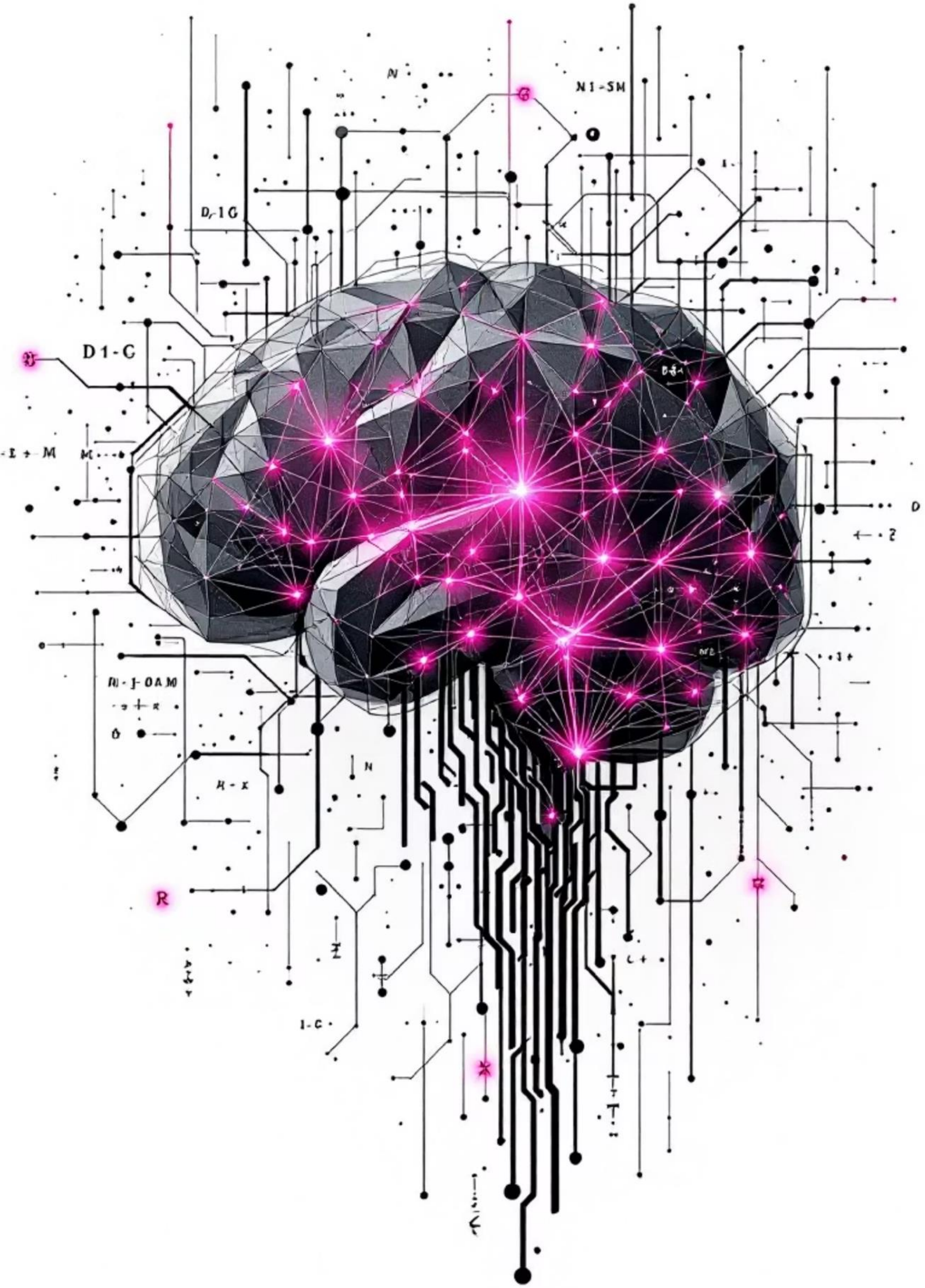


Working with Naive Bayes Algorithm

Prasanna Ranjith Christodoss
Associate Professor of Computer Science
Department of Computing, Mathematics & Physics
Messiah University
Mechanicsburg, PA 17055
T 717-796-1800 | **Ext** 2739
E prchristodoss@messiah.edu | **W** messiah.edu

Why Naive Bayes Matters: Fast, Simple, and Powerful

Inbox Protection

Powers spam filters protecting billions of inboxes daily, analyzing millions of messages in real-time with remarkable accuracy.

Text Classification

Drives sentiment analysis and news categorization, helping businesses understand customer feedback and organize information at scale.

Lightning Fast Training

Trains incredibly fast even on massive datasets with thousands of features, making it ideal for production environments.

The Core Idea: Bayes' Theorem in Action

At its heart, Naive Bayes leverages one of probability theory's most powerful insights: the ability to reverse conditional probabilities and make predictions based on evidence.

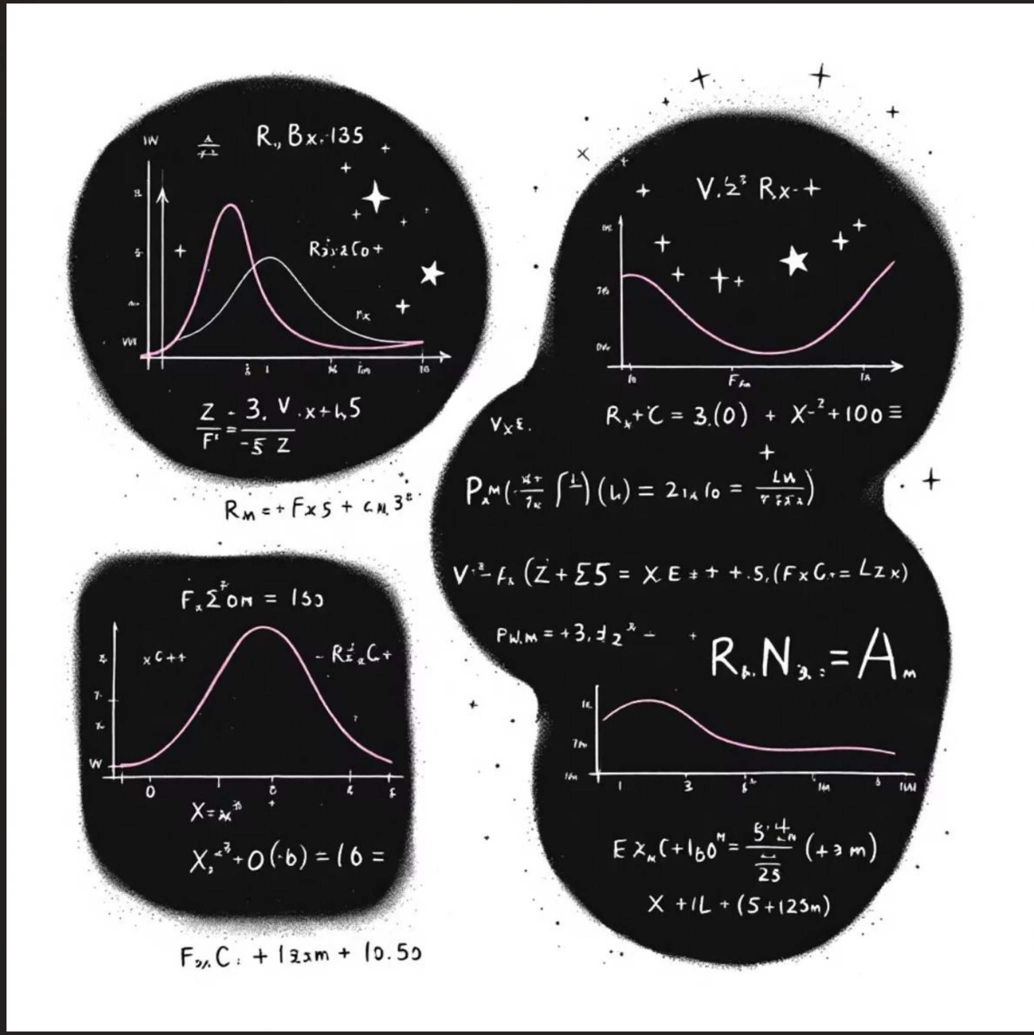
Bayes' Theorem Formula

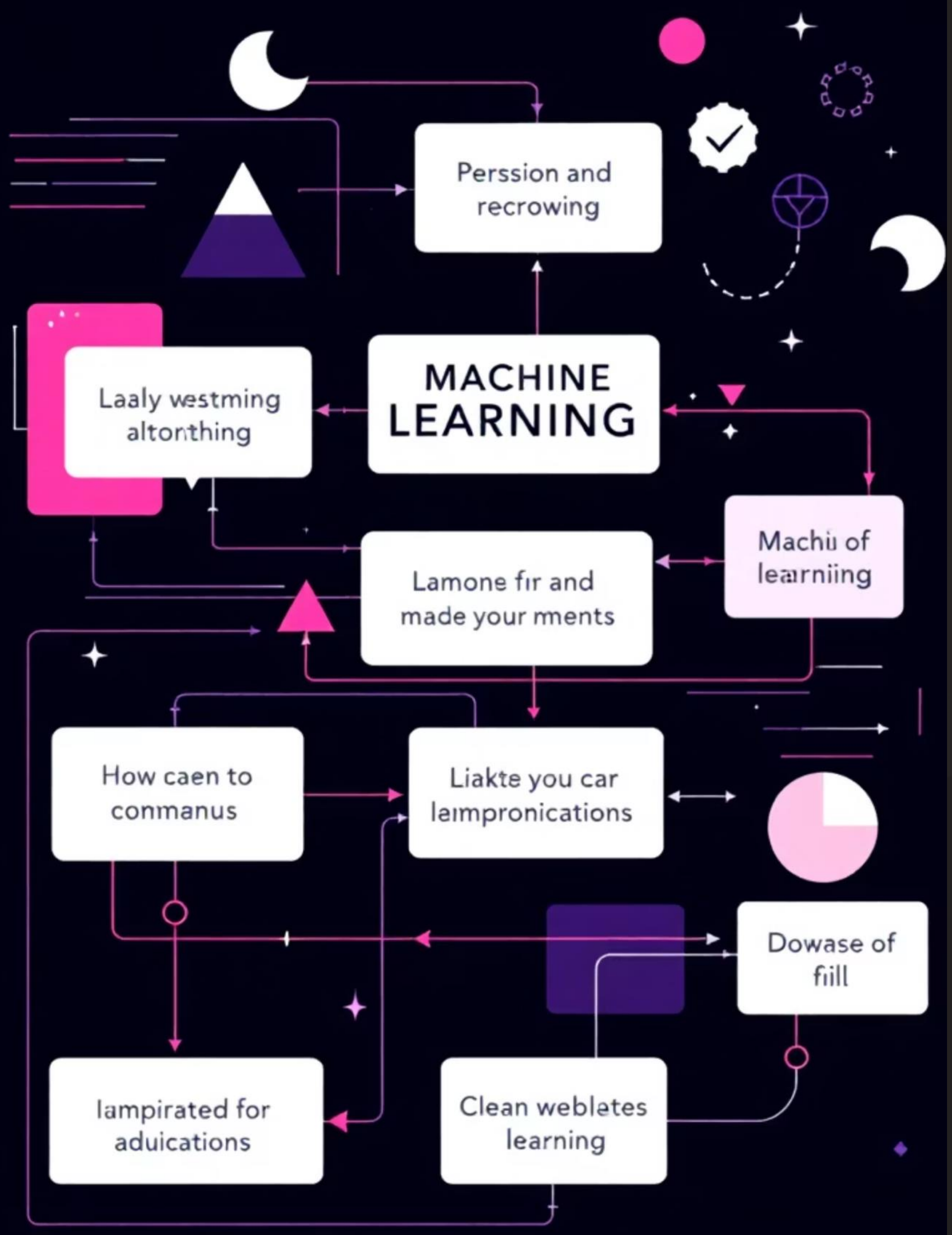
$$P(\text{Class} \mid \text{Features}) \propto P(\text{Class}) \times P(\text{Features} \mid \text{Class})$$

This elegant equation reverses conditional probabilities, allowing us to predict a class given observed features by combining prior knowledge with likelihood.

