AI-POWERED SPAM CLASSIFIER

TEAM MEMBER

R.ANITHA-950921104001

Project Title: AI-POWERED SPAM CLASSIFIER

Phase 3 : Development Part 1

TOPIC: AI- Powered Spam Classifier by Loading and Preprocessing the dataset.

INTRODUCTION:

★An AI-powered spam classifier is a machine learning model that is trained to identify and classify spam messages. Spam messages are unsolicited messages that are often used to spread malware, phishing attacks, or other malicious content. AI spam classifiers can help to reduce spam by filtering out spam messages before they reach the user's inbox.

★AI spam classifiers are less likely to block legitimate messages as spam. This is because AI spam classifiers use a variety of features to classify messages, rather than just relying on a list of blacklisted words or phrases. This helps to reduce the number of false positives, which are legitimate messages that are incorrectly classified as spam.

★This introduction will guide you through the initial steps of the process. We'll explore how to import essential libraries, load the Spam dataset, and perform critical preprocessing steps. Data preprocessing is crucial as it helps clean, format, and prepare the data for further analysis. This includes handling missing values, encoding categorical variables, and ensuring that the data is appropriately scaled.

Given Dataset

V1	V2	Unnamed: 2	Unnamed: 3	Unnamed: 4	
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN

V1	V2	Unnamed: 2	Unnamed: 3	Unnamed: 4	
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

Necessary Step to follow:

1.Import Libraries:

Start by importing Libraries

Program:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split from
sklearn.preprocessing import StandardScaler
```

2.Load Dataset:

Load your dataset into a Pandas DataFrame. You can typically Email Spam Dataset in CSV format, but you can adapt this code to other formats as needed.

df = pd.read_csv ('/kaggle/input/sms-spam-collection-dataset/spam.csv', encoding=' SO-8859-1')

3.Exploratory Data Analysis (EDA):

Perform EDA to understand your data better. This includes checking for missing values, exploring the data's statistics, and visualizing it to identify patterns.

Checking the Missing Values

print(df.isnull().sum())

Program:

Pd.read()

Program:

Explore statistics

print(df.describe())

Visualize the data (e.g., histograms, scatter plots, etc.)

4.Feature Engineering:

Feature engineering is the process of creating new features from existing features in order to improve the performance of a machine learning model. In the context of AI spam classifiers, feature engineering can be used to create features that are more informative and discriminative for the task of spam classification.

PROGRAM

```
# Create a CountVectorizer for text feature extraction

featurizer = CountVectorizer(decode_error='ignore')
x_train = featurizer.fit_transform(df_train)
x_test = featurizer.transform(df_test)

#Check the result of feature extraction
x_train
```

OUTPUT:

```
<4457x7735 sparse matrix of type '<class 'numpy.int64'>'
        with 58978 stored elements in Compressed Sparse Row format>
```

4.Split the Dataset:

Split your dataset into training and testing sets.

Program:

```
from sklearn.model_selection

import train_test_split features = df1.drop('v1' , axis = 1)
```

```
label = df1['v1']

x_train , x_test , y_train , y_test = train_test_split(features , label , test_size = 0.3)

print(f"X train shape : {x_train.shape}\nY train shape : {y_train.shape}\nX test shape : {x_test.shape}\nY test shape : {y_test.shape}")
```

Importance of loading and processing dataset:

Loading and preprocessing the dataset is an important first step in building any machine learning model. However, it is especially important for AI Spam Classifier models, as Spam datasets are often complex and noisy.

By loading and preprocessing the dataset, we can ensure that the machine learning algorithm is able to learn from the data effectively and accurately.

Challenges involved in loading and preprocessing a Spam Classification dataset;

There are a number of challenges involved in loading and preprocessing a house price dataset, including:

Tokenization:

This step involves splitting the text into individual words or tokens. This can be done using a variety of different tokenization techniques, such as regular expressions or rule-based tokenizers.

Stop word removal:

Stop words are common words that do not add much meaning to the text, such as articles, prepositions, and pronouns. Stop words are typically removed from the text before training the machine learning model.

Lowercasing:

This step involves converting all of the words in the text to lowercase. This is important because many machine learning algorithms are case-insensitive.

How to overcome the challenges of loading and preprocessing a spam classification dataset:

There are a number of things that can be done to overcome the challenges of loading and preprocessing a spam classification dataset, including:

- **Use a data preprocessing library:**

There are a number of libraries available that can help with data preprocessing tasks, such as handling missing values, encoding categorical variables, and scaling the features.

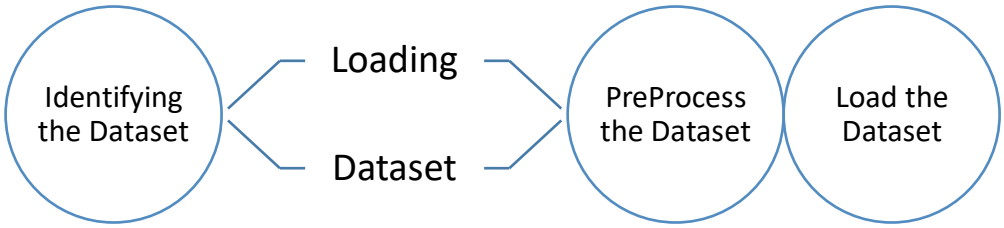
- **Carefully consider the specific needs of your model:**

The best way to preprocess the data will depend on the specific machine learning algorithm that you are using. It is important to carefully consider the requirements of the algorithm and to preprocess the data in a way that is compatible with the algorithm.

- **Validate the preprocessed data:**

It is important to validate the preprocessed data to ensure that it is in a format that can be used by the machine learning algorithm and that it is of high quality. This can be done by inspecting the data visually or by using statistical methods.

Loading Dataset:



Program:

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split from

sklearn.preprocessing import StandardScaler

df = pd.read_csv ('/kaggle/input/sms-spam-collection-dataset/spam.csv', encoding='
SO-8859-1')
```

