Supply Chain Analytics

Lecture 9: Machine learning in modelling discrete choices

Lecturer: Davranova Dilorom

Machine Learning

- **Supervised:** We are given input samples (X) and output samples (y) of a function y = f(X). We would like to "learn" f, and evaluate it on new data. Types:
  - **Classification:** y is discrete (class labels).
  - **Regression:** y is continuous, e.g. linear regression.

- **Unsupervised:** Given only samples X of the data, we compute a function f such that y = f(X) is "simpler".
  - **Clustering:** y is discrete
  - Y is continuous: **Matrix factorization, Kalman filtering, unsupervised neural networks**.

Machine Learning

- **Supervised:**
  - Is this image a cat, dog, car, house?
  - How would this user score that restaurant?
  - Is this email spam?
  - Is this blob a supernova?
- **Unsupervised**
  - Cluster some hand-written digit data into 10 classes.
  - What are the top 20 topics in Twitter right now?
  - Find and cluster distinct accents of people at Berkeley.

Techniques

- **Supervised Learning:**
  - kNN (k Nearest Neighbors)
  - Linear Regression
  - Naïve Bayes
  - Logistic Regression
  - Support Vector Machines
  - Random Forests
- **Unsupervised Learning:**
  - Clustering
  - Factor analysis
  - Topic Models

k-Nearest Neighbors

Given a query item:
Find k closest matches
in a labeled dataset ↓

k-Nearest Neighbors

Given a query item:
Find k closest matches

Return the most
Frequent label

k-Nearest Neighbors

k = 3 votes for “cat”

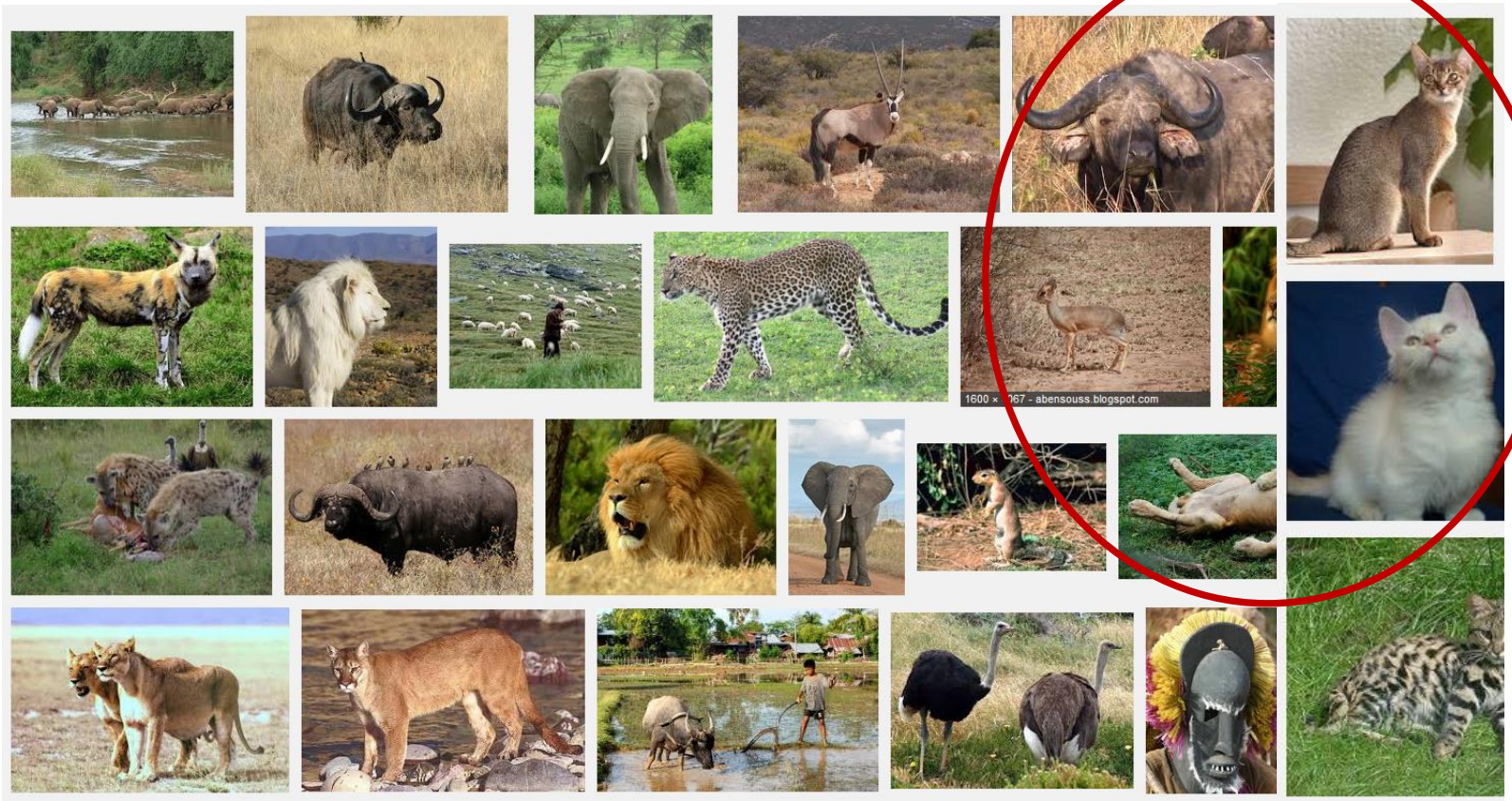
k-Nearest Neighbors

2 votes for cat,

1 each for Buffalo,

Deer, Lion

Cat wins…



k-NN issues

The Data is the Model

- No training needed.
- Accuracy generally improves with more data.
- Matching is simple and fast (and single pass).
- Usually need data in memory, but can be run off disk.

Minimal Configuration:

- Only parameter is k (number of neighbors)
- Two other choices are important:
  - Weighting of neighbors (e.g. inverse distance)
  - Similarity metric

K-NN metrics

- **Euclidean Distance:** Simplest, fast to compute

$$d(x, y) = \|x - y\|$$

- **Cosine Distance:** Good for documents, images, etc.

$$d(x, y) = 1 - \frac{x \cdot y}{\|x\|\|y\|}$$

- **Jaccard Distance:** For set data:

$$d(X, Y) = 1 - \frac{|X \cap Y|}{|X \cup Y|}$$

- **Hamming Distance:** For string data:

$$d(x, y) = \sum_{i=1}^{n} (x_i \neq y_i)$$

K-NN metrics

- **Manhattan Distance:** Coordinate-wise distance

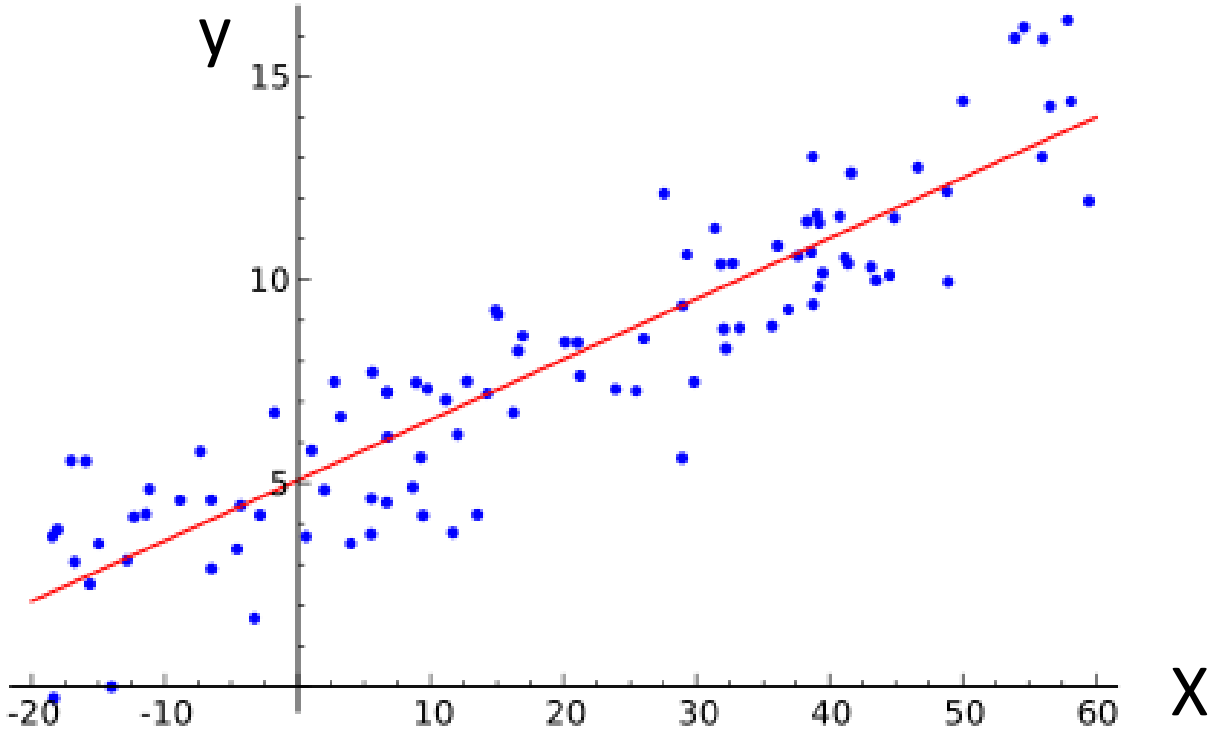
$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

- **Edit Distance:** for strings, especially genetic data.

