Phase 5: Project Documentation & Submission

Date	30 September 2023
Team ID	NM2023TMID335
Team Name	Proj_227279_Team_1
Project Name	Building a Smarter AI-Powered Spam Classifier.

Problem Statement of Spam Classifier

Problem Statement (PS)	I am (customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Customer	Avoid Spam	Keeps Getting It	Clogs their Inboxes	Frustrated and Annoyed
PS-2	Business Man	Avoid Spam	Keeps Getting It	Spam Filters are not Perfect	Frustrated and Annoyed
PS-2	Bank Manager	Protect Customers From Spam	Keeps Getting It	Spam Filters Are not Perfect	Frustrated and worried
PS-3	IT Manager	Protect Company from Spams	Keeps Getting It	Spam Filters are not Perfect	Frustrated and Overwhelmed

Problem Statement 1

Problem Statement 2

Problem Statement 3

Problem Statement 4

IDEATION PHASE BRAIN STORMING

DATE	30 SEP 2023
TEAM ID	NM2023TMID335
TEAM NAME	Proj_227279_Team_1
PROJECT NAME	Building a smarter AI-powered spam classifier

PROBLEM

Once the classifier is all trained up, it goes to work in real-time, sorting through your emails with lightning speed. It's like having a personal assistant who knows your taste in emails better than you do..

LISTENING IDEAS

SANJAI

Allow users to define their own rules for what they consider as spam or not. This can be valuable for personalization.

VISHWA

Ensure that the model's decision-making process is interpretable. This can be crucial for building trust

