# Introduction to Data Science

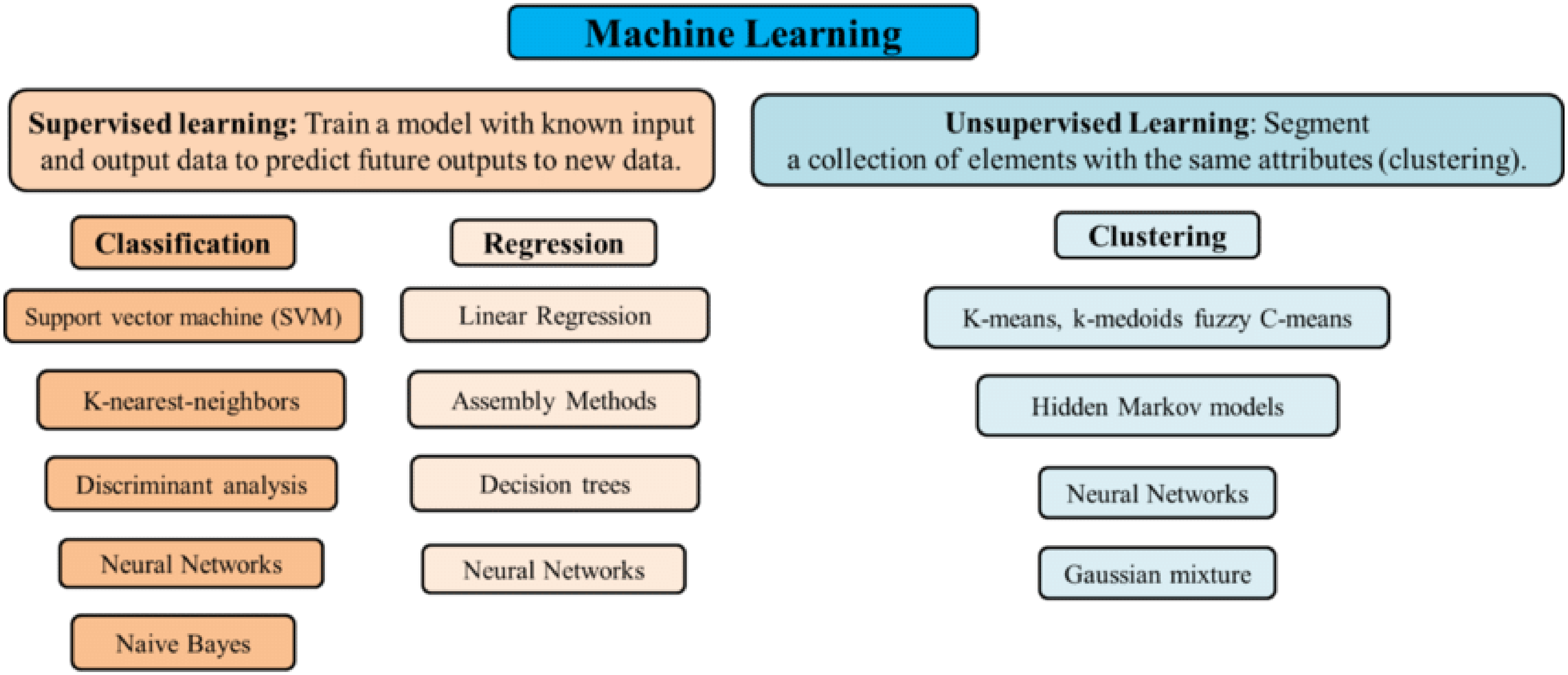
Dr. Irfan Yousuf
Department of Computer Science (New Campus)
UET, Lahore

(Lecture # 21; November 23, 2022)

# Outline

- Confusion Matrix
- Handling categorical Data
    - One Hot Encoding
- Spam Filtering

# Machine Learning Algorithms



Machine Learning

**Supervised learning:** Train a model with known input and output data to predict future outputs to new data.

**Unsupervised Learning**: Segment a collection of elements with the same attributes (clustering).

**Classification**

- Support vector machine (SVM)
- K-nearest-neighbors
- Discriminant analysis
- Neural Networks
- Naive Bayes

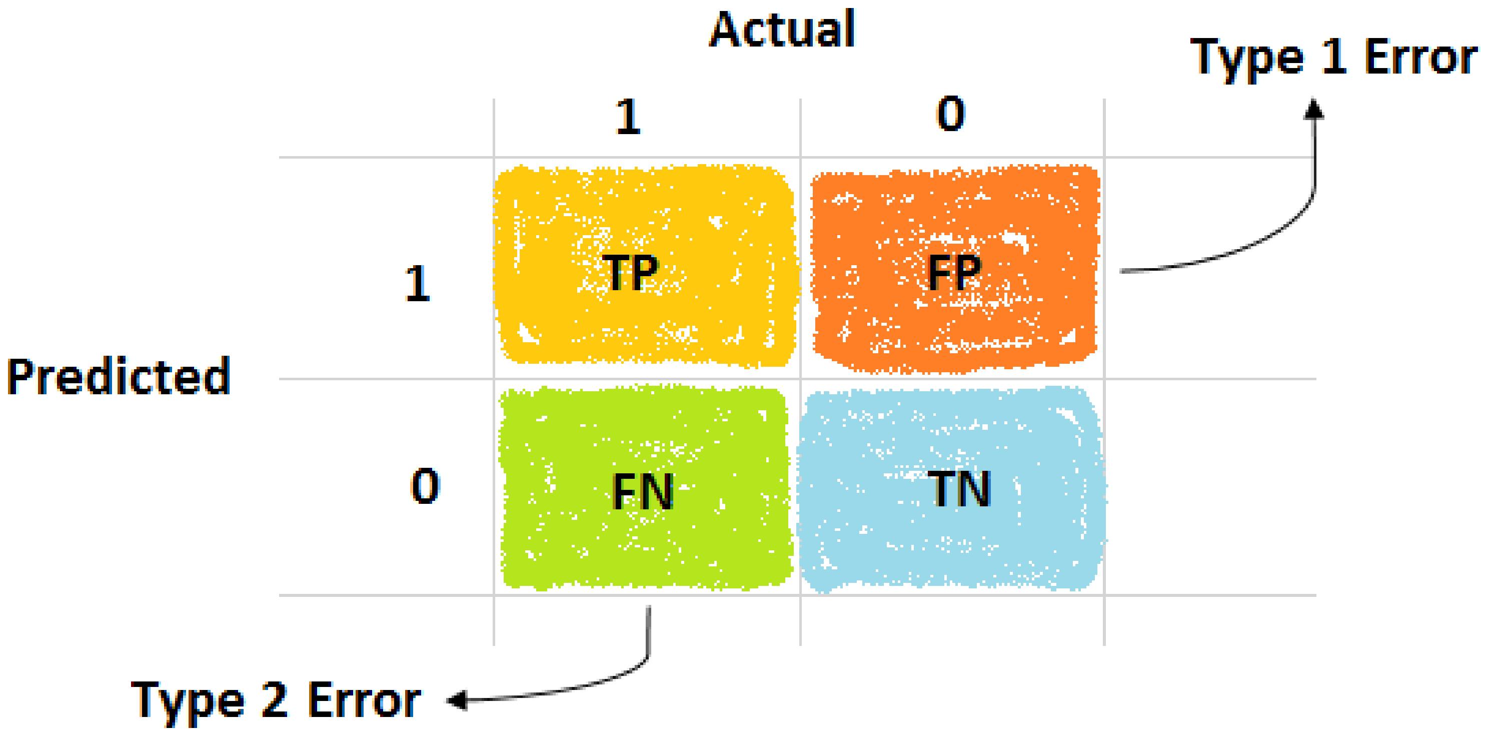
**Regression**

- Linear Regression
- Assembly Methods
- Decision trees
- Neural Networks

**Clustering**

- K-means, k-medoids fuzzy C-means
- Hidden Markov models
- Neural Networks
- Gaussian mixture

