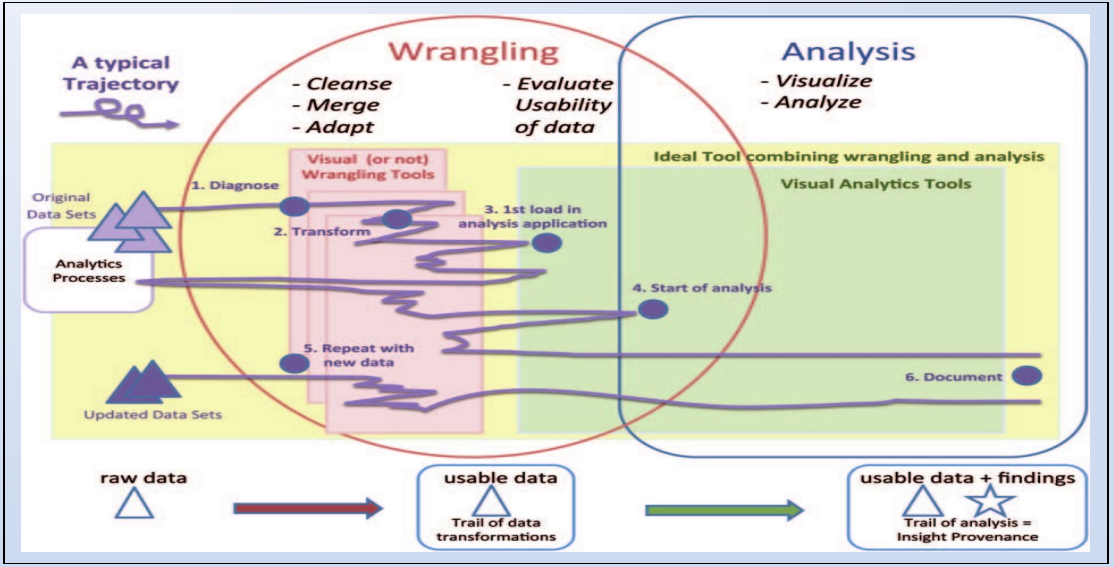
COMP20008 Elements of Data Processing

Experimental Design

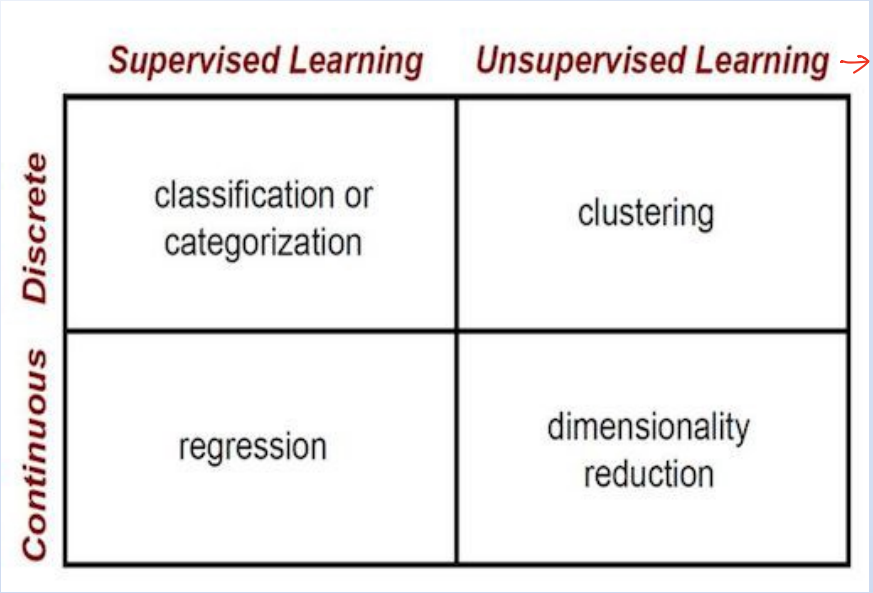
Semester 1, 2020

Contact: uwe.aickelin@unimelb.edu.au

Where are we now?



Supervised vs Unsupervised Learning



→ we don't tend to have the right answers

Experimental Design

- Feature Selection
- Dimensionality Reduction
- Performance Evaluation
- Real-world Example – Medical Datamining

Feature Selection

Feature selection for machine learning

- A series of data wrangling steps result in a set of features
- **Features** are inputs to a model
  - Choose attributes suitable for classifying the data according to the model
- Train the model
- Classify the data
- Evaluate the results

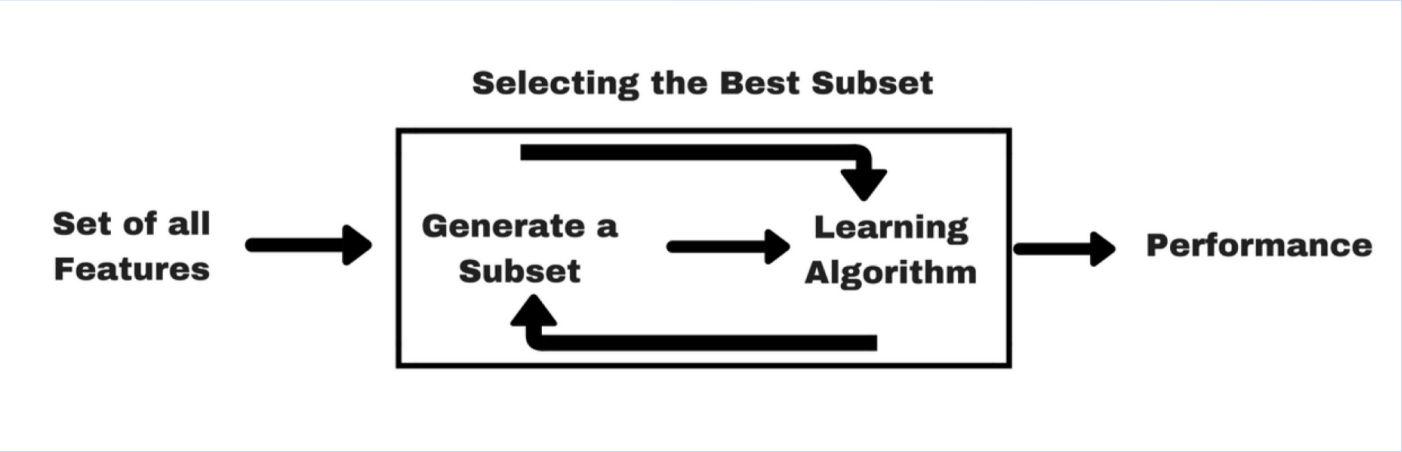
Good feature sets for machine learning

- Throw everything we can think of at the model and let the algorithm decide?
- Good features give better model performance
- Choose features suitable for classifying the data according to the model
  - Inspection
  - Intuition
  - Neither may be possible in practice
- Feature Selection Methods

