CZECH TECHNICAL UNIVERSITY IN PRAGUE

Faculty of Electrical Engineering
Department of Cybernetics

Bias-variance trade-off. Crossvalidation. Regularization.

Petr Pošík

How to evaluate a predictive model?

Model evaluation

Fundamental question: What is a good measure of “model quality” from the machine-learning standpoint?

Model evaluation

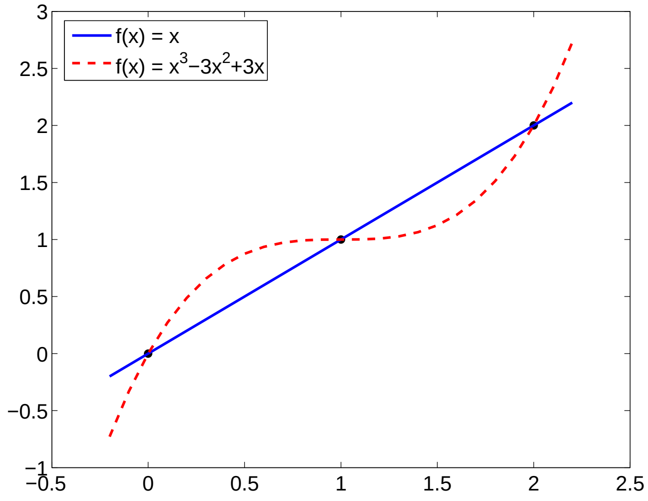
Fundamental question: What is a good measure of "model quality" from the machine-learning standpoint?

- We have various measures of model error:
  - For regression tasks: MSE, MAE, ...
  - For classification tasks: misclassification rate, measures based on confusion matrix, ...
- Some of them can be regarded as finite approximations of the *Bayes risk*.
- Are these functions *good approximations* when measured on the data the models were trained on?

Model evaluation

Fundamental question: What is a good measure of “model quality” from the machine-learning standpoint?

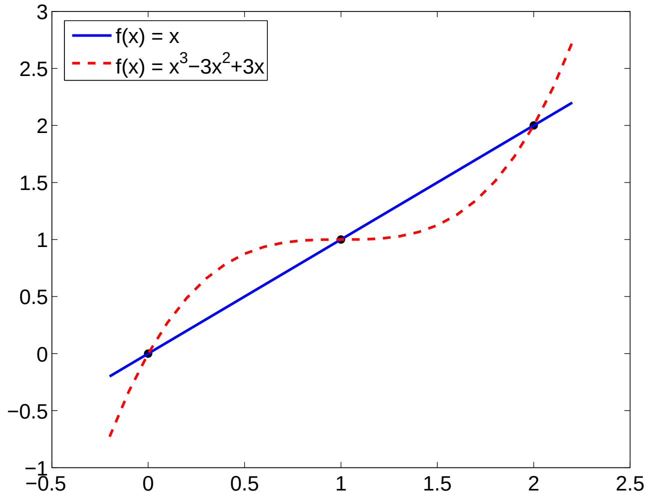
- We have various measures of model error:
  - For regression tasks: MSE, MAE, ...
  - For classification tasks: misclassification rate, measures based on confusion matrix, ...
- Some of them can be regarded as finite approximations of the *Bayes risk*.
- Are these functions *good approximations* when measured on the data the models were trained on?



Model evaluation

Fundamental question: What is a good measure of "model quality" from the machine-learning standpoint?

- We have various measures of model error:
  - For regression tasks: MSE, MAE, ...
  - For classification tasks: misclassification rate, measures based on confusion matrix, ...
- Some of them can be regarded as finite approximations of the *Bayes risk*.
- Are these functions *good approximations* when measured on the data the models were trained on?

