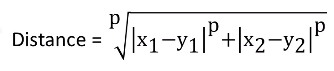
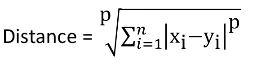
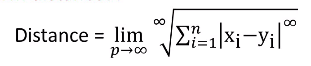
***Clustering***

* **Clustering** = taking a set of data points + dividing them into groups containing points that are similar to one another/close to each other
* Clustering helps segment customers into expected groups AND discover new, unexpected clusters
* Useful for:
* Targeted marketing/market segmentation **=** segment potential customers + give different ads to different groups to send a message most likely to get them to buy something
* Market size, versatility, or price of an SUV to different people
* May discover clusters of people who care about MPG, which you may not have expected
* Personalized medicine -> people who react similarly to a drug, and people who have specific reactions to it
* Locating areas for facilities🡪 finding people live + clustering them by that and placing police + fire stations, as well as medical clinics + libraries, in those area
* Can do the same for gas station, restaurants, coffee shop,
* Image analysis 🡪 recoloring pictures, recognizing faces + other objects
* Ex: clustering algorithms are getting better + better at recognizing CAPTCHA, which is why they’re getting more ambiguous
* Exploratory data analysis 🡪 if we can find obvious clusters of data, we can analyze each separately in case they have different properties
* Ex: Predicting how much customers will pay for an SUV may require different models depending on which attribute the customer cares about the most

*Distance Norms*

* **Euclidian (Straight-line) Distance**  
* **Rectilinear Distance** = 
* Sometime used when driving in a city w/ streets laid in a grid
* Also called **“Manhattan Distance”**
* Can generalize these both into the **p-norm/Minkowski distance** 
* P = 2 for straight-line distance and 1 for rectilinear
* In a space w/ n dimensions, we write this as 
* *Could choose any p value, but the 3rd most common after 1 and 2 is infinity* = **infinity-norm distance**
* Has a significant meaning

= 

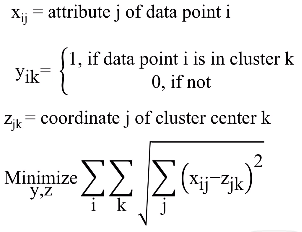
* So if we have n different #’s all to the infinity power, the largest difference will dominate the rest + the sum

 *=* 

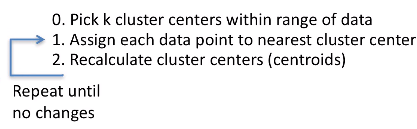
* We have the infinity-ith root of the infinity-ith power of the largest difference b/c the root and the power are the same so they cancel + we’re left w/ just the largest difference
* Therefore, ALL the infinity-norm is = this largest difference
* 
* Why use the infinity norm as a distance metric?
* Ex: Warehouse w/ automatic storage + retrieval system
* In each aisle, a machine moves up + down it + stretches up + down at the right area to reach the right height to get an object
* Total time = horizontal aisle time + vertical height time = *1-norm distance*
* But if the machine was designed in a smarter way, as it moves down the aisle, it *simultaneously* stretches up + down to the right height
* Now the total time is whichever is longest 🡪 horizontal or vertical travel time = infinity-norm

*K-means Clustering*

* Ex: SUV buyer data for past 8 years + want to define groups of them via a 2D graph
* For each data point, x = age, y = average daily temperate of the city data point lives in
* We have n cases + m attributes 🡪 x(ij) = the j attribute of the ith person
* Y is used to denote cluster membership 🡪 y(ik) = 1 if ith person is in cluster k and 0 if not
* z(kj) = the j dimension coordinate (jth dimension/attribute ) of cluster center k
* We want a set of k cluster centers + assignments of each data point to a cluster center w/ a minimized distance from each data point to its cluster center



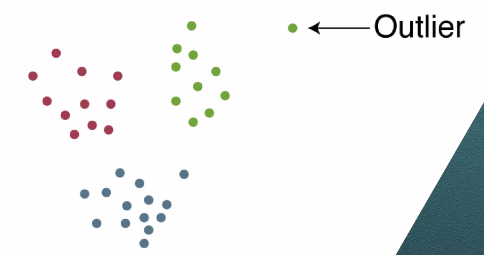
* We are taking the square root of all the distances from data points to cluster centers (when that data point is in that cluster)
* Also, every data point must only be assigned to ONLY 1 cluster = 
* This is in fact a hard optimization problem to solve, so we use the k-means algorithm
* 1) decide how many clusters we want the algorithm to result in (ex: 3)
* 2) pick k (the # of clusters we want) points inside the range of data 🡪 our **cluster centers**
* 3) temporarily assign each data point to the cluster center closes to it = 3 clusters
* 4) now original cluster centers are NOT in the center of the cluster 🡪 recalculate
* 5) find **centroid** of the data points w/in each cluster = *new* cluster center
* 6) moved cluster centers = data points may be in the wrong cluster now 🡪 reassign data points to cluster centers that are closest to them after this recalculation of the cluster centers
* 7) Repeat this over and over until we get to step where no data points change clusters, and therefore cluster centers will not change either



