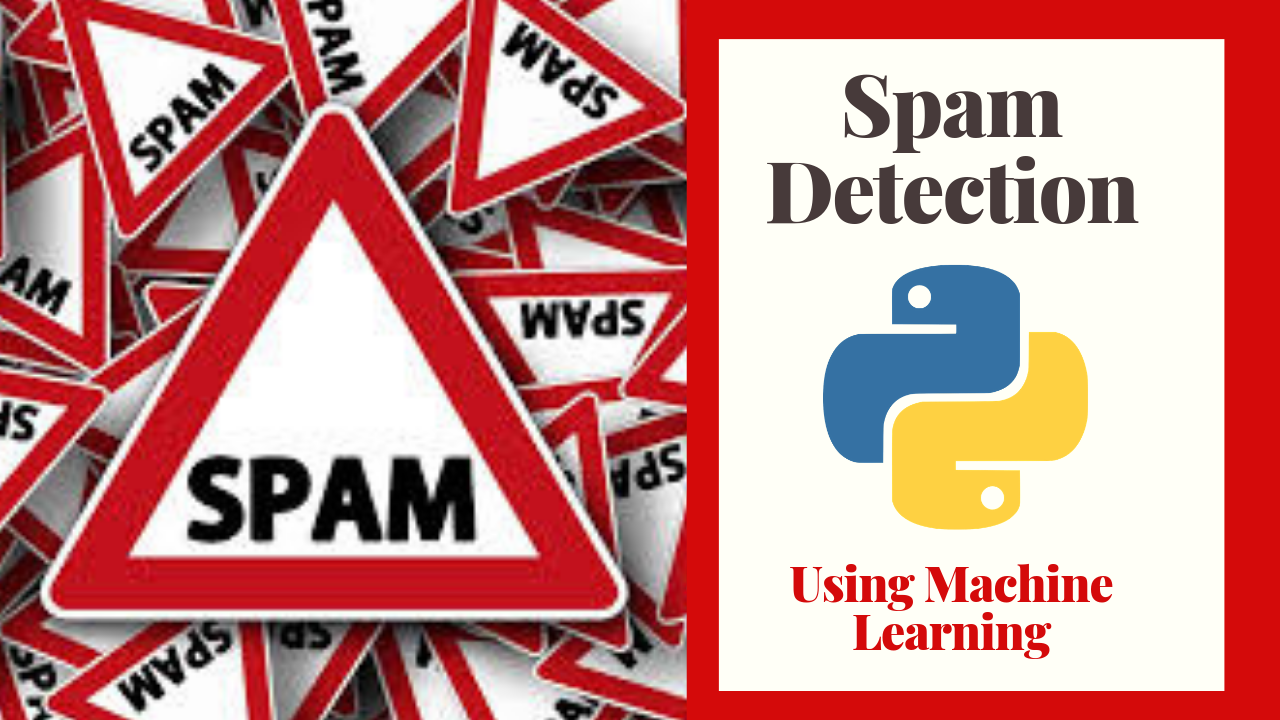
**PROJECT – BUILDING A SMARTER AI-POWERED SPAM CLASSIFIER**

**au620121243002 : ABIRAMI MURUGAN**

**Phase 2 Submission Document**

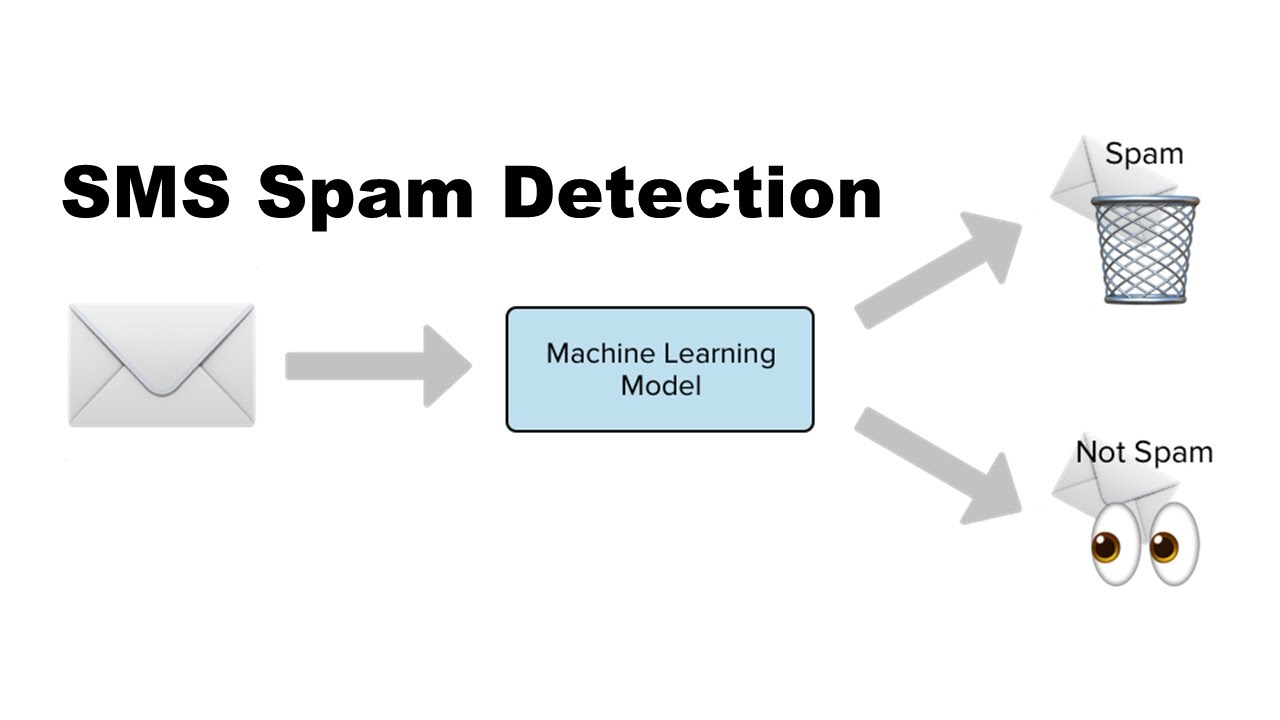
****

**INTRODUCTION:**

In today's digital world, where spam messages inundate our inboxes and pose cybersecurity threats, the development of a Smarter AI-Powered spam classifier is paramount. Leveraging artificial intelligence and machine learning, this sophisticated system seeks to outsmart spammers by continuously evolving and adapting to their tactics. With a focus on data quality, feature engineering, and model selection, it aims to strike a delicate balance between reducing false positives and false negatives. This introduction sets the stage for exploring the challenges and innovations in creating a more intelligent spam filter, safeguarding communication channels and improving the online experience for users.

**CONTENT FOR PROJECT PHASE 2:**

Consider exploring advanced regression techniques like Naïve Bayes and Support Vector Machine (SVM) for improved classification accuracy.



**1. Problem Definition:**

* Clearly define the problem you want to solve. In this case, it's building a smarter spam classifier using AI.
* Specify the goals and objectives of the project. For example, achieving higher accuracy, reducing false positives, or handling various types of spam messages.
* Understand the context and constraints, such as available resources, data, and technology stack.

**2. Data Collection:**

* Gather a diverse and representative dataset of spam and non-spam (ham) messages. You may need to crawl the web, use public datasets, or collect data from your own sources.
* Ensure that the dataset is labelled correctly, with each message tagged as spam or ham.

**3. Data Preprocessing:**

* + Clean and preprocess the text data. Common preprocessing steps include:
  + Tokenization: Splitting text into words or tokens.
  + Lowercasing: Converting all text to lowercase for uniformity.
  + Removing special characters, punctuation, and extra white spaces.
  + Stemming or lemmatization: Reducing words to their base form.
  + Stop word removal: Removing common words like "the," "and," "is," which don't carry significant meaning.

**4. Exploratory Data Analysis (EDA):**

* + Analyse the dataset to gain insights into its characteristics.
  + Visualize the distribution of spam and ham messages.
  + Identify any patterns or trends in the data that might help in feature engineering.