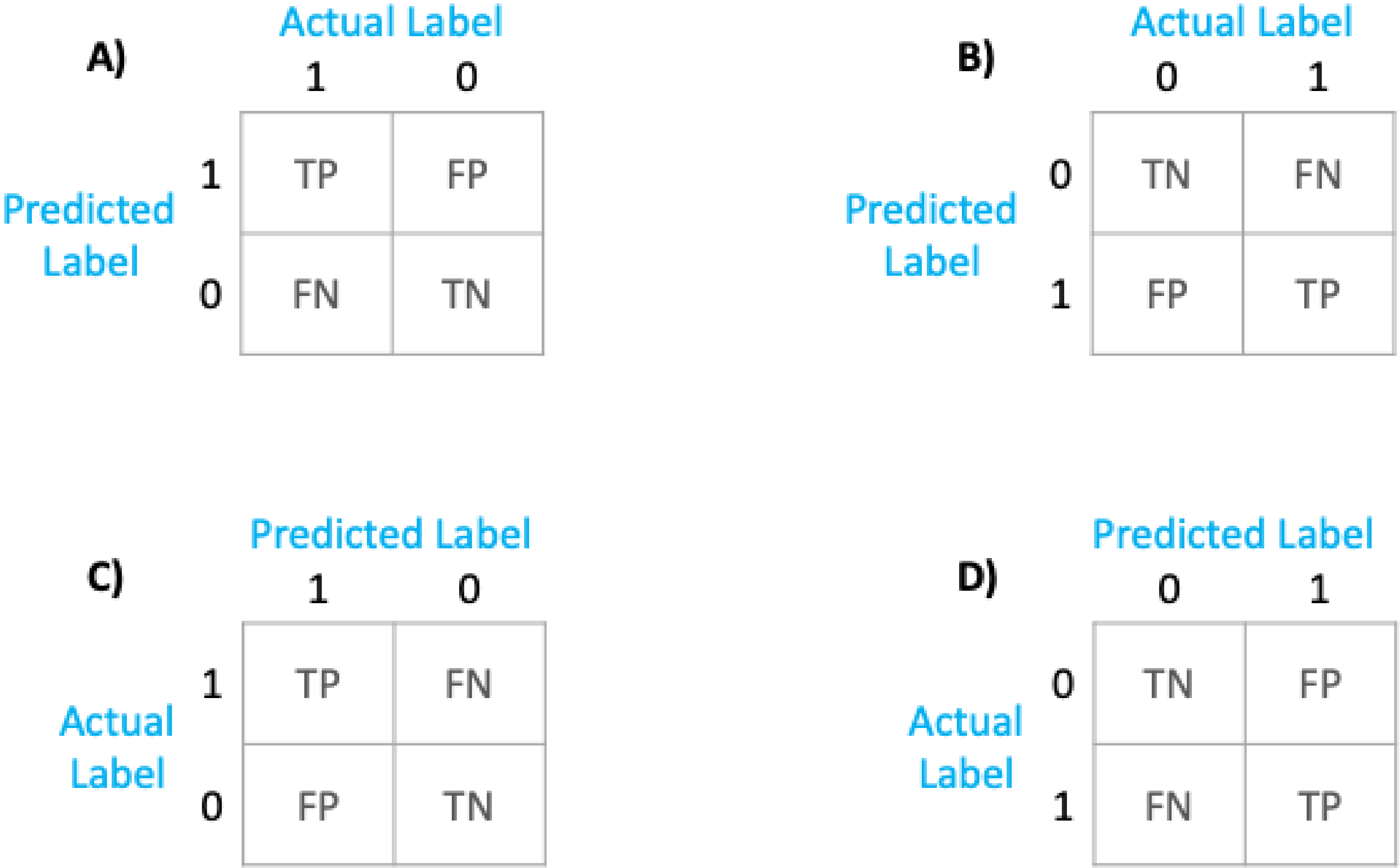
# Confusion Matrix



|  |  | Actual |  |
|---|---|---|---|
|  |  | 1 | 0 |
| Predicted | 1 | TP | FP |
|  | 0 | FN | TN |

Type 1 Error

Type 2 Error



A)

| | | Actual Label | |
|---|---|---|---|
| | | 1 | 0 |
| Predicted Label | 1 | TP | FP |
| | 0 | FN | TN |

B)

| | | Actual Label | |
|---|---|---|---|
| | | 0 | 1 |
| Predicted Label | 0 | TN | FN |
| | 1 | FP | TP |

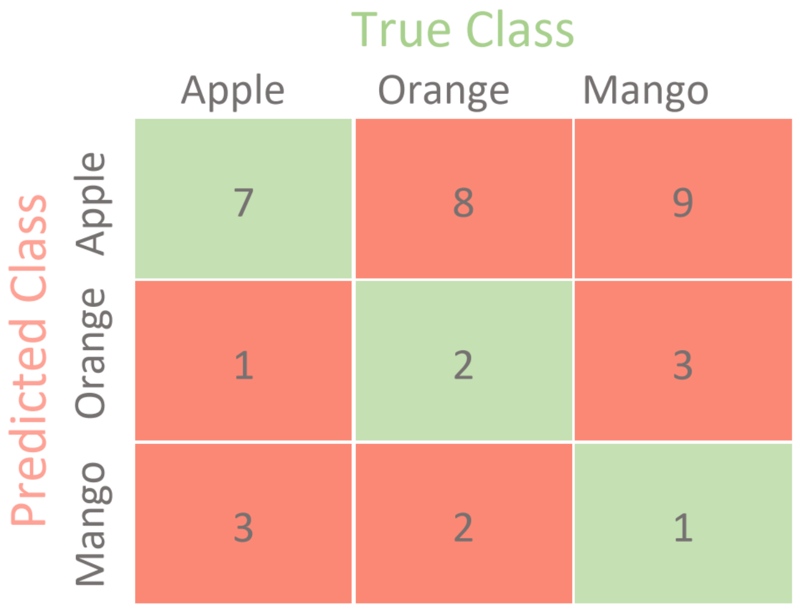
C)

| | | Predicted Label | |
|---|---|---|---|
| | | 1 | 0 |
| Actual Label | 1 | TP | FN |
| | 0 | FP | TN |

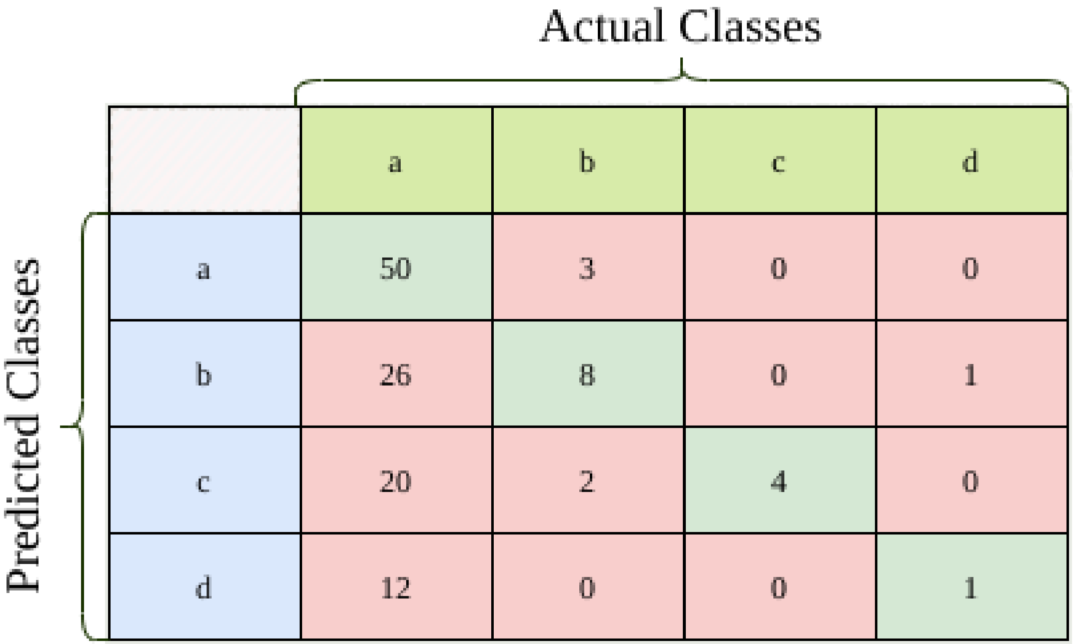
D)

| | | Predicted Label | |
|---|---|---|---|
| | | 0 | 1 |
| Actual Label | 0 | TN | FP |
| | 1 | FN | TP |

# Confusion Matrix for Multiclassification



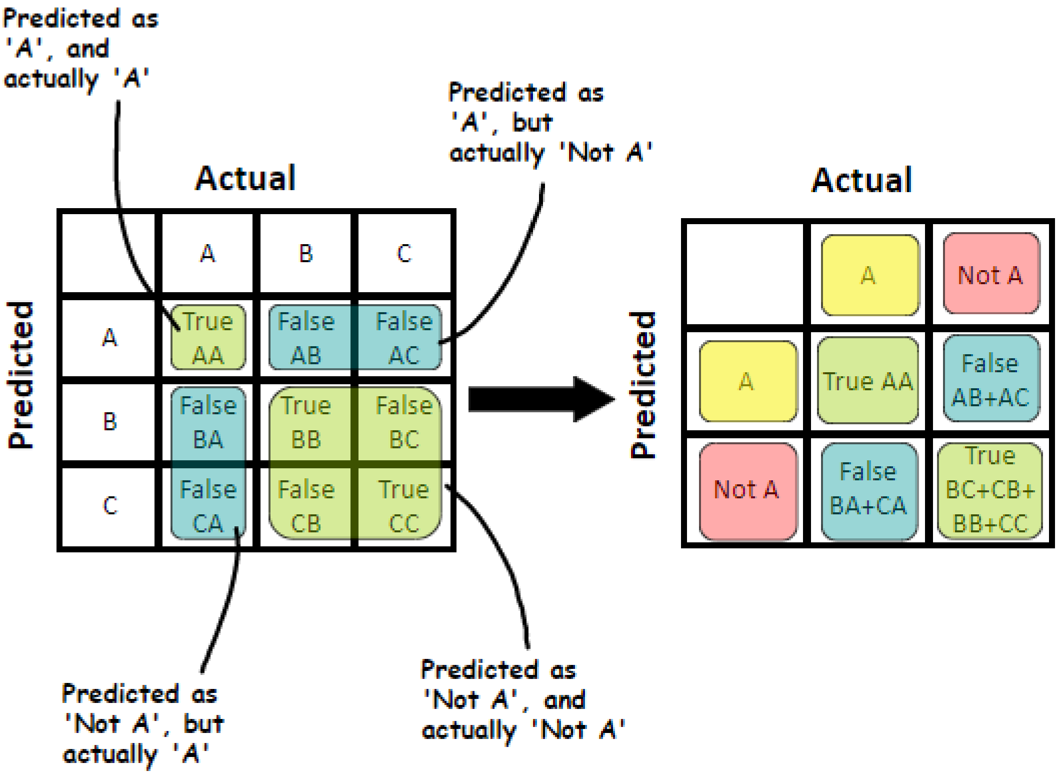
| | | True Class | | |
|---|---|---|---|---|
| | | Apple | Orange | Mango |
| Predicted Class | Apple | 7 | 8 | 9 |
| | Orange | 1 | 2 | 3 |
| | Mango | 3 | 2 | 1 |



| | | Actual Classes | | | |
|---|---|---|---|---|---|
| | | a | b | c | d |
| Predicted Classes | a | 50 | 3 | 0 | 0 |
| | b | 26 | 8 | 0 | 1 |
| | c | 20 | 2 | 4 | 0 |
| | d | 12 | 0 | 0 | 1 |