- **Mahalanobis Distance:** Normalized by the sample covariance matrix – unaffected by coordinate transformations.

Linear Regression

We want to find the best line (linear function y=f(X)) to explain the data.



Linear Regression

The predicted value of y is given by:

$$\hat{y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j$$

The vector of coefficients $\hat{\beta}$ is the regression model.

If $X_0 = 1$, the formula becomes a matrix product:

$$\hat{y} = \mathrm{X}\,\hat{\beta}$$

Linear Regression

We can write all of the input samples in a single matrix **X**:

i.e. **rows of** $\mathbf{X} = \begin{pmatrix} X_{11} & \cdots & X_{1n} \\ \vdots & \ddots & \vdots \\ X_{m1} & \cdots & X_{mn} \end{pmatrix}$

are **distinct observations**, **columns of X** are **input features**.

Residual Sum-of-Squares

To determine the model parameters $\hat{\beta}$ from some data, we can write down the Residual Sum of Squares:

$$\mathrm{RSS}(\beta) = \sum_{i=1}^{N} (y_i - \beta x_i)^2$$

or symbolically $\mathrm{RSS}(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta)$. To minimize it, take the derivative wrt β which gives:

$$\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) = 0$$

And if $\mathbf{X}^T\mathbf{X}$ is non-singular, the unique solution is:

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Iterative Regression Solutions

The exact method requires us to invert a matrix $(\mathbf{X}^T\mathbf{X})$ whose size is nfeatures x nfeatures. This will often be **too big**.

There are many gradient-based methods which reduce the RSS error by taking the **derivative wrt** β

$$\mathrm{RSS}(\beta) = \sum_{i=1}^{N} (y_i - \beta x_i)^2$$

which was

$$\nabla = \mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta)$$

Stochastic Gradient

A very important set of iterative algorithms use **stochastic gradient** updates.

They use a **small subset or mini-batch X** of the data, and use it to compute a gradient which is added to the model

$$\beta' = \beta + \alpha \nabla$$

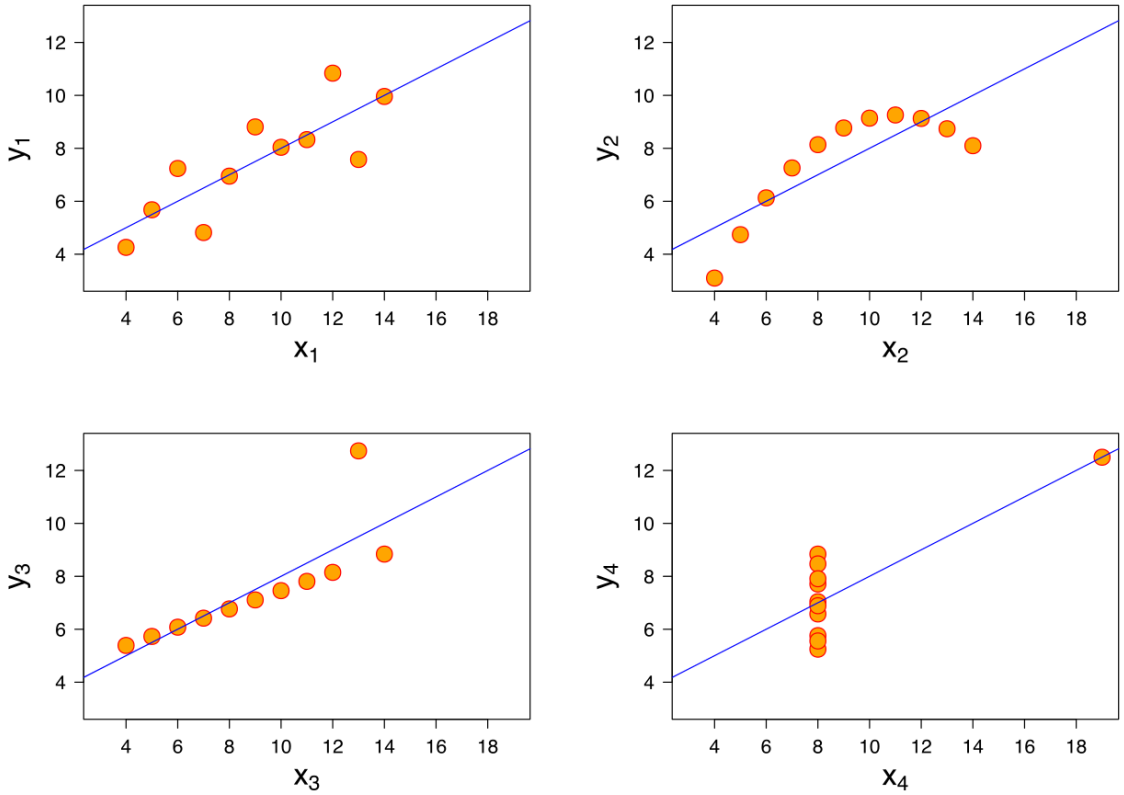
Where α is called the **learning rate**.

These updates happen **many times** in one pass over the dataset.

Its possible to compute high-quality models with very few passes, sometime with less than one pass over a large dataset.

R^2-values and P-values

We can **always** fit a linear model to any dataset, but how do we know if there is a **real linear relationship**?



R^2-values and P-values

Approach: Use a hypothesis test. The null hypothesis is that there is no linear relationship ($\beta = 0$).

Statistic: Some value which should be small under the null hypothesis, and large if the alternate hypothesis is true.

R-squared: a suitable statistic. Let $\hat{y} = \mathrm{X}\,\hat{\beta}$ be a predicted value, and $\bar{y}$ be the sample mean. Then the R-squared statistic is

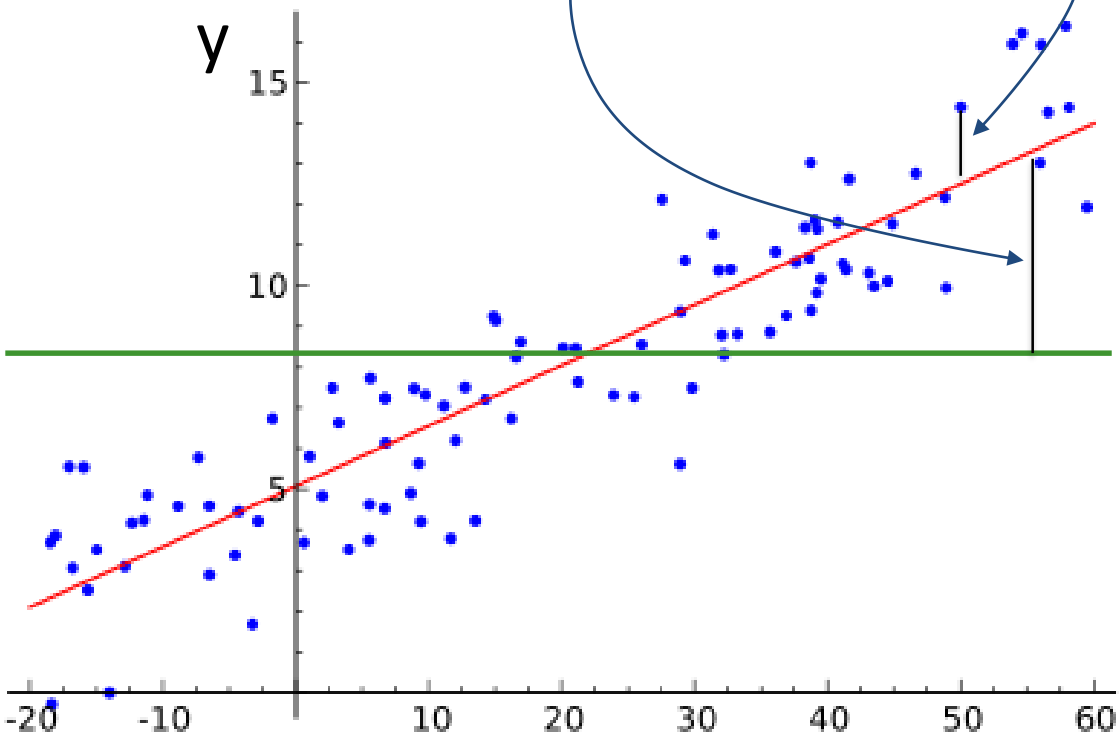
$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$

And can be described as the fraction of the total variance not explained by the model.

R-squared

Small if good fit

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$$



R^2-values and P-values

Statistic: From R-squared we can derive another statistic (using degrees of freedom) that has a standard distribution called an **F-distribution**.