The Naive Independence Assumption

Naive Bayes assumes features are independent of each other given the class. While this is rarely true in practice, this "naive" assumption simplifies calculations drastically and enables efficient, scalable classification across diverse applications.





How Naive Bayes Works: Step-by-Step

Calculate Prior Probabilities

Examine the training data to determine the baseline probability of each class appearing. For example, if 30% of emails are spam, P(spam) = 0.3.

Estimate Feature Likelihoods

For each class, calculate the probability of observing each feature. This tells us how likely specific words or attributes appear in spam versus legitimate emails.

Multiply Likelihoods

Assuming independence, multiply the likelihoods of all observed features together with the prior probability to compute posterior probabilities for each class.

Maximum A Posteriori Decision

Predict the class with the highest posterior probability. This represents the most likely classification given all available evidence.

Core Concept: Bayes’ Theorem

- Posterior = (Prior × Likelihood) ÷ Evidence
- Prior = how likely a class is before seeing features
- Likelihood = how likely features are given that class
- Evidence = how likely the features are in general
- In practice you often compare numerators since evidence is same for all classes

The "Naive" Part

- Instead of modeling $P(x_1, x_2, \ldots x_n \mid \text{class})$ (which is hard)
- We assume $x_1, x_2, \ldots$ are independent given the class → so:

$$P(x_1, x_2, \ldots x_n \mid \text{class}) \approx P(x_1|\text{class}) \times P(x_2|\text{class}) \times \ldots \times P(x_n|\text{class})$$

- This simplifies computation dramatically

Decision Rule

- For a given example with features $x_1 \ldots x_n$, compute for each possible class C_k:

$\text{Score}_k = P(C_k) \times \Pi_i \, P(x_i \mid C_k)$

- Choose the class with the highest score as the predicted class

Example

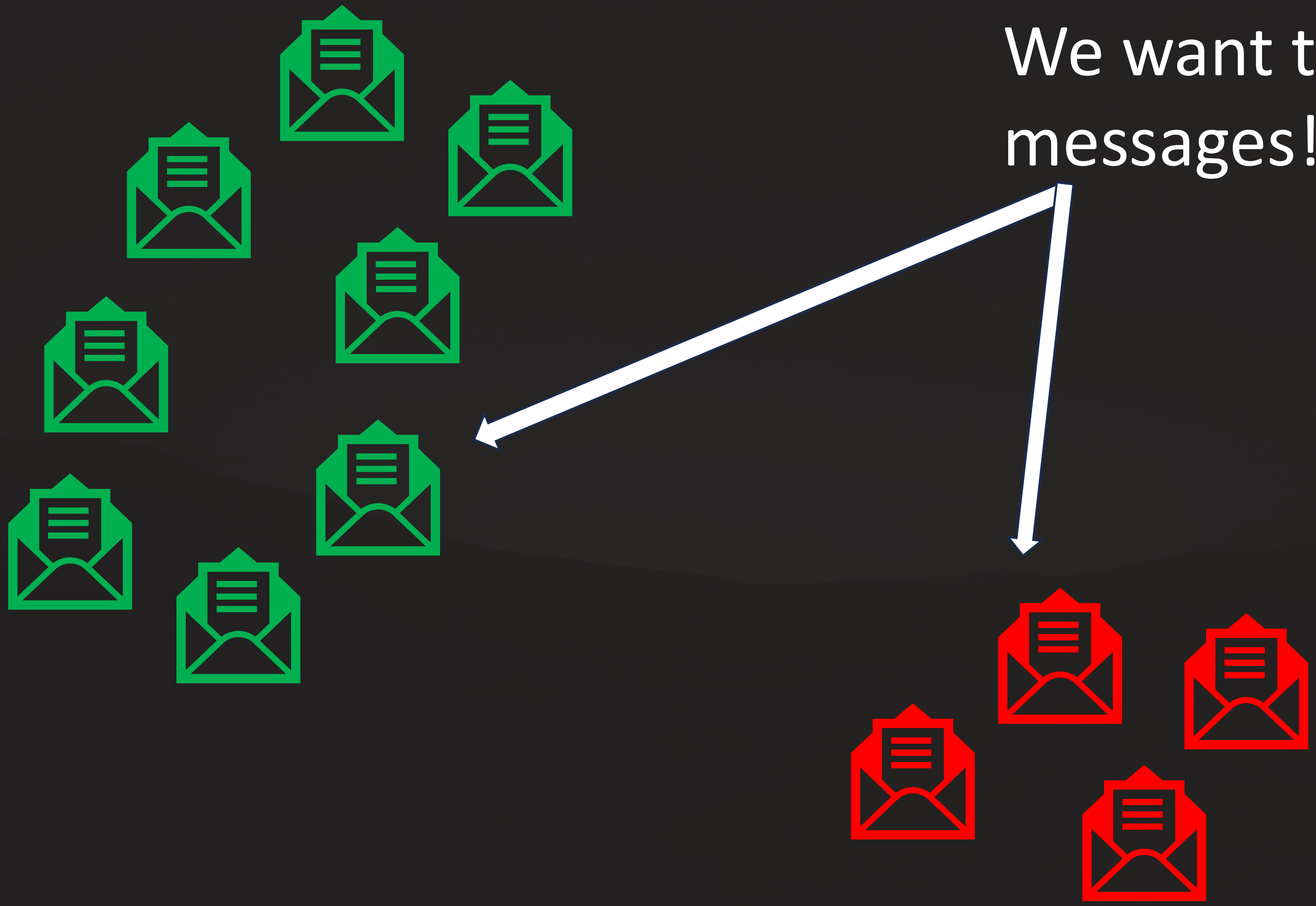
Imagine we received normal mails
from friends, colleagues & family

Example

We also receive some spam mails

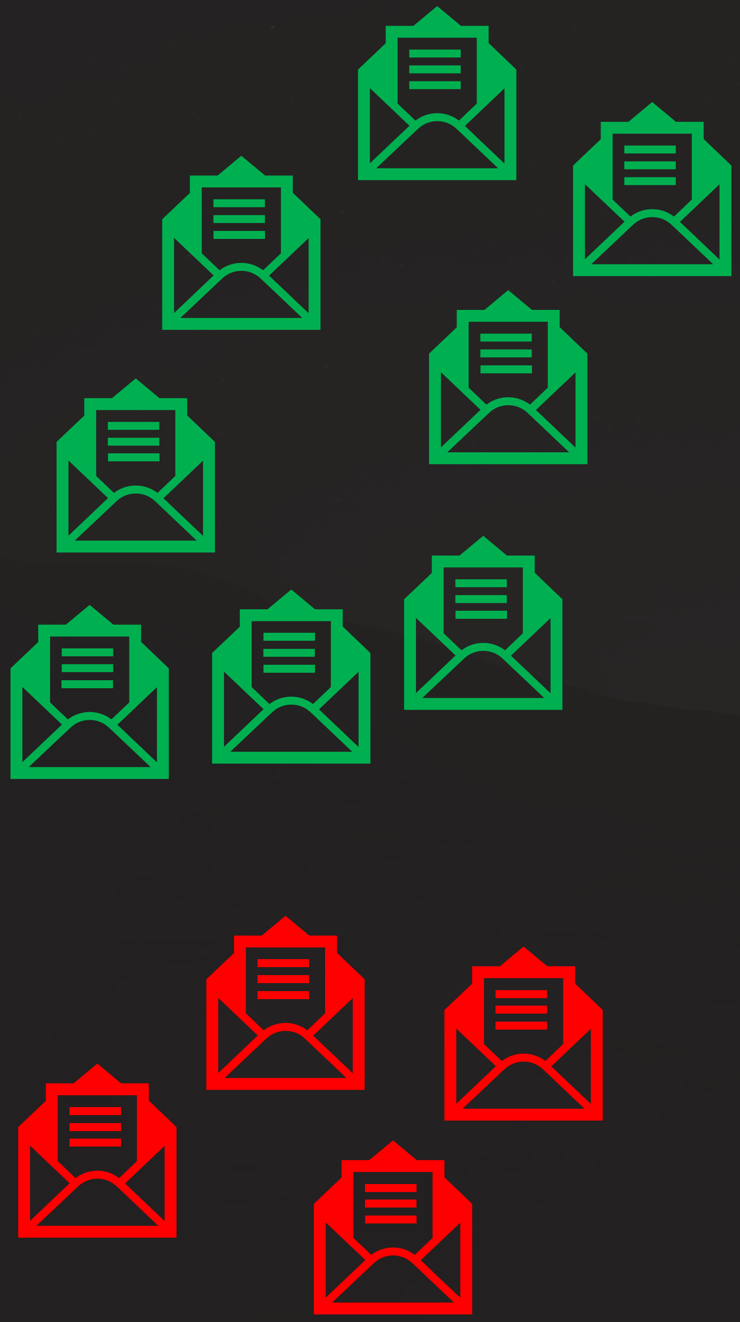
Example

We want to filter out the spam messages!



