NYC DATA SCIENCE ACADEMY

KNN & Naive Bayes

Data Science with R: Machine Learning

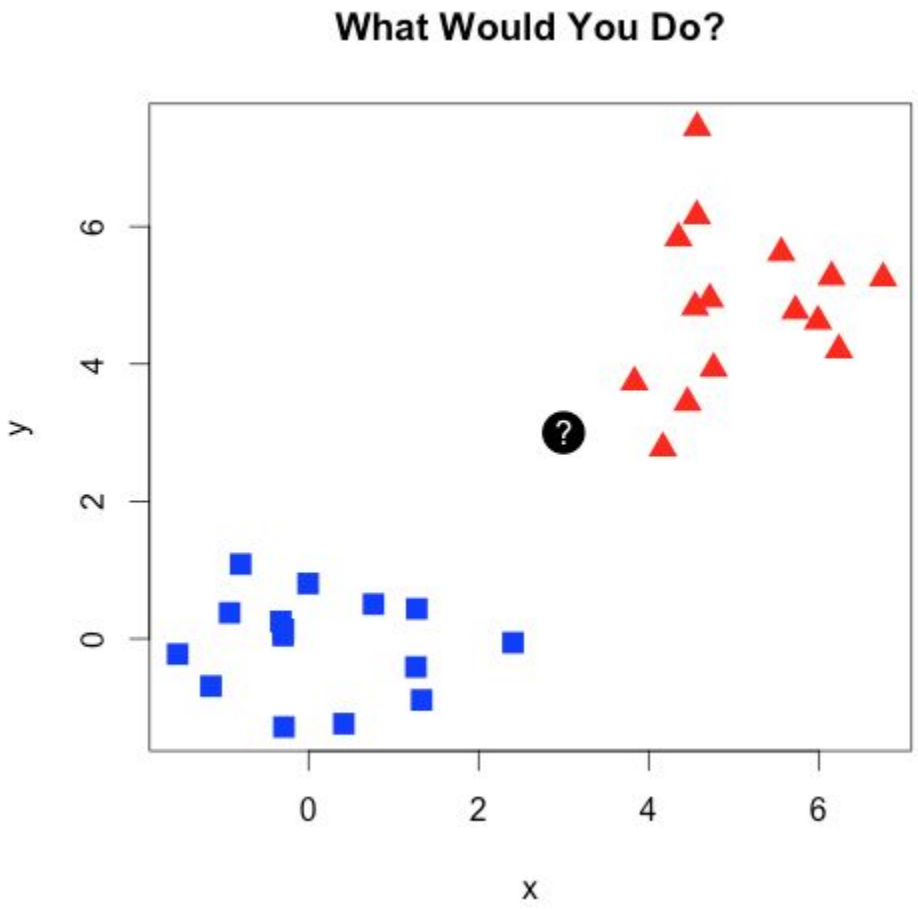
Outline

- Part 1: K-Nearest Neighbors
- Part 2: Naive Bayes
- Part 3: Review

PART 1

K-Nearest Neighbors

Motivation



Introduction to K-Nearest Neighbors

- The basic idea: Observations that are closest to an arbitrary point are the most similar.

- Can be used in both classification and regression settings (i.e., can have output take the form of class membership or property values).

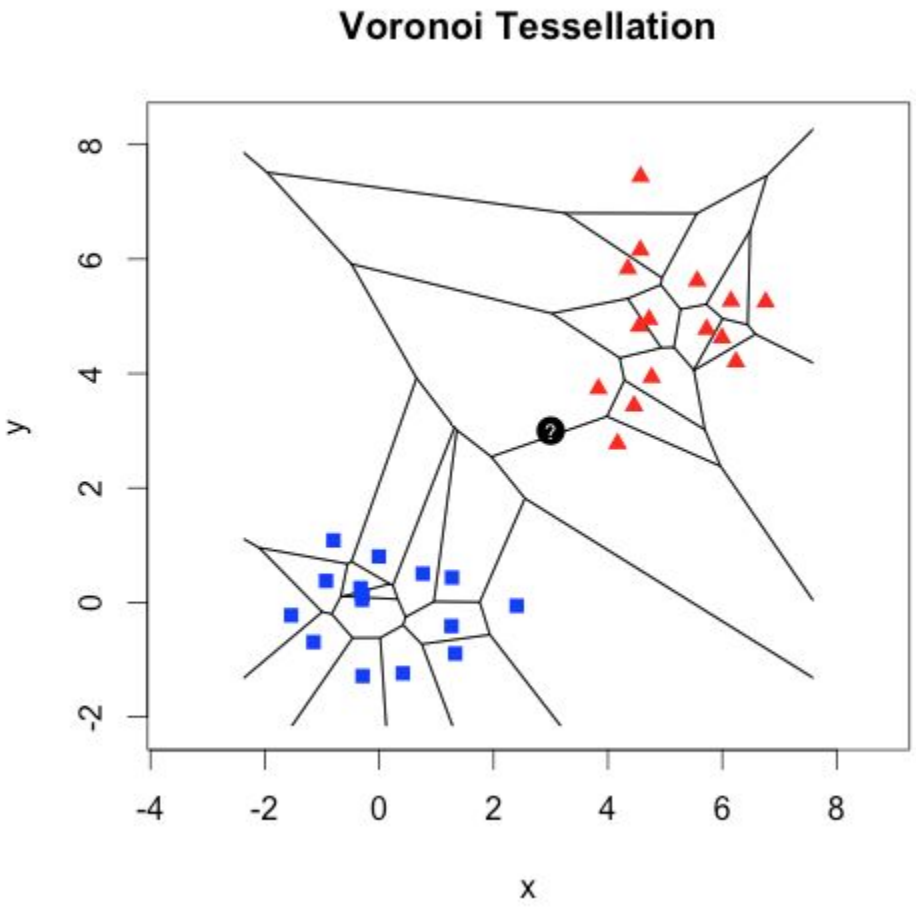
- For K-Nearest Neighbors we find the K closest observations to the data point in question, and predict the majority class as the outcome.
  - For 1-Nearest Neighbors, the single closest observation is the sole vote.