From the CDF for the F-distribution, we can derive a **P-value** for the data.

The P-value is, as usual, the probability of observing the data under the null hypothesis of no linear relationship.

If **p is small**, say less than 0.05, we conclude that **there is a linear relationship**.

Clustering – Why?

Clustering has one or more goals:

- Segment a large set of cases into small subsets that can be treated similarly - segmentation
- Generate a more compact description of a dataset - compression
- Model an underlying process that generates the data as a mixture of different, localized processes – representation

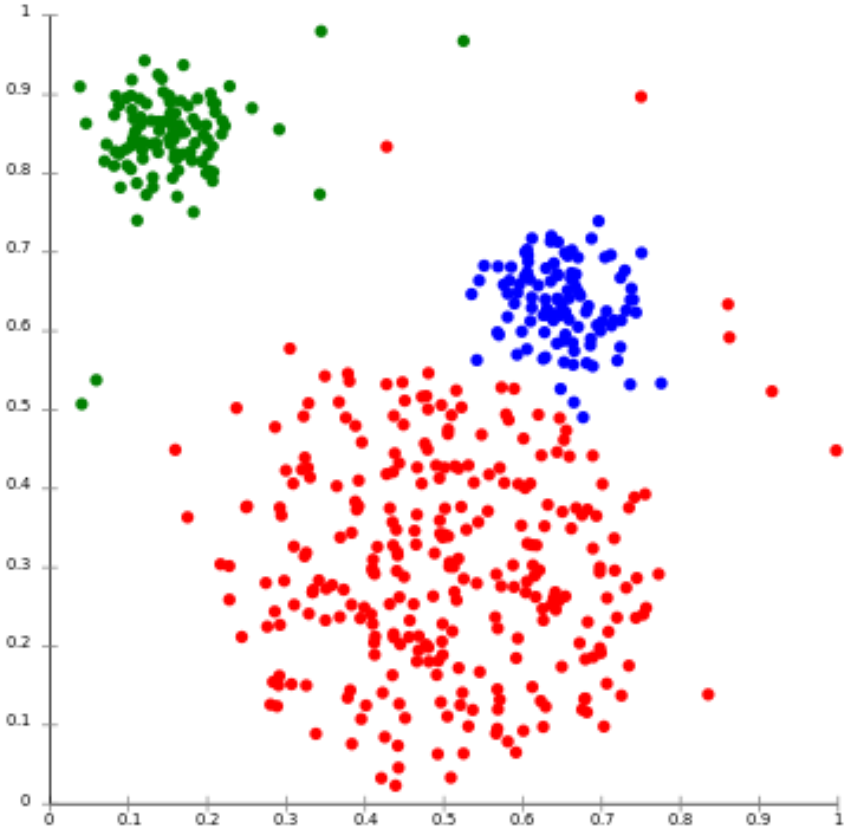
Clustering – Why?

Examples:

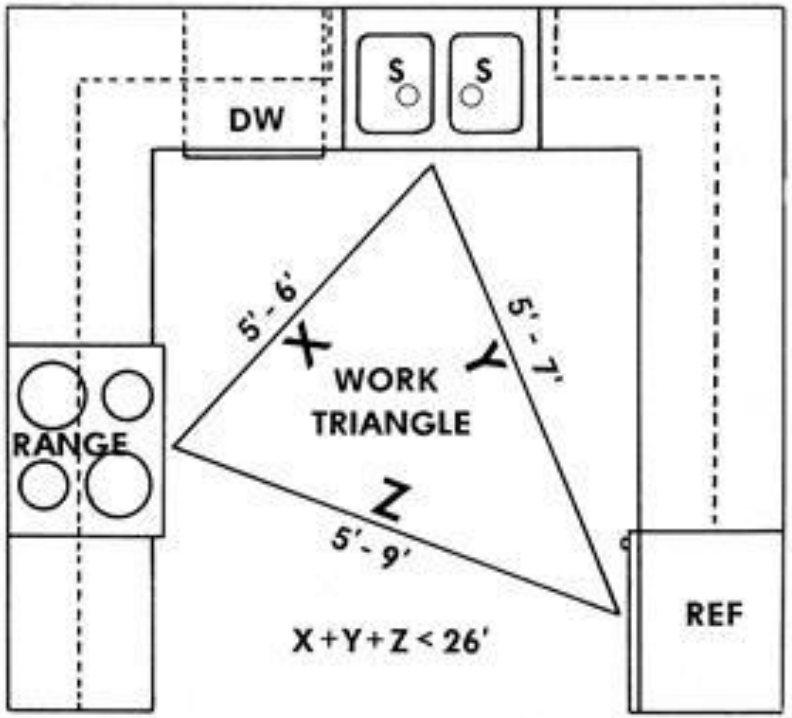
- Segment: image segmentation
- Compression: Cluster-based kNN, e.g. handwritten digit recognition.
- Underlying process: Accents of people at Berkeley (??) – because place of origin strongly influences the accent you have.

Stereotypical Clustering

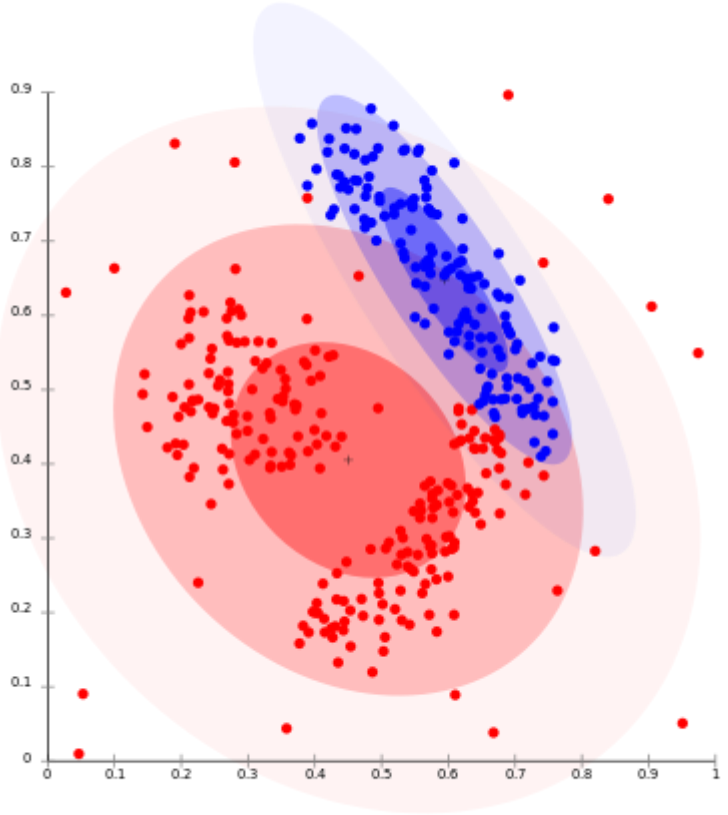
Note: Points are samples plotted in feature space, e.g. 10,000-dimensional space for 100x100 images.



