

DATA SCIENCE IN MANUFACTURING

WEEK 8

ANDREW SHERLOCK, JONATHAN CORNEY, DANAI KORRE

Activity Recognition



```
if(speed<4){  
    status=WALKING;  
}
```



```
if(speed<4){  
    status=WALKING;  
} else {  
    status=RUNNING;  
}
```



```
if(speed<4){  
    status=WALKING;  
} else if(speed<12){  
    status=RUNNING;  
} else {  
    status=BIKING;  
}
```



// Oh crap

RECALL LAST WEEK



Conventional Programming

Activity Recognition



```
01010010101001010  
1001010101001011101  
01001010100101001  
01010010101001010
```

Label = WALKING



```
10101001010010101  
0101010010010001  
00100111101010111  
1010100100111101011
```

Label = RUNNING



```
100101001111010101  
1101010111010101110  
1010101111010101011  
1111110001111010101
```

Label = BIKING



```
1111111110100111101  
0011111010111110101  
0101110101010101110  
1010101010100111110
```

Label = GOLFING
(Sort of)



Machine Learning

UNSUPERVISED LEARNING

Activity Recognition



0101001010100101010
1001010101001011101
0100101010010101001
0101001010100101010



1010100101001010101
0101010010010010001
00100111101010111
1010100100111010111



1001010011111010101
1101010111010101110
1010101111010101011
1111110001111010101

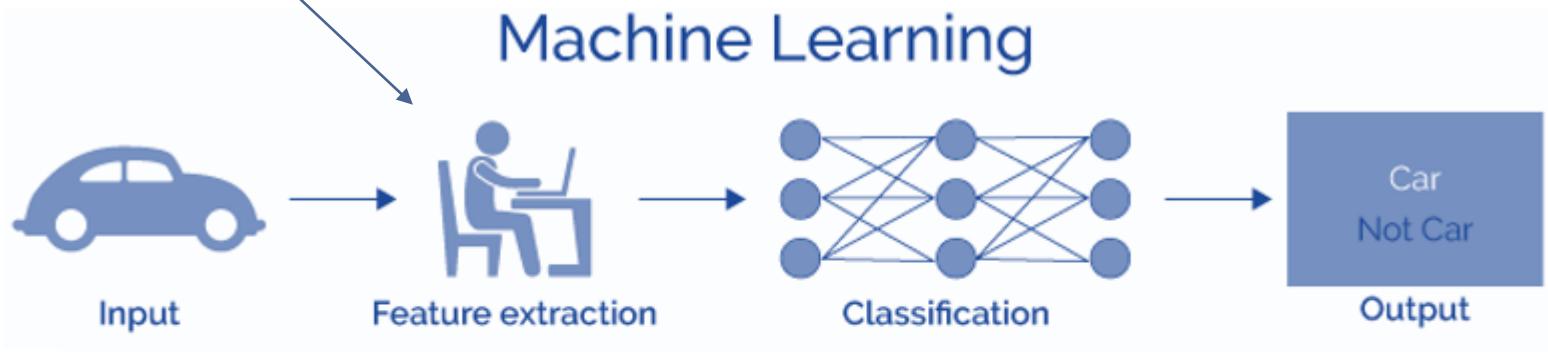


111111111010011101
001111010111110101
0101110101010101110
1010101010100111110

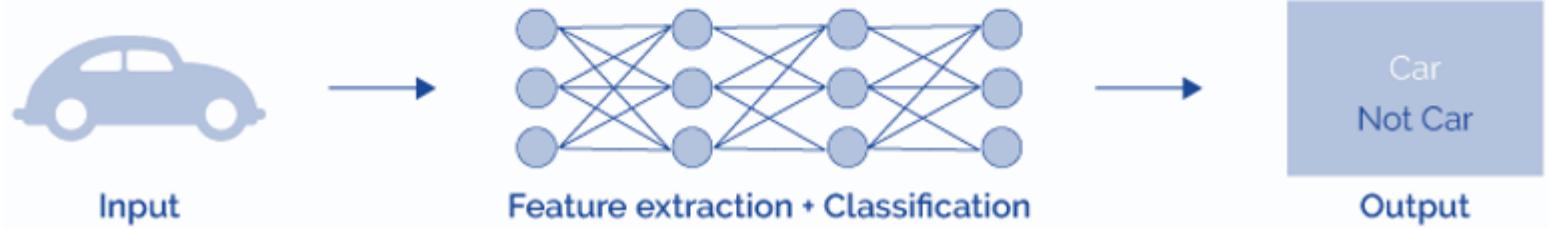




SUPERVISED LEARNING



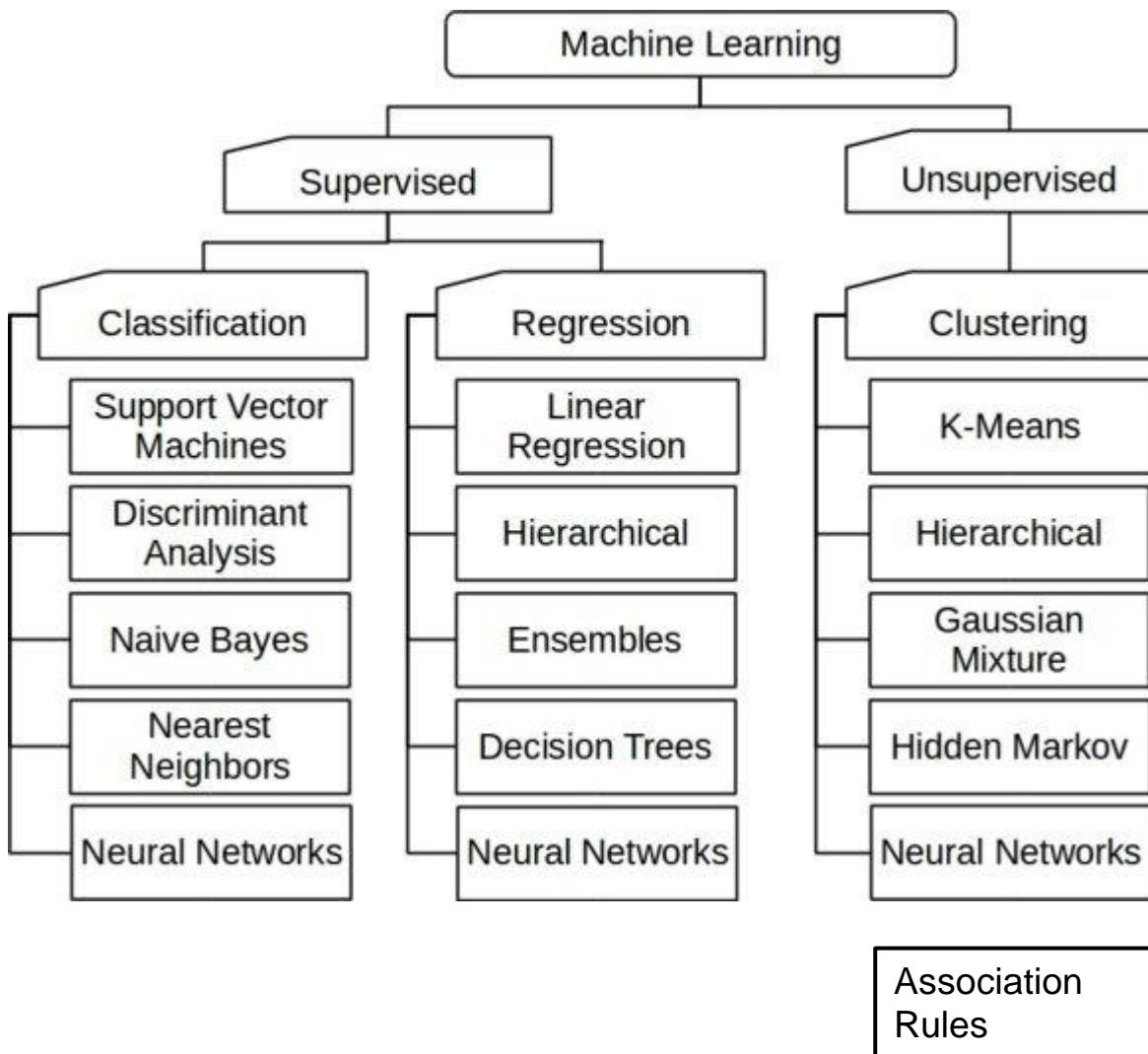
Deep Learning



UNSUPERVISED LEARNING



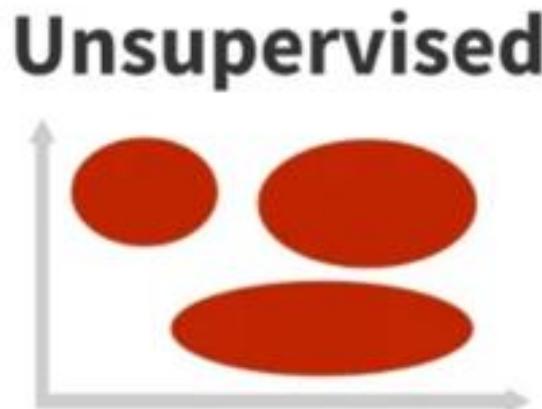
A MAJOR CLASS OF MACHINE LEARNING METHODS



We are just scratching the surface of a big subject

WHAT IS UNSUPERVISED LEARNING?

Doug Rose defines unsupervised learning as “The machine makes all the observations on its own. It might now know all the different names and labels, but it will find patterns on its own.” [5]



Source: Ben Freundorfer



THE UNIVERSITY of EDINBURGH
School of Engineering

Google's Artificial Brain Learns to Find Cat Videos

When computer scientists at Google's mysterious X lab built a neural network of 16,000 computer processors with one billion connections and let it browse YouTube, it did what many web users might do -- it began to look for cats.



By Liat Clark, Wired UK

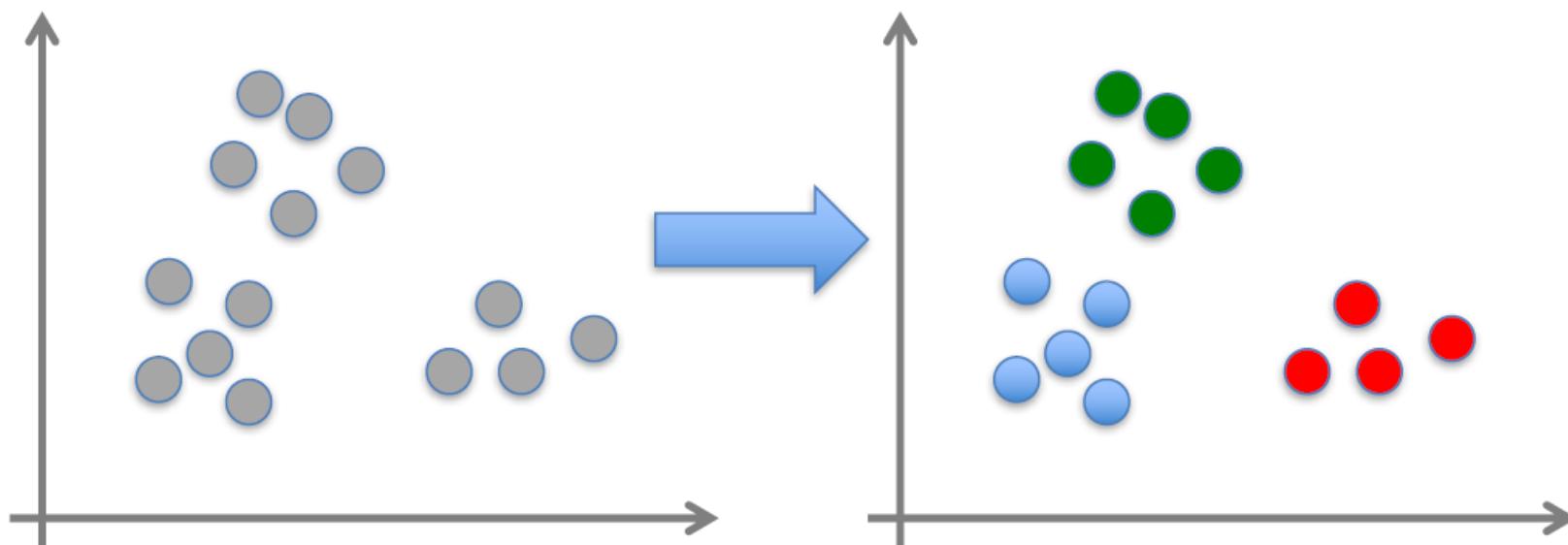
When computer scientists at Google's mysterious X lab built a neural network of 16,000 computer processors with one billion connections and let it browse YouTube, it did what many web users might do -- it began to look for cats.

[partner id="wireduk"] The "brain" simulation was exposed to 10 million randomly selected YouTube video thumbnails over the course of three days and, after being presented with a list of 20,000 different items, it began to recognize pictures of cats using a "deep learning" algorithm. This was despite being fed no information on distinguishing features that might help identify one.

Picking up on the most commonly occurring images featured on YouTube, the system achieved 81.7 percent accuracy in detecting human faces, 76.7 percent accuracy when identifying human body parts and 74.8 percent accuracy when identifying cats.

UNSUPERVISED LEARNING

- Given x_1, x_2, \dots, x_n (without labels)
- Output hidden structure behind the x 's
 - E.g., clustering



CLUSTERING

Clustering is one main approach to **unsupervised learning**.

- It finds similarity groups in data, called clusters,
 - it groups data instances that are similar to (near) each other in one cluster and data instances that are very different (far away) from each other into different clusters.

Clustering is often considered synonymous with unsupervised learning.

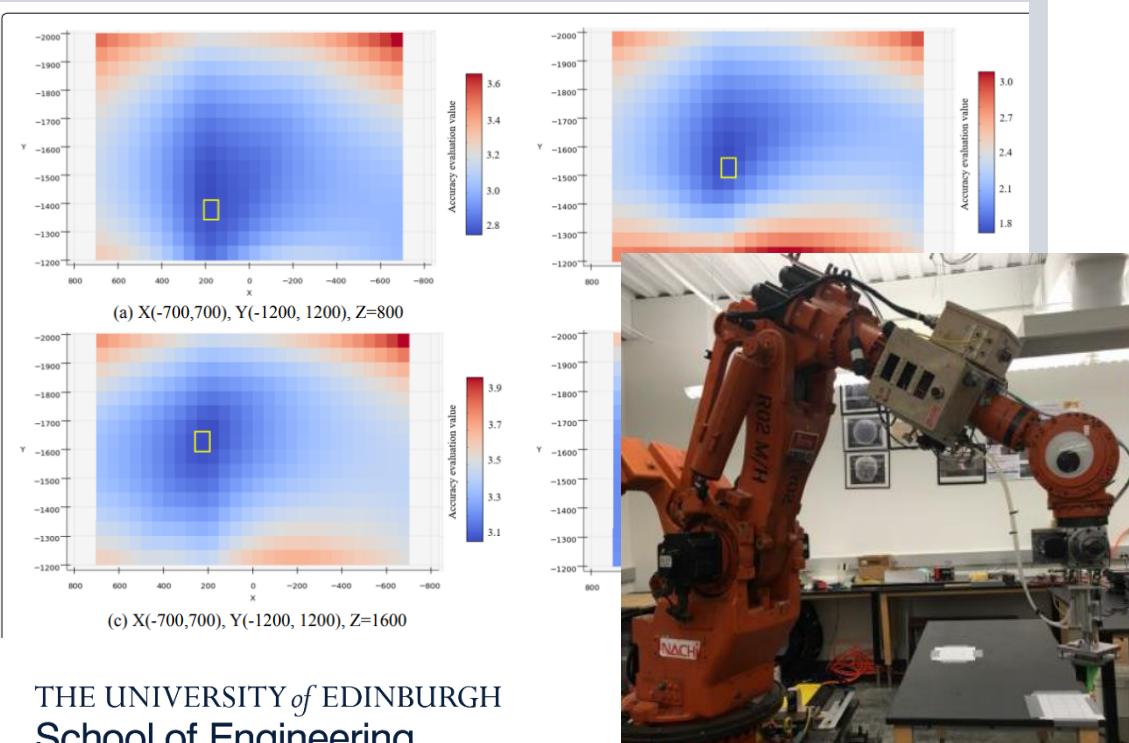
- But we'll look at some other types of pattern identification , such as association rule mining, that are also a form of unsupervised learning.

We will focuses on clustering first.



WHAT IS CLUSTERING FOR?

Example: Given a measurements of a robot's accuracy throughout its workspace, we want to organize them according to their content similarities,

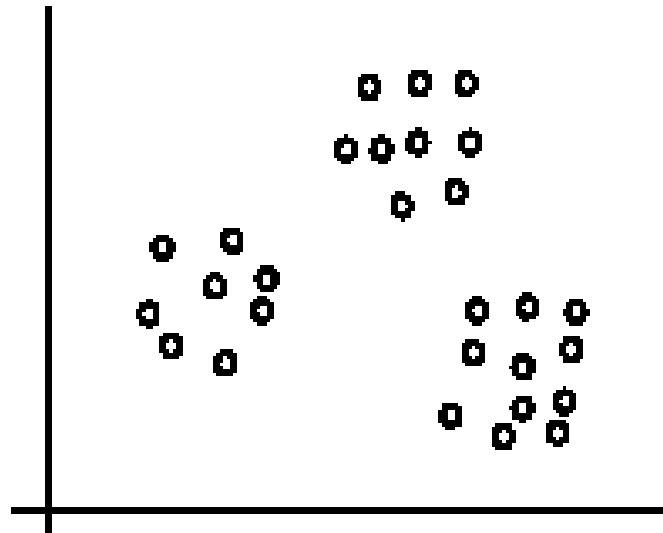


Clustering is one of the most utilized data mining techniques.

