Lecture 9: Feature Selection and Analysis

COMP90049
Introduction to Machine Learning

Semester 1, 2024

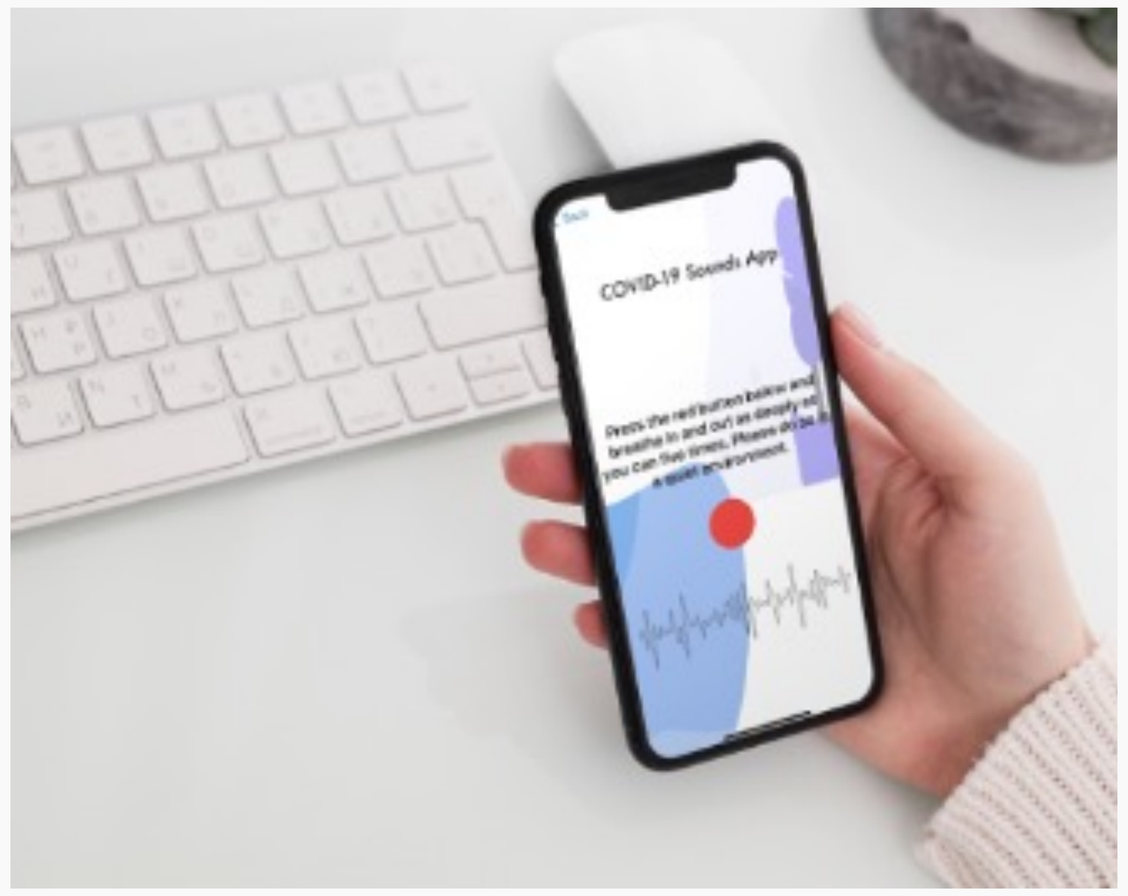
Ting Dang, CIS

Acknowledgement: Jeremy Nicholson, Tim Baldwin & Karin Verspoor

About me

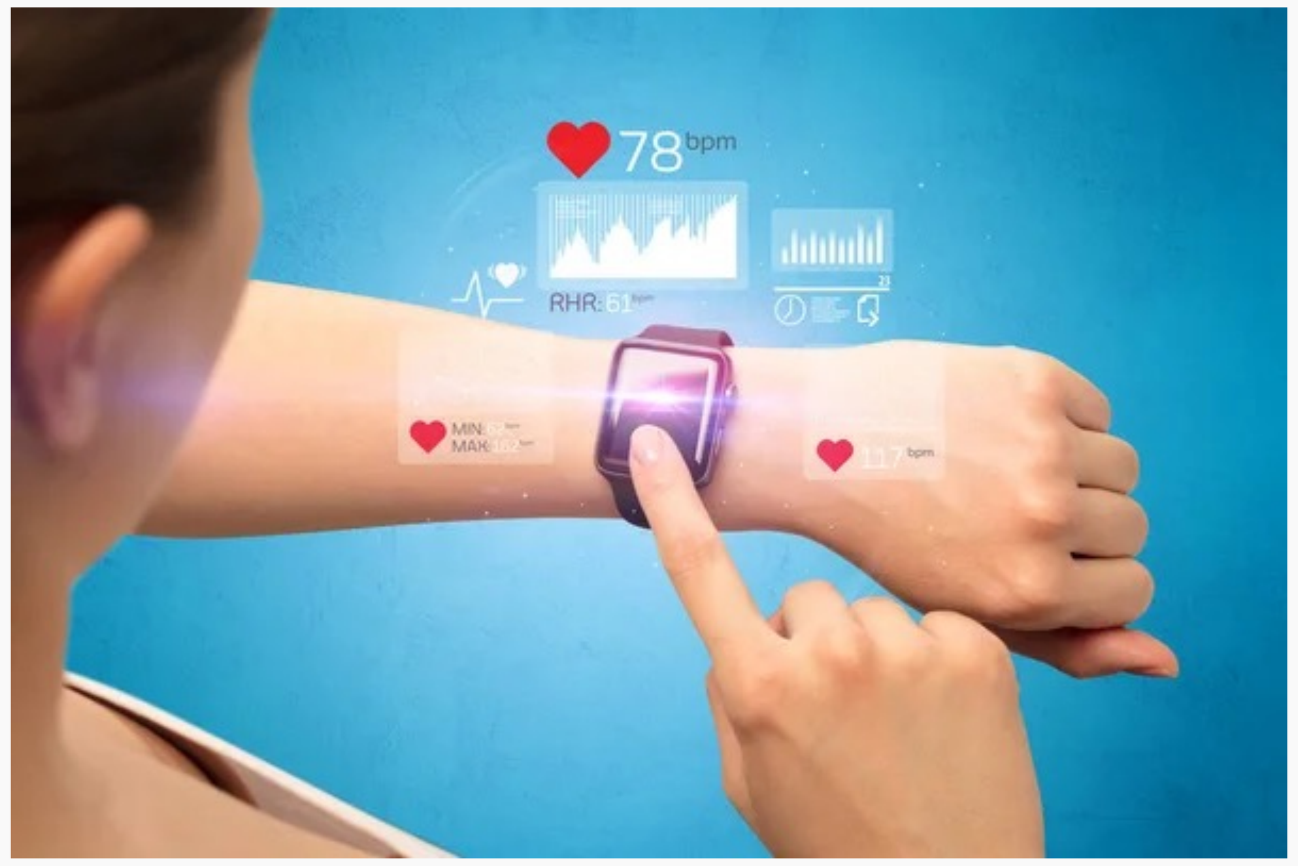
I am a Senior Lecturer in the School of Computing and Information Systems at the University of Melbourne. Previously, I was a Senior Research Scientist in Bell Labs (UK), and a Senior Research Associate at the University of Cambridge (UK) and RA at UNSW, where I did my PhD.

