Lecture 18: Feature Selection

COMP90049
Knowledge Technologies

Sarah Erfani and Karin Verspoor, and Jeremy Nicholson, CIS

Semester 2, 2017

THE UNIVERSITY OF MELBOURNE

Where we're at so far

We want to get knowledge out of a data set:

Where we're at so far

Data set:

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	high	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
⋮	⋮	⋮	⋮	⋮

Where we're at so far

Instances:

	Outlook	Temperature	Humidity	Windy	Play
INSTANCE$_1$	sunny	hot	high	FALSE	no
INSTANCE$_2$	sunny	hot	high	TRUE	no
	overcast	hot	high	FALSE	yes
	rainy	mild	high	FALSE	yes
	rainy	cool	normal	FALSE	yes
	rainy	cool	normal	TRUE	no
	⋮	⋮	⋮	⋮	⋮

Where we're at so far

Attributes:

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	high	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
⋮	⋮	⋮	⋮	⋮

ATTRIBUTE$_1$

ATTRIBUTE$_2$

Where we're at so far

We want to get knowledge out of a data set:

- Where do instances come from?
  - Examples from real world data

Where we're at so far

We want to get knowledge out of a data set:

- Where do instances come from?
  - Examples from real world data
- Where do attributes come from?
  - ???

Where we're at so far

We want to get knowledge out of a data set:

- Where do instances come from?
  - Examples from real world data
- Where do attributes come from?
  - (Hopefully) meaningful features of the problem
  - Anything that might capture regularity in the data

Where we're at so far

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	FALSE	no
sunny	hot	high	TRUE	no
overcast	hot	high	FALSE	yes
rainy	mild	high	FALSE	yes
rainy	cool	normal	FALSE	yes
rainy	cool	normal	TRUE	no
⋮	⋮	⋮	⋮	⋮

- `Windy` seems like a good predictor of `Play`
- `Humidity` seems like a less good predictor of `Play`

Where we're at so far

We want to get knowledge out of a data set:

- Where do instances come from?
  - Examples from real world data
- Where do attributes come from?
  - (Hopefully) meaningful features of the problem
  - Anything that might capture regularity in the data
- Where do models come from?
  - ???

Where we're at so far

Models:

Where we're at so far

We want to get knowledge out of a data set:

- Where do instances come from?
  - Examples from real world data
- Where do attributes come from?
  - (Hopefully) meaningful features of the problem
  - Anything that might capture regularity in the data
- Where do models come from?
  - Need to choose a model suitable for our data set

Machine Learning, revisited

How to do Machine Learning:

1. Pick a feature representation
2.
3.
4.
5.
6.

Machine Learning, revisited

How to do Machine Learning:

1. Pick a feature representation
2. Compile data
3.
4.
5.
6.

Machine Learning, revisited

How to do Machine Learning:

1. Pick a feature representation
2. Compile data
3. Pick a (suitable) algorithm for building a model
4.
5.
6.

Machine Learning, revisited

How to do Machine Learning:

1. Pick a feature representation
2. Compile data
3. Pick a (suitable) algorithm for building a model
4. Train the model
5.
6.

Machine Learning, revisited

How to do Machine Learning:

1. Pick a feature representation
2. Compile data
3. Pick a (suitable) algorithm for building a model
4. Train the model
5. Classify development data, evaluate results
6.

Machine Learning, revisited

How to do Machine Learning:

1. Pick a feature representation
2. Compile data
3. Pick a (suitable) algorithm for building a model
4. Train the model
5. Classify development data, evaluate results
6. Probably: *go to (1)*

Machine Learning, revisited

How to do Machine Learning:

1. [0.] Get hired!
2. Pick a feature representation
3. Compile data
4. Pick a (suitable) algorithm for building a model
5. Train the model
6. Classify development data, evaluate results
7. Probably: *go to (1)*

Machine Learning, revisited

Our job as Machine Learning experts:

- Choose an algorithm suitable for classifying the data according to the attributes
- Choose attributes suitable for classifying the data according to the model

Machine Learning, revisited

Our job as Machine Learning experts:

- Choose an algorithm suitable for classifying the data according to the attributes
- Choose attributes suitable for classifying the data according to the model
  - Inspection
  - Intuition

Machine Learning, revisited

Our job as Machine Learning experts:

- Choose an algorithm 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

Machine Learning, revisited

Our job as Machine Learning experts:

- Choose an algorithm 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
  - Throw everything we can think of at the problem and let the algorithm decide!

What makes features good?

What makes features good?

Better models!

- Better performance according to some evaluation metric

What makes features good?

Better models!

- Better performance according to some evaluation metric

Side-goal:

- Tell us interesting things about the problem

What makes features good?

Better models!

