**IBM AI –POWERED SPAM CLASSIFIER**

**PHASE 3: DEVELOPMENT OF PART-1**

* **data collection:**
* Gather a sizable dataset of emails of message, labeled as spam or not spam. Publicly available datasets like the Enron email dataset or online spam databases can be useful.
* **DATA PREPROCESSING:**
* **TEXT CLEANING:**
* Remove HTML tags, special characters, and any irrelevant content.
* **TOKENIZATION:**
* Split text into words or tokens.
* **LOWERCASTING:**
* Convert all text to lowercase to ensure consistency.
* **STOP WORD REMOVAL:**
* Eliminate common words like “the ”,”and”, etc., that don’t provide significant information.
* **STEMMING OR LEMMATIZATION:**
* Reduce words to their root form (e.g., “running” to “run”)
* **FEATURE EXTRACTION:**
* Convert text data into numerical format that AI models can understand. Common technique include TF-IDF (Term Frequency-Inverse Document Frequency) or word Embedding (word2Vec, Glove).
* **DATA SPLIT:**
* Split the dataset into a training set, a validation set, and a test set to train and evaluation your AI model.
* **Problem Definition:**
* Define the problem you want to solve, such as detecting spam messages in emails or text messages.
* **Data Collection:**
* Gather a large and diverse dataset of both spam and non-spam (ham) messages. This dataset is crucial for training and testing the spam classifier.
* **Data Preprocessing:**
* Clean and preprocess the data. This may involve tasks like tokenization, removing special characters, and lowercasing text.
* **Feature Extraction:**
* Extract relevant features from the text data. Common features include word frequencies, character n-grams, and various text statistics.
* **Data Labeling:**
* Annotate the dataset by labeling each message as spam or non-spam. This is typically done by human annotators or using pre-labeled datasets.
* **Model Selection:**
* Choose a machine learning or deep learning algorithm for spam classification. Common choices include Naive Bayes, Support Vector Machines, and neural networks.
* **Data Splitting:**
* Split the dataset into training, validation, and testing sets. The training set is used to train the model, the validation set helps with hyperparameter tuning, and the testing set assesses the model's performance.
* **Feature Engineering:**
* Experiment with different feature engineering techniques to improve the model's performance. This may involve feature scaling, dimensionality reduction, or feature selection.
* **Model Training:**
* Train the chosen model on the training dataset. During this phase, the model learns to distinguish between spam and non-spam messages.
* **Hyperparameter Tuning:**
* Fine-tune the model's hyperparameters using the validation set. Adjust parameters like learning rates, regularization strength, and network architecture for optimal performance.
* **Model Evaluation:**
* Evaluate the model's performance using the testing dataset. Common evaluation metrics include accuracy, precision, recall, F1-score, and ROC-AUC.
* **Model Deployment:**
* Once the model performs well, deploy it to a production environment. This may involve integrating it into an email system, a messaging app, or a web service.
* **Continuous Improvement:**
* Monitor the spam classifier's performance in production. Collect feedback and retrain the model periodically to adapt to evolving spam patterns.
* **Feedback Loop:**
* Implement a feedback loop where user interactions (e.g., marking messages as spam) are used to improve the model over time.
* **Security and Privacy:**
* Ensure that the spam classifier does not compromise user privacy and is secure against adversarial attacks.
* **Scalability:**
* Optimize the system for scalability to handle large volumes of messages and to be responsive to real-time demands.
* **Compliance:**
* Ensure that the spam classifier complies with relevant regulations and user data protection laws.