Voronoi Tessellation: Classification

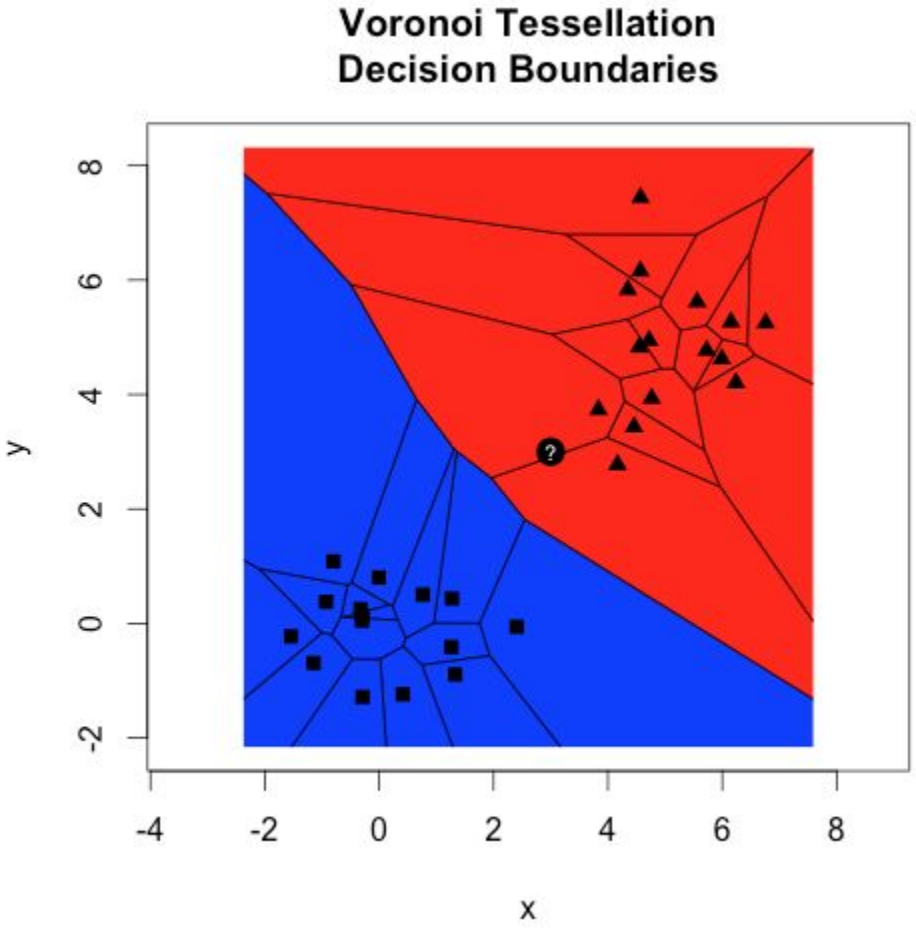
- The KNN algorithm partitions the feature space into different regions that represent classification rules; these regions are called Voronoi tessellations.
  - Boundaries represent areas where distances are equal in respect to different observations.

- By following the Voronoi tessellations, the overall decision boundary has the flexibility to be non-linear.

