# Stats for Data Science

Saumya Bhatnagar

February 14, 2020

# Table of contents

# Regression, Classification, Clustering

**Regression**

1. Linear
2. KNN
3. SVM
4. Random Forest

**Classification**

1. Logistic
2. KNN
3. SVM Classifier
4. Random Forest

**Clustering**

1. K-Means
2. Hierarchical
3. DBSCAN
4. HDBSCAN

Regression analysis is a statistical technique to assess the relationship between an predictor variable and one or more response factors.

| **Outcome Variable** | **GLM Family** | **Link** | **Mean to Variance** |
|---|---|---|---|
| Continuous, unbounded | Normal or Standard Gaussian | Identity | |
| Continuous, non-negative | Gamma or inverse Gamma | | |
| Discrete/ counts/ rate | Poisson Quassi-poisson or negative binomial | Log | Identity If not Identity |
| Count | Gamma | | Over dispersion |
| Counts with multiple zero | Zero inflated poisson may be checked for fitting | | |
| Binary | Binomial or Logistic regression | | |
| Nominal | Multinomial regression | | |

Regression Model Selection Criteria

Three methods to classifier

1. model a classification rule - knn, decision tree, perceptron, svm
2. model the probability of class membership given input data - perceptron with cross-entropy cost
3. make a probabilistic model of data within each class - naive bayes 1 & 2 are discriminative classifications 3 is generative classification 2 & 3 probabilistic classification

# Decision Tree

content...

# Random Forest

content...

“Help me understand our customers better so that we can market our products to them in a better manner!

**Monothetic**: Cluster members have some common property

Expectation–Maximization (EM) Clustering using Gaussian Mixture Models (GMM)

**Polythetic**: Cluster members are similar to each other. Distance between elements define relationship

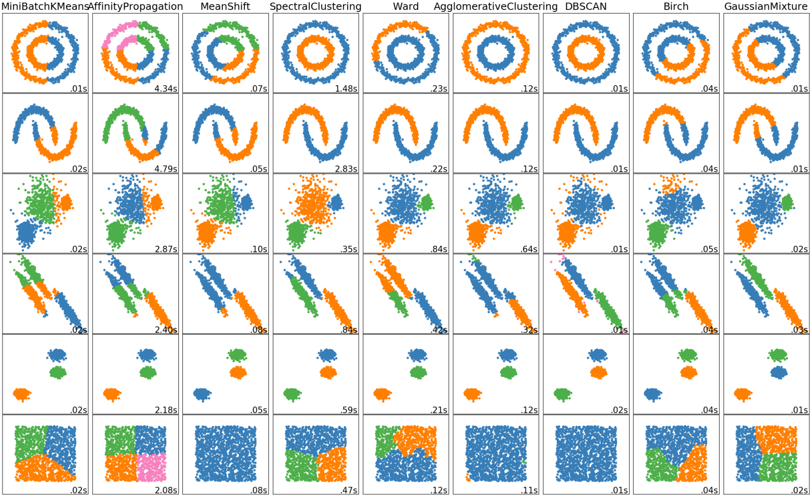
**Hard Clustering**: each data point either belongs to a cluster completely or not

**Soft Clustering**: a probability or likelihood of that data point to be in those clusters is assigned.

# Clustering Models

| Connectivity models | Distribution models | Centroid models | Density models |
|---|---|---|---|
| data points closer in data space exhibit more similarity to each other than the data points lying farther away | how probable is it that all data points in the cluster belong to the same distribution (e.g: Normal, Gaussian) | iterative clustering algorithms in which the notion of similarity is derived by the closeness of a data point to the centroid of the clusters | isolates various different density regions and assign the data points within these regions in the same cluster |
| hierarchical clustering | Expectation-maximization | K-Means, k-median | mean-shift, DBSCAN and OPTICS |
| Approaches: 1) Top-bottom, 2) bottom-up | EM uses multivariate normal distributions | DZA | DBSCAN uses radius $\epsilon$ and Center c |
| lacks scalability for handling big datasets, Time complexity: O($n^2$) | These models often suffer from over-fitting. Prior knowledge to define num clusters | important to have prior knowledge of the dataset. results change in every trial | DBSCAN doesn't perform as well when the clusters are of varying density |
| Results are reproducible | more flexibility in terms of cluster covariance due to $\mu and \sigma$ (additional $\sigma$) | can handle big data , Time complexity: O(n) | DBSCAN identifies outliers as noises |
| chk1 | elliptical shape (since we have a standard deviation in both the x and y directions | work well when the shape of the clusters is hyper spherical (like circle in 2D, sphere in 3D) | DBSCAN: can find arbitrarily sized and arbitrarily shaped clusters |
| Angola | GMMs support mixed membership since is probability based | AGO | DBSCAN: drawback in high-dimensional data since the distance threshold $\epsilon$ becomes challenging to estimate |

