09 KNN (Classification)

Created Date: 2022-11-07

> Metadata 🗃️
>
> - Title: K-Nearest-Neighbors (Classification)
> - Author: Andrew Jones
> - Reference: Data Science Infinity

> Links & Tags 🔗
>
> - Index: Course Note Index
> - Atomic Tag: #datascience
> - Subatomic Tags: #machinelearning #knn #classification #supervisedlearning

---

High-Level Overview

Jupyter Notebook: Basic Classification KNN Template

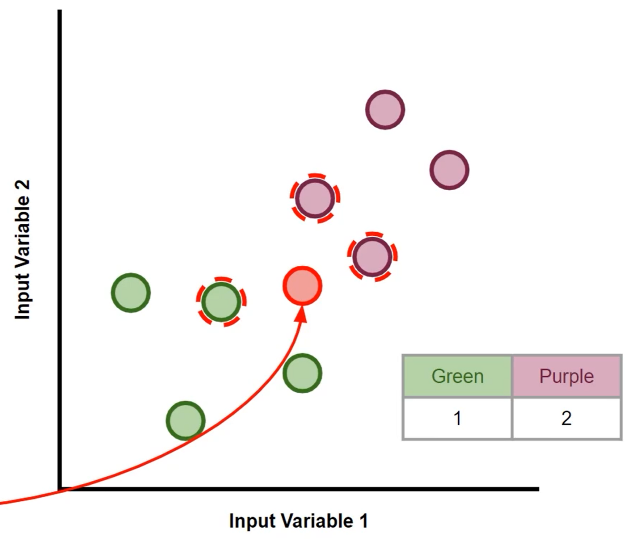
> K-Nearest-Neighbors (KNN): Predicts a class for an unknown data point using the most popular class of a number of nearby known data points. The number of nearby data points used to form the prediction is denoted by *k*.

- Model determines output classification based on nearest data points with known classification
- We can specify the amount of data points used for comparison with *k*
- In the example below, $k = 3$, therefore the model assesses the 3 closest data points and determines there are more known points classified as purple so the predicted output classification is purple

KNN for Classification

Input Variable 1	Input Variable 2	Output Variable
1.1	3.2	Green
2.3	1.0	Green
2.6	2.8	Green
3.3	1.8	Green
3.5	4.1	Purple
4.3	3.4	Purple
4.6	5.6	Purple
5.4	4.8	Purple

Input Variable 1	Input Variable 2	Output Variable
4.4	3.6	???
3.3	3.2	???



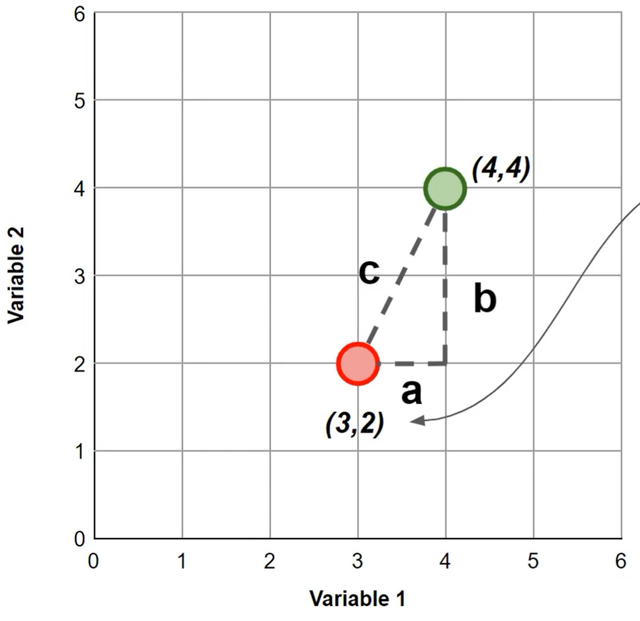
Green	Purple
1	2

Advanced Theory

Jupyter Notebook: Advanced Classification KNN Template

Measuring Distances in Multi-Dimensional Space

- Nearest neighbors are determined by the shortest Euclidean distance between input variables
- This distance is calculated using Pythagorean theorem as the distance between any two points is the hypotenuse (c)