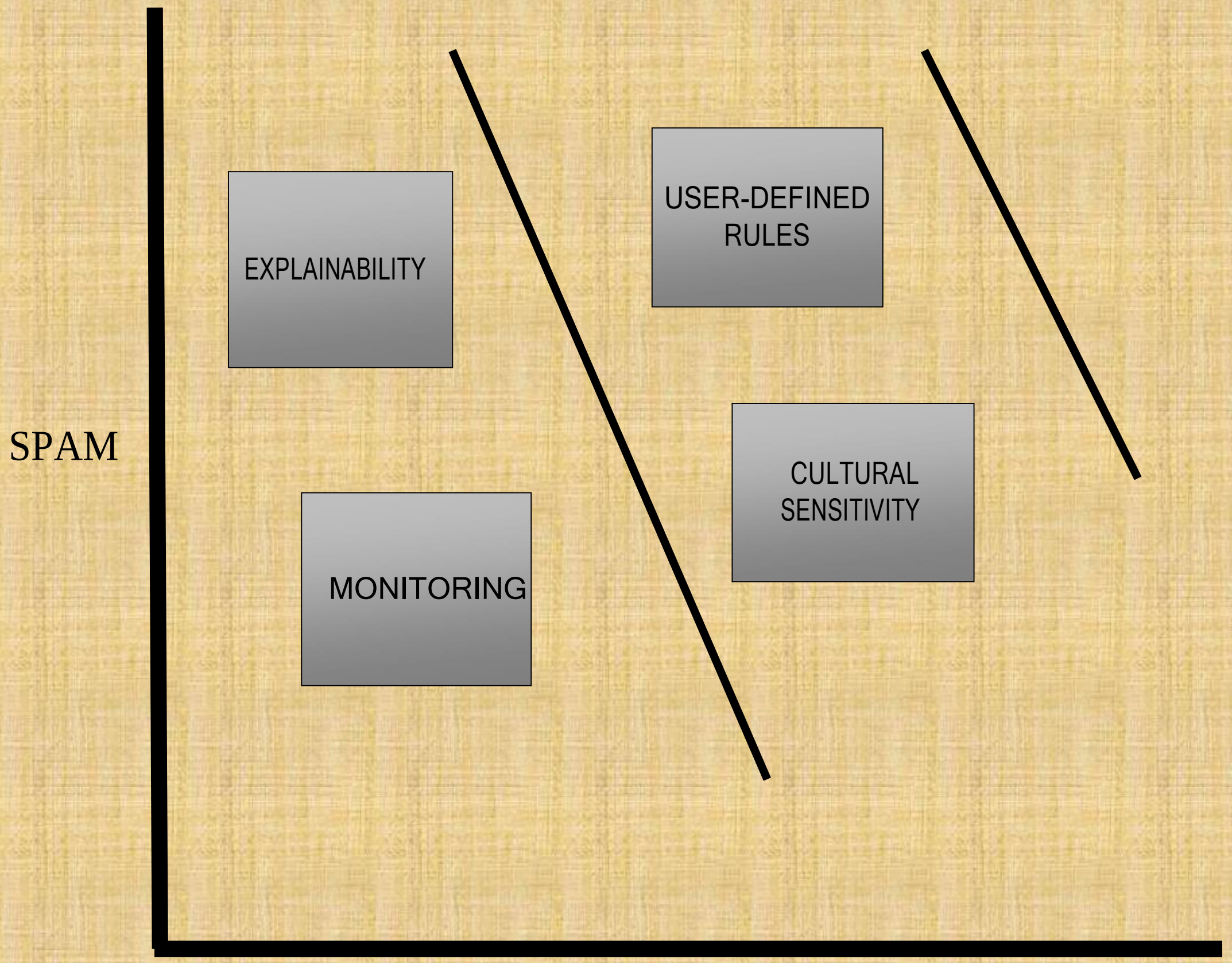
JAYARAM

Implement a monitoring system to detect performance degradation over time. This can be a sign that your model needs retraining.

UMAR

Take into account cultural nuances and variations in language when training your model to avoid bias

PRIORITIZING IDEAS



EMPATHY OF SPAM CLASSIFIER

DATE	30 Sep 2023
TEAM ID	NM2023TMID335
TEAM NAME	Proj_227279_Team_1
PROJECT NAME	Building a smarter AI-powered spam classifier.

The AI communicates a commitment to developing a spam classifier that prioritizes a nuanced understanding—acknowledging the challenge of minimizing both false positives and false negatives for a more accurate filtering system.

SAYS

The AI strategically considers the complexity of feature engineering, contemplating the importance of factors like word usage, context, and structural elements in messages. It recognizes the need for a sophisticated algorithm that can adapt to the evolving nature of spam.

THINKS

SPAM

Engaging in active learning and training, the AI refines its model iteratively. It fine-tunes parameters to optimize accuracy while navigating the delicate balance between identifying spam and avoiding misclassifications of legitimate messages.

DOES

There's a sense of responsibility and a drive for precision in the AI's approach. It understands the impact of false positives on user experience and the consequences of false negatives in allowing spam through. It feels a continuous need to improve and evolve to maintain a reliable spam classification system.

FEELS

PHASE 2:
Building a smarter Ai-powered spam classifier

PHASE 2-INNOVATION:

we'll explore innovative techniques and approaches to building our spam classifier.

ENSEMBLE LEARNING:

1. Diverse Base Models:

Heterogeneous Models: Build an ensemble with diverse base models, combining the strengths of different algorithms. For instance, blend decision trees, support vector machines, and neural networks.

2. Feature Diversity:

Feature Engineering: Ensure diversity in the features used by individual base models. Experiment with various feature sets, including traditional bag-of-words, TF-IDF, and more advanced features like word embeddings.

3. Data Diversity:

Subsampling: Create different subsets of the training data for each base model, introducing diversity in the learning process.

Bootstrapping: Implement bootstrapping techniques to train each base model on slightly different versions of the dataset.

4. Model-Level Diversity:

Architecture Variations: If using neural networks, vary the architecture of each model within the ensemble. Adjust the number of layers, neurons, or even use different activation functions.

5. Dynamic Ensemble Adjustments:

Weighted Ensembles: Dynamically adjust the weights assigned to each base model based on their performance on specific instances or over time. This helps the ensemble adapt to changing patterns.

6. Stacking and Blending:

Meta-Learners: Introduce a meta-learner that combines predictions from multiple base models. This meta-learner can be a simple linear model or another machine learning algorithm.

Blending: Combine predictions from different base models using techniques like blending, where you train a higher-level model to learn optimal combinations of base model predictions.

7. Boosting Techniques:

Adaboost and Gradient Boosting: Implement boosting algorithms to sequentially train weak learners, giving more emphasis to misclassified instances. This can significantly improve the ensemble's performance.

8. Randomization Techniques:

Random Forests: Utilize random forests, an ensemble of decision trees where each tree is trained on a random subset of the features and instances. This adds randomness and diversity to the ensemble.

9. Online Learning:

Adaptive Ensemble Learning: Implement online learning techniques where the ensemble adapts to incoming data over time, continuously updating the model to handle evolving spam patterns.