# Confusion Matrix for multiclassification

We can separate each class in a single confusion matrix to make calculations and visualizations easier



Predicted as 'A', and actually 'A'

Predicted as 'A', but actually 'Not A'

Predicted as 'Not A', but actually 'A'

Predicted as 'Not A', and actually 'Not A'

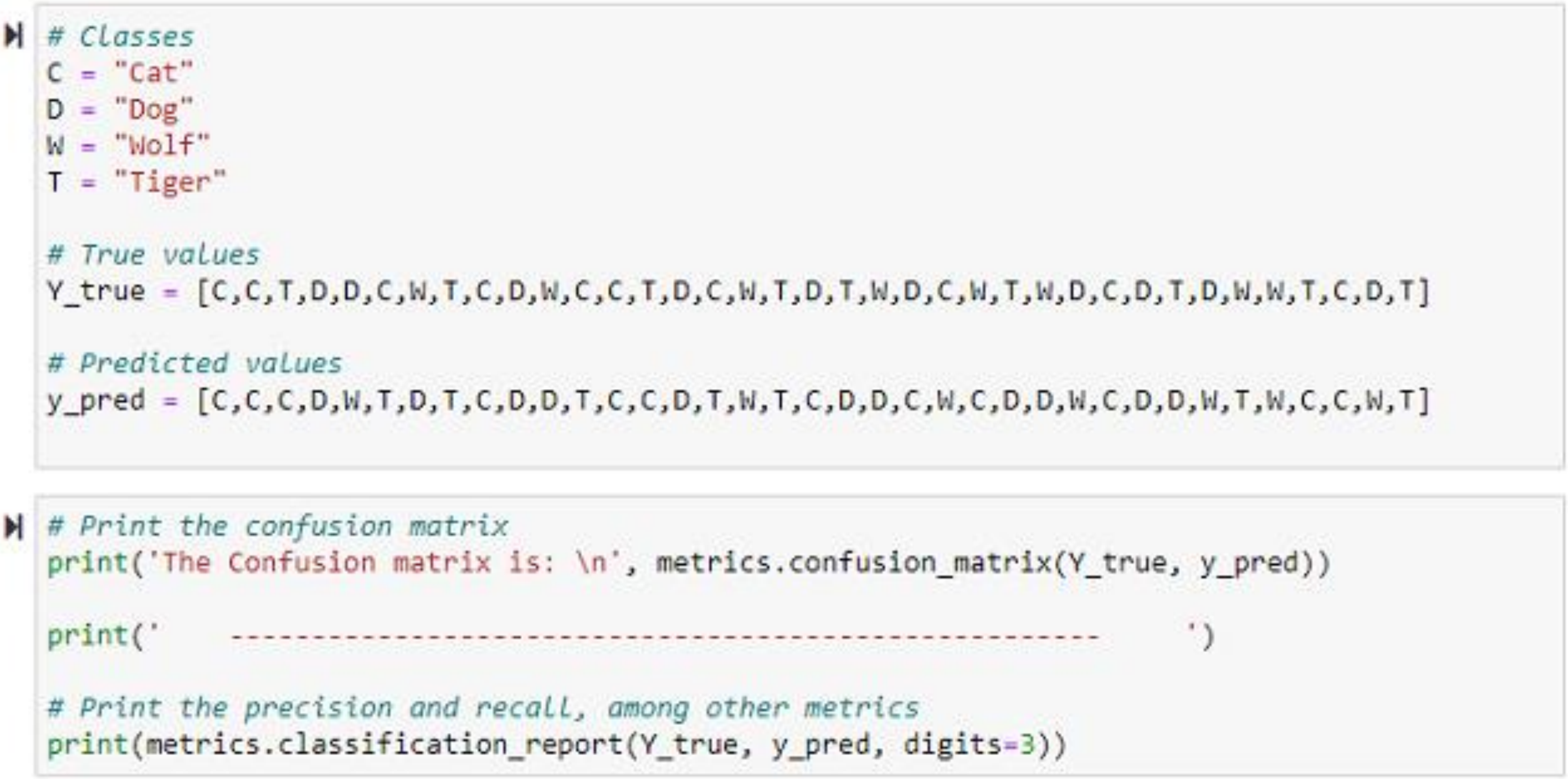
Actual

| Predicted | A | B | C |
|---|---|---|---|
| A | True AA | False AB | False AC |
| B | False BA | True BB | False BC |
| C | False CA | False CB | True CC |

Actual

| Predicted | A | Not A |
|---|---|---|
| A | True AA | False AB+AC |
| Not A | False BA+CA | True BC+CB+BB+CC |

# Confusion Matrix for multiclassification



```
# Classes
C = "Cat"
D = "Dog"
W = "Wolf"
T = "Tiger"

# True values
Y_true = [C,C,T,D,D,C,W,T,C,D,W,C,C,T,D,C,W,T,D,T,W,D,C,W,T,W,D,C,D,T,D,W,W,T,C,D,T]

# Predicted values
y_pred = [C,C,C,D,W,T,D,T,C,D,D,T,C,C,D,T,W,T,C,D,D,C,W,C,D,D,W,C,D,D,W,T,W,C,C,W,T]
```

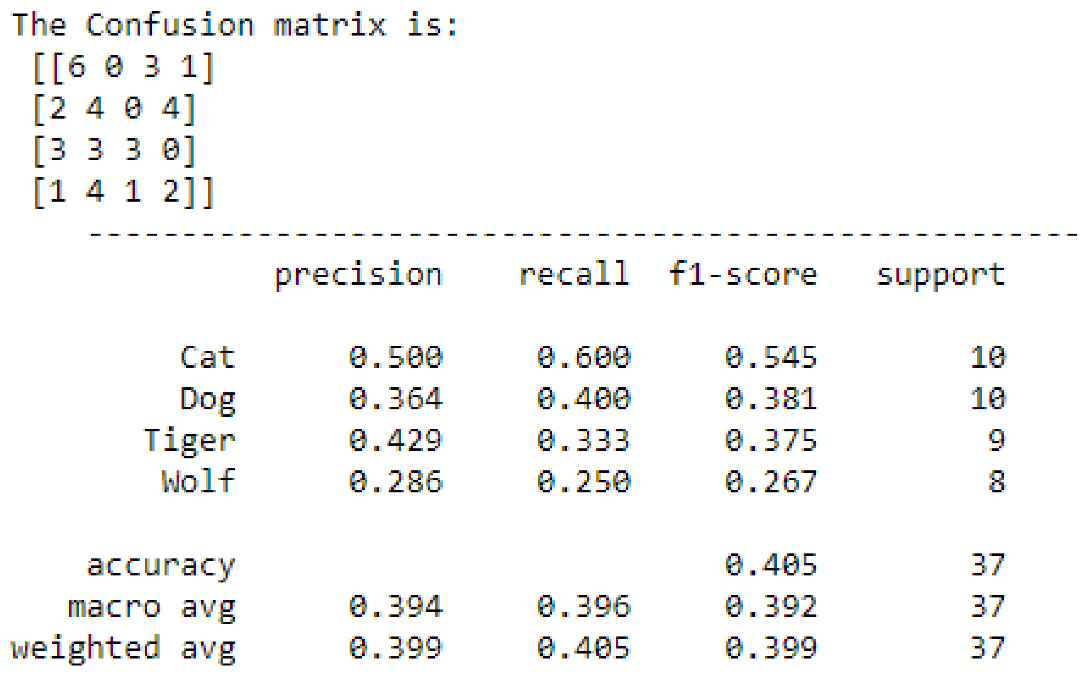
```
# Print the confusion matrix
print('The Confusion matrix is: \n', metrics.confusion_matrix(Y_true, y_pred))

print('    -----------------------------------------------------    ')

# Print the precision and recall, among other metrics
print(metrics.classification_report(Y_true, y_pred, digits=3))
```

https://bassantgz30.medium.com/performance-metrics-for-classification-models-in-machine-learning-part-ii-9303a1c7cadd

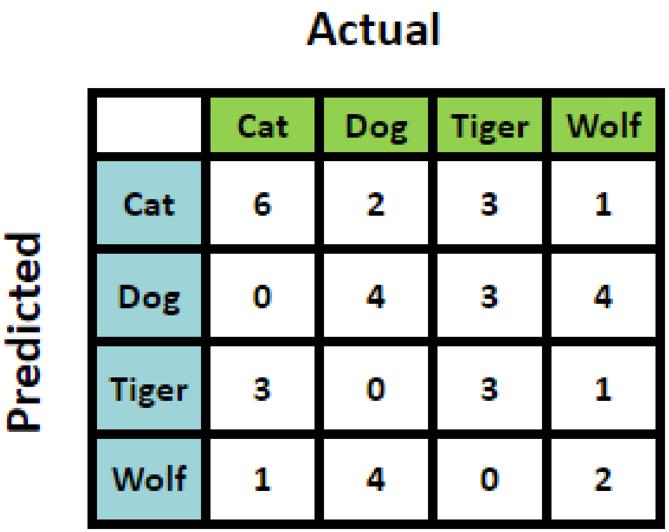
# Confusion Matrix for multiclassification



```
The Confusion matrix is:
 [[6 0 3 1]
 [2 4 0 4]
 [3 3 3 0]
 [1 4 1 2]]
    -----------------------------------------------------
              precision    recall  f1-score   support

         Cat      0.500     0.600     0.545        10
         Dog      0.364     0.400     0.381        10
       Tiger      0.429     0.333     0.375         9
        Wolf      0.286     0.250     0.267         8

    accuracy                          0.405        37
   macro avg      0.394     0.396     0.392        37
weighted avg      0.399     0.405     0.399        37
```