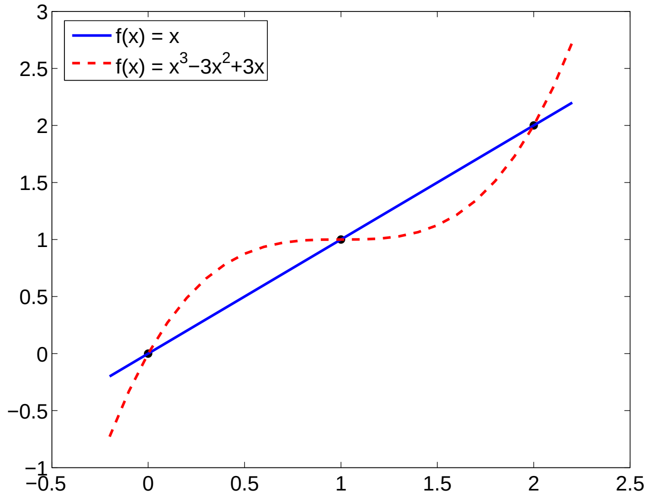


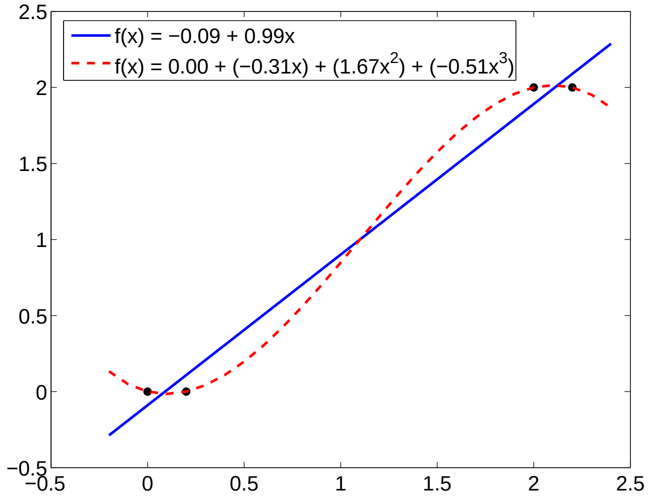
Using MSE only, both models are equivalent!!!

Model evaluation

Fundamental question: What is a good measure of "model quality" from the machine-learning standpoint?

- We have various measures of model error:
  - For regression tasks: MSE, MAE, ...
  - For classification tasks: misclassification rate, measures based on confusion matrix, ...
- Some of them can be regarded as finite approximations of the *Bayes risk*.
- Are these functions *good approximations* when measured on the data the models were trained on?



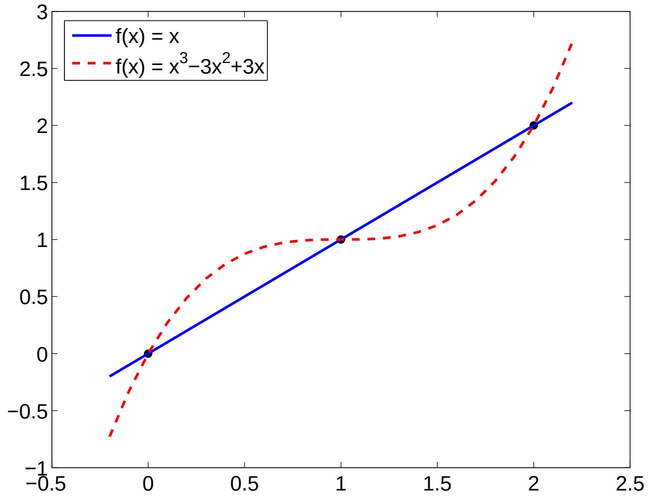


Using MSE only, both models are equivalent!!!

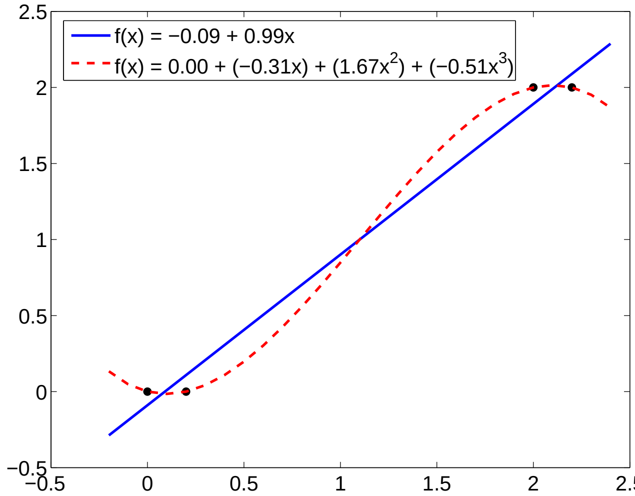
Model evaluation

Fundamental question: What is a good measure of "model quality" from the machine-learning standpoint?

- We have various measures of model error:
  - For regression tasks: MSE, MAE, ...
  - For classification tasks: misclassification rate, measures based on confusion matrix, ...
- Some of them can be regarded as finite approximations of the *Bayes risk*.
- Are these functions *good approximations* when measured on the data the models were trained on?



Using MSE only, both models are equivalent!!!

