Machine Learning Mastery
Making Developers Awesome at Machine Learning

K-Nearest Neighbors for Machine Learning

by **Jason Brownlee** on April 15, 2016 in **Machine Learning Algorithms**

Last Updated on August 12, 2019

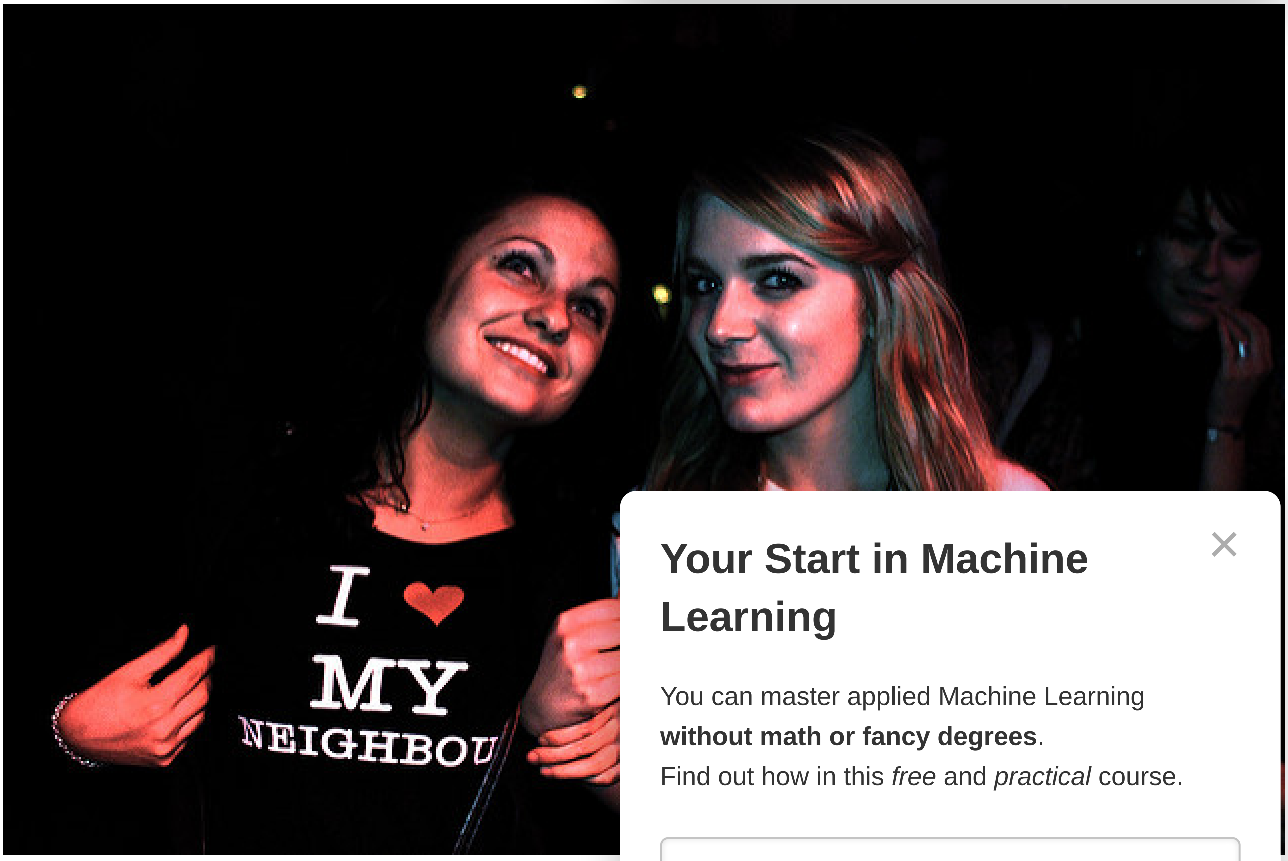
In this post you will discover the k-Nearest Neighbors (KNN) algorithm for classification and regression. After reading this post you will know.