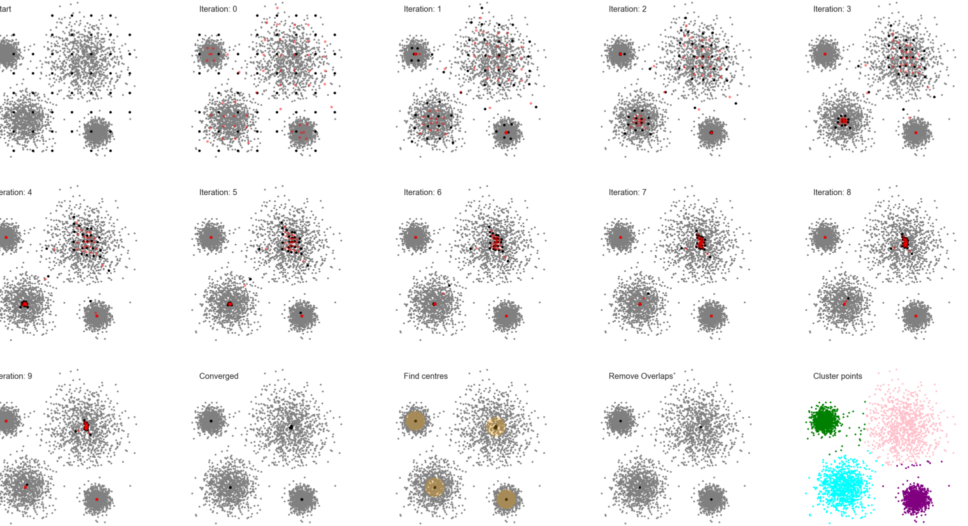
## Clustering Types



# mean-shift clustering

consider a set of points in two-dimensional space
a circular sliding window C centered and radius r as the kernel
hill-climbing algorithm that involves shifting this kernel iteratively to a higher density ($\propto$ number of points) region until convergence
At every iteration,
- shift the center point to the mean of the points within the window (hence the name)
-gradually move towards areas of higher point density
- until no longer increase in the density
- When multiple sliding windows overlap the window containing the most points is preserved. The data points are then clustered according to the sliding window in which they reside.

## Density Models



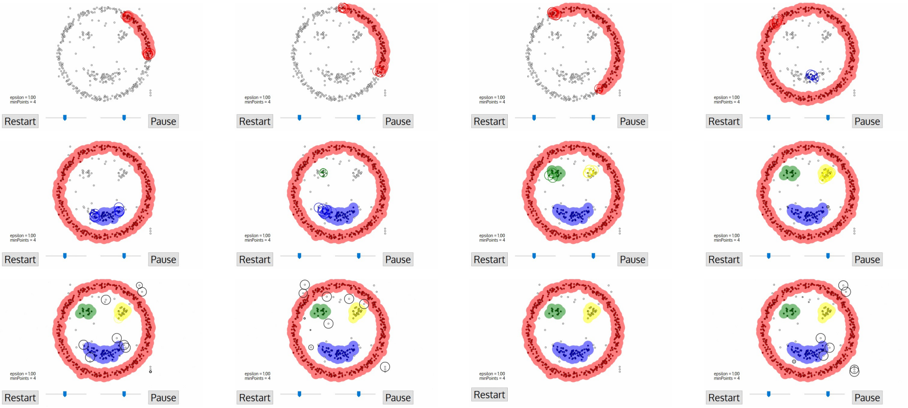
# Density-Based Spatial Clustering of Applications with Noise-DBSCAN

-label all data point to be unvisited. For all unvisited points:

1. All points which are within the $\epsilon$ distance are neighborhood points (part of the same cluster)
2. If neighborhood points ¿= minPoints, then the clustering process starts and the current data point becomes the first point in the new cluster - Otherwise, mark the point as noise - In both cases that point is marked as "visited"
3. repeated for all of the new points in the cluster group
4. next an new unvisited point is retrieved and processed

Since at the end of this all points have been visited, each point will have been marked as either belonging to a cluster or being noise.

## Density Models



# Hierarchical DBSCAN - HDBSCAN

content...