Voronoi Tessellation: Classification with 1NN



Voronoi Tessellation: Classification with 1NN

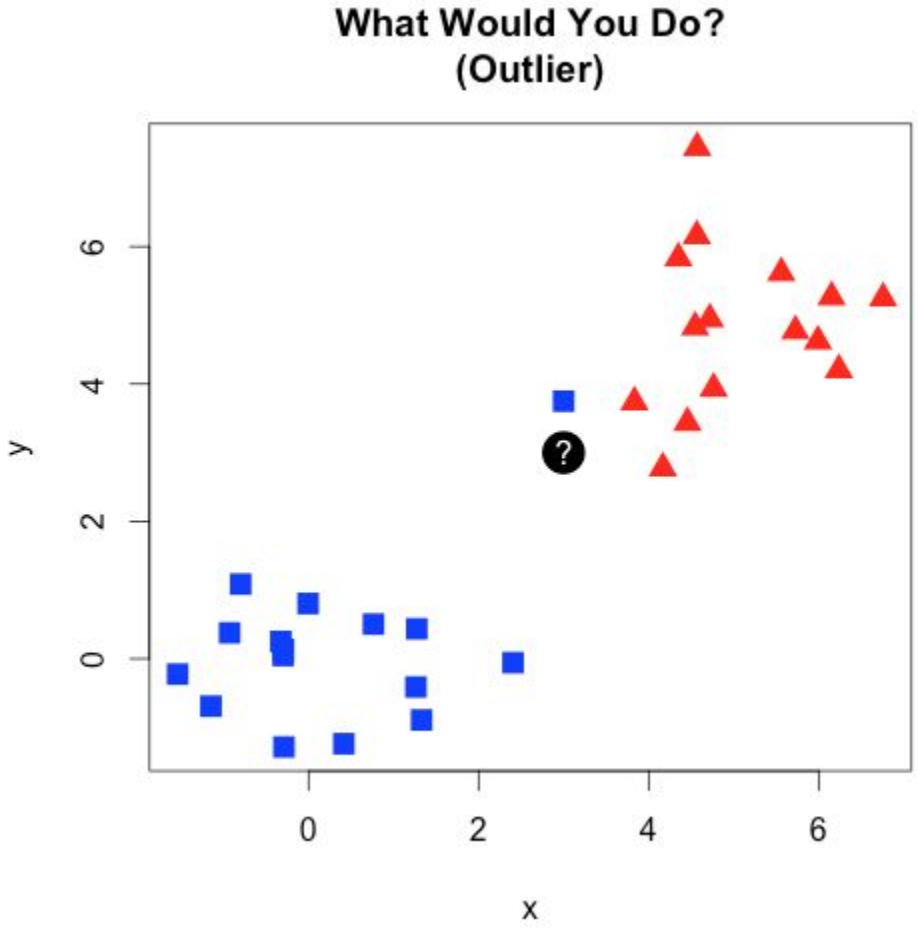


Limitations of 1-Nearest Neighbor

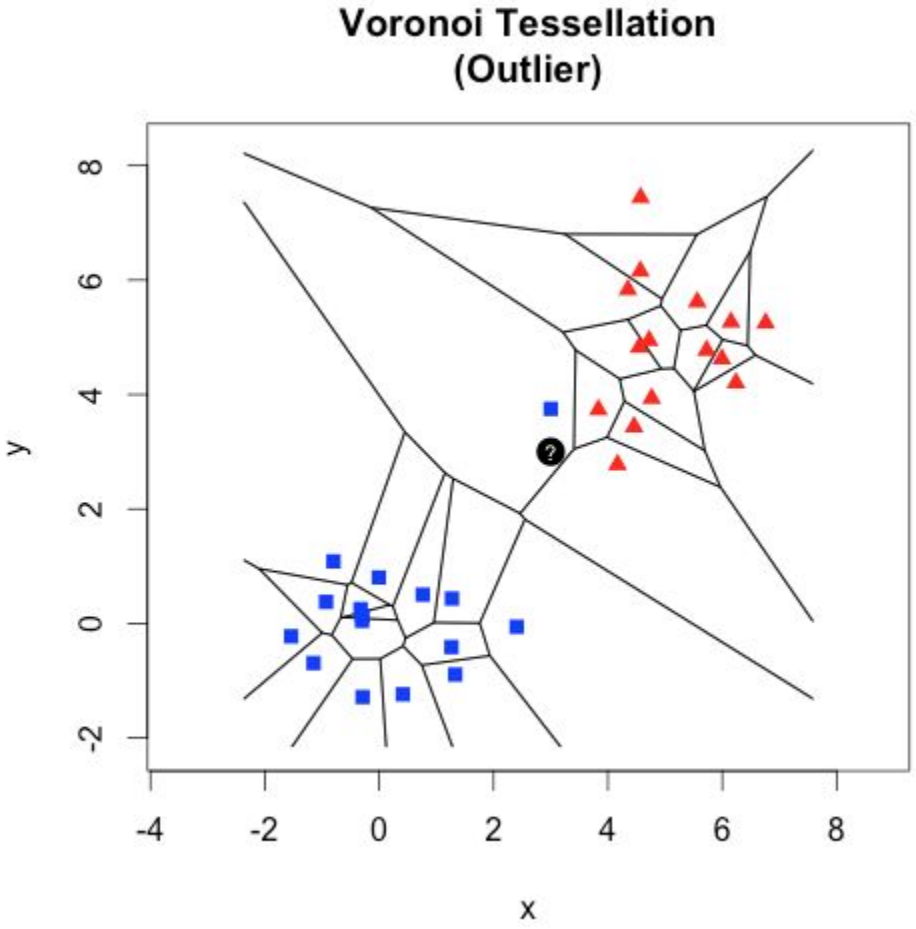
- While the algorithm is very simple to understand and implement, its simplicity comes along with some drawbacks:
  - 1NN is unable to adapt to outliers; a single outlier can dramatically change the Voronoi tessellations, and thus the decision boundaries.
  - There is no notion of class frequencies (i.e., the algorithm does not recognize that one class is more common than another).

- One way to get around these limitations and to add some stability is to consider more neighboring points (increasing the value of K), and assessing the majority vote.
  - What happens when we choose all neighbors?