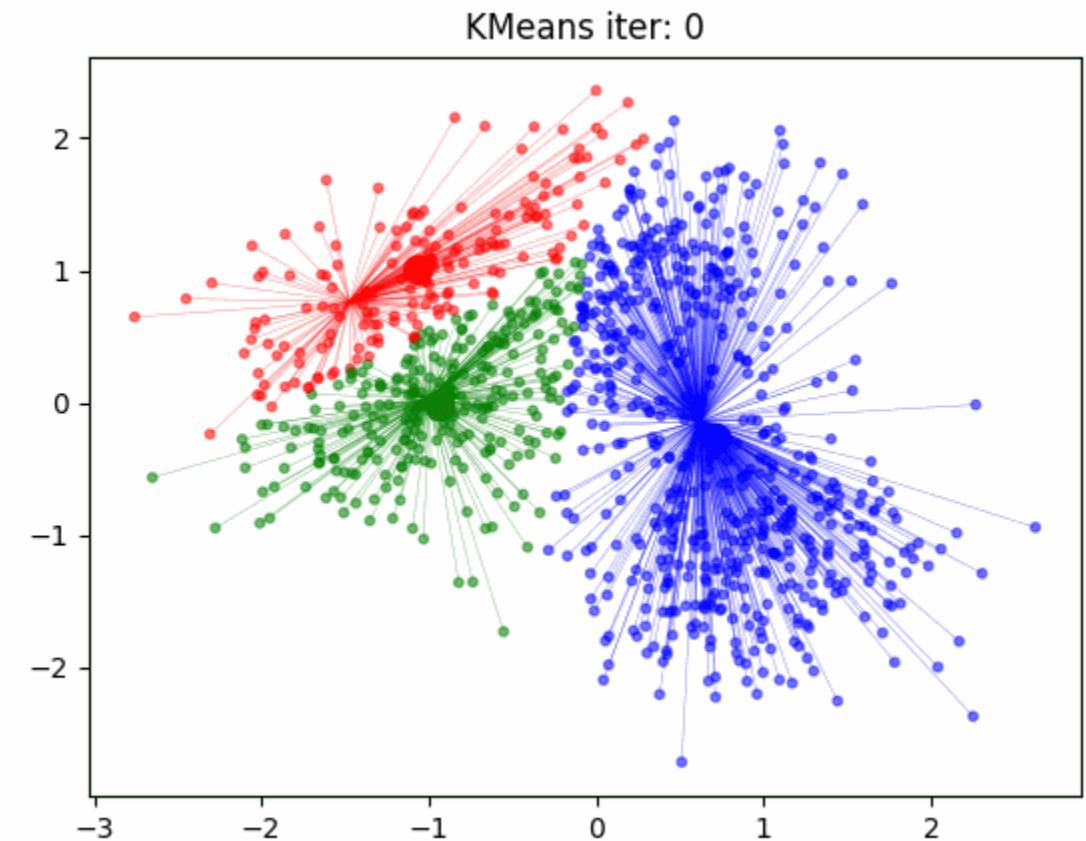
- It has a long history, and been used in almost every field, e.g., medicine, psychology, botany, sociology, biology, archeology, marketing, insurance, libraries, etc.
- In recent years, due to the rapid increase of online documents, text clustering becomes important.

Source: Bing Liu, UIC

“Easy” to cluster a data set which has three “natural” groups of data points, i.e., 3 natural clusters.



Source: Bing Liu, UIC

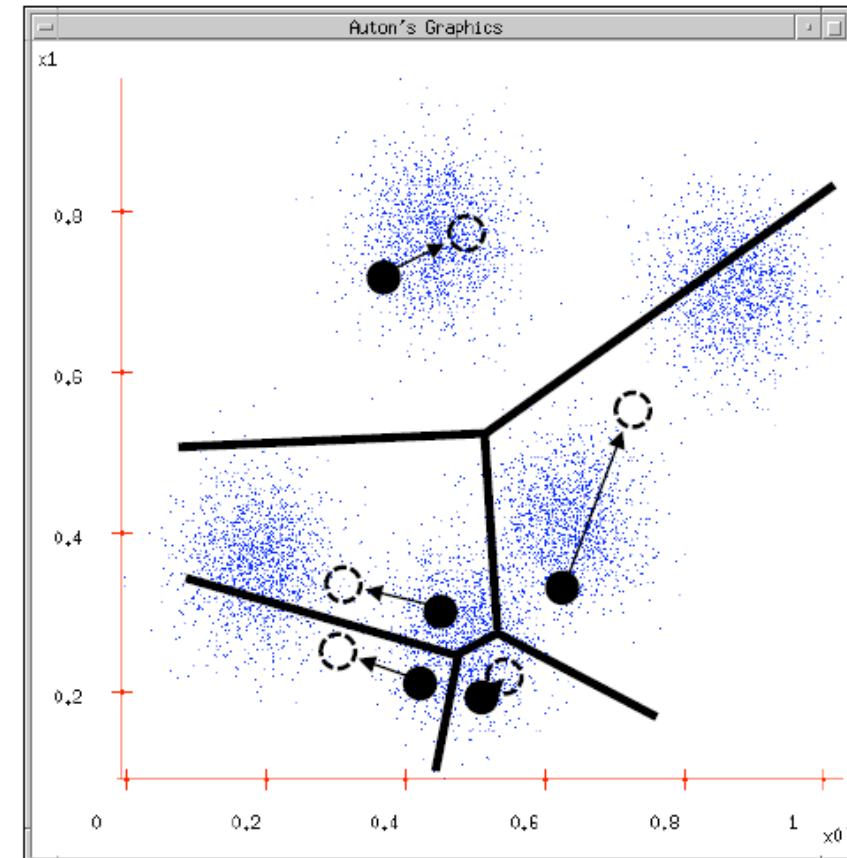


But often the clusters are not obvious so a range of algorithms have been developed to automatically collect points into groups or clusters

K-MEAN CLUSTERING

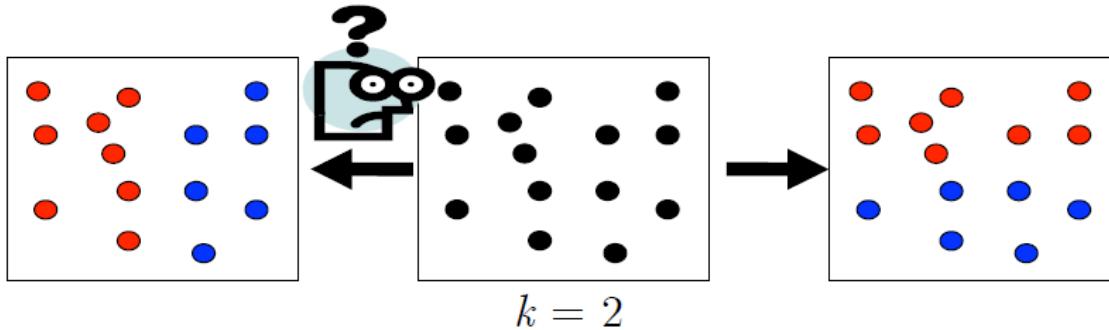
K-Means (k , X)

- Randomly choose k cluster center locations (centroids)
- Loop until convergence
 - Assign each point to the cluster of the closest centroid
 - Re-estimate the cluster centroids based on the data assigned to each cluster

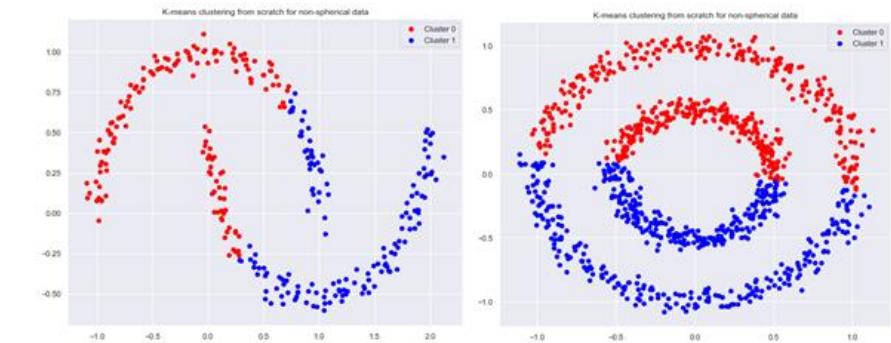
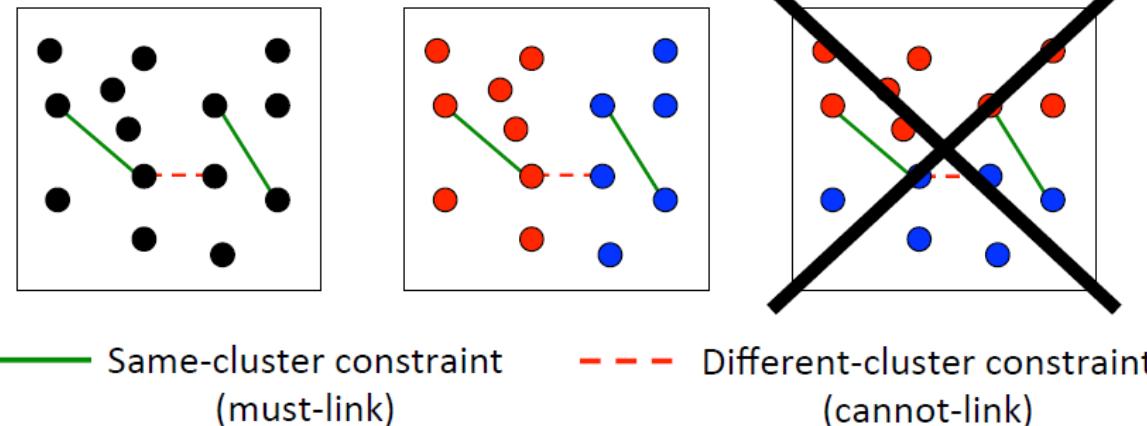


PROBLEMS WITH K-MEANS

- How do you tell it which clustering you want?

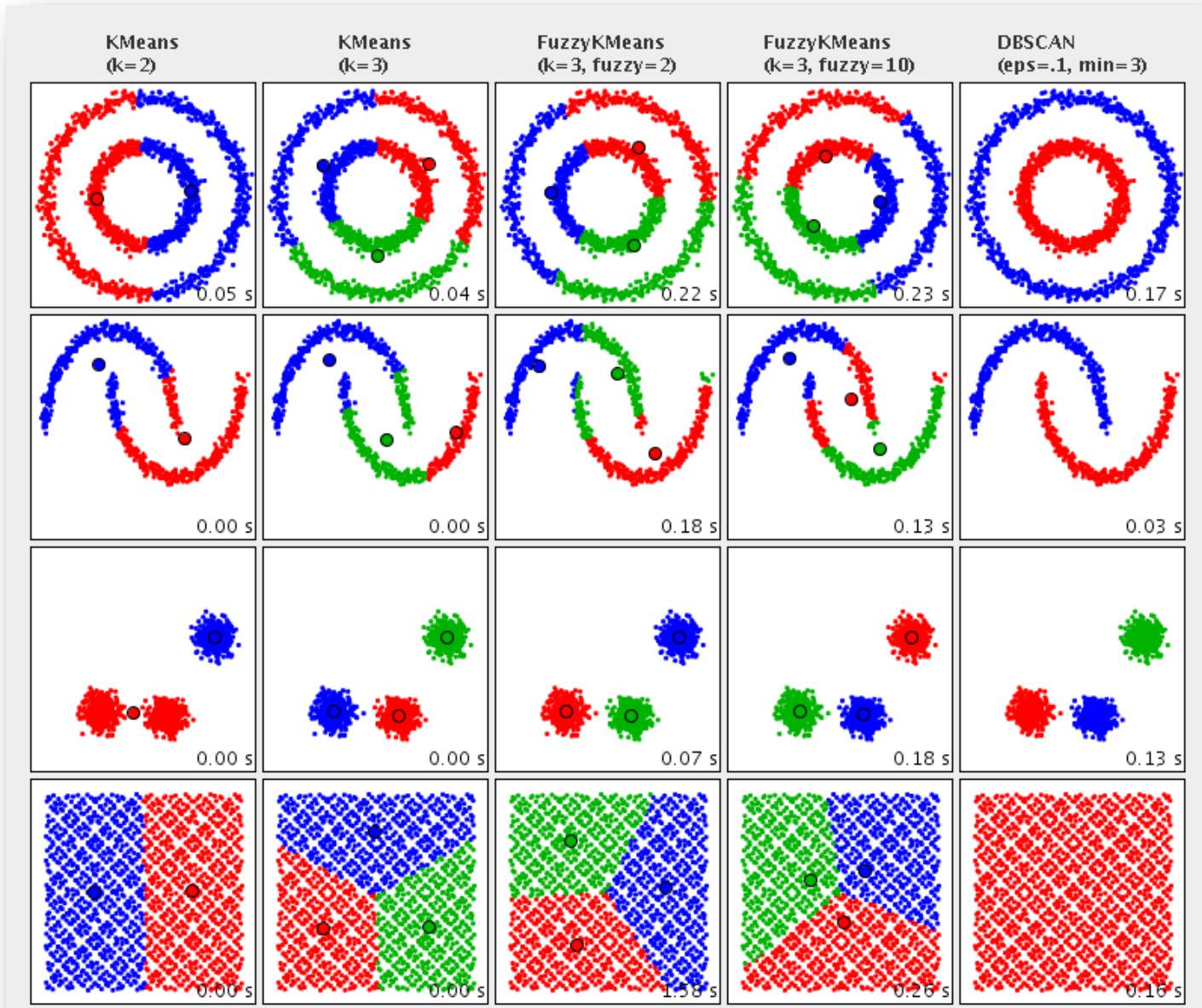


Constrained clustering techniques (semi-supervised)

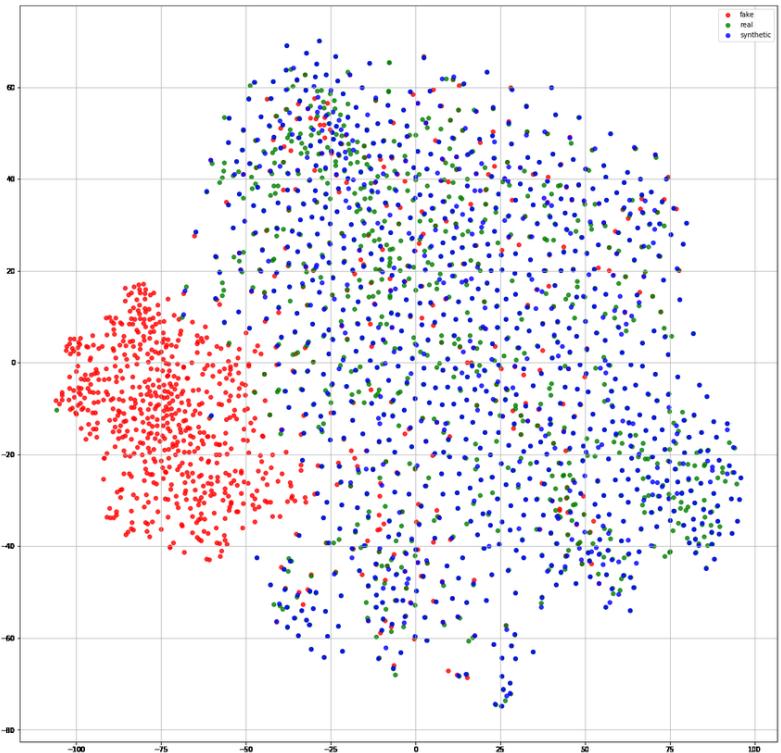


..and if your clusters are not blob shaped the results are not intuitive (i.e. what you expect)!

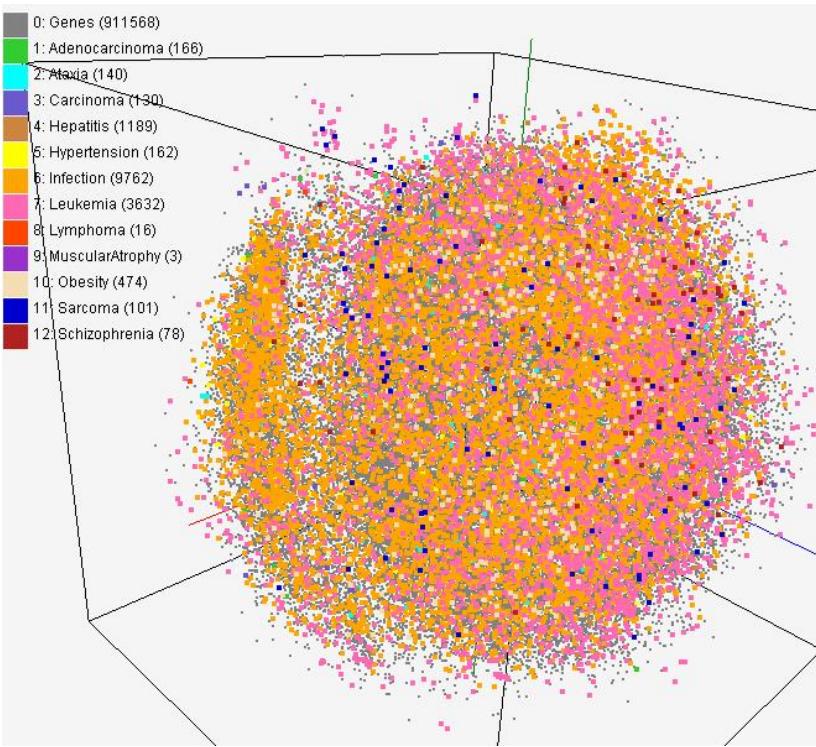
CLUSTERING ALGORITHMS FOR EVERY RESULT!



