Rozpoznávanie obrazcov - 7th lab
kNN and validation

Viktor Kocur
viktor.kocur@fmph.uniba.sk

DAI FMFI UK

6.4.2019

k nearest neighbors

Basics

In the feature space $\mathbb{R}^n$ with a metric ρ we can choose a class for a feature vector $\vec{x} \in \mathbb{R}^n$ by finding its k nearest neighbors from the training set and select the class which has the most members from the k neighbors.

Training

This method does not need training. The method always takes the whole training set and looks for neighbors. With large training set this method can be slow.

Metric functions

Definition

Let P be a set. Then a metric is a function $\rho : P \times P \mapsto \mathbb{R}_0^+$ such that $\forall p, q, r \in P$:

1. $\rho(p, q) \geq 0$
2. $\rho(p, q) = 0 \Leftrightarrow p = q$
3. $\rho(p, q) = \rho(q, p)$
4. $\rho(p, r) \leq \rho(p, q) + \rho(q, r)$

A set with a metric is also called a metric space.

Metrics

Metrics on $\mathbb{R}^n$

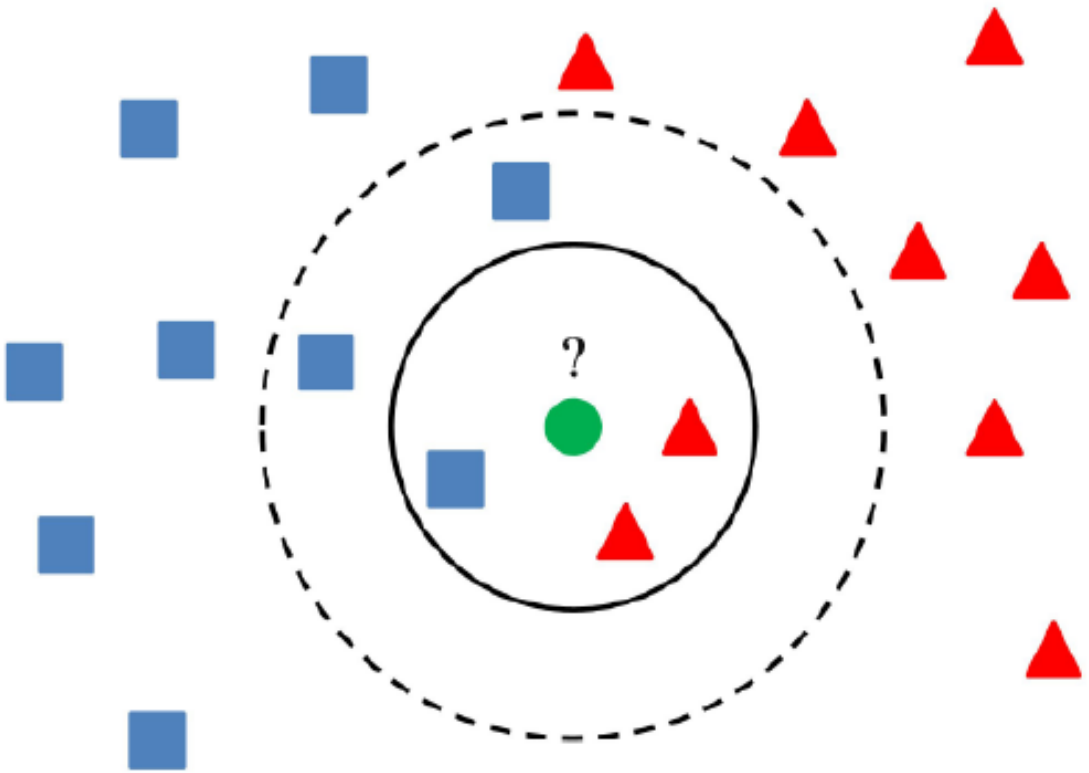
$$\rho_e(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