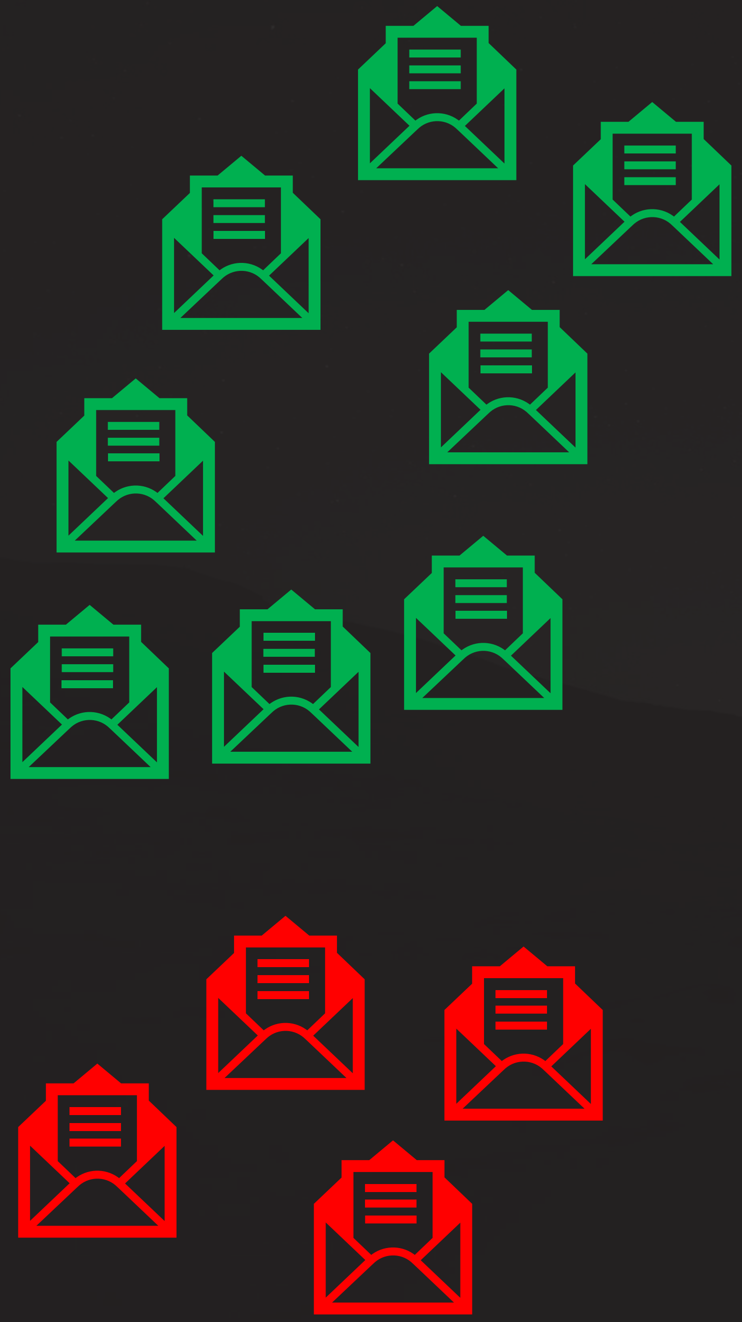
Example

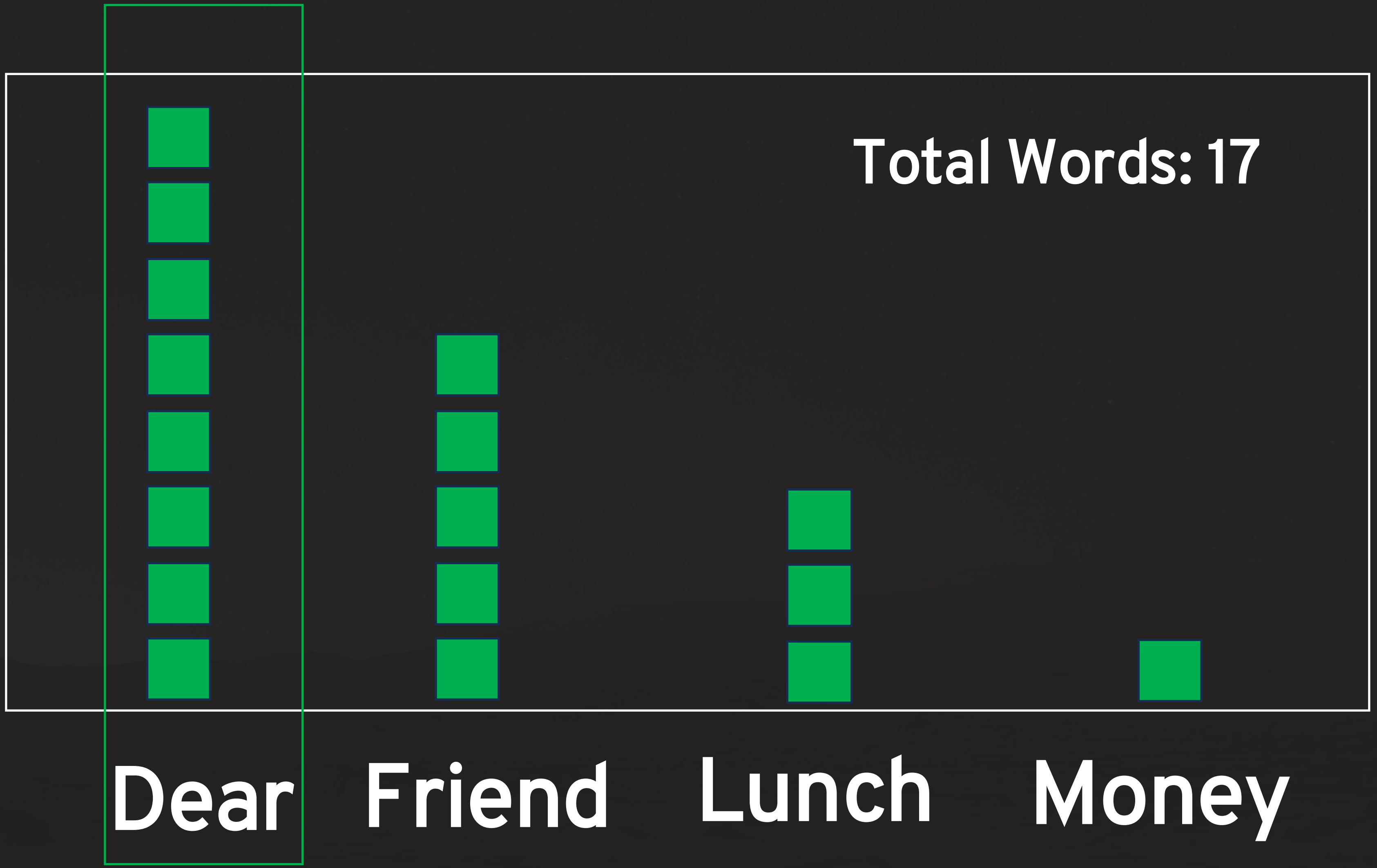
Let’s make a histogram of all the words that occur in the normal mails



Example

We can use the histogram to calculate the probabilities of seeing each word, given that it was in a normal mail.

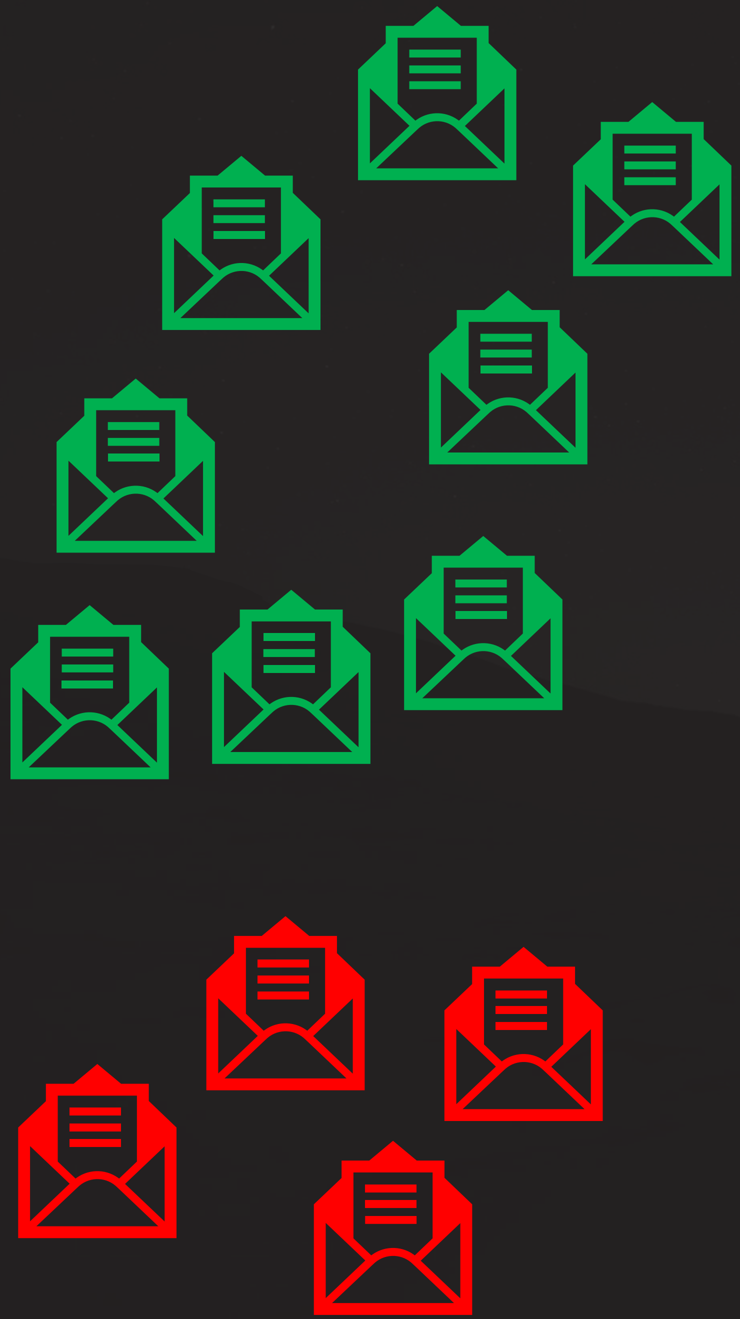


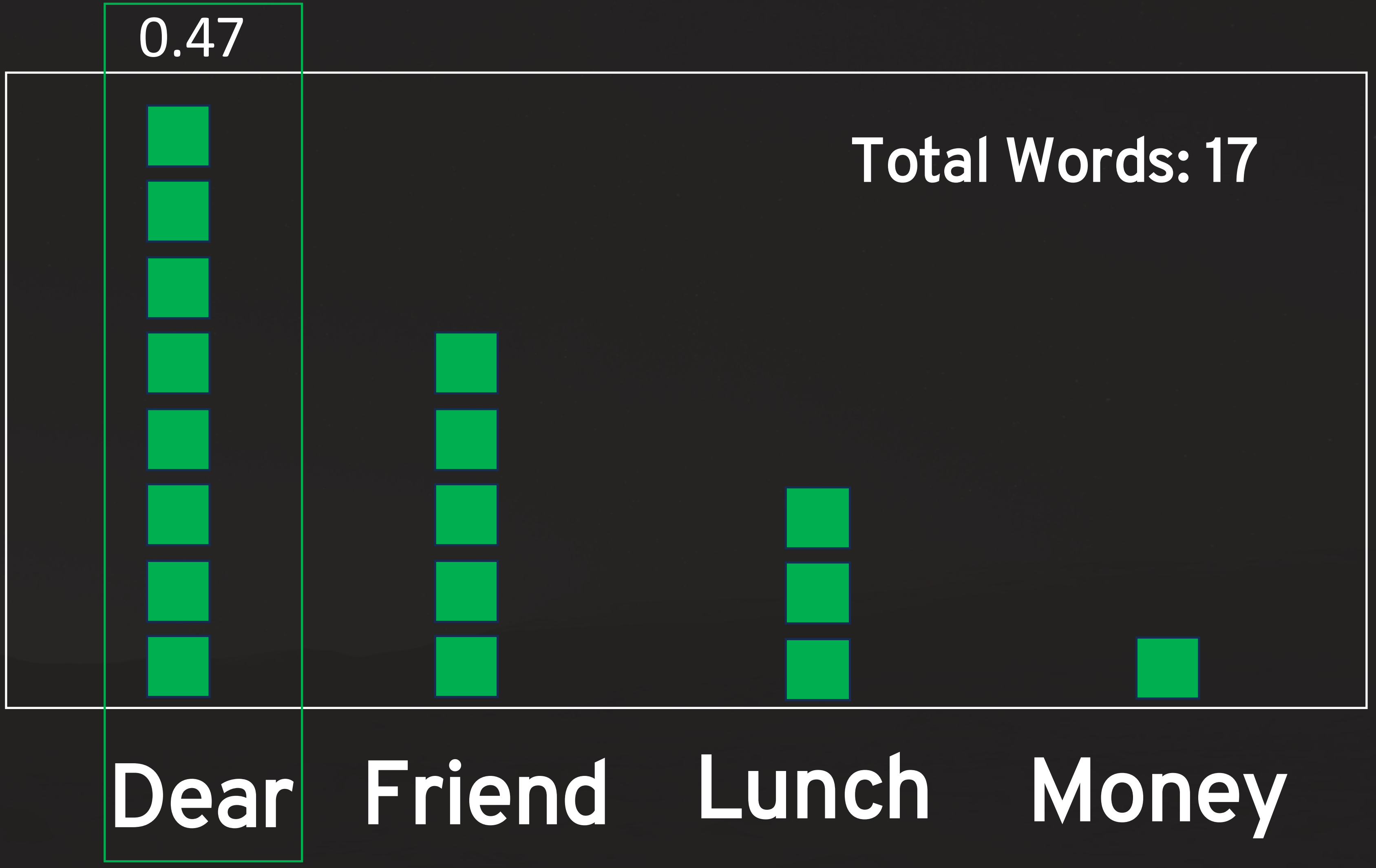


$$P(\text{Dear} \mid \text{Normal}) = \frac{8}{17} = 0.47$$

Example

We can use the histogram to calculate the probabilities of seeing each word, given that it was in a normal mail.

