INFO5002: Intro to Python for Info Sys

K-Nearest Neighbours

DSfS 165-174

Slides created by: Zachary Doucet

Northeastern University

Similar things are similar

- How do we organise a grocery store?
  - We place similar things next to each other.
- When searching for a specific item we look at what is around and try to find similar things to get closer.

How do we compare how similar something is?

- We need to use a distance function on a specific numerical feature.
  - E.g. comparing the heights between fruits.

Choose appropriate distance function

Let's say we have two points: x_1 and x_2

- manhattan: $\sum_{i=1}^{d} \left| x_{1,i} - x_{2,i} \right|$

  minkowski: $\left(\sum_{i=1}^{d} \left(x_{1,i} - x_{2,i} \right)^p \right)^{1/p}$

- euclidean: $\sqrt{\sum_{i=1}^{d} \left(x_{1,i} - x_{2,i} \right)^2}$

- cosine: $1 - \frac{x_1 \cdot x_2}{\left\| x_1 \right\| \left\| x_2 \right\|}$

Algorithm

- When we predict the label for **x** we calculate the distance between **x** to every datapoint in our training set according to the chosen distance function.
- We then select the k neighbours with the lowest distance.
- Based on these neighbours we can predict the class or regression. If class, return the most popular one and for regression can return the average. Can optionally weigh each “vote” by distance.

Curse of Dimensionality

- As you increase the number of dimensions you increase the total state space which decreases the likelihood of points being near to each other.
  - If you had 100 points between 0 and 1 in 1D they will randomly be closer together than 100 points in 2D which will further be closer than in 3D.

Reduce before use

- With high dimensions neighbours may not be close and thus cannot be representative.
- Try dimensionality reduction to reduce the number of dimensions first before using.
- Otherwise get exponentially more data as you increase the number of dimensions.

Scikit-Learn

- Need to decide between:
  - KNeighborsClassifier: when target continuous value
  - KNeighborsRegressor: when target is a class
  - KNeighborsTransformer: Transform features into weighted graph.

```
from sklearn.neighbors import KNeighborsClassifier

model = KNeighborsClassifier(n_neighbors=5, weights="uniform",
    algorithm="auto", leaf_size=30, p=2, metric="minkowski",
    metric_params=None, n_jobs=None)

# To train the model
model.fit(X_train, y_train)

# Get path of decision making (num_samples, num_nodes)
neigh_dist, neigh_ind = model.kneighbors(X=none)

 ...
```

Let’s practice

- On the Iris dataset (on Canvas) try to train a k-nearest neighbours model (KNeighborsClassifier).