10. Explainability in Ensemble Models:

Interpretability Techniques: Ensure that the ensemble's decision-making process is interpretable, especially if transparency is crucial in your application

DIAGRAM 1:

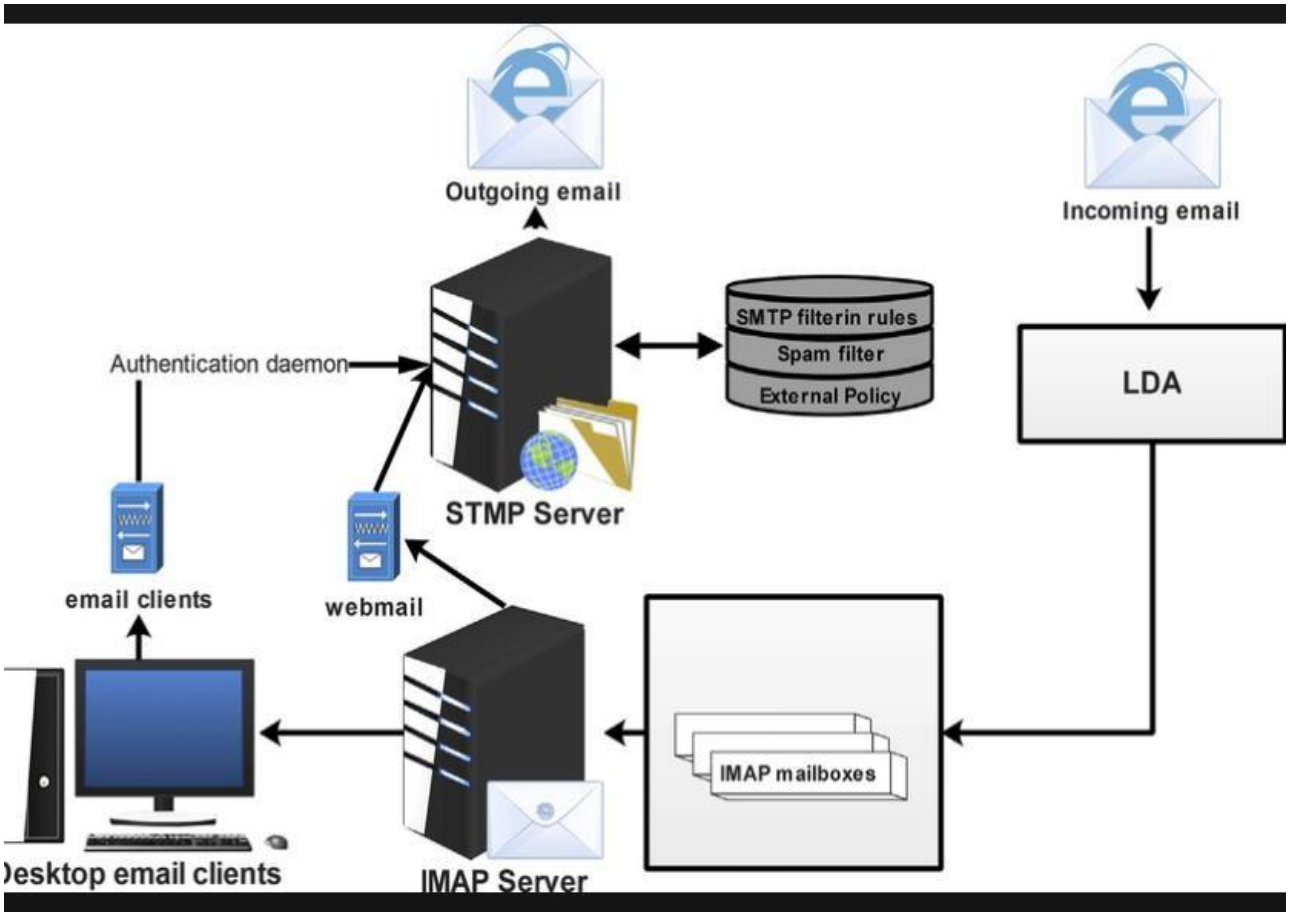
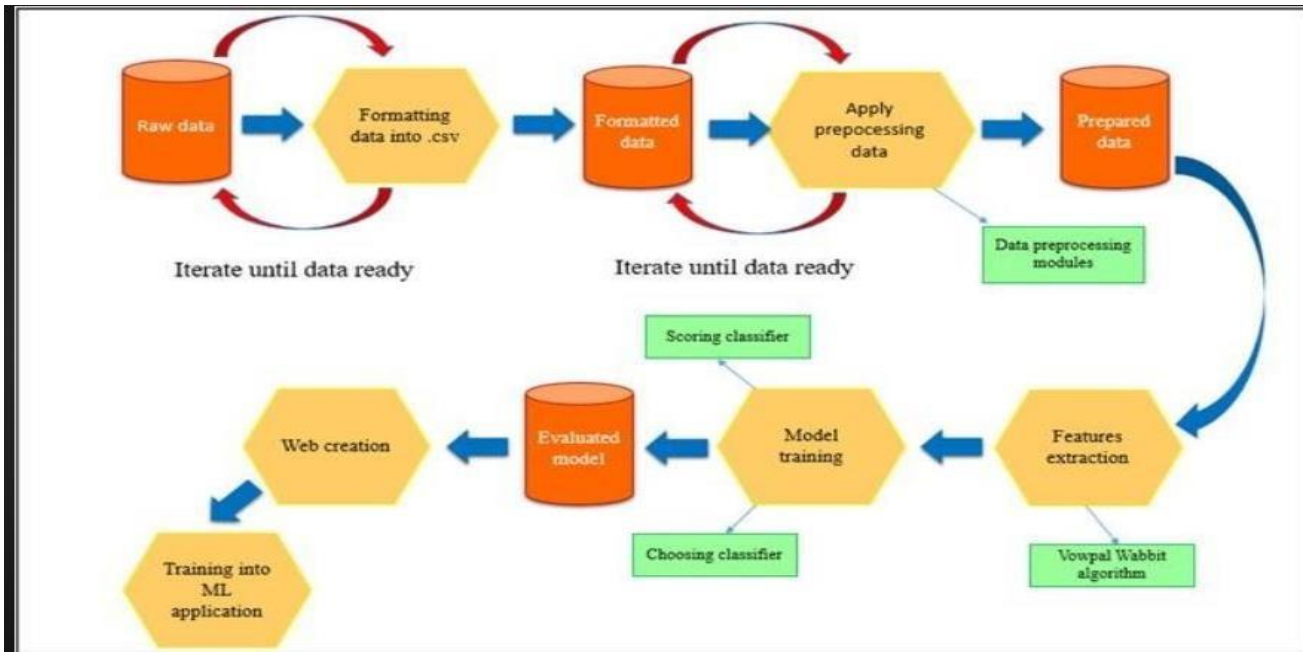