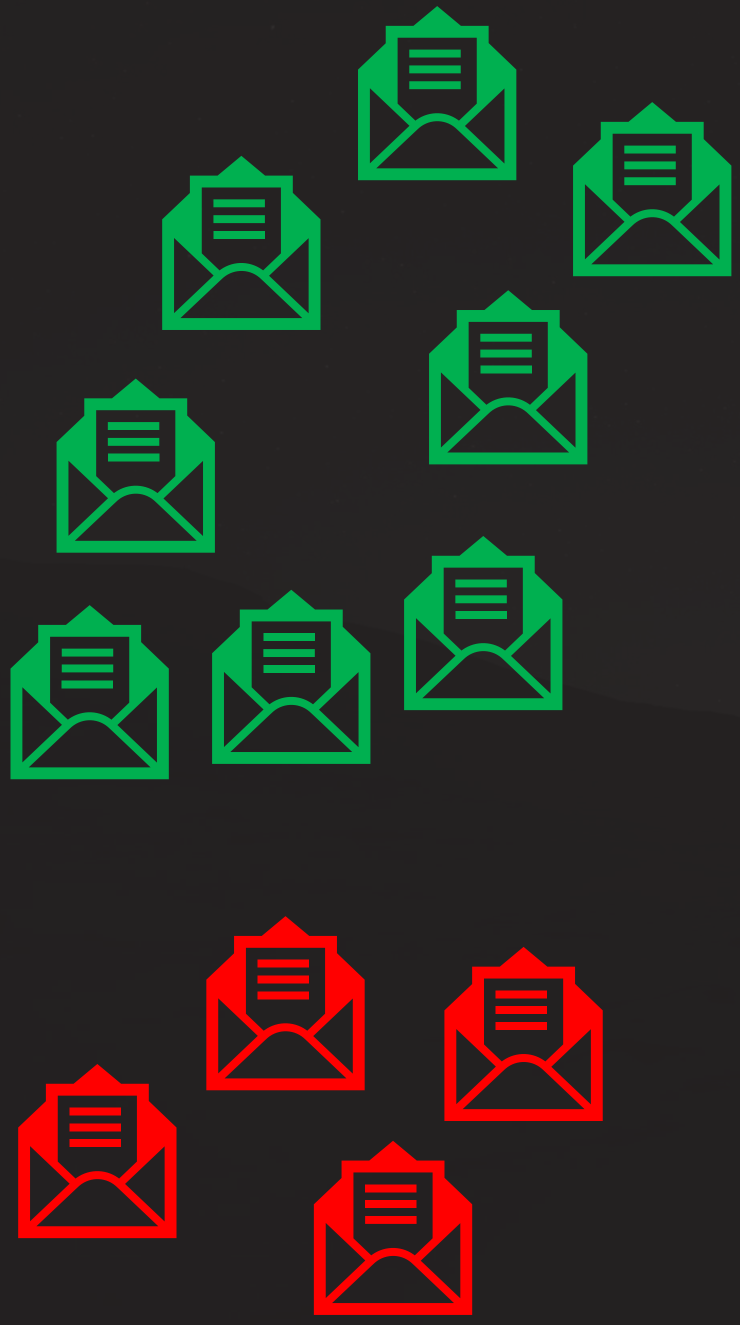


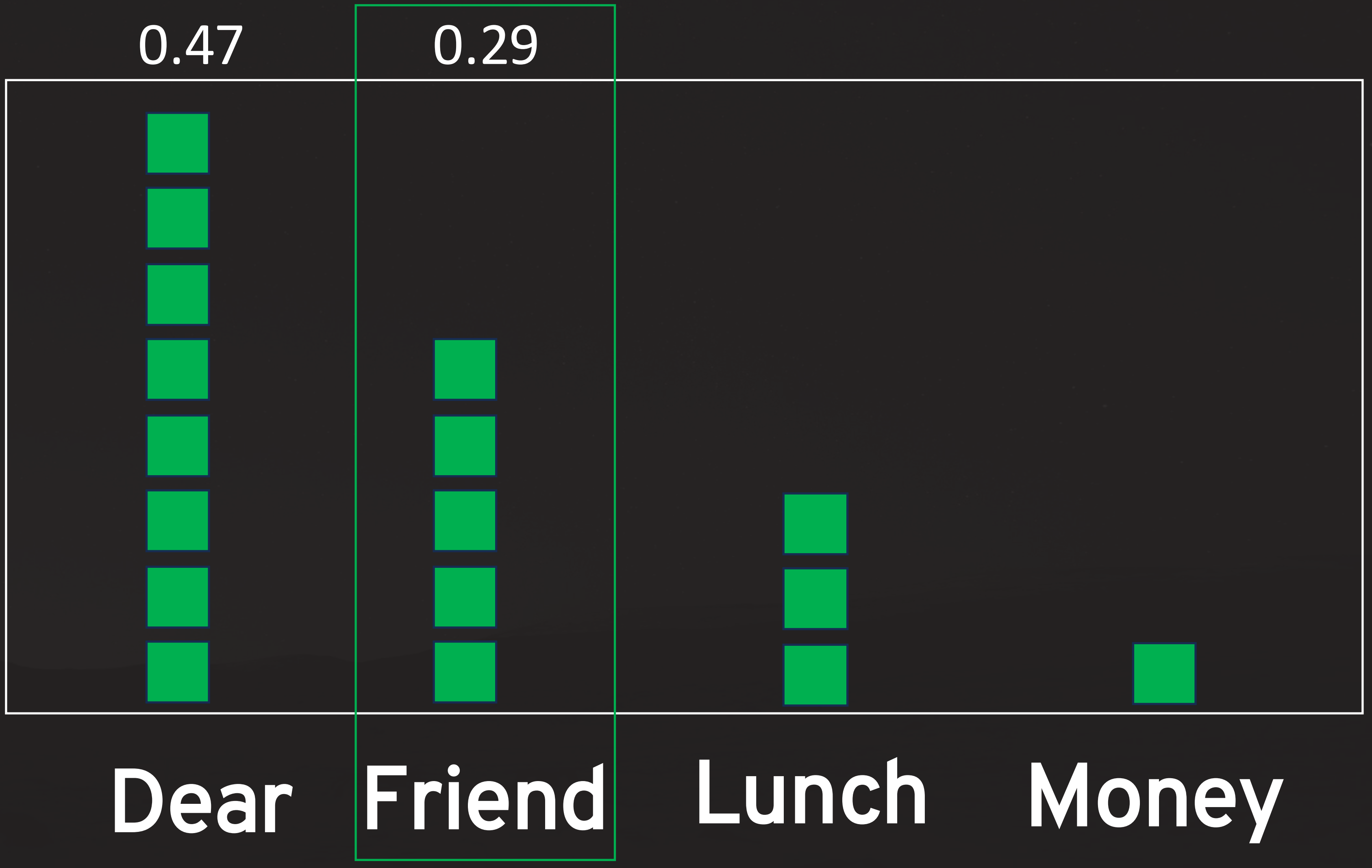


$$P(\text{Dear} \mid \text{Normal}) = \frac{8}{17} = 0.47$$

Example

Total Words: 17

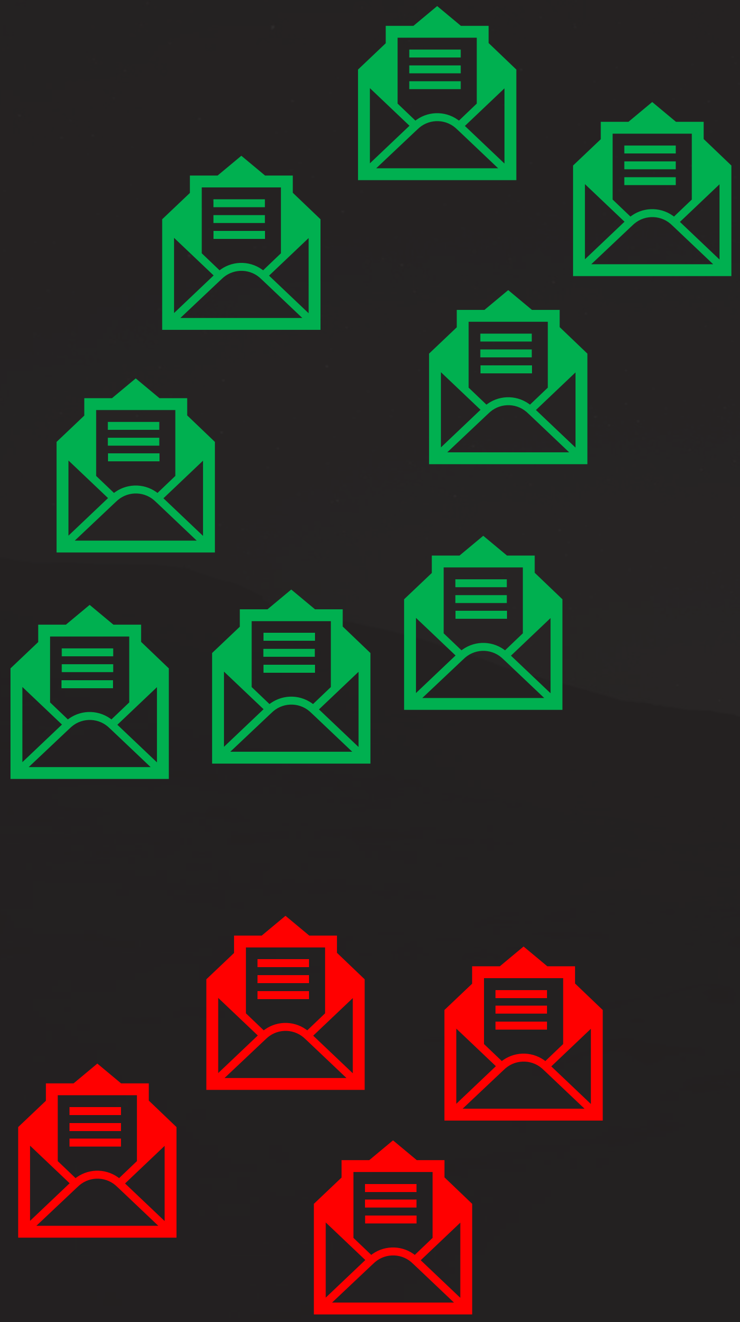


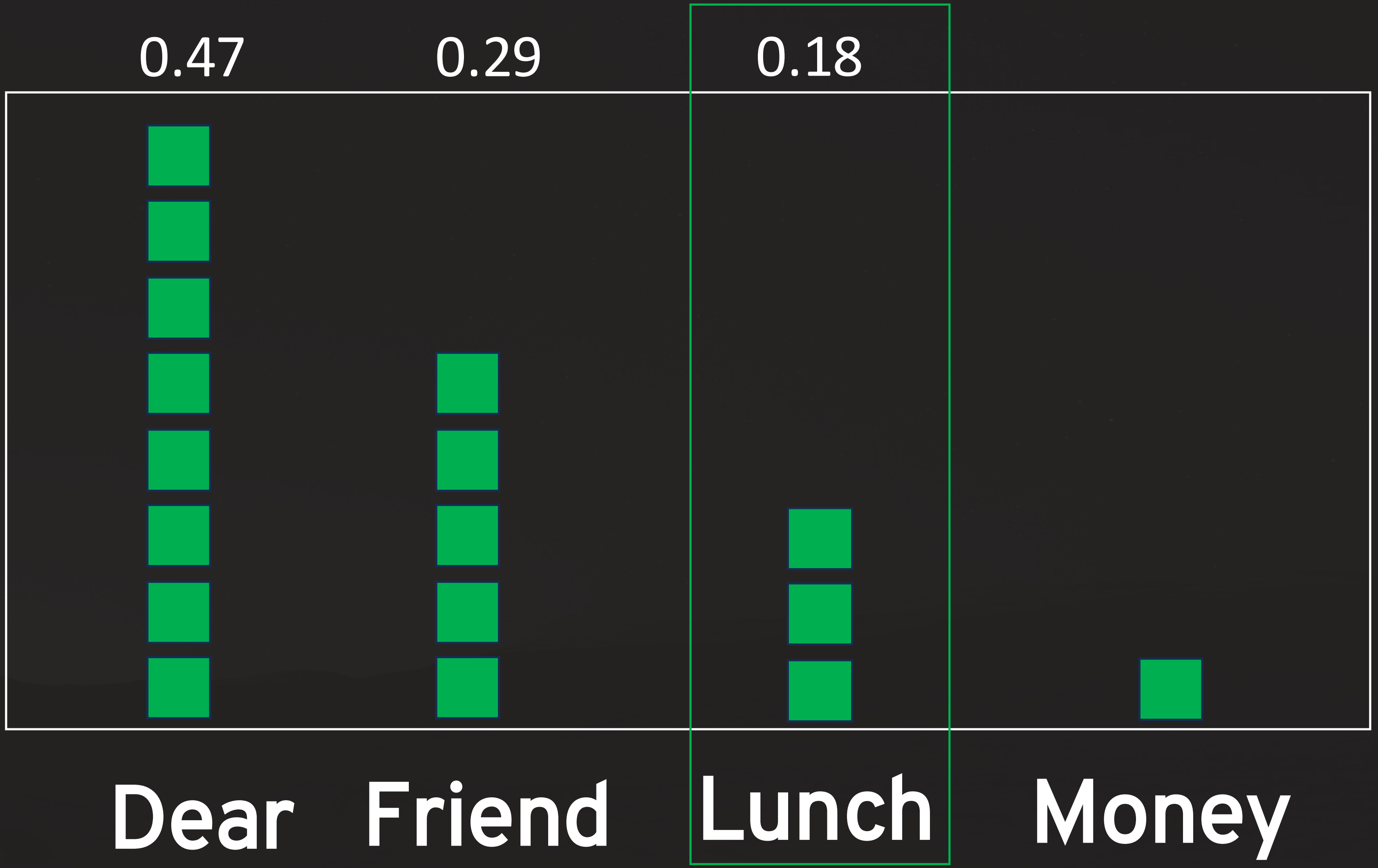


$$P(\text{Friend} \mid \text{Normal}) = \frac{5}{17} = 0.29$$

Example

Total Words: 17

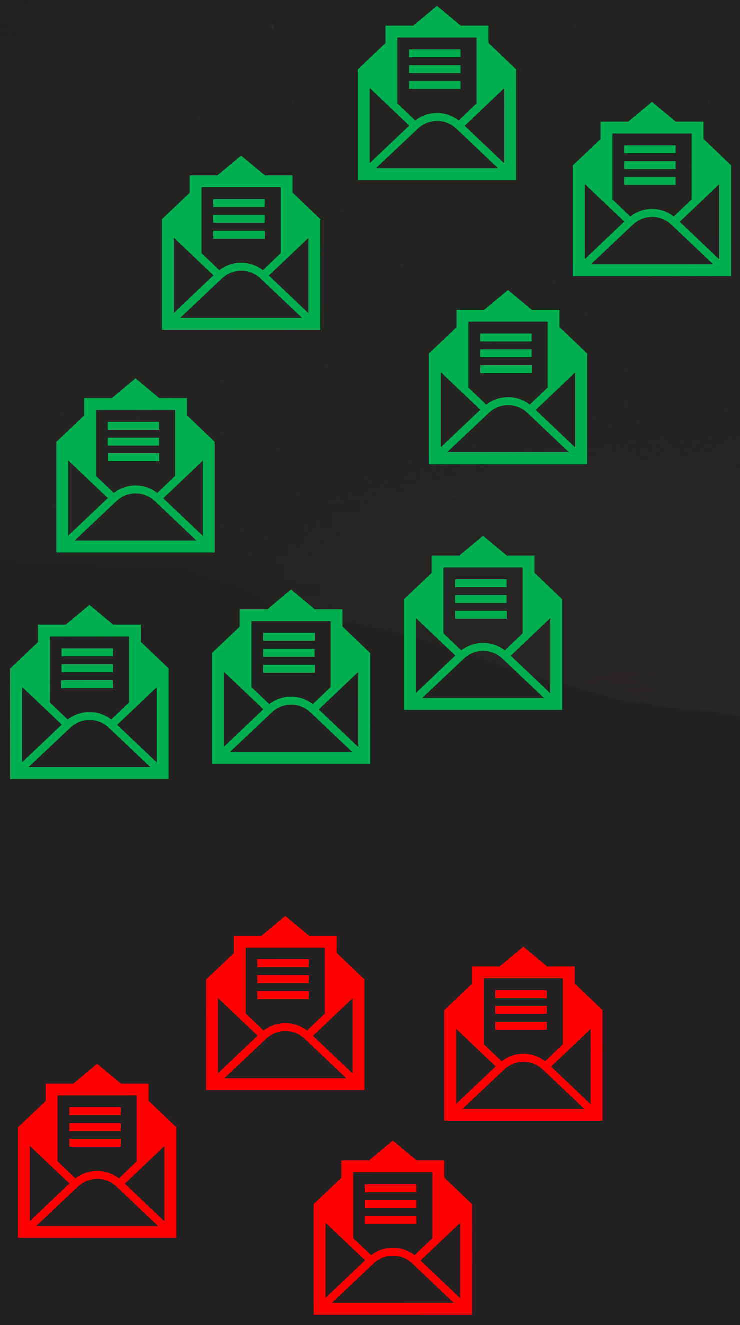


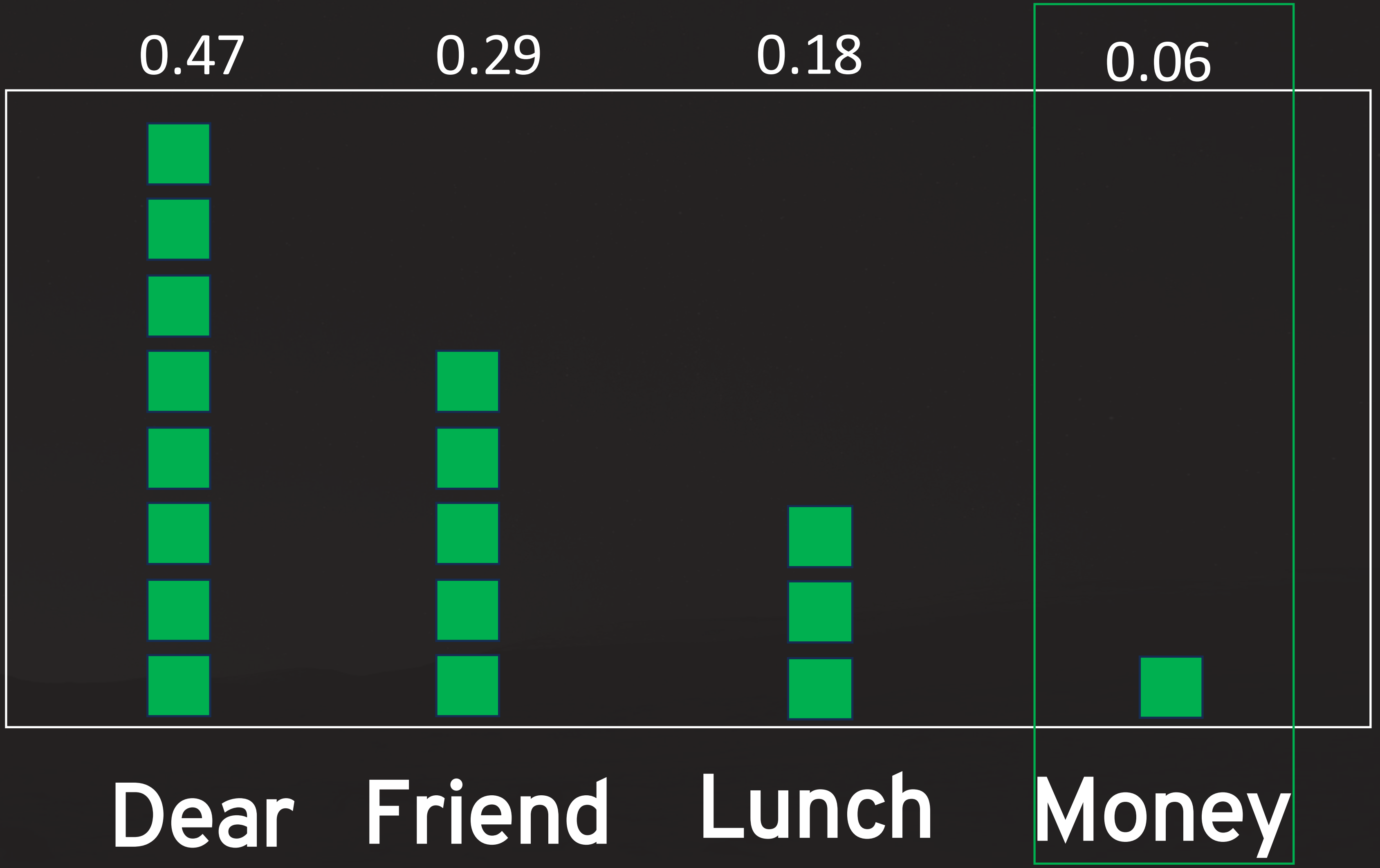


$$P(\text{Lunch} \mid \text{Normal}) = \frac{3}{17} = 0.18$$

Example

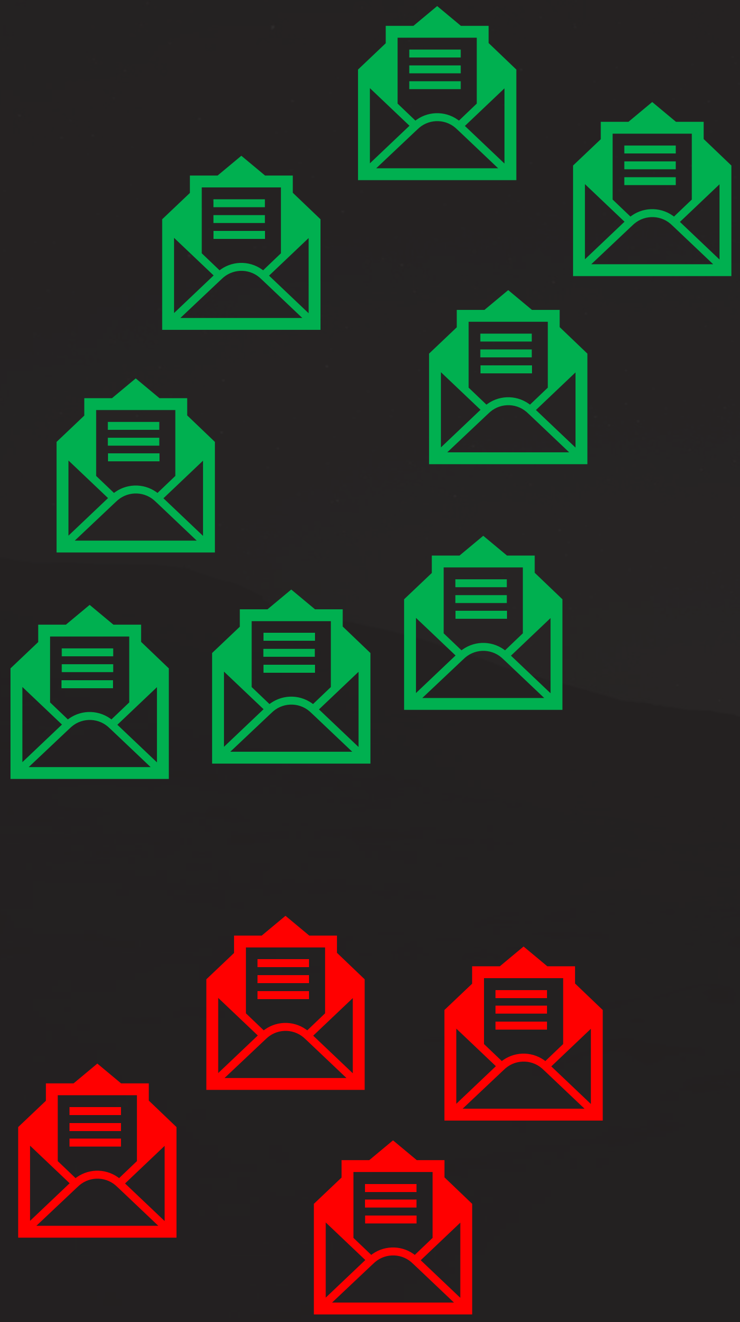
Total Words: 17