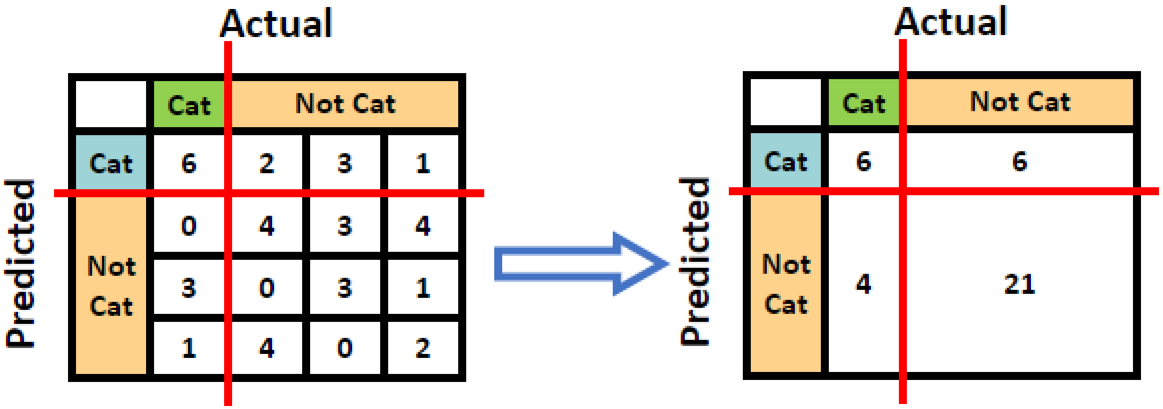


Actual

Predicted

| | Cat | Dog | Tiger | Wolf |
|---|---|---|---|---|
| Cat | 6 | 2 | 3 | 1 |
| Dog | 0 | 4 | 3 | 4 |
| Tiger | 3 | 0 | 3 | 1 |
| Wolf | 1 | 4 | 0 | 2 |

# Confusion Matrix for multiclassification



| | Cat | Not Cat | | |
|---|---|---|---|---|
| Cat | 6 | 2 | 3 | 1 |
| Not Cat | 0 | 4 | 3 | 4 |
| | 3 | 0 | 3 | 1 |
| | 1 | 4 | 0 | 2 |

| | Cat | Not Cat |
|---|---|---|
| Cat | 6 | 6 |
| Not Cat | 4 | 21 |

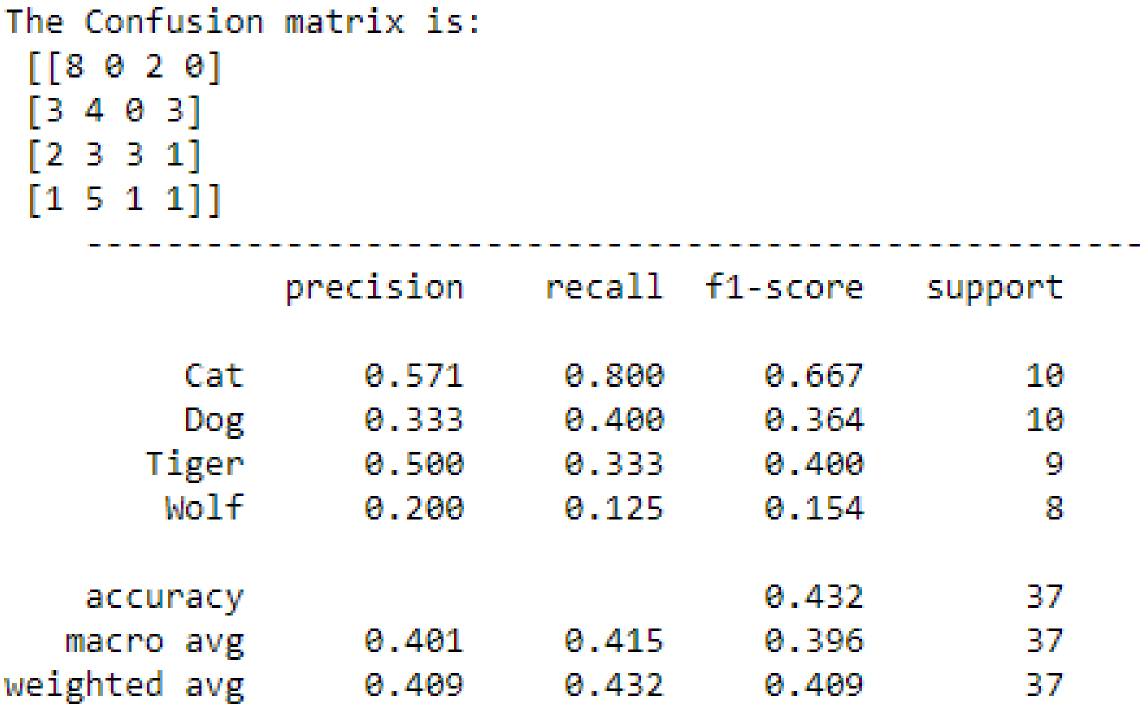
Recall = TP/(TP+FN) = 6/(6+4) = 0.6 → **Out of 10 actual cats, the model captured 6 cats correctly.**

Precision = TP/(TP+FP) = 6/(6+6) = 0.5 → **Out of 12 captured cats, there are 6 actual cats.**

The F1-Score = 2 * (Precision * Recall)/(Precision + Recall) = 0.545.

# Confusion Matrix for multiclassification

Assume we have made some changes to the model to capture more cats correctly and got the following results:

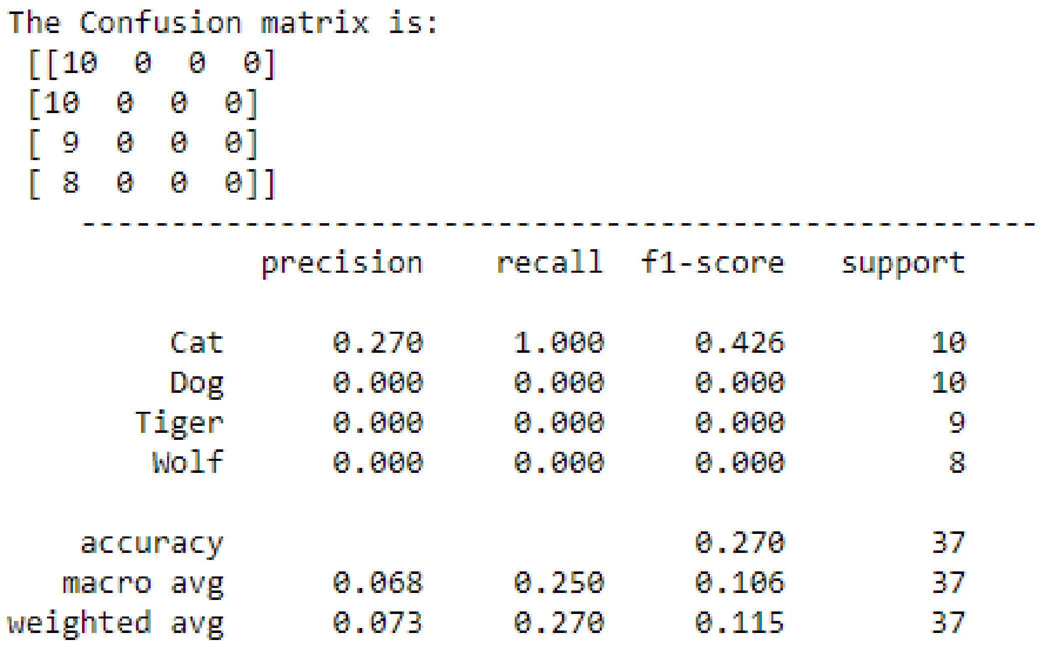


```
The Confusion matrix is:
 [[8 0 2 0]
 [3 4 0 3]
 [2 3 3 1]
 [1 5 1 1]]
   -----------------------------------------------------
              precision    recall  f1-score   support

         Cat      0.571     0.800     0.667        10
         Dog      0.333     0.400     0.364        10
       Tiger      0.500     0.333     0.400         9
        Wolf      0.200     0.125     0.154         8

    accuracy                          0.432        37
   macro avg      0.401     0.415     0.396        37
weighted avg      0.409     0.432     0.409        37
```

# Confusion Matrix for multiclassification



```
The Confusion matrix is:
 [[10  0  0  0]
 [10  0  0  0]
 [ 9  0  0  0]
 [ 8  0  0  0]]
    -----------------------------------------------------
              precision    recall  f1-score   support

         Cat      0.270     1.000     0.426        10
         Dog      0.000     0.000     0.000        10
       Tiger      0.000     0.000     0.000         9
        Wolf      0.000     0.000     0.000         8

    accuracy                          0.270        37
   macro avg      0.068     0.250     0.106        37
weighted avg      0.073     0.270     0.115        37
```

The precision, recall, and f1-score for all other classes are zeros. This means that the model failed to capture those classes. It classifies any picture as a ‘Cat’, so it can capture all the ‘Cats’ in our dataset, that’s why the recall is 1, but as the dataset have other pictures that are not ‘Cats’, the precision for the ‘Cat’ class is very low (0.27).

# Confusion Matrix Implementation

- Implementation of Confusion Matrix

# Handling Categorical Data in Machine Learning

- Machine learning models require all input and output variables to be **numeric**.

- This means that if your data contains **categorical data (Text / String)**, you must **encode it to numbers** before you can fit and evaluate a model.

- The two most popular techniques are:
  - Label Encoding
  - One-Hot Encoding.