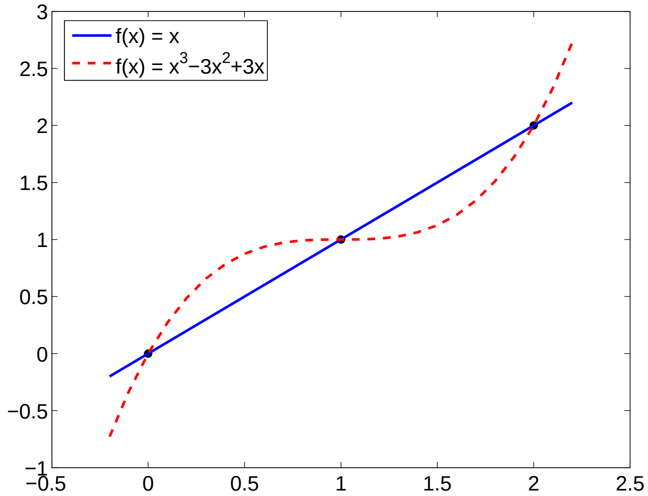


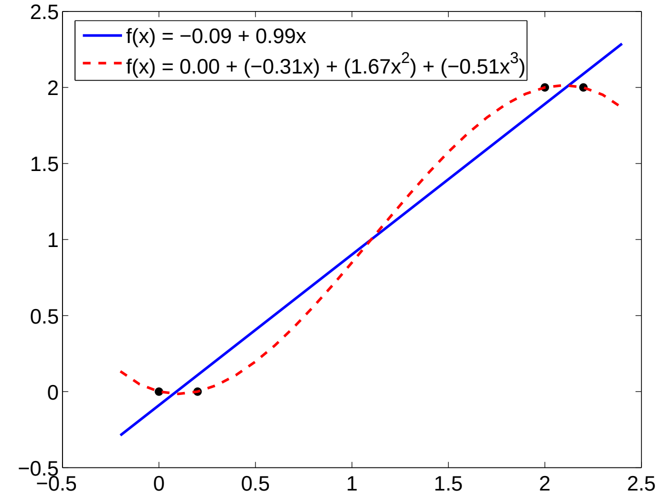
Using MSE only, the cubic model is better than linear!!!

Model evaluation

Fundamental question: What is a good measure of "model quality" from the machine-learning standpoint?

- We have various measures of model error:
  - For regression tasks: MSE, MAE, ...
  - For classification tasks: misclassification rate, measures based on confusion matrix, ...
- Some of them can be regarded as finite approximations of the *Bayes risk*.
- Are these functions *good approximations* when measured on the data the models were trained on?





Using MSE only, both models are equivalent!!!

Using MSE only, the cubic model is better than linear!!!

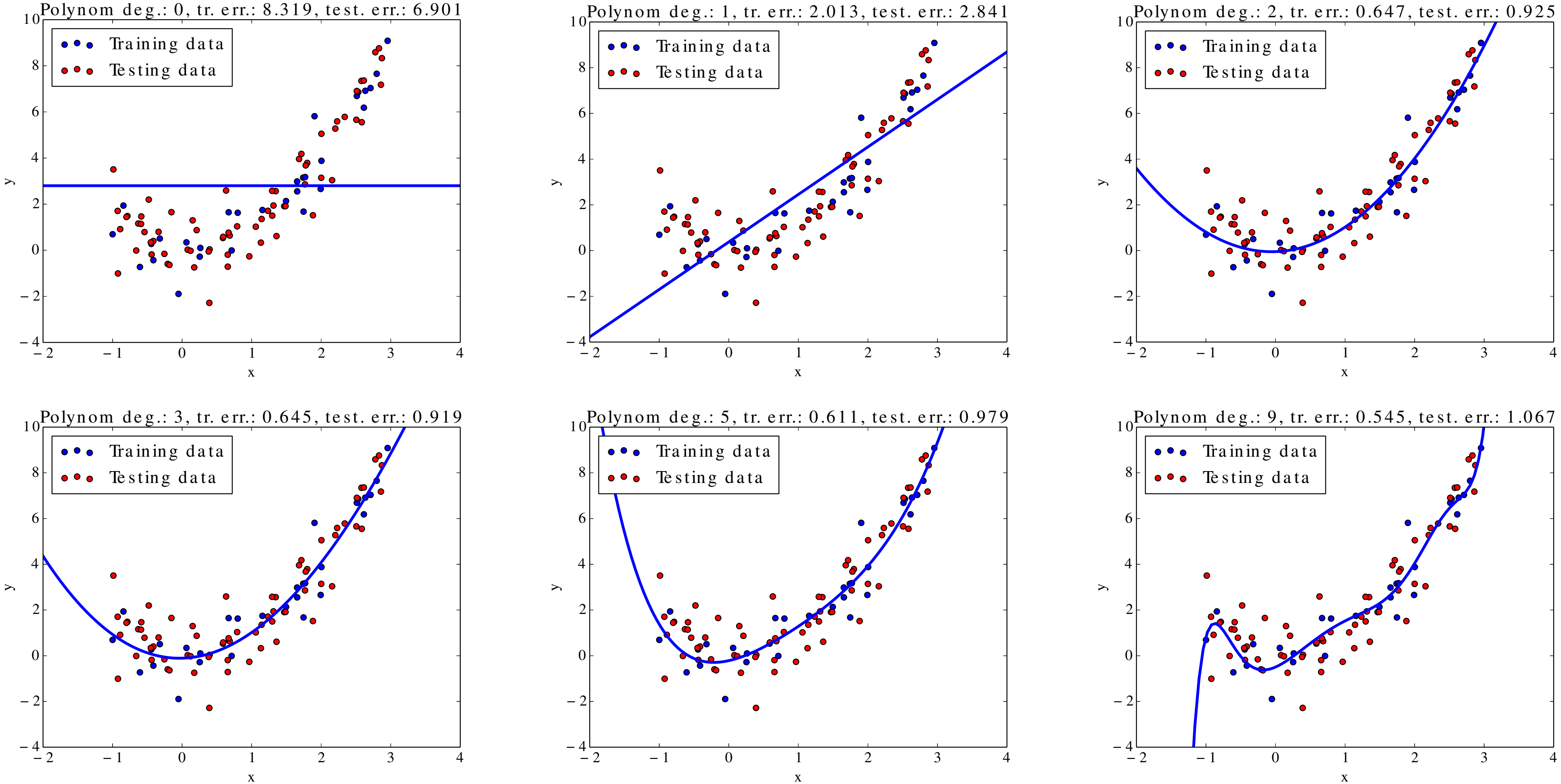
A basic method of evaluation is *model validation on a different, independent data set* from the same source, i.e. on **testing data**.