DATA DIMENSIONS



2D Data (x,y)



3D Data (x,y,z)

Basic symbol	d [mm]	D [inch]	D [mm]	B [inch]	[mm]	r_s [inch]	[mm]	d_s [inch]	[mm]	d_a [inch]	[mm]	D_a [inch]	[mm]	
15875A	15.875	.625	22.225	.875	3.967	.156	0.25	.010	16.9	.665	17.9	.705	20.6	.811
15875A-2Z	15.875	.625	22.225	.875	4.978	.196	0.25	.010	16.9	.665	17.9	.705	20.6	.811
15875A-2TS	15.875	.625	22.225	.875	4.978	.196	0.25	.010	16.9	.665	17.2	.677	20.6	.811
19050A	19.050	.750	25.400	1.000	3.967	.156	0.25	.010	20.1	.791	21.1	.831	23.7	.933
19050A-2Z	19.050	.750	25.400	1.000	4.978	.196	0.25	.010	20.1	.791	21.1	.831	23.7	.933
19050A-2Z	19.050	.750	25.400	1.000	4.978	.196	0.25	.010	20.1	.791	20.4	.803	23.7	.933
22225A	22.225	.875	28.575	1.125	3.967	.156	0.25	.010	23.3	.917	24.3	.957	26.9	1.059
22225A-2Z	22.225	.875	28.575	1.125	4.978	.196	0.25	.010	23.3	.917	24.3	.957	26.9	1.059
22225A-2TS	22.225	.875	28.575	1.125	4.978	.196	0.25	.010	23.3	.917	23.6	.929	26.9	1.059
26988A	26.988	1.063	33.338	1.313	3.967	.156	0.25	.010	28.1	1.106	29.1	1.146	31.7	1.248
26988A-2Z	26.988	1.063	33.338	1.313	4.978	.196	0.25	.010	28.1	1.106	29.1	1.146	31.7	1.248
26988A-2TS	26.988	1.063	33.338	1.313	4.978	.196	0.25	.010	28.1	1.106	28.4	1.118	31.7	1.248
31750A	31.750	1.250	38.100	1.500	3.967	.156	0.25	.010	32.8	1.291	33.8	1.331	36.4	1.433
31750A-2Z	31.750	1.250	38.100	1.500	4.978	.196	0.25	.010	32.8	1.291	33.8	1.331	36.4	1.433
31750A-2TS	31.750	1.250	38.100	1.500	4.978	.196	0.25	.010	32.8	1.291	33.1	1.303	36.4	1.433
34925A	34.925	1.375	41.275	1.625	3.967	.156	0.25	.010	36.0	1.417	37.0	1.457	39.5	1.555
34925A-2Z	34.925	1.375	41.275	1.625	4.978	.196	0.25	.010	36.0	1.417	37.0	1.457	39.5	1.555
34925A-2TS	34.925	1.375	41.275	1.625	4.978	.196	0.25	.010	36.0	1.417	36.3	1.429	39.5	1.555

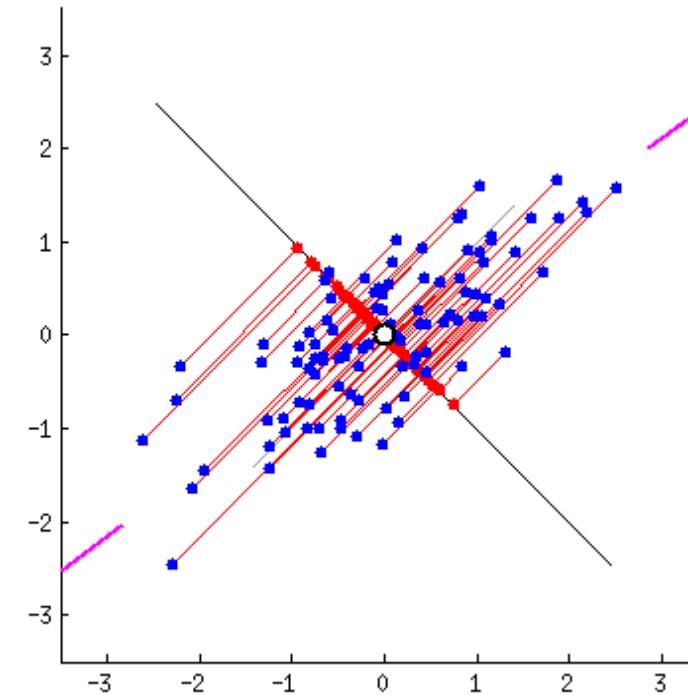
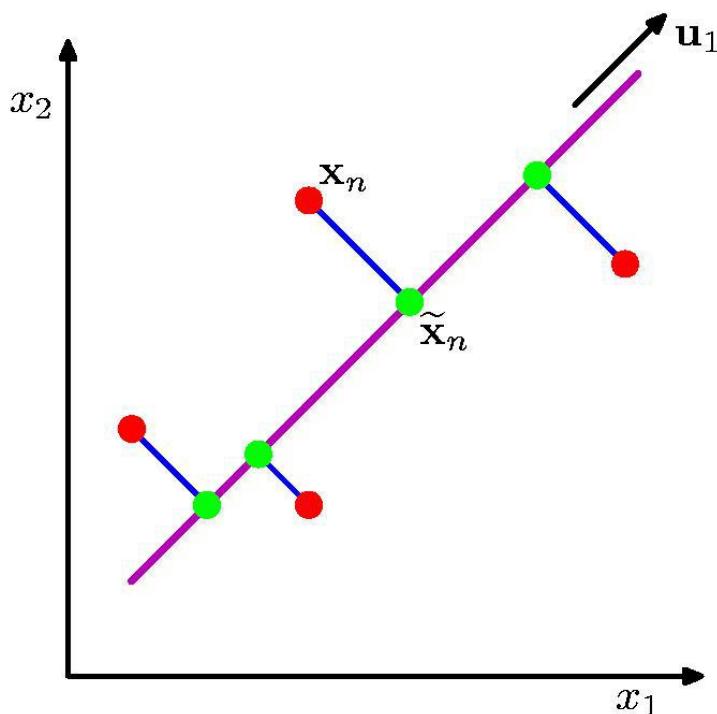
Subject to change.

7D Data (d,D,B,r,d1,d2,D1)

You can cluster high dimensional data ...but how can you visualize the results?



PRINCIPLE COMPONENT ANALYSIS (PCA)

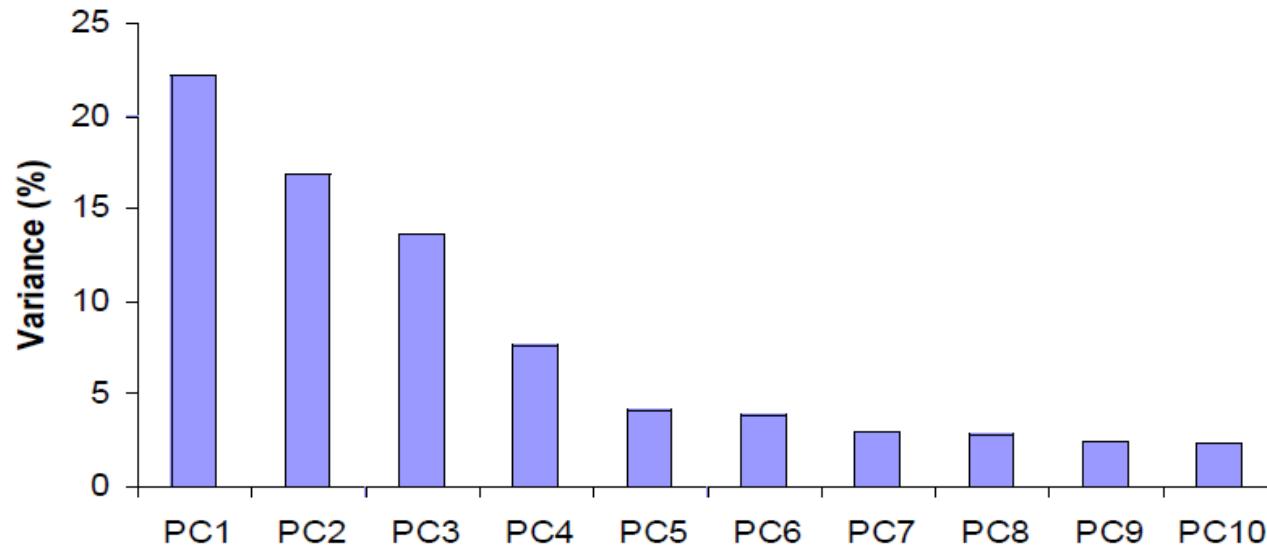


Orthogonal projection of data onto lower---dimension linear space that...

- Maximizes variance of projected data (purple line)
- Minimizes mean squared distance between data point and projections (sum of blue lines)

DIMENSIONALITY REDUCTION

Can *ignore* the components of lesser significance

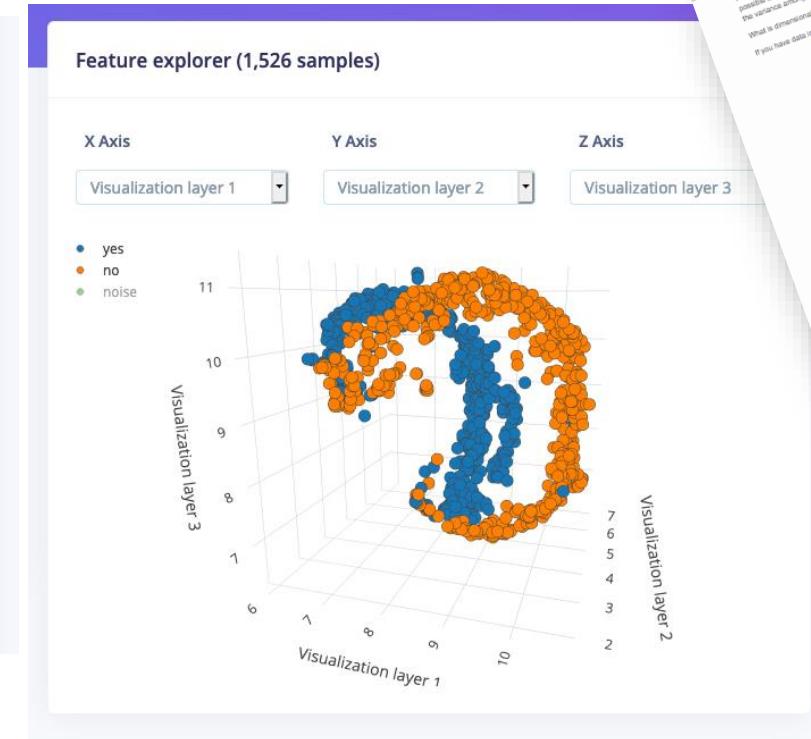
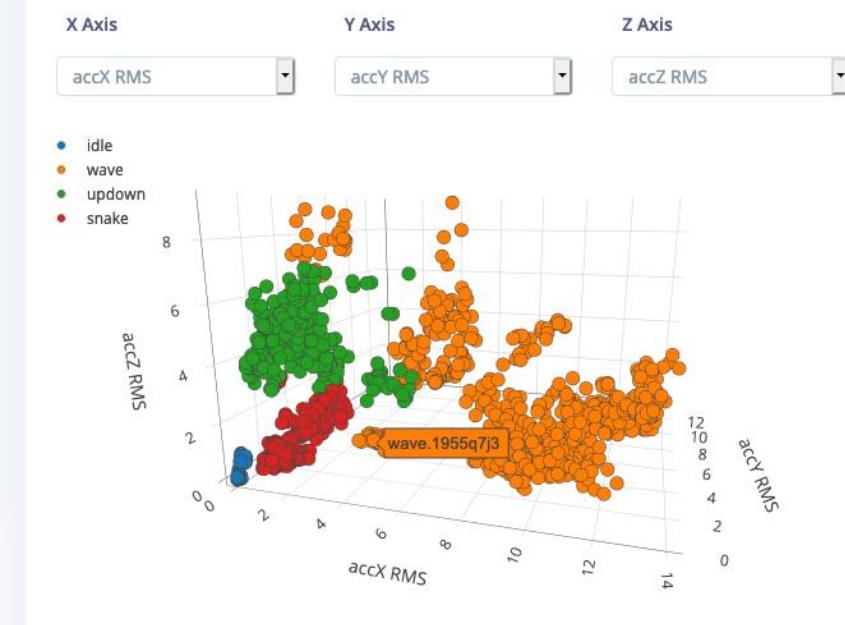


You do *lose some information*, but if the eigenvalues are small, you don't lose much

- Choose only the first k eigenvectors, based on their eigenvalues
- Final data set has only k dimensions



PCA APPLICATIONS



PCA is used for dimensionality reduction

It allows N-Dimensional Data to be “Projected” on to 3D or 2D spaces

After the break we will look at an example of using PCA to investigate Mechanical Seal Wear!

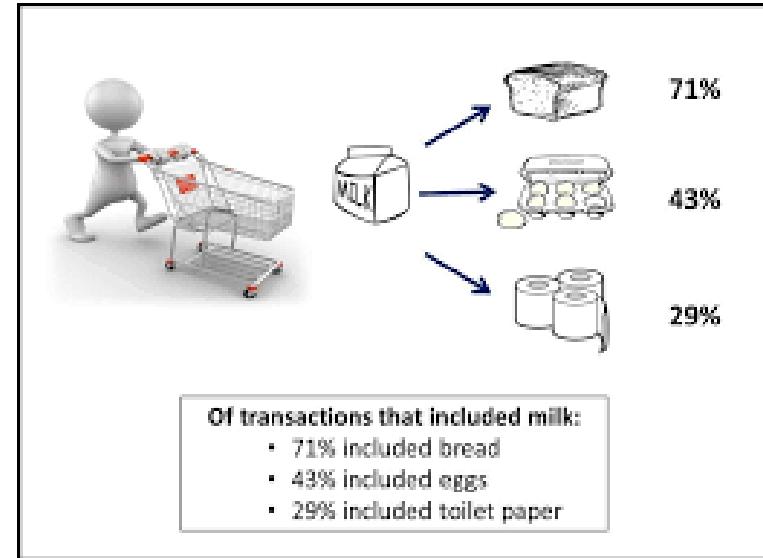


ITEM SETS & ASSOCIATION RULES

Market-Basket transactions

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

↑
Itemsets



Example of Association Rules

$\{\text{Diaper}\} \rightarrow \{\text{Beer}\}$,
 $\{\text{Milk, Bread}\} \rightarrow \{\text{Eggs, Coke}\}$,
 $\{\text{Beer, Bread}\} \rightarrow \{\text{Milk}\}$,

Implication means co-occurrence,
not causality!

ASSOCIATION RULES
HAVE TWO PROBABILITIES ASSOCIATED WITH THEM KNOWN AS SUPPORT (S) AND
CONFIDENCE (C)

<i>TID</i>	<i>Items</i>
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Rules:

- $\{\text{Milk}, \text{Diaper}\} \rightarrow \{\text{Beer}\}$ (s=0.4, c=0.67)
- $\{\text{Milk}, \text{Beer}\} \rightarrow \{\text{Diaper}\}$ (s=0.4, c=1.0)
- $\{\text{Diaper}, \text{Beer}\} \rightarrow \{\text{Milk}\}$ (s=0.4, c=0.67)
- $\{\text{Beer}\} \rightarrow \{\text{Milk}, \text{Diaper}\}$ (s=0.4, c=0.67)
- $\{\text{Diaper}\} \rightarrow \{\text{Milk}, \text{Beer}\}$ (s=0.4, c=0.5)
- $\{\text{Milk}\} \rightarrow \{\text{Diaper}, \text{Beer}\}$ (s=0.4, c=0.5)

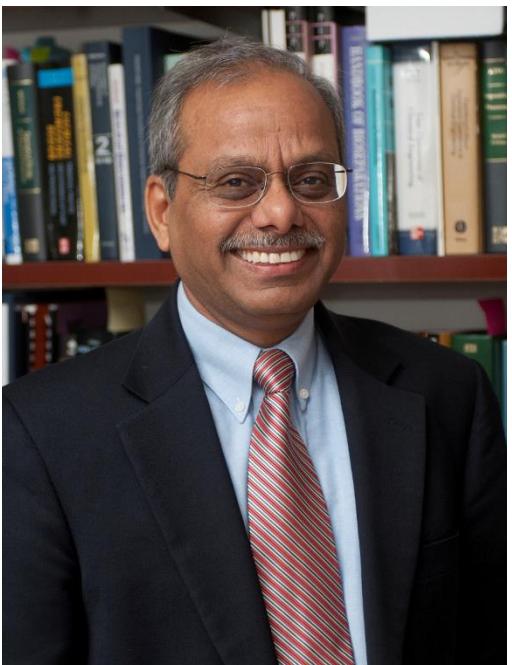


MINING ASSOCIATION RULES

- Two-step approach:
 1. **Frequent Itemset Generation**
 - Generate all itemsets whose support $\geq \text{minsup}$
 2. **Rule Generation**
 - Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset
- However frequent itemset generation appears to be a very **computationally expensive task!**



Solution is described in one of the most cited papers
in all of Computer Science....

