Data Mining

Clustering

Applications - Business

- Segment customers
  - Group for value
  - Group for what people respond to (marketing, service, deals, etc.)
- Productivity analysis
  - Identify groups in shipping, routes, factories etc.
- Analyzing focus groups/surveys
  - Understand how “types” of people react
  - Reason about your target demographics

Applications - Science

- Imaging
  - Group different types of tissues
  - Group different organisms (micro/macroscopic)
- Biology
  - Cluster gene groups
    - Different groups of expressions
    - Group genetic similarities at population levels
  - Explore new domain (Mushroom example in notebook)
- Education
  - Group usage patterns
  - Group learner types
  - Explore responsiveness per group

Applications - Machine Learning

- As a step in building a predictive model
- Grouping objects before predicting can:
  - Add semantics
  - “Zone in” on a signal
  - Improve accuracy
  - Introduce a non-linear component
  - Allow use of unlabeled data along with labeled data
  - Separate data for validation (e.g. if some images are from the same patient, cluster first and then CV across clusters).

Types of Clusters

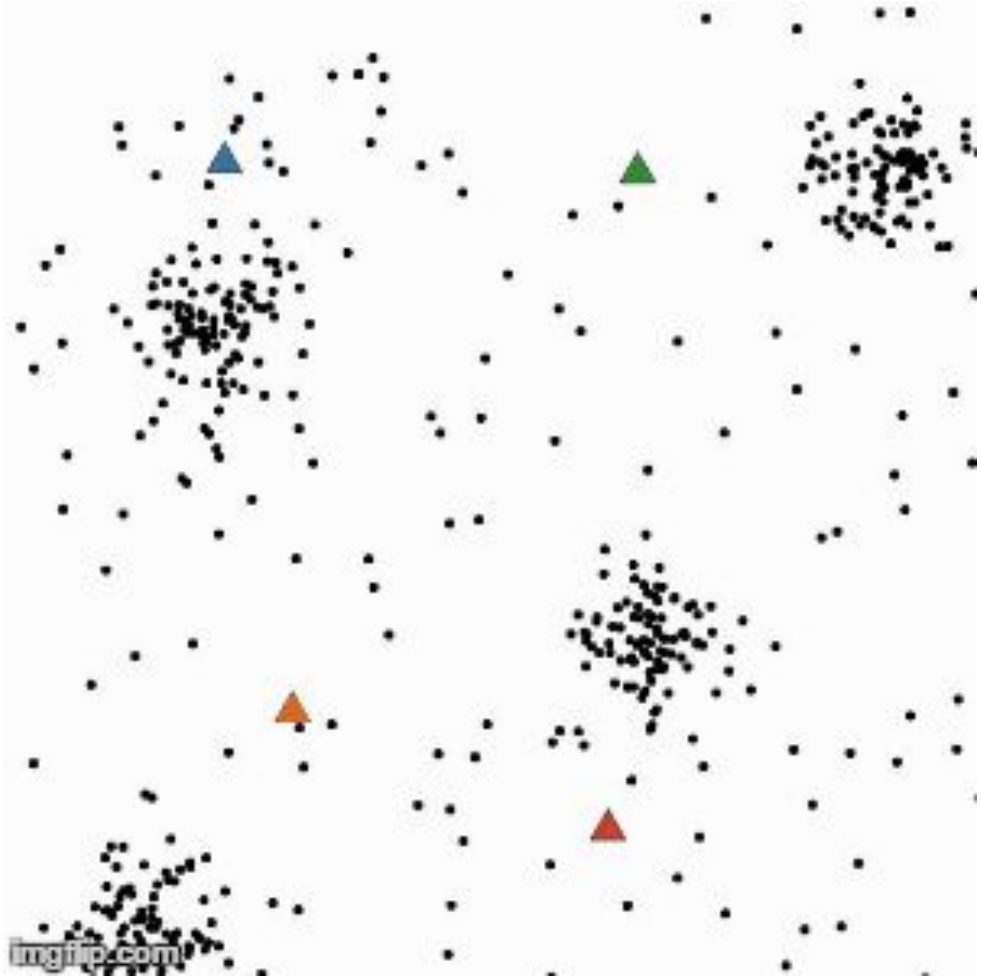
- Overlapping vs Exclusive
  - Overlapping - Observations can fall into more than one cluster
  - Exclusive - Observations belong to at most one cluster
- Exhaustive vs. Non-Exhaustive
  - Exhaustive - All observation are clustered
  - Non-exhaustive - Not all observations are clustered
- Fuzzy vs Non-fuzzy aka Soft vs Hard
  - Fuzzy clusters have a **value** or **degree** of membership
  - E.g. an observation will have (0.3, 0.1, 0.9) which means:
    - 0.3 to first cluster, 0.1 to second cluster, 0.9 to last cluster
  - We will focus on **hard** clusters