- Better performance according to some evaluation metric

Side-goal:

- Tell us interesting things about the problem

Side-goal:

- Fewer features $\rightarrow$ smaller models $\rightarrow$ faster answer

What makes features good?

Better models!

- Better performance according to some evaluation metric

Side-goal:

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

Side-goal:

- Fewer features $\rightarrow$ smaller models $\rightarrow$ 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

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 $\{\texttt{Outlook}\}$

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 $\{\texttt{Outlook}\}$
  - Train model on $\{\texttt{Temperature}\}$
  - ...

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`}

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 $\rightarrow$ 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

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

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

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

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 approach:

- Train and evaluate model on each single attribute
- Choose best attribute

More practical wrapper methods

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 approach:

- Bad news:
  - Takes $\frac{1}{2}m^2$ cycles, for m attributes
  - In theory, 386 attributes $\rightarrow$ days

More practical wrapper methods

Greedy approach:

- ~~Bad~~ Good news:
  - Takes $\frac{1}{2}m^2$ cycles, for m attributes
  - In practice, converges much more quickly than this

More practical wrapper methods

Greedy approach:

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

More practical wrapper methods

"Ablation" approach:

- Start with all attributes
- Remove one attribute, train and evaluate model

More practical wrapper methods

"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 ϵ

More practical wrapper methods

“Ablation” approach:

- Good news:
  - Mostly removes irrelevant attributes (at the start)

More practical wrapper methods

"Ablation" approach:

- Good news:
  - Mostly removes irrelevant attributes (at the start)
- Bad news:
  - Assumes independence of attributes (both approaches; worse than Naive Bayes!)
  - Actually does take $O(m^2)$ time; cycles are slower with more attributes

In-built feature selection

"Embedded" methods:

- Some models actually perform feature selection as part of the algorithm!

In-built feature selection

"Embedded" methods:

- Some models actually perform feature selection as part of the algorithm!
  - Most notably, linear classifiers
  - To some degree: Decision Trees
- (More on these models in later lectures.)

In-built feature selection

"Embedded" methods:

- Some models actually perform feature selection as part of the algorithm!
  - Most notably, linear classifiers
  - To some degree: Decision Trees
- Often benefit from other feature selection approaches anyway

Feature filtering

Intuition: possible to evaluate “goodness” of each feature, separate from other features

Feature filtering

Intuition: possible to evaluate “goodness” of each feature, separate from other features

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

Feature “goodness”

What makes a feature set good?

Feature "goodness"

What makes a feature set good?

- Better models!

Feature “goodness”

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

- ~~Better models!~~

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.

Recall independence:

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

Pointwise Mutual Information

Recall independence:

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

This formula holds if attribute is independent from class.

Pointwise Mutual Information

Recall independence, equivalently:

$$P(C|A) = P(C)$$

This formula holds if attribute is independent from class.

Pointwise Mutual Information

Recall independence, equivalently:

$$P(C|A) = P(C)$$

We clearly want attributes that are **not** independent from class.

Pointwise Mutual Information

Recall independence:

$$\frac{P(A, C)}{P(A)P(C)} = 1$$

Pointwise Mutual Information

Recall independence:

$$\frac{P(A, C)}{P(A)P(C)} = 1$$

- If LHS ~ 1, attribute and class occur together as often as we would expect from random chance
- If LHS $>> 1$, attribute and class occur together much more often than randomly.
- (If LHS $<< 1$, attribute and class are negatively correlated. More on this later.)

Pointwise Mutual Information

Pointwise mutual information:

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

Pointwise Mutual Information

Pointwise mutual information:

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

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

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

Toy example, revisited

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

$$\begin{aligned} PMI(a_1, c) &= \log_2 \frac{\frac{1}{2}}{\frac{1}{2} \cdot \frac{1}{2}} \\ &= \log_2(2) = 1 \end{aligned}$$

Toy example, revisited

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

$$P(a_2) = \frac{2}{4};\ P(c) = \frac{2}{4};\ P(a_1, c) = \frac{1}{4}$$

Toy example, revisited

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

$$
\begin{aligned}
PMI(a_2, c) &= \log_2 \frac{\frac{1}{4}}{\frac{1}{2} \cdot \frac{1}{2}} \\
&= \log_2(1) = 0
\end{aligned}
$$

Feature "goodness", revisited

What makes a single feature good?

- Well correlated with class

Feature “goodness”, revisited

What makes a single feature good?

