Building a smarter AI powered spam classifier

Phase-4

Loading and preprocessing a dataset for building a smarter AI-powered spam classifier involves several steps. Here's a high-level overview of the process:

Data Collection: Gather a labeled dataset that contains examples of both spam and non-spam (ham) messages. This dataset should be diverse and representative of the messages your classifier will encounter.

Data Cleaning: Remove any irrelevant or duplicate data, as well as any outliers. Ensure that your dataset is well-structured and consistent.

INTRODCTION:

- Text Preprocessing:
- Text Tokenization: Split the text into individual words or tokens.
- Lowercasing: Convert all text to lowercase to ensure consistency.
- Removing Punctuation: Eliminate punctuation marks that don't carry significant meaning.
- Stopword Removal: Exclude common words (e.g., "and," "the," "in") that are unlikely to help classify spam.
- Stemming or Lemmatization: Reduce words to their base or root form to handle variations (e.g., "running" to "run").
- Word Embeddings: Use pre-trained word vectors like Word2Vec or GloVe to capture semantic meaning.Split the Dataset into training, validation, and test sets to evaluate your model's performance.

Certainly! Selecting a machine learning algorithm, training the model, and evaluating its performance are crucial steps in building a machine learning model.

- 1. *Selecting a Machine Learning Algorithm*:
- - Start by understanding your problem and data. Is it a classification, regression, or clustering problem?
- - Consider the nature of your data: Is it structured or unstructured? How many features do you have?
- - Choose algorithms based on the problem type (e.g., decision trees for classification, linear regression for regression).

- Experiment with different algorithms and fine-tune hyperparameters to see which one works best.

- 2. *Training the Model*:
- - Preprocess the data: Handle missing values, scale or normalize features, encode categorical variables.
- - Split your data into training and testing sets to assess model generalization.
- - Train the model using the training data, feeding features and their corresponding labels.
- - Monitor the training process, adjusting hyperparameters if necessary.

- 3. *Evaluating Performance*:
- - Use evaluation metrics specific to your problem (e.g., accuracy, precision, recall, F1-score for classification; RMSE, MAE for regression).
- - Assess the model's performance on the testing dataset to ensure it generalizes well to new, unseen data.
- - Consider using cross-validation to get a more robust estimate of your model's performance.
- - Analyze any overfitting or underfitting issues and make necessary adjustments.
- Remember ,model selection,training,and evaluation are iterative processes.you may need to go back and make changes based on the your evaluation results until you achieve the desired model performance.

- GIVEN DATASETS:
- ham
- Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got a...
- 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 entr...
- 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...

- 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 s...
- 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 callertu...

- PROGRAM:

```
# necessary libraries
import openai
import pandas as pd
import numpy as np


# libraries to develop and evaluate a machine learning model
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score
from sklearn.ensemble import RandomForestClassifier
```

```
from sklearn.model_selection import train_test_split

from sklearn.metrics import classification_report, accuracy_score

from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import train_test_split

from sklearn.metrics import classification_report, accuracy_score

from sklearn.metrics import confusion_matrix
```

```
# replace "YOUR API KEY" with your generated API key
openai.api_key = "YOUR API KEY"

# while loading the csv, we ignore any encoding errors and skip any bad line
df = pd.read_csv('spam.csv', encoding_errors='ignore', on_bad_lines='skip')
print(df.shape)

# we have 3 columns with NULL values, to remove that we use the below line
df = df.dropna(axis=1)

# we are taking only the first 60 rows for developing the model
df =df.loc[:60]
```

```
# rename the columns v1 and v2 to
Output and Text respectively

df.rename(columns = {'v1':'OUTPUT',
'v2': 'TEXT'}, inplace = True)

print(df.shape)

df.head()
```

OUTPUT:

```
(5572, 5)
(60, 2)
```

	OUTPUT	TEXT
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

- CONCLUSION:

Thus, Building a smarter AI-powered spam classifier involves several steps, including data preprocessing, model development, and deployment.