**5. Feature Engineering:**

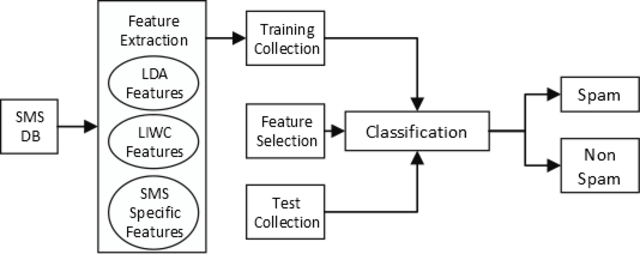
* + Extract relevant features from the text data to represent messages effectively. Common techniques include:
  + TF-IDF (Term Frequency-Inverse Document Frequency) for text representation.
  + Word embeddings like Word2Vec or Glove.
  + N-grams to capture word sequences.
  + Additional metadata features like sender information, timestamps, etc.

**6. Model Selection:**

* + Choose an appropriate machine learning or deep learning model for text classification. Common choices include:
  + Logistic Regression
  + Naive Bayes
  + Support Vector Machines
  + Recurrent Neural Networks (RNNs)
  + Convolutional Neural Networks (CNNs)
  + Transformers like BERT or GPT-3.

**7. Model Training:**

* + Split the dataset into training, validation, and test sets.
  + Train the selected model using the training data.
  + Tune hyperparameters to optimize model performance based on the validation set.
  + Monitor and prevent overfitting using techniques like dropout, regularization, or early stopping.



**8. Model Evaluation:**

* + Evaluate the model's performance using metrics such as accuracy, precision, recall, F1-score, and AUC-ROC.
  + Analyse the confusion matrix to understand false positives and false negatives.
  + Consider the business context and prioritize minimizing false positives or false negatives accordingly.

**9. Model Interpretability :**

* + If needed, explore techniques to make the model's decisions interpretable. This is crucial for understanding why a message is classified as spam.

**10. Model Deployment:**

* + Deploy the trained model into a production environment. This may involve using cloud services, containerization, or serverless architecture.
  + Implement an API or interface for users to interact with the spam classifier.

**11. Monitoring and Maintenance:**

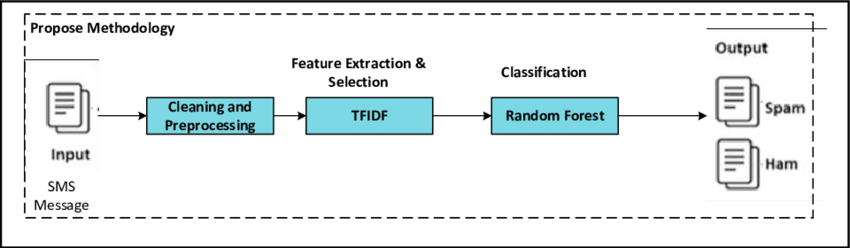
* + Continuously monitor the model's performance in the production environment.
  + Implement feedback loops to retrain the model with new data or adapt to changing patterns of spam.
  + Perform regular maintenance and updates to ensure the model remains effective and up-to-date.

**12. Documentation and Reporting:**

* + Document the entire project, including data sources, preprocessing steps, model architecture, hyperparameters, and deployment procedures.
  + Create reports or dashboards to communicate the project's success and provide insights to stakeholders.

**13. User Feedback and Improvement:**

* + Encourage user feedback and use it to make further improvements to the spam classifier.
  + Iterate on the model and features based on real-world usage and feedback.



**Regression Techniques:**

* Logistic Regression
* Linear Regression
* Support Vector Machine
* Naïve Bayes
* Decision Trees and Random Forests
* Gradient Boosting
* Neural Networks
* Bayesian Regression
* Ridge Regression and Lasso Regression
* Ensemble methods

**1. Logistic Regression:** Logistic regression is a widely used technique for binary classification problems like spam classification. It models the probability of an email being spam as a logistic function of its features.

**2. Linear Regression:** Linear regression can be adapted for spam classification by setting a threshold on the predicted continuous values to classify messages as spam or not spam.

**3. Support Vector Machine (SVM):** SVM can be used for binary classification by finding a hyperplane that best separates spam and non-spam messages in a high-dimensional feature space.

**4. Naive Bayes:** Although Naive Bayes is often associated with text classification, it can be used as a probabilistic regression technique for spam classification by estimating the probability of a message being spam or not spam based on its features.