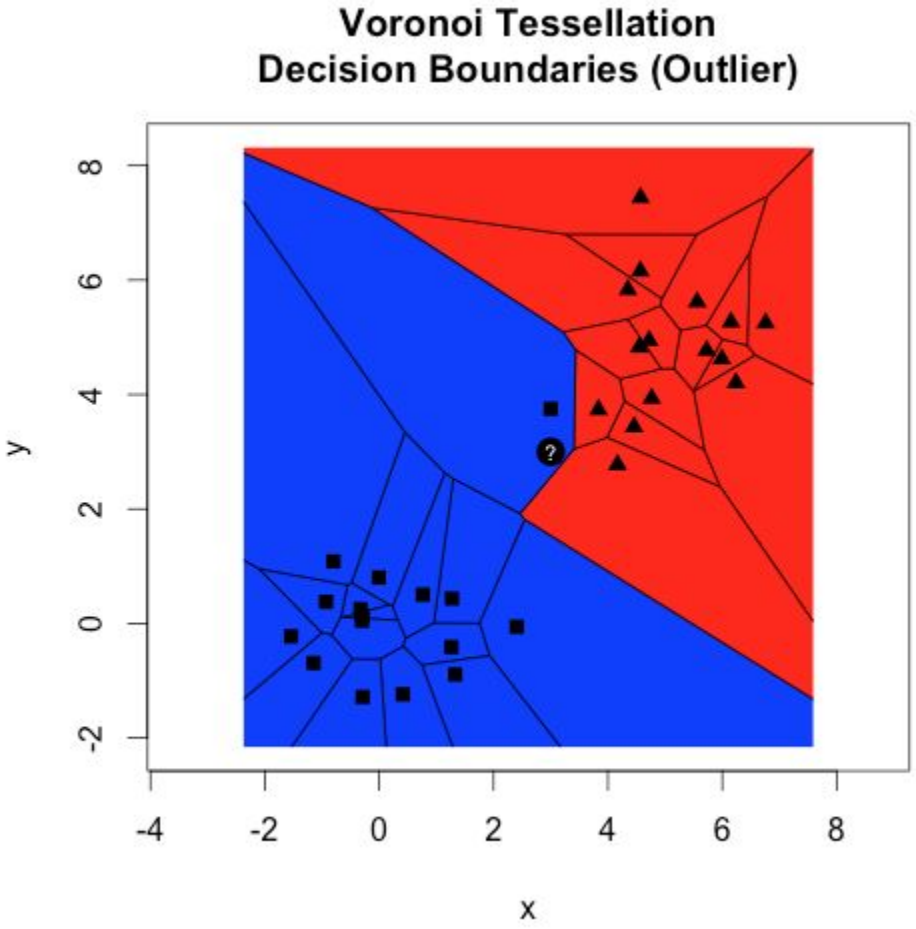
Limitations of 1-Nearest Neighbor



Limitations of 1-Nearest Neighbor



Limitations of 1-Nearest Neighbor



The K-Nearest Neighbors Classification Algorithm

- Given the following information:
  - The training set:
    - X_i: The feature values for the i^{th} observation (i.e., the location in space).
    - Y_i: The class value for the i^{th} observation (i.e., the group label).
  - The testing set:
    - X_*: The feature values for the new observation that we wish to classify.

- The KNN classification algorithm:
  - Calculate the distance between X_* and each observation X_i.
  - Determine the K observations that are closest to X_* (have the smallest distance).
  - Classify X_* as the most frequent class Y among the K selected observations.

The K-Nearest Neighbors Regression Algorithm

- Given the following information:
  - The training set:
    - X_i: The feature values for the i^{th} observation (i.e., the location in space).
    - Y_i: The real-valued target for the i^{th} observation (i.e., a continuous measurement).
  - The testing set:
    - X_*: The feature values for the new observation that we wish to regress.

- The KNN regression algorithm:
  - Calculate the distance between X_* and each observation X_i.
  - Determine the K observations that are closest to X_* (have the smallest distance).
  - Assign X_* the mean of the Y measurements among the K selected observations.

