Implementation Documentation

Email Spam Detection using Machine Learning & Neural Networks**

---

1. Overall Working of the Program

This project detects whether an email message is **Spam** or **Ham (Not Spam)** using Machine Learning and Deep Learning models.

The program follows these steps:

Step 1 — Load Dataset

- The dataset **spam.csv** is loaded from Kaggle.
- Only first two columns (label, text) are used.

Step 2 — Data Preprocessing

- Remove missing values.
- Convert text to lowercase (handled internally by TF-IDF).
- Convert labels:
  - *ham → 0*
  - *spam → 1*

Step 3 — Convert Emails to Numerical Form

- Use **TF-IDF Vectorizer** to convert email text into numerical vectors.
- Vector size = 5000 features.

Step 4 — Train Machine Learning Model

Two models are trained:

Model 1: Linear Regression (used as classifier)

- Outputs continuous values → converted to 0/1.

Model 2: Deep Neural Network

Architecture:

- Dense(512)

- Dropout(0.3)
- Dense(256)
- Dropout(0.3)
- Dense(1, sigmoid)

Model is trained on **10 epochs**.

Step 5 — Evaluate the Models

- Accuracy
- Precision
- Recall
- F1-score
- Confusion Matrix
- ROC-AUC score (for regression model)

Step 6 — Predict New Email Text

User gives an email → program returns:

- SPAM / HAM
- Confidence score

---

2. Functions and Classes Used

Machine Learning Section

Function/Class	Purpose
`pd.read_csv()`	Load CSV dataset
`TfidfVectorizer()`	Convert email text into numerical vectors
`LinearRegression()`	Train ML model
`accuracy_score()`	Evaluate classification accuracy
`confusion_matrix()`	Compute confusion matrix
`precision_recall_fscore_support()`	Compute precision, recall, F1
`joblib.dump()`	Save trained models

Deep Learning Section (Keras/TensorFlow)

Class/Function	Purpose
`Sequential()`	Build neural network

Class/Function	Purpose
`Dense()`	Fully connected layer
`Dropout()`	Prevent overfitting
`compile()`	Set optimizer, loss, metrics
`model.fit()`	Train for X epochs
`model.evaluate()`	Evaluate on test data
`model.predict()`	Predict new email result

📓 Custom Helper Functions

`predict_email_text(text)`

- Vectorizes text using TF-IDF
- Predicts using Linear Regression model
- Returns label + probability score

`predict_email(text)`

- Predicts spam/ham using Neural Network

---

3. Input and Output Format

Input

1. **Dataset File**:
   - `spam.csv` (must be in the same directory)
   - Format:

     label **text**
     ham hello how are you
     spam win 1000$ now!!

2. **User Input for Prediction**
   A string containing email text:
3. `predict_email("Congratulations! You won a prize.")`

Output

For Model Training

- Accuracy
- Loss
- Epoch-wise training progress
- Confusion matrix
- Sample predictions

For Single Email Prediction

```
{'label': 'SPAM', 'score': 0.92}
```

Or from NN model:

```
"SPAM"
```

---

4. Libraries and Frameworks Required

Python Libraries

```
pandas
numpy
scikit-learn
matplotlib
seaborn
scipy
joblib
```

Deep Learning Framework

```
tensorflow==2.x
keras
```

---

5. Execution Instructions (How to Run the Program)

Option 1 — Run in Google Colab (Recommended)

1. Upload:
   - `spam.csv`
   - Python notebook (`.ipynb`)
2. Install required packages (Colab already has most installed):

```
!pip install pandas numpy scikit-learn tensorflow seaborn matplotlib
```

3. Run each cell one by one.
4. After training, test your custom email:

```
predict_email("You have won a cash prize! Click the link now")
```

Option 2 — Run on Local Machine

Step 1: Install Python 3.10+

Step 2: Install libraries

```
pip install pandas numpy scikit-learn tensorflow seaborn matplotlib scipy
joblib
```

Step 3: Place files inside a folder

```
project/
   spam.csv
   spam_detection.py
```

Step 4: Run program

```
python spam_detection.py
```

Step 5: Predict a new email

Modify the last line of the script:

```
print(predict_email("Free lottery win now!"))
```

---

6. Summary

This documentation explains:

✔ Program workflow
✔ Functions used
✔ Input/output format
✔ Required libraries
✔ How to execute the project

This is complete and professional for submitting in your semester project.