Intro to clustering - Distance

- Need a measure of similarity/difference between observations
- Common distance metrics:
  - Euclidean distance
  - Manhattan distance
  - Cosine similarity
  - Custom function of two objects?
    - Sequencing
    - Strings

K-means

- Initialize K random points
- For each point in your dataset:
  - Determine the closest of the K random points
  - This assigns a clustering
- For each clustering:
  - Calculate the mean point
- Repeat with the means as your new central points
- Continue until the clusters do not change



https://commons.wikimedia.org/wiki/File:Kmeans_animation.gif

Scikit-learn

- http://scikit-learn.org/stable/
- Great python library for ML, DM
- http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans
  - K-mean implementation
- K-means isn’t that complicated why do I need a library?

K-means tricks

- Kmean++
  - Uses the data to inform good random point initialization
- Multiple runs
  - Run k-means multiple times
  - Select “best” clustering
- Inertia:
  - Within clusters sum of squares
  - Prefer “nice” shaped clusters
    - Spherical, normal
- K-mediods
  - What if space is hard to compute? I.e. gene sequences
  - Use original datapoints as centroids

Libraries

Numpy - matrix operations

Pandas - Transforming relational data

Scipy - stats and distributions

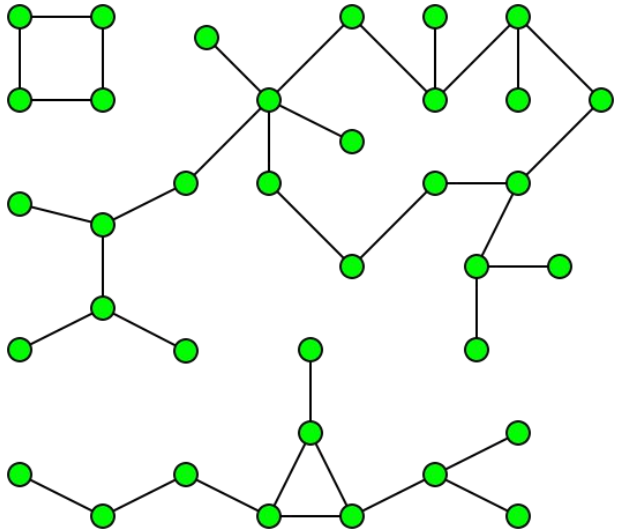
Sklearn - data mining and machine learning

DBSCAN

- http://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html
-
- Density based
- eps - max distance to consider two points as being “beside” each other
- min_samples - minimum number of points beside a point for that point to be considered a **core** point

DBSCAN

1. Compute eps points of every point
2. Determine core points
3. Find the connected components of core points
4. Connected components = clusters
5. For every remaining point **p**:
    a. If **p** is eps distance from a point in a cluster **c**:
        i. Assign **p** to **c**
    b. Else**:**
        i. Mark **p** as noise (no cluster assignment)

