BUILDING A SMARTER AI-POWERED SPAM CLASSIFIER

510521104003 :ANANDHI S

PHASE-4: 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-4: DEVELOPMENT PART-2:

In this part you will continue building your project.
In this phase, we'll continue building our spam classifier by:

- Selecting a machine learning algorithm
- Training the model
- Evaluating its performance.

DATASET LINK:

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

ABSTRACT:

Building a smarter AI-powered spam classifier is a compelling solution to address this issue. This abstract provides an overview of the development process and key elements involved in creating an intelligent and effective spam filter. The development journey begins with data collection, encompassing a diverse dataset of spam and non-spam (ham) messages. Model selection is a critical decision, with options ranging from traditional machine learning algorithms to advanced deep learning architectures.

INTRODUCTION:

A smarter AI-powered spam classifier leverages the capabilities of artificial intelligence, machine learning, and natural language processing to not only detect and filter spam but also to continually evolve and learn from new threats. It is an intelligent guardian that ensures that legitimate messages reach their intended recipients while relegating unwanted content to the digital wasteland. This development journey encompasses a series of crucial steps, each designed to enhance the classifier's efficacy. It begins with the collection of a diverse dataset containing examples of spam and legitimate messages.

Data preprocessing tasks prepare this data for model training, including text normalization and feature extraction. Model selection is a pivotal decision, where various machine learning algorithms or deep learning architectures are considered.

SELECTING AN MACHINE LEARNING ALGORITHM

List of Popular Machine Learning Algorithm

1. **Linear Regression Algorithm**
2. **Logistic Regression Algorithm**
3. **Decision Tree**
4. **SVM**
5. **Naïve Bayes**
6. **KNN**
7. **K-Means Clustering**

1. Linear Regression

Linear regression is one of the most popular and simple machine learning algorithms that is used for predictive analysis. Here, **predictive analysis** defines prediction of something, and linear regression makes predictions for *continuous numbers* such as **salary, age, etc.**

It shows the linear relationship between the dependent and independent variables, and shows how the dependent variable(y) changes according to the independent variable (x).

It tries to best fit a line between the dependent and independent variables, and this best fit line is knowns as the regression line.

The equation for the regression line is:

$y= a_0+ a*x+ b$

Here, y= dependent variable

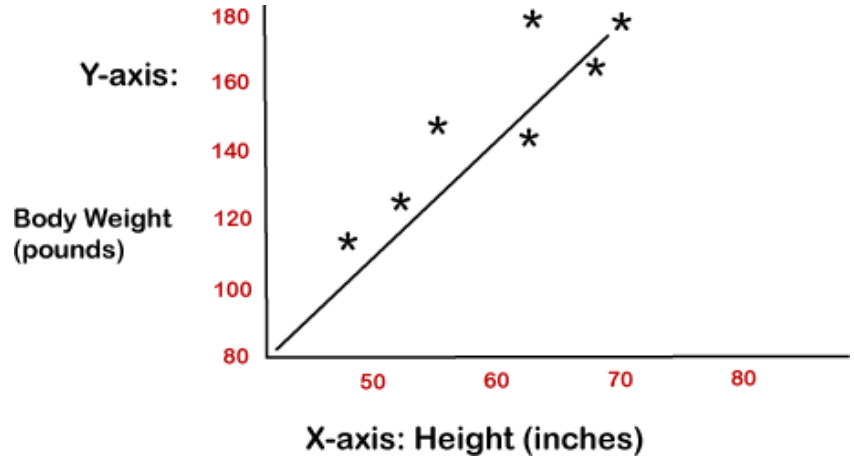
x= independent variable

a_0 = Intercept of line.

Linear regression is further divided into two types:

- **Simple Linear Regression:** In simple linear regression, a single independent variable is used to predict the value of the dependent variable.
- **Multiple Linear Regression:** In multiple linear regression, more than one independent variables are used to predict the value of the dependent variable.

The below diagram shows the linear regression for prediction of weight according to height:



2. Logistic Regression

Logistic regression is the supervised learning algorithm, which is used to **predict the categorical variables or discrete values**. It can be used for the *classification problems in machine learning*, and the output of the logistic regression algorithm can be either Yes or NO, 0 or 1, Red or Blue, etc.

Logistic regression is similar to the linear regression except how they are used, such as Linear regression is used to solve the regression problem and predict continuous values, whereas Logistic regression is used to solve the Classification problem and used to predict the discrete values.

