Logistic Regression

Logistic Regression

- In statistics, the logistic model is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick.
- Logistic regression is a generalized linear model using the same underlying formula, but instead of the continuous output, it is regressing for the probability of a categorical outcome.
- In other words, it deals with one outcome variable with two states of the variable - either 0 or 1.
- Despite its name, Logistic regression is a model for classification, not regression.

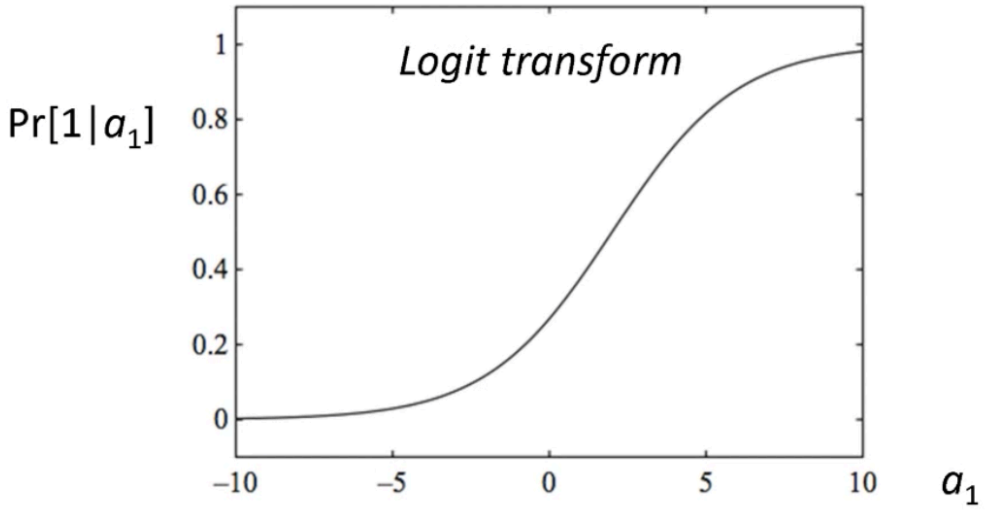
$$\ell = \log_b \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

- In the logistic model, the log-odds for the value labeled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable or a continuous variable.

Logistic Regression

- Linear regression: calculate a linear function and then a threshold
- Logistic regression: estimate class probabilities directly

$$\Pr[1 \mid a_1, a_2, \ldots, a_k] = 1/(1 + \exp(-w_0 - w_1 a_1 - \ldots - w_k a_k))$$



- Choose weights to maximize the log-likelihood (not minimize the squared error):

$$\sum_{i=1}^{n} (1 - x^{(i)}) \log(1 - \Pr[1 \mid a_1^{(1)}, a_2^{(2)}, \ldots, a_k^{(k)}]) + x^{(i)} \log(\Pr[1 \mid a_1^{(1)}, a_2^{(2)}, \ldots, a_k^{(k)}])$$

Receiver Operating Characteristic (ROC) Curve

- All our classification prediction models return a score which is then thresholded.

Example

$$threshold(score, 0.5) = \begin{cases} positive & \text{if } score \geq 0.5 \\ negative & otherwise \end{cases} \tag{10}$$

Table: A sample test set with model predictions and scores (threshold= 0.5.

ID	Target	Pred-iction	Score	Out-come
7	ham	ham	0.001	TN
11	ham	ham	0.003	TN
15	ham	ham	0.059	TN
13	ham	ham	0.064	TN
19	ham	ham	0.094	TN
12	spam	ham	0.160	FN
2	spam	ham	0.184	FN
3	ham	ham	0.226	TN
16	ham	ham	0.246	TN
1	spam	ham	0.293	FN

ID	Target	Pred-iction	Score	Out-come
5	ham	ham	0.302	TN
14	ham	ham	0.348	TN
17	ham	spam	0.657	FP
8	spam	spam	0.676	TP
6	spam	spam	0.719	TP
10	spam	spam	0.781	TP
18	spam	spam	0.833	TP
20	ham	spam	0.877	FP
9	spam	spam	0.960	TP
4	spam	spam	0.963	TP

- We have ordered the examples by score so the threshold is apparent in the predictions.
- Note that, in general, instances that actually should get a prediction of *'ham'* generally have a low score, and those that should get a prediction of *'spam'* generally get a high score.

- There are a number of performance measures that use this ability of a model to rank instances that should get predictions of one target level higher than the other, to assess how well the model is performing.
- The basis of most of these approaches is measuring how well the distributions of scores produced by the model for different target levels are separated