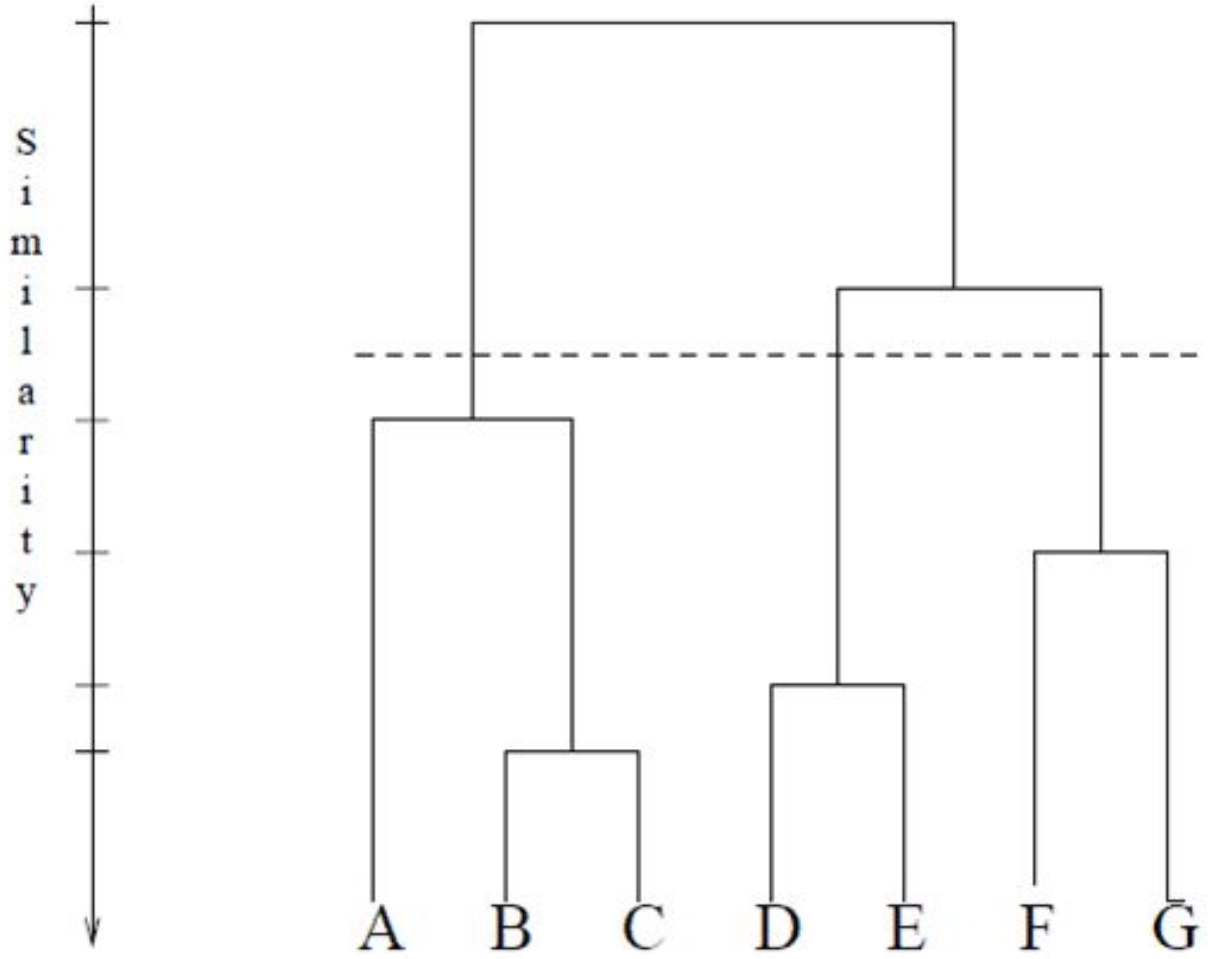


DBSCAN

- Strengths:
  - No need to choose k
  - Can detect irregularly shaped clusters
  - Acknowledges outliers
  - Reasonably fast (n*log(n) time)
- Weaknesses:
  - How to choose eps?
  - How to choose min_clusters?
  - Relies heavily on distance metric
  - Density isn’t always significant (in some domains a few points can be important)
- https://dashee87.github.io/data%20science/general/Clustering-with-Scikit-with-GIFs/

Hierarchical Clustering

- Agglomerative
  - Start with all single point clusters
  - Combine closest clusters
    - Continue until all one
- Divisive
  - Start with every point in one cluster
  - Divide up clusters based on distance of internal points
    - Continue until all single point clusters
- Results can be seen as a hierarchy