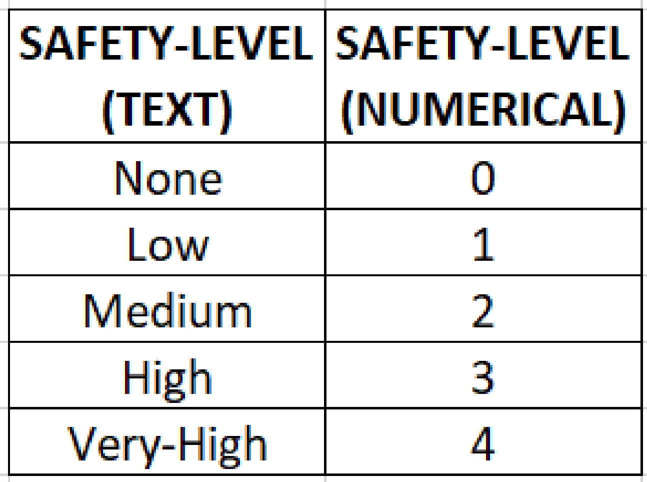
# Handling Categorical Data in Machine Learning

- Categorical variables are often called nominal.
- **Nominal Variable (Categorical).** Variable comprises a finite set of discrete values with no relationship between values.
- **Ordinal Variable.** Variable comprises a finite set of discrete values with a ranked ordering between values.
- Many machine learning algorithms cannot operate on nominal data directly. They require all input variables and output variables to be numeric.

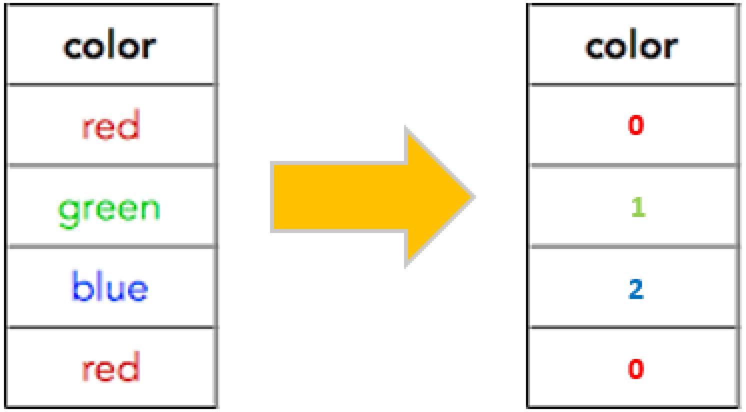
# Label Encoding

- In label encoding, each unique category value is assigned an integer value.

- This is also called an ordinal encoding or an integer encoding and is easily reversible. Often, integer values starting at zero are used.



| SAFETY-LEVEL (TEXT) | SAFETY-LEVEL (NUMERICAL) |
|---|---|
| None | 0 |
| Low | 1 |
| Medium | 2 |
| High | 3 |
| Very-High | 4 |



| color |
|---|
| red |
| green |
| blue |
| red |

| color |
|---|
| 0 |
| 1 |
| 2 |
| 0 |

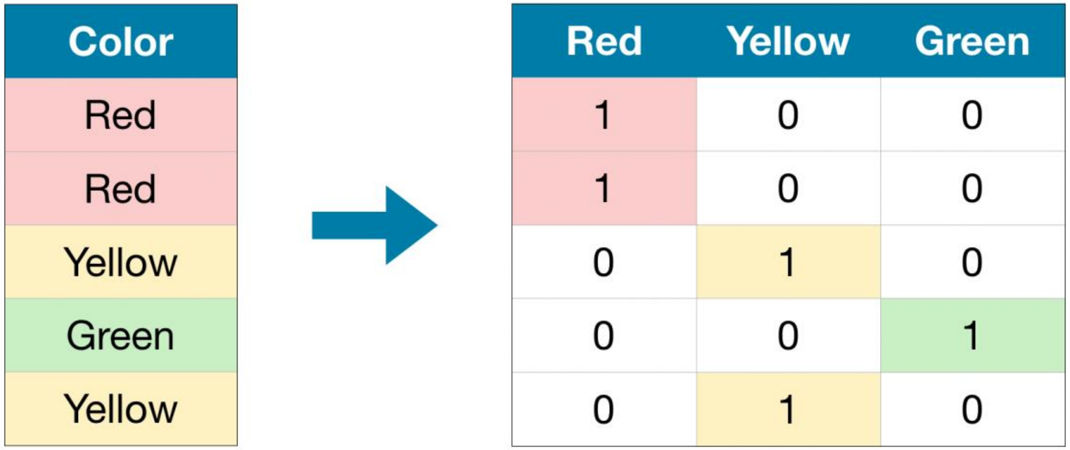
# One Hot Encoding

- Forcing an ordinal relationship via an ordinal or label encoding and allowing the model to assume a natural ordering between categories may result in poor performance or unexpected results.

- One-Hot Encoding is the most common, correct way to deal with categorical data.

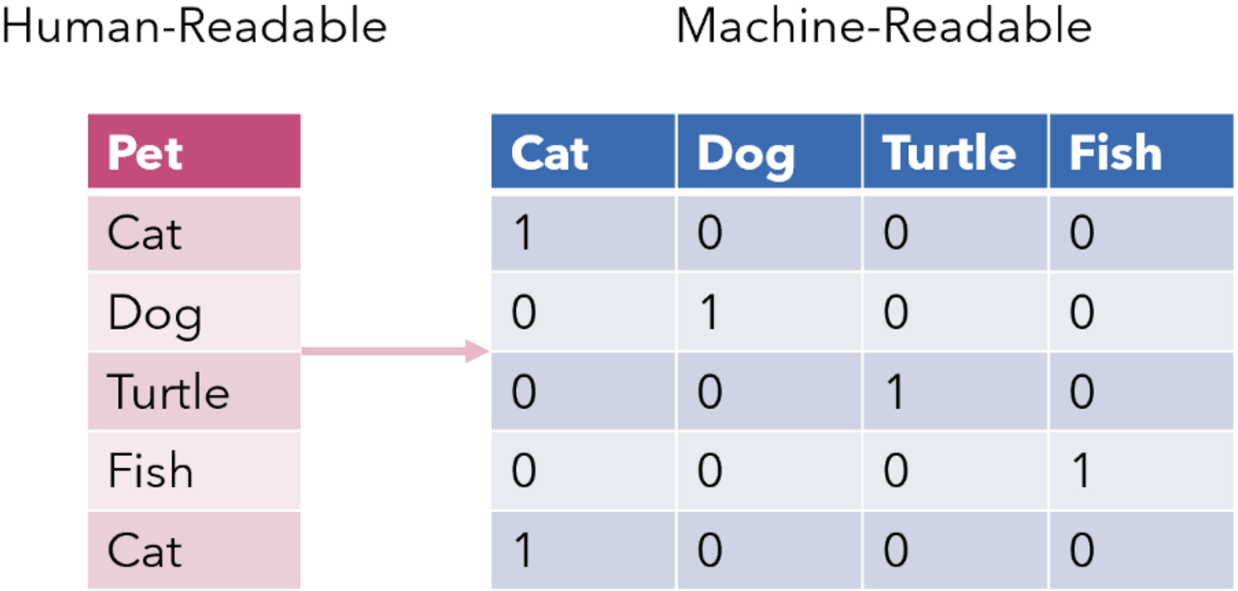
- It consists of creating an additional feature for each group of the categorical feature and mark each observation belonging (Value=1) or not (Value=0) to that group.

# One Hot Encoding



| Color |
|---|
| Red |
| Red |
| Yellow |
| Green |
| Yellow |

| Red | Yellow | Green |
|---|---|---|
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 1 |
| 0 | 1 | 0 |



Human-Readable

| Pet |
|---|
| Cat |
| Dog |
| Turtle |
| Fish |
| Cat |

Machine-Readable

| Cat | Dog | Turtle | Fish |
|---|---|---|---|
| 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 |

# One Hot Encoding Implementation

- Implementation of One Hot Encoding

# Spam Filtering

- Spam filters detect unsolicited, unwanted, and virus-infested email (called spam) and stop it from getting into email inboxes.

- Internet Service Providers (ISPs) use spam filters to make sure they aren’t distributing spam.

- Small- to medium- sized businesses (SMBs) also use spam filters to protect their employees and networks.

# How do Spam Filters Work

- Spam filters use “heuristics” methods, which means that each email message is subjected to thousands of predefined rules (algorithms).

- Each rule assigns a numerical score to the probability of the message being spam, and if the score passes a certain threshold the email is flagged as spam and blocked from going further.

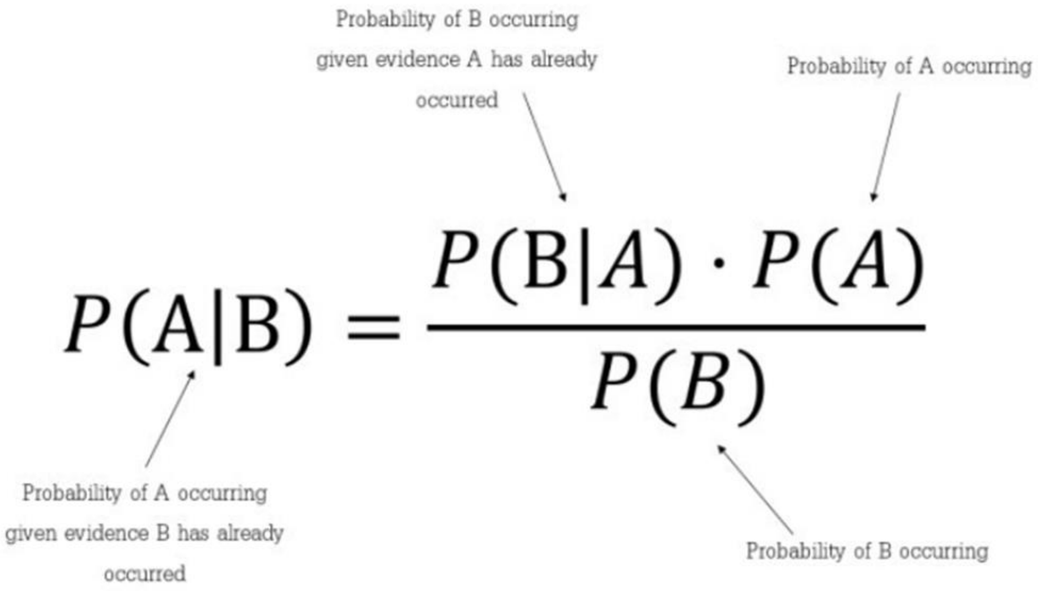
# Types of Spam Filters

- **Content filters:** parse the content of messages, scanning for words that are commonly used in spam emails.
- **Header filters:** examine the email header source to look for suspicious information (such as spammer email addresses).
- **Blocklist filters:** stop emails that come from a blocklist of suspicious IP addresses. Some filters go further and check the IP reputation of the IP address.
- **Rules-based filters:** apply customized rules designed by the organization to exclude emails from specific senders, or emails containing specific words in their subject line or body.