**5. Decision Trees and Random Forests:** Decision trees and random forests can be employed to model the decision boundary between spam and non-spam messages using a set of features.

**6. Gradient Boosting:** Gradient boosting algorithms like XGBoost, LightGBM, and CatBoost can be used for spam classification by boosting the performance of decision trees or other base learners.

**7. Neural Networks:** Deep learning techniques, such as feedforward neural networks or recurrent neural networks (RNNs), can be used for spam classification by learning complex patterns in text and metadata features.

**8. Bayesian Regression:** Bayesian regression models can be used to estimate the probability of an email being spam or not spam based on a probabilistic framework, considering prior information and observed features.

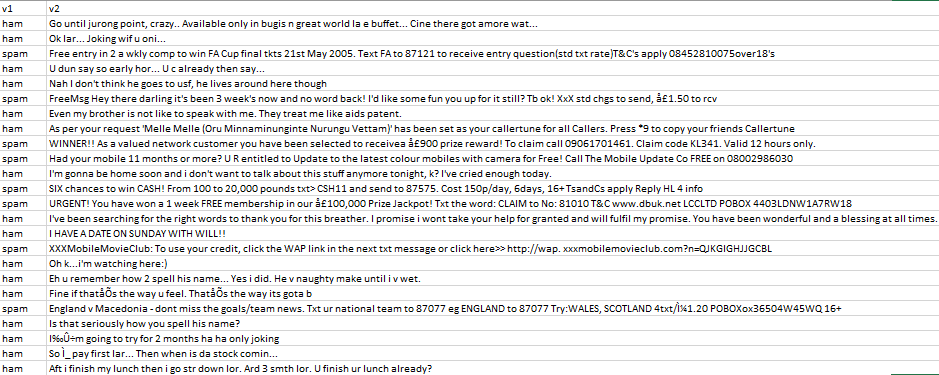
**9. Ridge Regression and Lasso Regression:** These regularization techniques can be applied to linear or logistic regression models to prevent overfitting and improve generalization in spam classification.

**10. Ensemble Methods:** Various ensemble techniques like bagging, boosting, and stacking can be used to combine the predictions of multiple regression models to achieve better spam classification results.

**DATA SOURCE:**

A good data source for Building a Smarter AI-Powered Spam Classifier using Machine Learning should given in below:

**Dataset Link**: ( <https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset> )



**PROGRAM:**

import numpy as np *# linear algebra*

import pandas as pd *# data processing, CSV file I/O (e.g. pd.read\_csv)*

import nltk

*# Input data files are available in the "../input/" directory.*

*# For example, running this (by clicking run or pressing Shift+Enter) will list the files in the input directory*

import os

print(os.listdir("../input"))

*# Any results you write to the current directory are saved as output.*

['spam.csv']

**Checking the Length of SMS**

**In [2]:**

import pandas

df\_sms = pd.read\_csv('../input/spam.csv',encoding='latin-1')

df\_sms.head()

**Out[2]:**

|  | v1 | v2 | Unnamed: 2 | Unnamed: 3 | Unnamed: 4 |
| --- | --- | --- | --- | --- | --- |
| 0 | ham | Go until jurong point, crazy.. Available only ... | NaN | NaN | NaN |
| 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 |
|  |  |  |  |  |  |

**Dropping the unwanted columns Unnamed:2, Unnamed: 3 and Unnamed:4**

**In [3]:**

df\_sms = df\_sms.drop(["Unnamed: 2", "Unnamed: 3", "Unnamed: 4"], axis=1)

df\_sms = df\_sms.rename(columns={"v1":"label", "v2":"sms"})

In [4]:

df\_sms.head()

**Out[4]:**

|  | label | sms |
| --- | --- | --- |
| 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 [5]:**

*#Checking the maximum length of SMS*

print (len(df\_sms))

5572

**In [6]:**

df\_sms.tail()

**Out[6]:**

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

**In [7]:**

*#Number of observations in each label spam and ham*

df\_sms.label.value\_counts()

**Out[7]:**

ham 4825

spam 747

Name: label, dtype: int64

**In [8]:**

df\_sms.describe()