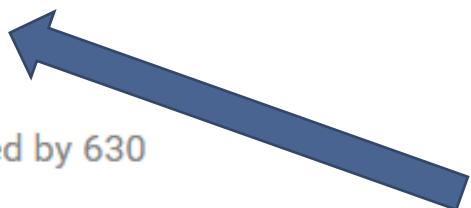


Scholarly articles for **fast algorithms for mining association rules**

[Fast algorithms for mining association rules](#) - Agrawal - Cited by 21740

[Fast algorithm for mining association rules](#) - Margahny - Cited by 58

[A fast distributed algorithm for mining association rules](#) - Cheung - Cited by 630



[PDF] [Fast Algorithms for Mining Association Rules](#) - Rakesh Agrawal

rakesh.agrawal-family.com/papers/vldb94apriori.pdf ▾

by R Agrawal - Cited by 21740 - Related articles

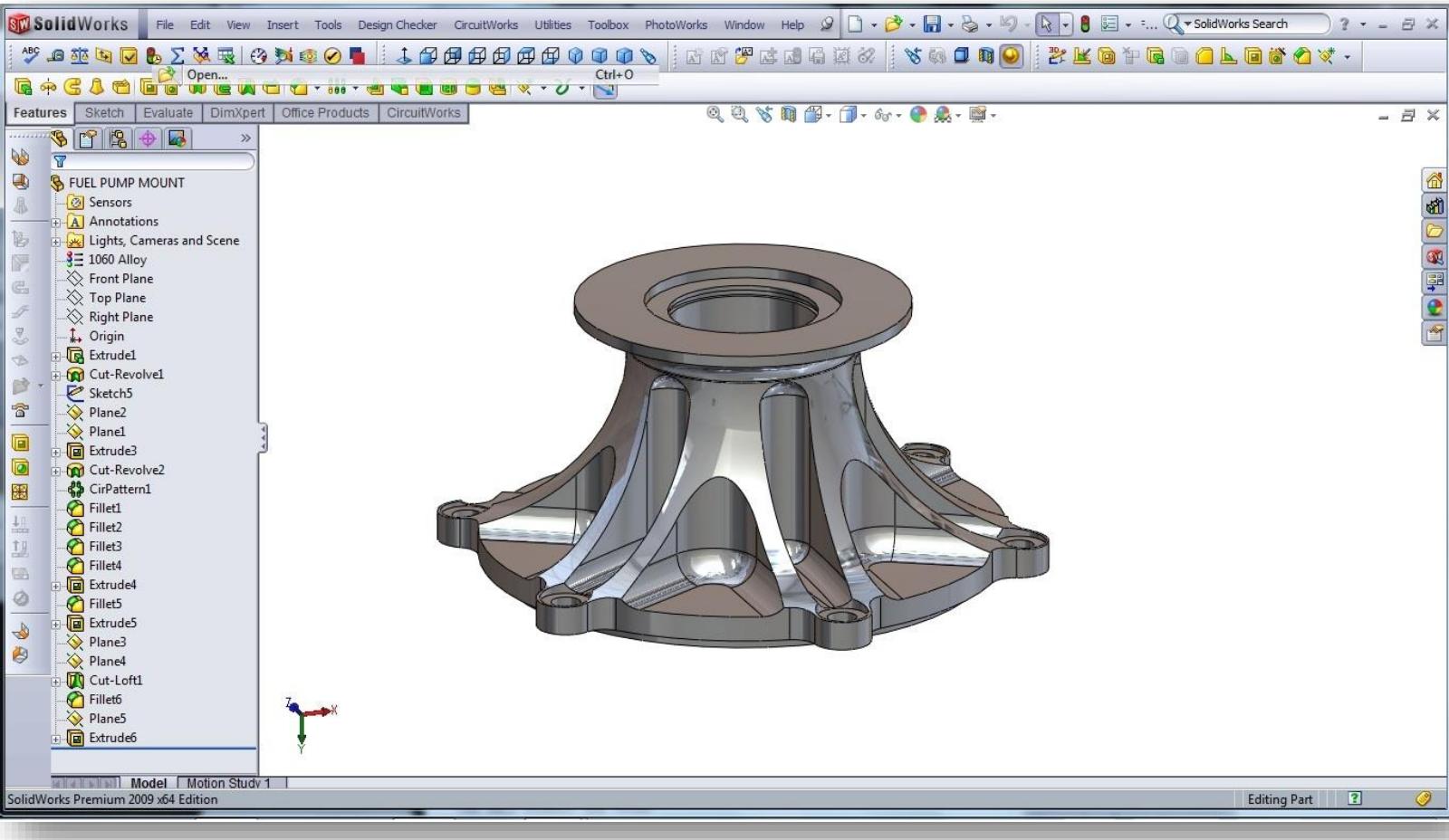
Fast Algorithms for Mining Association Rules. Rakesh Agrawal. Ramakrishnan Srikant. IBM Almaden Research Center. 650 Harry Road, San Jose, CA 95120.

.....It's a clever tree search

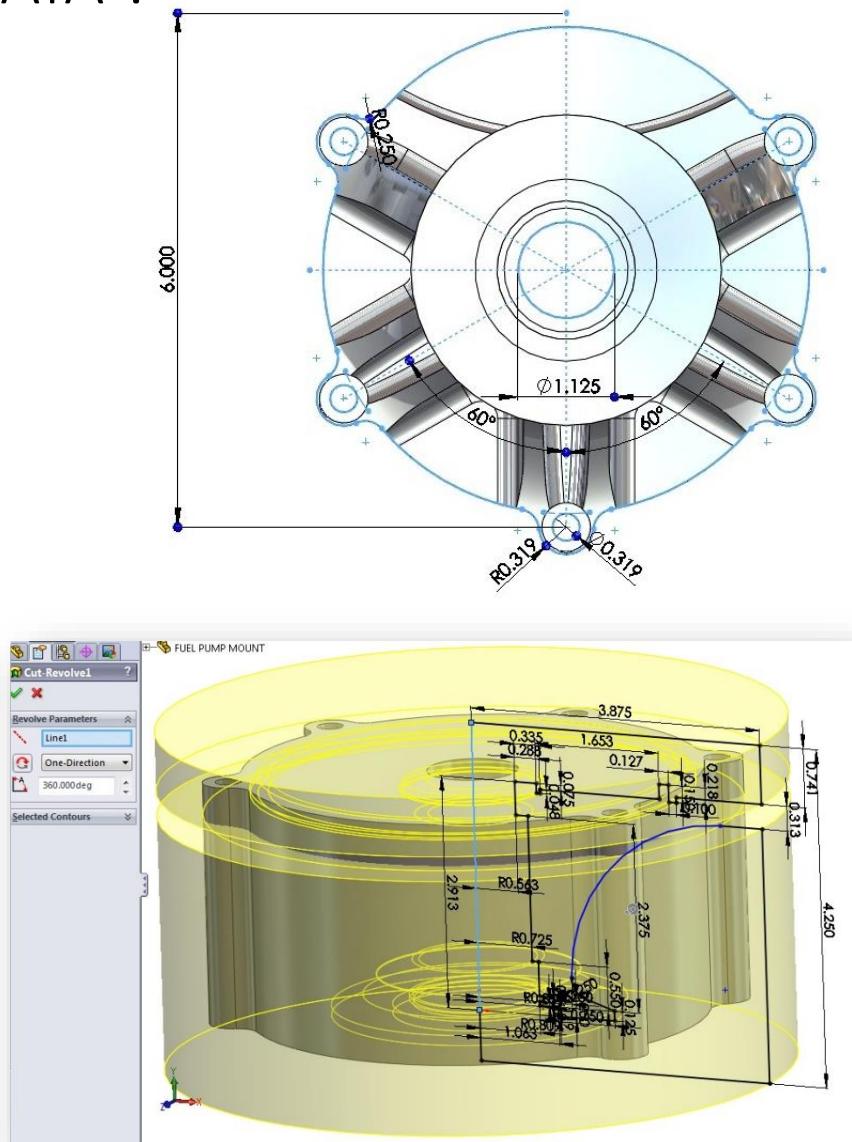
EXAMPLE APPLICATION: ASSOCIATION RULE MINING OF ENGINEERING DESIGNS



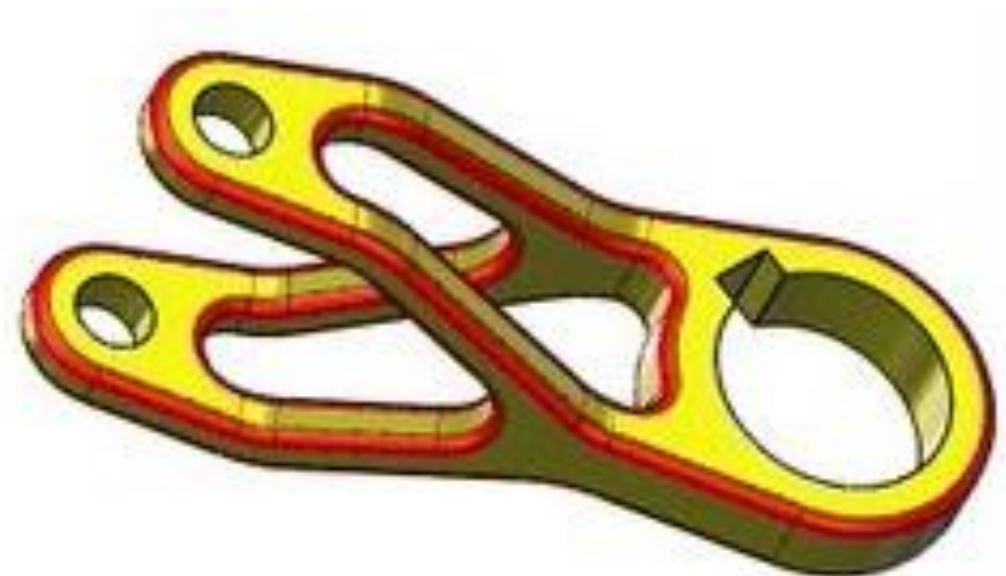
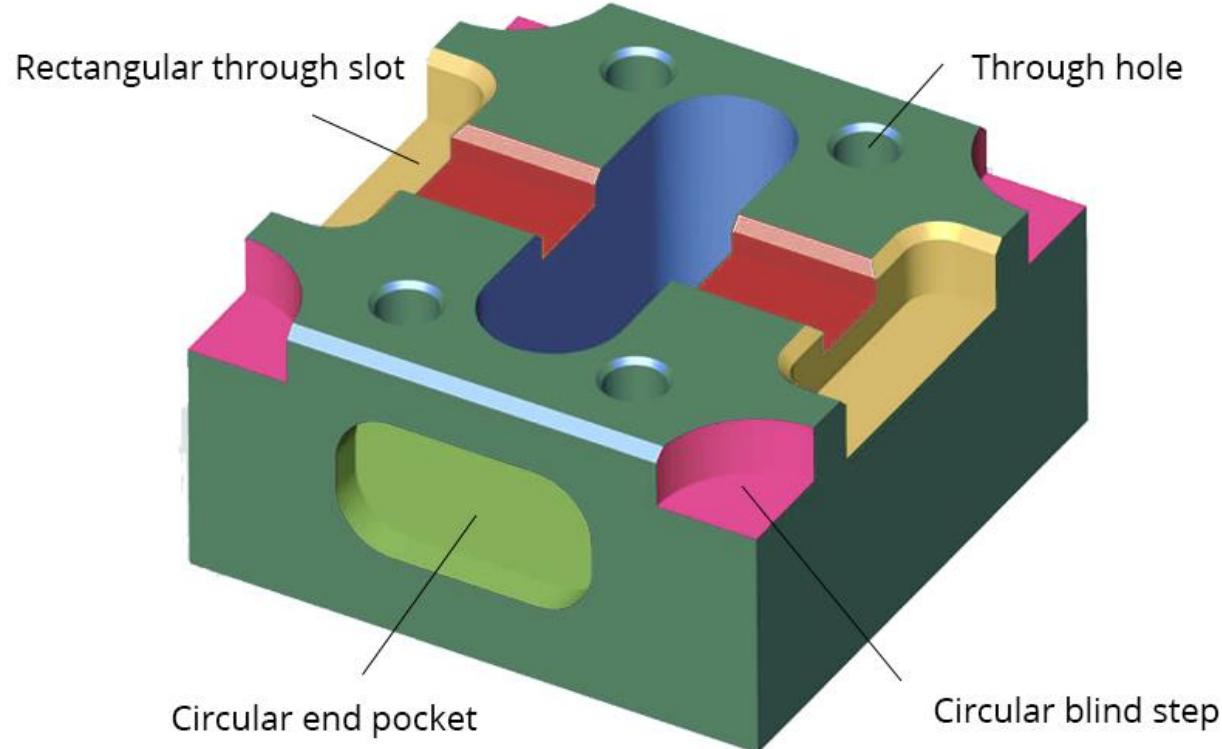
CAD DATA DEFINES THE DESIGNS BUT WHERE IS THE DATA ?



We can see it, most of the interpretation is done by the user not the computer



FOR SIMPLE SHAPES SO CALLED FEATURE INFORMATION CAN BE EXTRACTED, BUT FOR COMPLEX MODELS THE DEFINITIONS BREAKDOWN



How many holes?



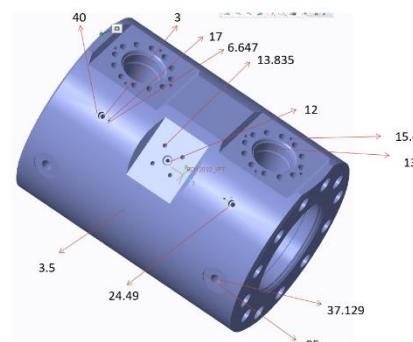
THE UNIVERSITY of EDINBURGH
School of Engineering

SPECTRUM OF INFORMATION EXTRACTABLE FROM CAD DATA

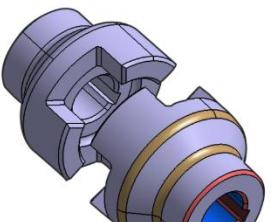


Mass Properties

Overall dimensions



Holes

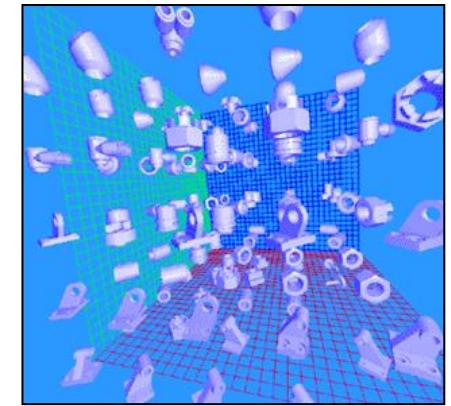


GTIN# 123 4567 890 234

Duplicates
(Same shape different part number)



