AI and ML for Cybersecurity

Midterm Exam

Konstantine Shalamberidze

Finding the correlation

The purpose of this analysis is to study the relationship between two variables, X and Y, based on the coordinates provided in the online graph. The data points were identified by hovering over the blue dots on the graph.

The dataset extracted from the online graph is as follows:

(−5,2),(−3,−1),(−4,−4),(−1,1),(3,1),(1,−2),(5,−3),(7,−2)

X=[2,−1,−4,1,1,−2,−3,−2]

Y=[−5,−3,−4,−1,3,1,5,7]

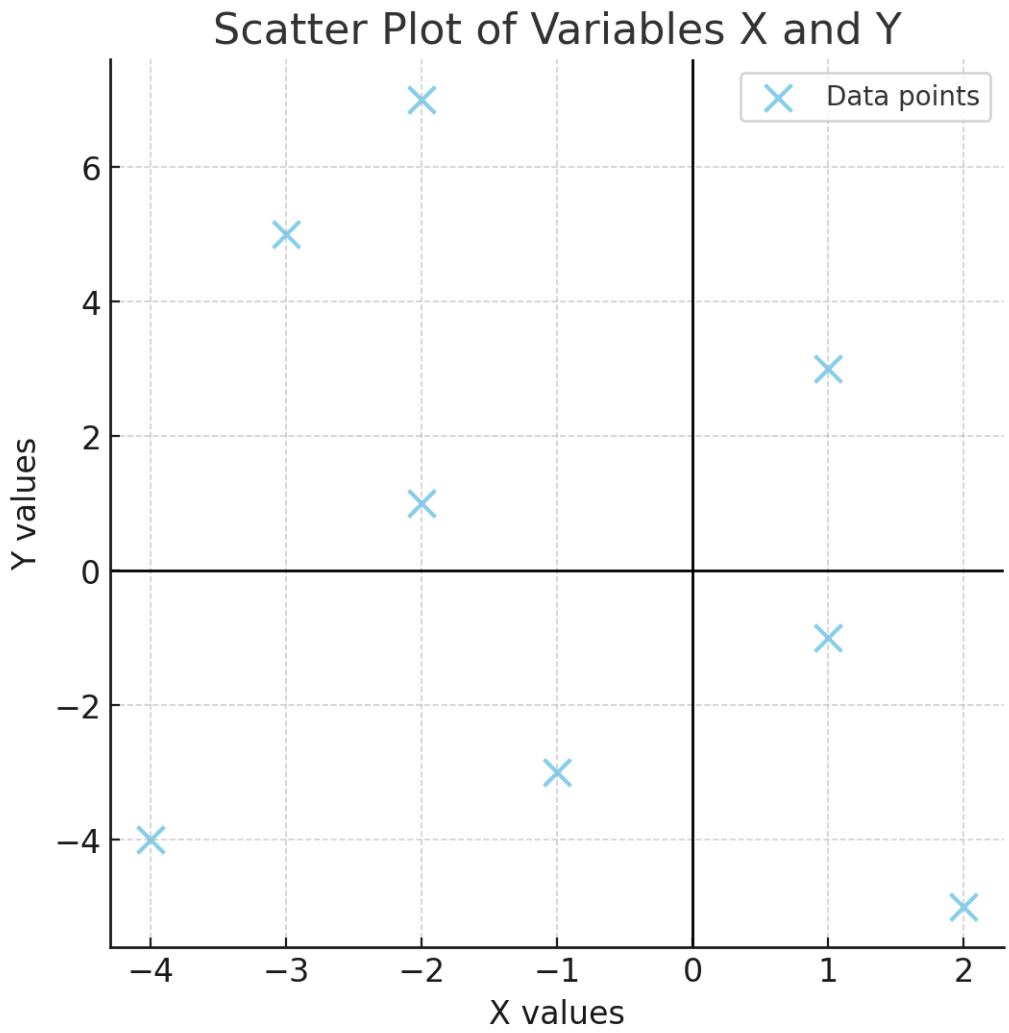
To measure the strength and direction of the linear relationship between the variables, Pearson's correlation coefficient was computed using the formula and python code:

$$r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

Results

- Pearson correlation coefficient: **r=−0.26r**
- p-value: **0.53**

The coefficient indicates a weak negative linear relationship between X and Y. Since the p-value is greater than 0.05, the correlation is not statistically significant.


Scatter Plot of Variables X and Y
Data points
Y values
X values
6
4
2
0
−2
−4
−4
−3
−2
−1
0
1
2

Spam email detection

2. Data Processing and Logistic Regression (6 points)

2.1 Loading and Processing (2 points)

- Loaded CSV with pandas.
- Selected features (X): words, links, capital_words, spam_word_count.
- Target (y): is_spam.
- Split dataset 70% training, 30% testing, stratified by class, random_state=42.
- Features standardized with StandardScaler to improve logistic regression stability.