Feature selection 1 - wrapper

“Wrapper” method:

Choose subset of attributes that give best performance on the data



https://www.analyticsvidhya.com/

Feature selection 1 - wrapper

Example: classification of Iris data, train and evaluate a model on each subset of attributes:

- {petal length} ; {petal width} ; {sepal width} ; {sepal length}
- {petal length, petal width}; {petal length, sepal length}; …
- {petal length, petal width, sepal length}; {sepal width}; …
- {petal length, petal width, sepal length, sepal width}

- Choose the subset giving the best performance
- Disadvantages: time consuming
  - Only practical for very small data sets

Practical wrapper approach - Greedy

"Greedy" approach:

- Train and evaluate model on each single feature
- Choose the best feature
- Repeat until performance stops increasing
  - Train and evaluate model on best feature(s) plus each remaining single feature
  - Add the best feature to the best feature set

add one feature at a time based on performance

Practical wrapper approach – Greedy

"Greedy" approach:

- Assumes independence of features since evaluate feature individually
- Quicker than the naïve wrapper approach
- May converge to a sub-optimal (and can be very bad) solution

Practical wrapper approach – Decremental

"Decremental" approach:

- Start with all features
- Remove one feature, train and evaluate model
- Repeat until performance degrades too much:
  - From remaining features, remove one at a time, train and evaluate model
  - Remove the feature that causes the least performance degradation

Practical wrapper approach – Decremental

"Decremental" approach:

- Assumes independence of features
- Mostly removes irrelevant features won't boost your performance
- Slower than the greedy approach with more features

Feature selection 2 - filtering

Intuition: possible to evaluate "goodness" of **each** feature, separate from other features

e.g. choose top-10 features based on how good they are

- Consider each feature separately: linear time in number of attributes
- Typically most popular strategy

good: only need to evaluate each feature once

Possible (**but difficult**) to control for inter-dependence of features

Feature filtering

What makes a single feature good?

- **Well correlated with class (ratio, interval, ordinal)** → correlation
- **Not independent of class (ordinal, nominal)** → association
- A simple example: Which of a_1, a_2 is a good feature?

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

perfectly associate

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

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

Feature filtering – Mutual information

What makes a single feature good?

- **Well correlated with class**

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

Mutual Information:

- $MI(X,Y) = H(Y) - H(Y|X)$

- $MI(c,a_1) = H(c) - H(c|a_1) = 1 - 0 = 1$
  - $H(c) = -\sum_{i\,\epsilon\{Y,N\}} p_i \log_2 p_i = -\left(\frac{1}{2}\times \log_2 \frac{1}{2} + \frac{1}{2}\times \log_2 \frac{1}{2}\right) = 1$
  - $H(c|a_1) = \sum_{x\,\epsilon\{Y,N\}} p(a_1 = x) H(c|a_1 = x) = \frac{1}{2}\times 0 + \frac{1}{2}\times 0 = 0$

high information gain → reduce entropy

- High information gain: a_1 can strongly predict c; keep a_1 in the feature set

Feature filtering – Mutual information

What makes a single feature good?

- **Well correlated with class**

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

Mutual Information:

- $MI(X,Y) = H(Y) - H(Y|X)$

- $MI(c,a_2) = H(c) - H(c|a_2) = 1 - 1 = 0$
  - $H(c) = -\sum_{i\,\epsilon\{Y,N\}} p_i \log_2 p_i = -\left(\frac{1}{2}\times \log_2 \frac{1}{2} + \frac{1}{2}\times \log_2 \frac{1}{2}\right) = 1$
  - $H(c|a_2) = \sum_{x\,\epsilon\{Y,N\}} p(a_2 = x) H(c|a_2 = x) = \frac{1}{2}\times 1 + \frac{1}{2}\times 1 = 1$
- No information gain: a_2 cannot predict c; remove a_2 from the feature set

Feature filtering – Chi-square χ^2

What makes a single feature good?

- **Not independent of class**
- Remove a feature if it is independent of the output class

Chi-square test is a statistical hypothesis test for quantifying the independence of pairs of **categorical (ordinal, nominal)** variables.

- H0: two variables are **independent**
- **Takes into account sample size & has a significance test (MI does not)**

Feature filtering – Chi-square χ^2

- Summarise frequencies of feature and the class in a **contingency table**
- **Is there a difference between observed and expected frequencies?**

Observed frequencies of a_1 and c, e.g. $O_{(c=Y,a_1=Y)} = 2$

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

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

Chi-square χ^2 – Null Hypothesis

H0: two variables are **independent**: $P(x \cap y) = P(x) \times P(y)$

- Find **expected** frequencies for all (a_1, c) pairs

P:$(c = Y \text{ and } a_1 = Y) = \frac{2}{4} \times \frac{2}{4} = \frac{1}{4}$; Expected value $E_{(c=Y, a_1=Y)} = 4 \times \frac{1}{4} = 1$

P:$(c = Y \text{ and } a_1 = N) = \frac{1}{4}$; Expected value $= 1$

P:$(c = N \text{ and } a_1 = Y) = \frac{1}{4}$; Expected value $= 1$

P:$(c = N \text{ and } a_1 = N) = \frac{1}{4}$; Expected value $= 1$

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

Chi-square χ^2 – calculate for a_1

- Calculate Difference between observed and expected values
- $\chi^2(a_1, c) = \sum_{i\epsilon\{c=Y, c=N\}} \sum_{j\epsilon\{a_1=Y, a_1=N\}} \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}}$ → calculate difference between observed and expected → normalise by expected value
- $\boldsymbol{\chi^2(a_1, c)} = \left(\frac{(2-1)^2}{1} + \frac{(0-1)^2}{1} + \frac{(0-1)^2}{1} + \frac{(2-1)^2}{1}\right) = \mathbf{4}$

Contingency table (**observed**)

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

Contingency table (**expected**)

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

Chi-square χ^2 – test for a_1

THE UNIVERSITY OF MELBOURNE

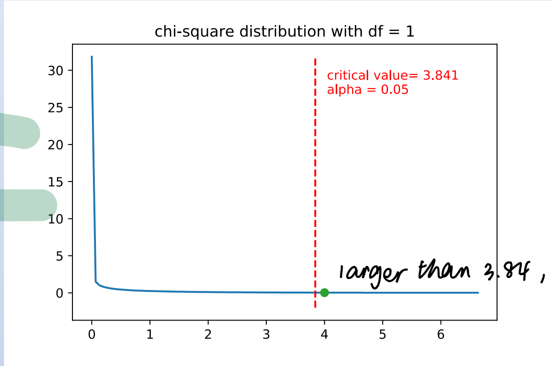
The Chi-square statistic for (a_1, c) is: $\chi^2(a_1, c) = 4$