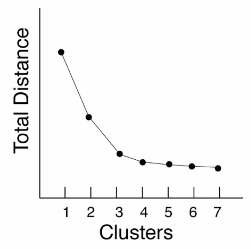
* This is a **heuristic machine learning algorithm** = *NOT* guaranteed to find the absolute best solution, but in many cases, gets very close to it + gets there quickly
* Trying to find the absolute best may take too long
* It’s also an **expectation-maximization algorithm** = g
* When calculating cluster centers, we’re taking the man of all points in the cluster, similar to finding an expectation
* When we reassign data points to new clusters, this is the maximizing step
* We’re actually minimizing the smallest distance to the cluster center, but think of it as maximizing the negative distance to the cluster center
* Expectation 🡪 maximize 🡪 expectation 🡪 maximize, etc.

*Practical Details for K-Means*

* What should we do w/ outliers? They will be assigned to a cluster by the k-means algorithm, but they really aren’t a part of a cluster:



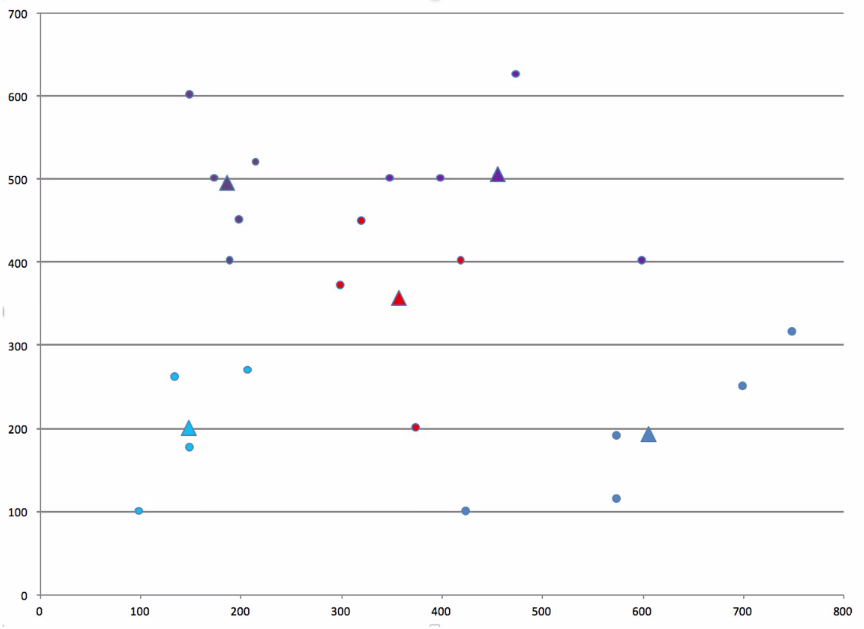
* Could remove it + re-run the algorithm so the outlier doesn’t drag its cluster center artificially to 1 side
* Better way 🡪 find out more about the outlier + what is means for our specific problem
* What makes it’s attributes so different? What’s the implication of putting this point in the nearest cluster?
* Answering these questions takes time + effort but it’s very important + part of what makes the difference between a top-notch analyst and a someone who blindly runs algorithms
* Extra value given by doing extra leg work can be significant
* There are MANY situations in analytics where the *algorithm is just a guide*
* Understand a situation helps us make more appropriate + valuable decisions
* Another issue w/ k-means:
* Remember it’s heuristic (fast + good but not always best clustering)
* Can take advantage of its speed by running the algorithm several times w/ each iteration having different k’s (initial cluster centers) + then choosing the best solution
* Can also use its speed to choose the right # of clusters to use instead of guessing by running k-means for different values of k
* More clusters is NOT always better than less
* If we add a cluster, even if it’s just 1 point, then the data pints in each cluster will be closer to their eventual cluster center 🡪 more minimized total distance of all data points to their cluster centers
* Most extreme case = 1 cluster for each data point where the data points is its own cluster center
* Best solution = the one that best fits the specific situation
* Ex: city only has budget for 4 new fire stations, only want 4 location/area clusters
* Quantitative to guide this decision (b/c seeing the clusters is impossible due to large # of attributes )
* Suppose we find k-means clusters for many different values of k
* For each, calculate total distance of all data points to their cluster centers
* Plot *this* in 2D w/ x = # of closers + y = total distance 🡪 **elbow diagram**