- Well correlated with class
  - Knowing *a* lets us predict *c* with more confidence

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
  - Just as good

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

Mutual information: combine each a, $\bar{a}$, c, $\bar{c}$ PMI

Aside: Contingency tables

Contingency tables: compact representation of these frequency counts

Aside: Contingency tables

Contingency tables: compact representation of these frequency counts

	a	$\bar{a}$
c	$\sigma(a, c)$	$\sigma(\bar{a}, c)$
$\bar{c}$	$\sigma(a, \bar{c})$	$\sigma(\bar{a}, \bar{c})$

Aside: Contingency tables

Contingency tables: compact representation of these frequency counts

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

Aside: Contingency tables

Contingency tables: compact representation of these frequency counts

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

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

Aside: Contingency tables

Contingency tables for toy example:

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

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

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

Mutual Information

$$\begin{aligned} MI(A, C) &= P(a, c)PMI(a, c) + P(\bar{a}, c)PMI(\bar{a}, c) + \\ &\quad P(a, \bar{c})PMI(a, \bar{c}) + P(\bar{a}, \bar{c})PMI(\bar{a}, \bar{c}) \end{aligned}$$

Mutual Information

$$\begin{aligned} MI(A, C) = \; & P(a,c)\log_2 \frac{P(a,c)}{P(a)P(c)} + P(\bar{a},c)\log_2 \frac{P(\bar{a},c)}{P(\bar{a})P(c)} + \\ & P(a,\bar{c})\log_2 \frac{P(a,\bar{c})}{P(a)P(\bar{c})} + P(\bar{a},\bar{c})\log_2 \frac{P(\bar{a},\bar{c})}{P(\bar{a})P(\bar{c})} \end{aligned}$$

Mutual Information

Often written more compactly as:

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

(This representation can be extended to different types of attributes more intuitively.)

Note that $0 \log 0 \equiv 0$.

Mutual Information Example

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

Contingency table for toy example:

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

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

Mutual Information Example

$$
\begin{aligned}
MI(A_1, C) &= P(a_1, c)\log_2 \frac{P(a_1, c)}{P(a_1)P(c)} + P(\bar{a}_1, c)\log_2 \frac{P(\bar{a}_1, c)}{P(\bar{a}_1)P(c)} + \\
&\quad P(a_1, \bar{c})\log_2 \frac{P(a_1, \bar{c})}{P(a_1)P(\bar{c})} + P(\bar{a}_1, \bar{c})\log_2 \frac{P(\bar{a}_1, \bar{c})}{P(\bar{a}_1)P(\bar{c})} \\
&= \frac{1}{2}\log_2 \frac{\frac{1}{2}}{\frac{1}{2}\frac{1}{2}} + 0\log_2 \frac{0}{\frac{1}{2}\frac{1}{2}} + 0\log_2 \frac{0}{\frac{1}{2}\frac{1}{2}} + \frac{1}{2}\log_2 \frac{\frac{1}{2}}{\frac{1}{2}\frac{1}{2}} \\
&= \frac{1}{2}(1) + 0 + 0 + \frac{1}{2}(1) = 1
\end{aligned}
$$

Mutual Information Example

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

Contingency table for toy example:

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

$$P(a,c) = \frac{1}{4};\ P(a) = \frac{2}{4};\ P(c) = \frac{2}{4}$$
$$P(\bar{a},\bar{c}) = \frac{1}{4};\ P(\bar{a}) = \frac{2}{4};\ P(\bar{c}) = \frac{2}{4}$$
$$P(\bar{a},c) = \frac{1}{4};\ P(a,\bar{c}) = \frac{1}{4}$$

Mutual Information Example

$$
\begin{aligned}
MI(A_2, C) &= P(a_2, c)\log_2 \frac{P(a_2, c)}{P(a_2)P(c)} + P(\bar{a}_2, c)\log_2 \frac{P(\bar{a}_2, c)}{P(\bar{a}_2)P(c)} + \\
&\quad P(a_2, \bar{c})\log_2 \frac{P(a_2, \bar{c})}{P(a_2)P(\bar{c})} + P(\bar{a}_2, \bar{c})\log_2 \frac{P(\bar{a}_2, \bar{c})}{P(\bar{a}_2)P(\bar{c})} \\
&= \frac{1}{4}\log_2 \frac{\frac{1}{4}}{\frac{1}{2}\frac{1}{2}} + \frac{1}{4}\log_2 \frac{\frac{1}{4}}{\frac{1}{2}\frac{1}{2}} + \frac{1}{4}\log_2 \frac{\frac{1}{4}}{\frac{1}{2}\frac{1}{2}} + \frac{1}{4}\log_2 \frac{\frac{1}{4}}{\frac{1}{2}\frac{1}{2}} \\
&= \frac{1}{4}(0) + \frac{1}{4}(0) + \frac{1}{4}(0) + \frac{1}{4}(0) = 0
\end{aligned}
$$

Chi-square

Check 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 lesser 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

Chi-square

Actual calculation (written more compactly):

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

(i sums over rows and j sums over columns.)
In practice, there are simpler ways to calculate this for 2×2 contingency tables.