Pd.read()

OUTPUT:

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

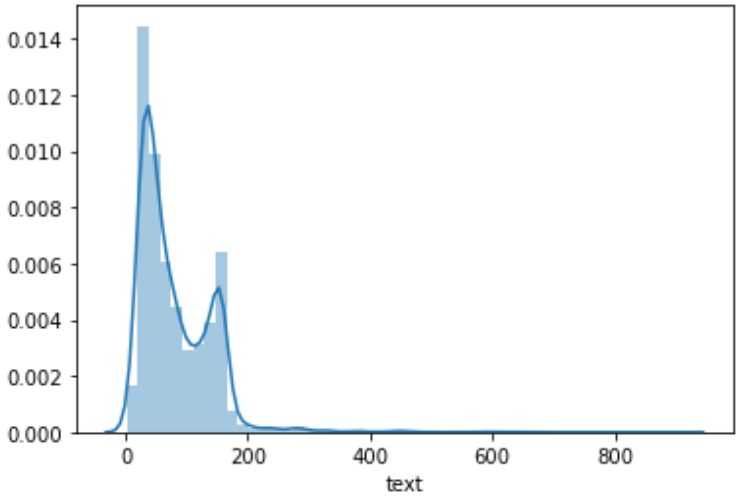
Preprocessing the dataset:

Data preprocessing is the process of cleaning, transforming, and integrating data in order to make it ready for analysis.

Program:

```
In[1] :   df.rename(columns={"v1": "target", "v2": "text"}, inplace=True)
          df['target'] = (df['target']== 'spam').astype(int)
In[2] :   sb.distplot(df.text.str.len())
```

Out [2] : <matplotlib.axes._subplots.AxesSubplot at 0x7f55aa751d10>



Data transformation:

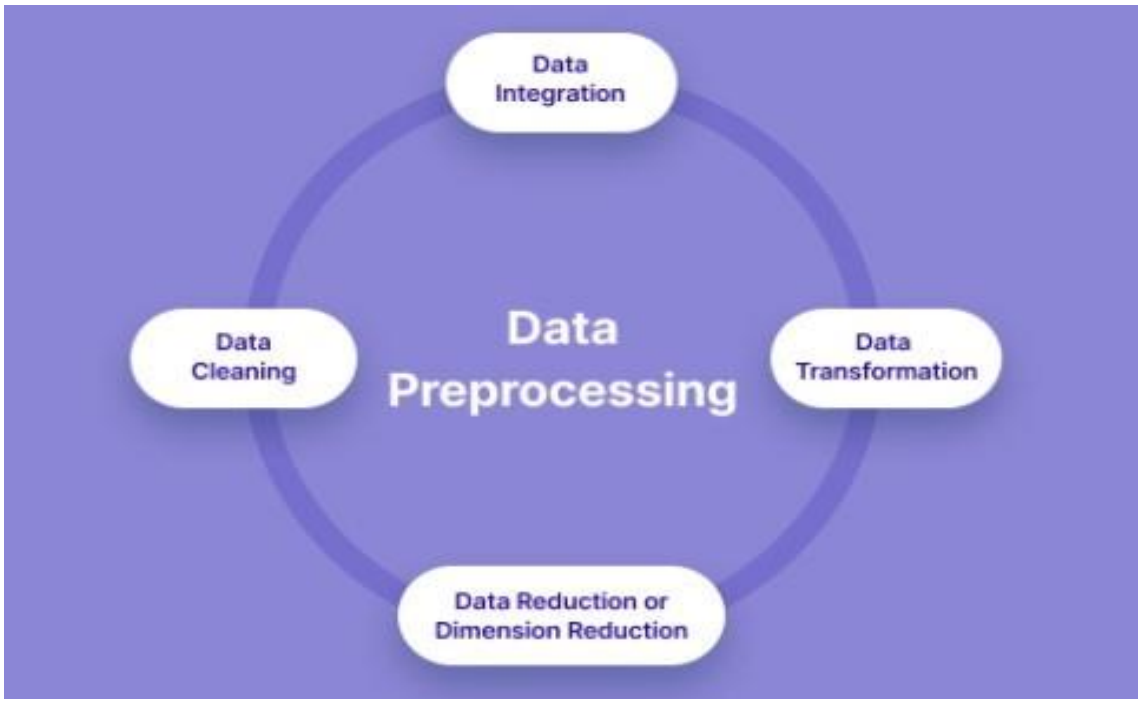
This involves converting the data into a format that is suitable for the analysis task. For example, this may involve converting categorical data to numerical data, or scaling the data to a suitable range.

