Distance-based Classification and Clustering

25th July, 2019

AI42001 MLFA

Data Representation

- Training data: (X_i, Y_i) where X: feature vector, Y: label
- Test data: ($X_{test,}$?)
- X: representation of data-points (feature)

A simple idea for classification

- Training data: (X_i, Y_i) where X: feature vector, Y: label
- Test data: (X_{test}, ?)
- X: representation of data-points (feature)

- Idea: things that "look" similar, are usually the same type!
- If X_{test} = X_i, then probably Y_{test} = Y_i !

But is it a good idea?

- Training data: (X_i, Y_i) where X: feature vector, Y: label
- Test data: $(X_{test}, ?)$
- X: representation of data-points (feature)

- Idea: things that "look" similar, are usually the same type!
- If $X_{test} = X\_i$, then probably $Y_{test} = Y\_i$!

- Is X a good enough representation?
- In reality, $X_{test} = X\_i$ will rarely happen (especially if X is continuous!)

But is it a good idea?

- Training data: (X_i, Y_i) where X: feature vector, Y: label
- Test data: (X_{test}, ?)
- X: representation of data-points (feature)

- Idea: things that "look" similar, are usually the same type!
- If X_{test} = X_i, then probably Y_{test} = Y_i !

- Is X a good enough representation? - Let's assume it is
- In reality, X_{test} = X_i will rarely happen (especially if X is continuous!)

Distance between feature vectors

- If X is continuous-valued, $X\_{test} = X\_i$ will almost surely never happen!
- But, $X\_{test} \sim X\_i$ is possible!

- $X\_{test} \sim X\_i$: Euclidean Distance between the two points is very less
- $|| X\_{test} - X\_i ||_2$ is very low!

- $||a - b||_2 = \sqrt{\sum(a\_i - b\_i)2}$ (also called the l_2-norm of a-b)

Nearest-Neighbor Classification

- Training: N labelled examples (X_i, Y_i) where i: 1 to N
- Function learnt: the training set itself!
- Testing: X_test
- $Y\_{pred} = Y\_{n}$, where $n = \arg\min_i ||X\_{test} – X\_i||_2$

Nearest-Neighbor Classification

- Training: N labelled examples (X_i, Y_i) where i: 1 to N
- Function learnt: the training set itself!
- Testing: X_test
- $Y\_{pred} = Y\_n$, where $n = \arg\min_i ||X\_{test} - X\_i||_2$

- Compute the Euclidean distance between the test datapoint and each of the N training datapoints
- Choose that training point for which this distance is minimum (Nearest-Neighbor)
- Use its label as the predicted label of the test point!

Nearest-Neighbor Classification

- Training: N labelled examples (X_i, Y_i) where i: 1 to N
- Function learnt: the training set itself!
- Testing: X_test
- $Y\_{pred} = Y\_{n}$, where $n = \arg\min_i ||X\_{test} – X\_i||_2$

- Not very robust due to outliers!
- Too much computation and storage required!

K-nearest Neighbors Classification

- Problem 1: The nearest neighbour of the test point may be outlier!

- Outliers are rare
- K nearest neighbors: unlikely to contain many outliers

- Solution:

1) Sort the training points according to distance from test point
2) Choose the first K training points
3) Predicted label = most frequent label among them!

K-nearest Neighbors Regression

- Problem 1: The nearest neighbour of the test point may be outlier!

- Outliers are rare
- K nearest neighbors: unlikely to contain many outliers

- Solution:

1) Sort the training points according to distance from test point
2) Choose the first K training points
3) Predicted label = mean of their labels!

Nearest Mean Classification

- Problem 2: Too much computation (N) and storage (N*(D+1)) required!

- One solution: keep only one representative from each class
- How to choose the representative?
- Mean of feature vectors all data-points in that class!

- Mean for class k: $\mu\_k = \sum 1(Y\_i=k) * X\_i\ 1(Y\_i=k) / \sum 1(Y\_i=k)$
- Compare test point X_{test} with each μ_k and choose label of the closest!
- $Y\_{pred} = \text{argmin}_k \, || X\_{test} - \mu\_k ||_2$

Nearest Mean Classification

- Problem 2: Too much computation (N) and storage (N*(D+1)) required!

- One solution: keep only one representative from each class
- How to choose the representative?
- Mean of feature vectors all data-points in that class!

- Mean for class k: $\mu\_{k} = \sum 1(Y\_{i}=k) * X\_{i}\ 1(Y\_{i}=k) / \sum 1(Y\_{i}=k)$
- Compare test point X_{test} with each μ_{k} and choose label of the closest!
- $Y\_{pred} = \text{argmin}_k \; ||X\_{test} - \mu\_{k}\;||_2$ [K comparisons instead of N]

Distance-based Clustering

- Input: feature vectors X_i for N points

- Initially, consider each point as a separate cluster
- Keep merging two clusters if they are close enough!

- Cluster 1: [X_{a1}, X_{a2}, …., X_{am}], Cluster 2: [X_{b1}, X_{b2}, …., X_{bn}]
- Are these two clusters “close”?
- Points are close based on Euclidean Distance, but what about point sets ??

Cluster Linkage Criteria

- We need a measure of distance between point sets
- Compute distances between m*n pairs of points from the two sets

- Single linkage: minimum distance between two points from the two sets
- Multiple linkage: maximum distance between two points from the two sets
- Average linkage: mean distance between two points from the two sets

Agglomerative Clustering

- Initially, consider each point as a separate cluster
- Merge a pair of clusters if they are closer than a threshold (choose any criteria)!
- Repeat till no more mergers possible!

- Final number of clusters: not fixed beforehand

- Single linkage: relaxed criteria, less clusters
- Multiple linkage: tough criteria, more clusters