Are a_1 and c independent? (the null hypothesis H0)

- With 95% confidence (alpha = 0.05)
- Degree of freedom = $(2-1) \times (2-1) = 1$ (a_1 has 2 values, c has 2 values)
- The Chi-Square critical value is 3.84 **(lookup this value)**

degree of freedom n个样本，如果在某种条件下，样本均值是先定的(fixed). 那么(只剩) n-1 个样本的值可以变化

- Reject H0
- a_1 is NOT independent of c
- a_1 is part of the feature set



larger than 3.84, reject H0

df	P = 0.05	P = 0.01	P = 0.001
1	3.84	6.64	10.83
2	5.99	9.21	13.82
3	7.82	11.35	16.27
4	9.49	13.28	18.47
5	11.07	15.09	20.52
6	12.59	16.81	22.46

Chi-square test in Python

```
from scipy.stats import chi2
```

Will practise in workshop

Feature Selection 3 - embedded

"Embedded" methods:

- Some models perform feature selection as part of the algorithm
  - Linear classifiers, e.g. regression
  - To some degree: Decision Trees → not predictive feature will get eliminated
- Often still benefit from other feature selection approaches

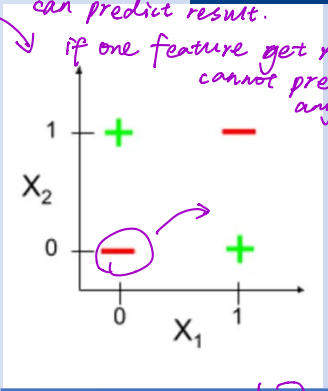
Feature Selection – potential issues

Difficult to control for inter-dependence of features

- Feature filtering of single features may remove important features. For example, where the class is the XOR of some features:
  - Given all the features, the class is totally predictable.
  - Given one of them, the MI or chi-square statistic is 0.
- In practice, feature engineering also used, i.e. construct new features out of existing ones, e.g. ratio / difference between features

if I have both features, can predict result.

if one feature get removed, cannot predict anything.



Income	Expenditure	i>e	Credit
120	100	1	Good
50	30	1	Good
50	70	0	Bad
200	40	1	Good
200	210	0	Bad
…	…	…	…
160	150	1	Good

new feature
income > expenditure

Dimensionality Reduction

Dimensionality reduction

Purpose:

- Avoid curse of dimensionality
- Reduce amount of time and memory required by algorithms
- Allow data to be more easily visualised
- May help to eliminate irrelevant features or reduce noise
  - Fewer features → smaller machine learning models → faster answer
  - Better feature set by transformation → more accurate answer

Motivation: High dimensional data

- **The curse of dimensionality**: “Data analysis techniques which work well at lower dimensions (fewer features), often perform poorly as the dimensionality of the analysed data increases (lots of features)”

- As dimensionality increases, data becomes increasingly sparse and all the distances between pairs of points begin to look the same. Impacts any algorithm that is based on distances between objects.

Dimensionality reduction

Input: a dataset with N features and K objects

Output: a transformed dataset with $\mathrm{n} \ll N$ features and K objects

- if n 2 or 3 transformed dataset can be easily visualised

Example: $n = 2$

Object id	Feature1	Feature2	….	FeatureN
1	…	…	…	…
…	….	…	…	…
K	….	….	…	…

Transformation

Object id	NewFeatureA	NewFeatureB
1	…	…
…	…	…
K	…	…

Transforming from N to $n \ll N$ dimensions

have minimum information loss

- The transformation should preserve the characteristics of the data, e.g. distances between pairs of objects
- If a pair of objects is close before the transformation, they should still be close after the transformation
- If a pair of objects is far apart before the transformation, they should still be far apart after the transformation
- The set of nearest neighbors of an object before the transformation should ideally be the same after the transformation

near before → also near after

Question

- Suppose we are given a dataset with the following N features, describing individuals in this class. Which two features would you select to represent people, in such a way that "distances" between pairs of people are likely to be preserved in the reduced dataset?

- Input: N=7 features
  1. Weighted average mark (WAM) → choose one → more fine feature
  2. Age (years) → similar age in the class → not very discriminated feature
  3. Height (cm)
  4. Weight (kg)

  correlated

  5. Number of pets owned → not very discriminated feature
  6. Number of subjects passed so far
  7. Amount of sleep last night (0=little, 1=medium, 2=a lot)
- Output: Select 2 of the above features

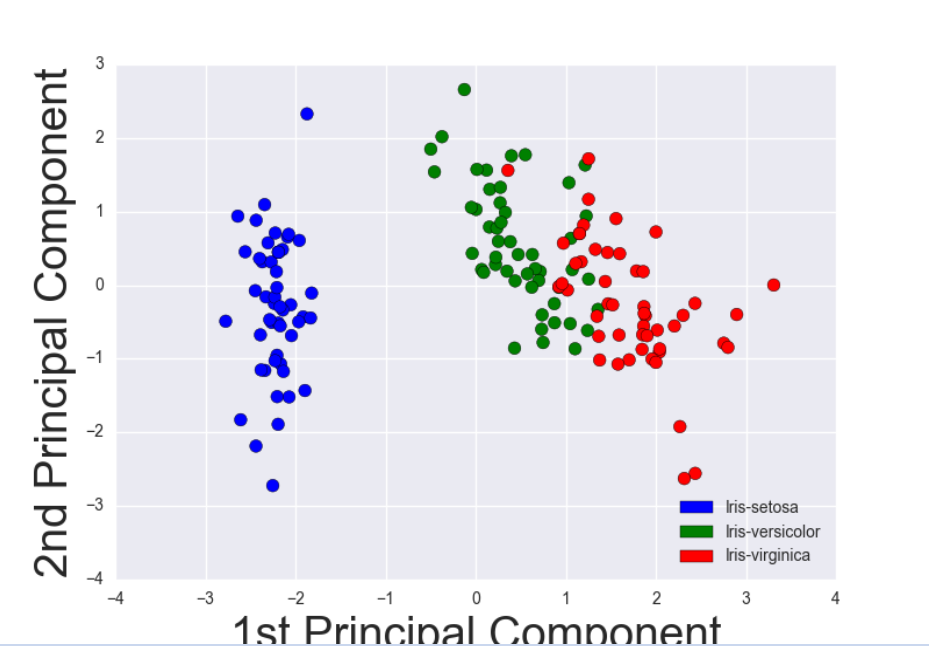
Dimensionality reduction

- Basic method: To reduce dimensionality perform feature selection.
  - Scatter plots for Iris dataset shown earlier – 2D visualisations of a 4D dataset. 2 features were selected from the original 4.
- In general, when transforming a dataset from N to n ($n \ll N$) features
  - *The output n features do not need to be a subset of the input N features.* Rather, they can be new features whose values are constructed using some function applied to the input N features