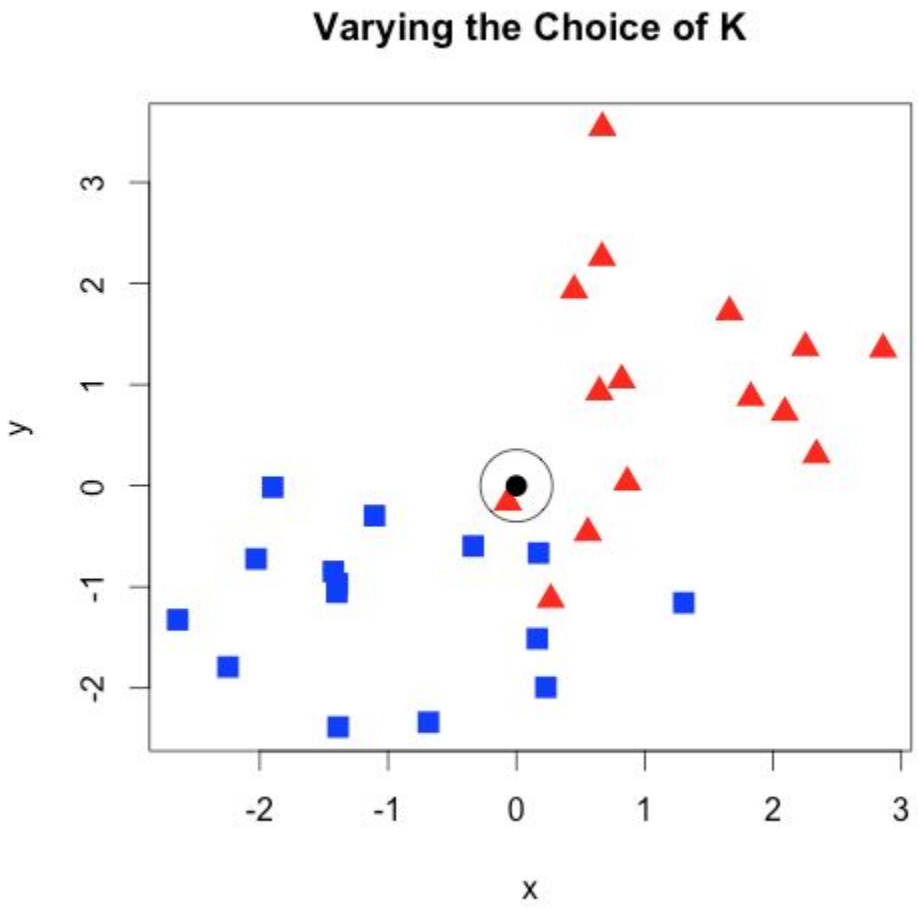
The Choice of K

- As we vary K the predicted classification rule will change, thus the choice of K has a large effect on the algorithm's performance. In general:
  - Small values of K:
    - Highlight local variations.
    - Are not robust to outliers.
    - Induce unstable decision boundaries.
  - Large values of K:
    - Highlight global variations.
    - Are robust to outliers.
    - Induce stable decision boundaries.

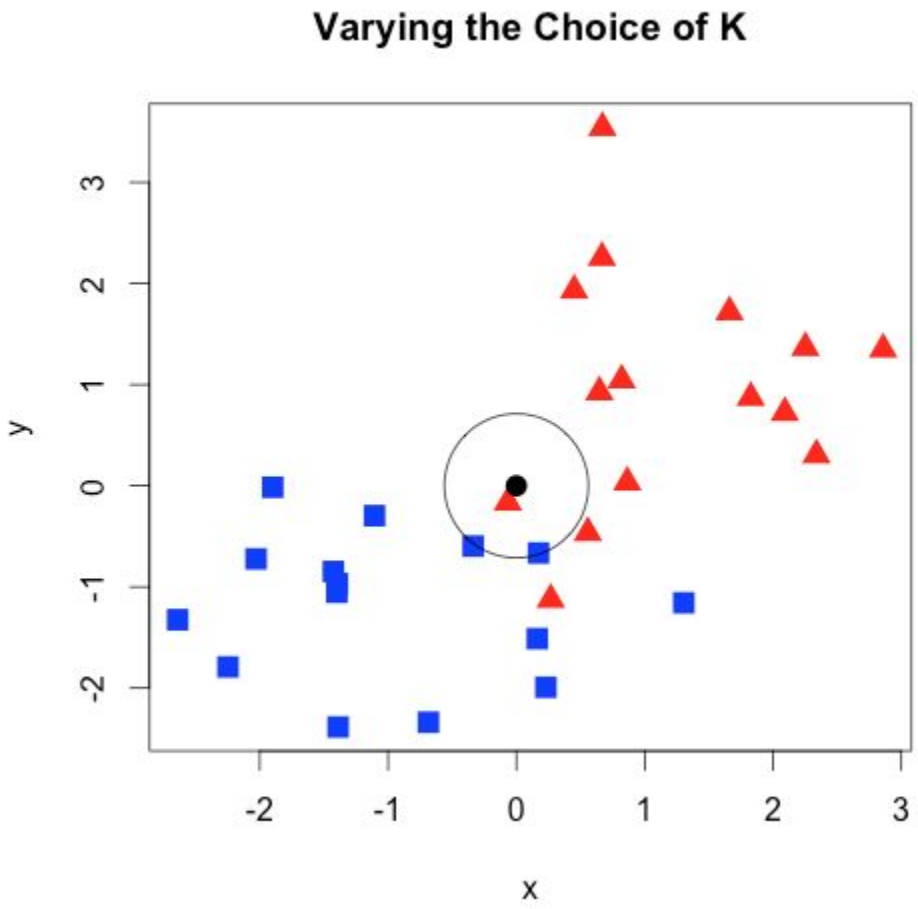
- **NB:** We will revisit the choice of K in more detail when we discuss the topic of cross-validation; however, in practice, a good balance is typically achieved with:

