**CA1 DATA SCIENCE AND**

**MACHINE LEARNING PORTFOLIO LOG**

**Date: February 28th 2025**

**Prepared by:**

Kirubel Temesgen

Student Number: C00260396

**Course:** Data Science and Machine Learning

**Supervisor:** Greg Doyle

**CRISP-DM Documentation for Spam Email Classification using Neural Network (MLP)**

**1. Business Understanding**

The objective of this project is to classify emails as either **Spam** or **Not Spam** using a **Neural Network (MLP)** classifier. The goal is to identify spam emails from a collection of email data based on various features extracted from the subject, body, and sender of the email.

Key business questions:

* How effective is a machine learning model in detecting spam in emails?
* Can we improve spam detection accuracy using features like subject, body content, and sender email domain?

**2. Data Understanding**

The dataset is composed of email data, including:

* **Message:** The full email content.
* **From:** The sender's email address.
* **Subject:** The email's subject.
* **Body:** The email body.

Steps taken:

1. **Data Loading:** The dataset is loaded into a pandas dataframe for analysis.
2. **Feature Extraction:** The 'From', 'Subject', and 'Body' of the emails are extracted using regular expressions.
3. **Data Preprocessing:**
   * Cleaning of text data by removing special characters, numbers, and common prefixes such as "Re:" or "FW:".
   * Rule-based spam detection using specific spam keywords (e.g., "free", "urgent", "money") and domain checks for non-"@enron.com" email senders.
4. **Missing Data:** Rows with missing key columns such as 'From', 'Subject', or 'Body' are dropped to ensure high-quality data for modeling.

**3. Data Preparation**

Data preprocessing steps:

1. **Text Cleaning:**
   * The clean() function normalises the text by converting it to lowercase, removing special characters, numbers, and common prefixes like "Re:" or "FW:".
   * Text cleaning is applied to both the subject and body of the email.
2. **Feature Engineering:**
   * The cleaned text from the subject and body is transformed using the **TF-IDF (Term Frequency-Inverse Document Frequency)** vectorizer.
   * TF-IDF is applied separately to the **Subject** and **Body** columns, and the results are combined into a single sparse matrix.
3. **Rule-based Spam Classification:**
   * A **rule-based system** checks for the presence of specific spam keywords and flags emails as spam if certain conditions are met. For example, emails from domains outside of "@enron.com" and those containing spam keywords are considered spam.
4. **Target Variable:**
   * The target variable **Spam\_Label** is created using the rule-based classification system. If an email meets the criteria (score ≥ 3), it is labeled as **Spam** (1), otherwise as **Not Spam** (0).

**4. Modeling**

* **Model Selection:** The **MLPClassifier** (Multi-layer Perceptron) is chosen as the model to classify emails. It is a type of neural network with one hidden layer, using the ReLU (Rectified Linear Unit) activation function.
* **Model Parameters:**
  + Hidden Layer Sizes: (64,)
  + Activation: ReLU
  + Solver: Adam (a popular optimisation algorithm)
  + Max Iterations: 100
  + Random State: 42 (for reproducibility)
* **Training and Testing:**
  + The data is split into **80% training** and **20% testing** using train\_test\_split.
  + The model is trained using the training data and then evaluated on the test data.

**5. Evaluation**

* **Model Accuracy:** The accuracy of the model on the test set is calculated and reported.
* **Classification Report:** The **precision**, **recall**, and **F1-score** are calculated for both **Not Spam** and **Spam** categories to evaluate model performance in detail.
* **Confusion Matrix:** A **confusion matrix** is plotted to visualise the performance of the model, showing the true positives, true negatives, false positives, and false negatives.
* **Key Metrics from Evaluation:**
  + **Accuracy:** The proportion of emails correctly classified as either spam or non-spam.
  + **Precision:** The proportion of emails predicted as spam that are actually spam.
  + **Recall:** The proportion of actual spam emails correctly identified by the model.
  + **F1-Score:** The harmonic mean of precision and recall, providing a balanced measure.

**6. Deployment**

Once the model is trained and evaluated, it can be deployed as part of an email spam detection system. This system can be integrated into email clients or services to automatically classify incoming emails as spam or not spam based on the learned model.

**Key Insights & Results:**

1. **Neural Network Performance:** The neural network (MLP) classifier achieved a good level of accuracy in classifying emails, with balanced precision and recall across both spam and non-spam categories.
2. **Classification Report:**
   * **Spam Class Precision (0.81)** and **Recall (0.68)** indicate that the model is relatively good at detecting spam but can still improve in minimising false negatives (missed spam).
   * **Non-Spam Precision (0.98)** and **Recall (0.99)** indicate that the model is highly accurate in classifying non-spam emails.
3. **Improvement Opportunities:**
   * The model's performance can be improved by fine-tuning hyperparameters or exploring more complex neural network architectures.
   * Further feature engineering, such as incorporating more contextual features or using advanced models LSTM may enhance the model’s ability to detect subtle patterns in the text.

References

*MLPClassifier* (no date) *scikit*. Available at: <https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html> (Accessed: 23 March 2025).

GeeksforGeeks (2025) *Classification using Sklearn Multi-layer Perceptron*, *GeeksforGeeks*. Available at: <https://www.geeksforgeeks.org/classification-using-sklearn-multi-layer-perceptron/> (Accessed: 22 March 2025).

Rcassani (no date) *RCASSANI/MLP-example: Code for a simple MLP (multi-layer perceptron)*, *GitHub*. Available at: <https://github.com/rcassani/mlp-example> (Accessed: 24 March 2025).

Nova (2023) *A comprehensive guide to using Scikit-Learn’s mlpclassifier*, *AITechTrend*. Available at: <https://aitechtrend.com/a-comprehensive-guide-to-using-scikit-learns-mlpclassifier/> (Accessed: 242 March 2025).