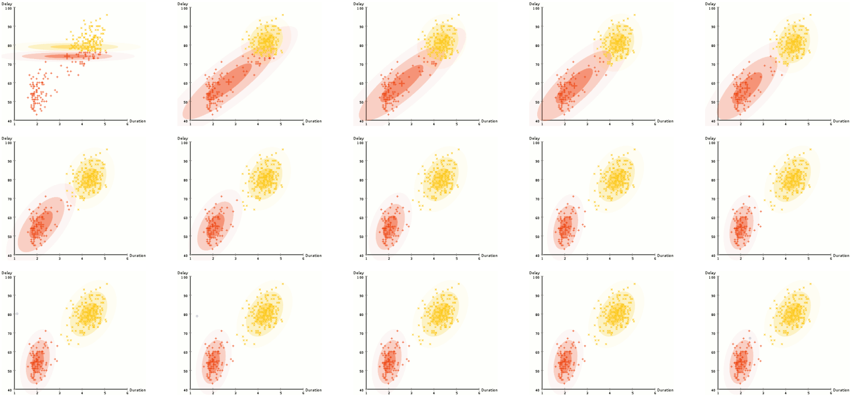
# Gaussian Mixture Models (GMMs)

Assumption: the data points are Gaussian distributed (parameters: the mean and the standard deviation)! Each Gaussian distribution is assigned to a single cluster. To find the parameters of the Gaussian for each cluster, use an optimization algorithm called Expectation–Maximization (EM).

# Expectation–Maximization (EM) using GMM

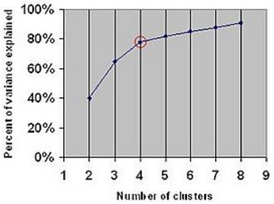
choose num of clusters
compute the probability that each data point belongs to a particular cluster. With a Gaussian distribution we are assuming that most of the data lies closer to the center of the cluster.
From probabilities $\rightarrow$ recompute set of parameters such that we maximize the probabilities of data points within the clusters
We compute these new parameters using a weighted sum of the data point positions, where the weights are the probabilities of the data point belonging in that particular cluster.
Repeat till convergence

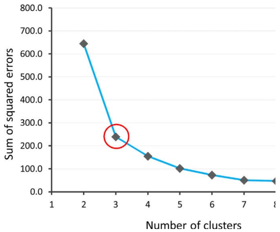
## Distribution Models



# K means

iterative clustering algorithm that aims to find local maxima in each iteration
take a quick look at the data and choose k (num clusters)
assign data points to the cluster $\leftrightarrow$ compute cluster centroid
repeat to reduce variation error
elbow method



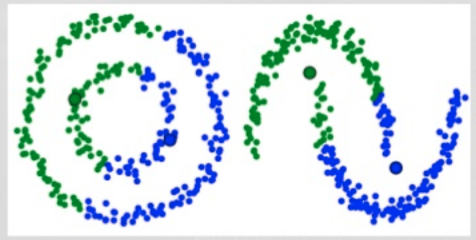



# k-median

**K median vs kmeans**: instead of recomputing the group center points using the mean (like in K-Means) we use the median vector of the group. This method is **less sensitive to outliers** (because of using the Median) but is **much slower for larger datasets as sorting is required** on each iteration when computing the Median vector

**mean-shift vs kmeans**: Instead of selecting the number of clusters as mean-shift automatically discovers this (advantage), the selection of the window size/radius "r" can be non-trivial.

# kmeans fail



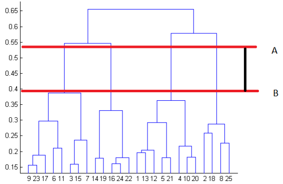
K-Means is actually a special case of GMM in which each cluster's covariance along all dimensions approaches 0

# Agglomerative Hierarchical Clustering

The decision of dividing into or merging **two** clusters is taken on the basis of closeness of these clusters. Metrics for deciding the closeness of two clusters:
Euclidean distance: $||a - b||_2 = \sqrt{\sum(a_i - b_i)}$
Squared Euclidean distance: $||a - b||_2^2 = \sum(a_i - b_i)^2$
Manhattan distance: $||a - b||_1 = \sum |a_i - b_i|$
Maximum distance: $||a - b||_{INFINITY} = \max_i |a_i - b_i|$
Mahalanobis distance: $\sqrt{(a - b)^T S^{-1}(-b)}$
Maybe, use **average linkage** which defines the distance between two clusters to be the average distance between data points in the first cluster and data points in the second cluster.

# hierarchical agglomerative clustering (HAC) or bottom-up

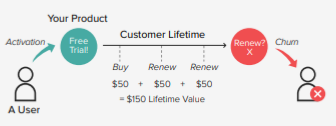
1. Each data point as a single cluster
2. select a distance metric
3. Iterate till convergence
   - combine two clusters with the smallest average linkage




The height in the dendrogram at which two clusters are merged represents the distance between two clusters in the data space. take 4 clusters as the red horizontal line in the dendrogram covers maximum vertical distance AB.