Digital Health via Acoustic Sensing and Analysis



Large scale screening

AI in mental health

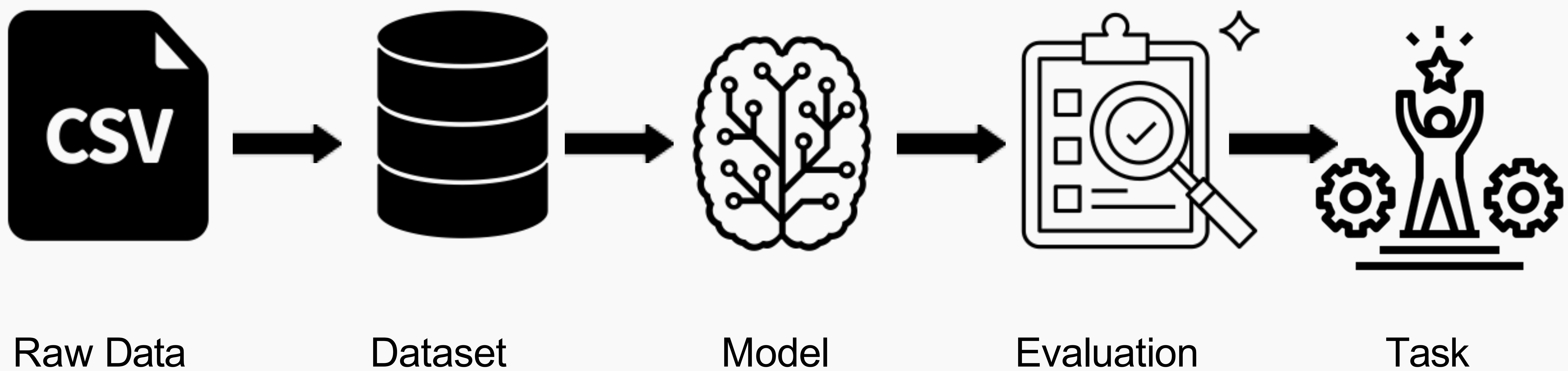


Wearable sensing

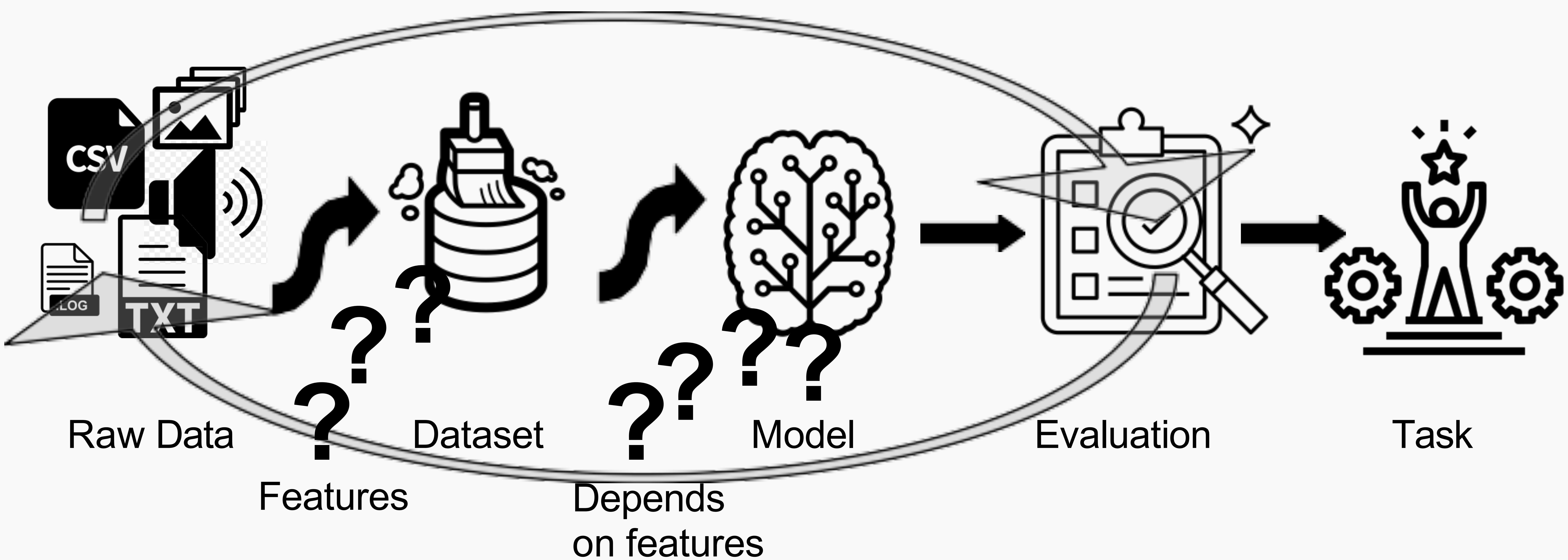
Features in Machine Learning

Machine Learning Workflow

The Dream



Reality



Data Preparation vs Feature Selection

GIGO: Garbage In, Garbage Out

Data Preparation and Cleansing (discussed before)