- The model representation used by KNN.
- How a model is learned using KNN (hint, it's not).
- How to make predictions using KNN
- The many names for KNN including how different fields refer to it.
- How to prepare your data to get the most from KNN.
- Where to look to learn more about the KNN algorithm.

This post was written for developers and assumes no background in statistics or mathematics. The focus is on how the algorithm works and how to use it for predictive modeling problems. If you have any questions, leave a comment and I will do my best to answer.

Discover how machine learning algorithms work including kNN, decision trees, naive bayes, SVM, ensembles and much more in my new book, with 22 tutorials and examples in excel.

Let's get started.



K-Nearest Neighbo
Photo by Valentin Otto

KNN Model Representation

The model representation for KNN is the entire trai

It is as simple as that.

KNN has no model other than storing the entire dataset, so there is no learning required.

Efficient implementations can store the data using complex data structures like k-d trees to make look-up and matching of new patterns during prediction efficient.

Because the entire training dataset is stored, you may want to think carefully about the consistency of your training data. It might be a good idea to curate it, update it often as new data becomes available and remove erroneous and outlier data.

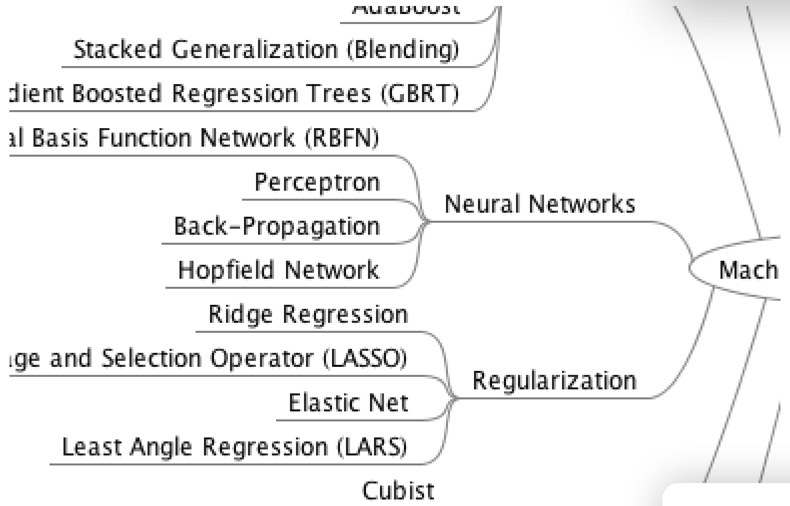
Get your FREE Algorithms Mind Map

I've created a handy mind map of 60+ algorithms organized by type.

Download it, print it and use it.

Download For Free

Also get exclusive access to the machine learning algorithms email mini-course.


AdaBoost
Stacked Generalization (Blending)
dient Boosted Regression Trees (GBRT)
al Basis Function Network (RBFN)
Perceptron
Back-Propagation
Hopfield Network
Neural Networks
Mach
Ridge Regression
ıge and Selection Operator (LASSO)
Elastic Net
Least Angle Regression (LARS)
Regularization
Cubist

Sample of the handy machine learning algorithms mind

Making Predictions with KNN

KNN makes predictions using the training dataset directly.

Predictions are made for a new instance

summarizing the output variable for those K instan
variable, in classification this might be the mode (c

To determine which of the K instances in the trainin ce
measure is used. For real-valued input variables, t
distance.

Euclidean distance is calculated as the square roo
new point (x) and an existing point (xi) across all ir

EuclideanDistance(x, xi

Other popular distance measures include:

- **Hamming Distance**: Calculate the distance between binary vectors (more).
- **Manhattan Distance**: Calculate the distance between real vectors using the sum of their absolute difference. Also called City Block Distance (more).
- **Minkowski Distance**: Generalization of Euclidean and Manhattan distance (more).