# Types of Spam Filters

- **Bayesian Filter:** A Bayesian filter is a filter that learns your spam preferences.  When you mark emails as spam, the system will note the characteristics of the email and look for similar characteristics in incoming email, filtering anything that fits the formula directly in to spam for you.
- A Bayesian filter is one of the most intelligent types of spam filter because it is able to learn and adapt on its own.

# How Bayesian Filter Works?



$$P(\mathrm{A}|\mathrm{B}) = \frac{P(\mathrm{B}|A) \cdot P(A)}{P(B)}$$

Probability of B occurring given evidence A has already occurred

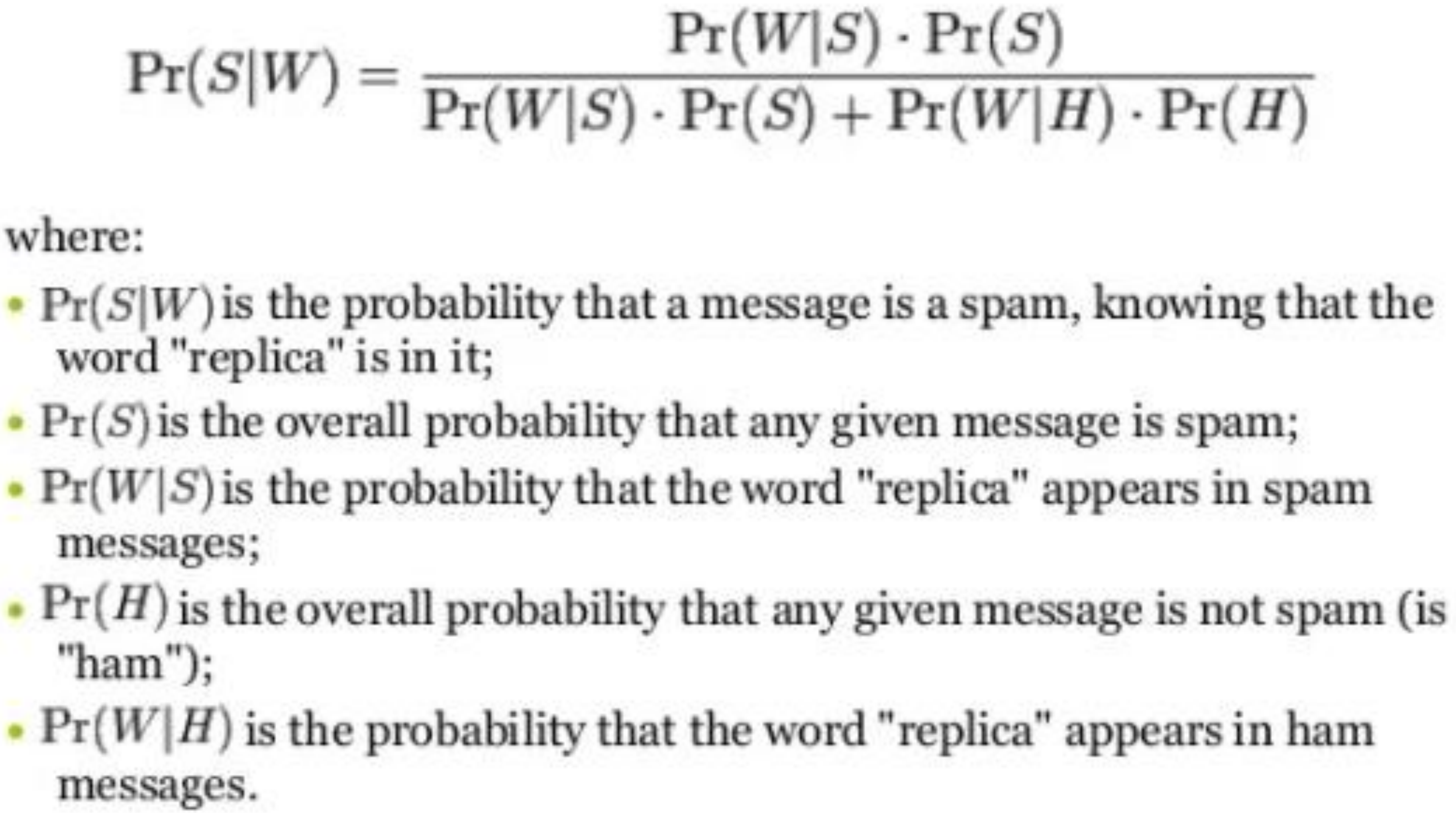
Probability of A occurring

Probability of A occurring given evidence B has already occurred

Probability of B occurring

$$P_r(spam|word) = \frac{P_r(word|spam)P_r(spam)}{P_r(word)}$$

# How Bayesian Filter Works?



$$\Pr(S|W) = \frac{\Pr(W|S)\cdot\Pr(S)}{\Pr(W|S)\cdot\Pr(S) + \Pr(W|H)\cdot\Pr(H)}$$

where:

- $\Pr(S|W)$ is the probability that a message is a spam, knowing that the word "replica" is in it;
- $\Pr(S)$ is the overall probability that any given message is spam;
- $\Pr(W|S)$ is the probability that the word "replica" appears in spam messages;
- $\Pr(H)$ is the overall probability that any given message is not spam (is "ham");
- $\Pr(W|H)$ is the probability that the word "replica" appears in ham messages.

# Spam vs. Ham Emails?

**Ham:** E-mail that is generally desired and isn't considered spam. Better to use ‘non-Spam’.

$$P(s|w) = \frac{P(w|s)P(s)}{P(w|s \cup h)}$$

The probability that an email is a spam message if a word *w* occurs is defined by the probability that word *w* is in a spam message *s* multiplied by the general probability that the email is a spam message *s*. This gets divided by the probability of that word occurring in an e-mail (spam and ham combined)

# Spam vs. Ham Emails?

**Ham:** E-mail that is generally desired and isn't considered spam. Better to use ‘non-Spam’.

$$P(s|w) = \frac{P(w|s)P(s)}{P(w|s \cup h)}$$

$$P(w|s) = \frac{\text{number of spam emails containing } w}{\text{total number of spam emails}}$$

$$P(w|\neg s) = \frac{\text{number of ham emails containing } w}{\text{total number of ham emails}}$$

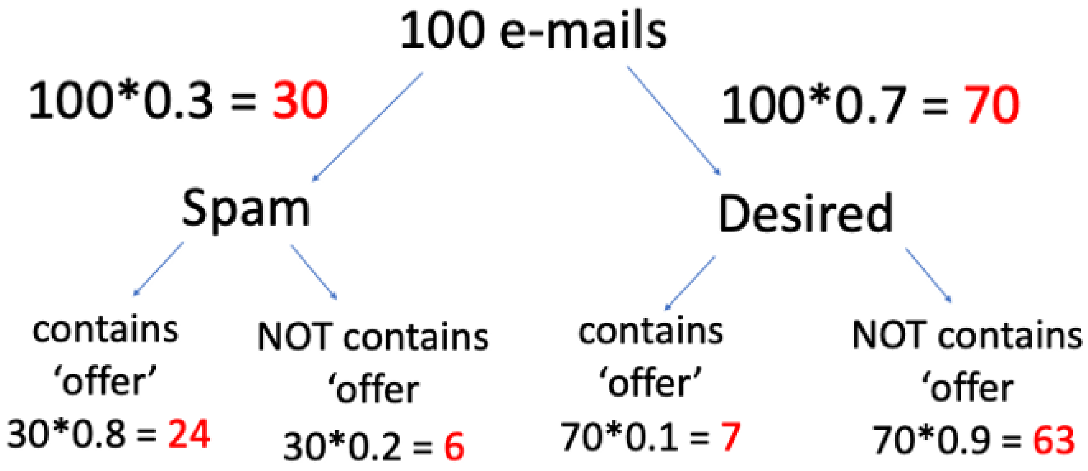
# Spam vs. Ham Emails?

$$P = \frac{p_1 p_2 \ldots p_n}{p_1 p_2 \ldots p_n + (1 - p_1)(1 - p_2) \ldots (1 - p_n)}$$

We can achieve this by multiplying the probabilities for every word together and dividing by the **combined probability** of every word for being in a spam message plus the probability of every word for not being in a spam message.

# Example

Let us assume we receive 100 emails. 30% are Spam and 70% are non-spam. The percentage of the word ‘offer’ that occurs in spam e-mails is 80%. The percentage of the word ‘offer’ that occurs in the desired e-mails is 10%.