Types of Attribute

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

- Nominal attributes
- Continuous attributes
- Ordinal attributes

Types of Attribute

Nominal attributes (e.g. `Outlook`=$\{$sunny, overcast, rainy$\}$).
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 is useful, but `Outlook`=overcast and `Outlook`=rainy are not useful... Should we use `Outlook`?
2. Modify contigency tables (and formulae)

Types of Attribute

Modified contingency table:

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

Modified MI:

$$
\begin{aligned}
MI(O,C) &= \sum_{i\in\{s,o,r\}} \sum_{j\in\{c,\bar{c}\}} P(i,j)\log_2 \frac{P(i,j)}{P(i)P(j)} \\
&= P(s,c)\log_2 \frac{P(s,c)}{P(s)P(c)} + P(s,\bar{c})\log_2 \frac{P(s,\bar{c}}{P(s)P(\bar{c})} + \\
& P(o,c)\log_2 \frac{P(o,c)}{P(o)P(c)} + P(o,\bar{c})\log_2 \frac{P(o,\bar{c})}{P(o)P(\bar{c})} + \\
& P(r,c)\log_2 \frac{P(r,c)}{P(r)P(c)} + P(r,\bar{c})\log_2 \frac{P(r,\bar{c})}{P(r)P(\bar{c})}
\end{aligned}
$$

- Biased towards attributes with many values. (Why?)

Types of Attribute

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 typically need to use messy binomial/multinomial distributions

All of this is (unsurprisingly) beyond the scope of this subject

Types of Attribute

Ordinal attributes (e.g. low, med, high or 1,2,3,4).
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 nominal (i.e. throw away ordering)
3. 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

Multi-class problems

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

What makes a single feature good?

- Highly correlated with class
- Highly reverse correlated with class
- Highly correlated (or reverse correlated) with not class

. . . What if there are many classes?

Multiclass (e.g., Boston, Houston, Seattle, San Diego, Washington) classification tasks are usually much more difficult.

Multi-class problems

What makes a feature **bad**?

- Irrelevant

Multi-class problems

What makes a feature **bad**?

- Irrelevant
- Correlated with other features

Multi-class problems

What makes a feature **bad**?

- Irrelevant
- Correlated with other features
- Good at only predicting one class (but is this truly bad?)

Multi-class problems

Consider multi-class problem over B, H, Se, SD, W:

- 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):

B	H	Se	SD	W
boston	houston	seattle	diego	dc
diego	diego	diego	san	diego
san	jupdicom	wa	chargers	san
httpbitlyczmk	tx	san	sd	obama
ma	san	cheezburger	sdut	health
redsox	httpbitlycdqk	boston	seattle	washington
seattle	seattle	bellevue	sandiego	bill

Multi-class problems

Intuitive features:

B	H	Se	SD	W
boston	**houston**	**seattle**	**diego**	**dc**
diego	diego	diego	**san**	diego
san	jupdicom	**wa**	**chargers**	san
httpbitlyczmk	**tx**	san	**sd**	**obama**
ma	san	cheezburger	**sdut**	health
redsox	httpbitlycdqk	boston	seattle	**washington**
seattle	seattle	**bellevue**	**sandiego**	**bill**

Features for predicting not class (MI):

B	H	Se	SD	W
boston	houston	seattle	diego	dc
diego	**diego**	**diego**	san	**diego**
san	jupdicom	wa	chargers	**san**
httpbitlyczmk	tx	**san**	sd	obama
ma	**san**	cheezburger	sdut	health
redsox	httpbitlycdqk	**boston**	**seattle**	washington
seattle	**seattle**	bellevue	sandiego	bill

Multi-class problems

Unintuitive features:

B	H	Se	SD	W
boston	houston	seattle	diego	dc
diego	diego	diego	san	diego
san	**jupdicom**	wa	chargers	san
httpbitlyczmk	tx	san	sd	obama
ma	san	**cheezburger**	sdut	**health**
redsox	**httpbitlycdqk**	boston	seattle	washington
seattle	seattle	bellevue	sandiego	bill

What's going on with MI?

Mutual Information is biased toward rare, uninformative features

- All probabilities: no notion of the raw frequency of events
- If a feature is seen rarely, but always with a given class, it will be seen as "good"
- For example: `httpbitlyczmk` occurs 447 times out of 750K instances, but often with `B`. Is this meaningful?
- Best features in the Twitter dataset only had MI of about 0.1 bits; 100$^{\text{th}}$ best for a given class had MI of about 0.0001 bits

So... Give up on feature selection then?

No way!

- Even marginally relevant features usually a vast improvement on an unfiltered data set
- Some models **need** feature selection
  - k-Nearest Neighbour, hugely
  - Naive Bayes, Decision Trees, andSVM to a lesser extent
- Machine learning experts (us!) need to think about the data!

Summary

- Wrappers vs. Embedded methods 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 often isn't the solution we were hoping for)

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