- Data Cleaning
- Data Aggregation
- Dealing with missing values
- Transformation (e.g., log transform)
- Binarization
- Binning
- Scaling or Normalization

Feature Selection (this lecture)

- Wrapper methods (aka recursive elimination)
- Filtering (aka univariate filtering)
- Glance into some other common approaches

THE UNIVERSITY OF MELBOURNE

Data Preparation vs Feature Selection

Our job as Machine Learning experts:

- Inspect / clean the data
- Choose a model suitable for classifying the data according to the attributes
- Choose attributes suitable for classifying the data according to the model

    - Inspection
    - Intuition

Data Preparation vs Feature Selection

Our job as Machine Learning experts:

- Inspect / clean the data
- Choose a model suitable for classifying the data according to the attributes
- Choose attributes suitable for classifying the data according to the model

  - ~~Inspection~~
  - ~~Intuition~~
  - Neither possible in practice

Feature Selection

What makes features good?

Lead to better models

- Better performance according to some evaluation metric

Side-goal 1

- Seeing important features can suggest other important features
- Tell us interesting things about the problem

Side-goal 2

- Fewer features → smaller models → faster answer
  - More accurate answer >> faster answer

Choosing a good feature set

"Wrapper" methods

- Choose subset of attributes that give best performance on the development data

- For example: for the Weather data set:

  - Train model on {*Outlook*}
  - Train model on {*Temperature*}
  - ...
  - Train model on {*Outlook*, *Temperature*}
  - ...
  - Train model on {*Outlook*, *Temperature*, *Humidity*}
  - ...
  - Train model on {*Outlook*, *Temperature*, *Humidity*, *Windy*}

Choosing a good feature set

"Wrapper" methods

- Choose subset of attributes that give best performance on the development data

- For example: for the Weather data set:

    - Evaluate model on {*Outlook*}
    - Evaluate model on {*Temperature*}
    - ...
    - Evaluate model on {*Outlook, Temperature*}
    - ...
    - Evaluate model on {*Outlook, Temperature, Humidity*}
    - ...
    - Evaluate model on {*Outlook, Temperature, Humidity, Windy*}

- Best performance on data set → best feature set

Choosing a good feature set

“Wrapper” methods

- Choose subset of attributes that give best performance on the development data
- Advantages:
  - Feature set with optimal performance on development data
- Disadvantages:
  - Takes a **long** time

Aside: how long does the full wrapper method take?

Assume we have a fast method (e.g. Naive Bayes) over a data set of non-trivial size ($\sim$10K instances):

- Assume: train–evaluate cycle takes 10 sec to complete

How many cycles? For *m* features:

- 2^m subsets $= \frac{2^m}{6}$ minutes
- $m = 10 \rightarrow$ 3 hours
- $m = 60 \rightarrow$ heat death of universe

Only practical for very small data sets.

More practical wrapper methods: Greedy search

Greedy approach

- Train and evaluate model on each single attribute
- Choose best attribute
- Until convergence:
  - Train and evaluate model on best attribute(s), plus each remaining single attribute
  - Choose best attribute out of the remaining set
- Iterate until performance (e.g. accuracy) stops increasing

More practical wrapper methods: Greedy search

Greedy approach

- Bad news:
  - Takes $\frac{1}{2}m^2$ cycles, for m attributes
  - In theory, 386 attributes → days
- Good news:
  - In practice, converges much more quickly than this
- Bad news again:
  - Convergences to a sub-optimal (and often very bad) solution

More practical wrapper methods: Ablation

"Ablation" approach

- Start with all attributes
- Remove one attribute, train and evaluate model
- Until divergence:
  - From remaining attributes, remove each attribute, train and evaluate model
  - Remove attribute that causes least performance degredation
- Termination condition usually: performance (e.g. accuracy) starts to degrade by more than g

More practical wrapper methods: Ablation

"Ablation" approach

for example:

- Start with all features
  - Train, evaluate model on $\{\text{Outlook},\ \text{Temperature},\ \text{Humidity},\ \text{Windy}\}$
- Consider feature subsets of size 3:
  - Train, evaluate model on $\{\text{Outlook},\ \text{Temperature},\ \text{Humidity}\}$
  - Train, evaluate model on $\{\text{Outlook},\ \text{Temperature},\ \text{Windy}\}$
  - Train, evaluate model on $\{\text{Outlook},\ \text{Humidity},\ \text{Windy}\}$
  - Train, evaluate model on $\{\text{Temperature},\ \text{Humidity},\ \text{Windy}\}$
- Choose best of previous five (let's say THW):
- Consider feature subsets of size 2:
  - Train, evaluate model on $\{\text{Temperature},\ \text{Humidity}\}$
  - Train, evaluate model on $\{\text{Temperature},\ \text{Windy}\}$
  - Train, evaluate model on $\{\text{Humidity},\ \text{Windy}\}$
- etc...

More practical wrapper methods: Ablation

"Ablation" approach

- Good news:
  - Mostly removes irrelevant attributes (at the start)
- Bad news:
  - Assumes independence of attributes
    (Actually, both approaches do this)
  - Takes $O(m^2)$ time; cycles are slower with more attributes
  - Not feasible on non-trivial data sets.

Filtering methods

Feature filtering

Intuition: Evaluate the “goodness” of each feature, separate from other features

- Consider each feature separately: linear time in number of attributes
- Possible (but difficult) to control for inter-dependence of features
- Typically most popular strategy

Feature "goodness"

What makes a ~~feature set~~ single feature good?

- ~~Better models!~~
- Well correlated with class

Toy example

a_1	a_2	c
Y	Y	Y
Y	N	Y
N	Y	N
N	N	N

Which of a_1, a_2 is good?

Toy example

a_1	a_2	c
Y	Y	Y
Y	N	Y
N	Y	N
N	N	N

a_1 is probably good.