Validation on testing data

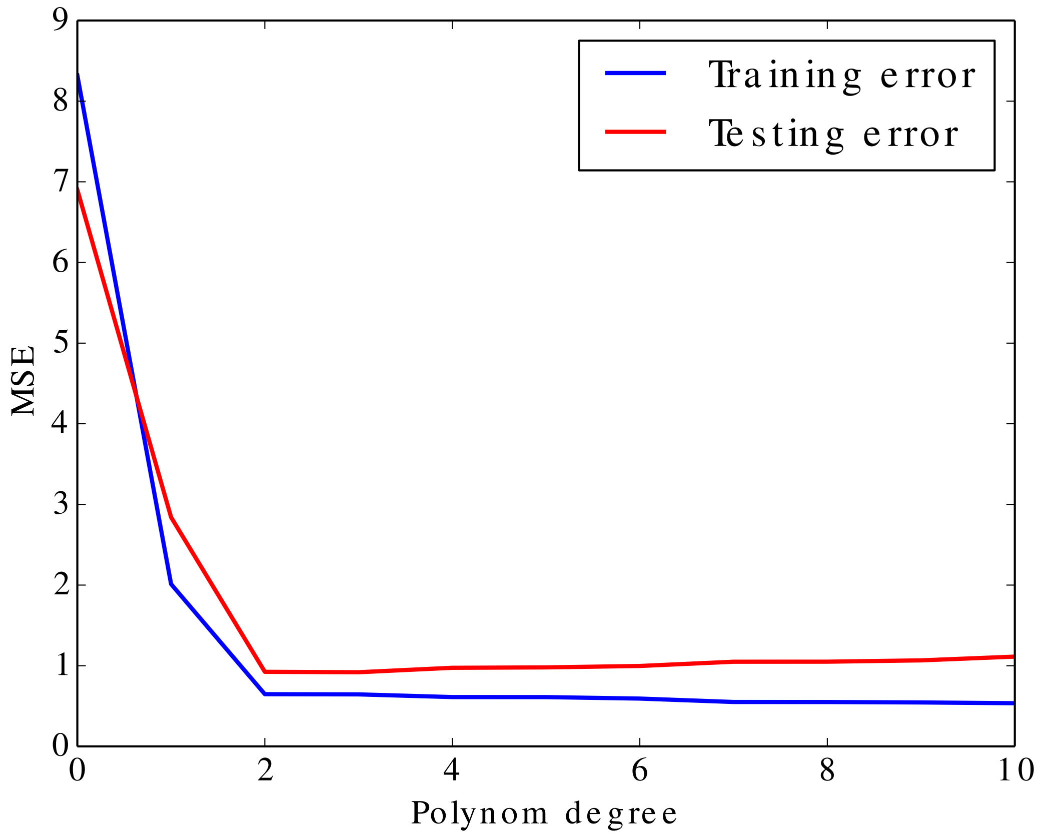
Example: Polynomial regression with varrying degree:

$$X \sim U(-1, 3)$$

$$Y \sim X^2 + N(0, 1)$$



Training and testing error



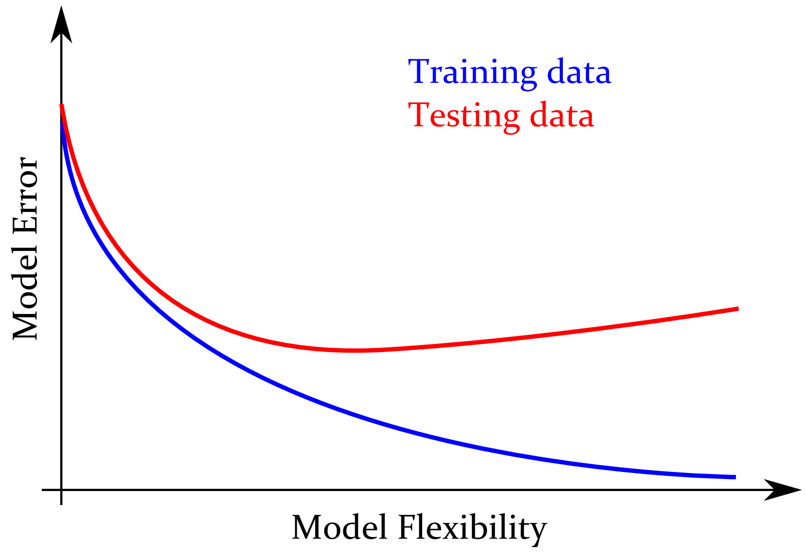
- The *training error* decreases with increasing model flexibility.
- The *testing error* is minimal for certain degree of model flexibility.

