COMP30027 Machine Learning
Feature Selection

Semester 1, 2019

Jeremy Nicholson & Tim Baldwin & Karin Verspoor

Lecture Outline

1. Features in Machine Learning
2. Feature Selection
   - Wrappers
   - Embedded
   - Filters
3. Filtering methods
   - PMI
   - MI
   - χ^2
4. Common Issues
5. Practical considerations
6. Summary

Where we're at so far I

We want to get knowledge out of a 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 II

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 I

We want to get knowledge out of a data set:

- Data mining
- Machine learning
  - Supervised machine learning $\leftarrow$ today (mostly)
  - Unsupervised machine learning

Machine Learning, revisited II

How to do (supervised) Machine Learning:

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

Machine Learning, revisited III

Our job as Machine Learning experts:

- 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
  - Throw everything we can think of at the problem and let the model decide!

Lecture Outline

1. Features in Machine Learning
2. Feature Selection
   - Wrappers
   - Embedded
   - Filters
3. Filtering methods
   - PMI
   - MI
   - χ^2
4. Common Issues
5. Practical considerations
6. Summary

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 I

"Wrapper" methods:

- Choose subset of attributes that give best performance on the development data (with respect to a single learner)
- 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 II

"Wrapper" methods:

- Choose subset of attributes that give best performance on the development data (with respect to a single learner)
- 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 III

"Wrapper" methods:

- Choose subset of attributes that give best performance on the development data (with respect to a single learner)
- Advantages:
  - Feature set with optimal performance on development data (for this learner)
- 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$50K instances):

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

How many cycles? For m attributes:

- 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 I

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 II

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
    - (Assumes independence of attributes)

More practical wrapper methods III

"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 IV

"Ablation" approach:

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

In-built feature selection

"Embedded" methods:

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

Feature filtering

Intuition: possible to evaluate "goodness" of each attribute, separate from other attributes

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

Lecture Outline

1. Features in Machine Learning
2. Feature Selection
   - Wrappers
   - Embedded
   - Filters
3. Filtering methods
   - PMI
   - MI
   - χ^2
4. Common Issues
5. Practical considerations
6. Summary

Feature "goodness"

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

- ~~Better models!~~
- Well correlated with [interesting] class

Toy example I

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 II

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

a_1 is probably good.

Toy example III

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

a_2 is probably not good.

Pointwise Mutual Information I

Recall independence:

$$
\begin{aligned}
P(A, C) &= P(A)P(C) \\
P(C|A) &= P(C)
\end{aligned}
$$

This formula holds if attribute is independent from class.
We clearly want attributes that are **not** independent from class.

Pointwise Mutual Information II

Recall independence:

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

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

Pointwise Mutual Information III

Pointwise mutual information:

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

Attributes with greatest PMI: best attributes (most correlated with class)
(Various notational shorthand conventions, like $A \rightarrow A = Y$, where Y is the "interesting" value of a binary attribute)

Toy example, revisited I

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 II

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 III

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 IV

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 [interesting] 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 **uninteresting** class
  - Knowing a lets us predict $\bar{c}$ with more confidence
  - Usually not quite as good, but still useful

Aside: Contingency tables I

Contigency 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 II

Contingency tables for toy example:

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

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

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

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 I

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 II

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

Mutual Information Example III

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 IV

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

Chi-square I

Similar idea, different solution:
Consider contingency table (shorthand):

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

If a, c were independent (uncorrelated), what value would I expect to be in W ($E(W)$)?

Chi-square II

a, c independent $\rightarrow P(a,c) = P(a)P(c)$

$$\begin{aligned}
P(a,c) &= P(a)P(c) \\
\frac{\sigma(a,c)}{N} &= \frac{\sigma(a)}{N}\frac{\sigma(c)}{N} \\
\sigma(a,c) &= \frac{\sigma(a)\sigma(c)}{N} \\
E(W) &= \frac{(W+Y)(W+X)}{W+X+Y+Z}
\end{aligned}$$

Chi-square III

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 IV

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

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

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

Because the values are squared, χ^2 becomes much greater when $| O - E |$ is large, but E is small.
In practice, there are simpler ways to calculate this for 2×2 contingency tables.