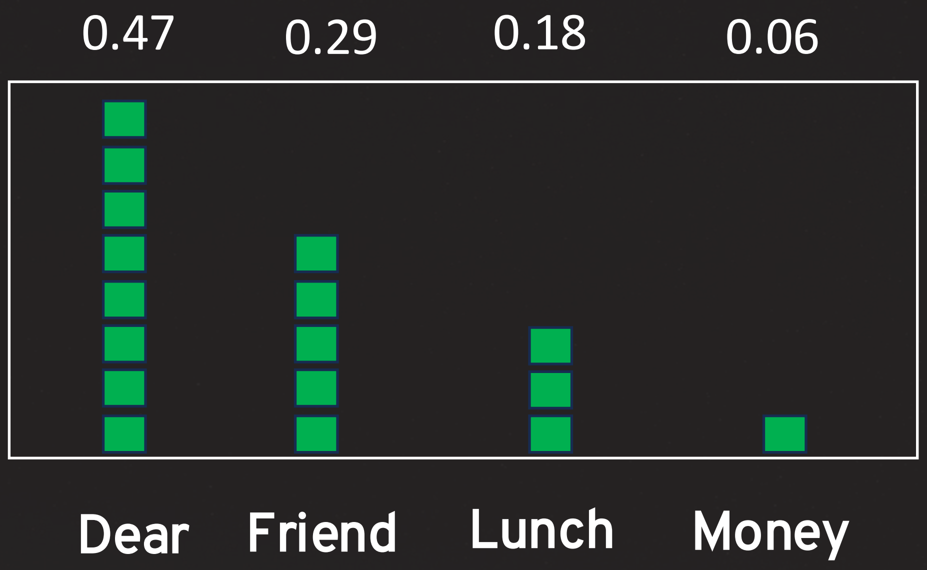




$$P(\text{Money} \mid \text{Normal}) = \frac{1}{17} = 0.06$$

Example

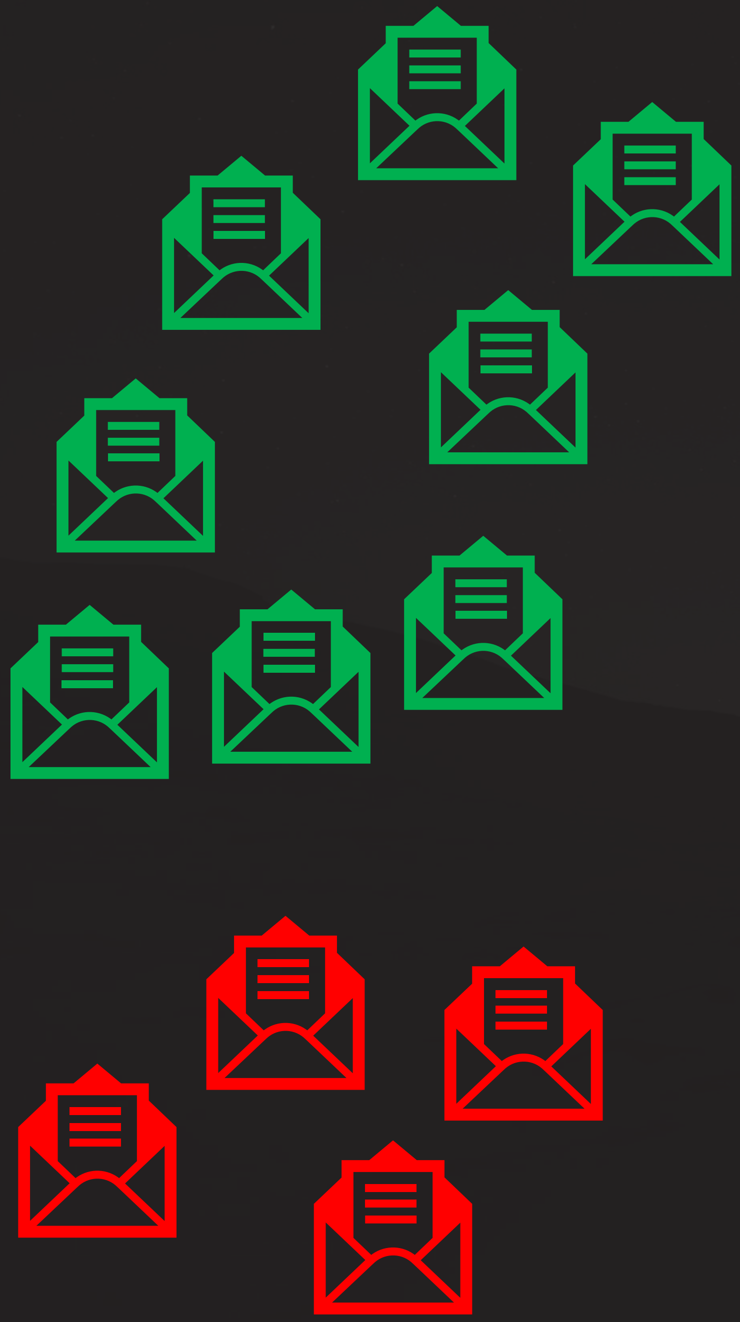


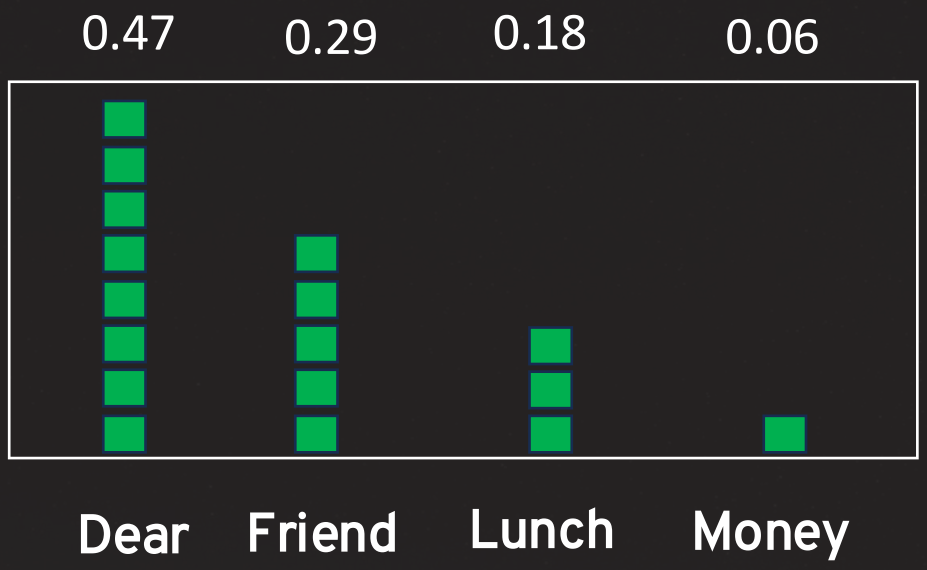


Total Words: 7

$$P(\text{Dear} \mid \text{Spam}) = \frac{2}{7} = 0.29$$

Example

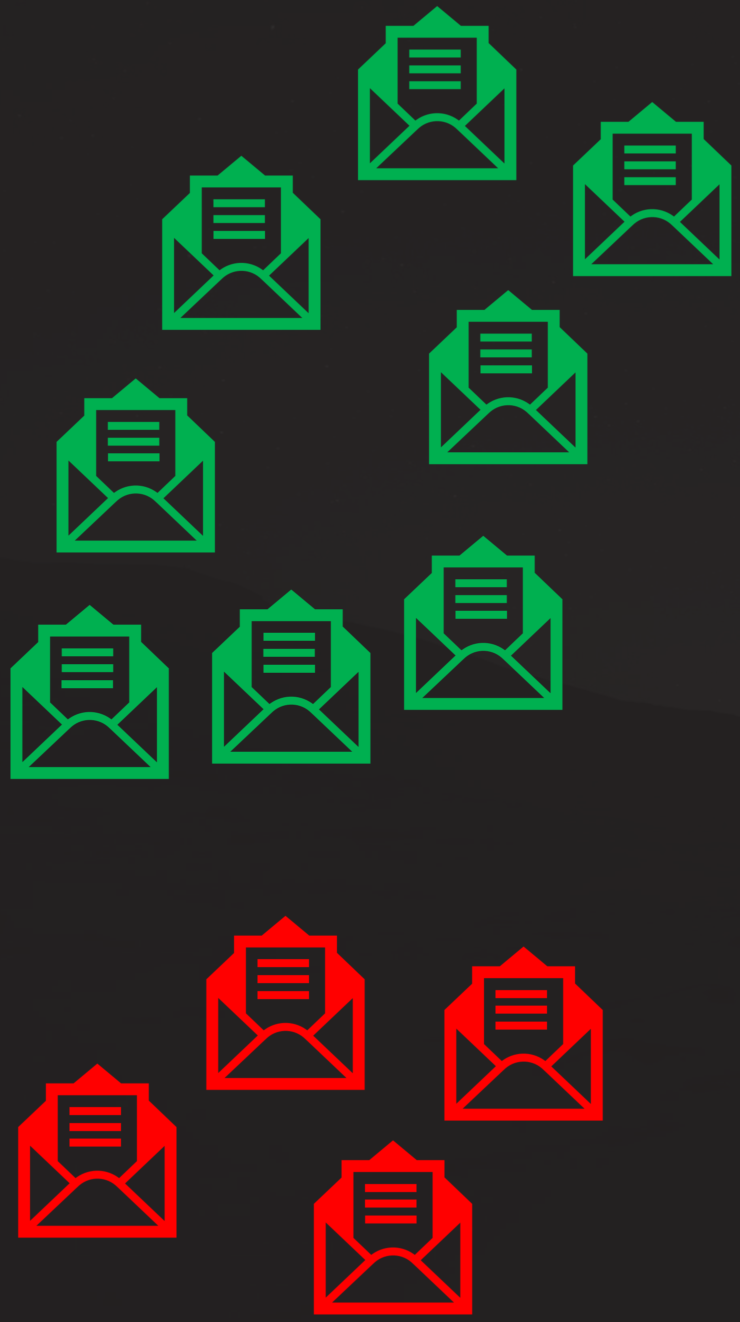


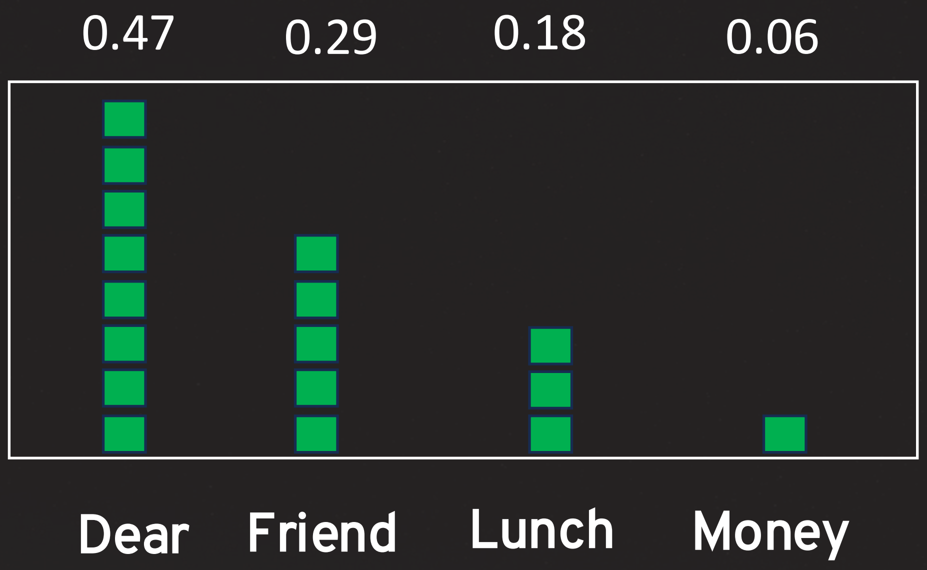


Total Words: 7

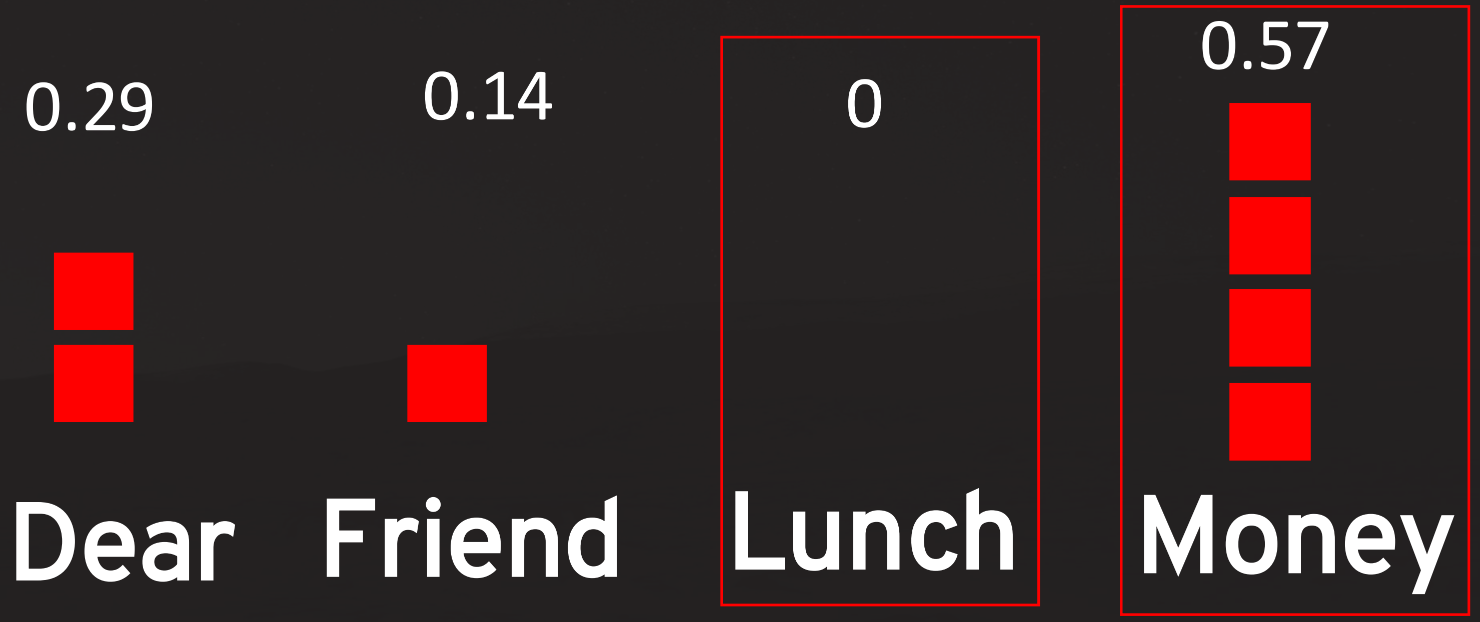
$$P(\text{Friend} \mid \text{Spam}) = \frac{1}{7} = 0.14$$

Example





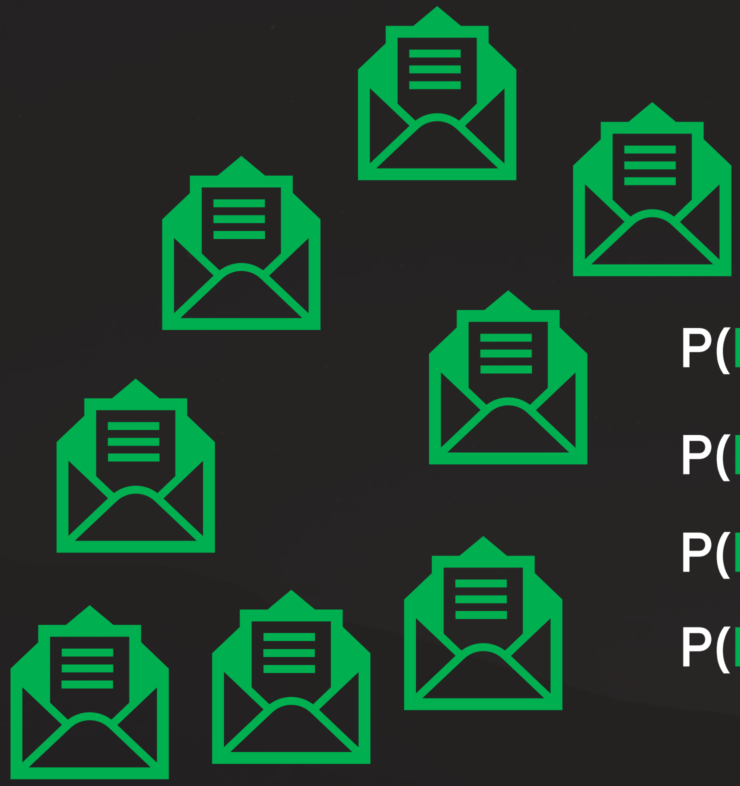
Total Words: 7