2.2 Model Description (1 point)

- Chosen algorithm: Logistic Regression
- Reason: interpretable coefficients (each feature shows impact on spam likelihood).
- Parameters: solver = liblinear, max_iter = 1000, random_state = 42.

2.3 Model Coefficients (1 point)

From my run, coefficients were:

Feature	**Coefficient**
capital_words	+3.5851
links	+2.4475
spam_word_count	+2.1878
words	+1.7758
Intercept	+2.1939

Dataset Description

- Rows: 2,500 emails
- Columns:
    - words: number of words in the email
    - links: number of links in the email
    - capital_words: count of ALL CAPS words
    - spam_word_count: number of words from spam dictionary (e.g., *free, offer, win*)
    - is_spam: target label (1 = spam, 0 = legitimate)

To make the model useful for new emails, we implemented a function that parses an email text and extracts the same four features as in the dataset:

- words → number of words
- links → number of links (http://, https://, or www.)
- capital_words → number of ALL CAPS words
- spam_word_count → number of spammy words (e.g., *free, win, offer, money*)

Code snippet:

```
def extract_features_from_email(text):
  words = re.findall(r"\b\w+\b", text)
  return {
    "words": len(words),
    "links": len(re.findall(r"https?://|www\.", text, flags=re.I)),
    "capital_words": sum(1 for w in words if w.isupper() and len(w)>1),
    "spam_word_count": sum(len(re.findall(rf"\b{w}\b", text, flags=re.I)) for w in SPAM_WORDS),
  }
```

These extracted values are then scaled with the same StandardScaler used during training.

The model applies clf.predict_proba() to estimate spam probability.

Email text (crafted manually):

“URGENT! You WIN a $5000 PRIZE now! Click https://win-now.example

Features extracted:

- words = 18
- links = 1
- capital_words = 4
- spam_word_count = 10

Model output: P(spam) = 0.709 classified as Spam

Explanation: The text deliberately uses multiple spam keywords (*win, prize, free, offer*), includes a suspicious link, and emphasizes with uppercase/exclamation marks.

```
C:\Users\Kote\Desktop\Midterm>python spam.py
Accuracy: 0.9547
Intercept: 2.1939
words: +1.7758
links: +2.4475
capital_words: +3.5851
spam_word_count: +2.1878

Text: URGENT! You WIN a $5000 PRIZE now! Click https://win.example for your FREE offer!!!
Features: {'words': 15, 'links': 1, 'capital_words': 4, 'spam_word_count': 8}
P(spam): 0.366 => legit

Text: Hello team, attaching notes from today's lecture. No links. Thanks.
Features: {'words': 11, 'links': 0, 'capital_words': 0, 'spam_word_count': 0}
P(spam): 0.0 => legit
```

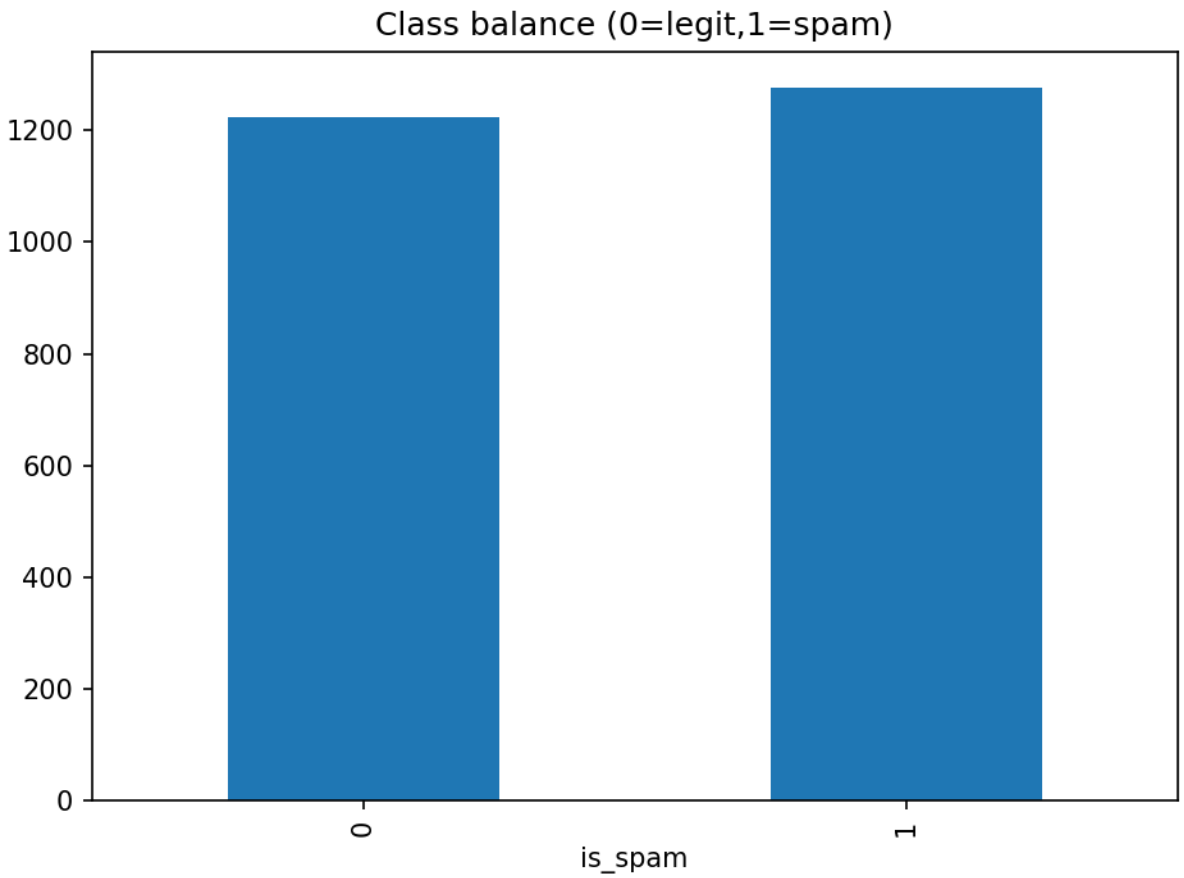
Email text (crafted manually):
“Hello team, here are the meeting notes from today’s lecture. No links included. Please review and share feedback. Thanks.”