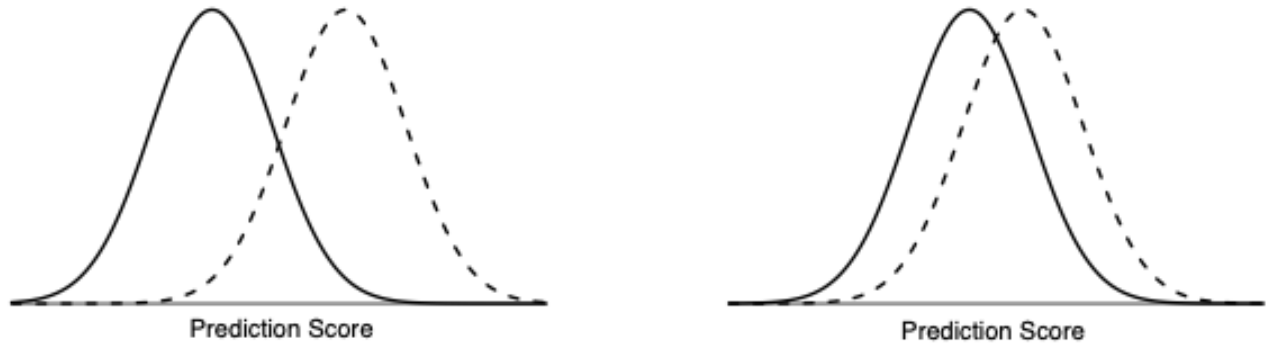
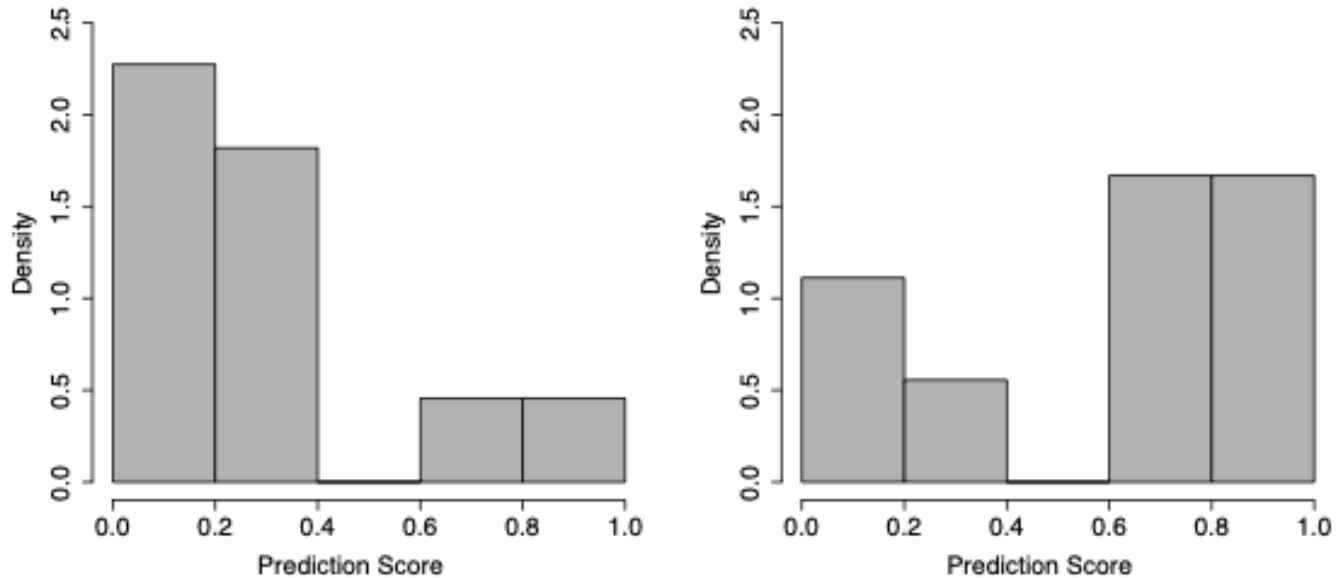


Figure: Prediction score distributions for two different prediction models. The distributions in (a) are much better separated than those in (b).



(a) spam (b) ham

Figure: Prediction score distributions for the (a) *'spam'* and (b) *'ham'* target levels based on the data in Table

- The receiver operating characteristic index (ROC index), which is based on the receiver operating characteristic curve (ROC curve), is a widely used performance measure that is calculated using prediction scores.
- TPR and TNR are intrinsically tied to the threshold used to convert prediction scores into target levels.
- This threshold can be changed, however, which leads to different predictions and a different confusion matrix.

Table: Confusion matrices for the set of predictions shown in Table using (a) a prediction score threshold of 0.75 and (b) a prediction score threshold of 0.25.