$$P(\text{Lunch} \mid \text{Spam}) = \frac{0}{7} = 0$$

$$P(\text{Money} \mid \text{Spam}) = \frac{4}{7} = 0.57$$

Example



$P(\text{Dear} \mid \text{Normal}) = 0.47$

$P(\text{Friend} \mid \text{Normal}) = 0.29$

$P(\text{Lunch} \mid \text{Normal}) = 0.18$

$P(\text{Money} \mid \text{Normal}) = 0.06$

$P(\text{Dear} \mid \text{Spam}) = 0.29$

$P(\text{Friend} \mid \text{Spam}) = 0.14$

$P(\text{Lunch} \mid \text{Spam}) = 0.00$

$P(\text{Money} \mid \text{Spam}) = 0.57$

Normal Message : Initial guess about the Probability that any message, regardless of what it says.

This guess can be any Probability that we want but a common guess is estimated from the training data

$$P(N) = \frac{8}{8+4} = 0.67$$

(8 of the 12 mails are normal mails)

Example

$P(N) = 0.67$

P(Dear | Normal) = 0.47

P(Friend | Normal) = 0.29

P(Lunch | Normal) = 0.18

P(Money | Normal) = 0.06

$P(S) = 0.33$

P(Dear | Spam) = 0.29

P(Friend | Spam) = 0.14

P(Lunch | Spam) = 0.00

P(Money | Spam) = 0.57

Normal Message/ Prior Probability :

Initial guess about the Probability that any message, regardless of what it says.