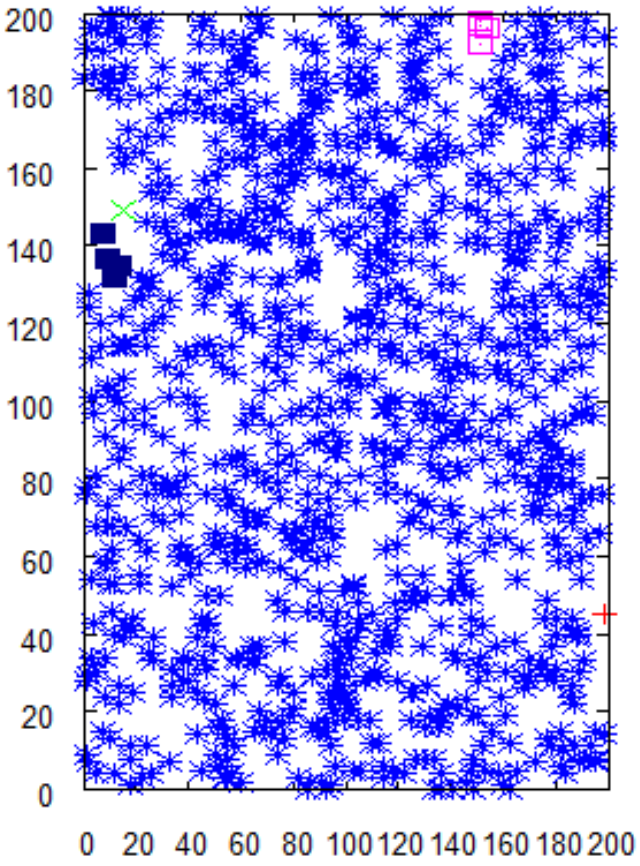
Model-based Clustering

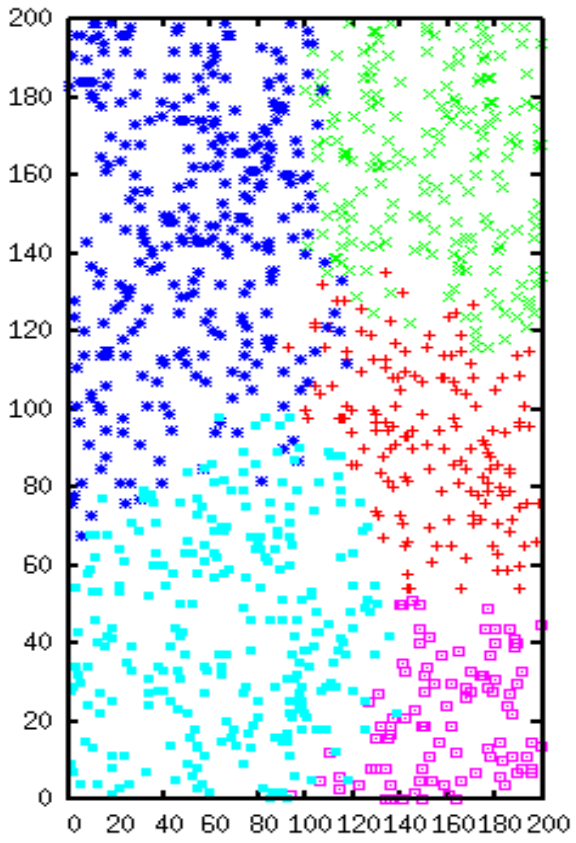


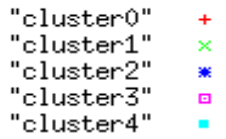
Model-based Clustering



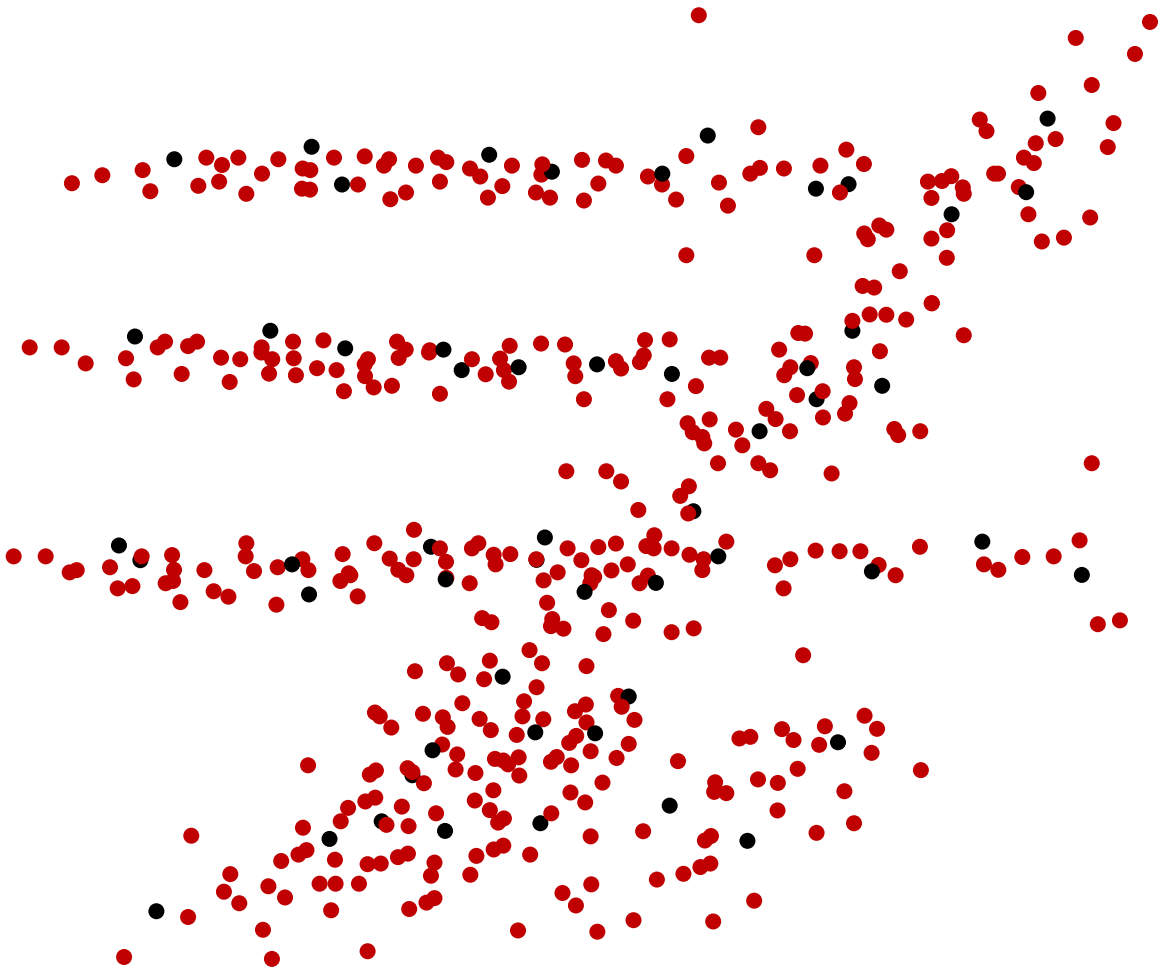
Clustering for Segmentation







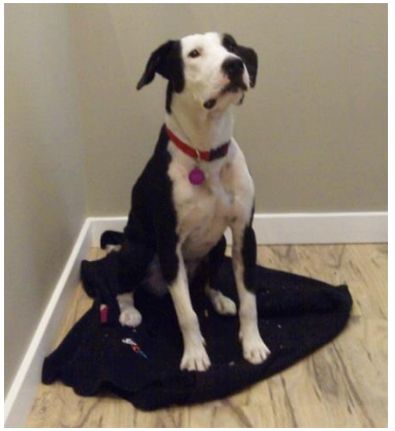
Condensation/Compression



“Cluster Bias”

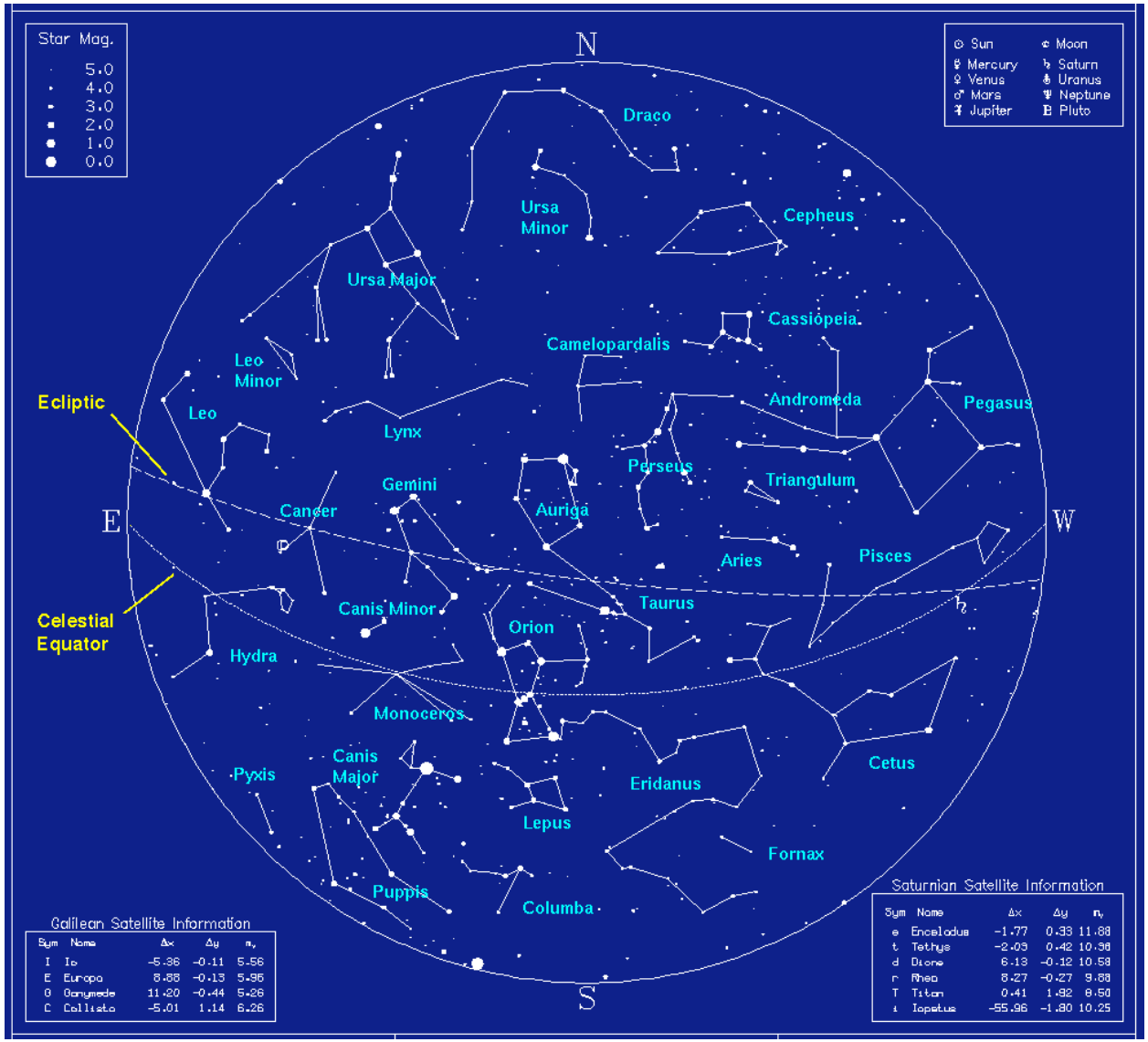
- Human beings conceptualize the world through categories represented as *examplars* (Rosch 73, Estes 94).