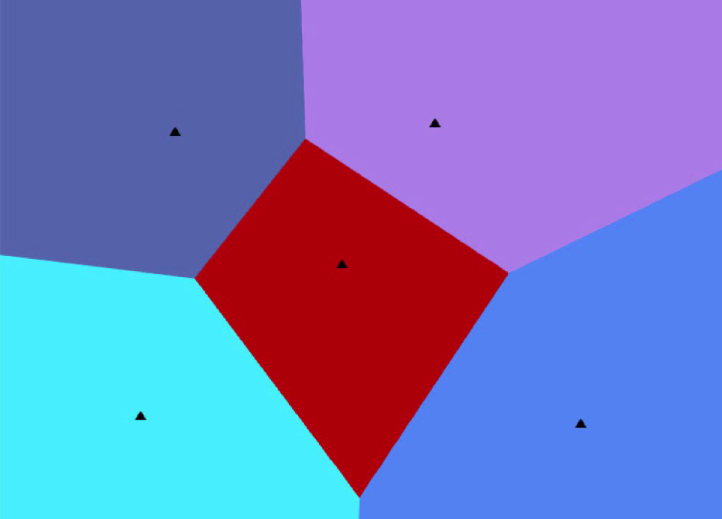
* Look to see where “kink” in the curve is = 3 🡪 this is where benefit of adding another cluster becomes marginal + exponentially smaller
* Elbow diagrams can be initially helpful as a starting point in deciding how many clusters to use, but also remember to think about the qualitative aspects of this problem as well

*Clustering for Prediction*

* Ex: set of data points divided into 5 clusters where triangles are cluster centers

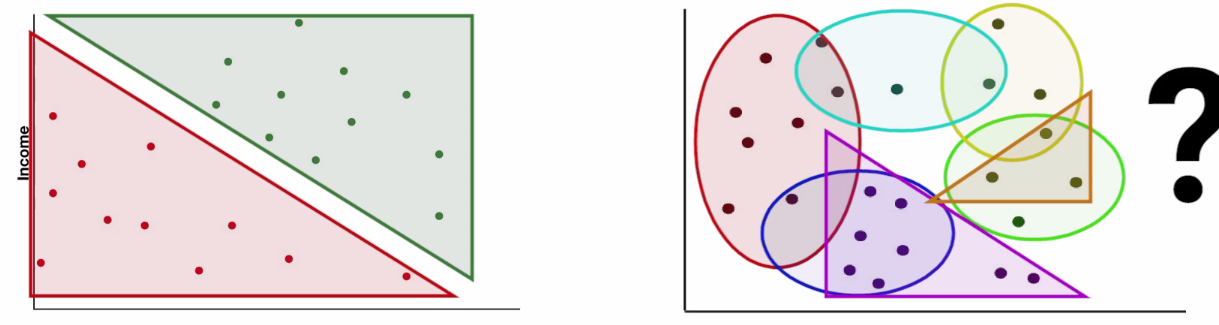


* Want to add a new data point, so which cluster should we predict it will be in?
* If the new point is *inside* an already define cluster, that’s easy
* If not, can choose the one w/ the closes center
* Can also answer “what range of possible data points would we assign to each cluster”?
* Each cluster could possible include all data pints closer to its center than any other center
* **Voronoi Diagram**



*Supervised vs Unsupervised Learning*

* In **classification,** we *knew* each data point’s attributes and *knew* the right classification
* Model uses both the attributes + response of each data point + this info helps us to decide how to classify a new data point(s) = **supervised learning**
* In clustering, we *know* the attributes of our data points but do NOT know the right grouping of them
* The *model* must decide how to cluster based *only on the attributes* = **unsupervised learning**