Toy example

a_1	a_2	c
Y	Y	Y
Y	N	Y
N	Y	N
N	N	N

a_2 is probably not good.

THE UNIVERSITY OF MELBOURNE

Pointwise Mutual Information (PMI)

Discrepancy between the **observed joint distribution** of two random variables *A* and *C* and the expected joint distribution **if *A* and *C* were independent.**

Recall independence: $P(C|A) = P(C)$

$$P(A,C) = P(A)P(C)$$

Pointwise Mutual Information

Discrepancy between the **observed joint distribution** of two random variables *A* and *C* and the expected joint distribution **if *A* and *C* were independent.**

Recall independence: $P(C|A) = P(C)$

$$P(A,C) = P(A)P(C)$$

PMI is defined as:

$$PMI(A,C) = \log_2 \frac{P(A,C)}{P(A)P(C)}$$

We want to find attributes that are **not** independent of the class.

- If PMI >> 0, attribute and class occur together much more often than randomly.
- If PMI ~0, attribute and class occur together as often as we would expect from random chance
- If PMI << 0, attribute and class are negatively correlated. (More on that later!)

Attributes with greatest PMI: best attributes

Toy example, revisited

a_1	a_2	c
Y	Y	Y
Y	N	Y
N	Y	N
N	N	N

Calculate PMI of a_1, a_2 with respect to c

Toy example, revisited

a_1	a_2	c
Y	Y	Y
Y	N	Y
N	Y	N
N	N	N

Calculate PMI of a_1 with respect to c

$$P(a_1 = Y) = \frac{2}{4} = 0.5$$

$$P(c = Y) = \frac{2}{4} = 0.5$$

$$P(a_1 = Y, c = Y) = \frac{2}{4} = 0.5$$

$$PMI(a_1, c = Y) = log_2 \frac{0.5}{0.5 \ * 0.5} = 1 > 0$$

Good one!

Toy example, revisited

a_1	a_2	c
Y	Y	Y
Y	N	Y
N	Y	N
N	N	N

Calculate PMI of a_2 with respect to c

$$P(a_2 = Y) = \frac{2}{4} = 0.5$$

$$P(c = Y) = \frac{2}{4} = 0.5$$

$$P(a_2 = Y, c = Y) = \frac{1}{4} = 0.25$$

$$PMI(a_2, c = Y) = log_2 \frac{0.25}{0.5\ * 0.5} = 0$$

Random chance!

Feature "goodness", revisited

What makes a single feature good?

- Well correlated with class
  - Knowing a lets us predict c with more confidence
- Reverse correlated with class
  - Knowing $\bar{a}$ lets us predict c with more confidence
- Well correlated (or reverse correlated) with not class
  - Knowing a lets us predict $\bar{c}$ with more confidence
  - Usually not quite as good, but still useful

Mutual Information

- Expected value of PMI over all possible events
- For our example: Combine PMI of all possible combinations: a, $\bar{a}$, c, $\bar{c}$

Aside: Contingency tables

Contigency tables: compact representation of these frequency counts

	$a(Y)$	$\bar{a}(N)$	$Total$
$c(Y)$	$\sigma(a, c)$	$\sigma(\bar{a}, c)$	$\sigma(c)$
$\bar{c}(N)$	$\sigma(a, \bar{c})$	$\sigma(\bar{a}, \bar{c})$	$\sigma(\bar{c})$
$Total$	$\sigma(a)$	$\sigma(\bar{a})$	N

$P(a, c) = \frac{\sigma(a,c)}{N}$, etc.

Aside: Contingency tables

a_1	a_2	c
Y	Y	Y
Y	N	Y
N	Y	N
N	N	N

Contingency tables for toy example:

	a_1 =Y	a_1 =N	Total
c =Y	2	0	2
c =N	0	2	2
Total	2	2	4

	a_2 =Y	a_2 =N	Total
c =Y	1	1	2
c =N	1	1	2
Total	2	2	4

Mutual Information

Combine PMI of all possible combinations: $a, \bar{a}, c, \bar{c}$

$$MI(A,C) = P(a,c)PMI(a,c) + P(\bar{a},c)PMI(\bar{a},c) + P(a,\bar{c})PMI(a,\bar{c}) + P(\bar{a},\bar{c})PMI(\bar{a},\bar{c})$$

$$MI(A,C) = P(a,c)log_2\frac{P(a,c)}{P(a)P(c)} + P(\bar{a},c)log_2\frac{P(\bar{a},c)}{P(\bar{a})P(c)} + P(a,\bar{c})log_2\frac{P(a,\bar{c})}{P(a)P(\bar{c})} + P(\bar{a},\bar{c})log_2\frac{P(\bar{a},\bar{c})}{P(\bar{a})P(\bar{c})}$$

Mutual Information

Combine PMI of all possible combinations: $a, \bar{a}, c, \bar{c}$

$$MI(A,C) = P(a,c)PMI(a,c) + P(\bar{a},c)PMI(\bar{a},c)$$
$$+P(a,\bar{c})PMI(a,\bar{c}) + P(\bar{a},\bar{c})PMI(\bar{a},\bar{c})$$

$$MI(A,C) = P(a,c)log_2\frac{P(a,c)}{P(a)P(c)} + P(\bar{a},c)log_2\frac{P(\bar{a},c)}{P(\bar{a})P(c)}$$
$$+P(a,\bar{c})log_2\frac{P(a,\bar{c})}{P(a)P(\bar{c})} + P(\bar{a},\bar{c})log_2\frac{P(\bar{a},\bar{c})}{P(\bar{a})P(\bar{c})}$$

Often written more compactly as:

$$MI(A,C) = \sum_{i\in\{a,\bar{a}\}} \sum_{j\in\{c,\bar{c}\}} P(i,j)log_2\frac{P(i,j)}{P(i)P(j)}$$

We define that $0 \log 0 \equiv 0$.