GTIN# 091 2345 678 9012



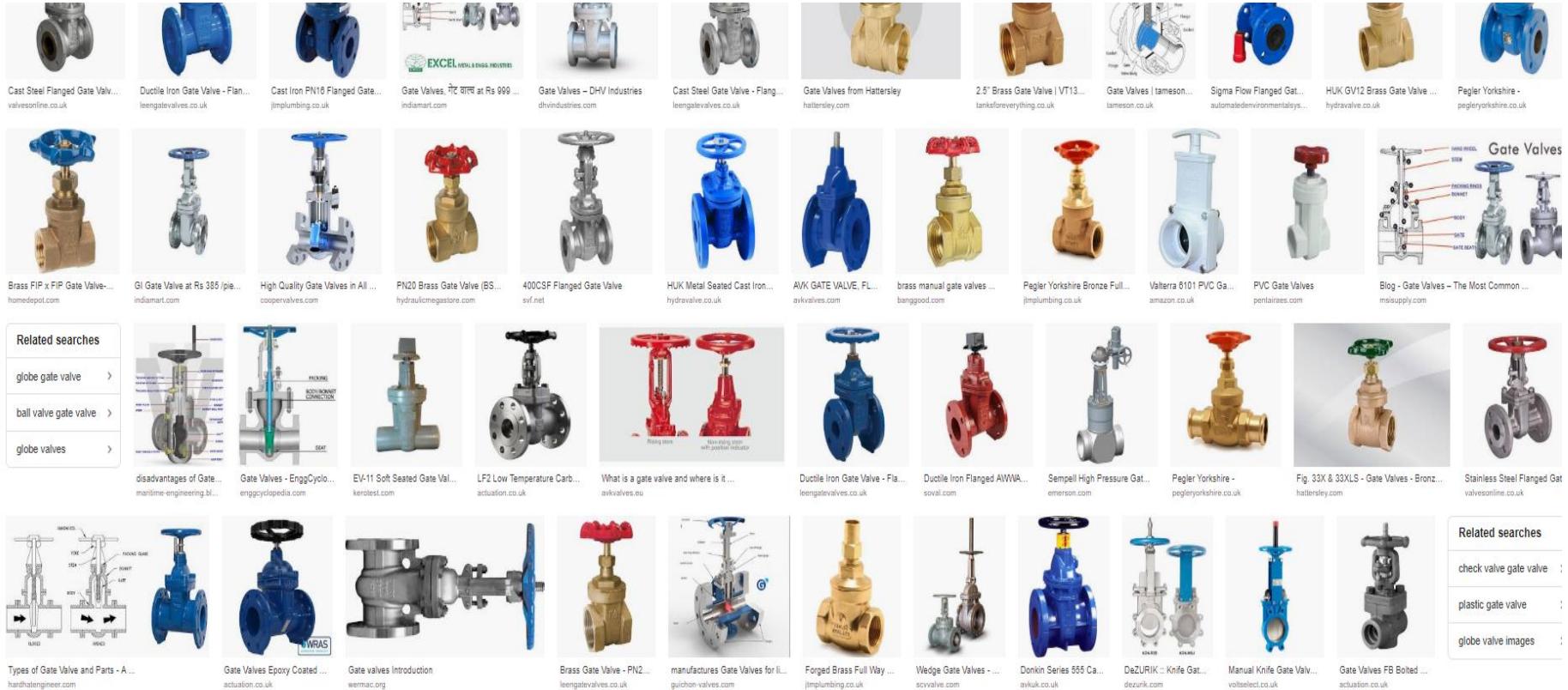
Partial Similarity

Easy

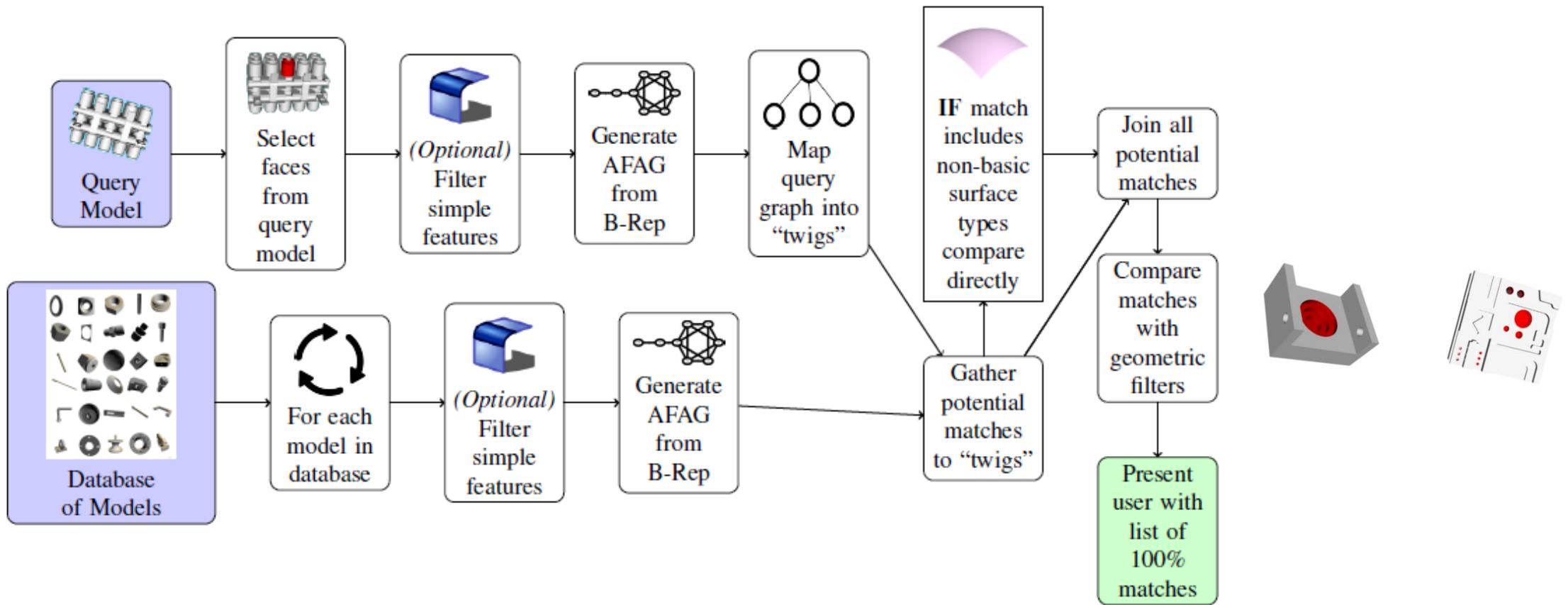
Hard

Impossible

3D CAD DATA FOR ONE COMPANIES 16,000 VALVES



High speed feature recognition able to process dataset of models to extract the holes



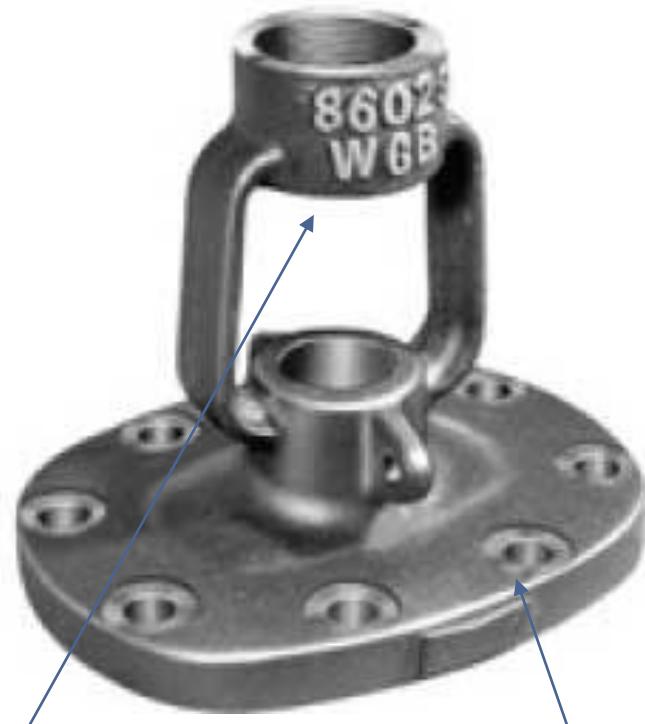
NUMBER AND DIAMETERS OF HOLES EXTRACTED FROM EACH 3D CAD MODEL

Body



8502-3FF: 10,10,10,10,8,8,8,8, 12,12,12,12, 36, 45

Bonnet



8602WGB: 12,12,12,12,12,12,12,12, 34,38, 35

Part number: List of hole diameters

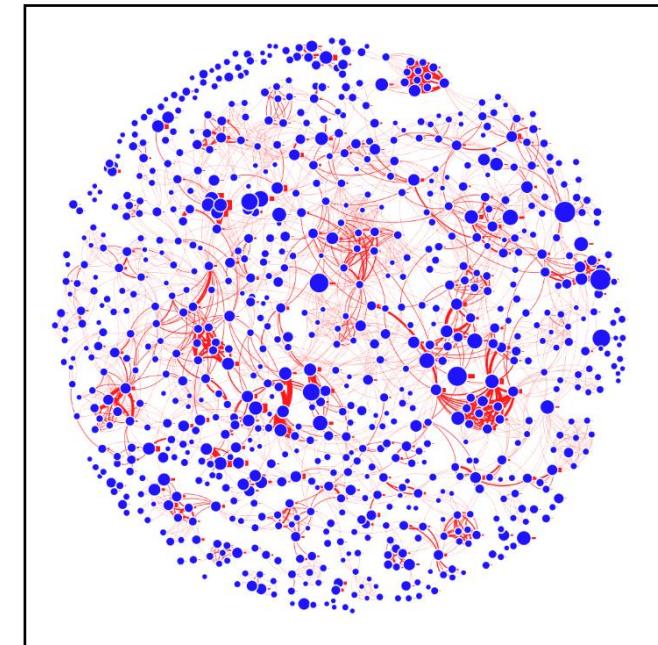
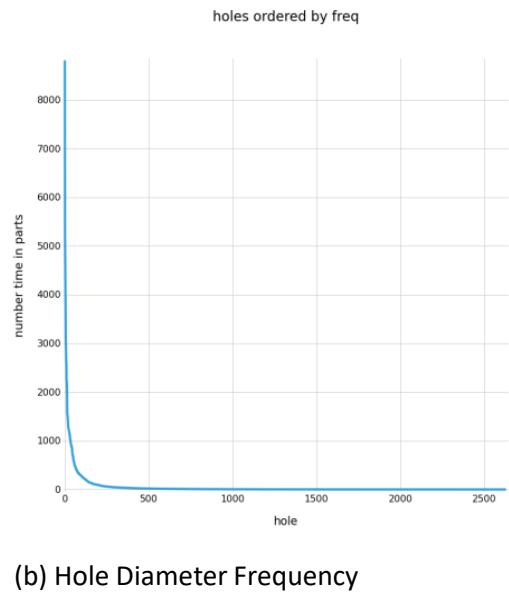
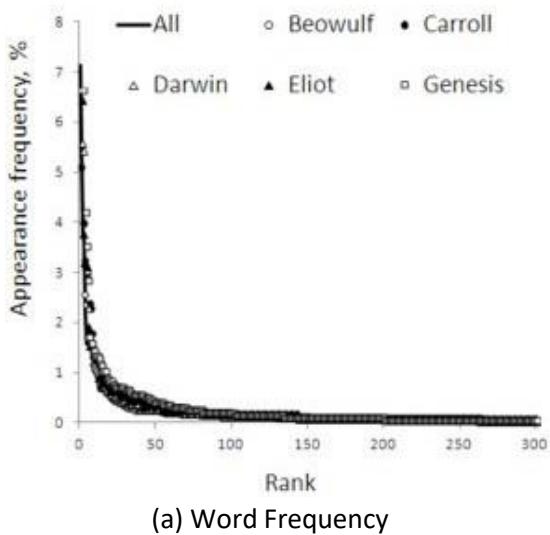


Hole Dataset

Component holes

[57.15, 12.0, 5.2, 46.375, 31.8, 3.5, 28.05, 50.0, 29.975, 84.07, 39.0, 12.85, 19.1, 63.375, 45.875, 22.025, 18.5, 17.175, 18.25, 24.915, 38.025, 54.86, 19.0, 63.88, 42.39, 26.75, 3.0, 6.647, 12.751, 15.05, 3]
[1.6, 10.106, 16.307, 34.0, 13.835, 30.0, 20.7, 6.25, 3.0, 44.1, 42.2, 42.375, 45.0, 34.95, 41.7, 32.64, 45.5, 4.917, 44.4, 5.2, 3.5, 25.4, 27.15, 18.0, 58.0, 19.05, 12.751, 24.0, 32.0, 23.0, 37.0]
[20.7, 33.5, 19.065, 6.25, 5.2, 13.835, 10.592, 31.75, 101.24, 18.075, 48.625, 11.1, 44.1, 3.0, 6.35, 23.0, 40.5, 10.0, 40.0, 30.0, 3.2, 32.64, 50.4, 37.0, 12.751, 18.02, 3.5, 34.95, 58.0, 4.917]
[60.0, 12.751, 19.0, 46.8, 3.242, 29.975, 18.0, 19.177, 25.0, 3.0, 5.2, 3.5, 35.0, 24.49, 11.1, 16.307, 6.647, 10.592, 13.835, 43.78, 19.1, 51.0, 31.8, 36.0426, 38.8, 34.0, 18.5, 20.0, 15.0]
[38.8, 20.0, 24.49, 5.2, 43.78, 12.751, 18.5, 35.0, 34.0, 3.5, 51.0, 11.1, 36.0426, 19.0, 31.8, 15.0, 29.975, 10.592, 60.0, 25.0, 6.647, 46.8, 18.0, 19.1, 13.835, 3.0, 3.242, 16.307]
[15.0, 14.0, 24.92, 23.8124, 20.0, 3.5, 28.05, 22.225, 4.134, 9.0, 34.25, 12.0, 3.0, 11.835, 6.647, 35.975, 27.625, 15.125, 45.7962, 26.0, 27.0, 24.69, 18.05, 45.2424, 22.0, 31.17, 12.85]
[46.6, 11.0, 10.592, 55.0, 3.242, 4.134, 28.05, 98.0, 19.177, 30.0, 15.125, 16.307, 3.5, 3.0, 69.08, 24.915, 12.751, 12.0, 45.0, 6.647, 18.1, 5.2, 32.0, 50.8, 8.4, 46.0, 12.85]
[27.15, 32.0, 25.5, 44.1, 3.5, 1.6, 13.835, 58.0, 30.0, 41.1, 8.376, 5.0, 13.6, 34.0, 20.7, 3.0, 34.95, 17.75, 42.375, 4.917, 46.0374, 32.639, 44.4, 37.0, 32.64, 28.0, 6.25]
[13.835, 58.0, 41.1, 13.6, 20.7, 17.75, 28.0, 42.375, 34.0, 44.1, 34.95, 32.0, 5.0, 32.639, 32.64, 3.0, 6.25, 1.6, 37.0, 3.5, 46.0374, 44.4, 8.376, 30.0, 4.917, 25.5, 27.15]
[16.307, 44.4, 1.6, 26.92, 12.751, 23.95, 58.0, 32.0, 27.2, 25.4, 35.9, 5.2, 3.0, 27.4, 20.7, 4.917, 29.975, 6.25, 13.835, 44.1, 39.1, 34.0, 42.375, 7.798, 3.5, 4.0, 23.0]
[28.0, 6.25, 1.6, 34.0, 32.0, 5.0, 44.1, 42.375, 13.6, 27.15, 5.2, 37.0, 34.95, 3.5, 30.0, 13.835, 12.751, 4.917, 41.4, 30.353, 46.038, 58.0, 3.0, 44.4, 32.64, 25.5, 20.7]
[37.0, 5.0, 27.15, 32.0, 53.025, 32.64, 13.6, 58.0, 3.5, 6.25, 30.0, 4.917, 13.835, 34.0, 42.0, 20.7, 34.95, 17.5, 44.1, 12.751, 5.2, 25.5, 44.4, 42.375, 3.0, 1.6, 18.25]
[18.25, 52.4, 46.2, 21.971, 6.0, 5.0, 60.0, 20.0, 49.5, 55.0, 35.0, 3.5, 33.5, 38.1, 329.2, 17.45, 48.0, 13.6, 53.0, 14.0, 200.0, 23.114, 29.4, 25.4, 26.05, 43.25, 13.835]
[3.0, 27.15, 20.7, 37.0, 30.0, 34.0, 3.5, 53.025, 32.0, 1.6, 28.0, 58.0, 44.1, 5.0, 4.917, 5.2, 25.5, 6.25, 42.375, 12.751, 44.4, 32.64, 34.95, 13.835, 13.6, 60.0, 30.353]
[34.95, 23.0, 37.0, 34.0, 6.0, 3.5, 13.835, 4.917, 42.375, 3.0, 44.1, 13.6, 32.0, 44.4, 30.0, 32.64, 10.592, 1.6, 16.307, 27.15, 25.5, 58.0, 20.7, 18.923, 2.0, 6.25, 5.0]
[25.9, 30.39, 32.75, 5.2, 4.134, 27.0, 12.5, 3.5, 11.7, 24.15, 12.7, 12.751, 25.1, 11.1, 19.05, 23.0, 30.0, 24.49, 22.23, 18.15, 3.0, 13.835, 25.15, 20.27, 10.106, 41.4]
[3.0, 41.4, 23.0, 30.0, 20.27, 25.15, 27.0, 32.75, 5.2, 24.49, 12.5, 13.835, 3.5, 30.39, 12.7, 22.23, 4.134, 19.05, 11.7, 11.1, 18.15, 25.9, 24.15, 25.1, 12.751, 10.106]
[13.5, 48.0, 5.2, 9.0, 4.917, 0.8, 12.751, 10.05, 27.65, 3.0, 10.23, 115.0, 27.025, 17.323, 13.6, 17.3, 7.798, 40.0, 25.0, 8.0, 6.35, 20.0, 10.0, 8.376, 3.242, 29.0]
[6.25, 1.6, 10.592, 30.0, 2.0, 6.0, 18.923, 25.5, 34.0, 5.0, 20.7, 37.0, 13.835, 3.0, 42.375, 44.4, 58.0, 19.177, 3.5, 32.0, 32.64, 34.95, 44.1, 27.15, 4.917, 13.6]
[1.6, 27.2, 29.975, 16.307, 23.0, 3.0, 27.15, 3.242, 44.4, 35.814, 26.92, 3.5, 32.0, 13.835, 23.95, 39.1, 12.751, 10.106, 42.375, 7.798, 25.4, 5.2, 20.7, 34.0, 58.0, 4.134]
[19.05, 13.835, 11.1, 25.9, 10.106, 12.751, 24.15, 3.0, 25.1, 22.23, 32.75, 18.15, 25.15, 30.39, 27.0, 24.49, 41.4, 30.0, 12.5, 4.134, 11.7, 3.5, 20.27, 23.0, 12.7, 5.2]
[27.0, 23.0, 4.134, 10.106, 30.39, 11.1, 12.7, 24.15, 19.05, 13.835, 12.751, 25.9, 3.5, 20.27, 22.23, 18.15, 12.5, 41.4, 30.0, 32.75, 3.0, 25.1, 25.15, 24.49, 11.7, 5.2]
[122.0, 50.0, 40.15, 6.647, 13.835, 35.0, 18.1, 105.15, 8.376, 84.0, 12.0, 15.125, 10.592, 28.05, 114.05, 45.15, 72.08, 43.235, 44.15, 3.5, 98.0, 24.915, 70.0, 18.25, 12.85]
[10.0, 13.5, 26.0, 25.0, 3.242, 20.0, 27.025, 12.1, 3.0, 13.6, 48.0, 5.2, 27.65, 18.923, 8.376, 4.134, 115.0, 7.798, 12.751, 8.0, 40.0, 6.35, 29.0, 9.0, 17.35]
[34.95, 1.6, 3.5, 13.835, 18.0, 32.0, 3.0, 53.025, 18.25, 32.64, 5.2, 30.0, 44.1, 20.7, 44.4, 4.917, 25.5, 42.0, 37.0, 27.15, 34.0, 6.25, 58.0, 12.751, 42.375]
[3.242, 30.0, 39.1, 36.05, 26.175, 53.025, 12.751, 4.134, 3.5, 20.7, 58.0, 42.375, 11.1, 29.975, 44.4, 25.4, 4.0, 10.592, 42.5, 31.0, 5.2, 24.49, 16.307, 3.0, 34.0]
[26.92, 35.814, 3.242, 42.375, 27.15, 23.0, 5.2, 25.4, 23.95, 20.7, 16.307, 32.0, 34.0, 12.751, 3.0, 4.134, 13.835, 39.1, 44.4, 29.975, 7.798, 27.2, 1.6, 3.5, 58.0]

THERE ARE PATTERNS IN THE HOLE DATA



Each node in the graph represents a specific diameter of hole: The size of the nodes is proportional to the frequency of the hole size within the CAD database. The thickness of the red arcs between nodes is proportional to the frequency with which the holes occur on the same component.