This guess can be any Probability that we want but a common guess is estimated from the training data

$$P(N) = \frac{8}{8+4} = 0.67$$

(8 of the 12 mails are normal mails)

$$P(S) = \frac{4}{4+8} = 0.33$$

(4of the 12 mails are spam mails)

Example

$p(N) = 0.67$

$p(\text{Dear} \mid \text{Normal}) = 0.47$

$p(\text{Friend} \mid \text{Normal}) = 0.29$

$p(\text{Lunch} \mid \text{Normal}) = 0.18$

$P(\text{Money} \mid \text{Normal}) = 0.06$

$p(S) = 0.33$

$p(\text{Dear} \mid \text{Spam}) = 0.29$

$p(\text{Friend} \mid \text{Spam}) = 0.14$

$p(\text{Lunch} \mid \text{Spam}) = 0.00$

$p(\text{Money} \mid \text{Spam}) = 0.57$

"Dear Friend"

?

N S

$p(N) * p(\text{Dear} \mid N) * p(\text{Friend} \mid N)$

$0.67 * 0.47 * 0.29 = 0.09$

$p(N \mid \text{Dear Friend}) \propto 0.09$

$p(S) * p(\text{Dear} \mid S) * p(\text{Friend} \mid S)$

$0.33 * 0.29 * 0.14 = 0.01$

$p(S \mid \text{Dear Friend}) \propto 0.01$

Example

p(N) = 0.67

p(Dear | Normal) = 0.47

p(Friend | Normal) = 0.29

p(Lunch | Normal) = 0.18

P(Money | Normal) = 0.06

p(S) = 0.33

p(Dear | Spam) = 0.29

p(Friend | Spam) = 0.14

p(Lunch | Spam) = 0.00

p(Money | Spam) = 0.57

"Dear Friend"

?

N S

p(N) * p(Dear | N) * p(Friend | N)

0.67 * 0.47 * 0.29 = 0.09

p(N | Dear Friend) ∝ 0.09

p(S) * p(Dear | S) * p(Friend | S)

0.33 * 0.29 * 0.14 = 0.01

p(S | Dear Friend) ∝ 0.01

Since 0.09 > 0.01

We decide **DEAR FRIEND**

is a **Normal** *Mail*

Example

"Lunch Money Money Money" ? N S

$p(N) = 0.67$

$p(\text{Dear} \mid \text{Normal}) = 0.47$

$p(\text{Friend} \mid \text{Normal}) = 0.29$

$p(\text{Lunch} \mid \text{Normal}) = 0.18$

$P(\text{Money} \mid \text{Normal}) = 0.06$

$p(S) = 0.33$

$p(\text{Dear} \mid \text{Spam}) = 0.29$

$p(\text{Friend} \mid \text{Spam}) = 0.14$

$p(\text{Lunch} \mid \text{Spam}) = 0.00$

$p(\text{Money} \mid \text{Spam}) = 0.57$

$$p(N) * p(\text{Lunch} \mid N) * p(\text{Money} \mid N)^4$$

$$0.67 * 0.18 * 0.06^4 = 0.000002$$

$$p(N \mid \text{Lunch M M M}) \propto 0.000002$$

$$p(S) * p(\text{Lunch} \mid S) * p(\text{Money} \mid S)^4$$

$$0.67 * 0.00 * 0.57^4 = 0.00$$

$$p(S \mid \text{Lunch M M M}) \propto 0.00$$

NB will always classify the mails with LUNCH in them as NORMAL MAIL!!!

Example

$p(N) = 0.67$

$p(Dear \mid Normal) = 0.47$

$p(Friend \mid Normal) = 0.29$

$p(Lunch \mid Normal) = 0.18$

$P(Money \mid Normal) = 0.06$

$p(S) = 0.33$

$p(Dear \mid Spam) = 0.29$

$p(Friend \mid Spam) = 0.14$

$p(Lunch \mid Spam) = 0.00$

$p(Money \mid Spam) = 0.57$

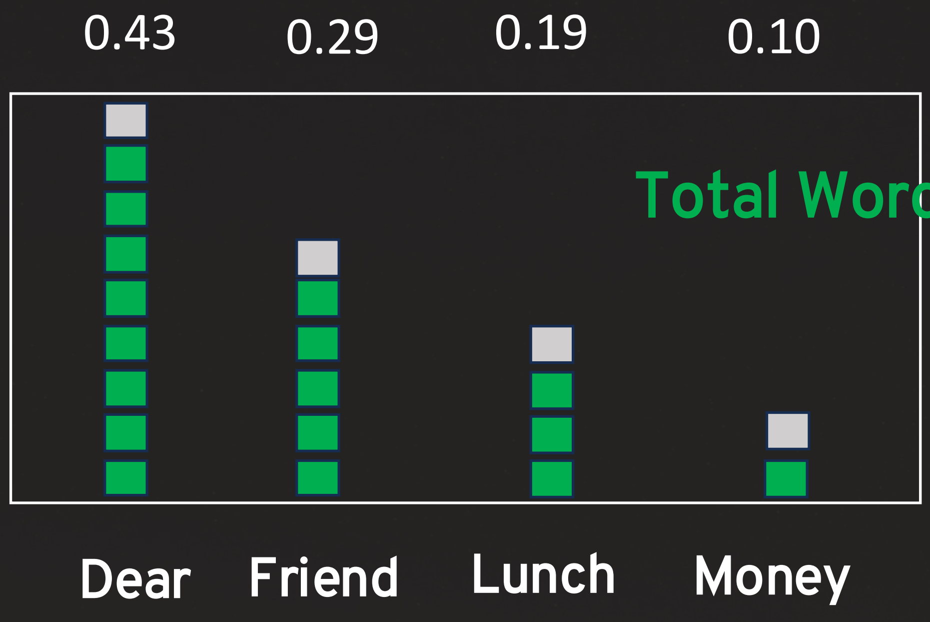
"Lunch Money Money Money" $\to$ N ? S

$$p(N) * p(Lunch \mid N) * p(Money \mid N)^4$$
$$0.67 * 0.18 * 0.06^4 = 0.000002$$
$$p(N \mid Lunch\ M\ M\ M) \propto 0.000002$$

$$p(S) * p(Lunch \mid S) * p(Money \mid S)^4$$
$$0.67 * 0.00 * 0.57^4 = 0.00$$
$$p(S \mid Lunch\ M\ M\ M) \propto 0.00$$

NB will always classify the mails with LUNCH in them as NORMAL MAIL!!!

Example



Total Words: 17 + 4

$$P(\text{Dear} \mid \text{Normal}) = \frac{9}{17+4} = 0.43$$

$p(N) = 0.67$

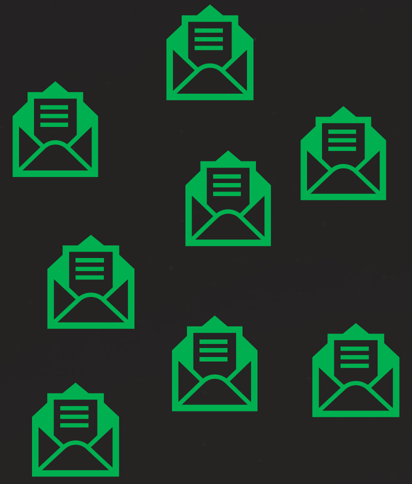
Total Words: 7 +4

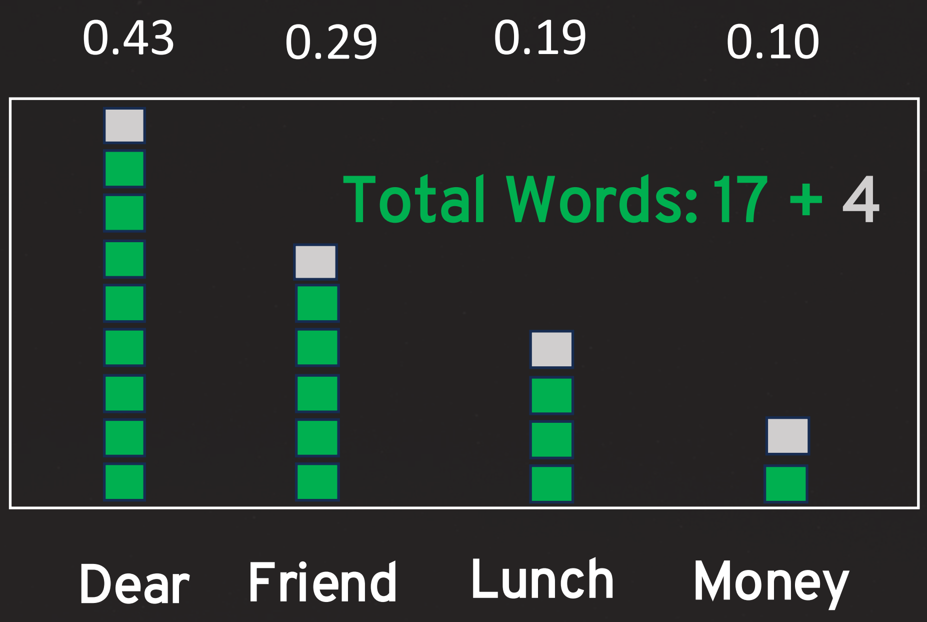
$$P(\text{Lunch} \mid \text{Spam}) = \frac{1}{7+4} = 0.09$$

$p(S) = 0.33$

Example

“Lunch Money Money Money Money” ? N / S





p(N) = 0.67

$$p(N) * p(Lunch \mid N) * p(Money \mid N)^4$$

$$0.67 * 0.19 * 0.10^4 = 0.000012$$

$$p(N \mid Lunch\ M\ M\ M) \propto 0.00001$$

p(S) = 0.33

$$p(S) * p(Lunch \mid S) * p(Money \mid S)^4$$

$$0.33 * 0.09 * 0.45 = 0.00122$$

$$p(S \mid Lunch\ M\ M\ M) \propto 0.00122$$

$$p(N \mid Lunch\ M\ M\ M) < p(S \mid Lunch\ M\ M\ M)$$

“Lunch Money Money Money Money” classified as SPAM

Strengths & Limitations

Understanding both the power and constraints of Naive Bayes helps you deploy it effectively in the right scenarios.

Strengths

Lightning Speed

Extremely fast training and prediction make it ideal for real-time applications and large-scale deployments.

High-Dimensional Excellence

Performs remarkably well with high-dimensional data where other algorithms struggle with the curse of dimensionality.

Data Efficiency

Requires relatively small training datasets to achieve good performance, making it accessible for projects with limited data.

Limitations

Independence Assumption

Assumes features are independent, which is rarely true in practice. Correlated features can reduce accuracy.

Probability Estimates

While classifications are often accurate, the output probabilities themselves are not always reliable or well-calibrated.

Representation Sensitivity

Performance depends heavily on how data is represented and preprocessed. Feature engineering is critical.

Naive Bayes in Practice: Implementation & Impact

Email Spam Filters

The backbone of modern email security, protecting billions of users from malicious content and unwanted messages every single day.

Sentiment Analysis

Powers customer feedback analysis, social media monitoring, and brand reputation management across industries.

Document Categorization

Automatically organizes and routes documents, news articles, and content at scale for media companies and enterprises.

Easy Implementation

Modern libraries make Naive Bayes accessible to developers at all skill levels. Just a few lines of Python code unlock powerful classification capabilities.

```
from sklearn.naive_bayes import GaussianNB, MultinomialNB#
Gaussian for continuous featuresmodel =
GaussianNB()model.fit(X_train, y_train)# Multinomial for text
classificationtext_model =
MultinomialNB()text_model.fit(word_counts, labels)
```