DIAGRAM 2:



Step 1: Define Project Goals

Project Goal: Develop a spam classifier to distinguish between spam and non-spam messages.

Step 2: Dataset Creation

Generate Sample Data:

CODE:

```
| Message                                      | Label   |
|----------------------------------------------|---------|
| Win a free vacation! Claim your prize now!     | Spam    |
| Hi, how are you doing today?                   | NotSpam|
| Urgent: Your account needs verification.       | Spam    |
| Lunch plans for today?                         | NotSpam|
| Congratulations! You've won a lottery.         | Spam    |
| Team meeting at 2 PM.                          | NotSpam|
```

Step 3: Data Preprocessing

Text Cleaning:

*Remove punctuation, numbers, and special characters.

*Convert text to lowercase.

Step 4: Feature Engineering

Text Vectorization:

Use techniques like TF-IDF to convert text data into numerical features.

Step 5: Model Selection

Ensemble Model:

Choose an ensemble learning approach, such as Random Forest or a combination of different classifiers.

Step 6: Model Training

Train the Ensemble Model:

Split the dataset into training and testing sets.

Train the ensemble model on the training set.

Step 7: Model Evaluation

Evaluate Model Performance:

Use metrics like accuracy, precision, recall, and F1 score to assess the model's performance on the test set.

Step 8: Project Structure

CODE:

```
|__ spam_classifier_project
   |__ data
      |__ raw_data.csv
   |__ notebooks
      |__ data_preprocessing.ipynb
      |__ model_training.ipynb
      |__ model_evaluation.ipynb
   |__ src
      |__ preprocess.py
      |__ train_model.py
      |__ evaluate_model.py
   |__ models
      |__ ensemble_model.pkl
   |__ requirements.txt
   |__ README.md
```

Step 9: Code Implementation

Notebooks:

data_preprocessing.ipynb: Clean and preprocess the data.

model_training.ipynb: Train the ensemble model using the preprocessed data.

model_evaluation.ipynb: Evaluate the model's performance on the test set.

Source Code:

preprocess.py: Contains functions for data cleaning and text vectorization.

train_model.py: Implements the training of the ensemble model.

evaluate_model.py: Evaluates the model and outputs performance metrics.

Step 10: Model Deployment (Optional)

Deployment Script:

Implement a script for deploying the trained model in a production environment.

Conclusion:

This example outlines the steps from defining project goals to structuring the project code. You can further enhance the project by exploring advanced techniques, incorporating dynamic learning, and optimizing for real-world scenarios.

