## Objectives

K Nearest Neighbors is a popular classification method because they are easy computation and easy to interpret. This module walks you through the theory behind k nearest neighbors as well as a demo for you to practice building k nearest neighbors models with sklearn.

* Identify common supervised machine learning algorithms
* Identify, use, and interpret k nearest neighbors models for classification
* Build k nearest neighbors models with sklearn

Neighbor count (k = )

What is need to select a knn

* Correct value for ‘k’
* How to measure closeness of neighbors?

Decision boundary

K=1 has high variance

K = all will underfit

Choosing the right value for k:

* Knn does not provide a ‘correct’ K
* The right value depends on which error metric is most important
* A common approach is to use an ‘elbow method’ approach
* This emphasizes kinks in a curve of the error rate as a function of k
* Beyond this point, the rate of improvement slows or stops

Measument of distance of knn

Euclidean distance (L2)

Manhattan distance (L1)

Scale for distance measurement

Feature scaling

K nearest Neighbors decision boundary

Regression with KNN – can answer how much

KNN overview:

Pros:

* Simple to implement (does not require estimation)
* Adapts well as new training data
* Easy to interpret

Cons:

* Slow to predict because many distance calculations
* Does not generate insight into data generating process (no model)
* Can require lots of memory if data set is large (or as it grows)
* When there are many predictors, KNN accuracy can break down due to curse of dimensionality.

Characteristics of KNN model

Linear Regression:

* Fitting involves minimizing cost function (slow)
* Model has few parameters (memory efficient)
* Prediction involves calculation (fast)

K Nearest Neighbors:

* Fitting involves storing training data (fast)
* Model has many parameters (memory intensive)
* Prediction involves finding closest neighbors (slow)

Code

# import the class contianing the classificaiton method

From sklearn.neighbors import KNeighborsClassifier

#create an instanc eof the class

KNN = KNeighborsClassifier(n\_neighbors=3)

# Fit the instanc eon the data and then predict the expected value

KNN = KNN.fit(X\_train, y\_train\_

Y\_predict = KNN.predict(X\_test)

The fit and predict/transform syntax will show up throughout the course.

Regression can be done with KNeighborsRegressor

Quiz:

A simplified way to interpret K Nearest Neighbors is by thinking of the output of this method as a decision boundary which is then used to classify new points.

* True

These are all characteristics of the K nearest neighbors algorithm Except:

* It determines the value of K

Characteristics of the k nearest neighbors algorithm are:

* It determines decision boundaries to make predictions
* It is well suited to predict variables with multiple classes
* It is sensitive to scaling

K nearest neighbor's notebook part 1.

Df.describe(),2 means we are rounding numbers to 2 decimal points

Find uniques with

* Df.nunique()

Make lists of: Ordinal variables, numerical variables, categorical variables, binary variables

Pd.cut[df[‘months’], bins=5]

* This will cut the months into bins

K nearest neighbors notebook part 2.

Change columsn to the ordinal variables with

For column in ordinal\_variables:

One hot encoding for binary variables

One hot encode with get dummies

* Drop first = true

Question 4

Y is churn value

Y are all other columns

90 percent precision

O = didn’t churn

1 = did churn

Weighted average is given more weight to 0’s

Blue = incorect values

## End of module review: K Nearest Neighbors

### **K Nearest Neighbor Methods for Classification**

K nearest neighbor methods are useful for classification. The elbow method is frequently used to identify a model with low K and low error rate.

These methods are popular due to their easy computation and interpretability, although it might take time scoring new observations, it lacks estimators, and might not be suited for large data sets.