(a) Threshold: 0.75

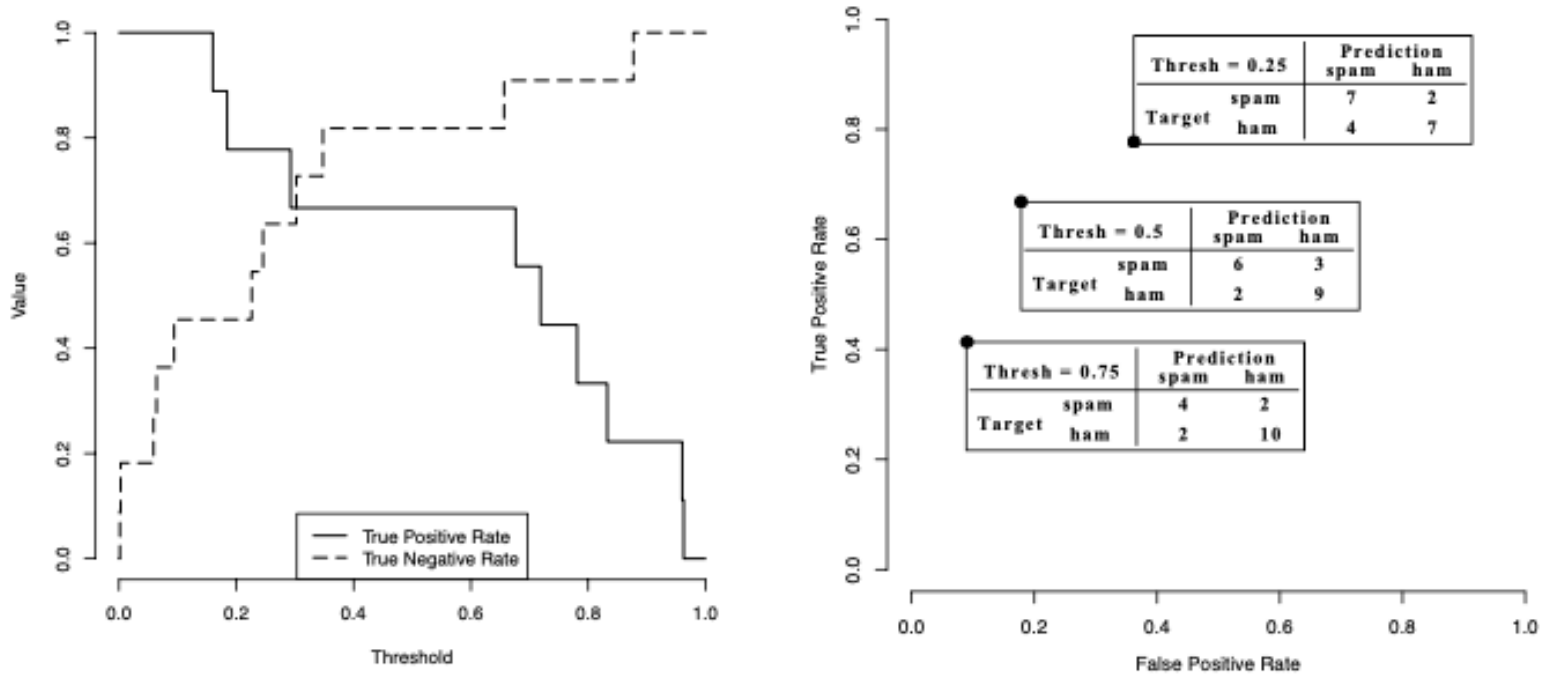
		Prediction	
		'spam'	***'ham'***
Target	***'spam'***	4	4
	'ham'	2	10

(b) Threshold: 0.25

		Prediction	
		'spam'	***'ham'***
Target	***'spam'***	7	2
	'ham'	4	7

ID	Target	Score	Pred. (0.10)	Pred. (0.25)	Pred. (0.50)	Pred. (0.75)	Pred. (0.90)
7	ham	0.001	ham	ham	ham	ham	ham
11	ham	0.003	ham	ham	ham	ham	ham
15	ham	0.059	ham	ham	ham	ham	ham
13	ham	0.064	ham	ham	ham	ham	ham
19	ham	0.094	ham	ham	ham	ham	ham
12	spam	0.160	spam	ham	ham	ham	ham
2	spam	0.184	spam	ham	ham	ham	ham
3	ham	0.226	spam	ham	ham	ham	ham
16	ham	0.246	spam	ham	ham	ham	ham
1	spam	0.293	spam	spam	ham	ham	ham
5	ham	0.302	spam	spam	ham	ham	ham
14	ham	0.348	spam	spam	ham	ham	ham
17	ham	0.657	spam	spam	spam	ham	ham
8	spam	0.676	spam	spam	spam	ham	ham
6	spam	0.719	spam	spam	spam	ham	ham
10	spam	0.781	spam	spam	spam	spam	ham
18	spam	0.833	spam	spam	spam	spam	ham
20	ham	0.877	spam	spam	spam	spam	ham
9	spam	0.960	spam	spam	spam	spam	spam
4	spam	0.963	spam	spam	spam	spam	spam
Misclassification Rate			0.300	0.300	0.250	0.300	0.350
True Positive Rate (TPR)			1.000	0.778	0.667	0.444	0.222
True Negative rate (TNR)			0.455	0.636	0.818	0.909	1.000
False Positive Rate (FPR)			0.545	0.364	0.182	0.091	0.000
False Negative Rate (FNR)			0.000	0.222	0.333	0.556	0.778

- Note: as the threshold increases TPR decreases and TNR increases (and vice versa).
- Capturing this tradeoff is the basis of the ROC curve.



(a) (b)

Figure: (a) The changing values of TPR and TNR for the test data shown in Table 36 [37] as the threshold is altered; (b) points in ROC space for thresholds of 0.25, 0.5, and 0.75.