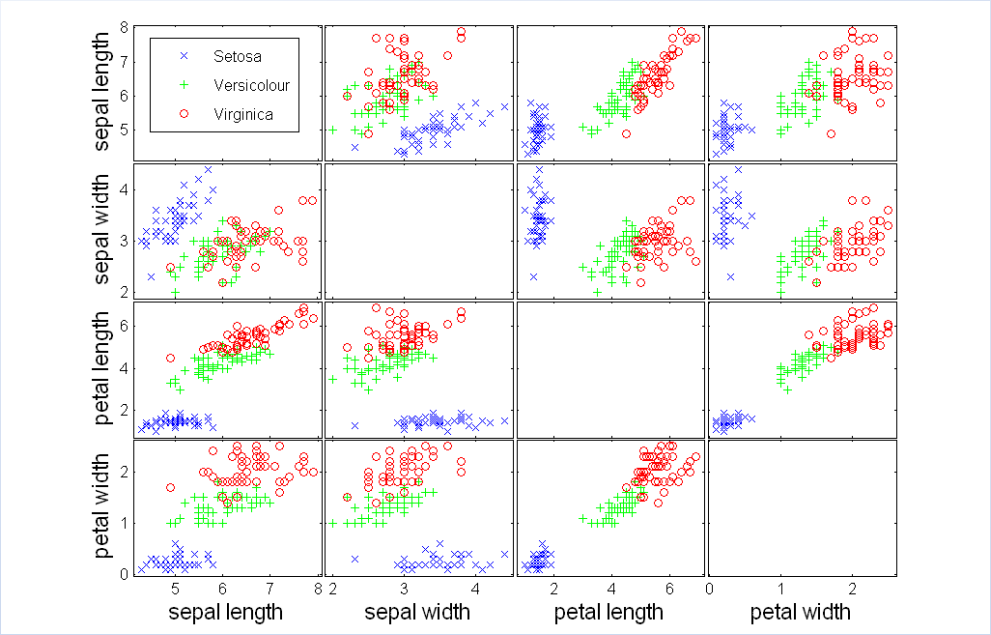
Principal components analysis (PCA)

- Find a new set of features that better captures the variability of the data
  - First dimension chosen to capture as much of the variability as possible;
  - The second dimension is orthogonal to the first, and subject to that constraint, captures as much of the remaining variability as possible,
  - The third dimension is orthogonal to the first and second, and subject to that constraint, captures as much of the remaining variability as possible,
  - Etc. Linear weighted combination of features (feature engineering).
- We will not be covering the mathematical details
  - Many tutorials available on the web. Nice application of linear algebra.
- A good visualisation: http://setosa.io/ev/principal-component-analysis

Iris dataset: Reduced from 4 to 2 dimensions using PCA



Contrast PCA against a scatter matrix



Scatter plots for iris dataset

AFL football dataset: From http://afltables.com/afl/stats/

[Stats Main][AFL Main]

[2013 Stats][2015 Stats]

2014 Player Stats

[2014 Stats Summary]

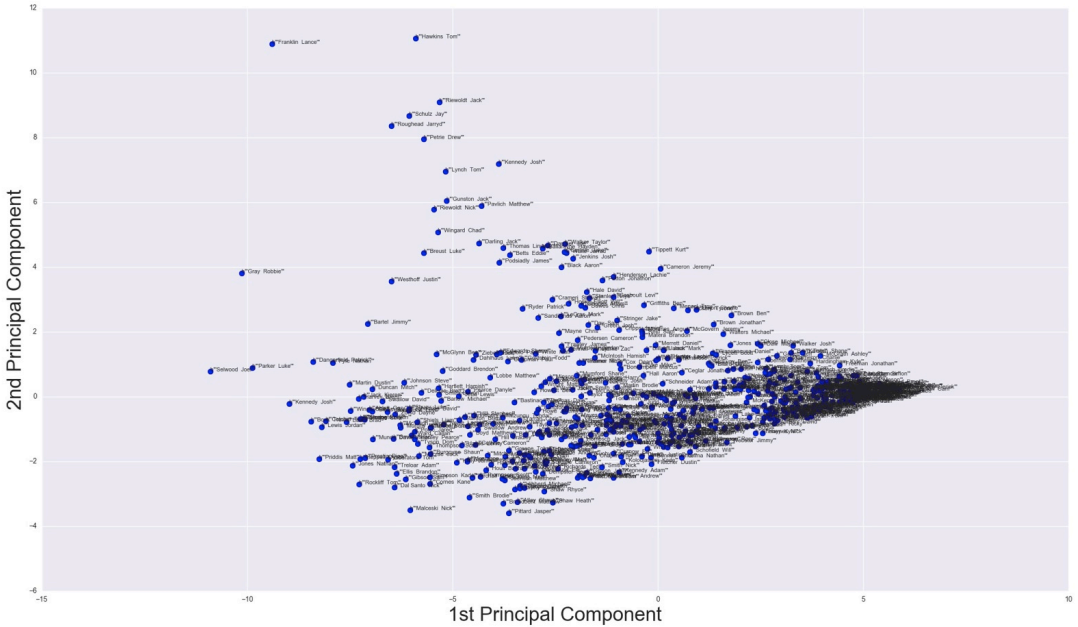
[Adelaide][Brisbane Lions][Carlton][Collingwood][Essendon][Fremantle][Geelong][Gold Coast][Greater Western Sydney][Hawthorn]
[Melbourne][North Melbourne][Port Adelaide][Richmond][St Kilda][Sydney][West Coast][Western Bulldogs]
[All Teams]

Abbreviations key

Adelaide [Game by Game]