Mutual Information Example

Contingency Table for attribute a_1

	a_1 =Y	a_1 =N	Total
c =Y	2	0	2
c =N	0	2	2
Total	2	2	4

Mutual Information Example

Contingency Table for attribute a_1

	a_1 =Y	a_1 =N	Total
c =Y	2	0	2
c =N	0	2	2
Total	2	2	4

$P(a_1, c) = \frac{2}{4}$; $P(\bar{a}_1, c) = 0$; $P(a_1, \bar{c}) = 0$; $P(\bar{a}_1, \bar{c}) = \frac{2}{4}$

$P(a_1) = \frac{2}{4}$; $P(\bar{a}_1) = \frac{2}{4}$; $P(c) = \frac{2}{4}$; $P(\bar{c}) = \frac{2}{4}$

Mutual Information Example

Contingency Table for attribute a_1

	a_1 =Y	a_1 =N	Total
c =Y	2	0	2
c =N	0	2	2
Total	2	2	4

$P(a_1, c) = \frac{2}{4}$; $P(\bar{a}_1, c) = 0$; $P(a_1, \bar{c}) = 0$; $P(\bar{a}_1, \bar{c}) = \frac{2}{4}$

$P(a_1) = \frac{2}{4}$; $P(\bar{a}_1) = \frac{2}{4}$; $P(c) = \frac{2}{4}$; $P(\bar{c}) = \frac{2}{4}$

$$MI(A_1, C) = P(a_1, c) log_2 \frac{P(a_1, c)}{P(a_1)P(c)} + P(\bar{a}_1, c) log_2 \frac{P(\bar{a}_1, c)}{P(\bar{a}_1)P(c)}$$
$$+ P(a_1, \bar{c}) log_2 \frac{P(a_1, \bar{c})}{P(a_1)P(\bar{c})} + P(\bar{a}_1, \bar{c}) log_2 \frac{P(\bar{a}_1, \bar{c})}{P(\bar{a}_1)P(\bar{c})}$$

Mutual Information Example

Contingency Table for attribute a_1

	a_1 =Y	a_1 =N	Total
c =Y	2	0	2
c =N	0	2	2
Total	2	2	4

$P(a_1, c) = \frac{2}{4}$; $P(\bar{a}_1, c) = 0$; $P(a_1, \bar{c}) = 0$; $P(\bar{a}_1, \bar{c}) = 0$

$P(a_1) = \frac{2}{4}$; $P(\bar{a}_1) = \frac{2}{4}$; $P(c) = \frac{2}{4}$; $P(\bar{c}) = \frac{2}{4}$

$$MI(A_1, C) = P(a_1, c) log_2 \frac{P(a_1, c)}{P(a_1)P(c)} + P(\bar{a}_1, c) log_2 \frac{P(\bar{a}_1, c)}{P(\bar{a}_1)P(c)}$$

$$+ P(a_1, \bar{c}) log_2 \frac{P(a_1, \bar{c})}{P(a_1)P(\bar{c})} + P(\bar{a}_1, \bar{c}) log_2 \frac{P(\bar{a}_1, \bar{c})}{P(\bar{a}_1)P(\bar{c})}$$

$$= \frac{1}{2} log_2 \frac{\frac{1}{2}}{\frac{1}{2} * \frac{1}{2}} + 0 log_2 \frac{0}{\frac{1}{2} * \frac{1}{2}} + 0 log_2 \frac{0}{\frac{1}{2} * \frac{1}{2}} + \frac{1}{2} log_2 \frac{\frac{1}{2}}{\frac{1}{2} * \frac{1}{2}}$$

$$= \frac{1}{2} + 0 + 0 + \frac{1}{2} = 1$$

Mutual Information Example continued

Contingency Table for attribute a_2

	a_2 =Y	a_2=N	Total
c =Y	1	1	2
c =N	1	1	2
Total	2	2	4

Mutual Information Example continued

Contingency Table for attribute a_2

	a_2 =Y	a_2=N	Total
c =Y	1	1	2
c =N	1	1	2
Total	2	2	4

$$P(a,c) = \frac{1}{4}\ ;\ P(\bar{a},c) = \frac{1}{4};\ P(a,\bar{c}) = \frac{1}{4};\ P(\bar{a},\bar{c}) = \frac{1}{4}$$

$$P(a) = \frac{2}{4}\ ; \quad P(\bar{a}) = \frac{2}{4}; \quad P(c) = \frac{2}{4}; \quad P(\bar{c}) = \frac{2}{4}$$

Mutual Information Example continued

Contingency Table for attribute a_2

	a_2 =Y	a_2=N	Total
c =Y	1	1	2
c =N	1	1	2
Total	2	2	4

$$P(a_2, c) = \frac{1}{4}\,;\ P(\bar{a}_2, c) = \frac{1}{4};\ P(a_2, \bar{c}) = \frac{1}{4};\ P(\bar{a}_2, \bar{c}) = \frac{1}{4}$$

$$P(a_2) = \frac{2}{4}\,;\quad P(\bar{a}_2) = \frac{2}{4};\quad P(c) = \frac{2}{4};\quad P(\bar{c}) = \frac{2}{4}$$

$$MI(A_2, C) = P(a_2, c) log_2 \frac{P(a_2, c)}{P(a_2)P(c)} + P(\bar{a}_2, c) log_2 \frac{P(\bar{a}_2, c)}{P(\bar{a}_2)P(c)}$$

$$+ P(a_2, \bar{c}) log_2 \frac{P(a_2, \bar{c})}{P(a_2)P(\bar{c})} + P(\bar{a}_2, \bar{c}) log_2 \frac{P(\bar{a}_2, \bar{c})}{P(\bar{a}_2)P(\bar{c})}$$

$$= \frac{1}{4} log_2 \frac{\frac{1}{4}}{\frac{1}{2} * \frac{1}{2}} + \frac{1}{4} log_2 \frac{\frac{1}{4}}{\frac{1}{2} * \frac{1}{2}} + \frac{1}{4} log_2 \frac{\frac{1}{4}}{\frac{1}{2} * \frac{1}{2}} + \frac{1}{4} log_2 \frac{\frac{1}{4}}{\frac{1}{2} * \frac{1}{2}}$$

$$= \frac{1}{4} * 0 + \frac{1}{4} * 0 + \frac{1}{4} * 0 + \frac{1}{4} * 0 = 0$$

Chi-square χ^2

Statistical method used to determine if there is a significant association between two variables in a contingency table

	$a(Y)$	$\bar{a}(N)$	$Total$
$c(Y)$	$\sigma(a,c)$	$\sigma(\bar{a},c)$	$\sigma(c)$
$\bar{c}(N)$	$\sigma(a,\bar{c})$	$\sigma(\bar{a},\bar{c})$	$\sigma(\bar{c})$
$Total$	$\sigma(a)$	$\sigma(\bar{a})$	N

Contingency table (shorthand):

	$a(Y)$	$\bar{a}(N)$	$Total$
$c(Y)$	W	X	$W + X$
$\bar{c}(N)$	Y	Z	$Y + Z$
$Total$	$W + Y$	$X + Z$	$N = W + X + Y + Z$

If a, c were independent (uncorrelated), what value would you expect in W?