Component holes
[57.15, 12.0, 5.2, 46.375, 31.8, 3.5, 28.05, 50.0, 29.975, 84.07, 39.0, 12.85, 19.1, 63.375, 45.875, 22.025, 18.5, 17.175, 18.25, 24.915, 38.025, 54.86, 19.0, 63.88, 42.39, 26.75, 3.0, 6.647, 12.751, 15.05, 3.0]
[1.6, 10.106, 16.307, 34.0, 13.835, 30.0, 20.7, 6.25, 3.0, 44.1, 42.2, 42.375, 45.0, 34.95, 41.7, 32.64, 45.5, 4.917, 44.4, 5.2, 3.5, 25.4, 27.15, 18.0, 58.0, 19.05, 12.751, 24.0, 32.0, 23.0, 37.0]
[20.7, 33.5, 19.065, 6.25, 5.2, 13.835, 10.592, 31.75, 101.24, 18.075, 48.625, 11.1, 44.1, 3.0, 6.35, 23.0, 40.5, 10.0, 40.0, 30.0, 3.2, 32.64, 50.4, 37.0, 12.751, 18.02, 3.5, 34.95, 58.0, 4.917]
[60.0, 12.751, 19.0, 46.8, 3.242, 29.975, 18.0, 19.177, 25.0, 3.0, 5.2, 3.5, 35.0, 24.49, 11.1, 16.307, 6.647, 10.592, 13.835, 43.78, 19.1, 51.0, 31.8, 36.0426, 38.8, 34.0, 18.5, 20.0, 15.0]
[38.8, 20.0, 24.49, 5.2, 43.78, 12.751, 18.5, 35.0, 34.0, 3.5, 51.0, 11.1, 36.0426, 19.0, 31.8, 15.0, 29.975, 10.592, 60.0, 25.0, 6.647, 46.8, 18.0, 19.1, 13.835, 3.0, 3.242, 16.307]
[15.0, 14.0, 24.92, 23.8124, 20.0, 3.5, 28.05, 22.225, 4.134, 9.0, 34.25, 12.0, 3.0, 11.835, 6.647, 35.975, 27.625, 15.125, 45.7962, 26.0, 27.0, 24.69, 18.05, 45.2424, 22.0, 31.17, 12.85]
[46.6, 11.0, 10.592, 55.0, 3.242, 4.134, 28.05, 98.0, 19.177, 30.0, 15.125, 16.307, 3.5, 3.0, 69.08, 24.915, 12.751, 12.0, 45.0, 6.647, 18.1, 5.2, 32.0, 50.8, 8.4, 46.0, 12.85]
[27.15, 32.0, 25.5, 44.1, 3.5, 1.6, 13.835, 58.0, 30.0, 41.1, 8.376, 5.0, 13.6, 34.0, 20.7, 3.0, 34.95, 17.75, 42.375, 4.917, 46.0374, 32.639, 44.4, 37.0, 32.64, 28.0, 6.25]
[13.835, 58.0, 41.1, 13.6, 20.7, 17.75, 28.0, 42.375, 34.0, 44.1, 34.95, 32.0, 5.0, 32.639, 32.64, 3.0, 6.25, 1.6, 37.0, 3.5, 46.0374, 44.4, 8.376, 30.0, 4.917, 25.5, 27.15]
[16.307, 44.4, 1.6, 26.92, 12.751, 23.95, 58.0, 32.0, 27.2, 25.4, 35.9, 5.2, 3.0, 27.4, 20.7, 4.917, 29.975, 6.25, 13.835, 44.1, 39.1, 34.0, 42.375, 7.798, 3.5, 4.0, 23.0]
[28.0, 6.25, 1.6, 34.0, 32.0, 5.0, 44.1, 42.375, 13.6, 27.15, 5.2, 37.0, 34.95, 3.5, 30.0, 13.835, 12.751, 4.917, 41.4, 30.353, 46.038, 58.0, 3.0, 44.4, 32.64, 25.5, 20.7]
[37.0, 5.0, 27.15, 32.0, 53.025, 32.64, 13.6, 58.0, 3.5, 6.25, 30.0, 4.917, 13.835, 34.0, 42.0, 20.7, 34.95, 17.5, 44.1, 12.751, 5.2, 25.5, 44.4, 42.375, 3.0, 1.6, 18.25]
[18.25, 52.4, 46.2, 21.971, 6.0, 5.0, 60.0, 20.0, 49.5, 55.0, 35.0, 3.5, 33.5, 38.1, 329.2, 17.45, 48.0, 13.6, 53.0, 14.0, 200.0, 23.114, 29.4, 25.4, 26.05, 43.25, 13.835]
[3.0, 27.15, 20.7, 37.0, 30.0, 34.0, 3.5, 53.025, 32.0, 1.6, 28.0, 58.0, 44.1, 5.0, 4.917, 5.2, 25.5, 6.25, 42.375, 12.751, 44.4, 32.64, 34.95, 13.835, 13.6, 60.0, 30.353]
[34.95, 23.0, 37.0, 34.0, 6.0, 3.5, 13.835, 4.917, 42.375, 3.0, 44.1, 13.6, 32.0, 44.4, 30.0, 32.64, 10.592, 1.6, 16.307, 27.15, 25.5, 58.0, 20.7, 18.923, 2.0, 6.25, 5.0]
[25.9, 30.39, 32.75, 5.2, 4.134, 27.0, 12.5, 3.5, 11.7, 24.15, 12.7, 12.751, 25.1, 11.1, 19.05, 23.0, 30.0, 24.49, 22.23, 18.15, 3.0, 13.835, 25.15, 20.27, 10.106, 41.4]
[3.0, 41.4, 23.0, 30.0, 20.27, 25.15, 27.0, 32.75, 5.2, 24.49, 12.5, 13.835, 3.5, 30.39, 12.7, 22.23, 4.134, 19.05, 11.7, 11.1, 18.15, 25.9, 24.15, 25.1, 12.751, 10.106]
[13.5, 48.0, 5.2, 9.0, 4.917, 0.8, 12.751, 10.05, 27.65, 3.0, 10.23, 115.0, 27.025, 17.323, 13.6, 17.3, 7.798, 40.0, 25.0, 8.0, 6.35, 20.0, 10.0, 8.376, 3.242, 29.0]
[6.25, 1.6, 10.592, 30.0, 2.0, 6.0, 18.923, 25.5, 34.0, 5.0, 20.7, 37.0, 13.835, 3.0, 42.375, 44.4, 58.0, 19.177, 3.5, 32.0, 32.64, 34.95, 44.1, 27.15, 4.917, 13.6]
[1.6, 27.2, 29.975, 16.307, 23.0, 3.0, 27.15, 3.242, 44.4, 35.814, 26.92, 3.5, 32.0, 13.835, 23.95, 39.1, 12.751, 10.106, 42.375, 7.798, 25.4, 5.2, 20.7, 34.0, 58.0, 4.134]
[19.05, 13.835, 11.1, 25.9, 10.106, 12.751, 24.15, 3.0, 25.1, 22.23, 32.75, 18.15, 25.15, 30.39, 27.0, 24.49, 41.4, 30.0, 12.5, 4.134, 11.7, 3.5, 20.27, 23.0, 12.7, 5.2]
[27.0, 23.0, 4.134, 10.106, 30.39, 11.1, 12.7, 24.15, 19.05, 13.835, 12.751, 25.9, 3.5, 20.27, 22.23, 18.15, 12.5, 41.4, 30.0, 32.75, 3.0, 25.1, 25.15, 24.49, 11.7, 5.2]
[122.0, 50.0, 40.15, 6.647, 13.835, 35.0, 18.1, 105.15, 8.376, 84.0, 12.0, 15.125, 10.592, 28.05, 114.05, 45.15, 72.08, 43.235, 44.15, 3.5, 98.0, 24.915, 70.0, 18.25, 12.85]
[10.0, 13.5, 26.0, 25.0, 3.242, 20.0, 27.025, 12.1, 3.0, 13.6, 48.0, 5.2, 27.65, 18.923, 8.376, 4.134, 115.0, 7.798, 12.751, 8.0, 40.0, 6.35, 29.0, 9.0, 17.35]
[34.95, 1.6, 3.5, 13.835, 18.0, 32.0, 3.0, 53.025, 18.25, 32.64, 5.2, 30.0, 44.1, 20.7, 44.4, 4.917, 25.5, 42.0, 37.0, 27.15, 34.0, 6.25, 58.0, 12.751, 42.375]
[3.242, 30.0, 39.1, 36.05, 26.175, 53.025, 12.751, 4.134, 3.5, 20.7, 58.0, 42.375, 11.1, 29.975, 44.4, 25.4, 4.0, 10.592, 42.5, 31.0, 5.2, 24.49, 16.307, 3.0, 34.0]
[26.92, 35.814, 3.242, 42.375, 27.15, 23.0, 5.2, 25.4, 23.95, 20.7, 16.307, 32.0, 34.0, 12.751, 3.0, 4.134, 13.835, 39.1, 44.4, 29.975, 7.798, 27.2, 1.6, 3.5, 58.0]

Perhaps the collection of holes on each component could be analogous to items in shopping basket



Example of Association Rules

$\{\text{Diaper}\} \rightarrow \{\text{Beer}\}$,
 $\{\text{Milk}, \text{Bread}\} \rightarrow \{\text{Eggs}, \text{Coke}\}$,
 $\{\text{Beer}, \text{Bread}\} \rightarrow \{\text{Milk}\}$,

Implication means co-occurrence, not causality!

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Itemsets



List of Hole
Diameters on
each
component

Component holes						
[57.15, 12.0, 5.2, 46.375, 31.8, 3.5, 28.05, 50.0, 29.975, 84.07, 39.0, 12.85, 19.1, 63.375, 45.875, 22.025, 18.5, 17.175, 18.25, 24.915, 38.025, 54.86, 19.0, 63.88, 42.39, 26.75, 3.0, 6.647, 12.751, 15.05, 1.6, 10.106, 16.307, 34.0, 13.835, 30.0, 20.7, 6.25, 3.0, 44.1, 42.2, 42.375, 45.0, 34.95, 41.7, 32.64, 45.5, 4.917, 44.4, 5.2, 3.5, 25.4, 27.15, 18.0, 58.0, 19.05, 12.751, 24.0, 32.0, 23.0, 37.0]						
[20.7, 33.5, 19.065, 6.25, 5.2, 13.835, 10.592, 31.75, 101.24, 18.075, 48.625, 11.1, 44.1, 3.0, 6.35, 23.0, 40.5, 10.0, 40.0, 30.0, 3.2, 32.64, 50.4, 37.0, 12.751, 18.02, 3.5, 34.95, 58.0, 4.917]						
[60.0, 12.751, 19.0, 46.8, 3.242, 29.975, 18.0, 19.177, 25.0, 3.0, 5.2, 3.5, 35.0, 24.49, 11.1, 16.307, 6.647, 10.592, 13.835, 43.78, 19.1, 51.0, 31.8, 36.0426, 38.8, 34.0, 18.5, 20.0, 15.0]						
[38.8, 20.0, 24.49, 5.2, 43.78, 12.751, 18.5, 35.0, 34.0, 3.5, 51.0, 11.1, 36.0426, 19.0, 31.8, 15.0, 29.975, 10.592, 60.0, 25.0, 6.647, 46.8, 18.0, 19.1, 13.835, 3.0, 3.242, 16.307]						
[15.0, 14.0, 24.92, 23.8124, 20.0, 3.5, 28.05, 22.225, 4.134, 9.0, 34.25, 12.0, 3.0, 11.835, 6.647, 35.975, 27.625, 15.125, 45.7962, 26.0, 27.0, 24.69, 18.05, 45.2424, 22.0, 31.17, 12.85]						
[46.6, 11.0, 10.592, 55.0, 3.242, 4.134, 28.05, 98.0, 19.177, 30.0, 15.125, 16.307, 3.5, 3.0, 69.08, 24.915, 12.751, 12.0, 45.0, 6.647, 18.1, 5.2, 32.0, 50.8, 8.4, 46.0, 12.85]						
[27.15, 32.0, 25.5, 44.1, 3.5, 1.6, 13.835, 58.0, 30.0, 41.1, 8.376, 5.0, 13.6, 34.0, 20.7, 3.0, 34.95, 17.75, 42.375, 4.917, 46.0374, 32.639, 44.4, 37.0, 32.64, 28.0, 6.251]						
[13.835, 58.0, 41.1, 13.6, 20.7, 17.75, 28.0, 42.375, 34.0, 44.1, 34.95, 32.0, 5.0, 32.631, 16.307, 44.4, 1.6, 26.92, 12.751, 23.95, 58.0, 32.0, 27.2, 25.4, 35.9, 5.2, 3.0, 27.4, 20.8, 6.25, 1.6, 34.0, 32.0, 5.0, 44.1, 42.375, 13.6, 27.15, 5.2, 37.0, 34.95, 3.5, 30.0, 37.0, 5.0, 27.15, 32.0, 53.025, 32.64, 13.6, 58.0, 3.5, 6.25, 30.0, 4.917, 13.835, 34.0, 18.25, 52.4, 46.2, 21.971, 6.0, 5.0, 60.0, 20.0, 49.5, 55.0, 35.0, 3.5, 33.5, 38.1, 329.2, 3.0, 27.15, 20.7, 37.0, 30.0, 34.0, 3.5, 53.025, 32.0, 1.6, 28.0, 58.0, 44.1, 5.0, 4.917, 34.95, 23.0, 37.0, 34.0, 6.0, 3.5, 13.835, 4.917, 42.375, 3.0, 44.1, 13.6, 32.0, 44.4, 30.9, 25.9, 30.39, 32.75, 5.2, 4.134, 27.0, 12.5, 3.5, 11.7, 24.15, 12.7, 12.751, 25.1, 11.1, 3.0, 41.4, 23.0, 30.0, 20.27, 25.15, 27.0, 32.75, 5.2, 24.49, 12.5, 13.835, 3.5, 30.39, 13.5, 48.0, 5.2, 9.0, 4.917, 0.8, 12.751, 10.05, 27.65, 3.0, 10.23, 115.0, 27.025, 17.32, 6.25, 1.6, 10.592, 30.0, 2.0, 6.0, 18.923, 25.5, 34.0, 5.0, 20.7, 37.0, 13.835, 3.0, 42.3, 1.6, 27.2, 29.975, 16.307, 23.0, 3.0, 27.15, 3.242, 44.4, 35.814, 26.92, 3.5, 32.0, 13.8, 19.05, 13.835, 11.1, 25.9, 10.106, 12.751, 24.15, 3.0, 25.1, 22.23, 32.75, 18.15, 25.1, 27.0, 23.0, 4.134, 10.106, 30.39, 11.1, 12.7, 24.15, 19.05, 13.835, 12.751, 25.9, 3.5, 122.0, 50.0, 40.15, 6.647, 13.835, 35.0, 18.1, 105.15, 8.376, 84.0, 12.0, 15.125, 10.592, 10.0, 13.5, 26.0, 25.0, 3.242, 20.0, 27.025, 12.1, 3.0, 13.6, 48.0, 5.2, 27.65, 18.923, 8.34.95, 1.6, 3.5, 13.835, 18.0, 32.0, 3.0, 53.025, 18.25, 32.64, 5.2, 30.0, 44.1, 20.7, 44.2, 3.242, 30.0, 39.1, 36.05, 26.175, 53.025, 12.751, 4.134, 3.5, 20.7, 58.0, 42.375, 11.1, 26.92, 35.814, 3.242, 42.375, 27.15, 23.0, 5.2, 25.4, 23.95, 20.7, 16.307, 32.0, 34.0, 11.1, 26.92, 35.814, 3.242, 42.375, 27.15, 23.0, 5.2, 25.4, 23.95, 20.7, 16.307, 32.0, 34.0, 11.1]	L3					

K = 3 Itemsets of hole
diameters

3	3.5	15.125	support	:	110
3	3.5	16.307	support	:	143
3	3.5	18.05	support	:	157
3	3.5	18.1	support	:	62
3	3.5	18.923	support	:	117
3	3.5	20	support	:	62
3	3.5	23	support	:	57
3	3.5	24.49	support	:	79
3	3.5	24.69	suppo	:	
3	3.5	24.915	suppo	:	
3	3.5	26.289	suppo	:	
3	3.5	28.05	suppo	:	

Frequency with which
the Itemsets occurs

ANALYSIS OF MANY DATA SET PRODUCES MANY DIFFERENT ITEM SETS

2.5	4.8	7.75	sup=1																						
2.5	5.675	6.8	sup=1																						
2.5	11.2	sup=1																							
2.5	32.5	42.45	42.5	44.7	sup=1																				
2.5	3	sup=19																							
2.5	3	3.242	sup=14																						
	2.5	3	3.242	4.917	sup=12																				
		2.5	3	3.242	4.917	5.2	sup=10																		
			2.5	3	3.242	4.917	5.2	6.35	7.798	sup=6															
				2.5	3	3.242	4.917	5.2	6.35	7.798	11	12.751	13	13.5	17.05	17.3228									
					2.5	3	3.242	4.917	5.2	6.35	7.798	9	11	12.751	13.5	13.6	17.3228								
						2.5	3	3.242	4.917	5.2	6	7.798	10.592	11	12.751	13	13.5	17.05	20						
						2.5	3	3.242	4.917	5.2	7.798	sup=3													
							2.5	3	3.242	4.917	5.2	7.798	11	12.751	13	13.5	13.6	17.3228	20						
							2.5	3	3.242	4.917	5.2	7.798	8	11	12.751	13	13.5	13.6	17.3228						
							2.5	3	3.242	4.917	5.1	5.2	sup=2												
								2.5	3	3.242	4.917	5.1	5.2	11.835	12.751	16.667	21	24.1	25.2	25.4	30.675				
									2.5	3	3.242	4.917	5.1	5.2	10.106	12.751	16.667	21	23.82	25.2	25.4	30.65			
									2.5	3	3.242	5.2	sup=2												
										2.5	3	3.242	5.2	10.106	12.751	12.86	15	16.667	16.8	18	21	24.8	25.4	26.475	
											2.5	3	3.242	5.2	6.35	7.798	9	11	12.751	13.6	14	17.3228	20	25	27.025
											2.5	3	5.2	sup=5											

PARTS WITH THE SAME ITEMSETS ARE SIMILAR

The screenshot shows a Jupyter Notebook interface running on a browser. The title bar indicates the notebook is titled "explore.itemsets.JRC" and the last checkpoint was 10 hours ago (unsaved changes). The interface includes a toolbar with various icons for file operations, cell execution, and help.