Chi-square Example I

Contingency table for toy example (observed values):

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

Chi-square Example II

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

Chi-square Example III

Contingency table for toy example (observed values):

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

Contingency table for toy example (expected values):

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

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

Lecture Outline

1. Features in Machine Learning
2. Feature Selection
   - Wrappers
   - Embedded
   - Filters
3. Filtering methods
   - PMI
   - MI
   - χ^2
4. Common Issues
5. Practical considerations
6. Summary

Types of Attribute I

Nominal attributes (e.g. `Outlook`={sunny, overcast, rainy}).
Two common strategies:

1. Treat as multiple binary attributes (one-hot):
   - 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`?

Types of Attribute II

Nominal attributes (e.g. `Outlook`={sunny, overcast, rainy}).
Two common strategies:

2. Modify contigency tables (and formulae)

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

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

Types of Attribute III

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 Attribute IV

Continuous attributes:

- Probabilities can be estimated by fitting a Gaussian
- The values can be discretised

Types of Attribute V

Ordinal attributes (e.g. low, med, high or 1,2,3,4).
Three possibilities, roughly in order of popularity:

- Treat as binary
    - Particularly appropriate for frequency counts where events are low-frequency (e.g. words in short documents)
- 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
- Treat as nominal (i.e. throw away ordering)

Multi-class problems I

So far, we've only looked at binary (Y/N) classification tasks.
Multiclass (e.g. Melbourne, Sydney, Brisbane, Perth, Adelaide) classification tasks are usually much more difficult.
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?

Multi-class problems II

What makes a feature **bad**?

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

Multi-class problems III

Consider multi-class problem over Melbourne, Sydney, Brisbane, Perth, Adelaide:

- We choose some features: `swanston`, `fed`, `mcg`, `docklands`, `afl`, `birrarung`, ...
- What happens?

Multi-class problems IV

Consider multi-class problem over Melbourne, Sydney, Brisbane, Perth, Adelaide:

- 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 V

[Example will have to wait for Project 2's release!]

Lecture Outline

1. Features in Machine Learning
2. Feature Selection
   - Wrappers
   - Embedded
   - Filters
3. Filtering methods
   - PMI
   - MI
   - χ^2
4. Common Issues
5. Practical considerations
6. Summary

What's going on with MI?

Mutual Information is biased toward common, uninformative features

- All probabilities: no notion of the raw frequency of events
- For example: 10% of the instances, a common attribute occurs with a particular class, but 11% of instances are truly of that class. Is this meaningful?
- Best features in a typical dataset might only have MI of about 0.03 bits; 100^{th} best for a given class, perhaps 0.001 bits
- Many very common features will be selected

But Chi-square is better? I

Chi-square is biased toward rare, "informative" features

- This happens because of squaring the difference (rare means small E)
- If a feature is seen rarely, but always with a given class, it will be seen as "good"
- For example: a particular attribute is present in 100 out of 200K instances, but always with (a relatively rare) class. Is this meaningful?
- Various "humps" where infrequent (1 instance, 2 instances, 3 instances, etc.) attributes are tied in the ranking

But Chi-square is better? II

- Typically, both methods have a very high overlap
- Values for chi-square are often huge (>10000), much larger than the critical value for the distribution (five classes: 9.49)
    - Even stringently applying the significance test won't help
- Looks pretty unhelpful at a casual glance

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, especially for feature **weighting**
  - Naive Bayes/Decision Trees, to a lesser extent
  - SVMs can be better off without the feature selection, even when it's working well!
- Machine learning experts (us!) need to think about the data!

Lecture Outline

1. Features in Machine Learning
2. Feature Selection
   - Wrappers
   - Embedded
   - Filters
3. Filtering methods
   - PMI
   - MI
   - χ^2
4. Common Issues
5. Practical considerations
6. Summary

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