Tiffany Jaya's ML notes
Last updated: 6/30/2018

Problem Setup

1. **All model is wrong but useful to make predictions**.
2. **Intercept (aka bias in ML)** tells you featureless version of the model that just predicts the mean. It is the **baseline** for linear model.
3. **R^2 t**ells you what % is explained by the model.
    a. With more features (aka params), you can get higher R^2. For that reason, you need to use test set to double check that the model is a good model if you found that the R^2 is high on the training set.
4. What are the different data sets used for:
    a. Train with **training set**
    b. Look at results with **dev set** to analyze errors
        i. Also, you can use the dev set to choose hyperparameters: parameters you tweak that can affect the outcome
    c. Run once on the **test set** to ensure model is a good model
5. To separate the different data sets, you apply **cross validation** (split single set of data into training and test sets in many different way if you have a small data set).
    a. Types of cross validation:
        i. Jack-knife: splits data into training set and test set
        ii. Leave one out: use all of the data for training except for one
        iii. Randomized splits (recommended): jack knife split with random partitions, can include dev data

Nearest Neighbors [[Classification]]

aka: instance-based, nonparametric modeling
Child of: kernel (weighting scheme) density estimation (approximation of data with a function)
Hyperparameter: k, distance (aka similarity measure, kernel measure in ML)

1. **Output differs based on the distance metrics you choose.**
2. What it does? Nearest neighbors model **predicts the same label as the nearest neighbor in the training data**.
3. Different **types of distance metrics**:
    a. **Ln-norm**: you care about the **size of the difference**

$$L^n(x_1, x_2) = \sqrt[n]{\sum_{i=1}^{\#\text{dim}} |x_{l,i} - x_{2,i}|^n}$$

        i. n = 0: **Hamming distance** (categorical)
        Measure things that do not overlap between two vectors

ii. n = 1: **Manhattan distance** (numerical)
Basically it adds up the absolute value of the distance difference
i.e. used **mostly in board games with discrete moves**

iii. n = 2: **Euclidean distance** (numerical)
Compare any given point to all other points
i.e. used more broadly

Pros	Cons
Symmetrical, spherical, treats all dimensions equally	**Very sensitive to deviation** even if it's just a single attribute

b. **Cosine similarity**: you don't care about the size of the difference but you **care about the angle** in between them
i.e. used mostly during **information retrieval in search engines**

$$\frac{A \cdot B}{\|A\| \ \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} \left(A_i\right)^2} \times \sqrt{\sum_{i=1}^{n} \left(B_i\right)^2}}$$

4. **Curse of dimensionality**: adding dimensions increases feature space exponentially, and statistical problems arise as a result. This is a problem for k-nearest neighbors.
   a. How to **fix** curse of dimensionality?
      i. **Normalize numeric values** by setting mean = 0, variance = 1
      ii. **Feature selection**: choose a subset of features
         1. Forward selection: choose the single best feature first
         2. Backward selection: remove the least important features first
5. Types of k-nearest neighbor
   a. Weighted k-nearest neighbor
   Weight each label by its similarity (which is the inverse distance)
      i. Why do weighted k-nearest neighbor?
         1. You don't need to specify k anymore, but on the negative side, it can slow down computations because you're measuring all the distance.
         2. Unweighted voting will be biased by a non-uniform distribution of training examples; in other words, if there are more of group A than group B in the sample data but not in reality, unweighted voting means that it will be biased towards group A.
   b. Edited k-nearest neighbor
   Remove outliers if all neighbors are in a different class
      i. Why do edited k-nearest neighbor? Reduce impact of outliers (noise) in the training set

c. Prototypes
   Remove training sample that does not contribute much to the training boundary
d. Coarse-to-fine
   Use simplified feature set to choose candidate neighbors.
   Then, use the full feature set to perform the final distance measurements.
e. Neighbor graph (similar to binary search for a graph)
   Create a KD-tree for partitioning feature space

6. Assumptions that similarity metric (or distance metric) fulfill:
   (unlike similarity measure which is not mathematically defined)
   a. d(x, y) >= 0: non-negativity or separation axiom
      Distance is not negative
   b. d(x, y) = 0 if and only if x = y: coincidence axiom
      It's the same data point
   c. d(x, y) = d(y, x): symmetry
      Distance is symmetric
   d. d(x, z) <= d(x, y) + d(y, z): subadditivity / triangle inequality
      If everything is fulfilled except d, then you label it as semi-metric instead of similarity metric. i.e. networks: not all your friends' friends are your friends

When do you want to use Nearest Neighbor:

- Small dataset or really massive dataset

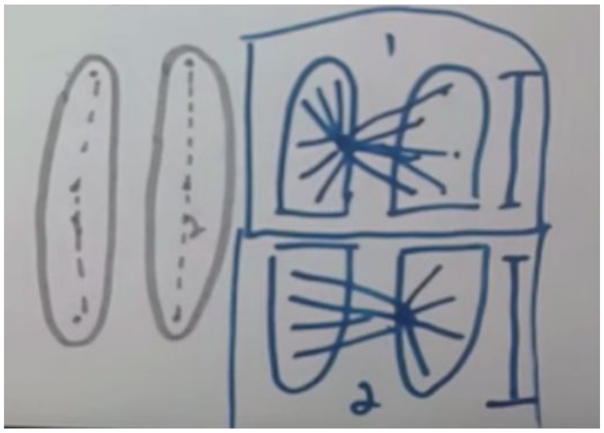
Nearest Neighbors Pros and Cons list:

Pros	Cons
No training step. Therefore, training and testing are the same thing and you can use this during runtime or online fashion.	Load all training into memory. Therefore, the larger the dataset, the slower it runs and the more memory it takes up.
Simple and straightforward up to gigabyte of data.	Curse of dimensionality, meaning it is difficult for K-nearest neighbors to classify if there are multiple features (aka multiple dimensions).

K-means [[Clustering]]

1. How do you **select the k value** for number of groups?
   a. **You don't want k to be an even number** so you don't get stuck with an even vote; you don't want to feel divided between the groups
   b. Example:
      i. If num_group = 2, then k = 3

ii. If num_group = 3, then k = 5

Pros	Cons
No training step. Therefore, training and testing are the same thing and you can use this during runtime or online fashion.	Load all training into memory. Therefore, the larger the dataset, the slower it runs and the more memory it takes up.
It can be calculated in parallel.	You have to specify ahead of time the number of k clusters. Therefore, you can draw wrong conclusions.
	If the data contain two clusters that are close to one another, k-means might incorrectly identify the cluster because it tries to find cluster that minimize the distance. For example, actual is specified by gray and k-means is specified by blue. 
	Curse of dimensionality, meaning it is difficult for K-nearest neighbors to classify if there are multiple features (aka multiple dimensions).

To fix these problem, use affinity-based clustering, also known as spectral clustering.

Naive Bayes

1.

Datasets

archive.ics.uci.edu/ml/datasets.html

References

- https://www.youtube.com/channel/UCUcpVoi5KkJmnE3bvEhHR0Q
- https://www.youtube.com/user/joshstarmer
- https://www.youtube.com/user/sentdex