#	Player	GM	KI	MK	HB	DI	DA	GL	BH	HO	TK	RB	IF	CL	CG	FF	FA	BR	CP	UP	CM	MI	1%	BO	GA	%P	SU
32	Dangerfield, Patrick	22	276	74	272	548	24.91	17	22	28	78	33	104	136	66	34	19	21	341	210	25	16	35	18	10	83.7	
9	Sloane, Rory	22	269	105	252	521	23.68	13	9	19	147	45	99	92	50	26	15	10	275	256	9	7	64	5	21	87.2	
5	Thompson, Scott	19	257	69	262	519	27.32	3	7	2	86	28	77	118	61	19	22	14	224	280	3	5	21	1	7	81.7	0/2
33	Smith, Brodie	22	287	108	209	496	22.55	11	8		35	109	76	18	45	9	6	4	142	319	7	2	56	46	7	87.0	
10	Jaensch, Matthew	22	297	126	166	463	21.05	7	5		54	89	54	7	34	19	10		106	325	16	1	57	34	3	81.3	
26	Douglas, Richard	19	266	52	147	413	21.74	11	8	4	91	21	96	91	38	22	17		182	228	2	6	36	13	11	86.4	
11	Wright, Matthew	20	224	89	150	374	18.70	14	8		68	22	47	39	27	30	6		141	227	4	12	26	6	17	80.0	1/2
24	Jacobs, Sam	22	193	90	165	358	16.27	7	3	763	46	20	40	69	33	11	15	6	150	189	19	4	63	1	10	87.9	0/1
14	Mackay, David	19	168	58	174	342	18.00	11	7		77	30	62	32	31	22	13		127	224	5	3	34	37	8	81.1	0/2
18	Betts, Eddie	22	167	53	123	290	13.18	51	22		74	8	37	30	39	19	16	4	149	136	3	29	21	8	29	87.7	
1	Podsiadly, James	21	189	119	101	290	13.81	26	14	2	37	17	52	2	63	14	25	4	132	165	41	35	60	1	16	90.1	
16	Brown, Luke	22	138	55	148	286	13.00	1	1		54	37	16	8	18	13	5		81	205	1	1	42	1	4	84.5	
2	Crouch, Brad	11	125	26	147	272	24.73	5	6	1	61	22	40	56	30	8	6		114	156	1	2	17	9	6	83.7	0/1
36	Martin, Brodie	17	155	65	109	264	15.53	8	15		45	30	38	23	40	13	11		97	174	7	12	34	11	4	69.2	2/1
12	Talia, Daniel	22	167	105	93	260	11.82		1		24	45	25		29	11	12		79	183	13		149	1	2	90.0	0/1
29	Laird, Rory	16	126	65	129	255	15.94	2	2		37	21	34	15	31	8	8		81	177	1	1	25	2	2	75.4	2/0
4	Jenkins, Josh	20	170	86	64	234	11.70	40	26	55	27	13	46	11	36	12	8	3	97	140	21	32	48	10	7	90.6	
13	Walker, Taylor	15	138	84	82	220	14.67	34	22		24		50		47	10	21	5	102	120	23	31	20		17	90.3	
3	Reilly, Brent	10	130	65	63	193	19.30				19	32	17	8	30	3	13		46	139	7		19	24	1	81.0	
17	Kerridge, Sam	14	72	33	84	156	11.14	10	1		52	10	23	26	25	3	14		54	97	2	9	9	4	5	83.7	0/1

AFL football dataset: PCA in 2D



Principal components analysis in Python

`sklearn.decomposition`

Will practise in workshop

Performance Evaluation

Performance Evaluation

- Metrics for Performance Evaluation
- Training versus Testing Data
- Sampling and Cross-Validation

Unsupervised algorithm evaluation

How would you evaluate the performance of a k-nearest neighbour clustering?

1. If a clustering 'looks good'?
2. Compare to some 'known points'?

→ internal validation
→ how tight are clusters (far apart)

Thoughts?

Supervised algorithm Evaluation

For an accurate decision tree classifier, we want to minimise both:

- False positives (saying yes when we should say no)
- False negatives (saying no when we should say yes)

Thoughts?

Metrics for Performance Evaluation

For supervised algorithms - can be summarized in a Confusion Matrix (contingency table)

- Actual class: {yes, no, yes, yes, …}
- Predicted class: {no, yes, yes, no…}

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

correct answer

all answers

Metrics for Performance Evaluation

- Actual class: {yes, no, yes, yes, no, yes, no, no}
- Predicted: {no, yes, yes, no, yes, no, no, yes}

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a= 1 (TP)	b=3 (FN)
	Class=No	c=3 (FP)	d=1 (TN)

Accuracy=? $\frac{2}{8} = \frac{1}{4}$

Limitations of accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

- If a model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
  - Accuracy is misleading because we do not detect any class 1 example
  - Other metrics can be used instead of accuracy to address this problem

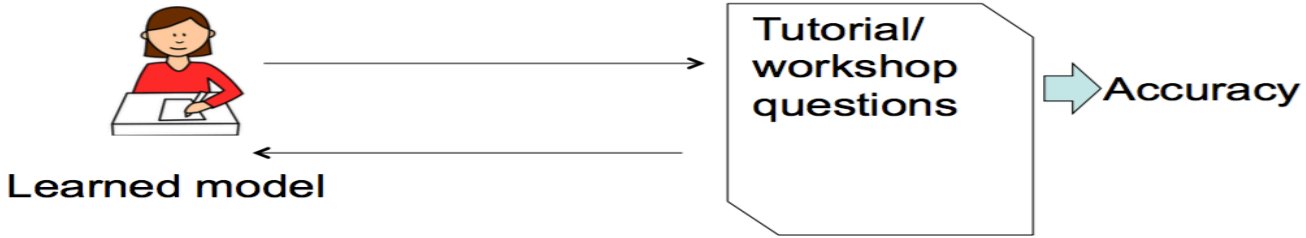
Imbalanced data - Bootstrapping

Procedure:

1. Draw k datasets from original data by sampling with replacement
2. Run the algorithm for each dataset and compute accuracy
3. Return the mean and standard deviation of accuracies

- Can estimate confidence in a prediction y=f(x)
- Can handle imbalanced data sets – biased sampling.

Why do we split the dataset into training and testing for evaluating accuracy?



Why do we split the dataset into training and testing for evaluating accuracy?

Learned model

Tutorial/ workshop questions

(training set)

Accuracy

Final exam

(test set)

Accuracy

Why do we need to split?

Overfitting

Decision tree: training and testing

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

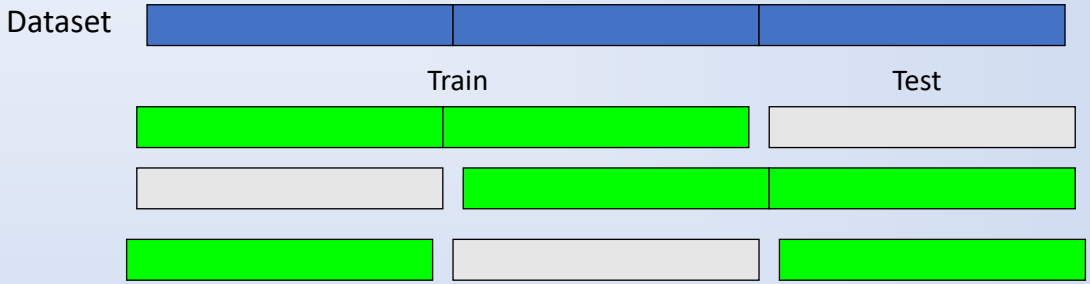
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

- Divide data into:
  - Training set (e.g. 2/3)
  - Testing set (e.g. 1/3)
- Learn decision tree using the training set
- Evaluate performance of decision tree on the testing set