Feature engineering:

This involves creating new features from the existing data. For example, this may involve creating features that represent interactions between variables, or features that represent summary statistics of the data.

Data integration:

This involves combining data from multiple sources into a single dataset. This may involve resolving inconsistencies in the data, such as different data formats or different variable names. Data preprocessing is an essential step in many data science projects.



PROGRAM

AI POWERED SPAM CLASSIFIER

```
# Import Libraries

import numpy as np
import pandas as pd

# Reading the dataset

df = pd.read_csv ('/kaggle/input/sms-spam-collection-dataset/spam.csv', encoding='
SO-8859-1')
```

```
df.head ()

# Separating X and y

X = df ['v2']
y = df['v1']
display(X, y)

# Encoding the Labels

from sklearn.preprocessing
import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
display(y)

# Load the Dataset

    csvfile = '../input/sms-spam-collection-dataset/spam.csv'
```

OUTPUT:

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

5567    This is the 2nd time we have tried 2 contact u...
5568                Will Ì_ b going to esplanade fr home?
5569    Pity, * was in mood for that. So...any other s...
5570    The guy did some bitching but I acted like i'd...
5571                         Rofl. Its true to its name

Name: v2, Length: 5572, dtype: object
0        ham
1        ham
2       spam
3        ham
4        ham
        ...

5567    spam
5568     ham
5569     ham
5570     ham
5571     ham
Name: v1, Length: 5572, dtype: object
```

Logistic Regression:

```
In[5] : import pandas as pd
        import numpy as np
        from sklearn.linear_model import LogisticRegression
        from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score
        from sklearn.metrics import confusion_matrix
In[5] : data = pd.read_csv('../input/sms-data-labelled-spam-and-non-
spam/SMSSpamCollection', sep='\t', names=['label', 'message'])
In[6] :    data.head()
Out[6] :    label          message
  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...
In[7] :   data.info()
In[8] :   data['label'] = data.label.map({'ham':0, 'spam':1})
          data.head()
Out[8] :      label   message
      0         0    Go until jurong point, crazy.. Available only ...
      1         0    Ok lar... Joking wif u oni...
      2         1    Free entry in 2 a wkly comp to win FA Cup fina...
      3         0    U dun say so early hor... U c already then say...
      4         0    Nah I don't think he goes to usf, he lives aro...
In[9] : from sklearn.model_selection import train_test_split
        X_train, X_test, y_train, y_test = train_test_split(data['message'],
                                          data['label'],
                                          test_size =0.2,
                                          random_state=1)
        print('Number of rows in the total set: {}'.format(data.shape[0]))
```

```
print('Number of rows in the training set: {}'.format(X_train.shape[0]))

print('Number of rows in the test set: {}'.format(X_test.shape[0]))
```

```
Out[9] : Number of rows in the total set: 5572
         Number of rows in the training set: 4457
         Number of rows in

the test set: 1115
```

Reducing false positives and false negatives

There are a variety of techniques that can be used to reduce false positives and false negatives, such as:

- Using a balanced dataset: A balanced dataset has an equal number of spam and non-spam messages. This helps to prevent the model from becoming biased towards one class or the other.
- Using a validation set: A validation set is a subset of the training data that is used to tune the parameters of the model. This helps to prevent overfitting, which can lead to poor performance on new data.
- Using ensemble methods: Ensemble methods combine the predictions of multiple models to produce a more accurate prediction. This can help to reduce both false positives and false negatives.

Achieving a high level of accuracy

To achieve a high level of accuracy, it is important to use a large and diverse dataset, a variety of features, and a well-tuned machine learning model. It is also important to monitor the performance of the model over time and update it with new data and features as needed.

Preprocessing Complete

Conclusion:

In the quest to build a AI spam Classification model, we have embarked on a critical journey that begins with loading and preprocessing the dataset.We have traversed through essential steps, starting with importing the necessary libraries to facilitate data manipulation and analysis.

Understanding the data's structure, characteristics, and any potential issues through exploratory data analysis (EDA) is essential for informed decision-making.

Data preprocessing emerged as a pivotal aspect of this process. It involves cleaning, transforming, and refining the dataset to ensure that it aligns with the requirements of machine learning algorithms.

With these foundational steps completed, our dataset is now primed for the subsequent stages of building and training a Spam Classification model.