ASSIGNMENT 1: KNN

Johannes Kofler

JOHANNES KEPLER
UNIVERSITY LINZ

Institute for
Machine Learning

Copyright statement:

Supervised learning in a nutshell:

1. Acquire labeled dataset (input features + target values)
2. Divide dataset into training and test set
3. Select preprocessing pipeline, features, and model class based on training set
4. Optimize the model parameters on the training set
5. Optionally use validation set or CV to determine best model
6. Go back to step 3 if evaluation on validation/training set gave new insights
7. Use test set to calculate estimate for generalization error/risk

k-Nearest Neighbors Classifier (1) – Basics

- Suppose we have a labeled data set $\mathbf{Z}$ and a distance measure on the input space. Then the k-nearest neighbors classifier is defined as follows:

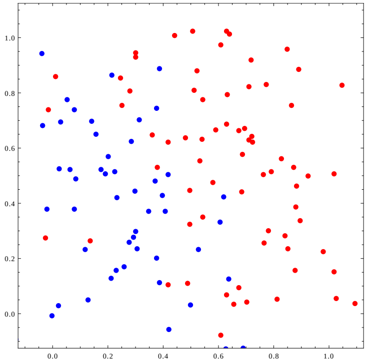
$$g_{k\text{-NN}}(\mathbf{x};\mathbf{Z}) = \text{class that occurs most often among } k \text{ samples closest to } \mathbf{x}$$

- For $k = 1$: nearest neighbors classifier:

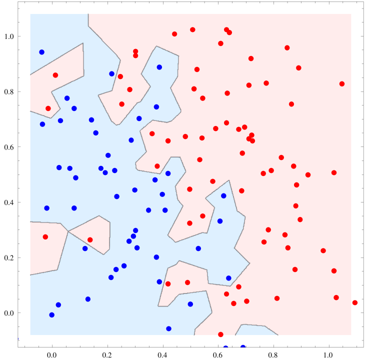
$$g_{\text{NN}}(\mathbf{x};\mathbf{Z}) = \text{class of the sample that is closest to } \mathbf{x}$$

- In case of ties: e.g. random class assignment or class with larger number of samples is assigned.
- k-NN regression: assign average value of nearest neighbors

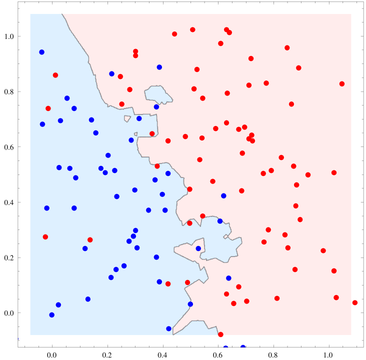
k-Nearest Neighbors Classifier (2) – Data set



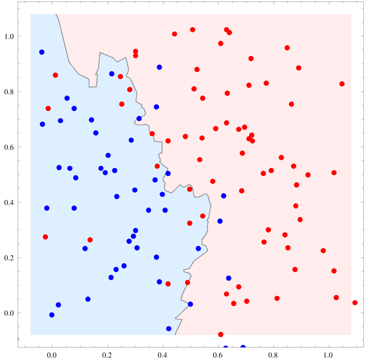
k-Nearest Neighbors Classifier (3) – kNN with $k = 1$



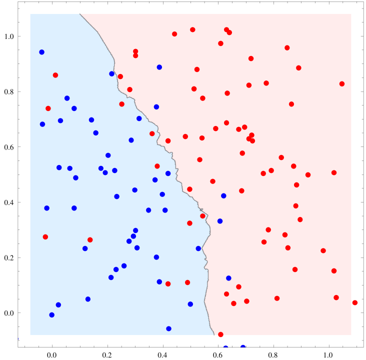
k-Nearest Neighbors Classifier (4) – kNN with $k = 5$



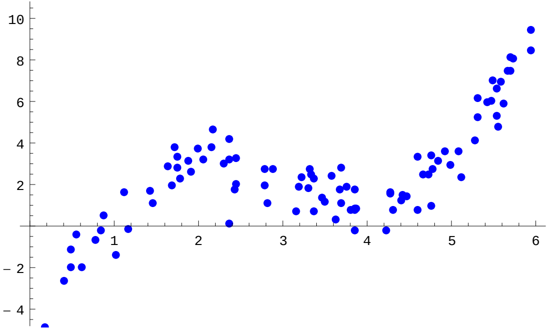
k-Nearest Neighbors Classifier (5) – kNN with $k = 13$



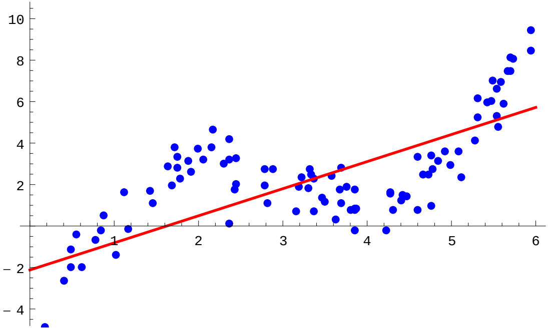
k-Nearest Neighbors Classifier (6) – kNN with $k = 25$



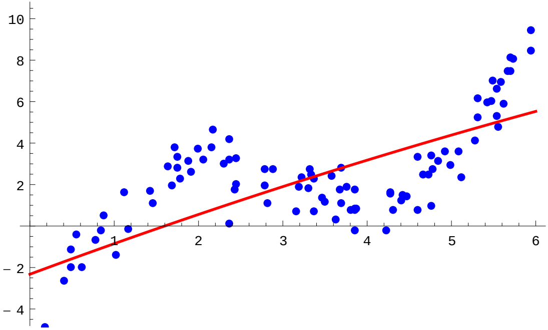
Polynomial regression in $d = 1$ (1) – Plot of data



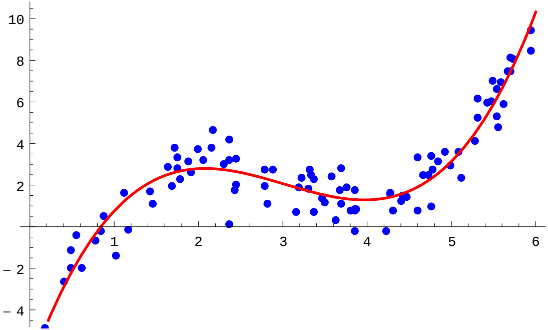
Polynomial regression in $d = 1$ (2) – Regression with degree $m = 1$



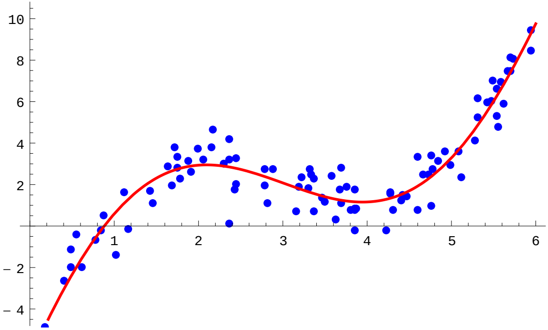
Polynomial regression in $d = 1$ (3) – Regression with degree $m = 2$



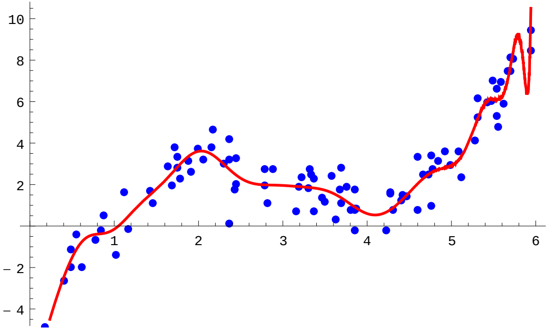
Polynomial regression in $d = 1$ (4) – Regression with degree $m = 3$



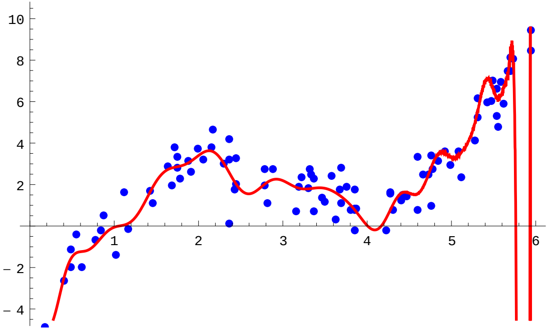
Polynomial regression in $d = 1$ (5) – Regression with degree $m = 5$



Polynomial regression in $d = 1$ (6) – Regression with degree $m = 25$



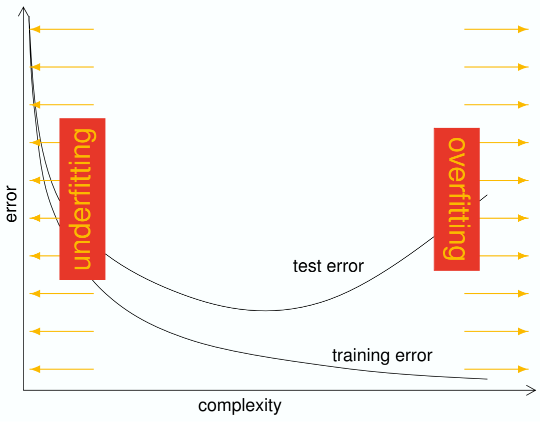
Polynomial regression in $d = 1$ (7) – Regression with degree $m = 75$



Bias-Variance Trade-off – Intuition

- Previous example: instance of one of basic problems of supervised machine learning: bias-variance trade-off.
- Recall from Unit 1:
    1. Underfitting: model is too coarse to fit training or test data (too low model class complexity): e.g. $m = 1$
    2. Overfitting: model fits well to training data but not to future/test data (too high model class complexity): e.g. $m = 75$

Bias-Variance Trade-off (2) Notorious situation in practice



Recap of formal framework: Generalization error/risk and Empirical Risk Minimization

- The generalization error or risk is the expected loss on future data:

$$R(g(.;\mathbf{w})) = \int\limits_X \int\limits_{\mathbb{R}} L(y, g(\mathbf{x};\mathbf{w}))\, p(\mathbf{x}, y) dy d\mathbf{x}$$

- In practice, we hardly have any knowledge about $p(\mathbf{x}, y)$. Precise definition: next slide.
- In practise: minimize the empirical risk R_{emp} on our dataset (=Empirical Risk Minimization):

$$R_{\mathrm{emp}}(g(.;\mathbf{w}), \mathbf{Z}_l) = \frac{1}{l} \cdot \sum_{i=1}^{l} L(y_i, g(\mathbf{x}_i;\mathbf{w}))$$

Risk estimation: Test set method

- Assume our data samples are independently and identically distributed (i.i.d.)*
- We can split our dataset of l samples into 2 subsets:

  Training set: the subset with m samples we perform ERM on (i.e. optimize parameters on)

  Test set: a subset with $l - m$ samples we use to estimate the risk

- Our estimate R_{emp} on test set will show if we overfit to noise in training set

[*) i.i.d.: each sample has the same probability distribution as the others and all are mutually independent.]

Test set method: Practical hints

- No overlap between training and test set samples (i.i.d.!)
- Random sampling of training and test set samples (i.i.d.!)
- Test set samples are not to be used for preprocessing, feature selection, model selection, etc.
- We might want to use 3 separate subsets:

  Training set: subset we train a model on (optimize model parameters)

  Validation set: subset we select best model from training on (=model selection)

  Test set: subset we use to estimate risk

Cross Validation

- For small datasets, the requirement that training and test set must not overlap is painful
- Solution: Cross Validation (CV)
  - Split dataset into n disjoint folds
  - Use $n - 1$ folds as training set, left-out fold as test set
  - Train n times, every time leaving out a different fold as test set
  - Average over n estimated risks on test sets to get better estimate of generalization capability

Cross Validation

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
CV 1	V	T	T	T	T
CV 2	T	V	T	T	T
CV 3	T	T	V	T	T
CV 4	T	T	T	V	T
CV 5	T	T	T	T	V

5-fold Cross Validation

T: Training set; V: Test set