100 e-mails

100*0.3 = 30

100*0.7 = 70

Spam

Desired

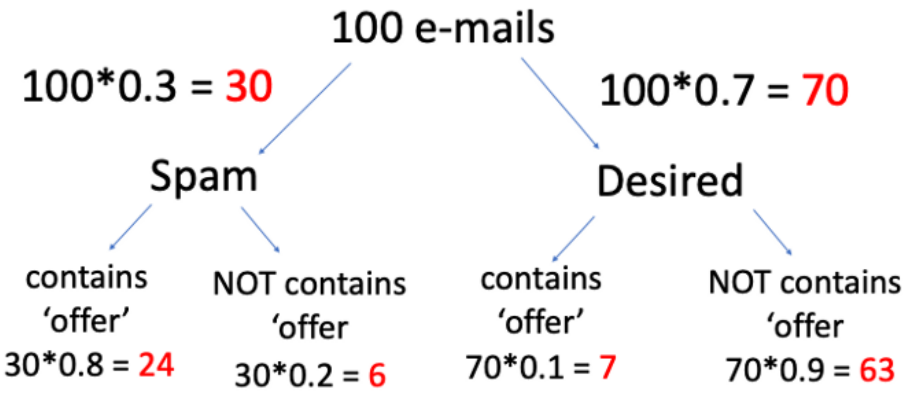
contains ‘offer’ 30*0.8 = 24

NOT contains ‘offer 30*0.2 = 6

contains ‘offer’ 70*0.1 = 7

NOT contains ‘offer 70*0.9 = 63

# Example



100 e-mails

100*0.3 = 30

100*0.7 = 70

Spam

Desired

contains ‘offer’

30*0.8 = 24

NOT contains ‘offer

30*0.2 = 6

contains ‘offer’

70*0.1 = 7

NOT contains ‘offer

70*0.9 = 63

$$P(spam|contains\ offer) = \frac{P(contains\ offer|spam) * P(spam)}{P(contains\ offer)}$$

$$P(spam|contains\ offer) = \frac{0.8 * 0.3}{0.31} = 0.774$$

# Solved Example

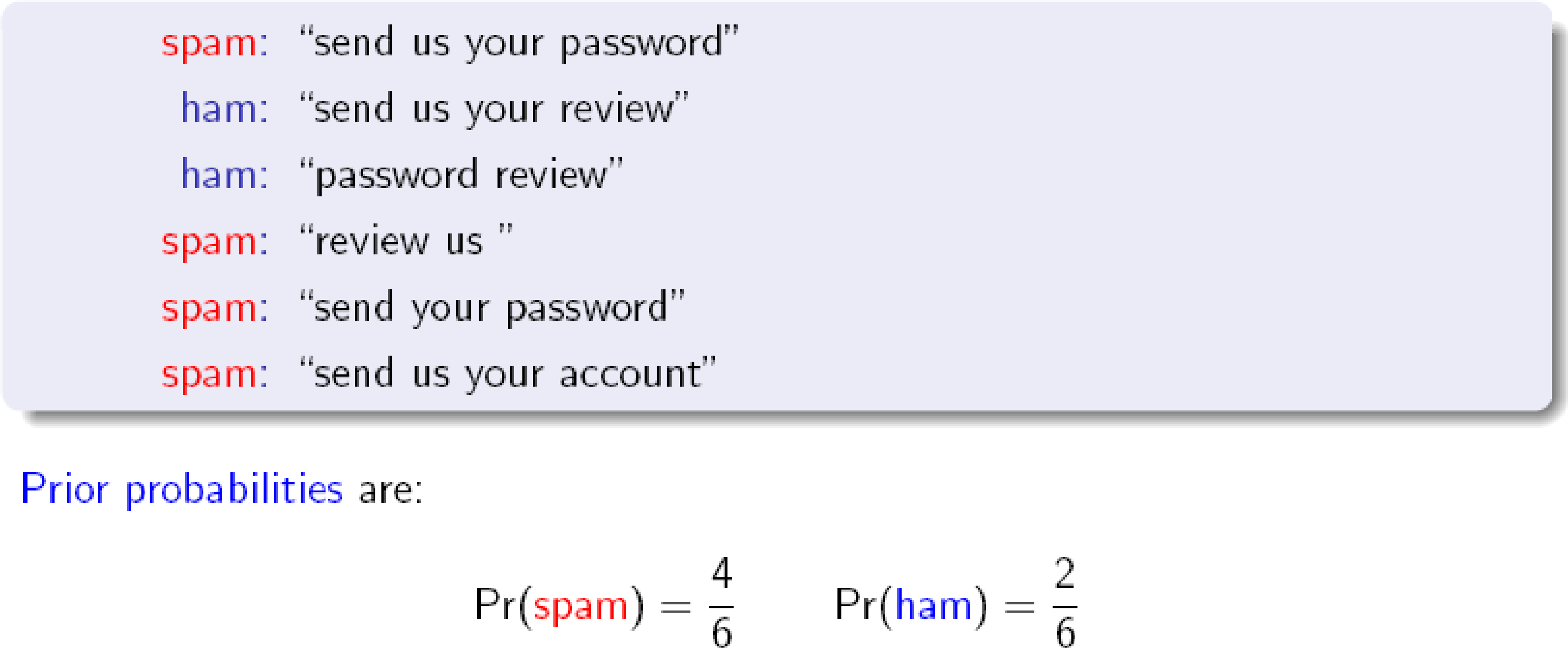
Assume that we have the following set of email classified as spam or ham.

| | |
|---|---|
| spam: | “send us your password” |
| ham: | “send us your review” |
| ham: | “password review” |
| spam: | “review us ” |
| spam: | “send your password” |
| spam: | “send us your account” |

We are interested in classifying the following new email as spam or ham:

new email “review us now”

# Solved Example



spam: “send us your password”

ham: “send us your review”

ham: “password review”

spam: “review us ”

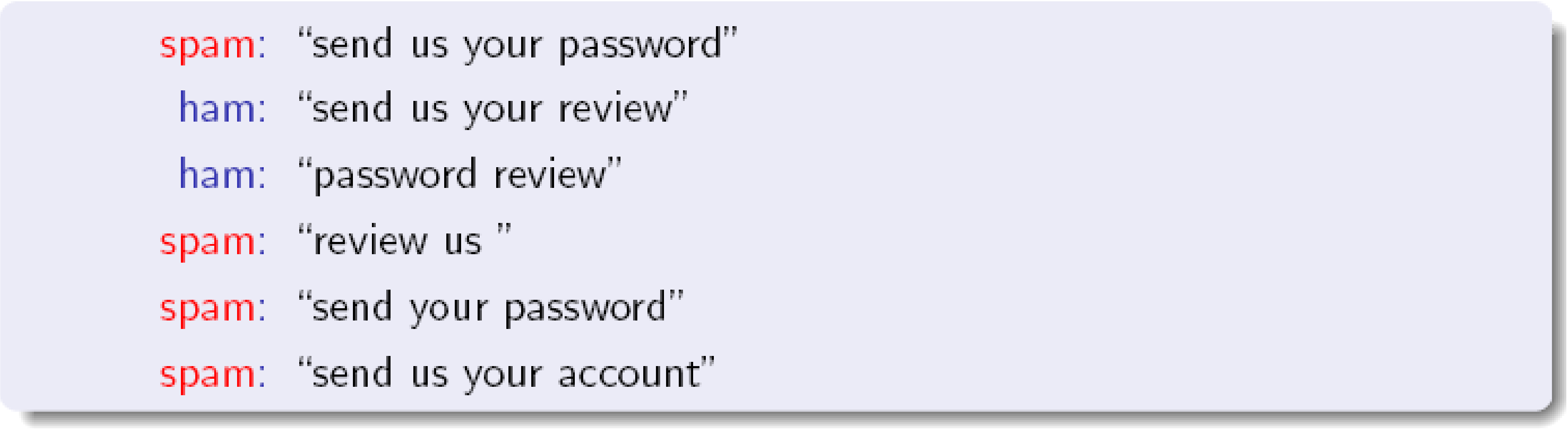
spam: “send your password”

spam: “send us your account”

Prior probabilities are:

$$\Pr(\text{spam}) = \frac{4}{6} \qquad \Pr(\text{ham}) = \frac{2}{6}$$

# Solved Example



spam: “send us your password”

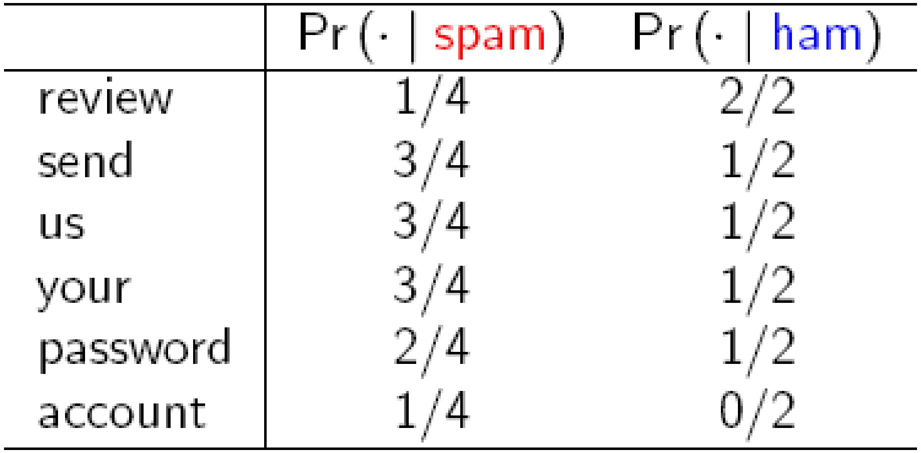
ham: “send us your review”

ham: “password review”

spam: “review us ”