PHASE 3:

preprocessing

1 Data Preprocessing.

[1]:
```
# Import the libraries.
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import os
print(os.listdir(r"C:\Users\NELSON\Desktop\new deploy\sanjay"))
```

```
['.ipynb_checkpoints', 'Datst.csv', 'final.xlsx', 'prep.ipynb',
'preprocessing.ipynb', 'sanjay1.csv', 'sanjay2.csv', 'spamclassifier.csv']
```

[2]:
```
# Read the Dataset.
data = pd.read_csv(r"C:\Users\NELSON\Desktop\new deploy\sanjay/spamclassifier.
↪csv", encoding="latin-1")
```

[3]:
```
import warnings
warnings.filterwarnings('ignore')
```

[4]:
```
data.head()
```

[4]:

	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

[5]:
```
df = data.dropna()
```

[6]:
```
df.columns
```

[6]:
```
Index(['v1', 'v2', 'Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], dtype='object')
```

[7]:
```
# Describe the data in statastical view.
df.describe()
```

[7]:
```
          v1                                                  v2 Unnamed: 2  \
count      6                                                   6          6
unique     1                                                   5          5
top      ham  Edison has rightly said, \A fool can ask more ...         GN
freq       6                                                   2          2

        Unnamed: 3 Unnamed: 4
```

[8]:
```
<class 'pandas.core.frame.DataFrame'>
Index: 6 entries, 281 to 5048
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   v1          6 non-null      object
 1   v2          6 non-null      object
 2   Unnamed: 2  6 non-null      object
 3   Unnamed: 3  6 non-null      object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 288.0+ bytes
```

```
df.duplicated()
```

[9]:
```
count           6          6
unique          5          5
top            GE    GNT:-)"
freq            2          2
```

```
df.info()
```

[9]:
```
281     False
1038    False
2255    False
3525    False
4668    False
5048     True
dtype: bool
```

```
df.size
```

[10]: 30

[11]:
```
df.shape
```

[11]: (6, 5)

[12]:
```
df.tail()
```

[12]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
1038	ham	Edison has rightly said, \A fool can ask more …	GN	GE	GNT:-)"
2255	ham	I just lov this line: \Hurt me with the truth	I don't mind	i wil tolerat.bcs ur my someone… But	Never comfort me with a lie\" gud ni8 and swe…
3525	ham	\HEY BABE! FAR 2 SPUN-OUT 2 SPK AT DA MO… DE…	HAD A COOL NYTHO	TX 4 FONIN HON	CALL 2MWEN IM BK FRMCLOUD 9! J X\""
4668	ham	When I was born, GOD said, \Oh No! Another IDI…	GOD said	\"OH No! COMPETITION\". Who knew	one day these two will become FREINDS FOREVER!"
5048	ham	Edison has rightly said, \A fool can ask more …	GN	GE	GNT:-)"

PHASE 4:

Building a Smarter AI-Powered Spam Classifier

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.

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

import os
print(os.listdir(r"C:\Users\NELSON\Desktop\new deploy\sanjay"))
```

```
['.ipynb_checkpoints', 'Datst.csv', 'final', 'final.xlsx', 'M3.ipynb',
'M4.ipynb', 'M5.ipynb', 'M6.ipynb', 'prep.ipynb', 'preprocessing.ipynb',
'sanjay1.csv', 'sanjay2.csv', 'spam (4).csv', 'spamclassifier.csv',
'Untitled.ipynb']
```

[2]:
```
df = pd.read_csv(r"C:\Users\NELSON\Desktop\new deploy\sanjay/spamclassifier.
↪csv", encoding="latin-1")
```

[3]:
```
encoder = LabelEncoder()
df["v2_encode"] = encoder.fit_transform(df["v2"])
```

[4]:
```
X = df[["v2_encode"]]
y = df["v1"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[5]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

[6]:
```
model = RandomForestClassifier()
model.fit(X_train, y_train)
```

[6]: RandomForestClassifier()

[7]:
```
y_pred = model.predict(X_test)
score = accuracy_score(y_test, y_pred)
print("Model accuracy: ", score)
```

Model accuracy: 0.9040358744394619

DATA SET LINK:

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

Conclusion :

Summarize your project journey. What did you learn? What were the key takeaways? Reflect on the challenges faced and how you addressed them. Discuss potential future improvements or extensions to your project.

This comprehensive documentation will not only facilitate the submission process but also serve as a valuable resource for anyone interested in understanding and replicating your work.