Similarity
A
B
C
D
E
F
G

Agglomerative clustering

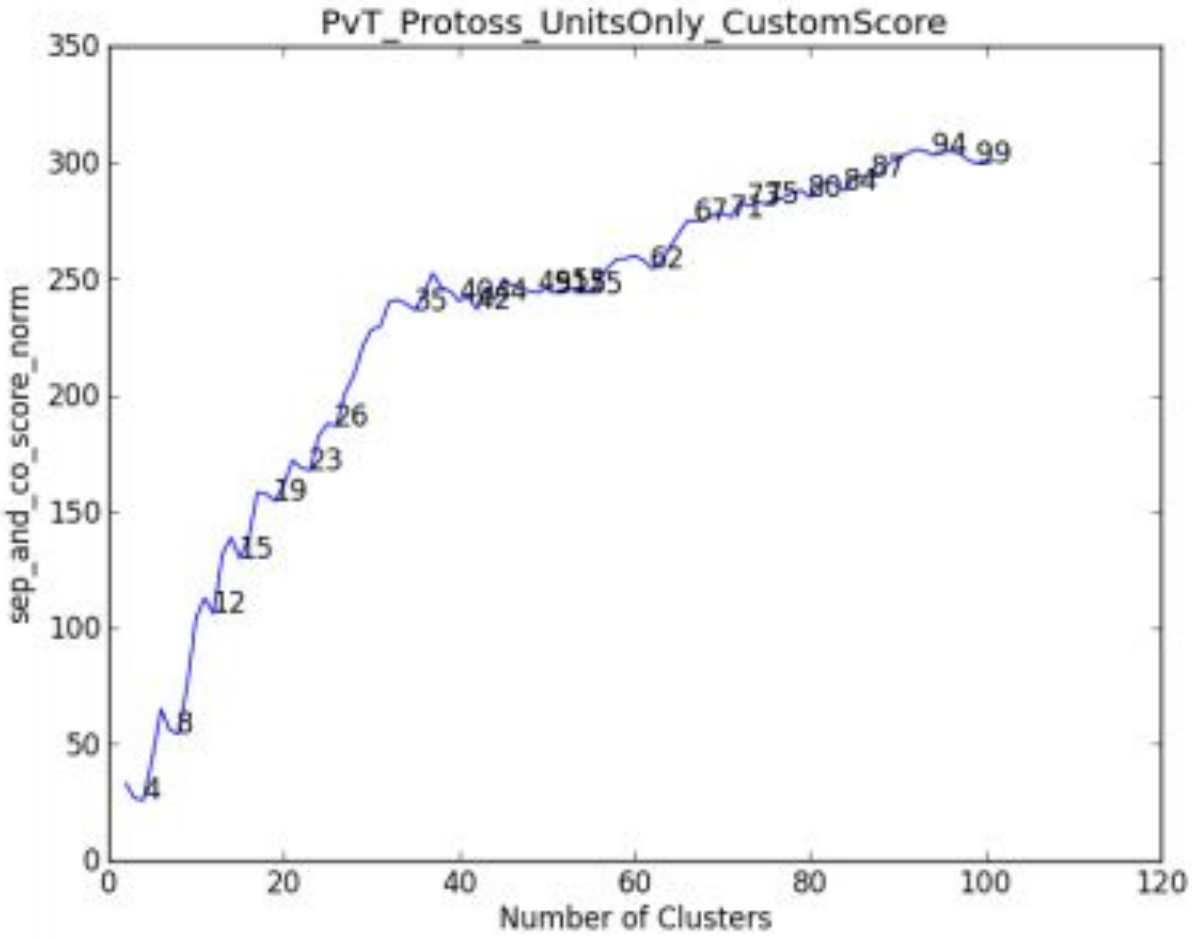
- http://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html
- Possible ‘linkage’ criteria - minimize the following:
  - Average distance of points between clusters
  - Maximum distance of points in each cluster
  - Variance of cluster if merged (Ward)
- Ward is a good nieve choice

Evaluation

- The hardest part!
- A good clustering can:
  - Satisfy a metric (i.e. minimize within-cluster sum of squared distances)
  - Serve a purpose:
    - Provide useful knowledge
    - Make money
    - Expose different “clumps” in dataset

Metrics

- Cohesion
  - Inter-cluster similarity
  - Sum distances between points in a cluster
- Separation
  - Intra-cluster similarity
  - Sum distances between points in one cluster and points in another cluster
  - Can be extended to all other clusters (for a single cluster)
- Sum Separation / Cohesion for all clusters
- Inertia (used in k-means)
  - SSE for each point to its clusters centroid