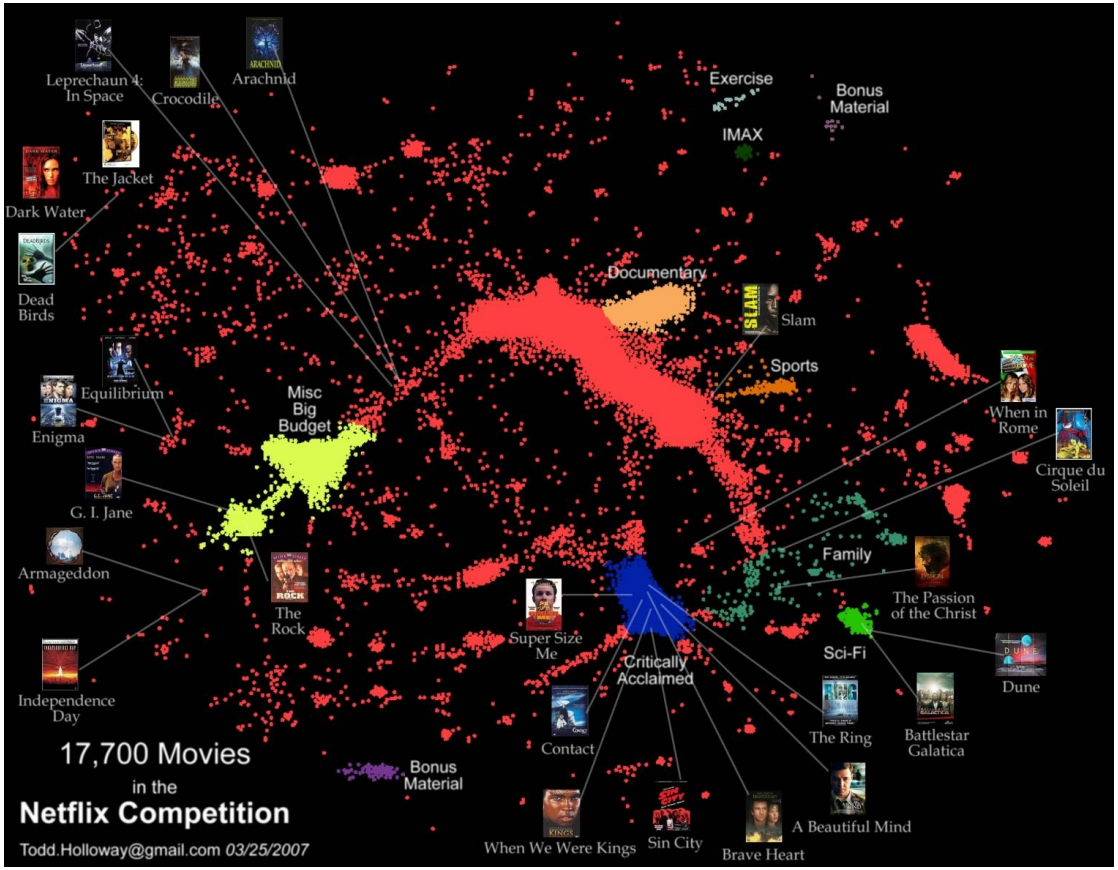
- We tend to see cluster structure whether it is there or not.
- Works well for dogs, but…

Cluster Bias



Netflix

- More of a continuum than discrete clusters
- Factor models, kNN do much better than discrete cluster models.



“Cluster Bias”

Upshot:

- **Clustering is used more than it should be**, because people assume an underlying domain has discrete classes in it.
- This is especially true for characteristics of people, e.g. Myers-Briggs personality types like “ENTP”.
- In reality the underlying data is usually **continuous**.
- Just as with Netflix, continuous models (dimension reduction, kNN) tend to do better.

Terminology

- **Hierarchical clustering:** clusters form a hierarchy. Can be computed bottom-up or top-down.
- **Flat clustering:** no inter-cluster structure.

- **Hard clustering:** items assigned to a unique cluster.
- **Soft clustering:** cluster membership is a real-valued function, distributed across several clusters.

K-means clustering

The standard k-means algorithm is based on **Euclidean distance**.

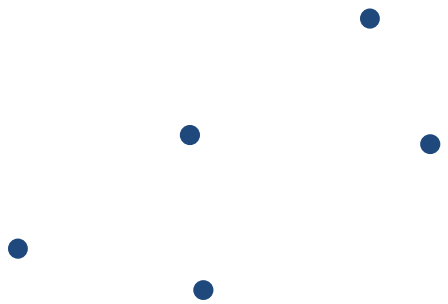
The cluster quality measure is an **intra-cluster measure only**, equivalent to the sum of item-to-centroid kernels.

A simple greedy algorithm locally optimizes this measure (usually called Lloyd's algorithm):

- **Find the closest cluster center** for each item, and assign it to that cluster.
- **Recompute the cluster centroid** as the mean of items, for the newly-assigned items in the cluster.

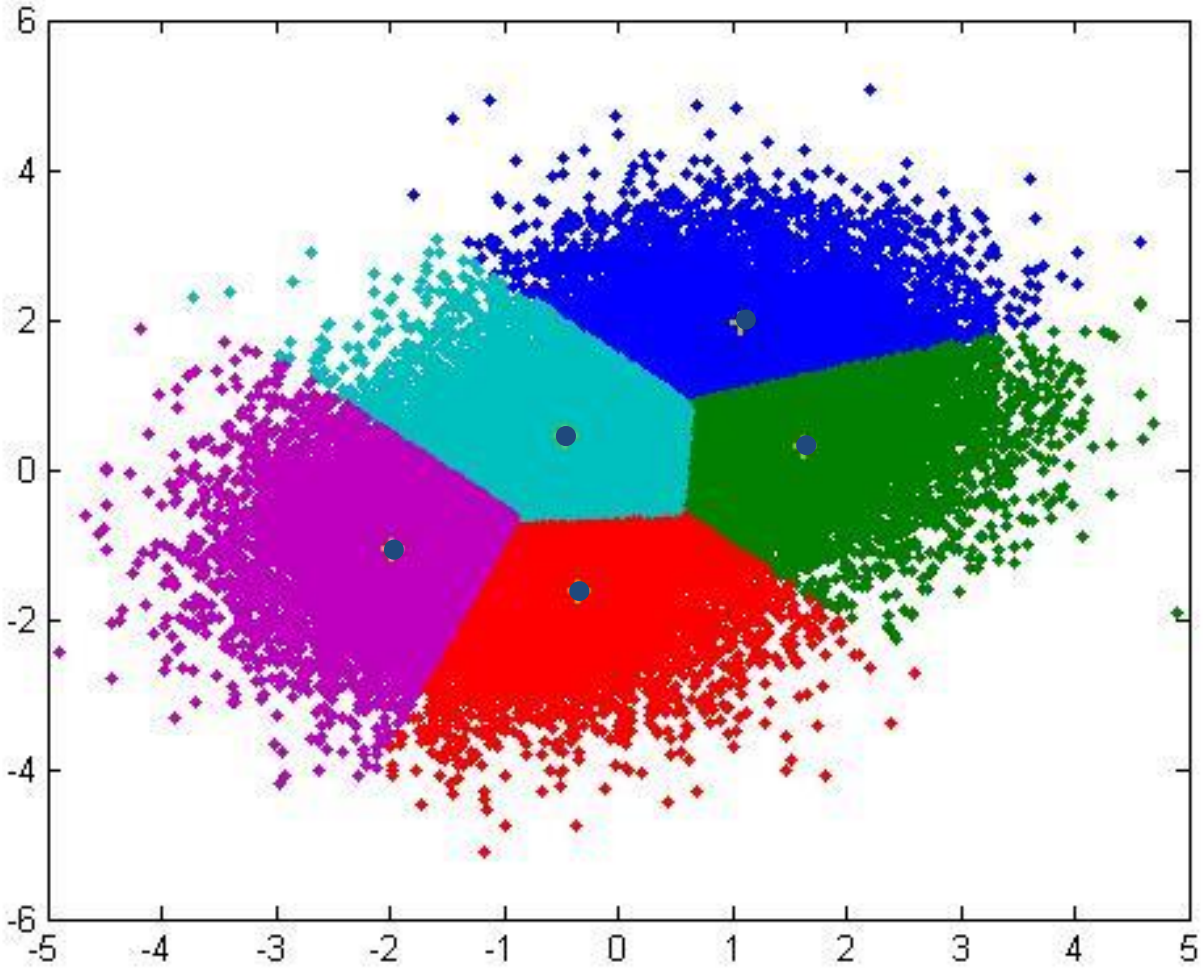
K-means clustering

Cluster centers – can pick by sampling the input data.