Overfitting

Definition of overfitting:

- Let H be a hypothesis space.
- Let $h \in H$ and $h' \in H$ be 2 different hypotheses from this space.
- Let $\mathrm{Err}_{\mathrm{Tr}}(h)$ be an error of the hypothesis h measured on the training dataset (training error).
- Let $\mathrm{Err}_{\mathrm{Tst}}(h)$ be an error of the hypothesis h measured on the testing dataset (testing error).
- We say that h is overfitted if there is another h' for which

$$\mathrm{Err}_{\mathrm{Tr}}(h) < \mathrm{Err}_{\mathrm{Tr}}(h') \wedge \mathrm{Err}_{\mathrm{Tst}}(h) > \mathrm{Err}_{\mathrm{Tst}}(h')$$

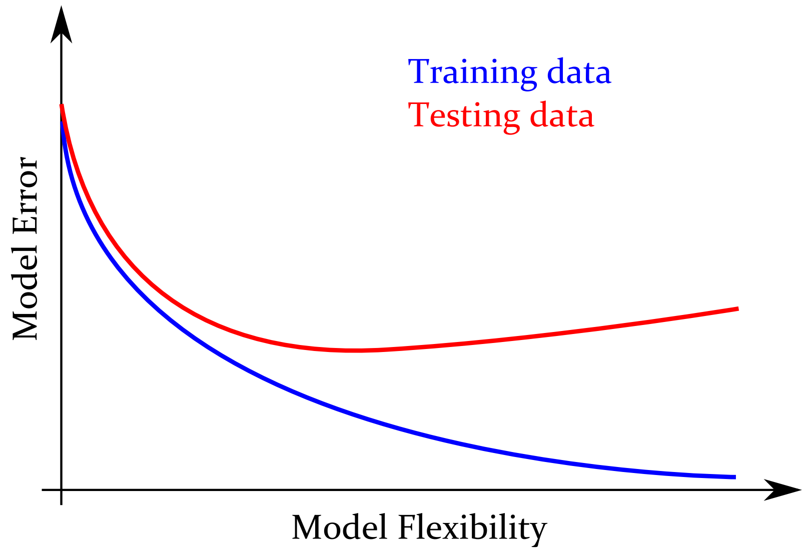


- "When overfitted, the model works well for the training data, but fails for new (testing) data."
- Overfitting is a general phenomenon *affecting all kinds of inductive learning*.

Overfitting

Definition of overfitting:

- Let H be a hypothesis space.
- Let $h \in H$ and $h' \in H$ be 2 different hypotheses from this space.
- Let $\mathrm{Err}_{\mathrm{Tr}}(h)$ be an error of the hypothesis h measured on the training dataset (training error).
- Let $\mathrm{Err}_{\mathrm{Tst}}(h)$ be an error of the hypothesis h measured on the testing dataset (testing error).
- We say that h is overfitted if there is another h' for which



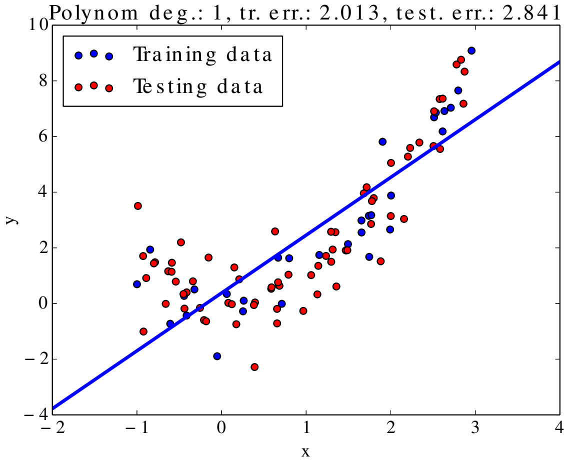
$$\mathrm{Err}_{\mathrm{Tr}}(h) < \mathrm{Err}_{\mathrm{Tr}}(h') \wedge \mathrm{Err}_{\mathrm{Tst}}(h) > \mathrm{Err}_{\mathrm{Tst}}(h')$$

- "When overfitted, the model works well for the training data, but fails for new (testing) data."
- Overfitting is a general phenomenon *affecting all kinds of inductive learning*.