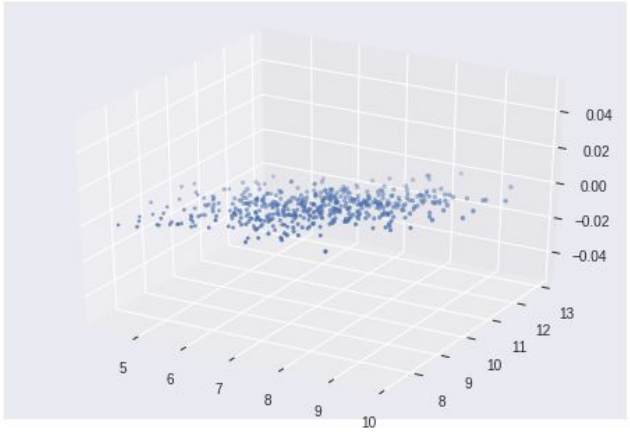

PvT_Protoss_UnitsOnly_CustomScore
sep_and_co_score_norm
Number of Clusters
0
50
100
150
200
250
300
350
0
20
40
60
80
100
120
4
8
12
15
19
23
26
62
94
99

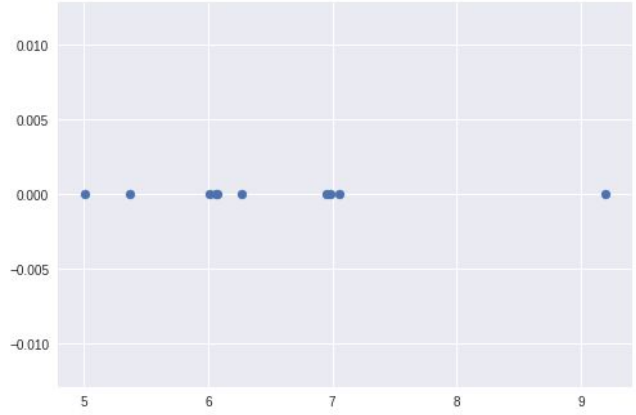
Testing a hypothesis

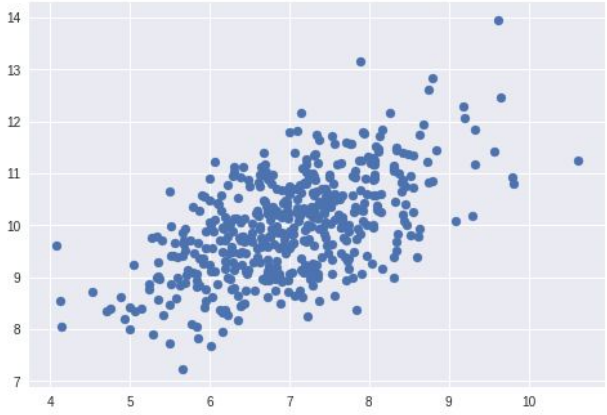
- ANOVA
- Should have hypothesis before clustering
- Multiple cohorts needed for validity
- Don’t use a feature that was used to make the clusters
- E.g. Different shopper groups (behavior) spend different average amounts

