# KIET GROUP OF INSTITUTIONS

## ACADEMIC SESSION 2024-2025

## INTRODUCTION TO AI

## Assessment Report On

# DETECT SPAM EMAILS

| | |
|---|---|
| NAME | Ayush Rao Chaudhary |
| BRANCH | CSE (AI & ML ) |
| SEC | A |
| ROLL.NO | 67 |
| SUBJECT | Introduction To AI |
| SUB CODE | AI101B |
| UNDER THE SUPERVISION OF | MR Bikki Gupta Sir |

# INTRODUCTION

Email Spam Is A Serious Concern In The Digital World. Identifying Whether An Email Is Spam Or Not Helps In Improving Email User Experience And Security. In This Project, A Logistic Regression Model Is Implemented To Classify Spam Emails Based On Features Available In The Dataset.We Used Python With Libraries Such As Numpy, Pandas, Seaborn, And Scikit-learn. The Dataset Used Contains Labeled Spam And Non-spam Emails.

# METHODOLOGY

## Data Loading

A Csv File Containing Email Data Was Loaded Using Pandas.

## Preprocessing

- the Target Variable Is_spam Was Encoded To Binary (Yes → 1, No → 0).
- Features And Labels Were Separated.
- StandardScaler Was Used To Standardize The Feature Set.

## Model Building

A Logistic Regression Model Was Trained On The Training Data.

## Prediction & Evaluation

The Model's Performance Was Evaluated Using Confusion Matrix, Accuracy, Precision, And Recall.

# CODE IMPLEMENTATION

```
# Cell 1: Import Libraries And Load Data
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load Dataset
df = pd.read_csv("/content/spam_emails.csv")

# Display First Few Rows
df.head()
```

```
# Cell 2: Preprocessing and Train-Test Split
from sklearn.model_selection import
train_test_split
from sklearn.preprocessing import
StandardScaler

# Encode target variable
df['is_spam'] = df['is_spam'].map({'yes': 1,
'no': 0})

# Split features and target
X = df.drop('is_spam', axis=1)
y = df['is_spam']

# Standardize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test =
train_test_split(X_scaled, y, test_size=0.2,
random_state=42)
```

```
#Cell 3: Train Logistic Regression Model
from sklearn.linear_model import
LogisticRegression

# Initialize and train model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

#Cell 4: Confusion Matrix Heatmap
from sklearn.metrics import confusion_matrix

# Generate confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Plot heatmap
plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt='d',
cmap='Blues',
            xticklabels=['Not Spam', 'Spam'],
            yticklabels=['Not Spam', 'Spam'])

plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

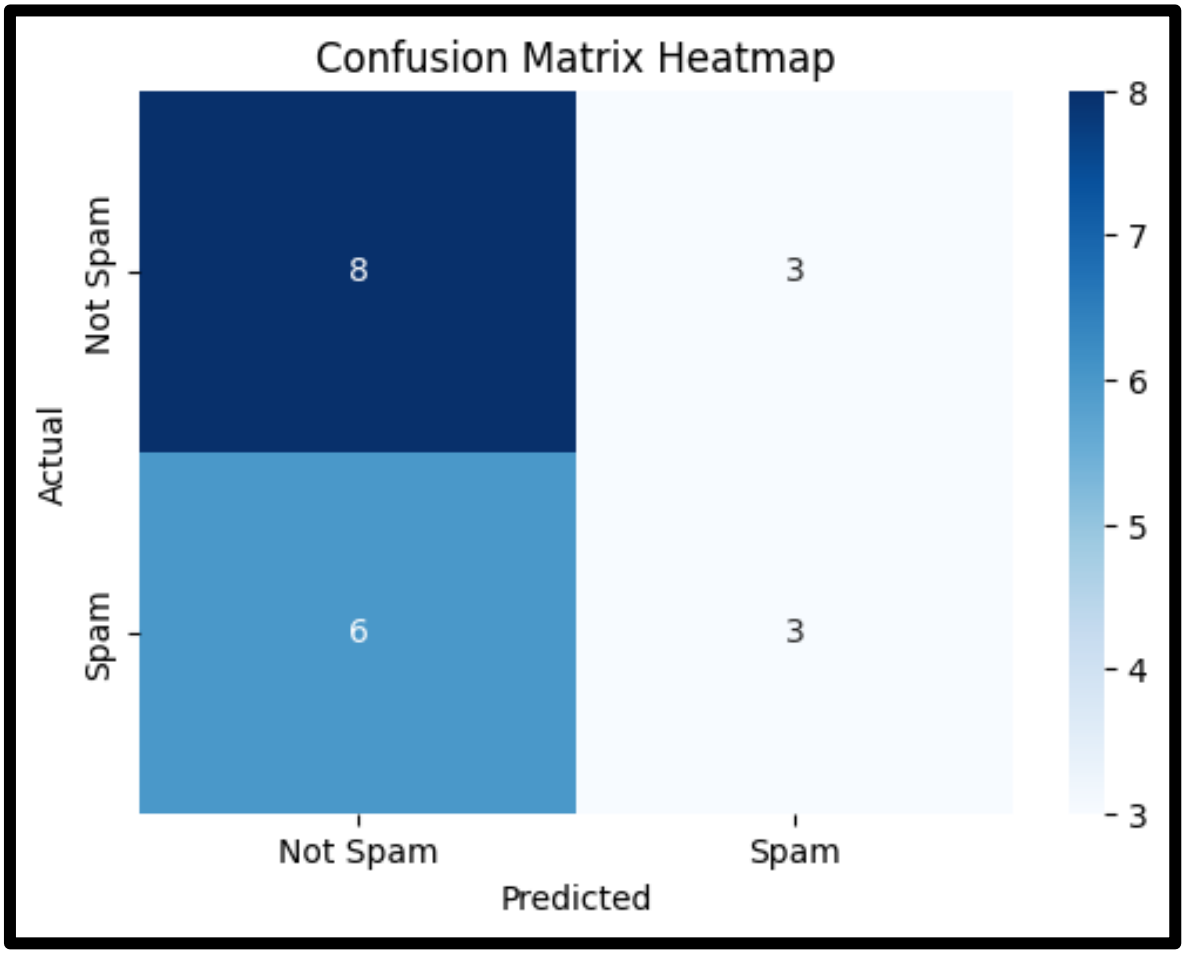
```
Cell 5: Evaluation Metrics
from sklearn.metrics import accuracy_score,
precision_score, recall_score

# Calculate evaluation metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)

# Display results
print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
```

# Output & Results

## ✅ Confusion Matrix Heatmap



# ✅ Display Results

```
Accuracy: 0.55
Precision: 0.50
Recall: 0.33
```

# REFERENCES & CREDITS

- **Python Documentation:**
  https://docs.python.org/3/

- **Matplotlib Documentation:**
  https://matplotlib.org/

- **Pandas Documentation:**
  https://pandas.pydata.org/

- **Number Theory Concepts:**
  https://en.wikipedia.org/wiki/Prime_number