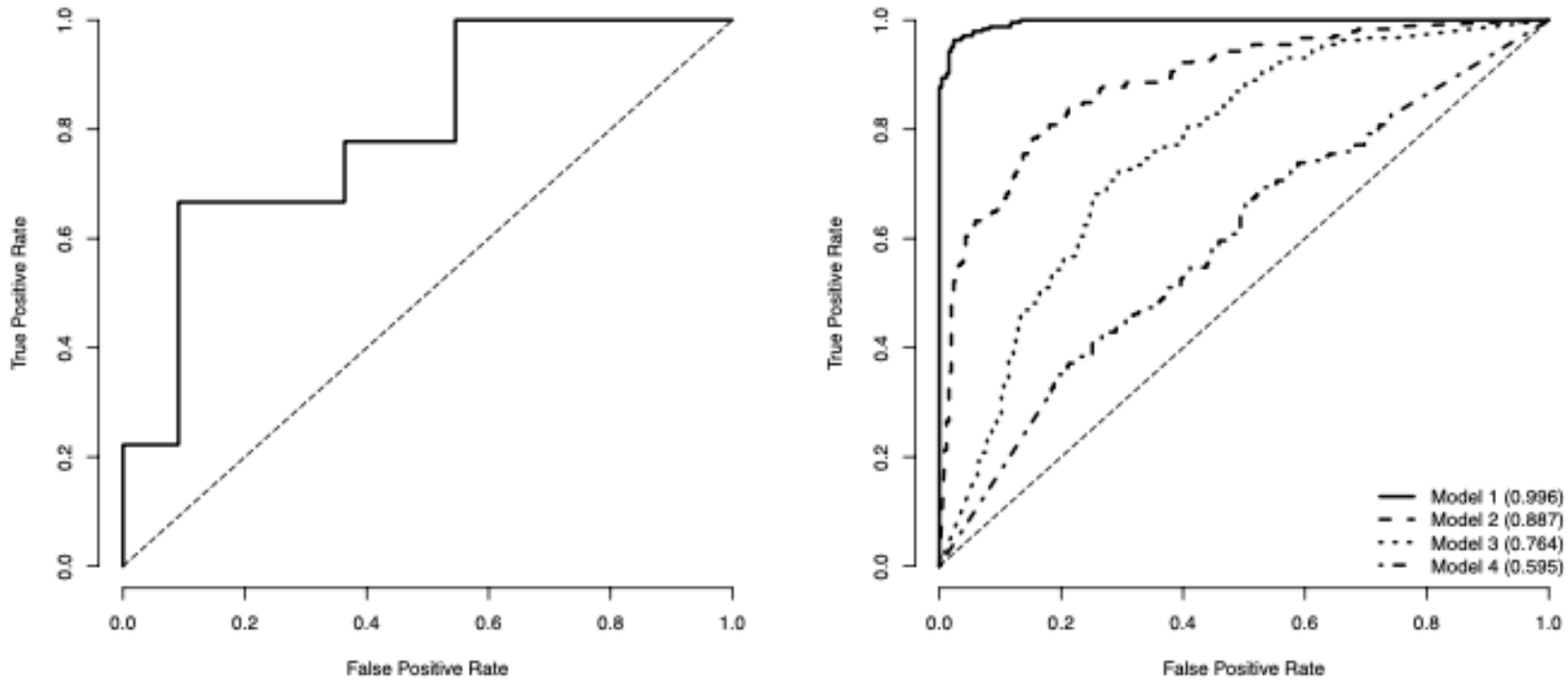


Figure: (a) A complete ROC curve for the email classification example; (b) a selection of ROC curves for different models trained on the same prediction task.

- We can also calculate a single performance measure from an ROC curve
- The ROC Index measures the area underneath an ROC curve.

$$\text{ROC index} = \sum_{i=2}^{|\mathbf{T}|} \frac{(FPR(\mathbf{T}[i]) - FPR(\mathbf{T}[i-1])) \times (TPR(\mathbf{T}[i]) + TPR(\mathbf{T}[i-1]))}{2}$$

- The **Gini coefficient** is a linear rescaling of the ROC index

$$\text{Gini coefficient} = (2 \times \text{ROC index}) - 1$$

- The **Kolmogorov-Smirnov statistic** (K-S statistic) is another performance measure that captures the separation between the distribution of prediction scores for the different target levels in a classification problem.

- To calculate the K-S statistic, we first determine the cumulative probability distributions of the prediction scores for the positive and negative target levels:

$$CP(positive, ps) = \frac{\text{num positive test instances with score} \leq ps}{\text{num positive test instances}}$$

$$CP(negative, ps) = \frac{\text{num negative test instances with score} \leq ps}{\text{num negative test instances}}$$

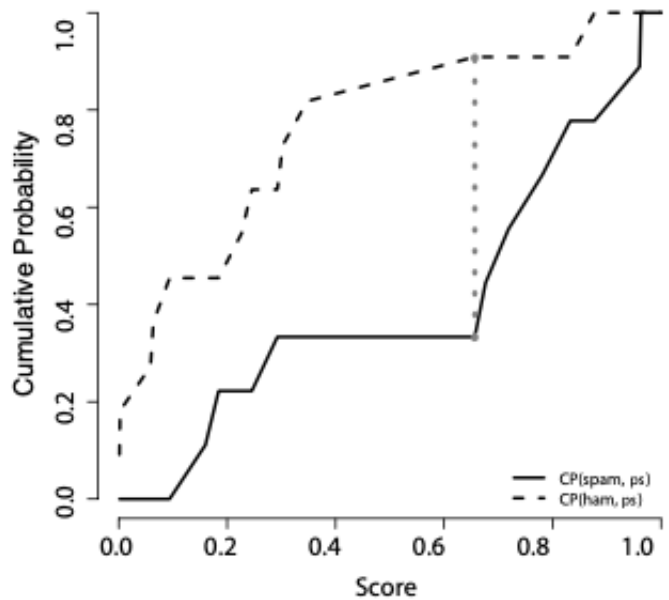


Figure: The K-S chart for the email classification predictions shown

- The K-S statistic is calculated by determining the maximum difference between the cumulative probability distributions for the positive and negative target levels.