Dimensionality Reduction

Visualizing Complex Data

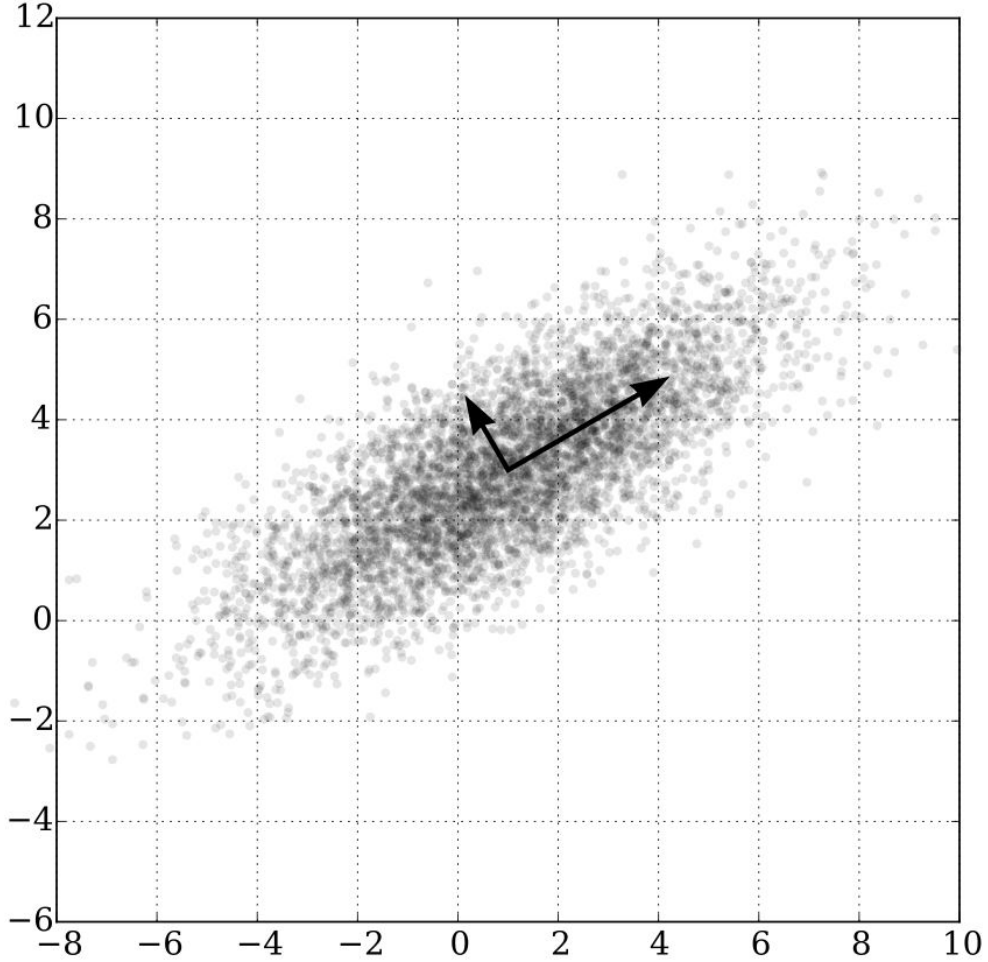






Principal Component Analysis (PCA)

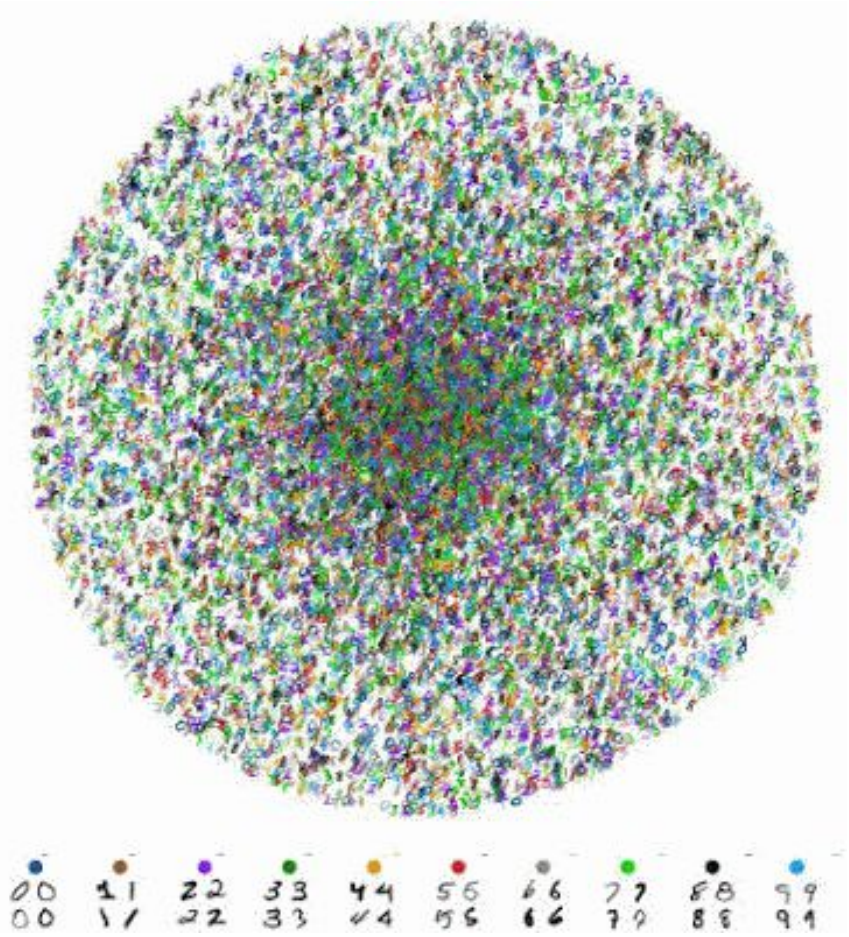
- Linear transformation
- min(observations, features) principal components
- Principal components can be treated like new axes (dimensions)
- First component accounts for **maximum variance**
- Second component accounts for next most variance etc.
- Transform data -> plot on the first few components



https://upload.wikimedia.org/wikipedia/commons/f/f5/GaussianScatterPCA.svg

t-Distributed Stochastic Neighbor Embedding (t-SNE)