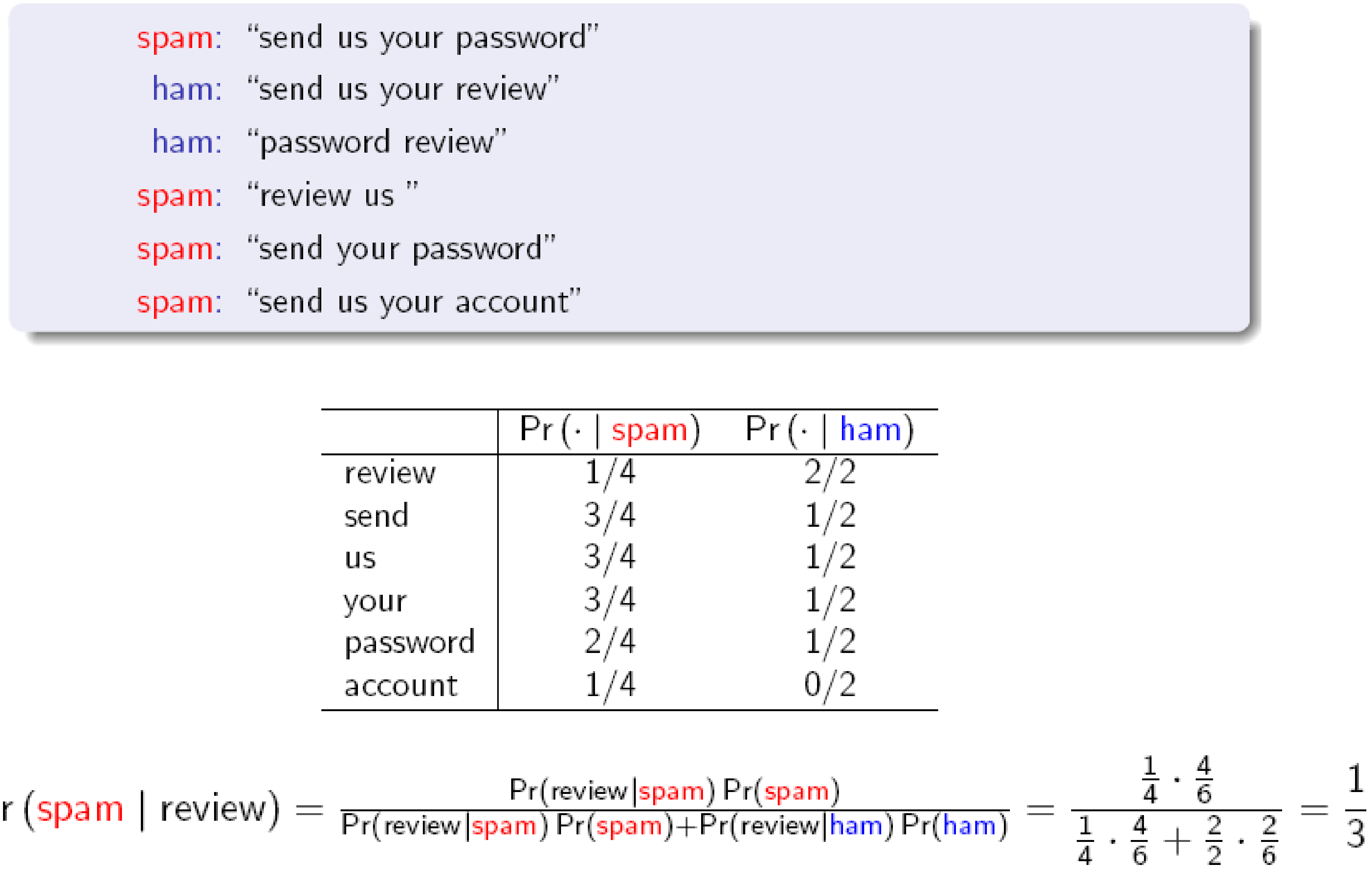
spam: “send your password”

spam: “send us your account”



|  | $\Pr(\cdot \mid$ spam$)$ | $\Pr(\cdot \mid$ ham$)$ |
|---|---|---|
| review | 1/4 | 2/2 |
| send | 3/4 | 1/2 |
| us | 3/4 | 1/2 |
| your | 3/4 | 1/2 |
| password | 2/4 | 1/2 |
| account | 1/4 | 0/2 |

# Solved Example



spam: “send us your password”

ham: “send us your review”

ham: “password review”

spam: “review us ”

spam: “send your password”

spam: “send us your account”

|  | $\Pr(\cdot \mid \text{spam})$ | $\Pr(\cdot \mid \text{ham})$ |
|---|---|---|
| review | $1/4$ | $2/2$ |
| send | $3/4$ | $1/2$ |
| us | $3/4$ | $1/2$ |
| your | $3/4$ | $1/2$ |
| password | $2/4$ | $1/2$ |
| account | $1/4$ | $0/2$ |

$$\Pr(\text{spam} \mid \text{review}) = \frac{\Pr(\text{review}|\text{spam})\Pr(\text{spam})}{\Pr(\text{review}|\text{spam})\Pr(\text{spam})+\Pr(\text{review}|\text{ham})\Pr(\text{ham})} = \frac{\frac{1}{4}\cdot\frac{4}{6}}{\frac{1}{4}\cdot\frac{4}{6}+\frac{2}{2}\cdot\frac{2}{6}} = \frac{1}{3}$$

# Solved Example

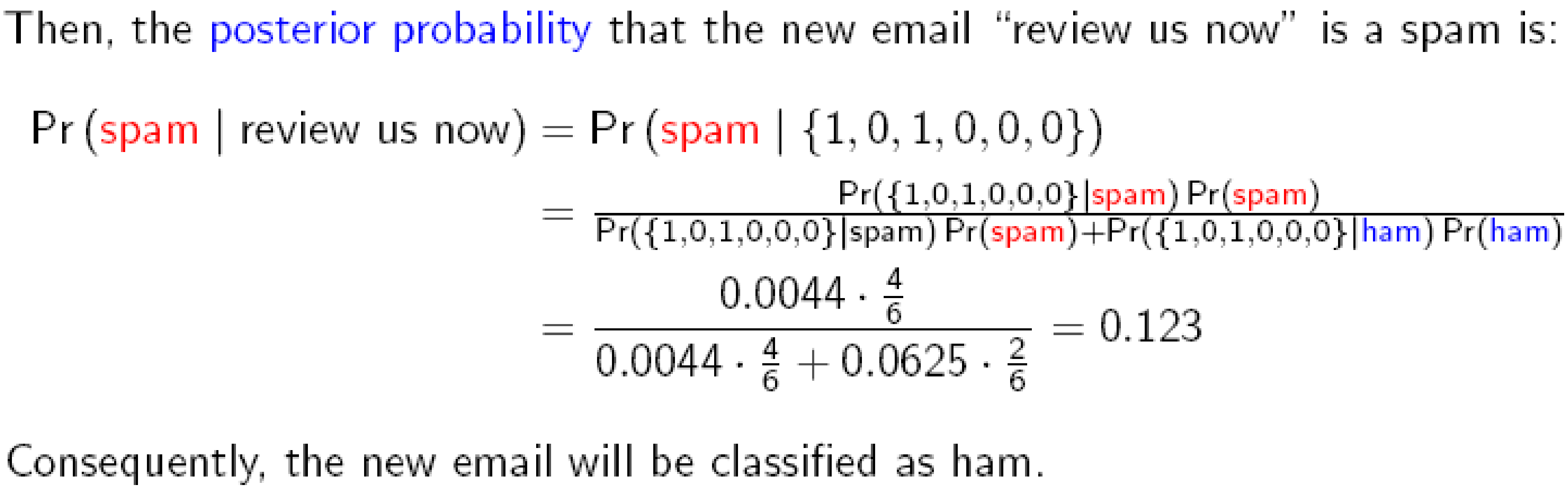
| | $\Pr(\cdot \mid \text{spam})$ | $\Pr(\cdot \mid \text{ham})$ |
|---|---|---|
| review | 1/4 | 2/2 |
| send | 3/4 | 1/2 |
| us | 3/4 | 1/2 |
| your | 3/4 | 1/2 |
| password | 2/4 | 1/2 |
| account | 1/4 | 0/2 |

- Assuming that the words in each message are independent events:

$$\begin{aligned}\Pr(\text{review us now} \mid \text{spam}) &= \Pr(\{1,0,1,0,0,0\} \mid \text{spam}) \\ &= \frac{1}{4}\left(1-\frac{3}{4}\right)\frac{3}{4}\left(1-\frac{3}{4}\right)\left(1-\frac{2}{4}\right)\left(1-\frac{1}{4}\right) = 0.0044\end{aligned}$$

$$\begin{aligned}\Pr(\text{review us now} \mid \text{ham}) &= \Pr(\{1,0,1,0,0,0\} \mid \text{ham}) \\ &= \frac{2}{2}\left(1-\frac{1}{2}\right)\frac{1}{2}\left(1-\frac{1}{2}\right)\left(1-\frac{1}{2}\right)\left(1-\frac{0}{4}\right) = 0.0625\end{aligned}$$

# Solved Example



Then, the posterior probability that the new email "review us now" is a spam is:

$$\begin{aligned}
\Pr(\text{spam} \mid \text{review us now}) &= \Pr(\text{spam} \mid \{1,0,1,0,0,0\}) \\
&= \frac{\Pr(\{1,0,1,0,0,0\}|\text{spam})\,\Pr(\text{spam})}{\Pr(\{1,0,1,0,0,0\}|\text{spam})\,\Pr(\text{spam})+\Pr(\{1,0,1,0,0,0\}|\text{ham})\,\Pr(\text{ham})} \\
&= \frac{0.0044 \cdot \frac{4}{6}}{0.0044 \cdot \frac{4}{6} + 0.0625 \cdot \frac{2}{6}} = 0.123
\end{aligned}$$

Consequently, the new email will be classified as ham.

# Working of a Spam Filter

- Filter filler words and special characters out of e-mail
- Split data into test and train set
- Calculate the total probability of an e-mail being spam or ham ($P(s)$ and $P(\mathrm{h})$)
- Calculate the conditional probability of a word occurring in a spam ($P(w|s)$)
- Calculate the total probability of $P$ (spamminess) for every email in the train set.
- Search for best threshold
- Test on test set
- Evaluate results

# Spam Filter

Implement Spam Filter in Python

# Summary

- Confusion Matrix
- Encoding categorical data
- Spam Filtering