Instead of fitting the best fit line, it forms an S-shaped curve that lies between 0 and 1. The S-shaped curve is also known as a logistic function that uses the concept of the threshold. Any value above the threshold will tend to 1, and below the threshold will tend to 0.

3. Decision Tree Algorithm

A decision tree is a supervised learning algorithm that is mainly used to solve the classification problems but can also be used for solving the regression problems. It can work with both categorical variables and continuous variables. It shows a tree-like structure that includes nodes and branches, and starts with the root node that expand on further branches till the leaf node. The **internal node** is used to represent the **features of the dataset, branches show the decision rules,** and **leaf nodes represent the outcome of the problem.**

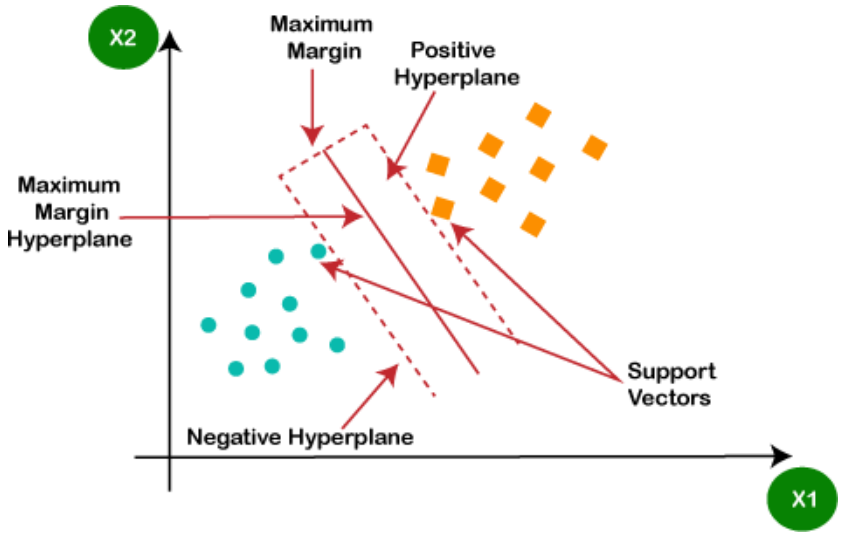
Some real-world applications of decision tree algorithms are identification between cancerous and non-cancerous cells, suggestions to customers to buy a car, etc.

4. Support Vector Machine Algorithm

A support vector machine or SVM is a supervised learning algorithm that can also be used for classification and regression problems. However, it is primarily used for classification problems. The goal of SVM is to create a hyperplane or decision boundary that can segregate datasets into different classes.

The data points that help to define the hyperplane are known as **support vectors**, and hence it is named as support vector machine algorithm.

Some real-life applications of SVM are **face detection, image classification, Drug discovery**, etc. Consider the below diagram:



As we can see in the above diagram, the hyperplane has classified datasets into two different classes.

5. Naïve Bayes Algorithm:

Naïve Bayes classifier is a supervised learning algorithm, which is used to make predictions based on the probability of the object. The algorithm named as Naïve Bayes as it is based on **Bayes theorem**, and follows the *naïve* assumption that says' variables are independent of each other.

The Bayes theorem is based on the conditional probability; it means the likelihood that event(A) will happen, when it is given that event(B) has already happened. The equation for Bayes theorem is given as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Naïve Bayes classifier is one of the best classifiers that provide a good result for a given problem. It is easy to build a naïve bayesian model, and well suited for the huge amount of dataset. It is mostly used for **text classification**.

6. K-Nearest Neighbour (KNN)

K-Nearest Neighbour is a supervised learning algorithm that can be used for both classification and regression problems. This algorithm works by assuming the similarities between the new data point and available data points. Based on these similarities, the new data points are put in the most similar categories. It is also known as the lazy learner algorithm as it stores all the available datasets and classifies each new case with the help of K-neighbours. The new case is assigned to the nearest class with most similarities, and any distance function measures the distance between the data points. The distance function can be **Euclidean, Minkowski, Manhattan, or Hamming distance**, based on the requirement.

7. K-Means Clustering

K-means clustering is one of the simplest unsupervised learning algorithms, which is used to solve the clustering problems. The datasets are grouped into K different clusters based on similarities and dissimilarities, it means, datasets with most of the commonalties remain in one cluster which has very less or no commonalities between other clusters. In K-means, K-refers to the number of clusters, and **means** refer to the averaging the dataset in order to find the centroid.