- Non-linear
- Computes probability of similarity between objects
  - Not exact - allows efficient computation in high-dimensional spaces
- Creates a low-dimensional mapping such that:
  - Minimizes difference in distributions between low and high dimensional spaces
- Optimization stage is non-convex = slow for really large spaces/number of samples
- For very high dimensions use PCA to get < ~50 features, then use t-SNE


00 11 22 33 44 55 66 77 88 99
00 11 22 33 44 55 66 77 88 99

Association Rules

What is an Association Rule?

- Rule of the form: X => Y
- Read like “If X happens, then Y is likely”
- Associations are probabilistic - “If X happens, then Y is likely but not guaranteed”
- Association Rule Mining (Learning) is the application of algorithms to data sets to discover the rules

Applications

- Understand the patterns in your data:
  - Shopping data
  - Surveys
  - Health Outcomes
- Data should be “transactional”
  - Each data point is an “event” that has some attributes
  - E.g. Each customer check-out is a transaction
  - E.g. Each transaction contains the items purchased

Grocery Store

Burgers	Ketchup	Chips
Burgers	Mustard	Ketchup
Burgers	Ketchup	
Burgers	Salad	
Salad	Cheese	
Bacon	Eggs	Hashbrowns

Burgers => Ketchup

Definitions

Antecedent

- Left hand side of rule e.g. X in X => Y
- Can be a combination of items

Consequent

- Right hand side of rule e.g. Y in X => Y
- Is usually a single item

Definitions

Support:

- Support(X) is the probability of X
- I.e. the frequency of X occurring on its own
- Count the number of occurrences of X / Total number of transactions
- Support(Burgers) = 4/6 = 0.66
- Support({Burgers, Ketchup}) = 0.5

Definitions

Confidence:

- How often the rule holds?
- A measure of “given X”, how often is Y true?
- Conf(X => Y) = Support(X and Y) / Support(X)
- Conf(Burgers => Ketchup) = Support({Burgers, Ketchup}) / Support(Burgers) = 0.5 / 0.66 = 0.75
- When burgers are bought, Ketchup is bought 75% of the time

Definitions

Lift

- Used as a measure of “interestingness”
- Assume X and Y are independent
  - Probability theory says Support(X and Y) = Support(X) * Support(Y) IFF X and Y are independent
- Lift(X => Y) = Support(X and Y) / Support(X) * Support(Y)
- Lift > 1 means the rule has some merit.
- Lift >> 1 can be used to rank rules

Apriori

- Algorithm for discovering Association Rules in transactions
- Given a minimum support it returns subsets that have at least that support
- Bottom-up approach
  - Starts with single items, leaves only those with min support
  - Out of those candidates, combinations can be searched
    - Which potential rules would pass a minimum support?
- Will find all subsets with a given support but it’s quite slow
  - Lots of candidates generated
  - Frequent scans through database

Burgers	Ketchup	Chips
Burgers	Mustard	Ketchup
Burgers	Ketchup	
Burgers	Salad	
Salad	Cheese	Chips
Bacon	Eggs	Hashbrowns
Burger	Mustard	
Salad	Chips	Ketchup
Bacon	Ketchup	Chips
Salad	Ketchup	

Find association rules with 30% support

1.
Ketchup -> 6
Burger -> 5
Chips -> 4
Salad -> 4
~~Bacon -> 2~~
~~Mustard -> 2~~
~~Cheese -> 1~~
~~Eggs -> 1~~
~~Hashbrowns -> 1~~

2.
Burger, Ketchup -> 3
~~Burger, Chips -> 1~~
~~Burger, Salad ->1~~
Ketchup, Chips -> 3
Ketchup, Salad -> 3
Chips, Salad-> 3

3.
~~Ketchup, Chips, Salad -> 1~~