Internship Project: Email Spam Detection using NLP and ML

Phase 0: Project Brief and Problem Framing

Objective:

Understand what the project is about and why spam detection matters.

Tasks:

1. Research:
    a. What is spam?
    b. Why is spam filtering important?
2. Write a 1-page document answering:
    a. What's the difference between spam and ham?
    b. What challenges exist in spam detection?

Deliverable:

A short written report (PDF/DOC) explaining the problem.

Phase 1: Dataset Loading and Exploration

Objective:

Load the dataset and inspect its structure.

Tasks:

1. Load CSV using pandas.
2. Use `.head()`, `.info()`, and `.describe()`.
3. Print the number of spam vs ham messages.
4. Visualize the label distribution using a bar plot.

Deliverable:

Notebook with code, visualization, and short explanation.

Hint:

Use `df['label'].value_counts()` and `seaborn.countplot()`.

Phase 2: Data Cleaning & Preprocessing

Objective:

Clean and transform raw email text into usable format.

Tasks:

1. Convert all text to lowercase.
2. Remove punctuation, special characters (use regex).
3. Tokenize the text.
4. Remove stopwords (use NLTK or spaCy).
5. Apply stemming or lemmatization.

Deliverable:

New column `clean_text` added to DataFrame. Show a few before/after examples.

Hint:

Use `re.sub`, `nltk.word_tokenize()`, `PorterStemmer()` or `WordNetLemmatizer()`.

Phase 3: Text Vectorization

Objective:

Convert processed text into numerical format.

Tasks:

1. Choose and apply either:
    a. Bag of Words (CountVectorizer)
    b. OR TF-IDF (TfidfVectorizer)
2. Fit-transform training data and transform test data.
3. Print shape of resulting vectors.

Deliverable:

Vectorized data arrays and short explanation.

Hint:

Use `sklearn.feature_extraction.text.TfidfVectorizer`.

Phase 4: Train/Test Split

Objective:

Prepare data for model training.

Tasks:

1. Create label column: 1 = spam, 0 = ham.
2. Split data into training and test sets (80/20).
3. Ensure random_state is set for reproducibility.

Deliverable:

X_train, X_test, y_train, y_test arrays with shapes printed.

Hint:

Use `train_test_split()` from `sklearn.model_selection`.

Phase 5: Model Training

Objective:

Train a machine learning classifier to detect spam.

Tasks:

1. Train at least one of:
    a. Multinomial Naive Bayes (recommended for text)
    b. Logistic Regression
    c. Support Vector Machine
2. Fit model on training data.
3. Save the model using `joblib`.

Deliverable:

Trained model file and training accuracy.

Phase 6: Model Evaluation

Objective:

Evaluate model performance using various metrics.

Tasks:

1. Predict on test data.
2. Calculate:
    a. Accuracy
    b. Precision
    c. Recall
    d. F1-score
3. Plot the confusion matrix.

Deliverable:

Evaluation report + visualization.

Hint:

Use `classification_report` and `confusion_matrix`.

Phase 7: Real-Time Prediction Function

Objective:

Enable prediction on new unseen emails.

Tasks:

1. Write a function:

```
def predict_email_spam(text):
    # clean, vectorize, predict
    return result
```

2. Test with 3–5 sample emails.

Deliverable:

Prediction function + test outputs.

Phase 8: Streamlit Web App

Objective:

Create a web-based UI to detect spam.

Tasks:

1. Use `st.text_area()` to collect user input.
2. Load model and vectorizer.
3. Display result with `st.success()` or `st.error()`.

Deliverable:

Streamlit app (`app.py`) that accepts text and shows prediction.

Phase 9: Final Report & Testing

Objective:

Wrap everything up and reflect.

Tasks:

1. Test app with edge-case emails.
2. Write a final report (1–2 pages):
    a. Summary of approach
    b. Challenges
    c. Future improvements (e.g., BERT, LSTM)

Final Student Deliverables:

- `Email_spam_detection.ipynb`
- `app.py`
- `spam_classifier_model.pkl`
- `tfidf_vectorizer.pkl`
- Report (DOC/PDF)
- Optional: deployed app link