It is a centroid-based algorithm, and each cluster is associated with a centroid. This algorithm aims to reduce the distance between the data points and their centroids within a cluster.

This algorithm starts with a group of randomly selected centroids that form the clusters at starting and then perform the iterative process to optimize these centroids' positions.

It can be used for spam detection and filtering, identification of fake news, etc.

TRAINING MODELS:

- Data Collection: Gather a substantial and diverse dataset of both spam and non-spam messages. This dataset should encompass various forms of communication like emails, text messages, and social media content.

- Data Preprocessing: Clean the data by removing irrelevant information (e.g., email headers, formatting) and standardizing the text. Text preprocessing tasks may include tokenization, stemming, and removing special characters or stop words.

- Feature Extraction: Convert the text data into numerical features that the machine learning model can understand. Common techniques include TF-IDF (Term Frequency-Inverse Document Frequency) and word embeddings (e.g., Word2Vec or GloVe).

- Labeling: Annotate the dataset, marking each message as either spam or non-spam. Ensure the dataset is well-balanced to avoid bias.Model Selection: Choose an appropriate machine learning algorithm or deep learning architecture for your spam classifier. Common choices include decision trees, random forests, support vector machines, or neural networks.

- Split Data: Divide the dataset into training, validation, and testing sets. The training set is used to teach the model, the validation set to fine-tune hyperparameters, and the testing set to evaluate the model's performance.

- Model Training: Train the selected model using the training data. The model learns to recognize patterns and features that distinguish spam from non-spam messages.

- Hyperparameter Tuning: Experiment with different hyperparameters (e.g., learning rates, batch sizes, number of layers) to optimize the model's performance. Use the validation set to assess the model's performance during this process.

- Evaluation: Evaluate the model's performance on the test dataset using various metrics like precision, recall, F1-score, and accuracy. These metrics help measure the model's ability to correctly classify spam

EVALUATING THE PERFORMANCE OF A SMARTER AI-POWERED SPAM CLASSIFIER

- Confusion Matrix: Create a confusion matrix to visualize the classifier's performance. It categorizes results into four groups: true positives, true negatives, false positives, and false negatives.

- Accuracy: Calculate accuracy by dividing the sum of true positives and true negatives by the total number of examples. However, accuracy alone can be misleading, especially in imbalanced datasets.

- Precision: Precision measures the proportion of true positive predictions among all positive predictions. It helps determine how often the classifier correctly identifies spam without falsely labeling non-spam as spam.

- Recall (Sensitivity): Recall calculates the proportion of true positive predictions among all actual positive cases. It shows how well the classifier captures all spam messages without missing too many.

- F1-Score: The F1-score combines precision and recall into a single metric, which is useful when you want to balance the trade-off between false positives and false negatives.

- Specificity: Specificity measures the proportion of true negative predictions among all actual negative cases. It's crucial to assess how well the classifier avoids falsely labeling non-spam as spam.

- ROC Curve and AUC: Plot the Receiver Operating Characteristic (ROC) curve to visualize the classifier's performance at different threshold levels. The Area Under the Curve (AUC) quantifies the overall performance. A higher AUC indicates better performance.

- Cross-Validation: Use k-fold cross-validation to assess the model's robustness and generalization across different subsets of the data. This helps avoid overfitting and provides a more accurate estimate of performance.

- Precision-Recall Curve: Plot a precision-recall curve to understand how precision and recall change at various threshold levels. This is particularly useful when dealing with imbalanced datasets.

- False Positive Rate (FPR): Calculate FPR as the proportion of false positives to actual negatives. It's essential to ensure that non-spam messages are not frequently misclassified as spam.

- False Negative Rate (FNR): Measure FNR as the proportion of false negatives to actual positives. It's important to minimize the risk of missing genuine spam messages.
- User Feedback: Gather feedback from users to identify false positives (legitimate messages classified as spam) and false negatives (spam messages that bypass the filter). This feedback can guide model improvements.
- Bias and Fairness Analysis: Evaluate the model for bias and fairness to ensure that it doesn't disproportionately impact specific user groups. Use fairness metrics like disparate impact and equal opportunity to assess this.
- A/B Testing: Conduct A/B testing by deploying the model in a controlled manner and comparing its performance to the previous system or versions.