Denote the expected value as E(W).

Chi-square χ^2

	$a(Y)$	$\bar{a}(N)$	$Total$
$c(Y)$	W	X	$W+X$
$\bar{c}(N)$	Y	Z	$Y+Z$
$Total$	$W+Y$	$X+Z$	$N=W+X+Y+Z$

Independence assumption

If a, c were independent, then $P(a,c) = P(a)P(c)$

$$P(a,c) = P(a)P(c)$$

$$\downarrow \qquad\qquad \downarrow$$

$$\frac{\sigma(a,c)}{N} = \frac{\sigma(a)}{N}\frac{\sigma(c)}{N}$$

$$\sigma(a,c) = \frac{\sigma(a)\sigma(c)}{N}$$

$$E(W) = \frac{(W+Y)(W+X)}{W+X+Y+Z}$$

Chi-square χ^2

Compare the value we actually observed $O(W)$ **with the expected value** $E(W)$:

- If the **observed value is much greater than the expected value**, a occurs more often with c than we would expect at random — **predictive**
- If the observed value is **much smaller than the expected value**, a occurs less often with c than we would expect at random — **predictive**
- If the **observed value is close to the expected value**, a occurs as often with c as we would expect randomly — **not predictive**

Similarly with X, Y, Z

	$a(Y)$	$\bar{a}(N)$	$Total$
$c(Y)$	W	X	$W+X$
$\bar{c}(N)$	Y	Z	$Y+Z$
$Total$	$W+Y$	$X+Z$	$N=W+X+Y+Z$

Chi-square χ^2

Actual calculation (to fit to a chi-square distribution)

$$\chi^2 = \frac{(O(W) - E(W))^2}{E(W)} + \frac{(O(X) - E(X))^2}{E(X)} + \frac{(O(Y) - E(Y))^2}{E(Y)} + \frac{(O(Z) - E(Z))^2}{E(Z)}$$

$$= \sum_{i=1}^{a} \sum_{j=1}^{c} \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}}$$

- *i* sums over rows and *j* sums over columns.
- Because the values are squared, χ^2 becomes much greater when $|O - E|$ is large, even if *E* is also large.

Chi-square Example

Contingency table for toy example (observed values):

	a_1 =Y	a_1 =N	Total
c =Y	2	0	2
c =N	0	2	2
Total	2	2	4

Contingency table for toy example (expected values):

	a_1 =Y	a_1 =N	Total
c =Y	1	1	2
c =N	1	1	2
Total	2	2	4

THE UNIVERSITY OF MELBOURNE

Chi-square Example

$$
\begin{aligned}
\chi^2(A_1, C) &= \frac{(O_{a,c} - E_{a,c})^2}{E_{a,c}} + \frac{(O_{\bar{a},c} - E_{\bar{a},c})^2}{E_{\bar{a},c}} + \\
&\quad \frac{(O_{a,\bar{c}} - E_{a,\bar{c}})^2}{E_{a,\bar{c}}} + \frac{(O_{\bar{a},\bar{c}} - E_{\bar{a},\bar{c}})^2}{E_{\bar{a},\bar{c}}} \\
&= \frac{(2-1)^2}{1} + \frac{(0-1)^2}{1} + \frac{(0-1)^2}{1} + \frac{(2-1)^2}{1} \\
&= 1 + 1 + 1 + 1 = 4
\end{aligned}
$$

$\chi^2(A_2, C)$ is obviously 0, because all observed values are equal to expected values.

Common Issues

Types of Attribute

So far, we've only looked at binary (Y/N) attributes:

- Nominal attributes
- Continuous attributes
- Ordinal attributes

Types of Attributes: Nominal

Two common strategies

1. Treat as multiple binary attributes:
    - e.g. sunny=Y, overcast=N, rainy=N, etc.
    - Can just use the formulae as given
    - Results often difficult to interpret
        - For example, *Outlook=sunny(Y)* is useful, but *Outlook=overcast(N)* and *Outlook=rainy(N)* are not useful... Should we use Outlook?

Types of Attributes: Nominal

Two common strategies

1. Treat as multiple binary attributes:
   - e.g. sunny=Y, overcast=N, rainy=N, etc.
   - Can just use the formulae as given
   - Results often difficult to interpret
     - For example, *Outlook=sunny(Y)* is useful, but *Outlook=overcast(N)* and *Outlook=rainy(N)* are not useful... Should we use Outlook?

2. Modify contingency tables (and formulae)

	s	o	r
c =Y	U	V	W
c =N	X	Y	Z

Modified MI:

$$MI(A,C) = \sum_{i \in \{s,o,r\}} \sum_{j \in \{c,\bar{c}\}} P(i,j) log_2 \frac{P(i,j)}{P(i)P(j)}$$

$$= P(s,c)log_2 \frac{P(s,c)}{P(s)P(c)} + P(s,\bar{c})log_2 \frac{P(s,\bar{c})}{P(s)P(\bar{c})}$$

$$+P(o,c)log_2 \frac{P(o,c)}{PoP(c)} + P(o,\bar{c})log_2 \frac{P(o,\bar{c})}{P(o)P(\bar{c})}$$

$$+P(r,c)log_2 \frac{P(r,c)}{P(r)P(c)} + P(r,\bar{c})log_2 \frac{P(r,\bar{c})}{P(r)P(\bar{c})}$$

- Biased towards attributes with many values.