$$K = \sqrt{n}$$

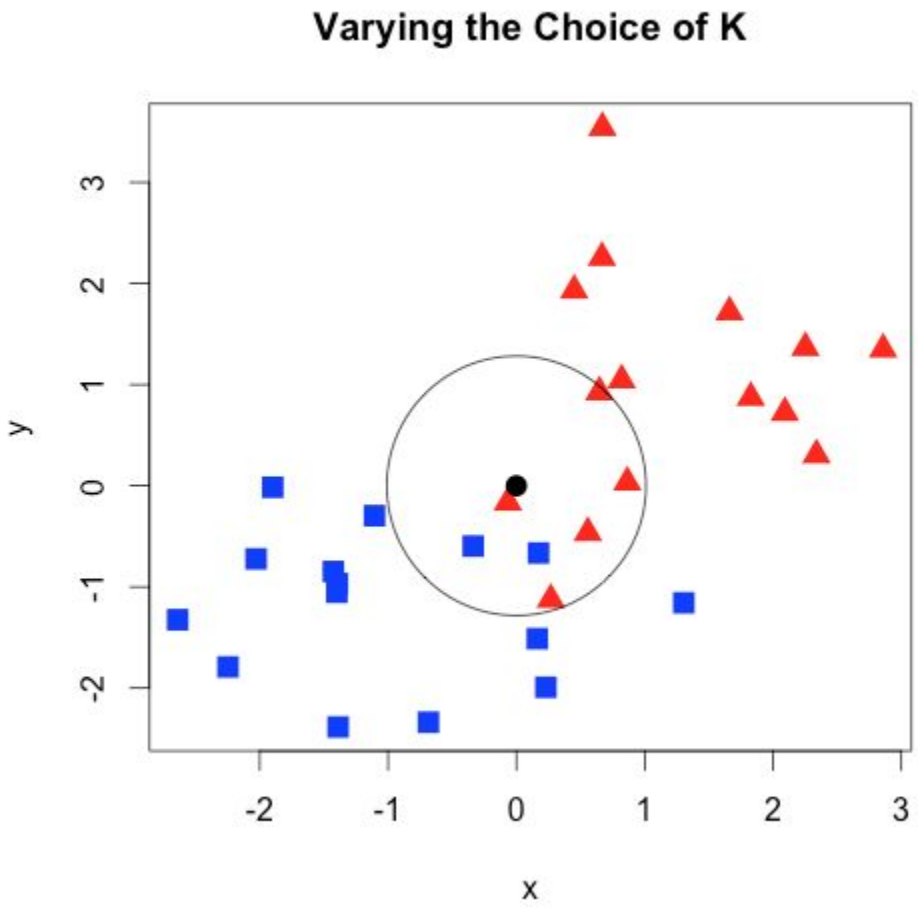
The Choice of K



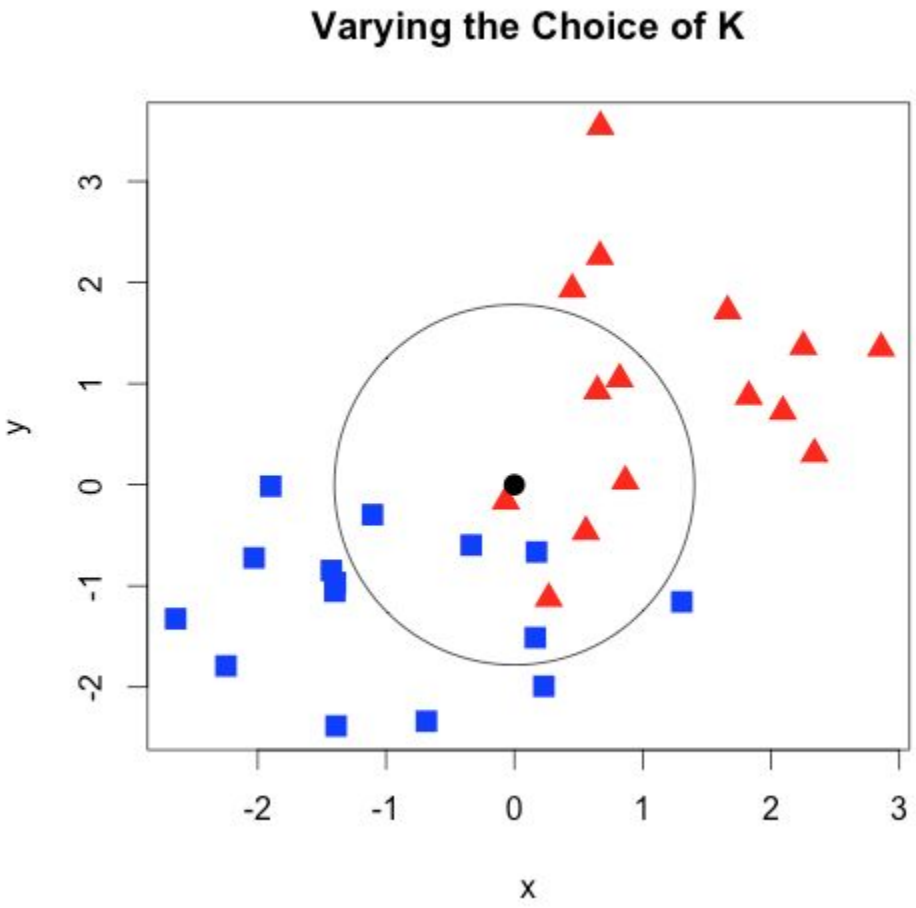
The Choice of K



The Choice of K



The Choice of K



The Choice of Distance Measure

- As we change the way we measure the distance between two points in our feature space, the classification rule will change. The choice of distance measure also has a large effect on the algorithm's performance.