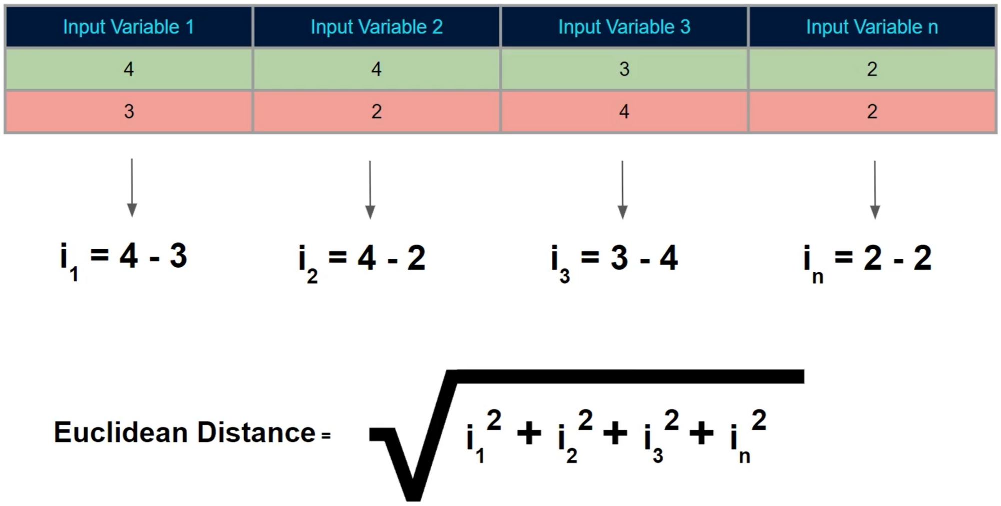


Variable 1	Variable 2
4	4
3	2

$a = 4 - 3$ ⇨ $a = 1$

$b = 4 - 2$ ⇨ $b = 2$

$$c = \sqrt{a^2 + b^2}$$



Input Variable 1	Input Variable 2	Input Variable 3	Input Variable n
4	4	3	2
3	2	4	2

$i_1 = 4 - 3$ $i_2 = 4 - 2$ $i_3 = 3 - 4$ $i_n = 2 - 2$

$$\text{Euclidean Distance} = \sqrt{i_1^2 + i_2^2 + i_3^2 + i_n^2}$$

Formula

- Euclidean Distance: $c = \sqrt{i_1^2 + i_2^2 + i_n^2}$
  - Where;
  - $i_n = q1 - p1$
  - Since we're taking the square root of these values, it doesn't matter the order as the result will always be positive

Importance of Feature Scaling

> Feature Scaling is where we force the values from different columns to exist on the same scale, in order to enhance the learning capabilities of the model. The two most common techniques are *Standardization* and *Normalization*.

- It's important to consider feature scaling when measuring distance between variables
- Standardization rescales data to have a mean of 0 and a standard deviation of 1
- Normalization rescales data so that it exists in a range between 0 and 1
- Normalization is more appropriate for the KNN algorithm
  - Comparable to categorical variables
  - Implements the same range for all variables

Formulas

- $X_{Standardized} = \frac{(X - mean(X))}{Std.Deviation(X)}$
- $X_{Normalized} = \frac{(X - min(X))}{max(X) - min(X)}$

What Value for K?

- Always use an odd number for *k*
  - If the number of classes is even, this will eliminate ties
- A low *k* value could cause incorrect classifications due to outliers
- A high k value could case incorrect classifications due to a higher volume of one classification
  - Look into class volume in data to assess if this could be an issue
- A common approach is to test different values in *k* and assessing/comparing the accuracy score for each test
  - Plotting the value of *k* against model accuracy should provide an idea for a sensible *k* value where accuracy is maximized