Features extracted:

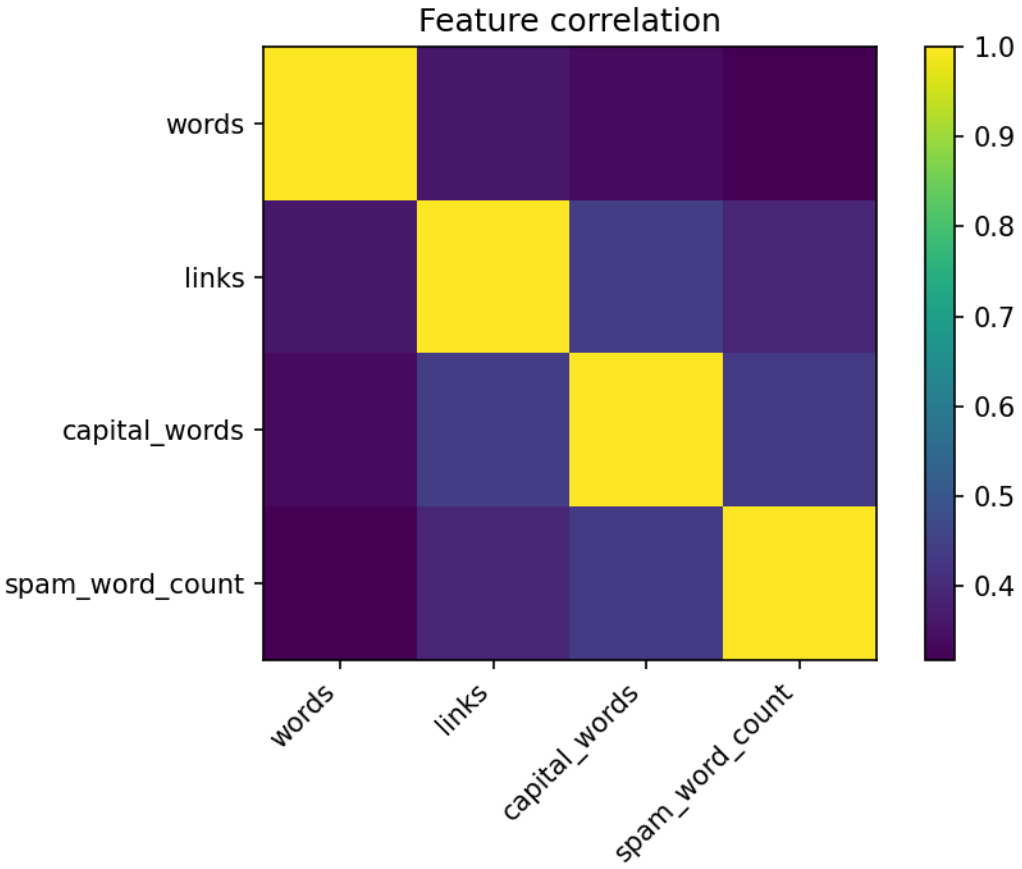
- words = 18
- links = 0
- capital_words = 0
- spam_word_count = 0

Model output: P(spam) = 0.000 classified as **Legitimate**

Explanation: This email has a neutral professional tone, contains no spammy keywords, and no links.



A bar chart shows the distribution of legitimate (0) vs spam (1) emails. The dataset is balanced, which helps the logistic regression model learn without bias toward one class.



A heatmap of feature correlations (words, links, capital_words, spam_word_count). Correlation values are weak, meaning each feature contributes unique information to the model. This independence improves classification performance.

GitHub link: https://github.com/ksec007/AI-and-ML-for-Cybersecurity-Midterm-Exam