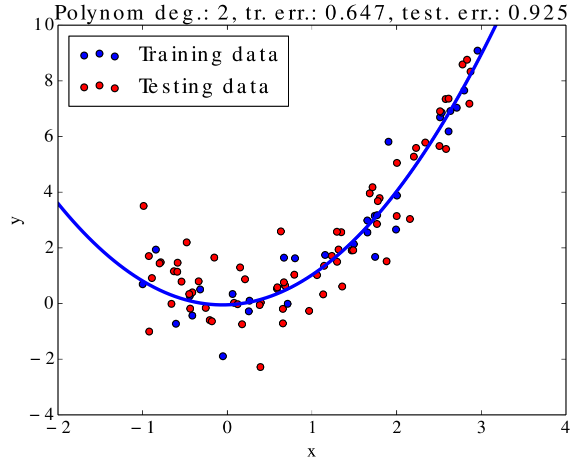
We want models and learning algorithms with a good **generalization ability**, i.e.

- we want models that encode *only the patterns valid in the whole domain*, not those that learned the specifics of the training data,
- we want algorithms able to find *only the patterns valid in the whole domain* and ignore specifics of the training data.

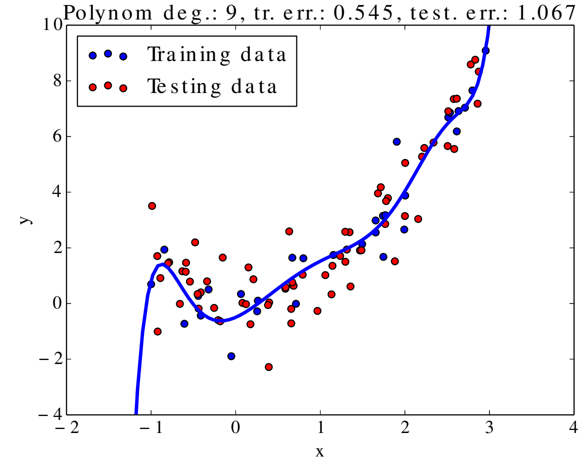
Bias vs Variance



High bias:
model not flexible enough
(Underfit)

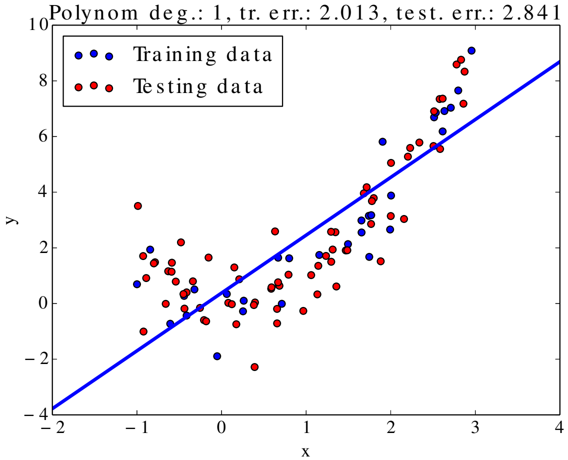


"Just right"
(Good fit)

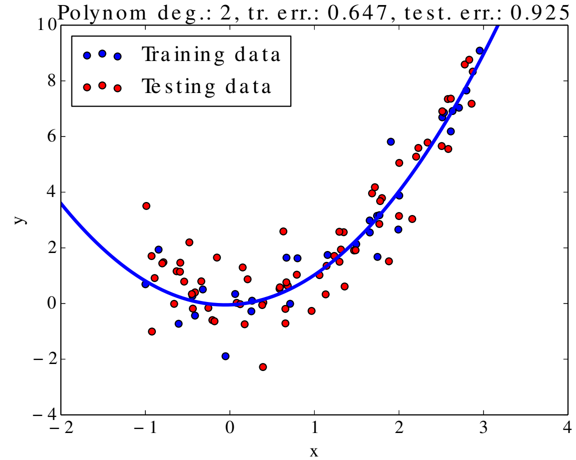


High variance:
model flexibility too high
(Overfit)

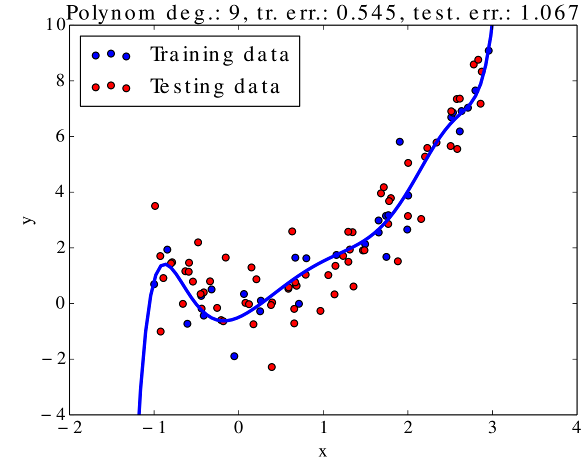
Bias vs Variance



High bias:
model not flexible enough
(Underfit)