$$\text{K-S} = \max_{ps} \left(CP(positive, ps) - CP(negative, ps) \right)$$

ID	Prediction Score	Positive (*'spam'*) Cumulative Count	Negative (*'ham'*) Cumulative Count	Positive (*'spam'*) Cumulative Probability	Negative (*'ham'*) Cumulative Probability	Distance
7	0.001	0	1	0.000	0.091	0.091
11	0.003	0	2	0.000	0.182	0.182
15	0.059	0	3	0.000	0.273	0.273
13	0.064	0	4	0.000	0.364	0.364
19	0.094	0	5	0.000	0.455	0.455
12	0.160	1	5	0.111	0.455	0.343
2	0.184	2	5	0.222	0.455	0.232
3	0.226	2	6	0.222	0.545	0.323
16	0.246	2	7	0.222	0.636	0.414
1	0.293	3	7	0.333	0.636	0.303
5	0.302	3	8	0.333	0.727	0.394
14	0.348	3	9	0.333	0.818	0.485
17	**0.657**	**3**	**10**	**0.333**	**0.909**	**0.576***
8	0.676	4	10	0.444	0.909	0.465
6	0.719	5	10	0.556	0.909	0.354
10	0.781	6	10	0.667	0.909	0.242
18	0.833	7	10	0.778	0.909	0.131
20	0.877	7	11	0.778	1.000	0.222
9	0.960	8	11	0.889	1.000	0.111
4	0.963	9	11	1.000	1.000	0.000

Decision Tree

(a) Brian (b) John (c) Aphra (d) Aoife

Figure: Cards showing character faces and names for the *Guess-Who* game

.

Man	Long Hair	Glasses	Name
Yes	No	Yes	Brian
Yes	No	No	John
No	Yes	No	Aphra
No	No	No	Aoife

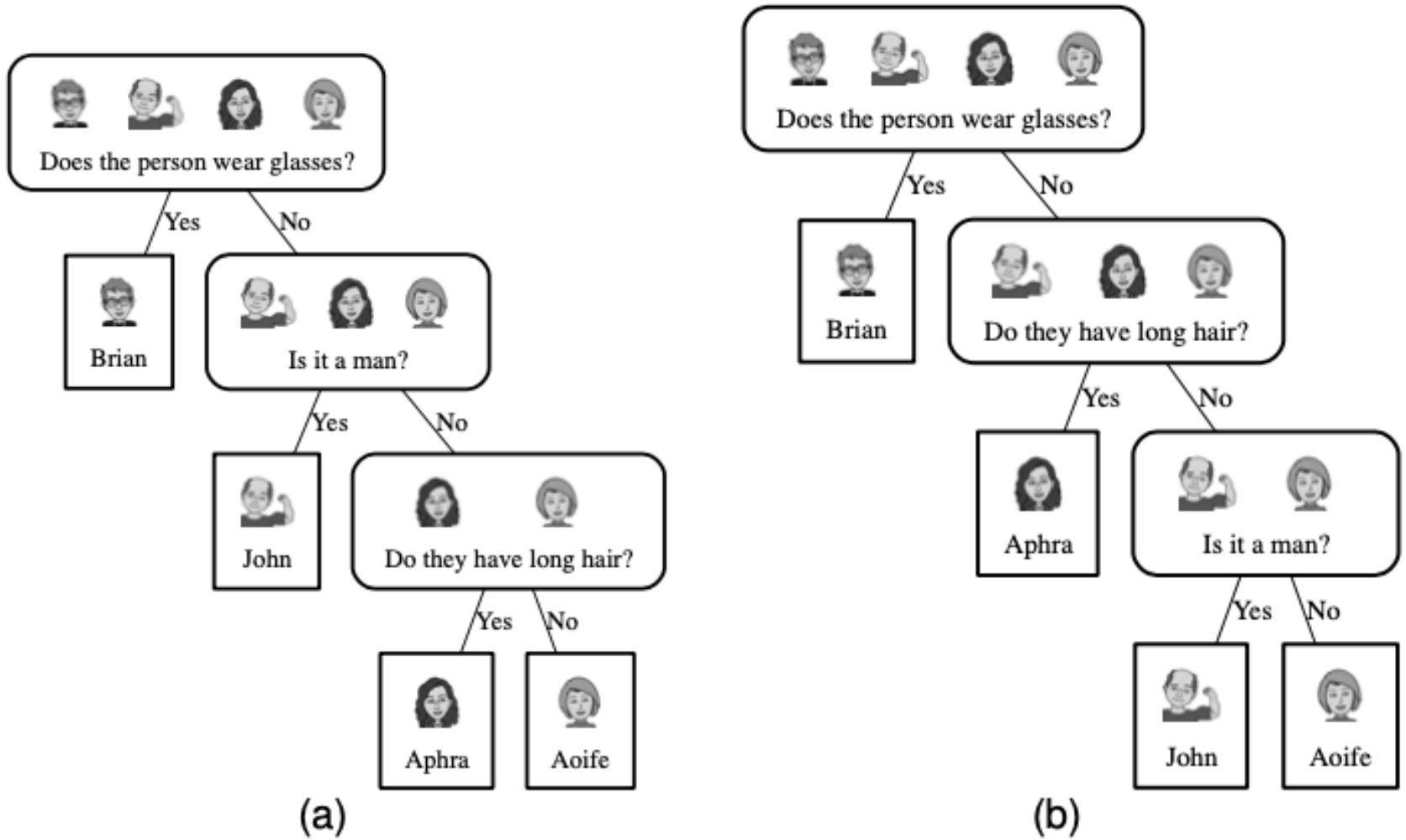
(a) Brian (b) John (c) Aphra (d) Aoife

Figure: Cards showing character faces and names for the *Guess-Who* game

.