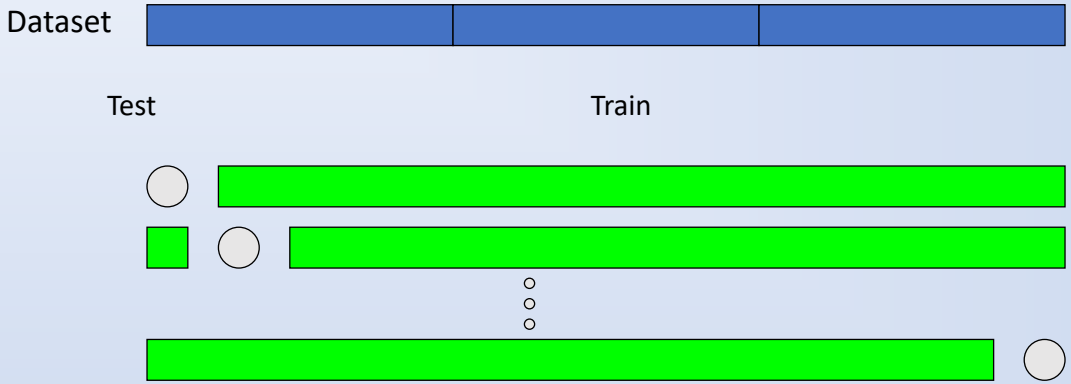
Common Splitting Strategies

k-fold cross-validation



Common Splitting Strategies

Leave-one-out (n-fold cross validation)



Computational complexity

- k-fold cross validation requires
  - k training steps on n(k-1)/k data points
  - k testing steps on n/k data points
- We typically also repeat the above 10-100 times to reduce any bias from random number choices
- Report mean results and perform statistical comparisons, e.g. T-tests

2017 exam question 3f

- (2 marks) What is the purpose of separating a dataset into training and test sets, when evaluating the performance of a classifier?

when we test on the training set directly, then we will get overfitting

⇒ split { training set / test set

learn from one set, test on the other set

Points to remember

- Understand the difference between supervised and unsupervised algorithms
- Understand the principles of experimental design: Feature Filtering, Dimensionality Reduction, Performance Evaluation
- Go from average to great data analysis

Cutting-edge research: Semi-Supervised Learning

JM Reps, JM Garibaldi, U Aickelin, D Soria, JE Gibson, RB Hubbard: “A Novel Semi-Supervised Algorithm for Rare Prescription Side Effect Discovery (DRESS)”, IEEE Journal of Biomedical and Health Informatics 18 (2), 537-547, 2014.

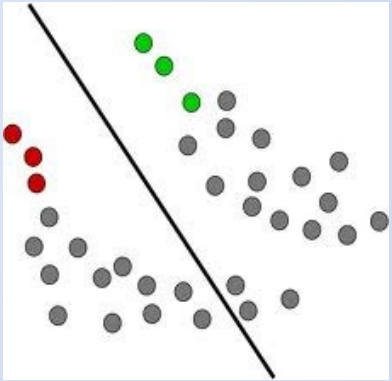
Can we automate the detection of adverse prescription side effects by utilising large databases of electronic health records?

Motivation

- Help patients
- Existing methods (unsupervised) have a high false positive rate
- Supervised methods may have better performance – but labels are rare
- If we use knowledge of known data, can generate ‘a few more’ labels?
- Semi-supervised methods?!

DRESS algorithm overview (1)

- The number of known Adverse Drug reactions for a specific drug is limited.
- When labels are limited, semi-supervised methods may be suitable.
- Semi-supervised clustering incorporates the limited additional knowledge to bias the clustering process



DRESS algorithm overview (2)

Step 1: Generate attributes (feature selection)

Step 2: Extract labels (special feature engineering)

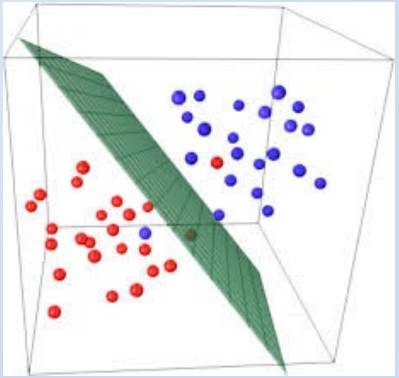
Step 3: Metric learning (dimensionality reduction)

Step 4: Semi-supervised K-means (clustering)

Step 5: Filter and Rank (performance evaluation)

DRESS algorithm 1- Generate Attributes

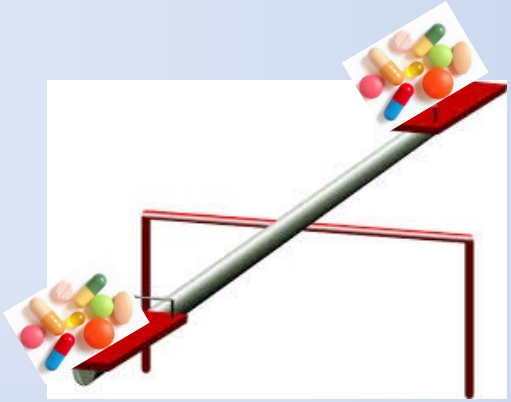
Need to generate attributes that create a space where Adverse Drug Reactions are separable from non-Adverse Drug Reactions



DRESS algorithm 1 – Generate attributes

- The attributes are based on existing methods
- The Before/After ratios 1: how often the medical event occurs during the month before the drug compared to the month after
- The Before/After ratios 2: a comparison between how often the drug occurs before the medical event or after

AFTER



BEFORE

DRESS algorithm 1 – Generate Attributes

Before and after ratio features:

- month after / month before
- year after / year before
- same day / year before
- month after and not in month before / month after
- month after / month before (first 3 Read code elements)
- month after /month before (first 2 Read code elements)

(A Read code is a UK hierarchical classification code of a medical event)

DRESS algorithm 1 – Generate Attributes

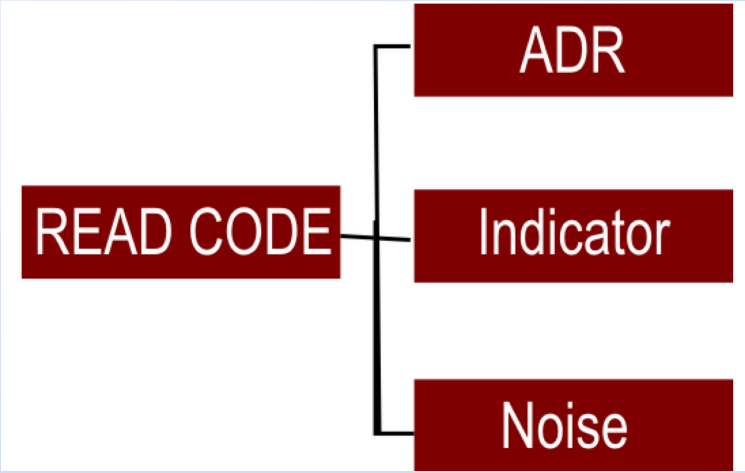
What the data looks like at this point:

So we have...

