the entire corpus.

These TF-IDF features can then be used as input to machine learning models for various NLP tasks, such as text classification, clustering, or information retrieval, where numerical data is required.

MODEL SELECTION:

Model selection is a critical step in the process of building a machine learning or deep learning-based text analysis system. The choice of the algorithm depends on the nature of your task, the size of your dataset, and the computational resources available. Here are some common machine learning algorithms, including traditional and deep learning approaches, that you can consider for text analysis:

1. **Naive Bayes**:
    - **Naive Bayes** algorithms are simple and often used as a baseline model for text classification tasks. They work well for tasks like spam detection and sentiment analysis.
2. **Support Vector Machines (SVM)**:
    - **SVMs** are effective for both binary and multi-class classification tasks. They can handle high-dimensional feature spaces, making them suitable for text classification and sentiment analysis.
3. **Logistic Regression**:
    - **Logistic Regression** is another straightforward algorithm often used for text classification, particularly when the number of features (words) is high.
4. **Random Forest** and **Gradient Boosting**:
    - **Random Forest** and **Gradient Boosting** are ensemble learning methods that can be effective for text classification tasks. They can handle complex relationships in the data.
5. **Deep Learning with Neural Networks**:
    - **Recurrent Neural Networks (RNNs)**: RNNs are suitable for sequential data like text. They are used

for tasks like sequence tagging, text generation, and sentiment analysis.

- **Convolutional Neural Networks (CNNs)**: CNNs can be used for text classification tasks, treating text as an image with one dimension.
- **Transformers**: Transformer-based models like **BERT**, **GPT-3**, and **XLNet** have achieved state-of-the-art results in various NLP tasks, including sentiment analysis, named entity recognition, and language generation.
-

6. **Word Embeddings**:
    - **Word2Vec**, **GloVe**, and other word embedding techniques can be used to create dense vector representations of words. These embeddings can be used as features for traditional machine learning

EVALUTAION:

Evaluating the performance of your text analysis model is essential to assess how well it's doing on the task at hand. The choice of evaluation metrics depends on the specific NLP task you're working on. Here are some common evaluation metrics for text classification tasks:

1. **Accuracy**:
    - **Accuracy** measures the proportion of correctly classified instances among all instances. It is a useful metric for balanced datasets but can be misleading for imbalanced datasets.
2. **Precision**:
    - **Precision** is the ratio of true positive predictions to the total number of positive predictions. It measures how many of the predicted positive instances were actually correct. Precision is important when false positives are costly.
3. **Recall** (Sensitivity or True Positive Rate):
    - **Recall** is the ratio of true positive predictions to the total number of actual positive instances. It measures the model's ability to correctly identify all positive instances. Recall is important when false negatives are costly.
4. **F1-Score**:
    - **F1-score** is the harmonic mean of precision and recall. It balances precision and recall and is especially useful when there is an imbalance between classes in the dataset.
5. **Specificity** (True Negative Rate):
    - **Specificity** measures the model's ability to correctly identify all negative instances. It is the ratio of true

negative predictions to the total number of actual negative instances.

6. **Area Under the Receiver Operating Characteristic Curve (AUC-ROC)**:
    - AUC-ROC measures the area under the ROC curve, which plots the true positive rate against the false positive rate at various thresholds. It is useful for binary classification tasks and can help assess the model's ability to distinguish between classes.
7. **Confusion Matrix**:
    - A **confusion matrix** is a tabular representation of the model's performance. It shows the number of true positives, true negatives, false positives, and false negatives, allowing for a detailed analysis of model errors.
8. **Mean Absolute Error (MAE)** and **Mean Squared Error (MSE)**:
    - For regression tasks, MAE and MSE can be used to measure the difference between predicted and actual numerical values.

When evaluating your text analysis model, it's essential to consider the specific requirements and constraints of your application. For example, if you are working on sentiment analysis, accuracy, precision, recall, and F1-score are suitable metrics. However, for information retrieval tasks, metrics like precision at k (P@k) and mean average precision (MAP) may be more relevant.

Additionally, it's a good practice to use techniques like cross-validation to assess the model's generalization performance and avoid overfitting. Make sure to select the most appropriate metrics based on your task's goals and interpretability requirements.

Iterative improvement:

Iterative improvement is a key component of the machine learning and model development process. It involves a cycle of experimentation, evaluation, and adjustment to enhance the performance of your text analysis model. Here's a step-by-step guide on how to approach iterative improvement:

1. **Baseline Model**:
    - Start by building a baseline model using a selected algorithm and hyperparameters. This serves as a starting point for further improvement.
2. **Data Splitting**:
    - Split your dataset into training, validation, and test sets. The training set is used to train the model, the validation set helps tune hyperparameters, and the test set is used for the final evaluation.
3. **Hyperparameter Tuning**:
    - Experiment with different hyperparameters for your chosen algorithm. This can include learning rates, regularization parameters, model architecture choices, and more.
    - Utilize techniques like grid search or random search to systematically explore the hyperparameter space.
    - Use the validation set to assess the model's performance with different hyperparameter configurations.
4. **Regularization**:
    - Apply regularization techniques such as L1 or L2 regularization to prevent overfitting. These techniques can help improve model generalization.
5. **Feature Engineering**:
    - Consider additional feature engineering steps based on domain knowledge. This might involve creating

new features or modifying existing ones to better represent the problem.

6. **Ensemble Methods**:
    - Explore ensemble methods like bagging and boosting to combine multiple models for improved performance. Techniques like Random Forest or Gradient Boosting can be effective.
7. **Cross-Validation**:
    - Utilize cross-validation techniques like k-fold cross-validation to get a better estimate of your model's performance and to reduce the impact of dataset variability.
8. **Monitoring Metrics**:
    - Continuously monitor the evaluation metrics on the validation set as you make changes to the model and hyperparameters. This helps you track progress.

9. **Early Stopping**:
    - Implement early stopping to prevent overfitting. It involves stopping training when the model's performance on the validation set starts to degrade.

10. **Iterate**:
    - Iterate through steps 3 to 9 multiple times, adjusting hyperparameters and model architecture as needed. Keep track of your experiments and their outcomes.

11. **Final Evaluation**:

- Once you're satisfied with the model's performance on the validation set, evaluate it on the test set to get an unbiased estimate of its generalization performance.

12. **Documentation**:
    - Keep detailed records of your experiments, including the hyperparameters, changes made, and their effects on model performance. This documentation can be valuable for future reference.

13. **Deployment and Monitoring**:
    - If your model is intended for deployment, ensure that it continues to perform well in a production environment. Implement monitoring to detect any degradation in performance over time.

Remember that the iterative improvement process can be time-consuming, but it's essential for achieving the best possible results in your text analysis task. Additionally, keep in mind that there may be trade-offs between different evaluation metrics, and the choice of the best model may depend on the specific goals of your project.