Which question would you ask first:

1. Is it a man?
2. Does the person wear glasses?


Does the person wear glasses?
Yes
No
Brian
Is it a man?
Yes
No
John
Do they have long hair?
Yes
No
Aphra
Aoife
(a)
Does the person wear glasses?
Yes
No
Brian
Do they have long hair?
Yes
No
Aphra
Is it a man?
Yes
No
John
Aoife
(b)

- In both of the diagrams:
  - one path is 1 question long,
  - one path is 2 questions long,
  - and two paths are 3 questions long.
- Consequently, if you ask Question (2) first the average number of questions you have to ask per game is:

$$\frac{1+2+3+3}{4} = 2.25$$

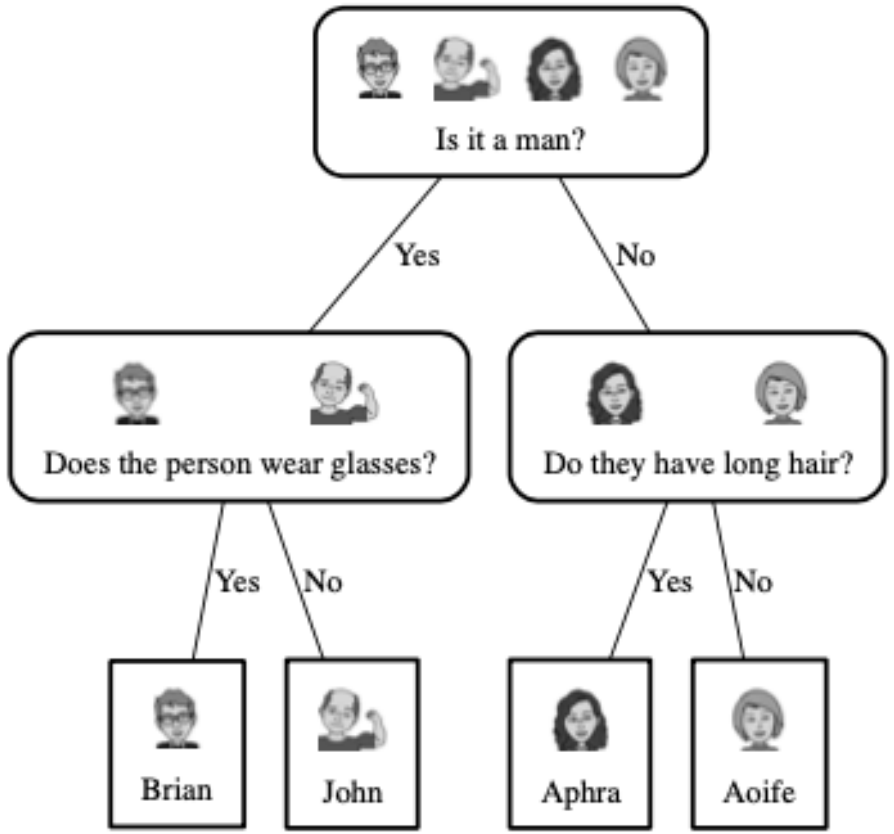


Figure: The different question sequences that can follow in a game of *Guess-Who* beginning with the question Is it a man?

- All the paths in this diagram are two questions long.
- So, on average if you ask Question (1) first the average number of questions you have to ask per game is:

$$\frac{2+2+2+2}{4} = 2$$

- On average getting an answer to Question (1) seems to give you more information than an answer to Question (2): less follow up questions.
- This is not because of the literal message of the answers: YES or NO.
- It is to do with how the answer to each questions splits the domain into different sized sets based on the value of the descriptive feature the question is asked about and the likelihood of each possible answer to the question.

Big Idea

- So the big idea here is to figure out which features are the most informative ones to ask questions about by considering the effects of the different answers to the questions, in terms of:
  1. how the domain is split up after the answer is received,
  2. and the likelihood of each of the answers.

- A decision tree consists of:
  1. a **root node** (or starting node),
  2. **interior nodes**
  3. and **leaf nodes** (or terminating nodes).
- Each of the non-leaf nodes (root and interior) in the tree specifies a test to be carried out on one of the query's descriptive features.
- Each of the leaf nodes specifies a predicted classification for the query.

Table: An email spam prediction dataset.

ID	Suspicious Words	Unknown Sender	Contains Images	Class
376	true	false	true	spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