Read code	Attributes	Label
G1234	2.3,1,0,2,2,1,1,3.4,5.6	?
H56g3	0.6,1,1,0.2,0.8,0.5,0.56,1.4,0.67	?
...	...	?

DRESS algorithm 2 – extract labels

Labels



a drug will have negative consequence

positive consequence

DRESS algorithm 2 – extract labels

Extract known Adverse Drug Reactions by mining the web. Search for medical events listed as Adverse Drug Reactions to the drug of interest on medical websites (e.g. bnf.org, patient.com)

- Do a string match to find the corresponding Read Codes
- Validate

DRESS algorithm 2 – extract labels

Extract indicators by mining the web. Search for medical events listed as indications (cause of taking the drug) to the drug of interest on medical websites (e.g. bnf.org, patient.com)

- Do a string match to find the corresponding Read Codes
- Validate

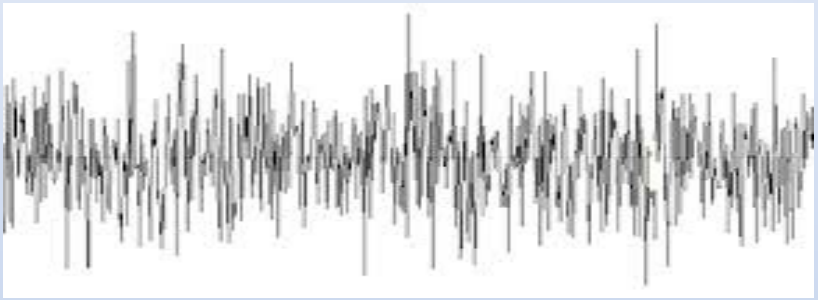
Indication: Diabetes

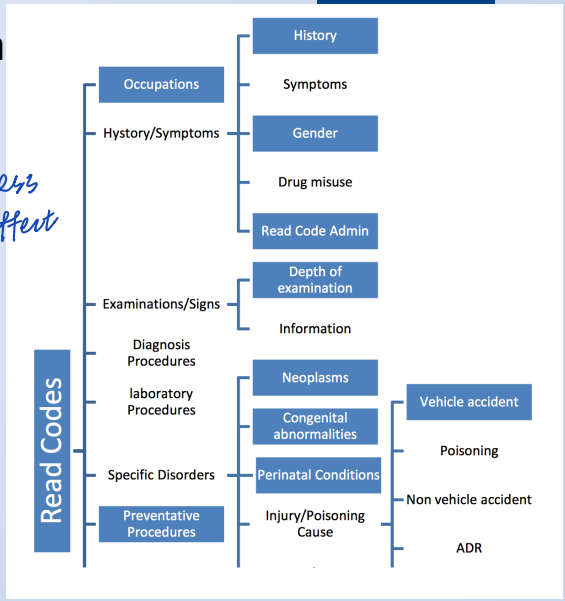
DRESS algorithm 2 – extract labels

Extract noise events using the hierarchal Read Code structure. Find Read Code branches via a manual search that are highly unlikely to correspond to acute side effects (e.g. cancer, chronic illnesses, family history,…)

long-term illnesses

long term illness not side-effect





DRESS algorithm 2 – extract labels

What the data look like at this stage:

So we have...

Read code	Attributes	Label
G1234	2.3,1,0,2,2,1,1,3.4,5.6	ADR
H56g3	0.6,1,1,0.2,0.8,0.5,0.56,1.4,0.67	?
...	...	?

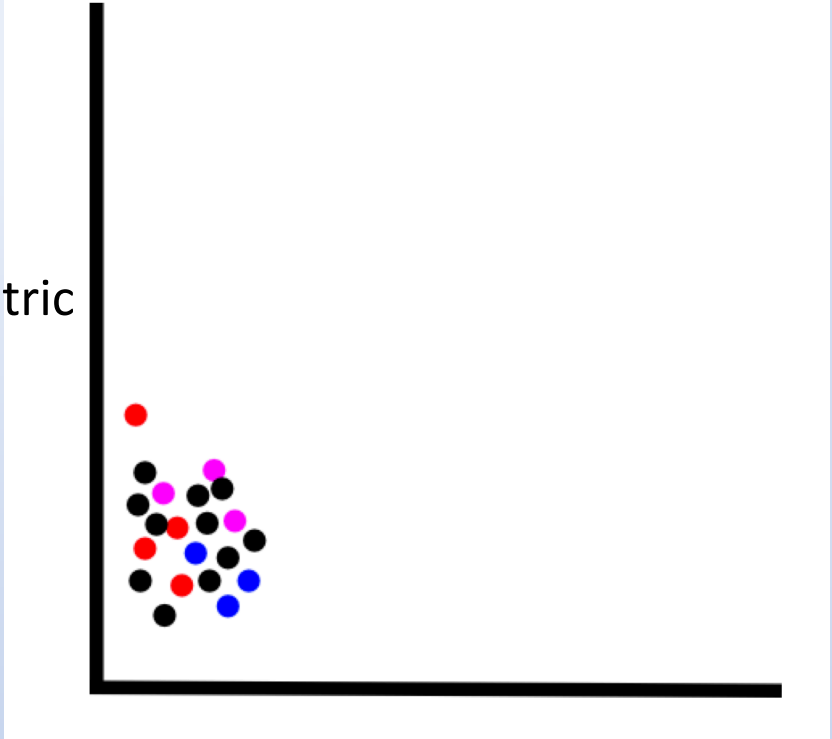
DRESS algorithm 3 – Metric learning

Use the attributes and labels generated previously to find the suitable metric that transforms the attribute space into one that optimises the class separations.

Ref: Y. Ying and P. Li "Distance metric learning with eigenvalue optimization", *J. Mach. Learn. Res.*, vol. 13, pp.1 -26 2012