# Churn or Retention Analysis

Customer Retention Rate: The percentage of customers who repurchase in a given time period compared to an equal and preceding time period
Churn Rate: The inverse of Customer Retention Rate, or the percent of users who did not repurchase or whom you lost




**proactive churns**: losing customers due to cancellations
**passive churn**: failures to renew

# Cohort Analysis and Life Time Value (LTV)

**LTV**: The expected amount of profit/revenue from a user
CLV = NPV (net present value) of the sum of all future revenues from a customer, minus all costs associated with that customer
**Why LTV**: - Tracking your LTV to Customer Acquisition Cost (CAC) ratio: Companies typically use the 3:1 CAC ratio or Cost Per Acquisition (CPA)
- Evaluating your most valuable marketing channels
- Focus on retaining your most valuable customers
**Historic CLV**: sum of the gross profit from all historic purchases for an individual customer

| | |
|---|---|
| Avg Order Value, AOV $= \frac{Revenue}{Orders}$ ; | Avg Purchase Rate $= \frac{Orders}{NumCustomers}$ |
| Avg Customer Value $= \frac{AvgPurchaseVal}{AvgPurchaseRate}$ <br> Avg Customer Lifespan $= \frac{SumCustomerLifespans}{NumCustomers}$ | gives LTV |
| LTV = | Avg Customer Value X Avg Customer Lifespan <br> ARPU X $\sum_{n=0}^{N}(1 - CR)^t$ .... [N=num months to examine] <br> ARPU/$CR_n$ ...[ARPU=Avg rev per User, for n months] <br> ARPU/$CR_n$ x DR ......... [for variable churn & n months] <br> ASP/CR + m(1-CR)/$CR^2$ ... [for account expansion] |
| CLV = | AGM x $\sum_0^{numTransactions} Transaction$ [AGM=Avg Gross Margin] <br> $((T_{avg} xAOV)$AGM)ALT =GML(gross margin per user lifespan) <br> GML(R/(1+D-R)); [account expansion] |

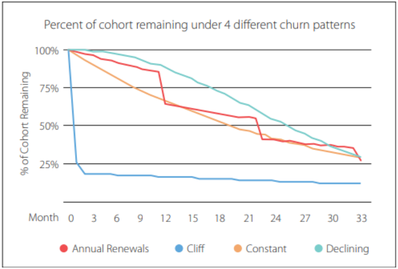
CR=Churn rate; ASP=Avg Selling Price; m=↑ARPU/user/month;
T_avg = avg monthly transactions; ALT=avg User Lifespan (in months)
D=monthly discount rate; R=monthly retention rate; DR=Discount Rate
to adjusts for mix churn (Annual Renewals, Constant, Declining and Cliff

Steps to LTV:

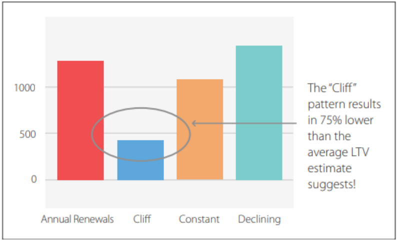
- Normalizing to Acquisition Date: Bin users into buckets like Day 0, Day 1, Day 2 or Week 0, Week 1, Week 2, and so on.
- Normalized to a Closed Time Limit: Broader questions "What is my total CLTV," should be replaced with "What is our 3-year or 5-year LTV?" & should be based on:
  1. Average Customer Lifespan
  2. Customer Retention Rate
  3. Churn Rate: The inverse of Customer Retention Rate.
  4. Time to General Profitability Against Acquisition Costs: If your business is a "Loss Leader" Model this time may be a longer length than businesses with lower acquisition costs and lower profitability.
  5. Rate of Discount

# Types of churn and LTV

- Annual Renewals: larger churn at each contract renewal.
- Cliff churn: majority of the churn within the first month, and then a small constant churn thereafter.
- Constant: steady, constant churn rate (shown as 3.5%).
- Declining: churn rate starts at zero, increases each month.




Effect on LTV:



# Survival Analysis

content...

# Sentiment Analysis

content...

# Propensity of Cross-sell

content...

# Whale Curve Analysis

**Pareto Principle**: for many events, roughly 80% of the effects come from 20% of the causes

# "Loss Leader" Model

"Loss Leader" Model, where you introduce new customers at a high cost in the hope of building a customer base or securing future revenue?

Thank You!