Types of Attributes: Nominal

Chi-square can be used as normal, with 6 observed/expected values.

- To control for score inflation, we need to consider “number of degrees of freedom”, and then use the significance test explicitly (beyond the scope of this subject)

Types of Attributes: Continuous

Continuous attributes

- Usually dealt with by estimating probability based on a Gaussian (normal) distribution
- With a large number of values, most random variables are normally distributed due to the **Central Limit Theorem**
- For small data sets or pathological features, we may need to use binomial/multinomial distributions

All of this is beyond the scope of this subject

Types of Attributes: Ordinal

Three possibilities, roughly in order of popularity:

1. Treat as binary
   - Particularly appropriate for frequency counts where events are low-frequency (e.g. words in tweets)
2. Treat as continuous
   - The fact that we haven't *seen* any intermediate values is usually not important
   - Does have all of the technical downsides of continuous attributes, however
3. Treat as nominal (i.e. throw away ordering)

Multi-class problems

So far, we've only looked at binary (c = Y/N) classification tasks.

Multiclass (e.g. LA, NY, C, At, SF) classification tasks are usually much more difficult.

Multi-class problems

Consider multi-class problem over LA, NY, C, At, SF:

- PMI, MI, χ^2 are all calculated *per-class*
- Some other feature selection metrics, e.g. Information Gain, work for all classes at once
- Need to make a point of selecting (hopefully uncorrelated) features for *each* class to give our classifier the best chance of predicting everything correctly.

Multi-class problems

Actual example (MI):

LA	NY	C	At	SF
la	nyc	chicago	atlanta	sf
angeles	york	bears	atl	httpdealnaycom
los	ny	il	ga	francisco
chicago	chicago	httpbitlyczmk	lol	san
hollywood	atlanta	cubs	u	u
atlanta	yankees	la	georgia	lol
lakers	sf	chi	chicago	save

Multi-class problems

Intuitive features:

LA	NY	C	At	SF
la	**nyc**	**chicago**	**atlanta**	**sf**
angeles	**york**	**bears**	**atl**	httpdealnaycom
los	**ny**	**il**	**ga**	**francisco**
chicago	chicago	httpbitlyczmk	lol	**san**
hollywood	atlanta	**cubs**	u	u
atlanta	**yankees**	la	**georgia**	lol
lakers	sf	**chi**	chicago	save

Multi-class problems

Features for predicting not class (MI):

LA	NY	C	At	SF
la	nyc	chicago	atlanta	sf
angeles	york	bears	atl	httpdealnaycom
los	ny	il	ga	francisco
chicago	**chicago**	httpbitlyczmk	lol	san
hollywood	**atlanta**	cubs	u	u
atlanta	yankees	**la**	georgia	lol
lakers	**sf**	chi	**chicago**	save

Multi-class problems

Unintuitive features:

LA	NY	C	At	SF
la	nyc	chicago	atlanta	sf
angeles	york	bears	atl	**httpdealnaycom**
los	ny	il	ga	francisco
chicago	chicago	**httpbitlyczmk**	**lol**	san
hollywood	atlanta	cubs	**u**	**u**
atlanta	yankees	la	georgia	**lol**
lakers	sf	chi	chicago	**save**

What's going on with MI?

Mutual Information is biased toward rare, uninformative features

- If a feature is seen rarely, but always with a given class, it will be seen as "good"
- All probabilities: no notion of the raw frequency of events
- Best features in the Twitter dataset only had MI of about 0.01 bits; 100th best for a given class had MI of about 0.002 bits

Glance into a few other common approaches to feature selection

A common (unsupervised) alternative

Term Frequency Inverse Document Frequency (TFIDF)

- Detect important words / Natural Language Processing
- Find words that are relevant to a document in a given document collection
- To be relevant, a word should be
  - Frequent enough in the corpus (TF). A word that occurs only 5 times in a corpus of 5,000,000 words is probably not too interesting
  - Special enough (IDF). A word that is very general and occurs in (almost) every document is probably not too interesting

A common (unsupervised) alternative

Term Frequency Inverse Document Frequency (TFIDF)

- Detect important words / Natural Language Processing
- Find words that are relevant to a document in a given document collection
- To be relevant, a word should be
  - Frequent enough in the corpus (TF). A word that occurs only 5 times in a corpus of 5,000,000 words is probably not too interesting
  - Special enough (IDF). A word that is very general and occurs in (almost) every document is probably not too interesting