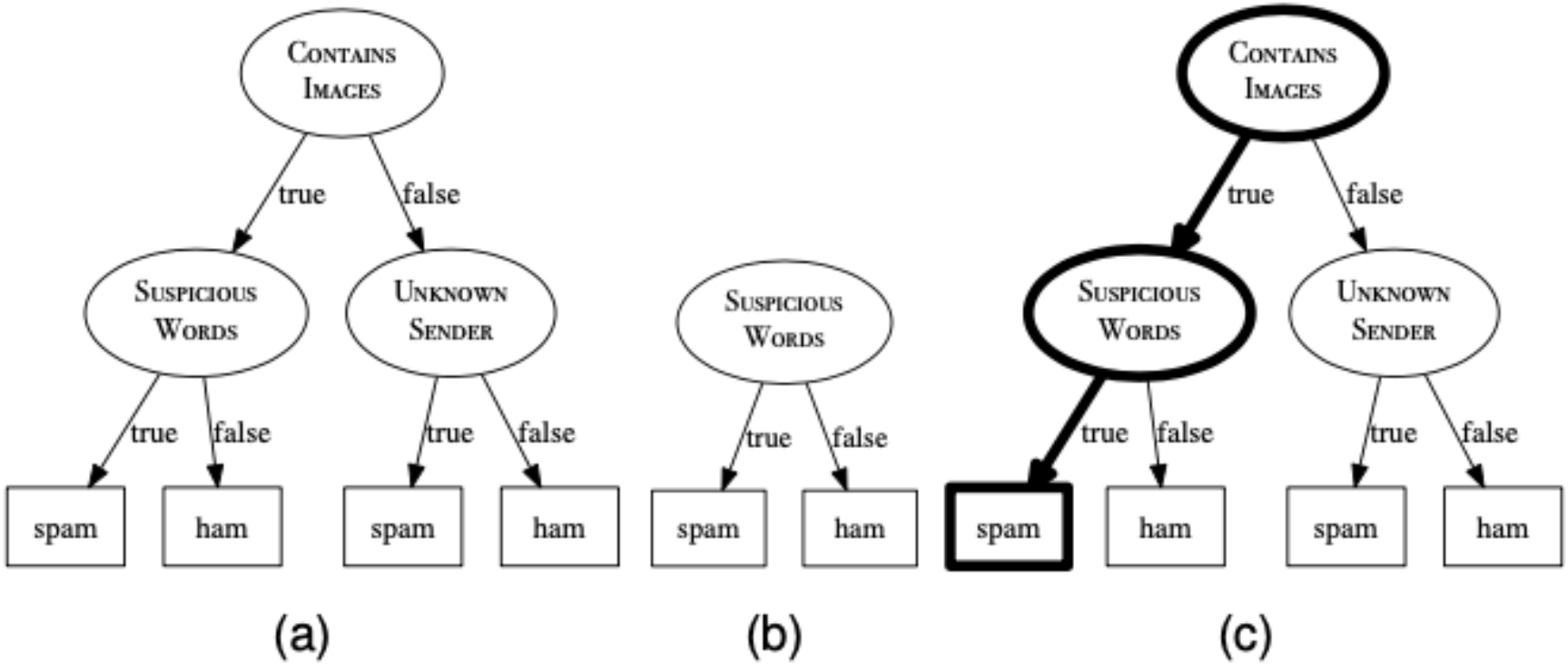


Figure: (a) and (b) show two decision trees that are consistent with the instances in the spam dataset. (c) shows the path taken through the tree shown in (a) to make a prediction for the query instance: SUSPICIOUS WORDS = *'true'*, UNKNOWN SENDER = *'true'*, CONTAINS IMAGES = *'true'*.

- Both of these trees will return identical predictions for all the examples in the dataset.
- So, which tree should we use?

- Apply the same approach as we used in the *Guess-Who* game: prefer decision trees that use less tests (shallower trees).
- This is an example of Occam's Razor.

How do we create shallow trees?

- The tree that tests SUSPICIOUS WORDS at the root is very shallow because the SUSPICIOUS WORDS feature perfectly splits the data into pure groups of *'spam'* and *'ham'*.
- Descriptive features that split the dataset into pure sets with respect to the target feature provide information about the target feature.
- So we can make shallow trees by testing the informative features early on in the tree.
- All we need to do that is a computational metric of the purity of a set: entropy

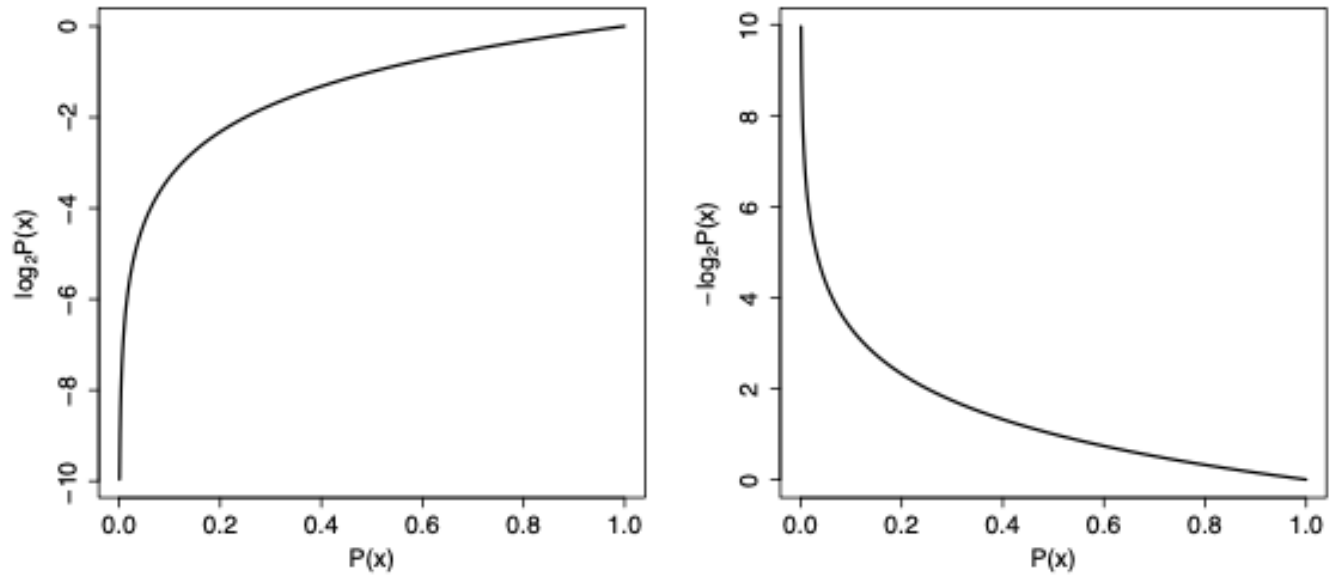
- Claude Shannon's entropy model defines a computational measure of the impurity of the elements of a set.
- An easy way to understand the entropy of a set is to think in terms of the uncertainty associated with guessing the result if you were to make a random selection from the set.