There are many other distance measures that can be used, such as Tanimoto, Jaccard, Mahalanobis and cosine distance. You can choose the best distance metric based on the properties of your data. If you are unsure, you can experiment with different distance metrics and different values of K together and see which mix results in the most accurate models.

Euclidean is a good distance measure to use if the input variables are similar in type (e.g. all measured widths and heights). Manhattan distance is a good measure to use if the input variables are not similar in type (such as age, gender, height, etc.).

The value for K can be found by algorithm tuning. It is a good idea to try many different values for K (e.g. values from 1 to 21) and see what works best for your problem.

The computational complexity of KNN increases with the size of the training dataset. For very large training sets, KNN can be made stochastic by taking a sample from the training dataset from which to calculate the K-most similar instances.

KNN has been around for a long time and has been very well studied. As such, different disciplines have different names for it, for example:

- **Instance-Based Learning**: The raw training instances are used to make predictions. As such KNN is often referred to as instance-based learning or a case-based learning (where each training instance is a case from the problem domain).
- **Lazy Learning**: No learning of the model is required and all of the work happens at the time a prediction is requested. As such, KNN is often referred to as a lazy learning algorithm.
- **Non-Parametric**: KNN makes no assumptions about the functional form of the problem being solved. As such KNN is referred to as a non-parametric machine learning algorithm.

KNN can be used for regression and classification problems.

KNN for Regression

When KNN is used for regression problems the pr e K-most similar instances.

KNN for Classification

When KNN is used for classification, the output ca frequency from the K-most similar instances. Each class with the most votes is taken as the predictior

Class probabilities can be calculated as the norma ass in the set of K most similar instances for a new dat problem (class is 0 or 1):

p(class=0) = count(class=0) / (count(class=0)+count(class=1))

If you are using K and you have an even number of classes (e.g. 2) it is a good idea to choose a K value with an odd number to avoid a tie. And the inverse, use an even number for K when you have an odd number of classes.

Ties can be broken consistently by expanding K by 1 and looking at the class of the next most similar instance in the training dataset.

Curse of Dimensionality

KNN works well with a small number of input variables (p), but struggles when the number of inputs is very large.

Each input variable can be considered a dimension of a p-dimensional input space. For example, if you had two input variables x1 and x2, the input space would be 2-dimensional.

As the number of dimensions increases the volume of the input space increases at an exponential rate.

In high dimensions, points that may be similar may have very large distances. All points will be far away from each other and our intuition for distances in s This might feel unintuitive at first, but this general proble

Best Prepare Data for KNN

- **Rescale Data**: KNN performs much better if all of the data has the same scale. Normalizing your data to the range [0, 1] is a good idea. It may also be a good idea to standardize your data if it has a Gaussian distribution.
- **Address Missing Data**: Missing data will mean that the distance between samples can not be calculated. These samples could be excluded or the missing values could be imputed.
- **Lower Dimensionality**: KNN is suited for lower dimensional data. You can try it on high dimensional data (hundreds or thousands of input variables) but be aware that it may not perform as well as other techniques. KNN can benefit from feature selection that reduces the dimensionality of the input feature space.

Further Reading

If you are interested in implementing KNN from scr

- Tutorial To Implement k-Nearest Neighbors in

Below are some good machine learning texts that ng
perspective.

1. Applied Predictive Modeling, Chapter 7 for reg
2. Data Mining: Practical Machine Learning Tools
3. Doing Data Science: Straight Talk from the Fr
4. Machine Learning, Chapter 8

Also checkout K-Nearest Neighbors on Wikipedia.

Summary

In this post you discovered the KNN machine learning algorithm. You learned that:

- KNN stores the entire training dataset which it uses as its representation.
- KNN does not learn any model.
- KNN makes predictions just-in-time by calculating the similarity between an input sample and each training instance.
- There are many distance measures to choose from to match the structure of your input data.
- That it is a good idea to rescale your data, such as using normalization, when using KNN.

If you have any questions about this post or the KNN algorithm ask in the comments and I will do my best to answer.