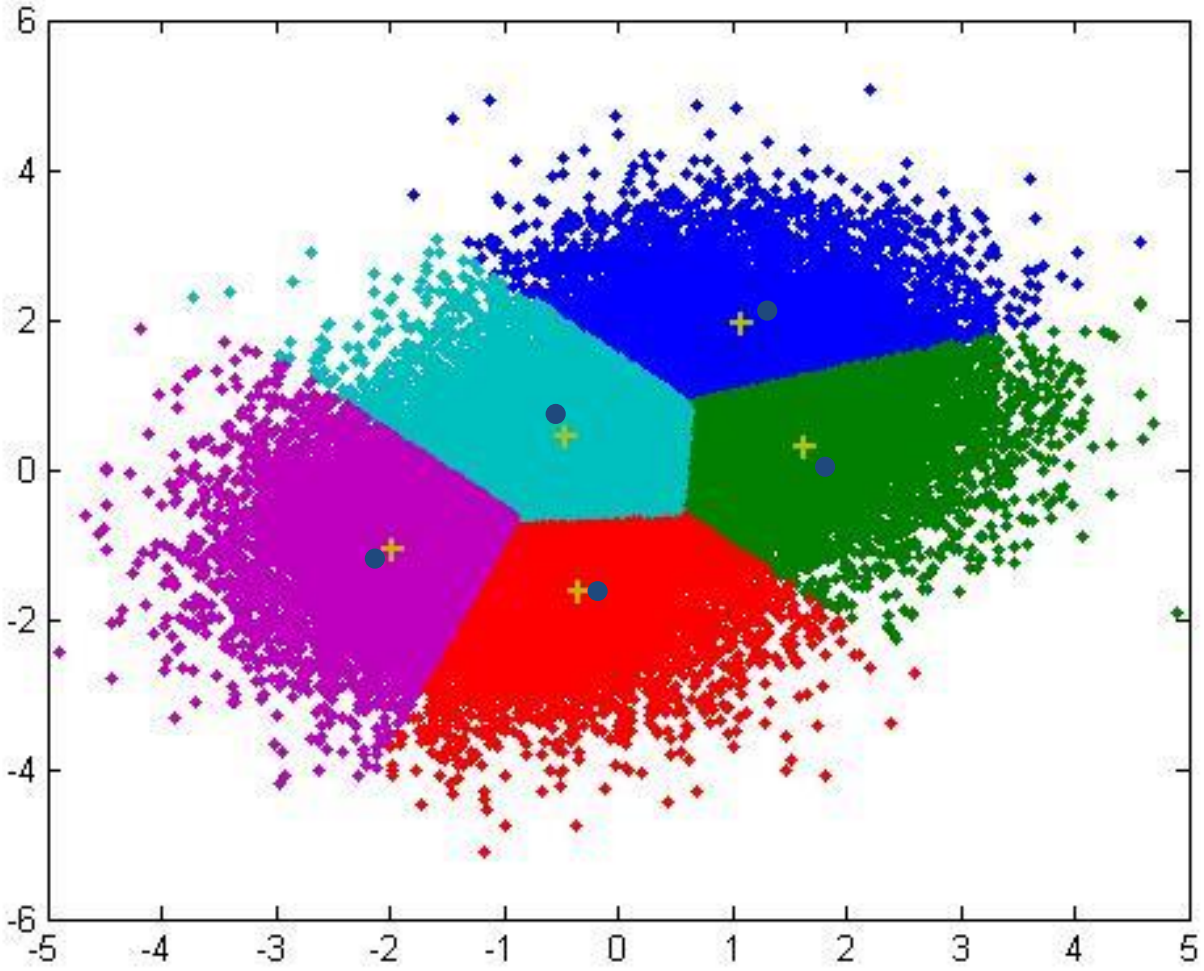
K-means clustering

Assign points to closest center



K-means clustering

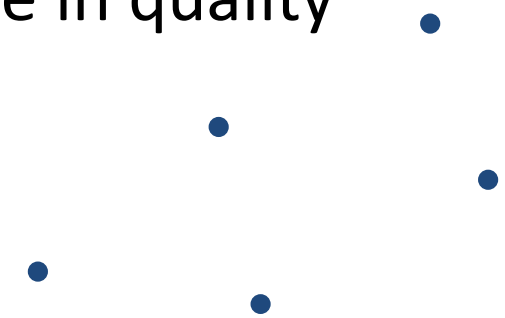
Recompute centers (old = cross, new = dot)



K-means clustering

Iterate:

- For fixed number of iterations
- Until no change in assignments
- Until small change in quality



K-means properties

- It's a greedy algorithm with random setup – **solution isn't optimal** and varies significantly with different initial points.
- Very simple convergence proofs.
- **Performance is O(nk) per iteration**, not bad and can be heuristically improved.
  n = total features in the dataset, k = number clusters
- Many generalizations, e.g.
  - Fixed-size clusters
  - Simple generalization to m-best soft clustering
- As a "local" clustering method, it works well for data condensation/compression.

Choosing clustering dimension

- AIC or Akaike Information Criterion:

$$\textbf{AIC:} \qquad K = \arg\min_{K}[-2L(K) + 2q(K)]$$

- K=dimension, L(K) is the likelihood (could be RSS) and q(K) is a measure of model complexity (cluster description complexity).

- AIC favors more compact (fewer clusters) clusterings.

- For sparse data, AIC will incorporate the number of non-zeros in the cluster spec. Lower is better.

5-minute break

Outline for this Evening

- Three Basic Algorithms
  - kNN
  - Linear Regression
  - K-Means
- **Training Issues**
  - Measuring model quality
  - Over-fitting
  - Cross-validation

Model Quality

Almost every model **optimizes some quality criterion**:

- For linear regression it was the **Residual Sum-of-Squares**
- For k-Means it is the **"Inertia"** – the mean squared distance from each sample to its cluster center.
- …

The quality criterion is chosen often because of its good properties:

- **Convexity:** so that there is a unique, best solution
- **Closed form** for the optimum (linear regression) or at least for the gradient (for SGD).
- An algorithm that **provably converges**.

Model Quality

There are typically other criteria used to measure the quality of models. e.g. for clustering models:

- Silhouette score
- Inter-cluster similarity (e.g. mutual information)
- Intra-cluster entropy

For regression models:

- Stability of the model (sensitivity to small changes)
- Compactness (sparseness or many zero coefficients)

Evaluating Clusterings: Silhouette

The silhouette score is

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

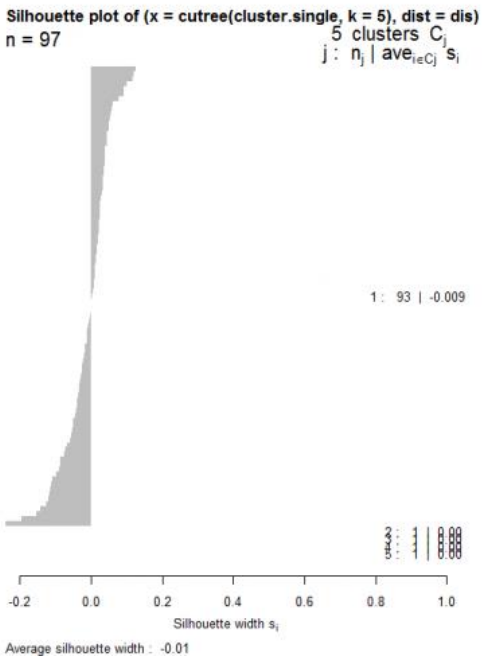
where a(i) is the mean distance from sample i to its own cluster, b(i) the mean distance from i to the second-closest cluster.

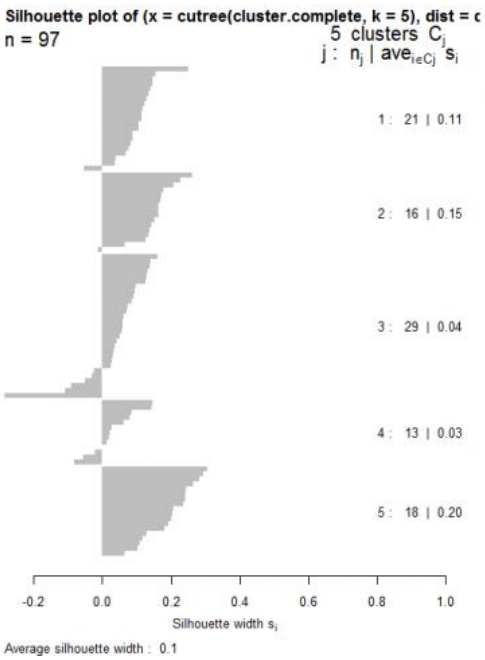
- Perhaps surprisingly, silhouette scores can be, and often are, negative.