$$\rho_m(\vec{x}, \vec{y}) = \sum_{i=1}^{n}|x_i - y_i|$$

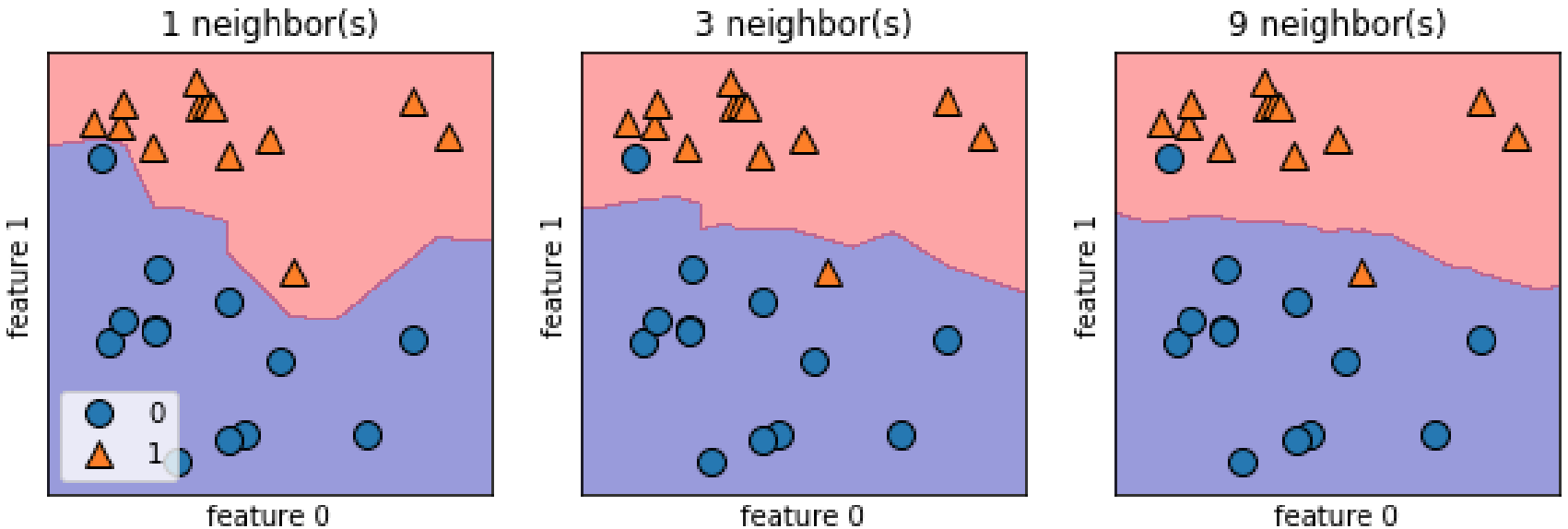
$$\rho_q(\vec{x}, \vec{y}) = \left(\sum_{i=1}^{n}(x_i - y_i)^q\right)^{\frac{1}{q}}$$

$$\rho_{max}(\vec{x}, \vec{y}) = max_i|x_i - y_i|$$

Choosing k



Choosing k



Exercise

Exercise

Create a function mykNN(k, X, y, p) which returns class for a vector p based on training data X with classes y.

Exercise

Test the function on the data from the last lab and the fisheriris database.

Exercise

fitcknn

Mdl = fitcknn(X,y) - creates a kNN classifier

predict

Mdl.predict(x) - retursn kNN model prediction

Properties

Interesting properties are: 'Standardize', 'Distance', 'NumNeighbors', 'NSMethod'. Check them in help.

Exercise

Change the m-file for displaying the SVM classifier from the previous lab and show the boundaries for kNN.

Data split

Training set

So far we have only used only the training set. All of the data we had were used to find the parameters of the model.

Test set

In case we want to verify that our model is reliable it is important to leave some of the data for testing. The testing data are only used at the very end our research when our model is ready. The test set is used to verify the model. You should never use the training set to select the method, its parameters or hyperparameters.

Data split

Validation set

Since the test set cannot be used to choose the model we need one more set for this purpose. The validation set is used to determine the correct approach and finding the optimal hyperparameters of the model.

Data split

We split the data based on their amount, properties and models. Huge amounts of data are required for neural networks, therefore a split can be 80/10/10. With some other methods a split of 60/20/20 is good enough. In some cases the spil can be even 40/20/40.

Validation

Hyperparameters

We can determine the correct hyperparameters on the validation. Hyperparameters are the settings/parameters, which affect how the model is trained or how the prediction works. Hyperparameters for SVM include the kernel type and scale. For kNN it can be the choice of k and the metric used.

Validation

We perform validation by training the model (for kNN just creating it) on the the training set with different hyperparameters. These models are then tested on the validation set. We select a reliability metric, ideally classification accuracy. Based on the results we choose the hyperparameters which achieved the best results.

Validation - exercise

Exercise

Split the data from the previous lab into train/val/test split of 60/20/20. Choose the best k parameter for the kNN classifier and a metric on the validation split.

Proper representation

Some datasets are ordered by class or occur in some othe regular order. It is therefore necessary to verify that the split is meningful. Ideally we would keep the same ratio of classes for all sets.

Crossvalidation

Crossvalidation

If we only have very little data we do not split it into the training and validation sets. The data are split into n approximately similar subsets. We train the model on all except one of the subsets and test it on the remaining set. We repeat this n times for all the subsets and average out the results.

Matlab

```
Mdl = fitcknn(X, y, 'NumNeighbors', k);
CVMdl = crossval(Mdl)
loss = kfoldLoss(CVMdl)
```

Crossvalidation

Automatic selection of hyperparameters

For most of the fitc.. functions it is possible to let Matlab select the hyperparameters automatically. Check help if you want to use this feature.

Matlab

```
Mdl = fitcknn(X,Y,'OptimizeHyperparameters','auto')
```