Contents:

- 1 Setup
 - 1.1 Adder functions
 - 1.2 Set up partbrowser
 - 1.3 Load graph/connect to graph
- 2 Queries
- 3 Itemsets
 - 3.1 Set up itemsets
 - 3.2 Calc itemsets
- 4 Search for similar

In []: itemsets_with_parts_rev_count_by_minsup
Last executed 2018-09-18 08:50:18 in 1.09s

4 Search for similar

```
In [ ]: def get_similar_for_part(part, is_for_parts, parts_for_is):
    #print(is_for_parts[part])
    part_is = is_for_parts[part]
    sim_parts = collections.Counter()
    for itemset in part_is:
        ps = parts_for_is[itemset]
        for p in ps:
            sim_parts[p] += 1
    return sim_parts
Last executed 2018-09-18 08:57:04 in 12ms
```

We have already calculated all the frequent itemsets of design features for this data set of around 10,000 components

```
In [ ]: itemsets_with_parts = itemsets_with_parts_by_minsup[display_minsup]
itemsets_with_parts_rev = itemsets_with_parts_rev_by_minsup[display_minsup]
Last executed 2018-09-18 08:57:05 in 5ms
```

```
In [ ]: itemsets_with_parts
Last executed 2018-09-18 08:56:13 in 913ms
```

```
In [ ]: #itemsets_with_parts_rev
Last executed 2018-09-18 08:56:14 in 13.7s
```

```
In [ ]: query_part = 'BD12027-AC-PW-MC'
#query_part = 'BT12817'
itemsets_with_parts_rev[query_part]
Last executed 2018-09-18 08:55:30 in 146ms
```

```
In [ ]: sims = get_similar_for_part(query_part, itemsets_with_parts_rev, itemsets_with_parts)
sims
Last executed 2018-09-18 08:55:31 in 75ms
```

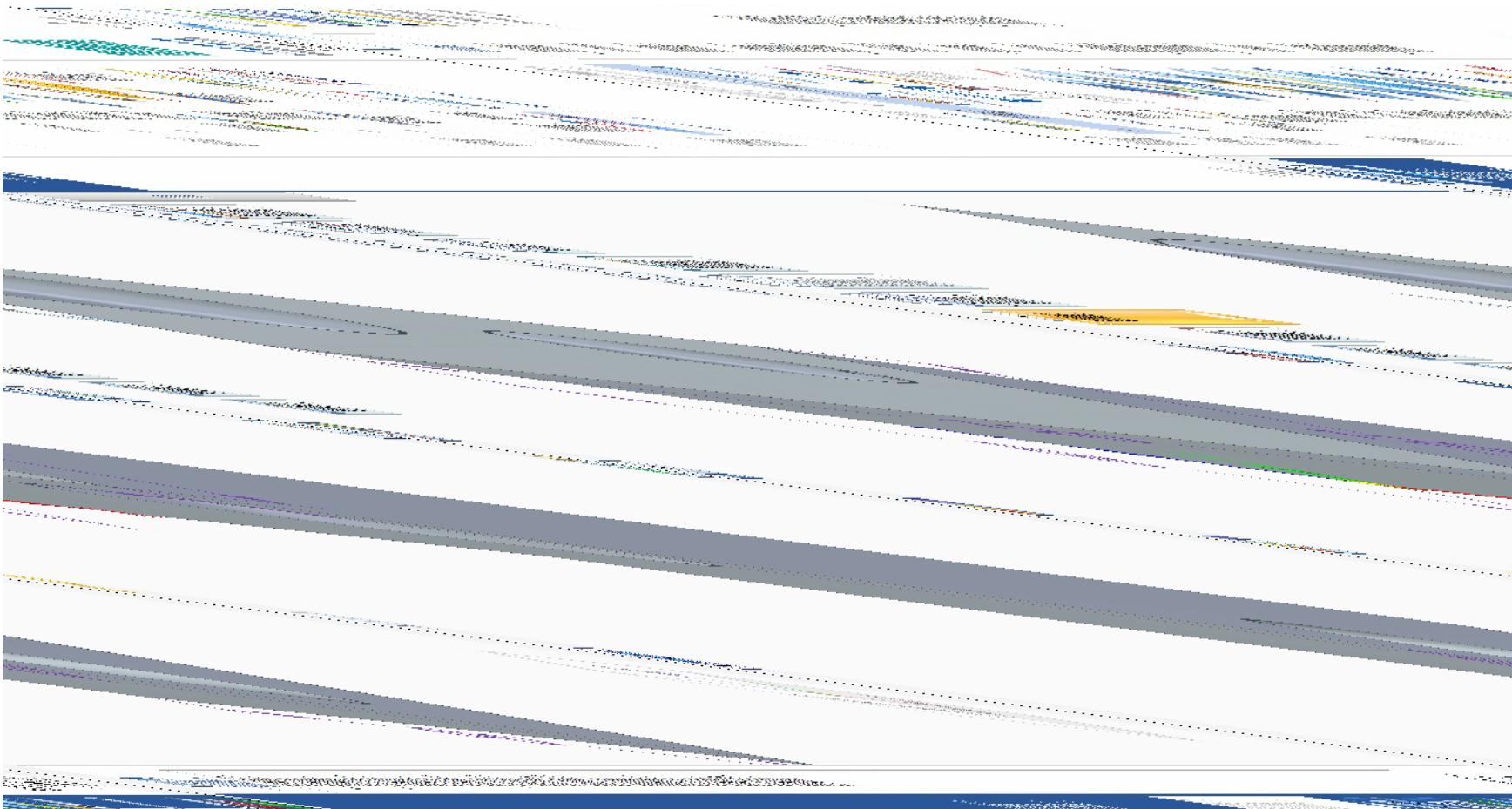
```
In [ ]: len(sims)
Last executed 2018-09-18 08:55:32 in 7ms
```

```
In [ ]: sims.most_common(200)
Last executed 2018-09-18 08:55:32 in 28ms
```

```
In [ ]: part_list = [part for (part, freq) in sims.most_common(1000)]
display = _massage_df(pd.DataFrame(part_list, columns=['code']), index=part_list), uri_from='code'
#display
```

The bottom of the screen shows the Windows taskbar with various pinned icons, and the system tray indicates the date and time as 18/09/2018 09:57.

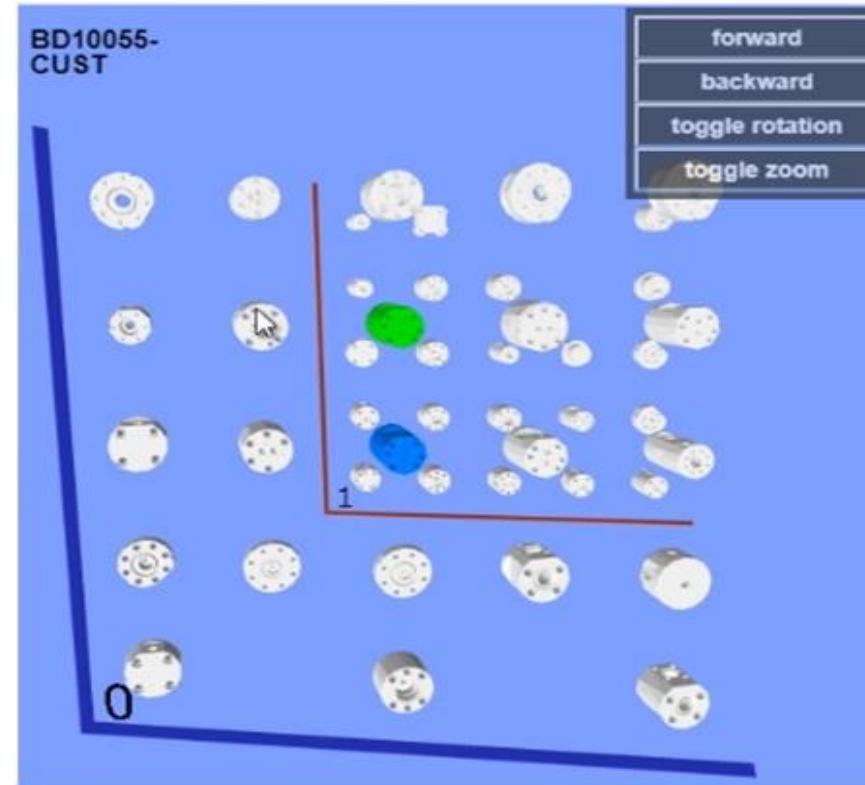
VERSION 0.1 OF OUR SOLIDEDGE “PREDICATIVE HOLE PLUG-IN”



THE UNIVERSITY of EDINBURGH
School of Engineering

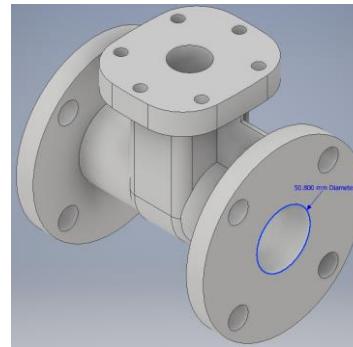
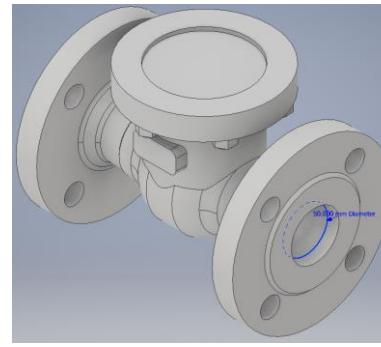
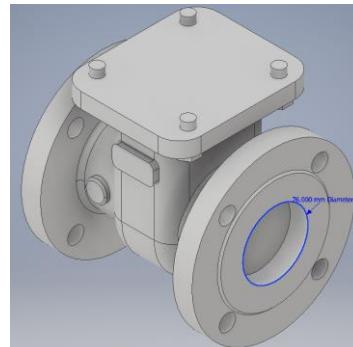
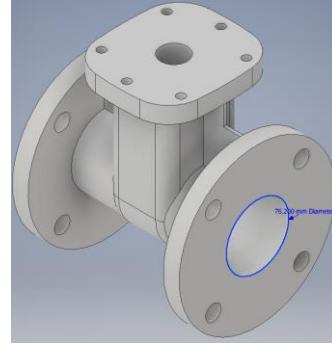
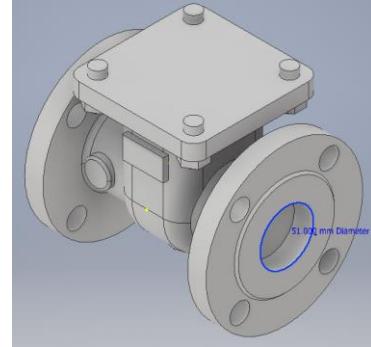
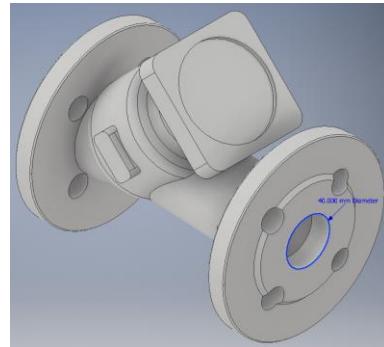
ONCE DATA IS EXTRACTED FROM THE CAD MODEL IT CAN ENABLE PORTFOLIO ANALYSES

- However each data extract process currently has to be bespoke dependent of the particular CAD systems and the CAD data



Similar Parts have Similar Itemsets
(i.e. Similar Patterns of holes)

ASSOCIATION RULES CAN ALSO BE USED TO IDENTIFY SUBSTITUTABLE FEATURES



- If the patterns of holes that occur frequently together have been identified then they can be used to suggest potential **substitutions**

AFTER THE BREAK WE WILL LOOK AT PYTHON NOTEBOOK EXAMPLE OF THE CLASSIC ITEM SET APPLICATION TO SHOPPING DATE

If could be represented as:

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

The associations rules could be:

- (Diaper, Beer) → Milk • Support = 2/5, Confidence = 2/3
- (Milk) → (Diaper, Beer) • Support = 2/5, Confidence = 2/4
- (Milk, Diaper) → Bread • Support = 2/5, Confidence = 2/3

where: "Support" defines how much historical data supports the rule and "Confidence" indicates probility that the rule holds.

```
In [2]: !pip install apyori --quiet #install the apyori library
In [3]: import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import io
```

Load the data from csv file describing the content of shopping baskets

Every row defines the contents of a different basket, every column has the name of a different item in the basket.

```
In [9]: df.set_option('display.max.columns', 5)
df2=pd.read_csv('groceries.csv')
In [10]: df2.head(4) #use the date, the first shopping basket contains citrus fruit, semi-finished bread, margarine ...etc
```

	Item 1	Item 2	Item 3	Item 30	Item 31	Item 32
0	citrus fruit, semi-finished bread	margarine	coffee	Nan	Nan	Nan
1	ice-cream	yogurt	Nan	Nan	Nan	Nan
2	whole milk	yogurt	cheese	Nan	Nan	Nan
3	peanut	cheese	Nan	Nan	Nan	Nan

4 rows x 32 columns

```
In [11]: df2.shape #the dataset has 9836 shopping baskets, containing a maximum of 32 items
Out[11]: (9836, 32)
```

Out[12]: a find item sets from apyori import apriori

```
In [13]: #fill an array of with the items in each row of the dataset
transactions = []
for i in range(0, 5000):
    transactions.append(str(df2.values[i,:])) for j in range(0, 20))
```

#just for illustration

```
print("transaction 1:", transactions[0][0], transactions[0][1], transactions[0][2], transactions[0][3])
print("transaction 2:", transactions[1][0], transactions[1][1], transactions[1][2], transactions[1][3])
```

Transaction 1: citrus fruit semi-finished bread margarine ready soups

Transaction 2: tropical fruit yogurt coffee nan



ML PREDICTION ALGORITHMS



MANY PREDICTION ALGORITHMS AVAILABLE ..THE PROBLEM IS ONE OF CHOICE

Type	Name	Description	Advantages	Disadvantages
Linear		Linear Regression The “best fit” line through all data points. Predictions are numerical.	Easy to understand — you clearly see what the biggest drivers of the model are.	Sometimes too simple to capture complex relationships between variables. Does poorly with correlated features.
		Logistic Regression The adaptation of linear regression to problems of classification (e.g., yes/no questions, groups, etc.)	Also easy to understand.	Sometimes too simple to capture complex relationships between variables. Does poorly with correlated features.