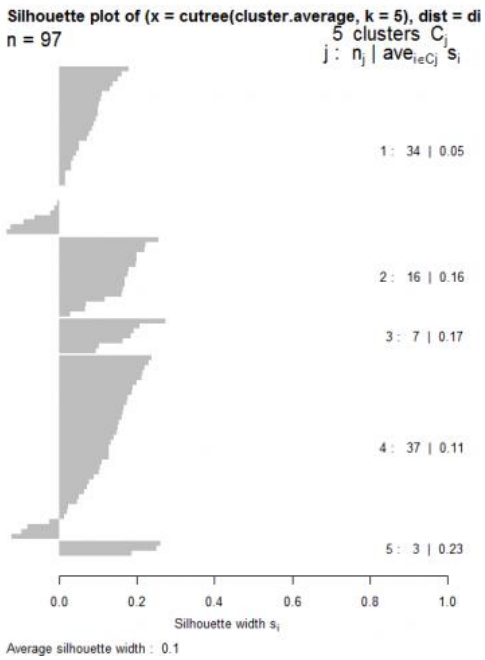
Evaluating Clusterings: Silhouette

Silhouette plot: horizontal bars with cluster score.

Sort (vertically) first by cluster, then by score.







Regularization with Secondary Criteria

While secondary criteria can be measured after the model is built, its too late then to affect the model.

Using secondary criteria **during** the optimization process is called **"regularization".**

Examples:

- **L1 regularization** adds a term to the measure being optimized which is the **sum of absolute value of model coefficients**.
- **L2 regularization** adds a term to the measure being optimized which is the **sum of squares of model coefficients**.

Regularization with Secondary Criteria

L1 regularization in particular is very widely used. It has the following impacts:

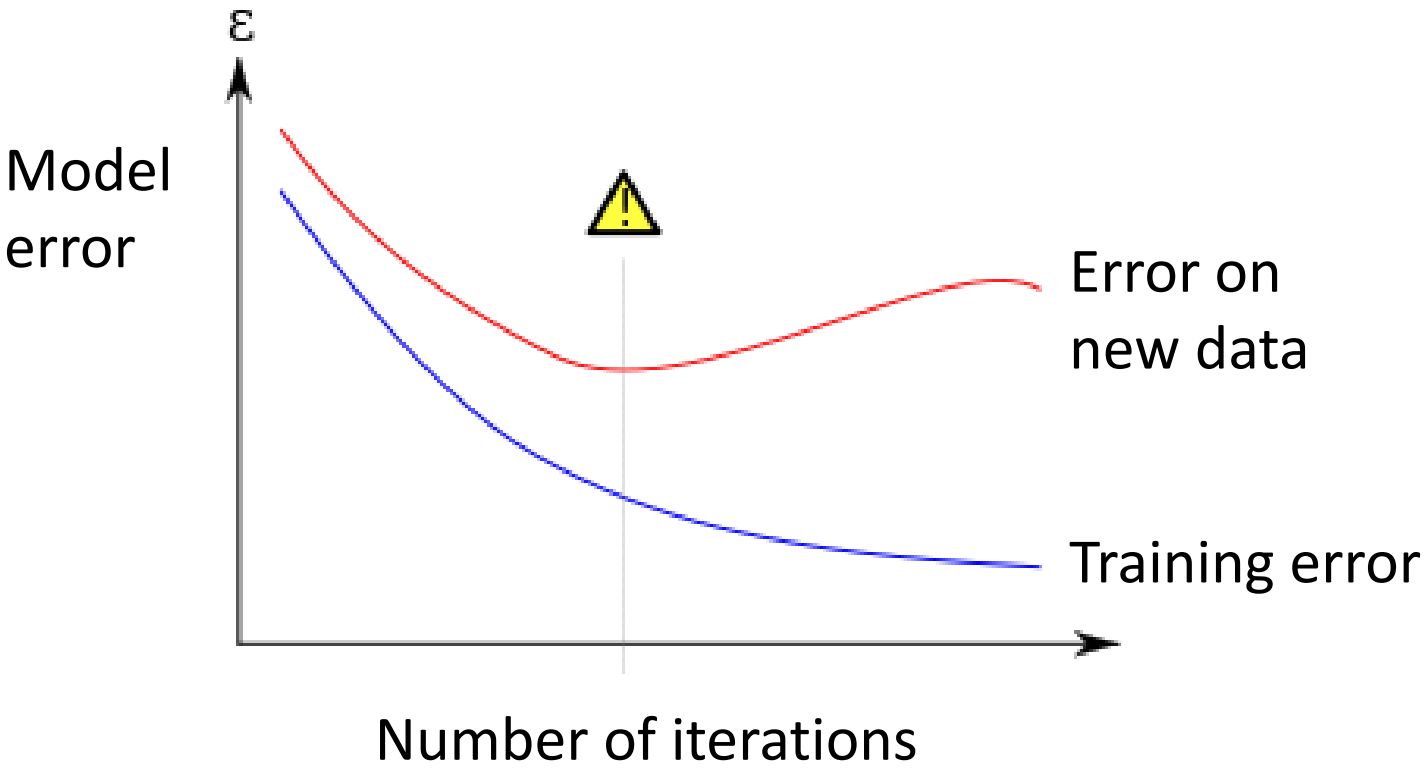
- Yields a **convex optimization** problem in many cases, so there is a unique solution.
- The solution is usually **stable** to small input changes.
- The solution is **quite sparse** (many zero coefficients) and requires less disk and memory to run.
- L1 regularization on factorization models tends to **decrease the correlation** between model factors.

Over-fitting

- Your model should ideally fit an **infinite sample** of the type of data you're interested in.
- In reality, you only have a **finite set** to train on. A good model for this subset is a good model for the infinite set, up to a point.
- Beyond that point, the model quality (measured on new data) starts to **decrease**.
- Beyond that point, the model is **over-fitting** the data.