- The most common distance measure for continuous observations is called the Euclidean distance, defined as:

$$D(x_1, x_2) = \sqrt[2]{\sum_d \mid x_{1d} - x_{2d} \mid^2}$$

- Euclidean distance is the "familiar" distance we typically use in everyday life; it is symmetric, treats all dimensions equally, and thus is sensitive to large deviations in a single dimension.

The Choice of Distance Measure

- The most common distance measure for categorical observations is called the Hamming distance, defined as:

$$D(x_1, x_2) = \sum_d 1_{x_{1d} \neq x_{2d}}$$

- Hamming distance looks at each attribute between observations and compares whether or not the observations are the same; each similarity is ignored while each difference is penalized.
  - The measure is symmetric and treats all dimensions equally.

The Choice of Distance Measure

- Although rarely used, there are a plethora of other distance measure choices. One family of distance functions is called the Minkowski p-norm, defined as:

$$D(x_1, x_2) = \sqrt[p]{\sum_d \mid x_{1d} - x_{2d} \mid^p}$$

- As we vary p, we define distance measures that each have different behaviors:
  - $p \rightarrow 0$: Logical And (assigns more significance to simultaneous deviations)
  - $p = 1$: Manhattan block distance (adds each component separately).
  - $p = 2$: Euclidean distance.
  - $p \rightarrow \infty$: Maximum distance, Logical Or (the largest difference among all attributes dominates the distance measure).

Breaking Ties

- What do we do if there is a tie? More specifically, how do we decide to classify an observation whose K-nearest neighborhood has an equal number of maximum group memberships?

- Some methods for breaking ties:
  - If there are only two groups, we can easily get around this by using an odd K. Why doesn't this work when there are more than two groups?
  - Use the maximum prior probability to uniformly decide all ties.
  - Randomly choose the group; for G groups:
    - Roll a G-sided die that has equally likely outcomes for each group.
    - Roll a G-sided die that has weighted outcomes for each group.
  - Use the 1NN to break the tie.

Pros & Cons of K-Nearest Neighbors

- Pros of K-Nearest Neighbors:
  - The only assumption we are making about our data is related to proximity (i. e., observations that are close by in the feature space are similar to each other in respect to the target value).
  - We do not have to fit a model to the data since this is a non-parametric approach.

- Cons of K-Nearest Neighbors:
  - We have to decide on K and a distance metric.
  - Can be sensitive to outliers or irrelevant attributes because they add noise.
  - Computationally expensive; as the number of observations, dimensions, and K increases, the time it takes for the algorithm to run and the space it takes to store the computations increases dramatically.
    - Why is this bad? We want more data!

PART 2

Naive Bayes

Naive Bayes

- Naive Bayes is a probabilistic supervised classification method concerned with describing uncertainty.

- In a nutshell, the Naive Bayes method employs a frequentist perspective on data analysis; it uses information about prior events to estimate the probability of future events.

- The method is particularly useful when your data has many categorical variables with many possible values, or if you are concerned with the interrelated nature of your variables.
  - Many algorithms tend to ignore features that have weak effects on the outcome; Naive Bayes utilizes all available evidence to make a prediction.
  - The idea is that many small effects "added together" could ladder up to have a meaningful impact.

Independent & Dependent Events

- If all events in the world were independent of one another, it would be impossible to accurately predict any future event based on data collected from another event.
  - In other words, if two events are independent, knowledge of one event does not inform knowledge of the other event.

- On the other hand, dependencies among events form the basis of predictive modeling; it can be helpful to use these dependencies in order to predict future events.
  - In other words, if two events are dependent, knowledge of one event does inform knowledge of the other event.

Conditional Probability: Bayes' Theorem

- The relationships between dependent events can be described using Bayes' Theorem. The conditional probability of of event A given event B is as follows:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} = \frac{P(A \cap B)}{P(B)}$$

- In other words, Bayes' Theorem states that the posterior is proportional to the likelihood times the prior.

- Unsurprisingly, Bayes' Theorem represents the overall underpinnings of the Naive Bayes classifier.
  - How does this theorem work?

Conditional Probability: Bayes' Theorem

- Suppose you wanted to predict whether or not an incoming email message was spam:
  - With no knowledge of the incoming email, the best guess you could give is the probability that any previous email was spam (i.e., the prior probability).

- What if you knew the incoming email contained the word "Viagra"? That knowledge should theoretically change the probability of the message being spam because of dependencies among the variables:
  - The probability that the word Viagra was used in previously observed spam messages is called the likelihood.
  - The probability that Viagra appeared in any given message is called the marginal likelihood.

Conditional Probability: Bayes' Theorem

- By implementing Bayes' Theorem, we can compute the posterior probability that measures the likelihood of the new message being spam given the evidence that the email contained the word Viagra:

$$P(spam|Viagra) = \frac{P(Viagra|spam)P(spam)}{P(Viagra)}$$

- If the posterior probability is greater than 50%, then it is more likely than not that the email is spam.
- If the posterior probability is less than 50%, then it is more likely than not that the email is not spam.