- Entropy is related to the probability of a outcome.
  - High probability $\rightarrow$ Low entropy
  - Low probability $\rightarrow$ High entropy
- If we take the log of a probability and multiply it by -1 we get this mapping!

What is a log?

Remember the *log* of a to the base b is the number to which we must raise b to get a.

- $log_2(0.5) = -1$ because $2^{-1} = 0.5$
- $log_2(1) = 0$ because $2^0 = 1$
- $log_2(8) = 3$ because $2^3 = 8$
- $log_5(25) = 2$ because $5^2 = 25$
- $log_5(32) = 2.153$ because $5^{2.153} = 32$



(a) (b)

Figure: (a) A graph illustrating how the value of a binary log (the log to the base 2) of a probability changes across the range of probability values. (b) the impact of multiplying these values by -1.

- Shannon's model of entropy is a weighted sum of the logs of the probabilities of each of the possible outcomes when we make a random selection from a set.

$$H(t) = -\sum_{i=1}^{I} (P(t = i) \times log_s(P(t = i))) \quad (1)$$

Table: The relationship between the entropy of a message and the set it was selected from.

Entropy of a Message	Properties of the Message Set
High	A large set of equally likely messages.
Medium	A large set of messages, some more likely than others.
Medium	A small set of equally likely messages.
Low	A small set of messages with one very likely message.

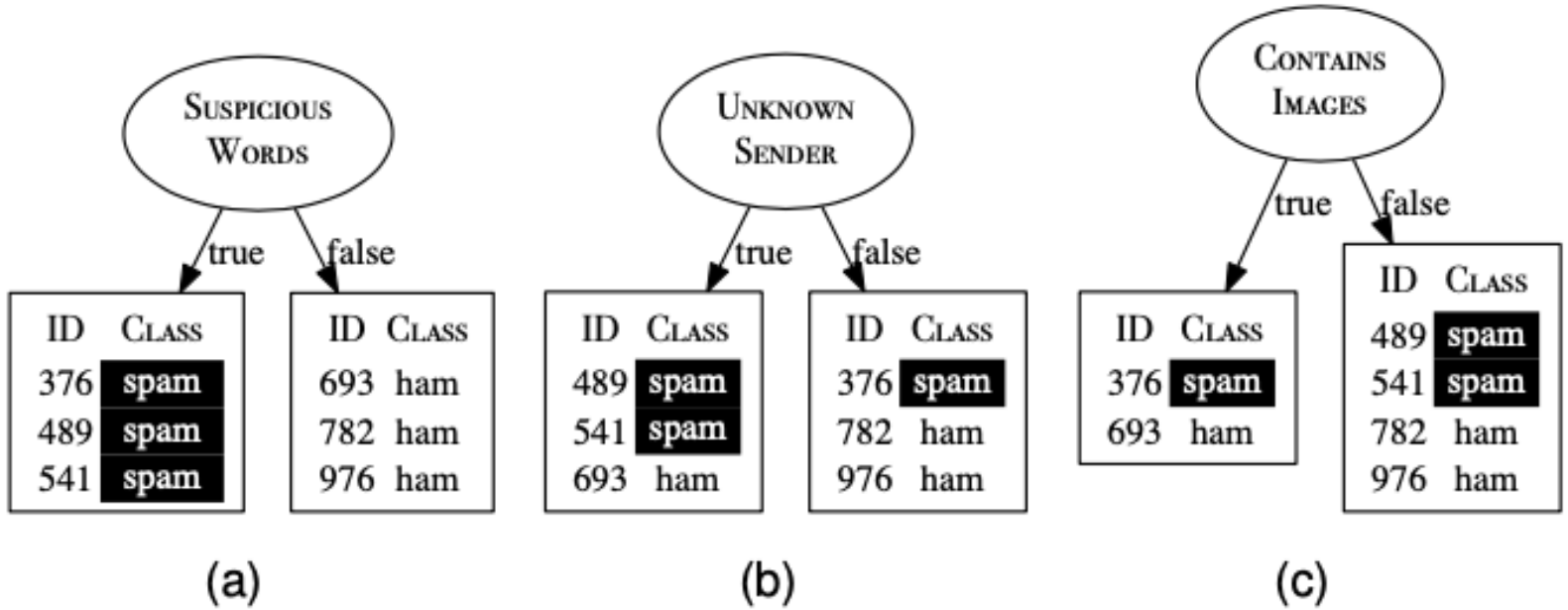


Figure: How the instances in the spam dataset split when we partition using each of the different descriptive features from the spam dataset in Table 1 [15]

- Our intuition is that the ideal discriminatory feature will partition the data into **pure** subsets where all the instances in each subset have the same classification.
  - SUSPICIOUS WORDS perfect split.
  - UNKNOWN SENDER mixture but some information (when *'true'* most instances are *'spam'*).
  - CONTAINS IMAGES no information.
- One way to implement this idea is to use a metric called **information gain**.

Information Gain

- The information gain of a descriptive feature can be understood as a measure of the reduction in the overall entropy of a prediction task by testing on that feature.

Decision Trees: Advantages

- interpretable.
- handle both categorical and continuous descriptive features.
- has the ability to model the interactions between descriptive features (diminished if **pre-pruning** is employed)
- relatively, robust to the **curse of dimensionality**.
- relatively, robust to noise in the dataset if **pruning** is used.

Decision Tress: Potential Disadvantages

- trees become large when dealing with continuous features.
- decision trees are very expressive and sensitive to the dataset, as a result they can overfit the data if there are a lot of features (curse of dimensionality)
- eager learner (concept drift).