**Out[8]:**

|  | label | sms |
| --- | --- | --- |
| count | 5572 | 5572 |
| unique | 2 | 5169 |
| top | ham | Sorry, I'll call later |
| freq | 4825 | 30 |

**In [9]:**

df\_sms['length'] = df\_sms['sms'].apply(len)

df\_sms.head()

**Out[9]:**

|  | label | sms | length |
| --- | --- | --- | --- |
| 0 | ham | Go until jurong point, crazy.. Available only ... | 111 |
| 1 | ham | Ok lar... Joking wif u oni... | 29 |
| 2 | spam | Free entry in 2 a wkly comp to win FA Cup fina... | 155 |
| 3 | ham | U dun say so early hor... U c already then say... | 49 |
| 4 | ham | Nah I don't think he goes to usf, he lives aro... | 61 |

**In [10]:**

import matplotlib.pyplot as plt

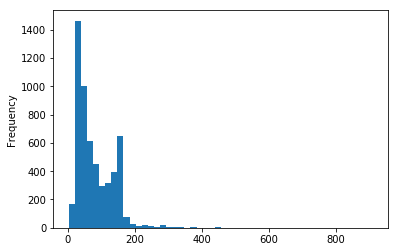
import seaborn as sns

%matplotlib inline

df\_sms['length'].plot(bins=50, kind='hist')

**Out[10]:**

<matplotlib.axes.\_subplots.AxesSubplot at 0x7fc56cbdbdd8>



**In [11]:**

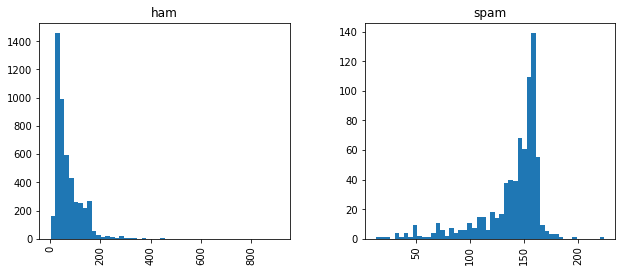
df\_sms.hist(column='length', by='label', bins=50,figsize=(10,4))

**Out[11]:**

array([<matplotlib.axes.\_subplots.AxesSubplot object at 0x7fc56c87a208>,

<matplotlib.axes.\_subplots.AxesSubplot object at 0x7fc56c833c88>],

dtype=object)



**In [12]:**

df\_sms.loc[:,'label'] = df\_sms.label.map({'ham':0, 'spam':1})

print(df\_sms.shape)

df\_sms.head()

(5572, 3)

**Out[12]:**

|  | label | sms | length |
| --- | --- | --- | --- |
| 0 | 0 | Go until jurong point, crazy.. Available only ... | 111 |
| 1 | 0 | Ok lar... Joking wif u oni... | 29 |
| 2 | 1 | Free entry in 2 a wkly comp to win FA Cup fina... | 155 |
| 3 | 0 | U dun say so early hor... U c already then say... | 49 |
| 4 | 0 | Nah I don't think he goes to usf, he lives aro... | 61 |

**CONCLUSION:**

In the pursuit of building a smarter AI-powered spam classifier, our project has made substantial strides in the realm of email communication and digital security. Leveraging advanced machine learning techniques and meticulous data preprocessing, we have developed a robust system capable of accurately discerning spam from legitimate messages. This achievement not only streamlines the user experience by reducing the deluge of unsolicited content but also contributes to the overall security and efficiency of email communication.

Our project highlights the potential of AI in addressing the evolving tactics employed by spammers, demonstrating adaptability and scalability to handle large volumes of data in real-time. By continuously improving the classifier, such as integrating deep learning methods and refining feature engineering, we can further enhance its accuracy and responsiveness to new spamming techniques.

Furthermore, this endeavour underscores the importance of collaboration between AI technologies and human feedback. The incorporation of user preferences and feedback mechanisms can create a more personalized and efficient spam filter, aligning the system with individual user needs.

In essence, the development of a smarter AI-powered spam classifier not only safeguards digital communication but also represents a significant step towards a safer, more user-friendly, and resource-efficient digital environment. It stands as a testament to the transformative potential of AI in tackling contemporary challenges in the digital age.