Conditional Probability: Bayes' Theorem

- Consider the following dataset of emails:

	Viagra	No Viagra	**Total**
Spam	4	16	20
Not Spam	1	79	80
Total	5	95	100

- What is the posterior probability that an email is spam given that it contains the word Viagra?
  - The likelihood P(Viagra | spam) = 4/20 = 0.2.
  - The prior P(spam) = 20/100 = 0.2.
  - The marginal likelihood P(Viagra) = 5/100 = 0.05.
  - The posterior probability P(spam | Viagra) = (0.2 * 0.2)/0.05 = 0.8; the probability that the message is spam given that it contains the word Viagra is 80%.

The Naive Bayes Algorithm

- The Naive Bayes algorithm uses Bayes' Theorem in order to perform classification. It is considered "naive" because it makes a couple of unrealistic assumptions about the data at hand:
  - All of the features in the dataset are equally important.
  - All of the features in the dataset are independent of one another.

- Luckily, even though these assumptions are rarely true in real applications, the Naive Bayes algorithm still performs quite well given assumption violations.
  - Because of its ease of application and versatility, Naive Bayes is generally a good "quick and dirty" classification method.
  - Sometimes, as long as the resulting class labels are accurate, it is not as important to obtain a precise estimate of probability (e.g., spam filtering).

- Why do we need these assumptions?

The Naive Bayes Algorithm

- Suppose we were looking at the presence of three different words when trying to classify an email as spam or not spam: W_1, W_2, and W_3. Applying Bayes' Rule, we would have:

$$P(spam|W_1 \cap W_2 \cap W_3) = \frac{P(W_1 \cap W_2 \cap W_3|spam)P(spam)}{P(W_1 \cap W_2 \cap W_3)}$$

- This formula can get unwieldy very quickly and be extremely computationally difficult to solve.
  - As more features are added, it is necessary to keep track of the probabilities of all combinations of intersecting events.
  - Extremely large training datasets would be required to ensure that all possible combinations are represented.

The Naive Bayes Algorithm

- The Naive Bayes algorithm attempts to overcome this problem by making a couple of assumptions. If we assume class-conditional independence, then the computation becomes much simpler:

$$P(spam|W_1 \cap W_2 \cap W_3) = \frac{P(W_1|spam)P(W_2|spam)P(W_3|spam)P(spam)}{P(W_1)P(W_2)P(W_3)}$$

- The resulting value of this formula can be compared to the probability that the same message is not spam given the same words.
  - Whichever probability is higher dictates the group to which the email message should belong.
  - **NB:** The denominator can be temporarily ignored for classification because it is the same in both cases; however, the denominator can help convert the values to probabilities.

The Naive Bayes Algorithm

- What happens if a specific event never occurs in our dataset? For example, suppose that in our previous example, W_2 didn't appear in any of the spam messages at all:

$$P(spam|W_1 \cap W_2 \cap W_3) = \frac{P(W_1|spam)*0*P(W_3|spam)P(spam)}{P(W_1)P(W_2)P(W_3)}$$

- Because probabilities are multiplied in the Naive Bayes algorithm, this 0 probability makes the posterior probability of spam 0 as well!
  - The absence of W_2 overruled and nullified the effects of all the other evidence.
  - Even if the email message was incredibly spammy with words W_1 and W_3 appearing in every spam email, it would still be classified as not spam!

The Laplace Estimator

- One simple solution to this problem is called the Laplace estimator:
  - The Laplace Estimator is a corrective measure that adds a small amount of error to each of the counts in the frequency table of words.
  - The addition of error ensures that each resulting probability of each event will necessarily be nonzero, even if the event did not appear in the training data.

- Typically, the Laplace estimator is chosen to be 1 so that each class/feature combination is "observed" at least once.

- With the addition of the Laplace estimator, we avoid the problem of having faulty probabilities that are strictly 0 even though we minorly artificially tamper with the raw data.

Pros & Cons of the Naive Bayes Algorithm

- Pros:
  - It is relatively simple to understand the application of the algorithm because it is based on one mathematical equation/rule.
  - Training the classifier does not require many observations, and the method also works well with large amounts of data.
  - It is easy to obtain the estimated probability for a classification prediction.

- Cons:
  - The method relies upon the faulty assumptions that the features in the dataset are independent and equally important.
  - While easily attainable, the estimated probabilities are often less reliable than the predicted class labels themselves.

PART 3

Review

Review

- Part 1: K-Nearest Neighbors
  - Motivation
  - Introduction to KNN
  - Voronoi Tessellation
    - Classification with 1NN
  - Limitations of 1NN
  - The K-Nearest Neighbors Algorithm
    - Classification
    - Regression
  - The Choice of K
  - The Choice of Distance Measure
  - Breaking Ties
  - Pros & Cons of K-Nearest Neighbors
- Part 2: Naive Bayes
  - Independent & Dependent Events
  - Conditional Probability: Bayes' Theorem
  - The Naive Bayes Algorithm
  - The Laplace Estimator
  - Pros & Cons of the Naive Bayes Algorithm
- Part 3: Review