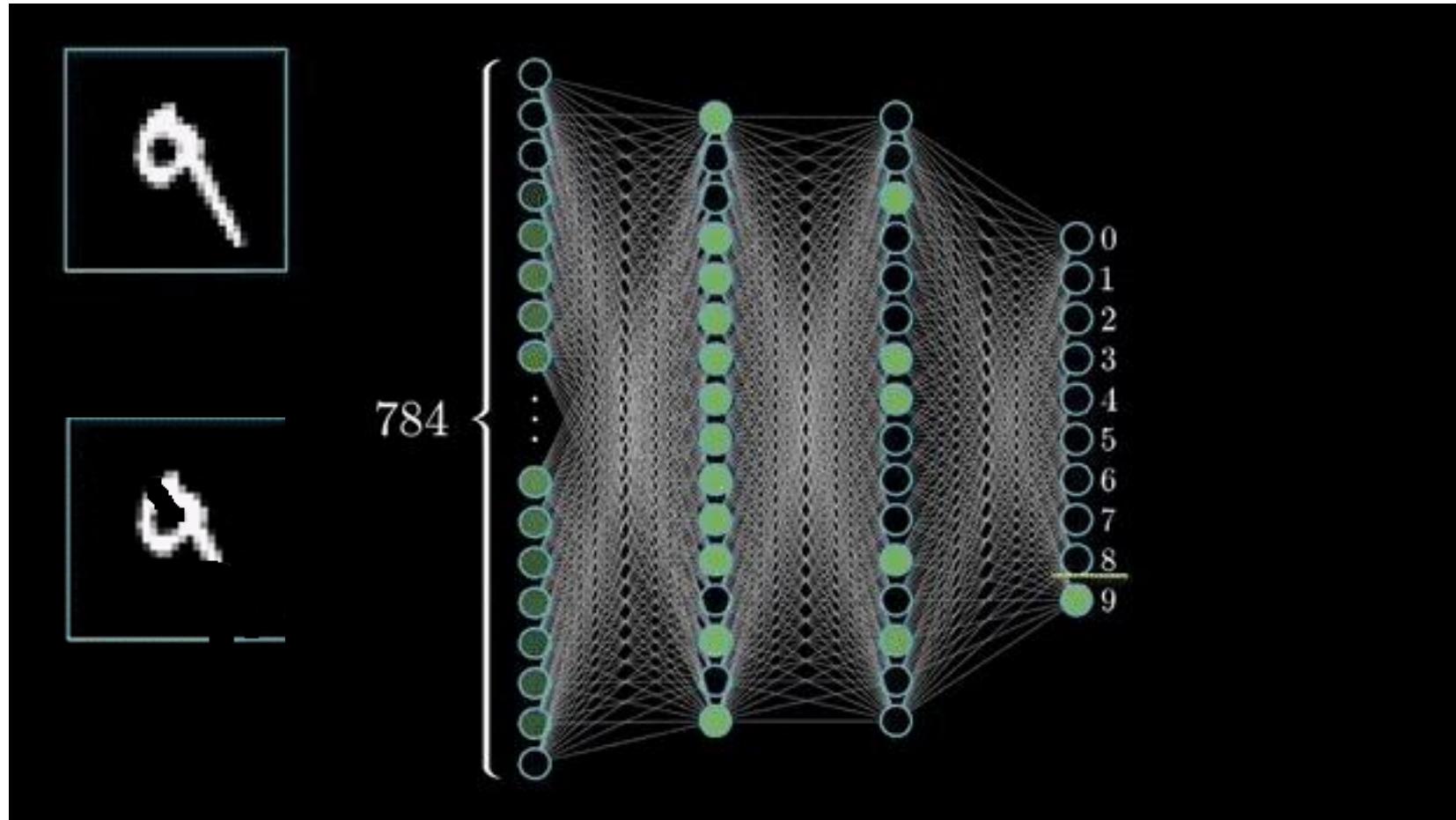
PREDICTION ALGORITHMS (CONTINUED)

Tree-Based		Decision Tree	A series of yes/no rules based on the features , forming a tree, to match all possible outcomes of a decision.	Easy to understand.	Not often used on its own for prediction because it's also often too simple and not powerful enough for complex data.
		Random Forest	Takes advantage of many decision trees, with rules created from subsamples of features. Each tree is weaker than a full decision tree, but by combining them we get better overall performance.	A sort of “wisdom of the crowd”. Tends to result in very high quality models. Fast to train.	Models can get very large. Not easy to understand predictions.
		Gradient Boosting	Uses even weaker decision trees, that are increasingly focused on “hard” examples.	High-performing.	A small change in the feature set or training set can create radical changes in the model. Not easy to understand predictions.

MANY OF THESE WE HAVE SEEN BEFORE ...THE COMPUTATIONAL DIFFERENCE BETWEEN
CLASSIFICATION AND PREDICTION IS OFTEN QUITE SMALL

Neural Networks	 Neural Networks	Interconnected “neurons” that pass messages to each other. Deep learning uses several layers of neural networks stacked on top of one another.	Can handle extremely complex tasks — no other algorithm comes close in image recognition.	Very slow to train, because they often have a very complex architecture. Almost impossible to understand predictions.
-----------------	---	--	---	--

RECALL THE NEURAL NETWORKS RESPONSE IS DETERMINED BY THE WEIGHTS APPLIED TO THE 1000'S OF ARCS CONNECTING THE “NEURONS”



The outputs each have a different “strength”; a different probability

The weights are calculated by the training process

SO LOTS OF CHOICE AND CONSEQUENTLY A LARGE
PART OF MOST PREDICTIONS PROJECTS IS FOCUSED
ON EVALUATING MODELS



TRAINING VS. TEST DISTRIBUTION

- We generally assume that the training and test examples are independently drawn from the same overall distribution of data
 - We call this “i.i.d” which stands for “independent and identically distributed”
- If examples are not independent, requires ***collective classification***
- If test distribution is different, requires ***transfer learning***

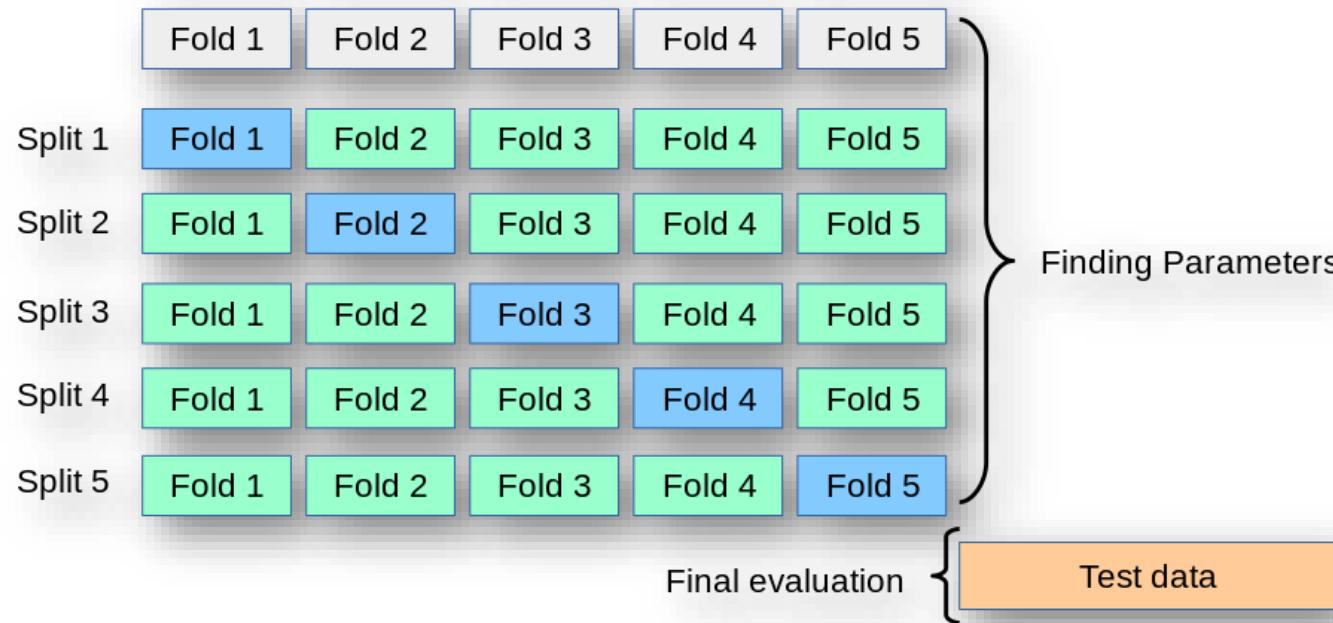
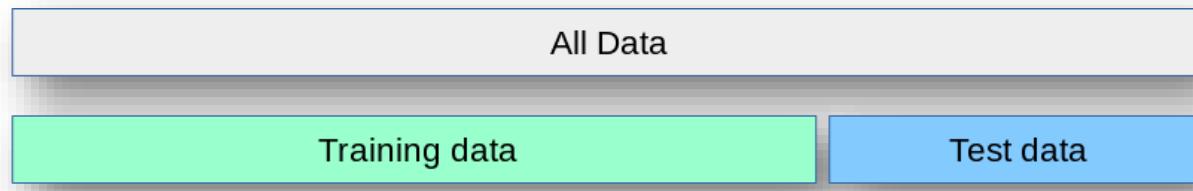


Slide credit: Ray Mooney

THE UNIVERSITY of EDINBURGH
School of Engineering

AGAIN MANY METHODS OF EVALUATING MODELS

- Accuracy
- Precision and recall
- Squared error
- Likelihood
- Posterior probability
- Cost / Utility
- Margin
- Entropy
- K-L divergence
- etc.



Learning the parameters of a prediction function and testing it on the same data is pointless: a model that would just repeat the labels of the samples that it has just seen would have a perfect score but would fail to predict anything useful on yet-unseen data. This situation is called overfitting. To avoid it, it is common practice when developing a machine learning implementation to hold out part of the available data as a test set

Source: Pedro Domingos

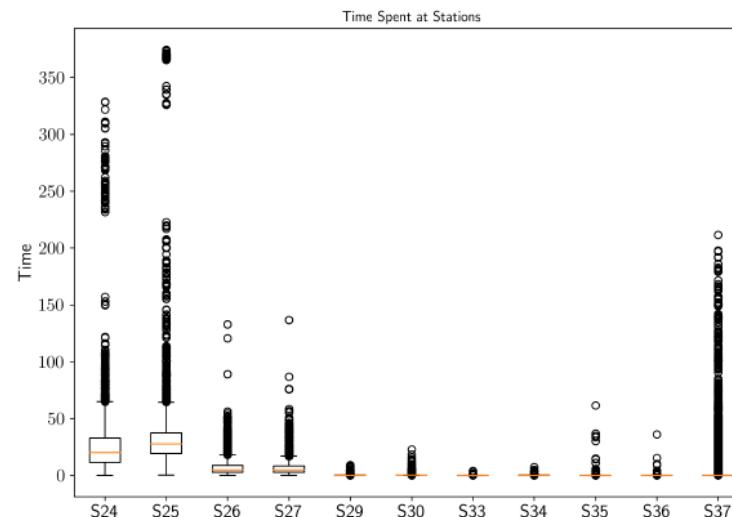
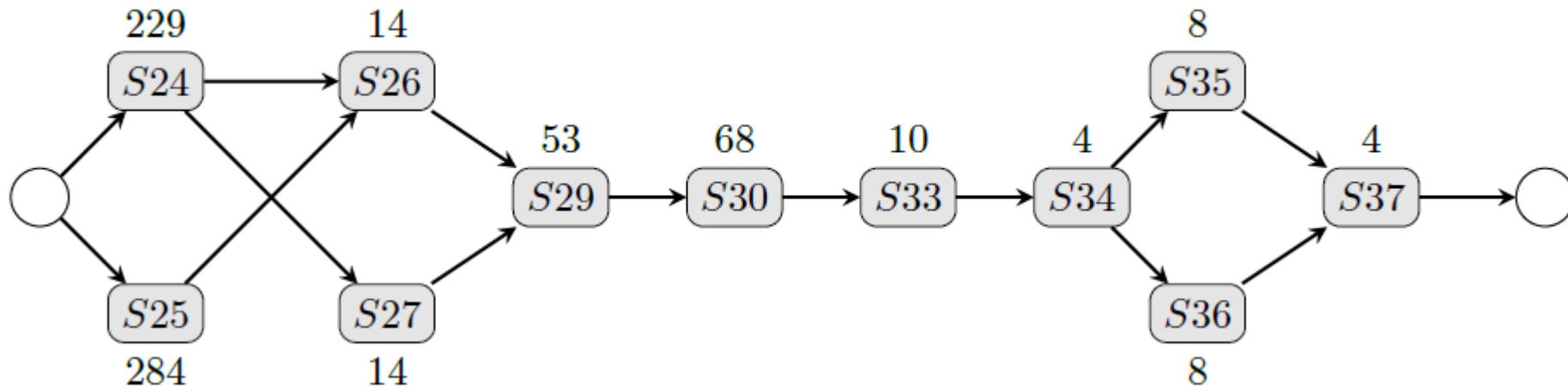


THE UNIVERSITY of EDINBURGH
School of Engineering

EXAMPLE : ANALYSIS OF THE BOSCH PRODUCTION DATA SET

Bosch Dataset records the time taken for products to pass through one of their production plants. The assembly line is divided into 4 segments and 52 workstations. Each workstation performs a variable number of tests and measurements on a given part, accounting in total for 4264 features. Different products may not share the same path along the assembly line, nor there seem to be a common starting or final workstation. Each of the 1,183,747 parts recorded in the dataset follows one of the 4700 unique combinations.





VALIDATION OF DIFFERENT MODELS

Table 3: Performance of SL algorithms with 10-fold cross validation for $n = 100$, $n = 200$ and $n = 270$ features selected

Perf. Measure	Prod.	Type	$n = 100$					$n = 200$					$n = 270$ (keep all features)				
			DNN	Ridge	RF	kNN	NN	DNN	Ridge	RF	kNN	NN	DNN	Ridge	RF	kNN	NN
R^2	1		0.93	0.64	0.90	0.89	0.89	0.95	0.67	0.91	0.89	0.90	0.96	0.67	0.91	0.87	0.90
	2		0.92	0.64	0.89	0.89	0.89	0.95	0.66	0.90	0.88	0.89	0.95	0.67	0.90	0.87	0.89
	3		0.86	0.62	0.83	0.83	0.81	0.87	0.64	0.83	0.82	0.78	0.88	0.65	0.83	0.80	0.75
	4		0.88	0.65	0.85	0.84	0.85	0.88	0.66	0.85	0.83	0.80	0.89	0.67	0.85	0.81	0.79
	5		0.91	0.74	0.91	0.89	0.88	0.93	0.77	0.92	0.91	0.87	0.93	0.77	0.91	0.89	0.85
	6		0.91	0.75	0.91	0.89	0.88	0.94	0.77	0.92	0.91	0.88	0.93	0.77	0.92	0.89	0.86
$RMSE$	1		3.92	8.64	4.64	4.74	4.69	3.11	8.24	4.37	4.87	4.44	2.93	8.22	4.43	5.15	4.54
	2		4.06	8.68	4.78	4.88	4.78	3.24	8.40	4.67	5.07	4.69	3.10	8.29	4.59	5.26	4.71
	3		5.52	9.10	6.01	6.13	6.37	5.22	8.82	6.03	6.33	6.93	5.20	8.75	6.13	6.51	7.31
	4		5.23	8.82	5.69	5.90	5.80	5.08	8.61	5.74	6.18	6.58	4.92	8.56	5.81	6.39	6.76
	5		4.37	7.55	4.33	4.85	5.04	3.90	7.17	4.28	4.51	5.42	3.89	7.15	4.32	4.87	5.69
	6		4.56	7.50	4.38	4.88	5.16	3.73	7.16	4.24	4.54	5.23	3.84	7.14	4.32	4.91	5.59
MAE	1		1.95	6.60	2.72	2.60	3.13	1.36	6.23	2.57	2.78	3.07	1.32	6.21	2.63	3.00	3.15
	2		2.00	6.58	2.77	2.61	3.15	1.44	6.31	2.73	2.85	3.21	1.39	6.22	2.68	3.01	3.24
	3		3.17	5.86	3.28	3.43	4.30	3.14	5.71	3.33	3.45	4.93	3.08	5.66	3.37	3.57	5.15
	4		3.12	5.80	3.22	3.49	4.02	3.08	5.61	3.25	3.36	4.68	3.01	5.58	3.29	3.52	4.80
	5		2.22	5.58	2.37	2.67	3.27	1.93	5.23	2.37	2.38	3.73	2.02	5.21	2.42	2.69	3.97
	6		2.42	5.55	2.42	2.81	3.37	1.90	5.23	2.34	2.40	3.62	1.99	5.21	2.40	2.72	3.92

TIME FOR BREAK: BACK IN 20MINS



REFERENCES

1. Zhydik, O., 2021. *Six Powerful Use Cases for Machine Learning in Manufacturing*. [online] Eleks.com. Available at: <https://eleks.com/blog/machine-learning-in-manufacturing/>
2. Green, J., 2020. *5 ways you can use Machine Learning in manufacturing*. [online] Ancoris.com. Available at: <https://www.ancoris.com/blog/5-ways-machine-learning-manufacturing>
3. Doty, C., 2021. *10 Ways Machine Learning Will Reshape Manufacturing in 2021 - RapidMiner*. [online] RapidMiner. Available at: <https://rapidminer.com/blog/10-ways-machine-learning-in-manufacturing-2021/>
4. Rai, R., Tiwari, M., Ivanov, D. and Dolgui, A., 2021. Machine learning in manufacturing and industry 4.0 applications. *International Journal of Production Research*, 59(16), pp.4773-4778.
5. GLOSSARY, Artificial Intelligence Foundations: Machine Learning, LinkedInLearn
6. McCarthy, J., 2004. WHAT IS ARTIFICIAL INTELLIGENCE?. Computer Science Department Stanford University Stanford.

RESOURCES

- AAAI. Machine Learning. <http://www.aaai.org/Pathfinder/html/machine.html>
- Dietterich, T. (2003). Machine Learning. *Nature Encyclopedia of Cognitive Science*.
- Doyle, P. Machine Learning. <http://www.cs.dartmouth.edu/~brd/Teaching/AI/Lectures/Summaries/learning.html>
- Dyer, C. (2004). Machine Learning. <http://www.cs.wisc.edu/~dyer/cs540/notes/learning.html>
- Mitchell, T. (1997). *Machine Learning*.
- Nilsson, N. (2004). Introduction to Machine Learning. <http://robotics.stanford.edu/people/nilsson/mlbook.html>
- Russell, S. (1997). Machine Learning. *Handbook of Perception and Cognition*, Vol. 14, Chap. 4.
- Russell, S. (2002). *Artificial Intelligence: A Modern Approach*, Chap. 18-20. <http://aima.cs.berkeley.edu>