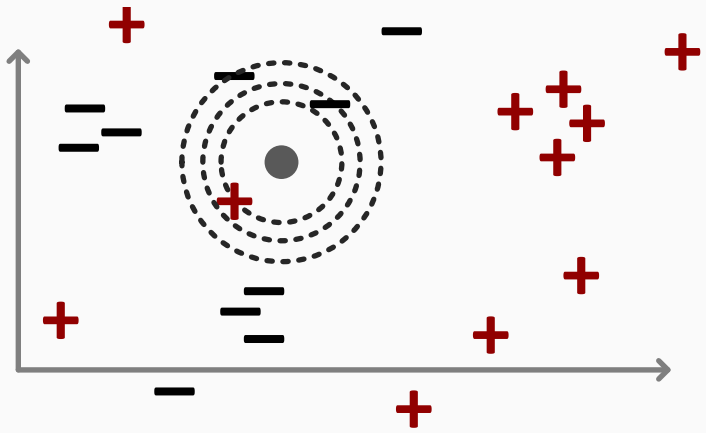
K Nearest Neighbour ⁘ (KNN)

KNN is one of the most basic machine learning algorithms because it does not require a training set or proprietary model to implement.

KNN can be used for both **Classification** and **Regression** problems by using x's K-Nearest Neighbors to predict what x's next label will likely be (majority vote through proximity ranges)

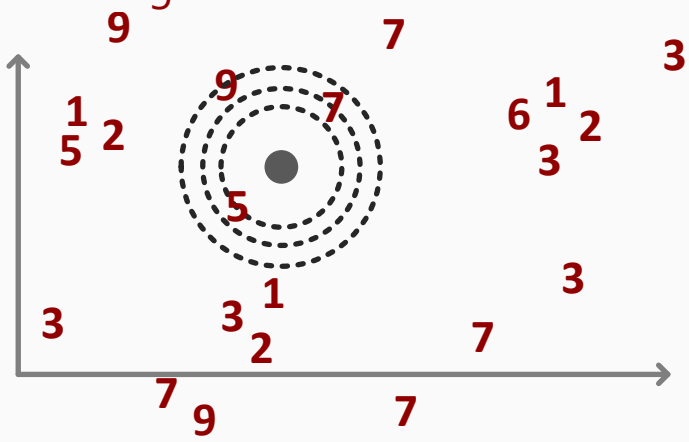
The choice of K-Neighbours determines the classification or label applied to new values based upon proximity to the initializing point:

KNN Classification



The illustration above:
1-NN is classified as **+**
2-NN is tied between **+** and **–**
3-NN is classified as **–**

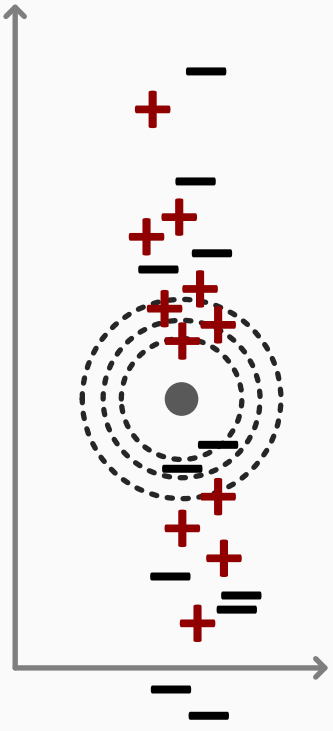
KNN Regression



The illustration above:
1-NN is classified as 5
2-NN is classified as 6 ([5+7]/2)
3-NN is classified as 7 ([5+7+9]/3)

The parameter ***k*** controls overfitting the dataset. Cross-validation can possibly help. K-Nearest Neighbours does not account for the physical distance between the nearest points; weighting cannot be assigned to Neighbours that are closer than others.

Changing how distance is measured can impact how the algorithm classifies Neighbour proximity. The distance measure has to be meaningful. Separately scaled attributes can be misclassified. For example, if two variables are income and height, the income scale will be large (thousands of dollars) while the height will be less significant (inches). Therefore, the model will classify the vertical axis containing the range of nearest neighbors which is truly not representative of the dataset (right figure).



Pros/Cons to KNN

Simple and powerful method that can be used instead of other classification/regression techniques. Training is not necessary and new examples easily added ("lazy model").

However, KNN is expensive and slow. Computation of a new point requires re-running the entire model to compute all distances again. Additionally, KNN might require heavy feature scaling.