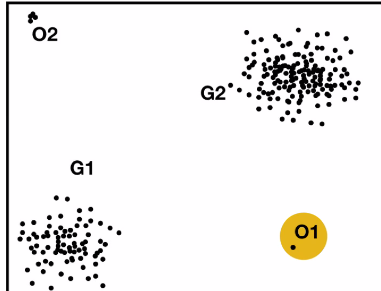
***Basic Data Prep***

*Intro to Data Prep*

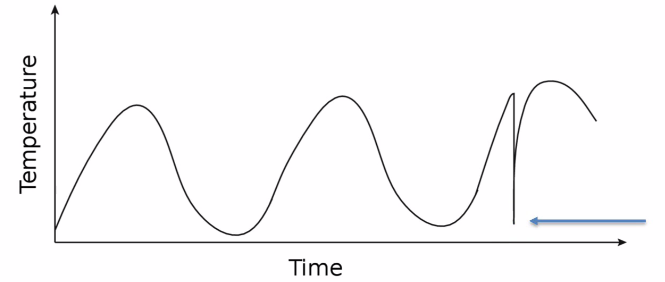
* Many times when talking about the data being used in analysis (whether **predictors** for regression or **factors** for classification etc.), it’s glossed over that a majority of the time data needs to be manipulated and many times it also needs to be **scaled**
* Avg. household income is usually 3+ orders of magnitude larger than credit score, so to use credit score to predict it would be bad w/out first scaling the data
* Could have 1 or 2 DPs whose effect on the whole population is magnified or of proportion b/c of how different they are from the rest (**outliers**)
* Sometimes data has a lot of extraneous info in it that could complicate a model + our ability to correctly interpret a solution
* All these potential problems can and should be corrected before building + solving models

*Outlier Detection*

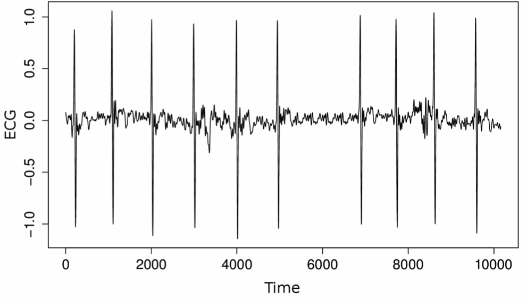
* **Outliers** = DPs very different from the rest of the data set
* What to do w/ them depends on the situation 🡪 both philosophical and statistical issues
* Most people think of a **point outlier** when hearing this term



* But maybe O2 above are really not outliers, and there’s a good reason for them being there
* **Contextual outliers** (relies on context of other points 🡪 Ex: Time-series outliers



* Value itself isn’t an outlier b/c it’s not far from the rest overall, but *the time it occurred* makes it one b/c it’s far from points near in time
* **Outlier by omission/** 🡪 Ex: Time-series outliers



* Missing heart beat around 6000 ms 🡪 hard to tell *which* specific data point is wrong, but some DP between 5000-7000 ms is off
* Can also be seen as **a Collective outlier** b/c *something* is missing in a range of points, but we can’t tell *exactly where* 🡪 data points are collectively seen as an outlier
* Can find them in multiple ways
* Ex: **box and whisker plot** helps find outliers in 1-dimension 🡪 outlier < IQR – Q1 or > IQR + Q3
* No good *all-purpose* way of detecting all types outliers, especially multidimensional outliers
* Could build the model, fit the parameters, and then see which points have a lot of error
* Can fit an exponential smoothing model to the temperature time series data, and whichever points has a very large error could be considered outlier, b/c the model would expect a certain value at certain points and this DP differs greatly
* Once we find an outlier, we have to decide what to do w/ it

*Dealing W/ Outliers*

* To decide what to do w/ outliers, find out what caused it/them
* Many times, it’s just bad data (failed sensor, contaminated experiment, bad data input):
* just remove it
* use **imputation** if we want to use the DP + want some new factor to replace the erroneous one
* Sometimes it *is real data* + we need to decide if our model will consider it or not via investigation:
* Where data came from, how it was compiled, are there unique situations for outlier DPs
* Real outliers commonly occur in very large data sets b/c enough true randomness will occur
* i.e. normal distribution 🡪 ~4% of data is outside 2 SD’s from expected mean (w/ 1M DP’s, 2000+ DPs will be more than 3 SD’s away from the expected value (mean)
* Need to consider what is going on + what we’re modeling carefully
* It is important to consider these points in our model?
* If measure/magnitude of model’s error is a measure of its value, may want to keep some outliers
* Ex: modeling time to transport perishable medicine from manufacturer in US to Africa
* Will sometimes have some impediments (weather, political issues) that could take a long time to resolve
* Considering outlying data points will help consider things that really occur
* Throwing these away makes model too optimistic + reality gives bad surprises
* Could build 1 model to estimate probability of outliers happening under different conditions, like logistic regression
* Then build model w/ removed outliers to see how long delivery time would be under “normal” conditions
* Sometimes we find outliers just aren’t predictable at all + may need to be removed even if they’re real data
* Ex: 1 time event of very high sales for Chick-fil-a protests + a larger counter-protest