$$tfidf(d, t, D) = tf + idf$$

$$tf = log(1 + freq(t, d))$$

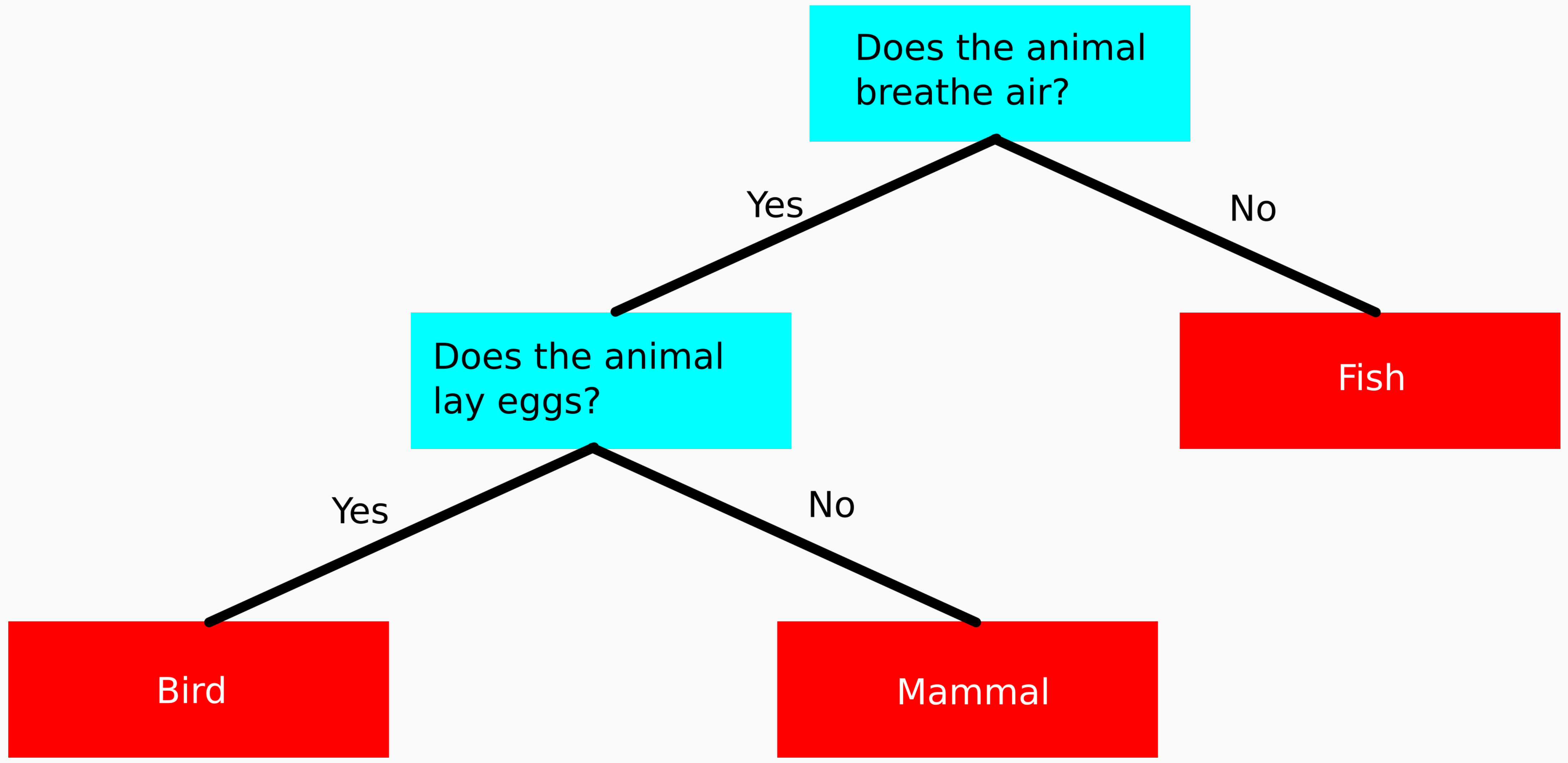
$$idf = log \frac{|D|}{count(d \in D : t \in d)}$$

d=document, t=term, D=document collection;
$|D|$=number of documents in D

Embedded Methods:

Some ML models include feature selection inherently

1. **Decision trees**: Generalization of 1-R



2. **Regression models with regularization**

$$\text{house\_price} = \beta_0 + \beta_1 \times \text{size} + \beta_2 \times \text{location} + \beta_3 \times \text{age}$$

Regularization (or ‘penalty’) nudges the weight β of unimportant features towards zero

Image:

https://towardsdatascience.com/a-beginners-guide-to-decision-tree-classification-6d3209353ea?gi=e0ee0b2b622e

And there are many more strategies

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_selection

sklearn.feature_selection: Feature Selection

The sklearn.feature_selection module implements feature selection algorithms. It currently includes univariate filter selection methods and the recursive feature elimination algorithm.

User guide: See the Feature selection section for further details.

feature_selection.GenericUnivariateSelect([...])	Univariate feature selector with configurable strategy.
feature_selection.SelectPercentile([...])	Select features according to a percentile of the highest scores.
feature_selection.SelectKBest([score_func, k])	Select features according to the k highest scores.
feature_selection.SelectFpr([score_func, alpha])	Filter: Select the pvalues below alpha based on a FPR test.
feature_selection.SelectFdr([score_func, alpha])	Filter: Select the p-values for an estimated false discovery rate
feature_selection.SelectFromModel(estimator, *)	Meta-transformer for selecting features based on importance weights.
feature_selection.SelectFwe([score_func, alpha])	Filter: Select the p-values corresponding to Family-wise error rate
feature_selection.SequentialFeatureSelector(...)	Transformer that performs Sequential Feature Selection.
feature_selection.RFE(estimator, *[, ...])	Feature ranking with recursive feature elimination.
feature_selection.RFECV(estimator, *[, ...])	Feature ranking with recursive feature elimination and cross-validated selection of the best number of features.
feature_selection.VarianceThreshold([threshold])	Feature selector that removes all low-variance features.

feature_selection.chi2(X, y)	Compute chi-squared stats between each non-negative feature and class.
feature_selection.f_classif(X, y)	Compute the ANOVA F-value for the provided sample.
feature_selection.f_regression(X, y, *[, center])	Univariate linear regression tests.
feature_selection.mutual_info_classif(X, y, *)	Estimate mutual information for a discrete target variable.
feature_selection.mutual_info_regression(X, y, *)	Estimate mutual information for a continuous target variable.

So ... is feature selection worth it?

Absolutely!

- Even marginally relevant features usually lead to a vast improvement on an unfiltered data set

- Some models **need** feature selection
  - k-Nearest Neighbors, hugely
  - Naive Bayes, to a lesser extent

- Machine learning experts (us!) need to think about the data!

Summary

Today

- Wrappers vs. Filters
- Popular filters: PMI, MI, χ^2, how should we use them and what are the results going to look like
- Importance of feature selection for different methods (even though it sometimes isn't the solution we were hoping for)

Next Lecture(s):

- Iterative optimization
- Logistic regression

References

Guyon, Isabelle, and Andre Elisseeff. 2003. An introduction to variable and feature selection. *The Journal of Machine Learning Research*. Vol 3, 1157–1182.

John, George, Ron Kohavi, and Karl Pfleger. 1994. Irrelevant features and the subset selection problem. In *Proceedings of the 11th International Conference on Machine Learning*, 121–9.

Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar. 2006. *Introduction to Data Mining*. Addison Wesley.

Witten, Ian, and Eibe Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. San Francisco, USA: Morgan Kaufmann.

Yang, Yiming, Jan Pedersen. 1997. A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, 412–20.