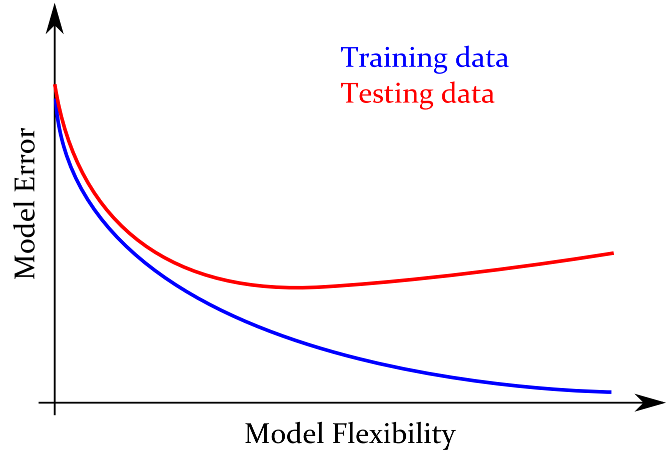
"Just right"
(Good fit)



High variance:
model flexibility too high
(Overfit)

High bias problem:

- $\mathrm{Err}_{\mathrm{Tr}}(h)$ is high
- $\mathrm{Err}_{\mathrm{Tst}}(h) \approx \mathrm{Err}_{\mathrm{Tr}}(h)$



High variance problem:

- $\mathrm{Err}_{\mathrm{Tr}}(h)$ is low
- $\mathrm{Err}_{\mathrm{Tst}}(h) >> \mathrm{Err}_{\mathrm{Tr}}(h)$

Crossvalidation

Simple crossvalidation:

- Split the data into training and testing subsets.
- Train the model on training data.
- Evaluate the model error on testing data.

Crossvalidation

Simple crossvalidation:

- Split the data into training and testing subsets.
- Train the model on training data.
- Evaluate the model error on testing data.

K-fold crossvalidation:

- Split the data into k folds (k is usually 5 or 10).
- In each iteration:
  - Use $k-1$ folds to train the model.
  - Use 1 fold to test the model, i.e. measure error.

Iter. 1	Training	Training	Testing
Iter. 2	Training	Testing	Training
Iter. k	Testing	Training	Training

- Aggregate (average) the k error measurements to get the final error estimate.
- Train the model on the whole data set.

Crossvalidation

Simple crossvalidation:

- Split the data into training and testing subsets.
- Train the model on training data.
- Evaluate the model error on testing data.

K-fold crossvalidation:

- Split the data into k folds (k is usually 5 or 10).
- In each iteration:
  - Use $k-1$ folds to train the model.
  - Use 1 fold to test the model, i.e. measure error.

Iter. 1	Training	Training	Testing
Iter. 2	Training	Testing	Training
Iter. k	Testing	Training	Training

- Aggregate (average) the k error measurements to get the final error estimate.
- Train the model on the whole data set.

Leave-one-out (LOO) crossvalidation:

- $k = |T|$, i.e. the number of folds is equal to the training set size.
- Time consuming for large m.

How to determine a suitable model flexibility

Using simple crossvalidation:

1. *Training data:* use cca 50 % of data for model building.
2. *Validation data:* use cca 25 % of data to search for the suitable model flexibility.
3. Train the suitable model on training + validation data.
4. *Testing data:* use cca 25 % of data for the final estimate of the model error.

How to determine a suitable model flexibility

Using simple crossvalidation:

1. *Training data:* use cca 50 % of data for model building.
2. *Validation data:* use cca 25 % of data to search for the suitable model flexibility.
3. Train the suitable model on training + validation data.
4. *Testing data:* use cca 25 % of data for the final estimate of the model error.

Using *k*-fold crossvalidation

1. *Training data:* use cca 75 % of data to find and train a suitable model using crossvalidation.
2. *Testing data:* use cca 25 % of data for the final estimate of the model error.

How to determine a suitable model flexibility

Using simple crossvalidation:

1. *Training data:* use cca 50 % of data for model building.
2. *Validation data:* use cca 25 % of data to search for the suitable model flexibility.
3. Train the suitable model on training + validation data.
4. *Testing data:* use cca 25 % of data for the final estimate of the model error.

Using k-fold crossvalidation

1. *Training data:* use cca 75 % of data to find and train a suitable model using crossvalidation.
2. *Testing data:* use cca 25 % of data for the final estimate of the model error.

The ratios are not set in stone, there are other possibilities, e.g. 60:20:20, etc.

How to prevent overfitting?

1. Reduce number of features.
   - Select manually, which features to keep.
   - Try to identify a suitable subset of features during learning phase.
2. Regularization
   - Keep all features, but reduce the magnitude of parameters $\boldsymbol{w}$.
   - Works well, if we have a lot of features each of which contributes a bit to predicting y.