DRESS algorithm 3 – Metric Learning

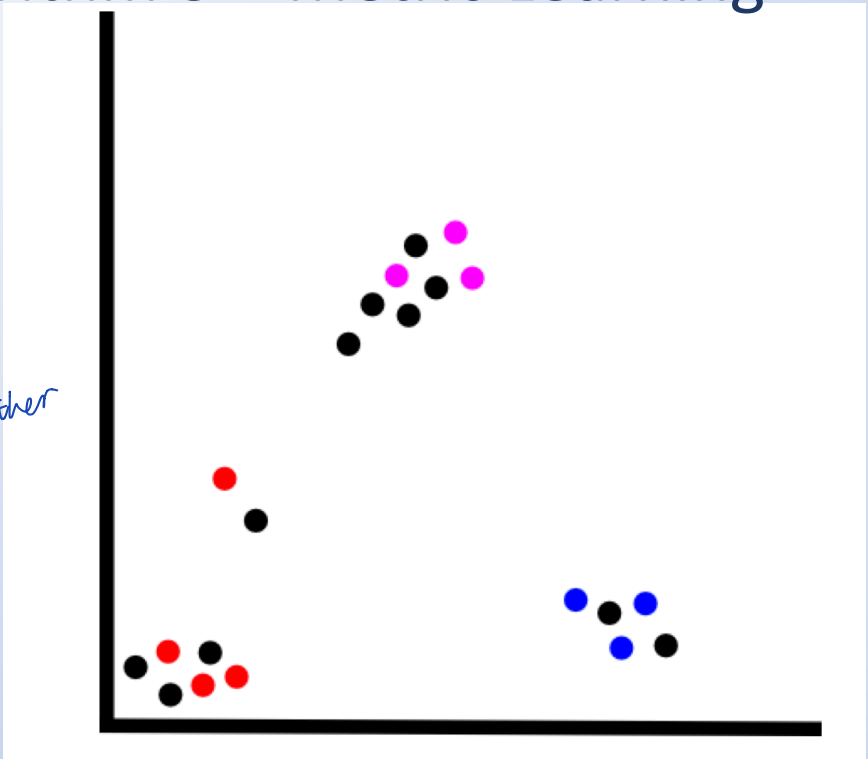
Before Metric Learning



DRESS algorithm 3 – Metric Learning

After Metric Learning

map a data to another space, more easy to separate by classifier



DRESS algorithm 4 – Semi-supervised K-Means

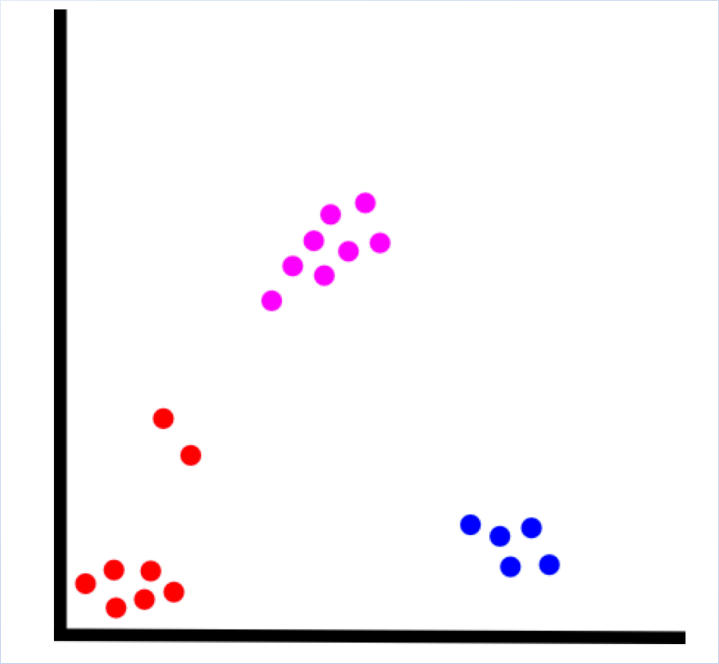
instead of random clustering seed, we use cluster seed which has labels.

- In the transformed space apply constrained k-means*
- The seeds are the known label sets (seed 1: Adverse drug reactions, seed 2: indications and seed 3: noise)
- The initial k-means centers are created using the labeled data averages for each seed
- K-means then assigns the unlabeled data point to the closest cluster and the labeled points remain fixed

*S. Basu , A. Banerjee and R. J. Mooney "Semi-supervised clustering by seeding", *Proc. 19th Int. Conf. Mach. Learn.*, vol. 2, pp.27 -34 2002

DRESS algorithm 4 – Semi-supervised K-Means

Step 4: Apply Semi-supervised K-means



DRESS algorithm 4 – Semi-supervised K-Means

What the data look like at this stage:

So we have...

Read code	Attributes	Label
G1234	2.3,1,0,2,2,1,1,3.4,5.6	ADR
H56g3	0.6,1,1,0.2,0.8,0.5,0.56,1.4,0.67	Indicator
...	...	...

DRESS algorithm 5 – Filter and Rank

- Filter Read codes that cannot be immediately occurring ADRs (cancers, admin, ...)
- Filter Read codes in the indicator cluster

DRESS algorithm 5 – Filter and Rank

Order by: month after not month before/month before * (1/weight), weight = 1 for Read codes in ADR cluster and weight = 3 for Read codes in noise cluster

more timely the event happens, more likely whether it is a real side effect

DRESS algorithm 5 – Filter and Rank

Final example of data:

So we end up with...

Rank	Read code	Attributes	Label	Measure
1	G1234	2.3,1,0,2,2,1,1,3.4,5.6	ADR	5.6
2	D1...	3.2,0,0,2.1,1.5,3,1,1,6.2	Noise	5.5
3	A12..	2.2, 0,0,3,1,4,3.2,1.4,1.6	ADR	4.8
...	...	...	...	

Measure = month after not month before/month before * (1/weight)

DRESS - RESULTS

		Ours	Existing		
Drug	ADR	DRESS	OE ratio	HUNT	MUTARA
Rofecoxib	Inferior myocardial infarction	83	302	196	1212
Rofecoxib	Heart failure	121	805	504	562
Rimonabant	Depressed mood	16	32	15	68
Rimonabant	Depressive disorder NEC	23	180	295	1157
Enalapril	Acute pancreatitis	53	700	416	2081
Naproxen	Acute renal failure	20	159	138	1313
Naproxen	Jaundice	52	774	3394	4190
Naproxen	Hepatitis unspecified	241	35	852	3502
Naproxen	Suicide and selfinflicted injury	104	223	242	1808
Nifedipine	Weight increasing	362	29	1517	1492
Nifedipine	Feels hot/feverish	314	428	1972	7480
Ciprofloxacin	Nontraumatic rupture of Achilles tendon	13	42	1273	5506
Levofloxacin	Rupture Achilles tendon	76	296	-	-
Levofloxacin	Nontraumatic rupture of Achilles tendon	298	279	320	1189
Rifampicin	Thrombocytopenia NOS	75	418	265	1003
Metformin	Lactic acidosis	-	-	-	-
Metformin	Acidosis	449	813	466	3223

Acknowledgements

- Materials are partially adopted from …
  - Previous COMP2008 slides including material produced by James Bailey, Pauline Lin, Chris Ewin, Uwe Aickelin and others
  - “Data Mining Concepts and Techniques”, Han et al, 2nd edition 2006.
  - “Introduction to Data Mining”, Tan et al 2005.
  - https://www-users.cs.umn.edu/~kumar/dmbook/ch4.pdf
  - http://people.duke.edu/~kh269/teaching/b351/20evaluatinglearning.pptx