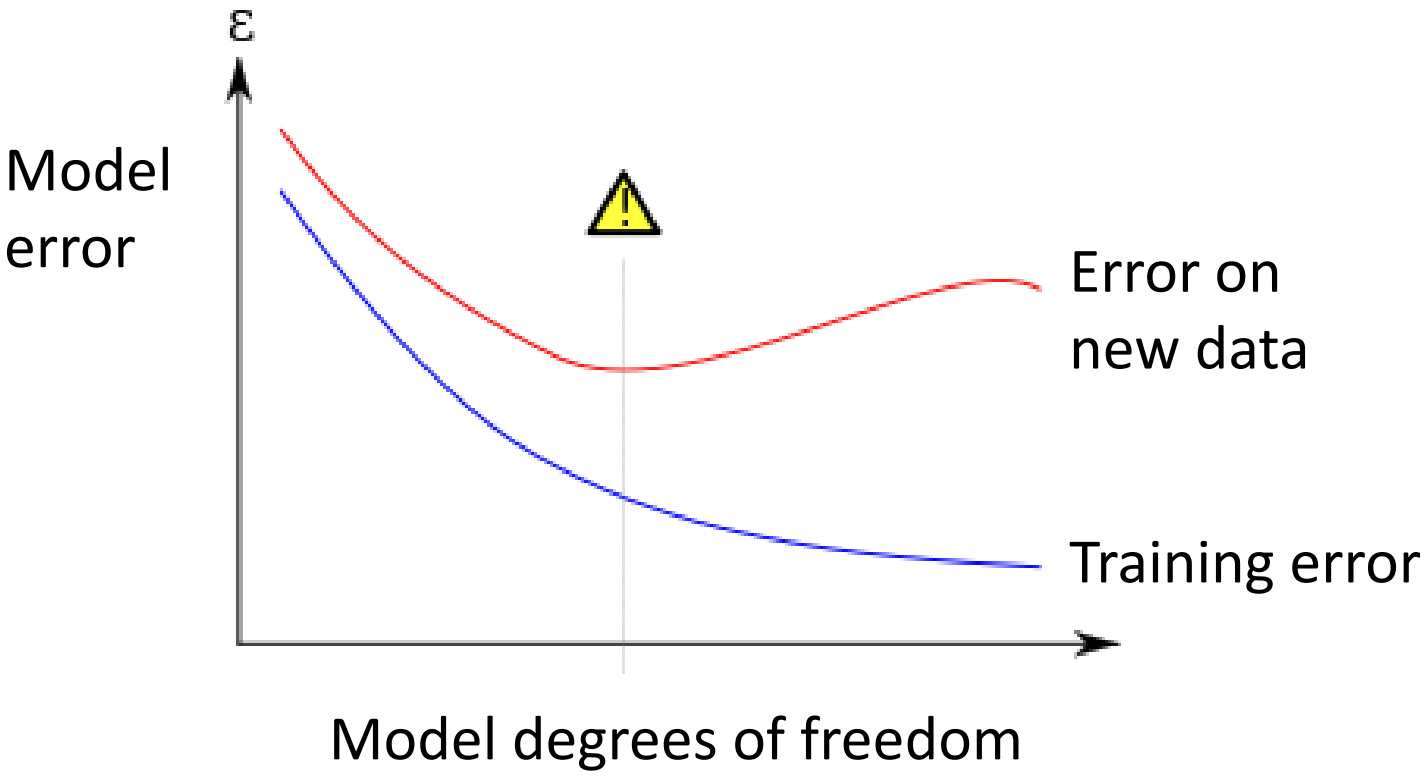
Over-fitting

Over-fitting during training



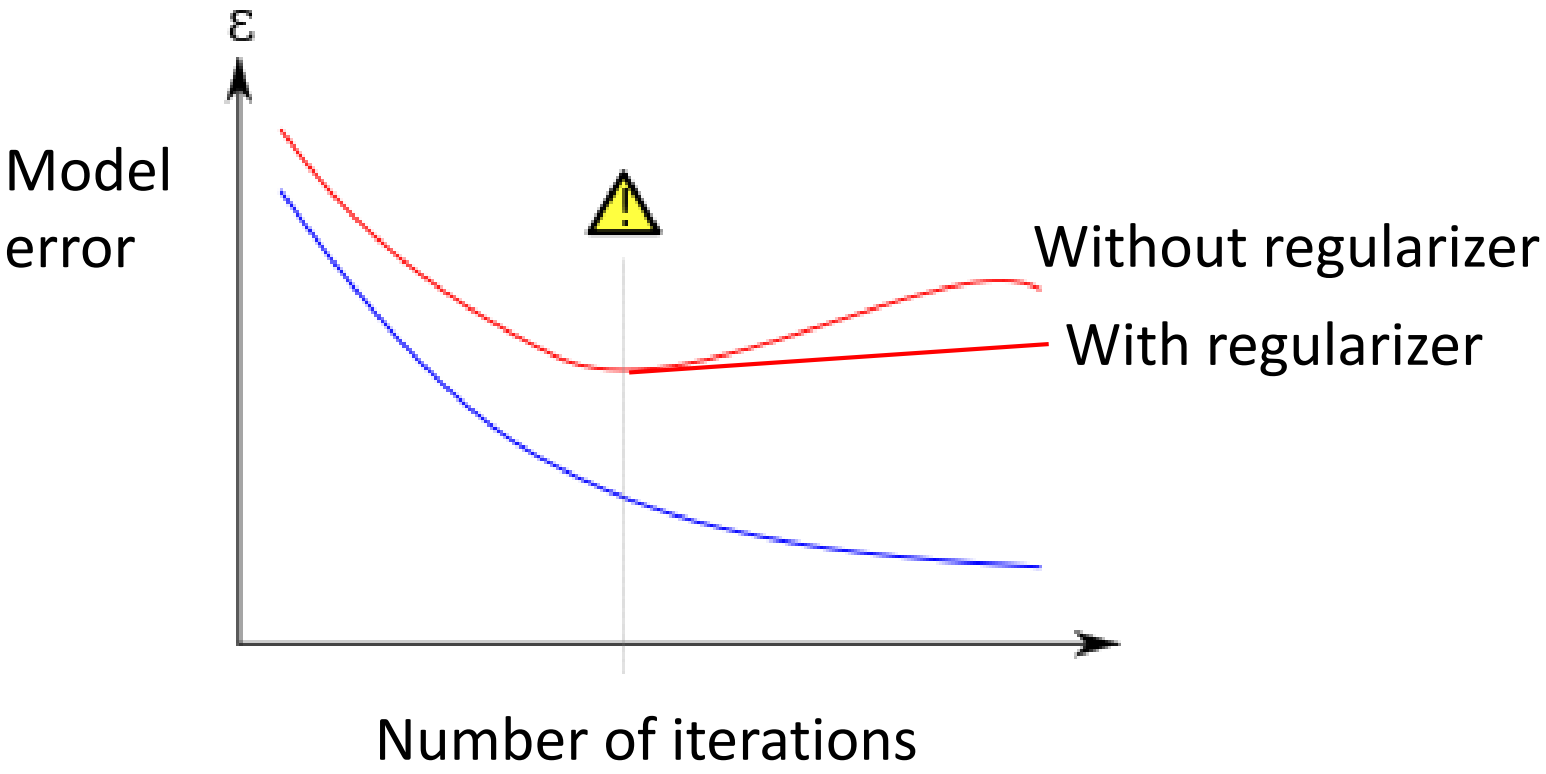
Over-fitting

Another kind of over-fitting



Regularization and Over-fitting

Adding a regularizer:



Cross-Validation

- Cross-validation involves **partitioning** your data into distinct **training** and **test** subsets.

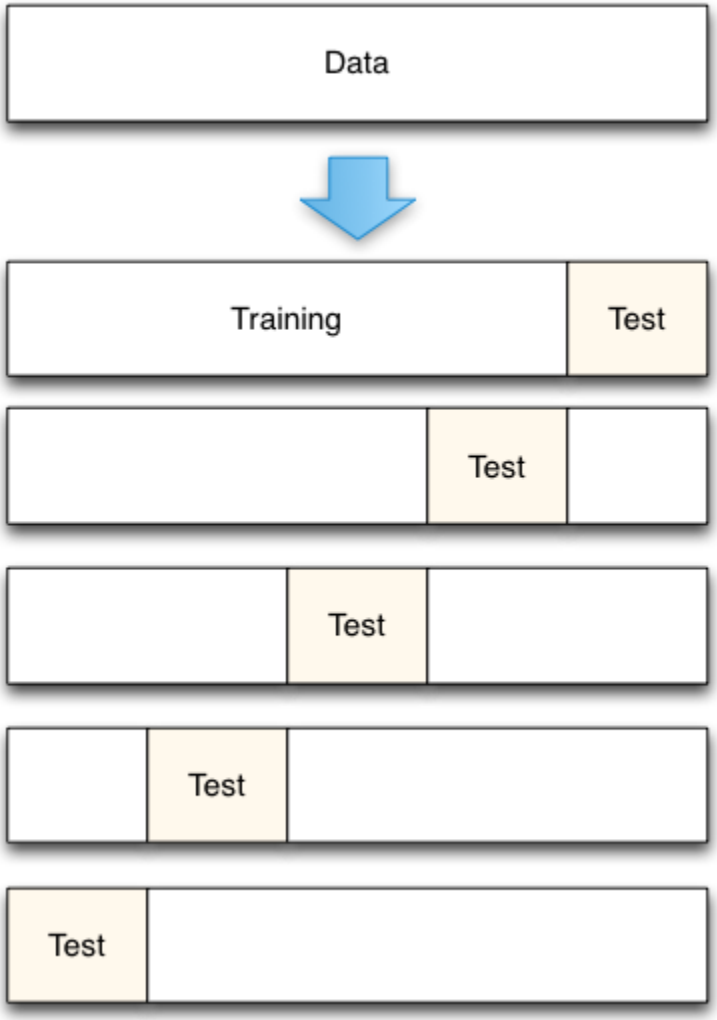
- The test set **should never** be used to **train** the model.

- The test set is then used to **evaluate** the model after training.

K-fold Cross-Validation

- To get more accurate estimates of performance you can do this k times.
- Break the data into k equal-sized subsets A_i
- For each i in 1,…,k do:
  - Train a model on all the other folds A_1,…, A_{i-1}, A_{i+1},…, A_k
  - Test the model on A_i
- Compute the **average performance** of the k runs

5-fold Cross-Validation



Summary

- Three Basic Algorithms
  - kNN
  - Linear Regression
  - K-Means
- Training Issues
  - Measuring model quality
  - Over-fitting
  - Cross-validation

References

- 1. Supply Chain 4.0: Improving Supply Chains with Analytics and Industry 4.0 Technologies 1st Edition by Dr Emel Aktas (Author), Professor Michael Bourlakis (Author), Ioannis Minis (Author), Vasileios Zeimpekis (Author)
- 2. Supply Chain 4.0: From Stocking Shelves to Running the World Fuelled by Industry 4.0 – April 28, 2018 by Alasdair Gilchrist (Author)
- 3. Supply Chain Analytics: Using Data to Optimise Supply Chain Processes 1st Edition by Peter W. Robertson (Author)
- 4. Networks Against Time: Supply Chain Analytics for Perishable Products (SpringerBriefs in Optimization) 2013th Edition by by Anna Nagurney (Author), Min Yu (Author), Amir H. Masoumi (Author), Ladimer S. Nagurney (Author)
- 5. Data Science for Supply Chain Forecasting – March 22, 2021 by Nicolas Vandeput (Author)
- 6. Inventory Analytics: Prescriptive Analytics in Supply Chains – June 2, 2020 by Horst Tempelmeier (Author)