Regularization

Ridge regularization (a.k.a. Tikhonov regularization)

Ridge regularization penalizes the size of the model coefficients:

- Modification of the optimization criterion:

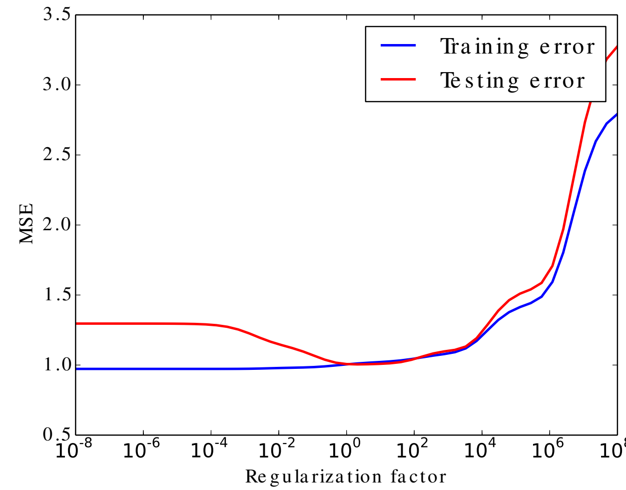
$$J(\boldsymbol{w}) = \frac{1}{|T|}\sum_{i=1}^{|T|}\left(y^{(i)} - h_{\boldsymbol{w}}(\boldsymbol{x}^{(i)})\right)^2 + \alpha \sum_{d=1}^{D} w_d^2.$$

- The solution is given by a modified **Normal equation**

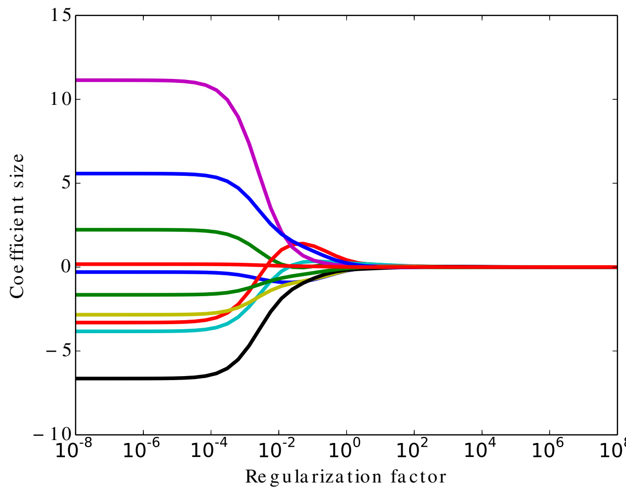
$$\boldsymbol{w}^* = (\boldsymbol{X}^T\boldsymbol{X} + \alpha\mathbf{I})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

- As $\alpha \to 0$, $\boldsymbol{w}^{\text{ridge}} \to \boldsymbol{w}^{\text{OLS}}$.
- As $\alpha \to \infty$, $\boldsymbol{w}^{\text{ridge}} \to 0$.

Training and testing errors as functions of regularization parameter:



The values of coefficients as functions of regularization parameter:



Lasso regularization

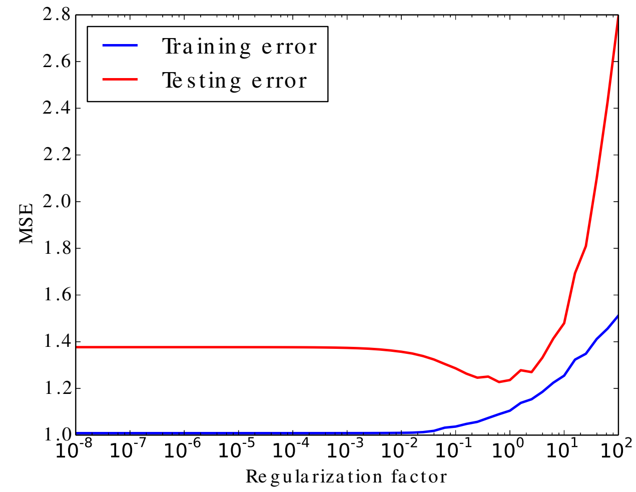
Lasso regularization penalizes the size of the model coefficients:

- Modification of the optimization criterion:

$$J(\boldsymbol{w}) = \frac{1}{|T|}\sum_{i=1}^{|T|}\left(y^{(i)} - h_{\boldsymbol{w}}(\boldsymbol{x}^{(i)})\right)^2 + \alpha \sum_{d=1}^{D} |w_d|.$$

- Solution is usually found by quadratic programming.
- As $\alpha \to \infty$, Lasso regularization *decreases the number of non-zero coefficients*.

Training and testing errors as functions of regularization parameter:



The values of coefficients as functions of regularization parameter: