# Optimization for Deep Learning: Overfitting


**Industrial AI Lab.**

**Prof. Seungchul Lee**

# Overfitting

- You want to hire someone, and you evaluate candidates by asking them ten technical yes/no questions.

- Would you feel confident if you interviewed one candidate and he makes a perfect score?

- What about interviewing ten candidates and picking the best? What about interviewing one thousand?

# Overfitting Example

- A simple classification procedure is the "K-nearest neighbors."

- Given

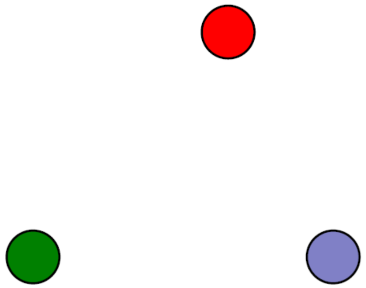
$$(x_n, y_n) \in \mathbb{R}^D \times \{1, \ldots, C\}, \;\; n = 1, \ldots, N$$

- to predict the $y$ associated to a new $x$, take the $y_n$ of the closest $x_n$:

$$\begin{aligned} n^*(x) &= \underset{n}{\operatorname{argmin}} \|x_n - x\| \\ f^*(x) &= y_{n^*(x)}. \end{aligned}$$

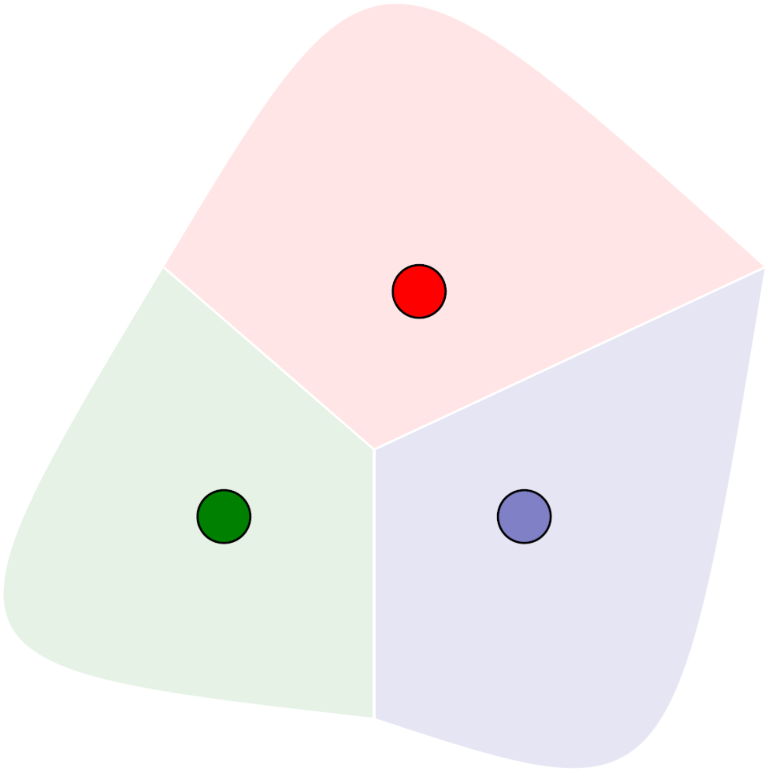
- This recipe corresponds to $K = 1$, and makes the empirical training error zero

# Overfitting Example



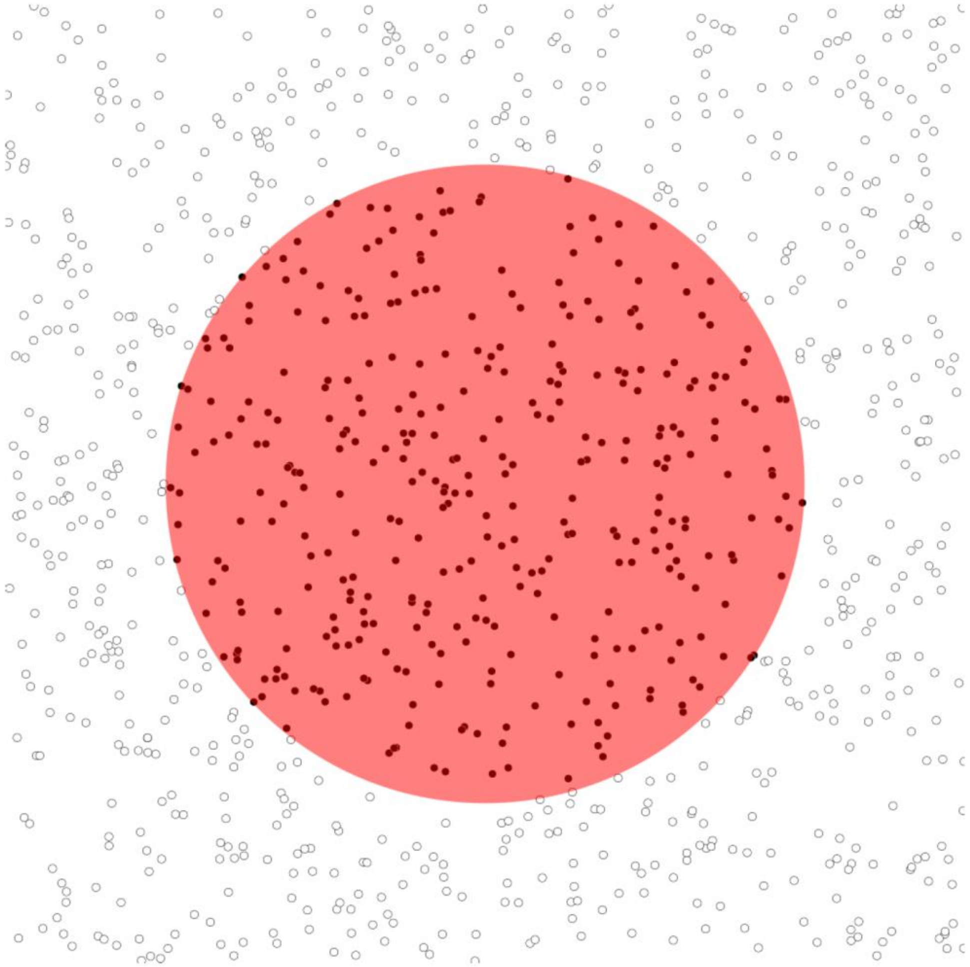
# Overfitting Example

- K = 1



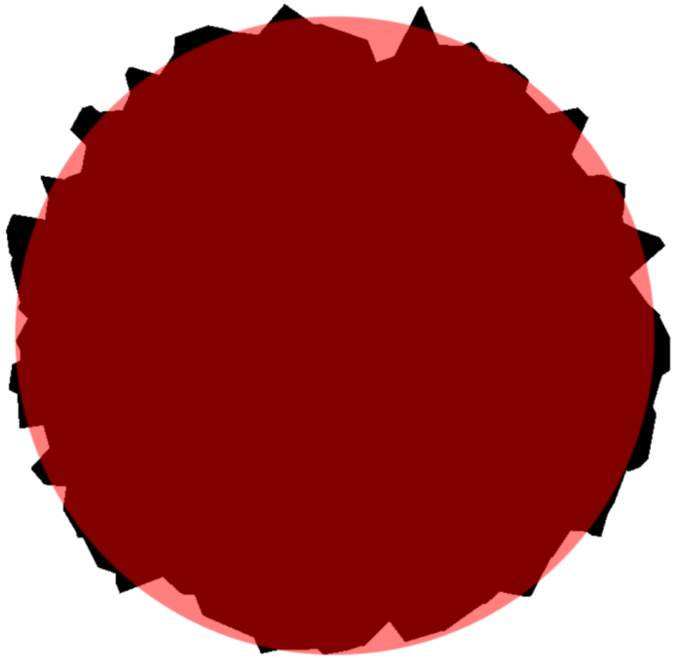
K = 1

# Overfitting Example



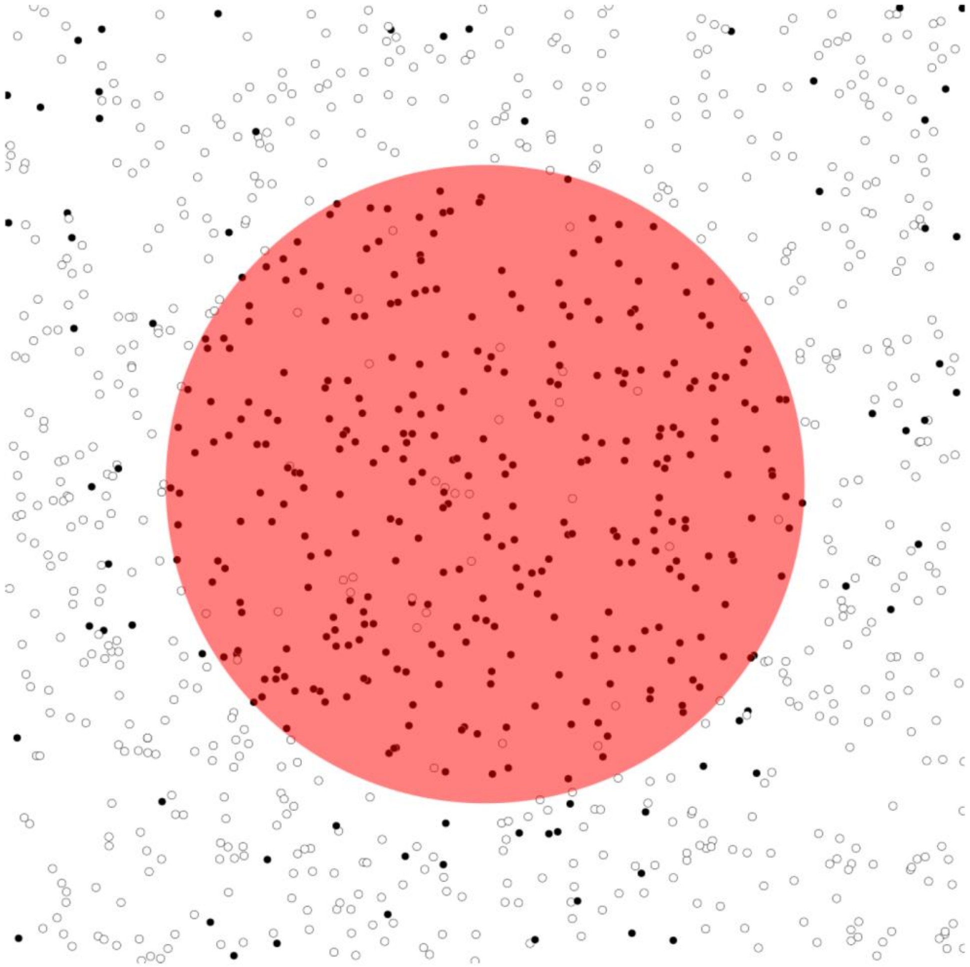
Training set

- K = 1
- Too noisy



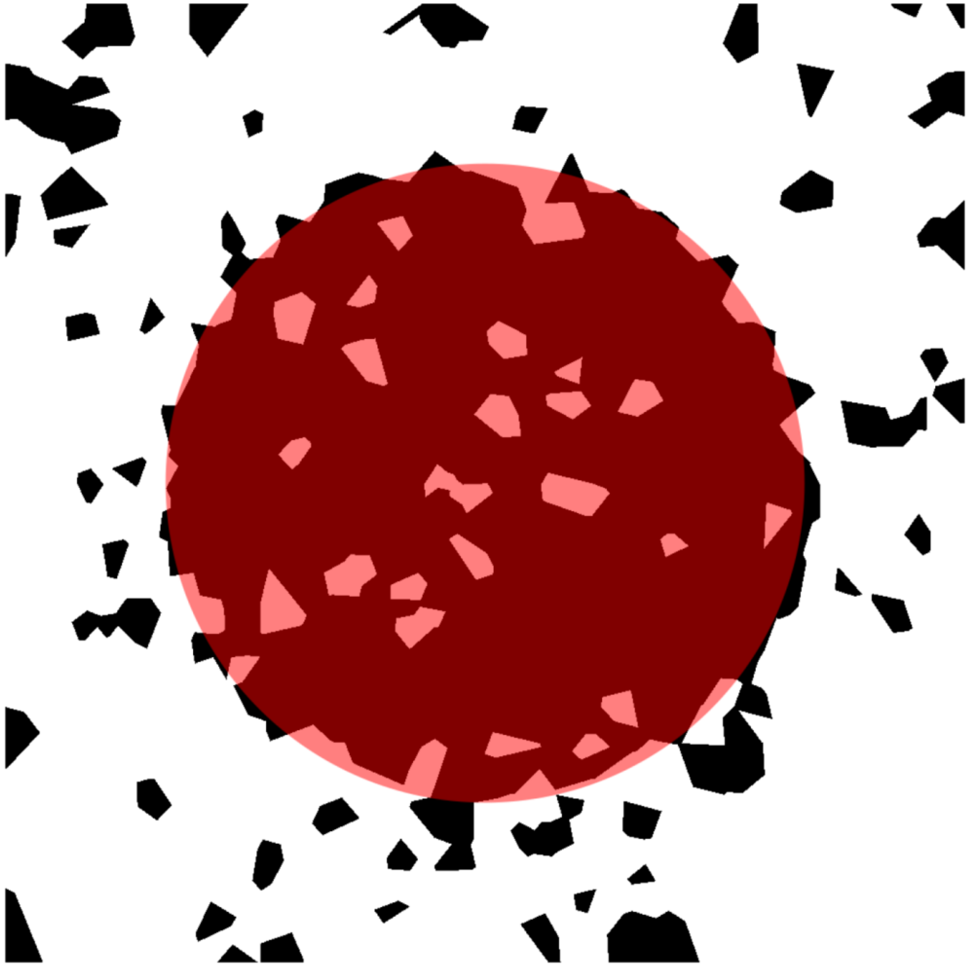
Prediction (K=1)

- K = 1
- With outliers



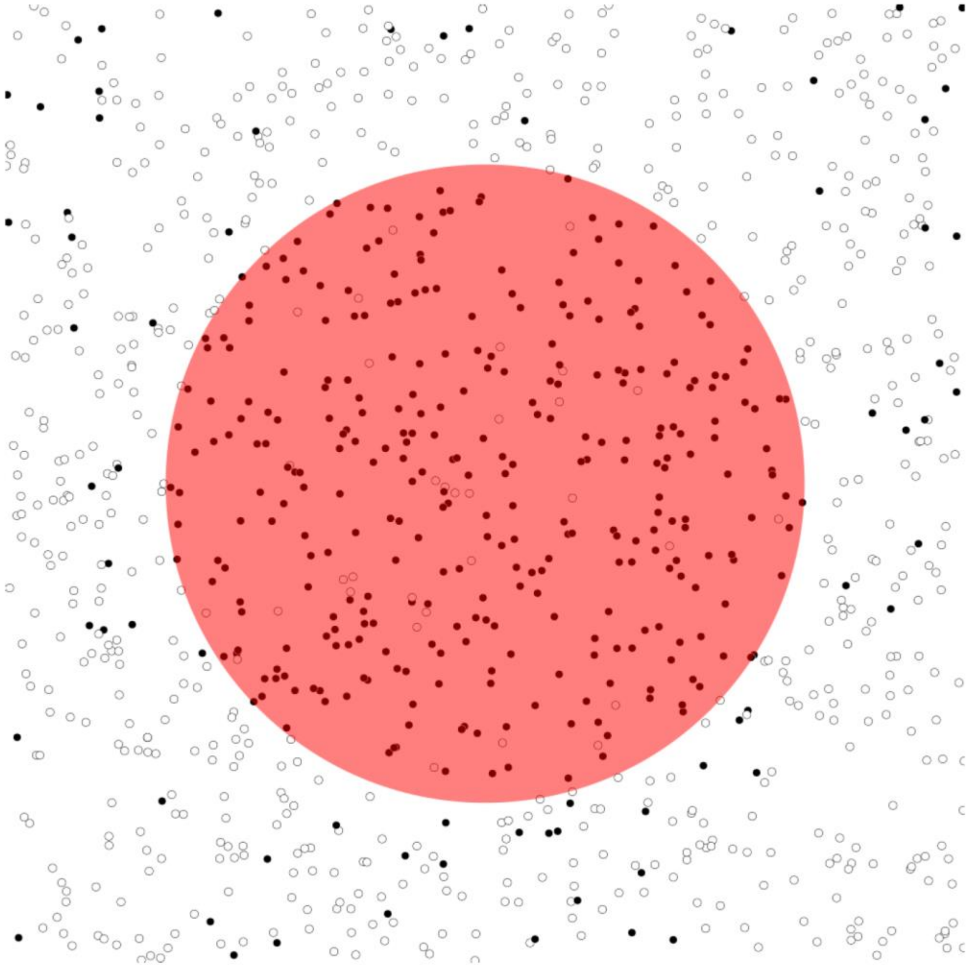
Training set

- K = 1
- With outliers



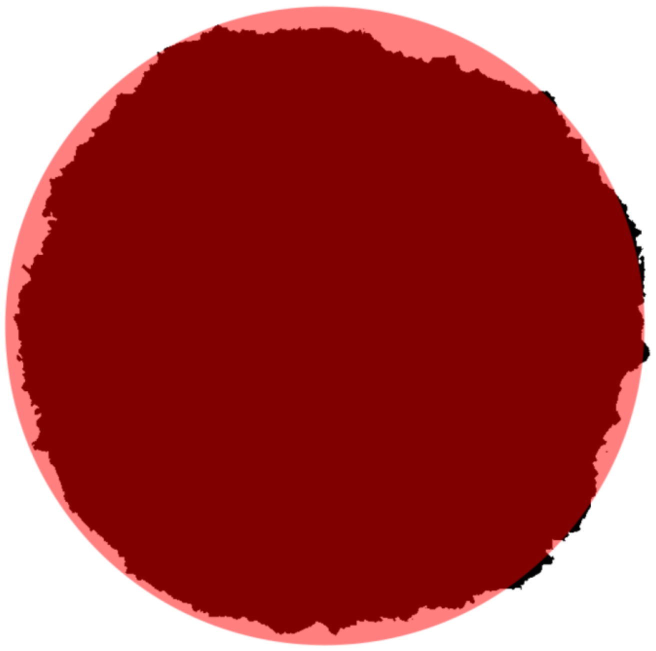
Prediction (K=1)

- K = 51
- With outliers



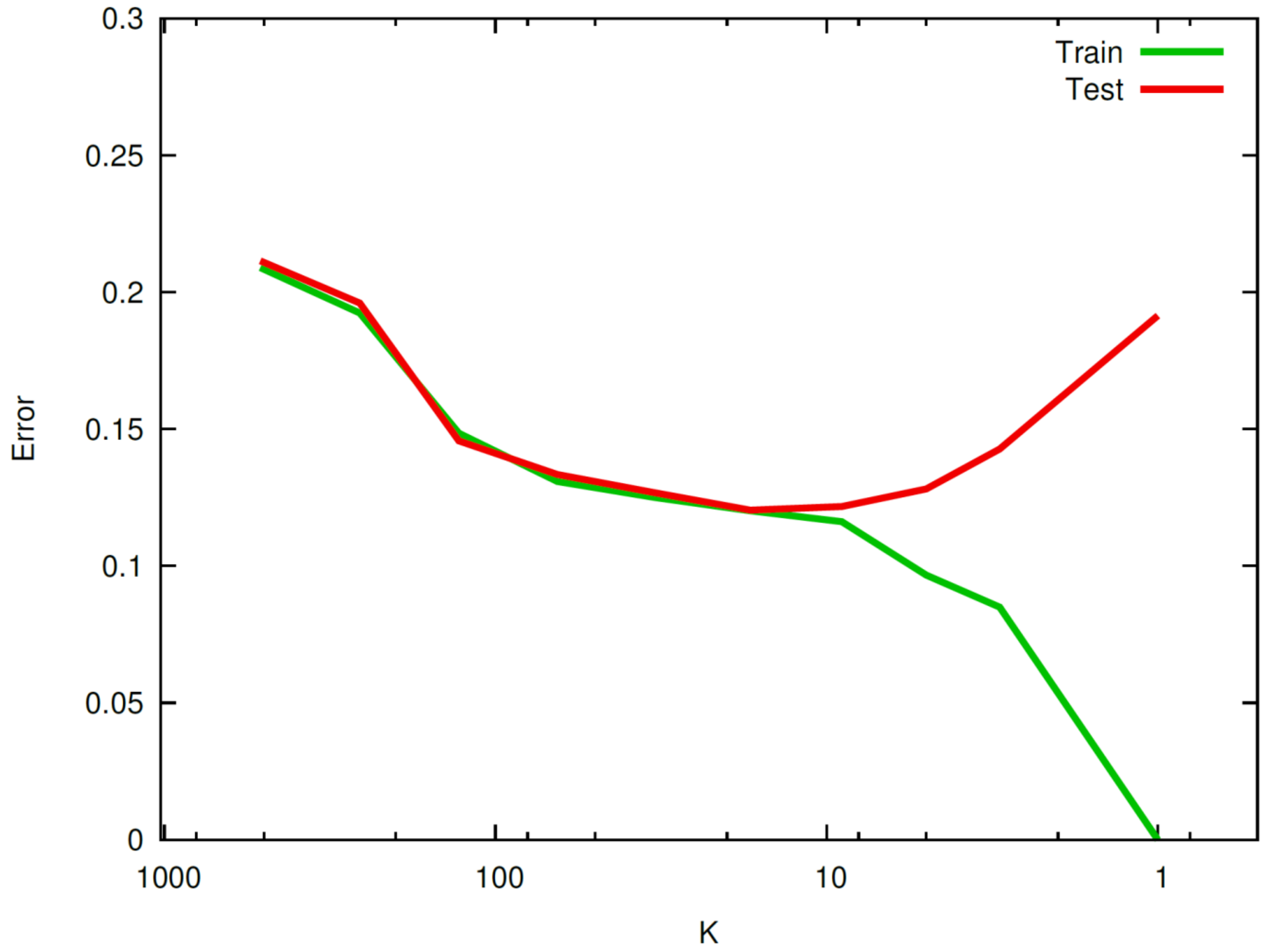
Training set

- K = 52
- With outliers
- Robust and smooth

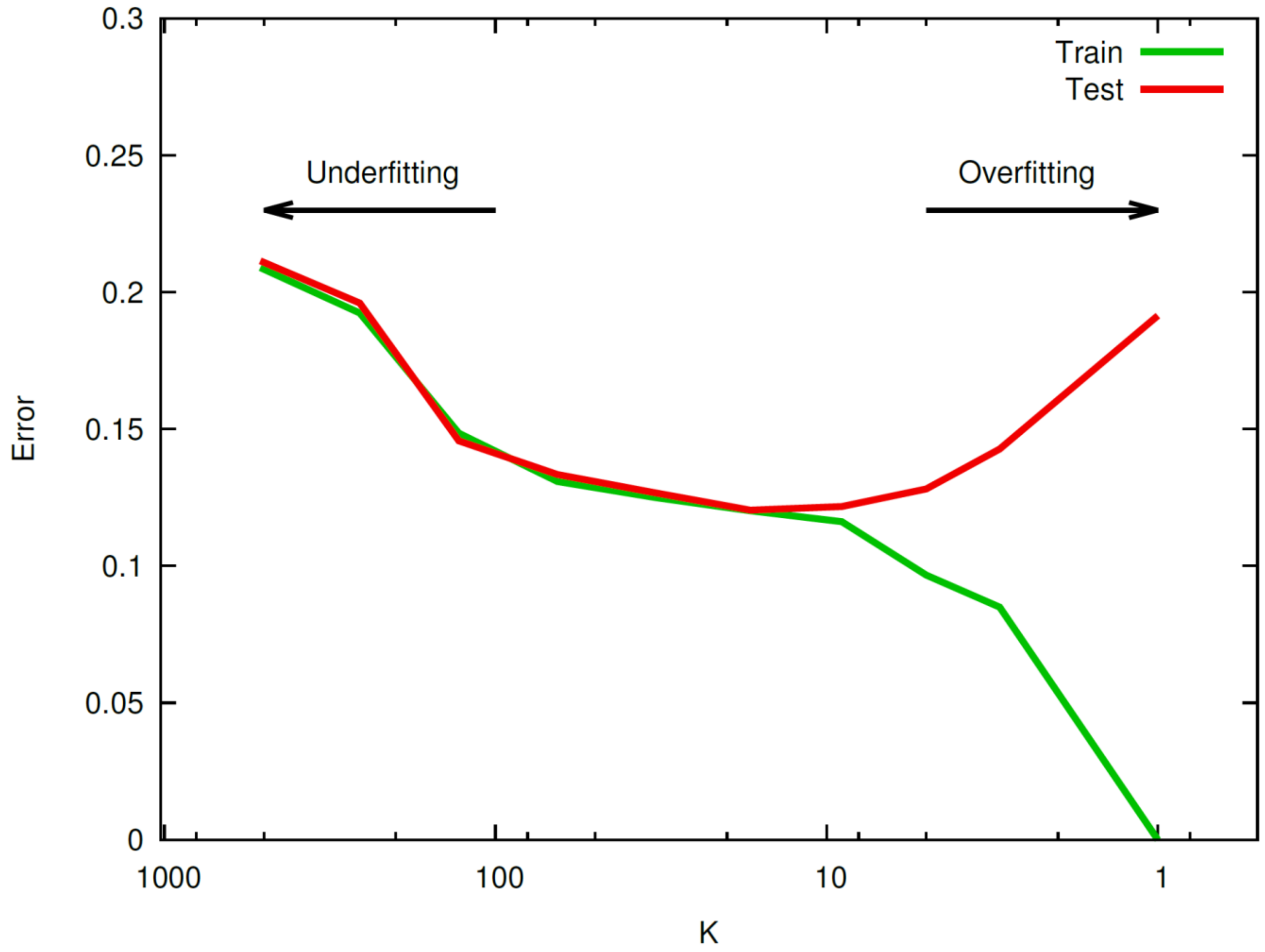


Prediction (K=51)

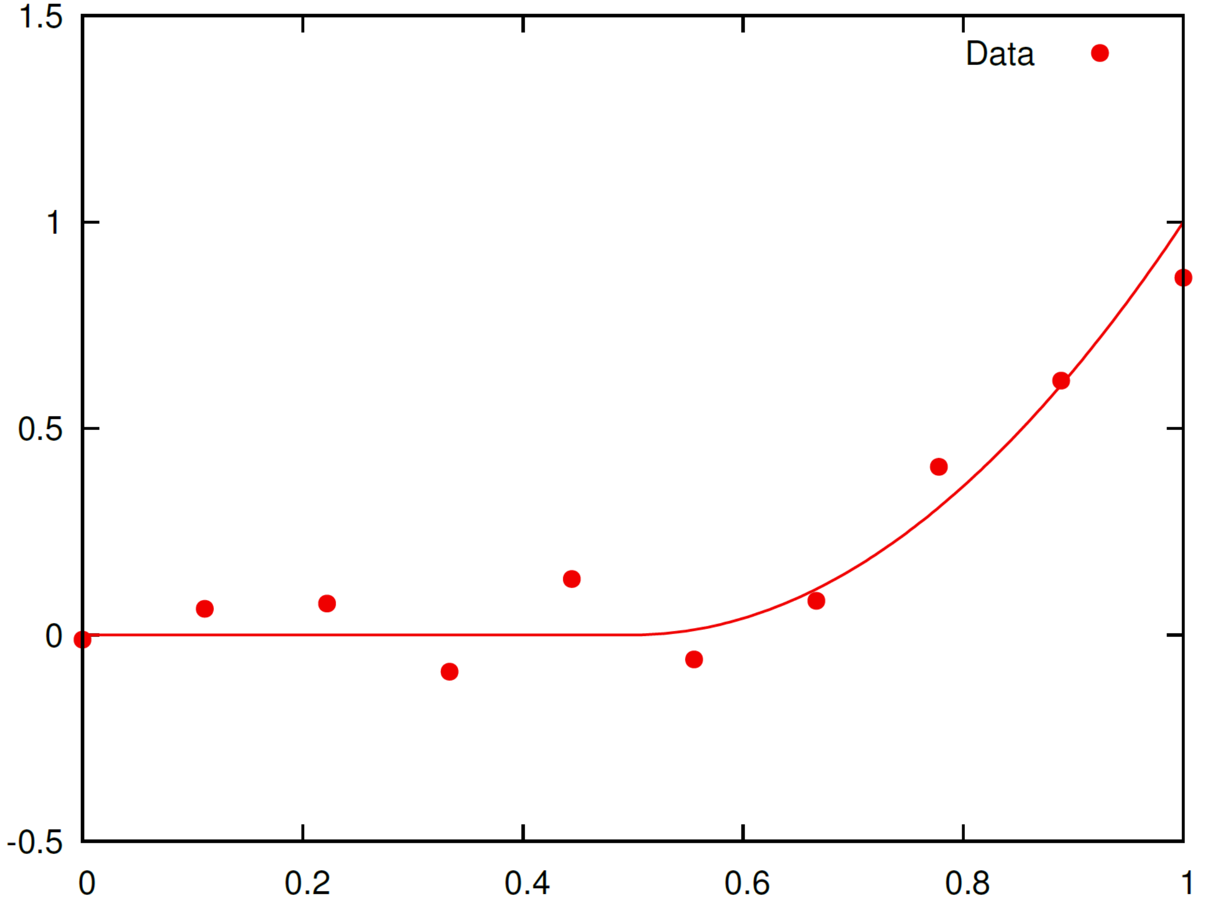
# Errors on Train and Test Datasets

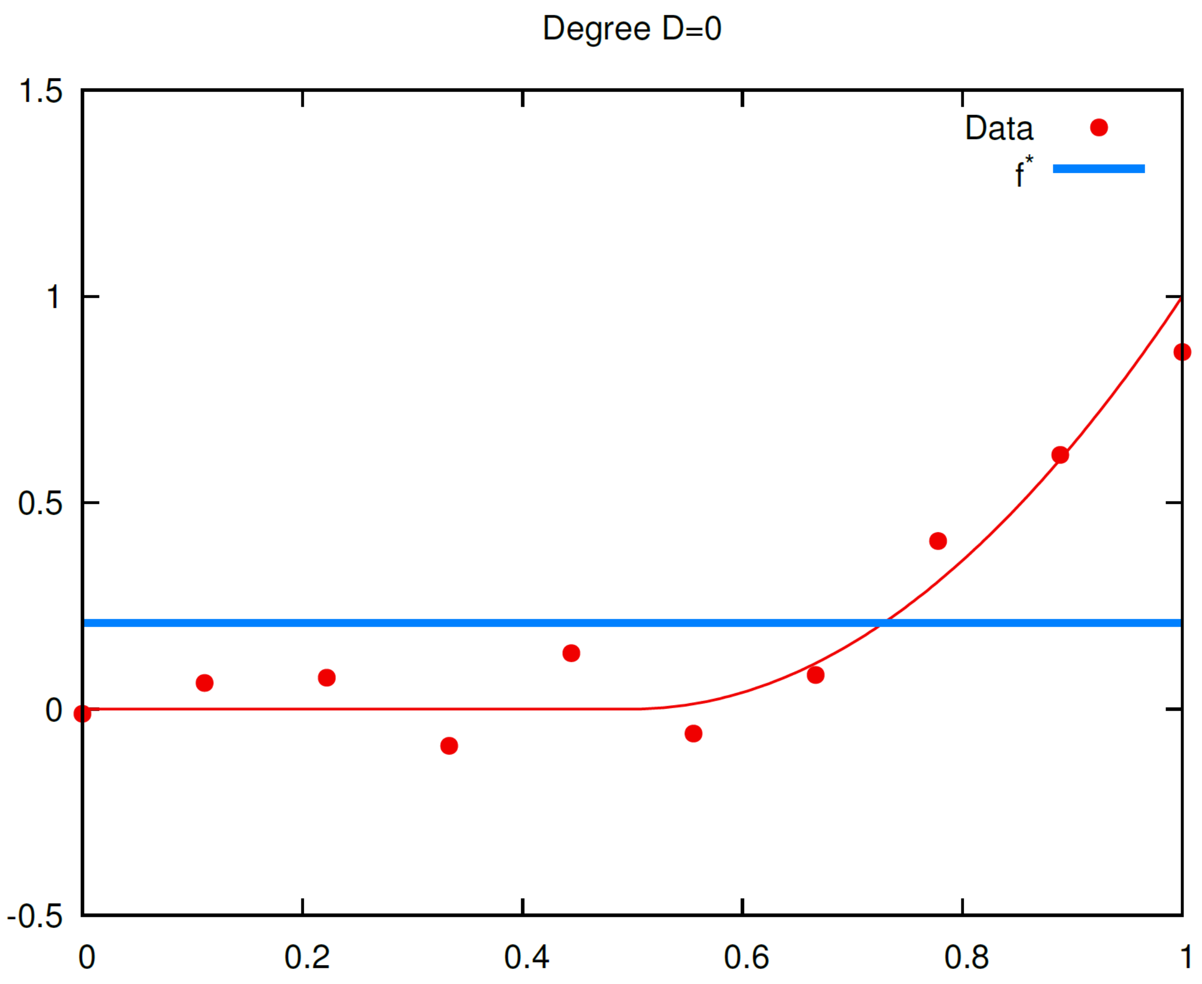


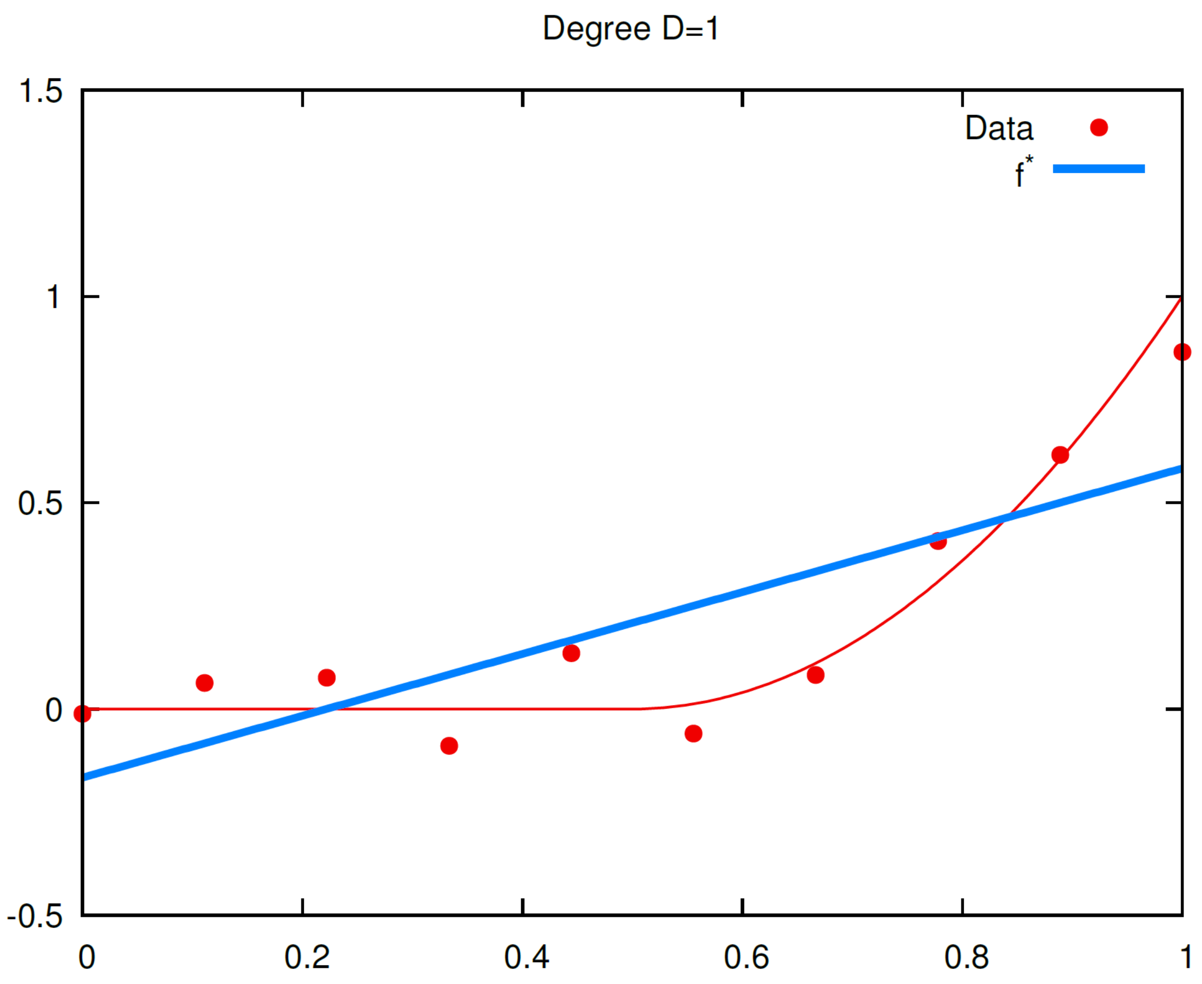
# Errors on Train and Test Datasets

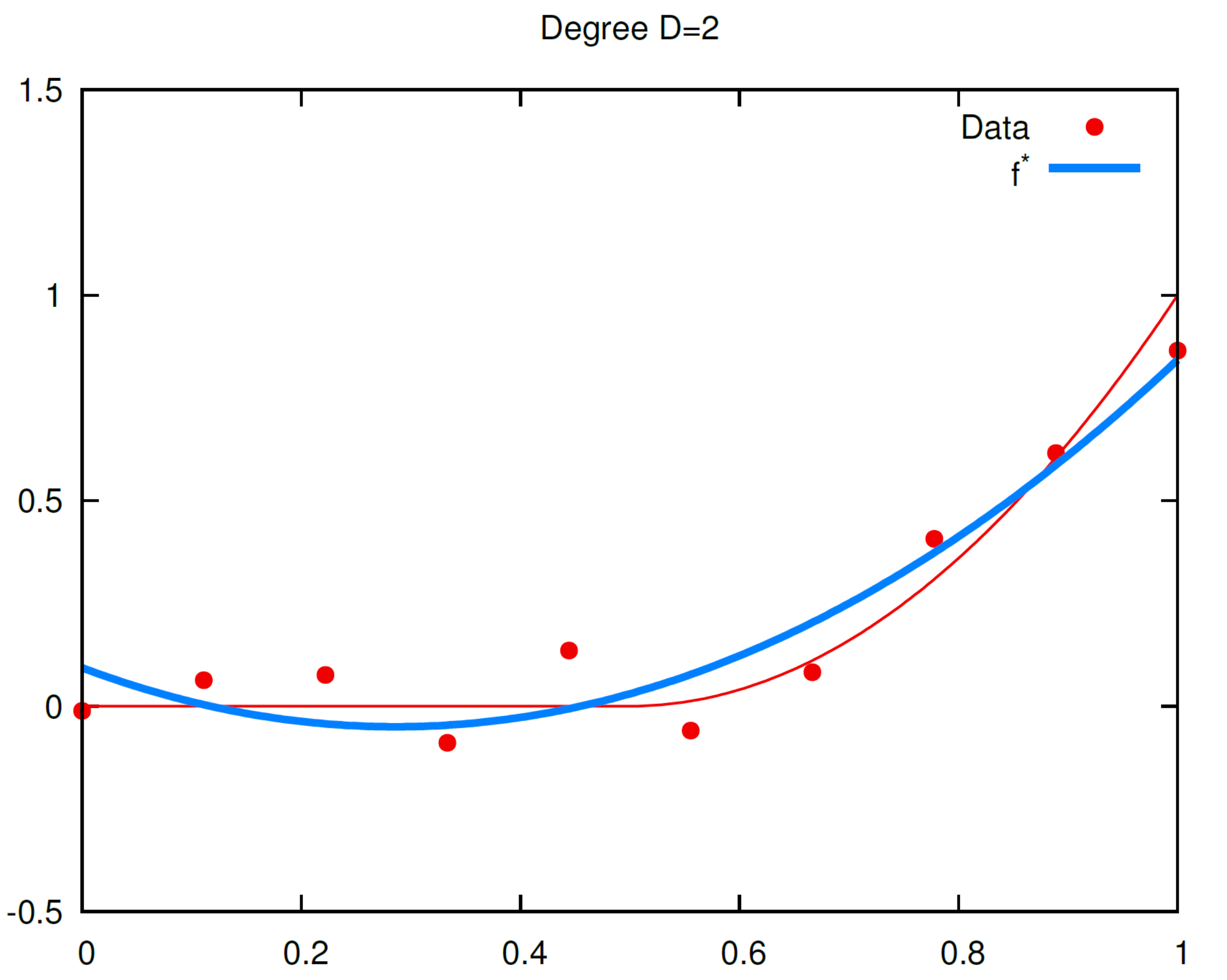


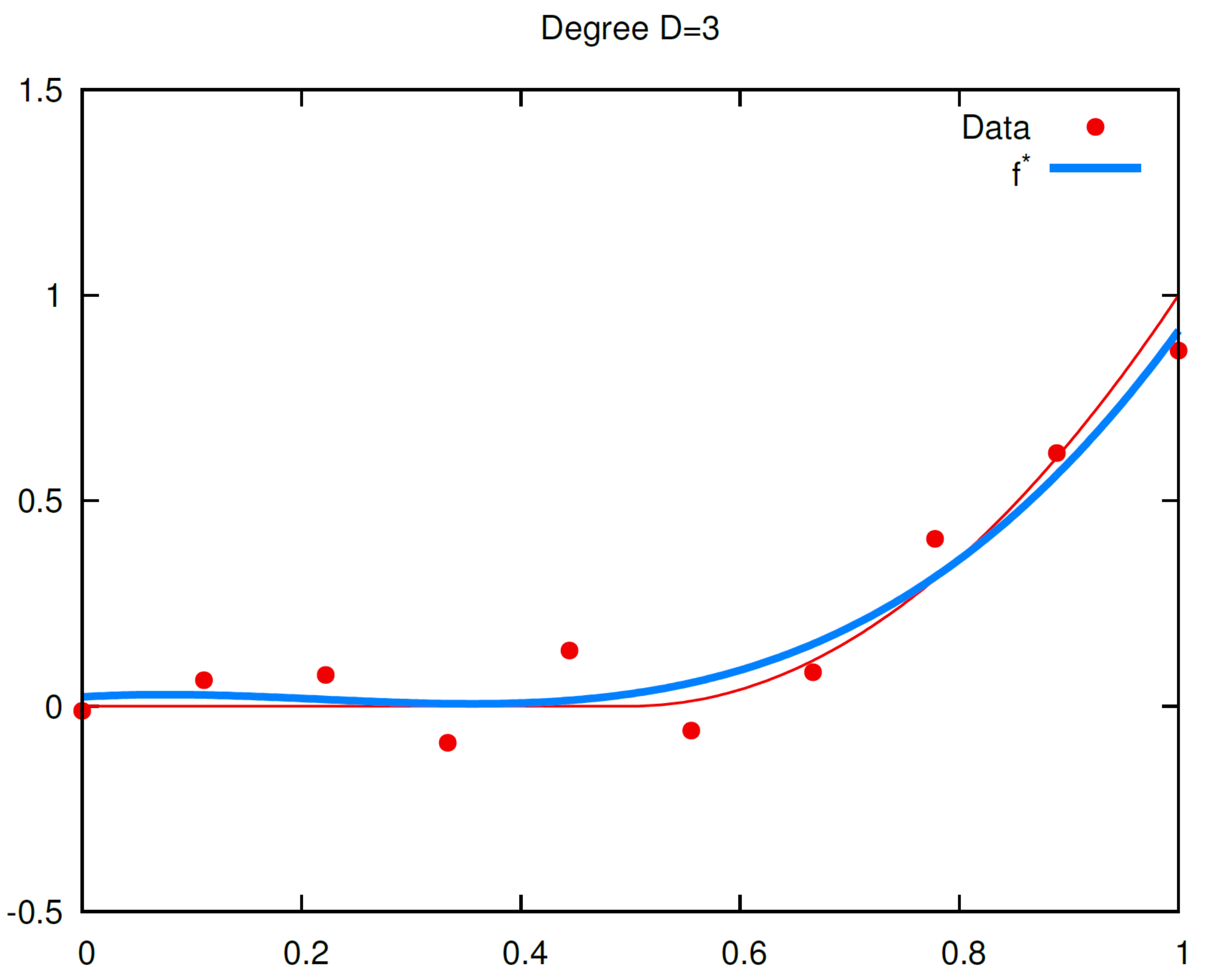
# Polynomial Regression



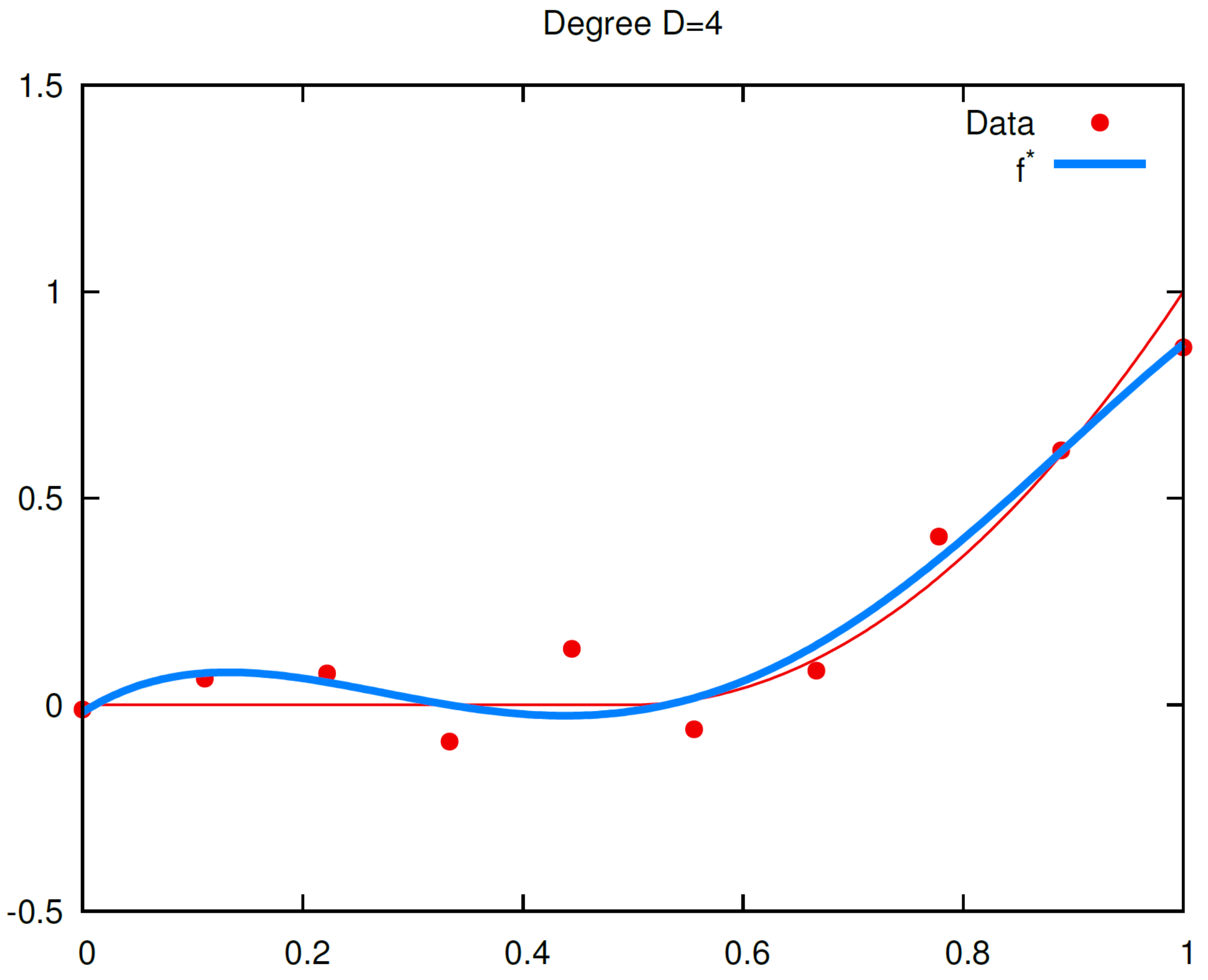

Degree D=0
Data
f*
1.5
1
0.5
0
-0.5
0
0.2
0.4
0.6
0.8
1

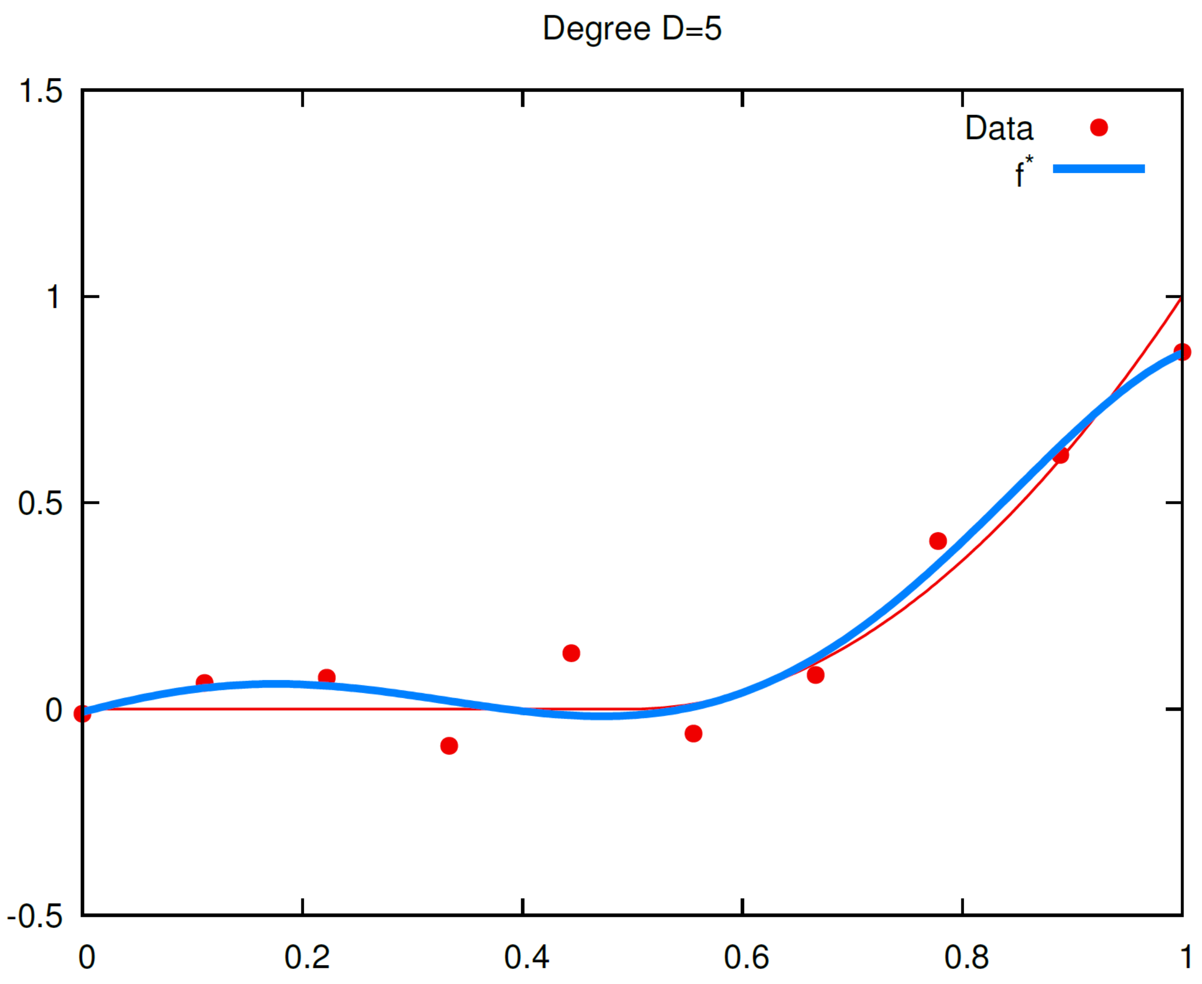

Degree D=1
Data
f*
1.5
1
0.5
0
-0.5
0
0.2
0.4
0.6
0.8
1


Degree D=2
Data
f*
1.5
1
0.5
0
-0.5
0
0.2
0.4
0.6
0.8
1

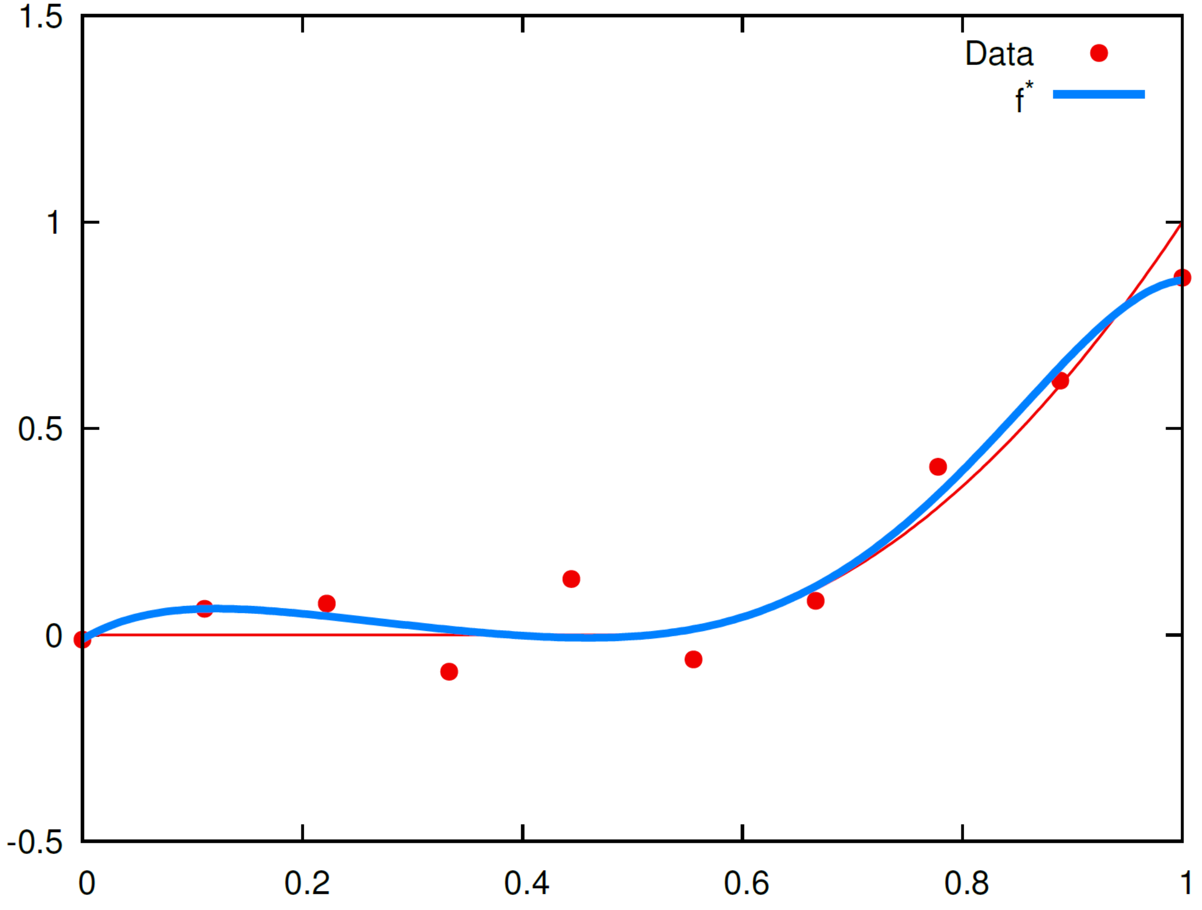


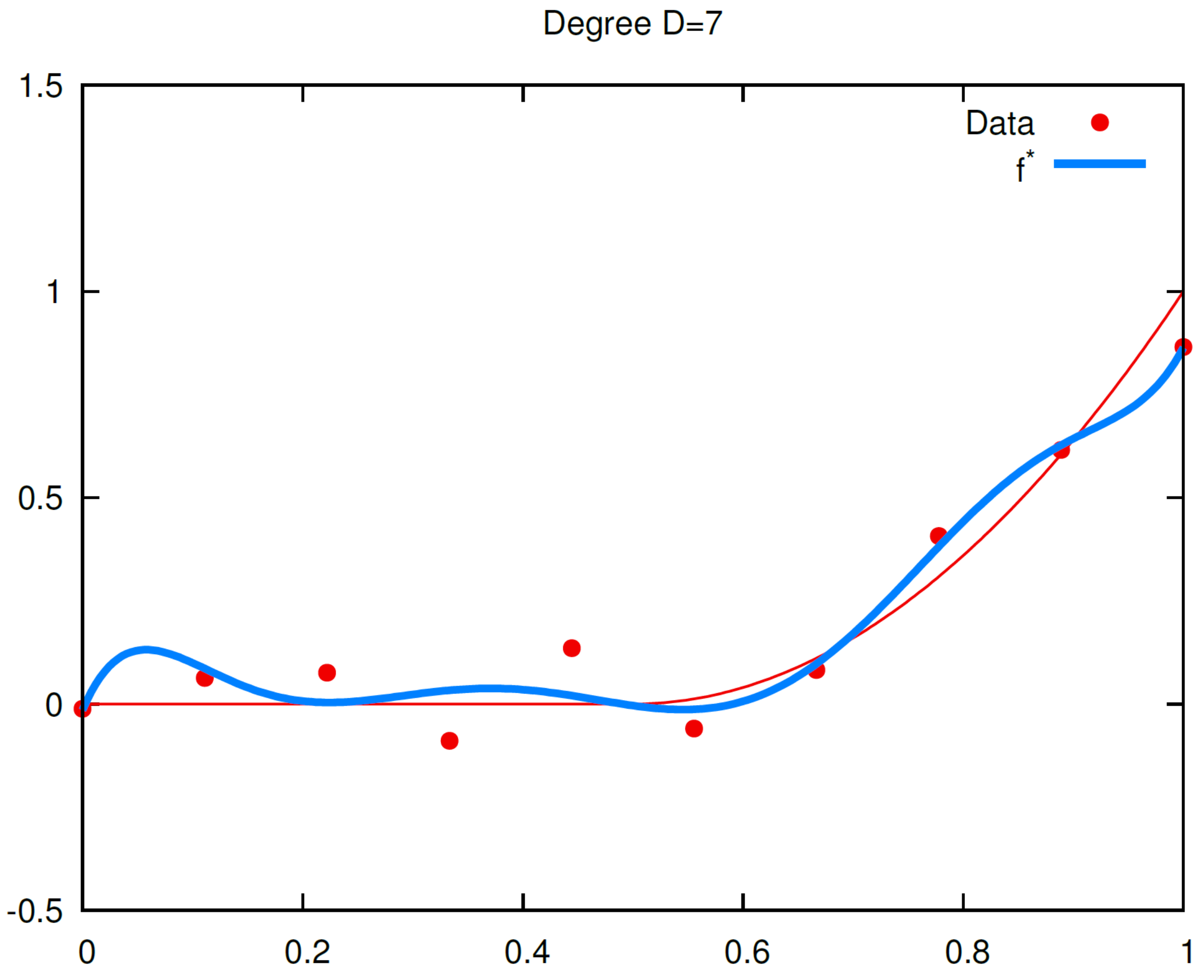

POSTECH


Degree D=4
1.5
1
0.5
0
-0.5
0
0.2
0.4
0.6
0.8
1
Data
f*

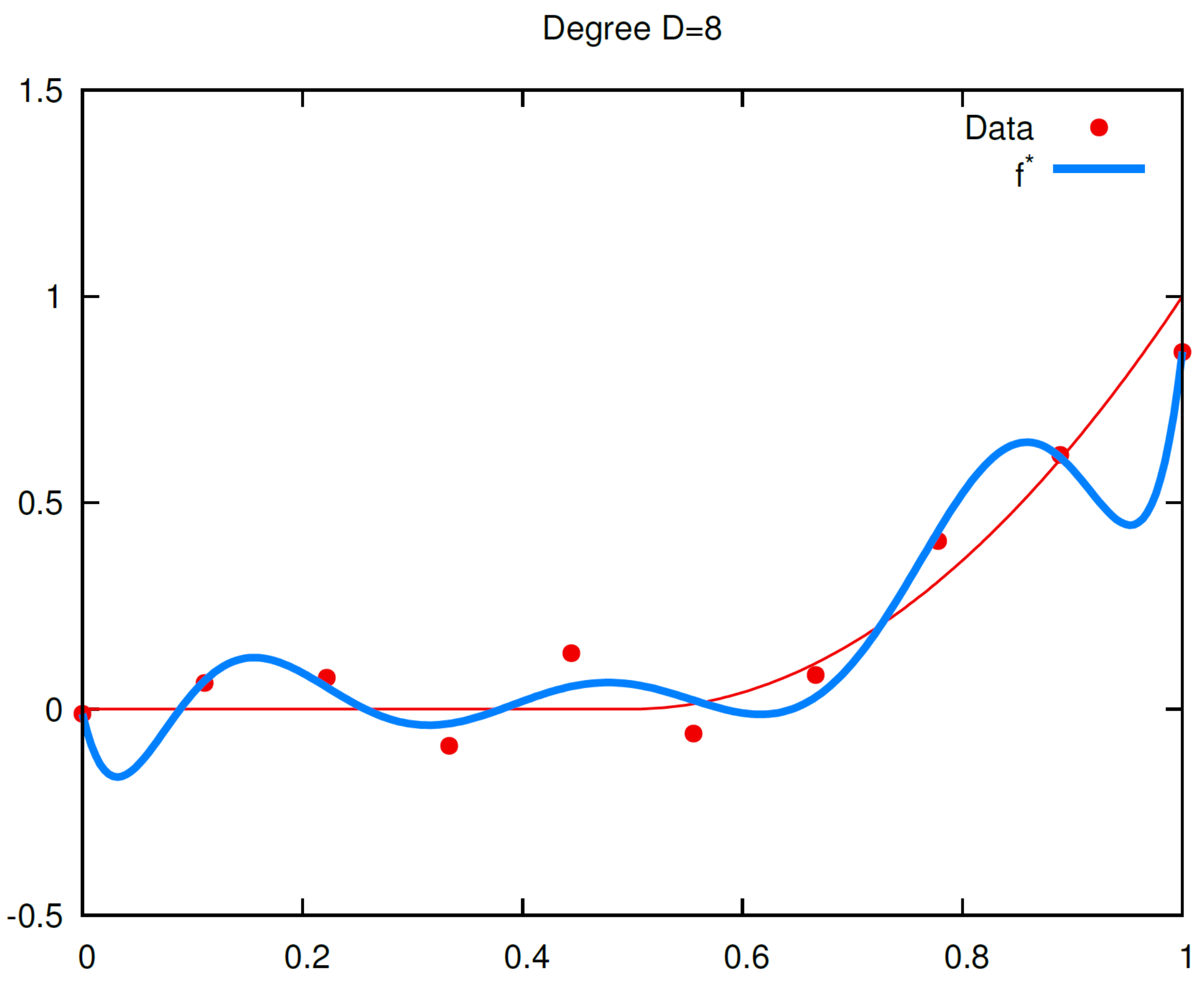

Degree D=5
1.5
1
0.5
0
-0.5
0
0.2
0.4
0.6
0.8
1
Data
f*

Degree D=6

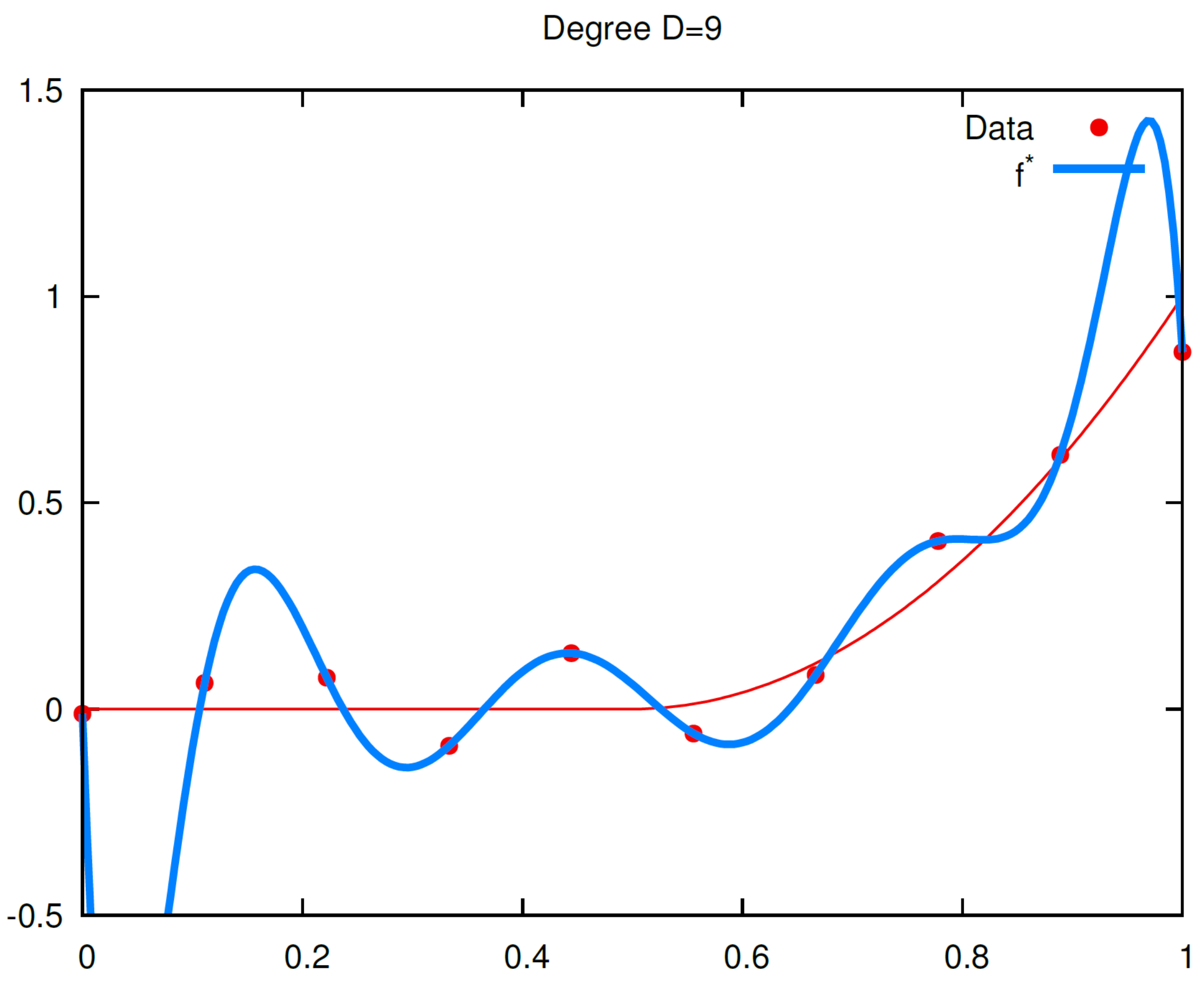




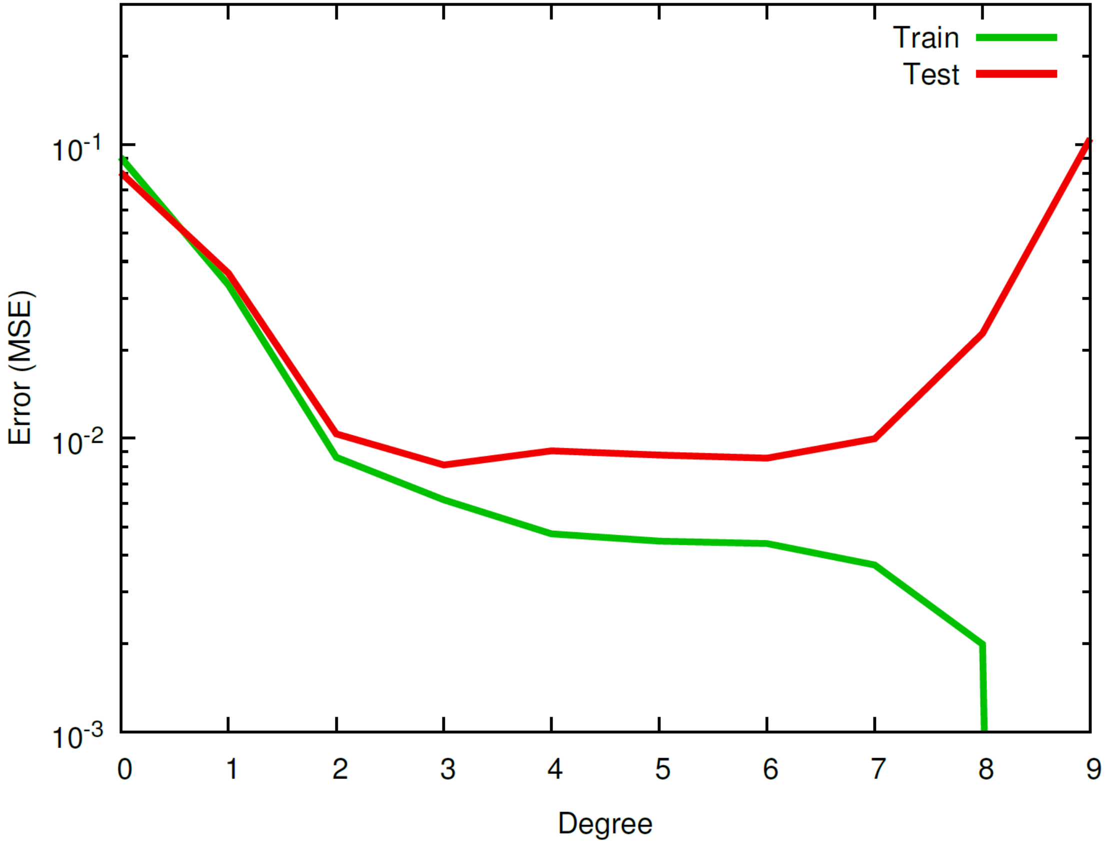
Degree D=7
Data
f*
1.5
1
0.5
0
-0.5
0
0.2
0.4
0.6
0.8
1



Degree D=8
Data
$f^*$
1.5
1
0.5
0
-0.5
0
0.2
0.4
0.6
0.8
1


Degree D=9
1.5
1
0.5
0
-0.5
0
0.2
0.4
0.6
0.8
1
Data
f*

# Errors on Train and Test Datasets



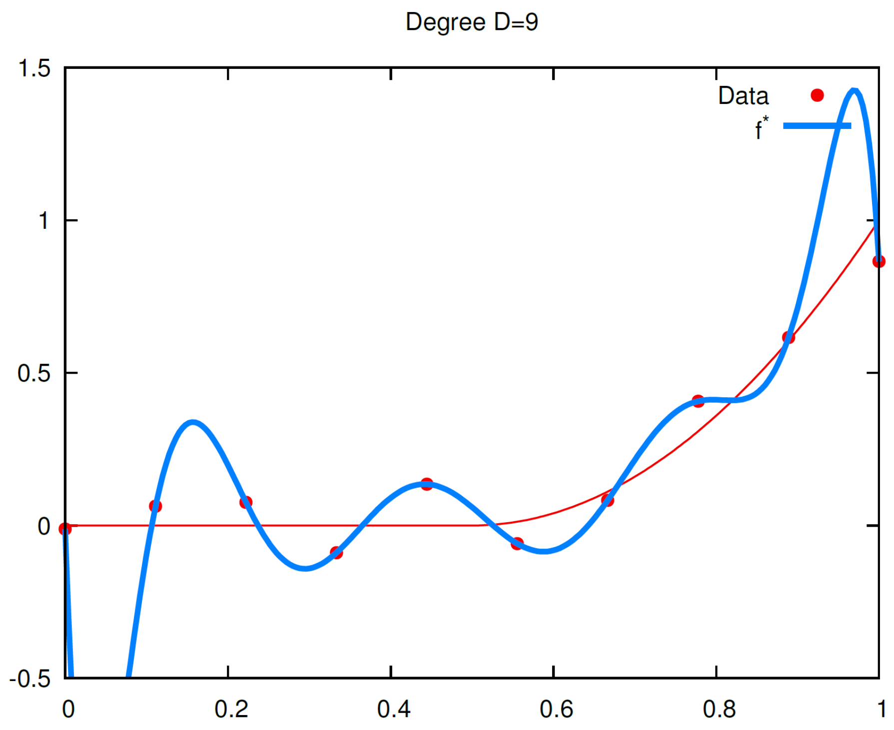
# Overfitting Problem

- One of the most common problem data science professionals face is to avoid overfitting.

- Have you come across a situation where your model performed exceptionally well on train data, but was not able to predict test data.

# Issue with Rich Representation

- Low error on input data points, but high error nearby
- Low error on training data, but high error on testing data

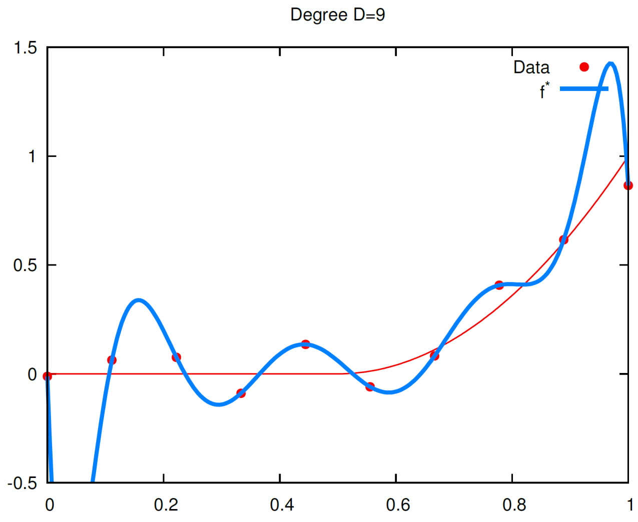


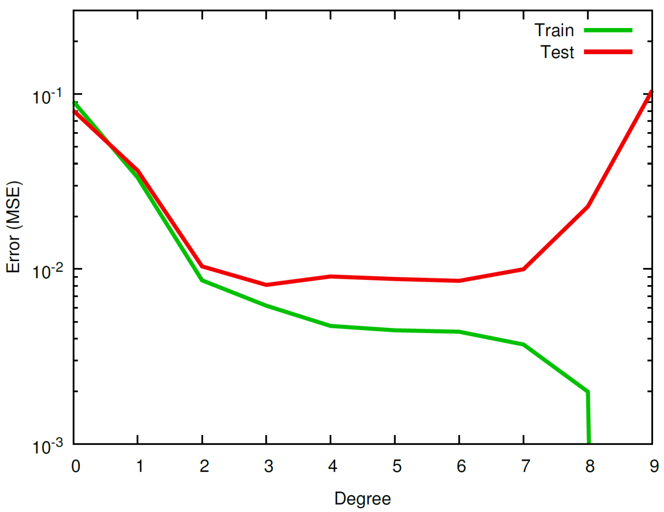
# Generalization Error

- Fundamental problem: we are optimizing parameters to solve

$$\min_{\theta} \sum_{i=1}^{m} \ell(y_i, \hat{y}_i) = \min_{\theta} \sum_{i=1}^{m} \ell(y_i, \Phi\theta)$$

- But what we really care about is loss of prediction on new data $(x, y)$
  – also called generalization error




- Divide data into training set, and validation (testing) set

# Representational Difficulties

- With many features, prediction function becomes very expressive (model complexity)

  – Choose less expensive function (e.g., lower degree polynomial, fewer RBF centers, larger RBF bandwidth)

  – Keep the magnitude of the parameter small

  – Regularization: penalize large parameters $\theta$

$$\min \ \|\Phi\theta - y\|_2^2 + \lambda\|\theta\|_2^2$$

  – $\lambda$: regularization parameter, trades off between low loss and small values of $\theta$

# Regularization (Shrinkage Methods)

- Often, overfitting associated with very large estimated parameters
- We want to balance
  - how well function fits data
  - magnitude of coefficients

$$\text{Total cost} = \underbrace{\text{measure of fit}}_{RSS(\theta)} + \lambda \cdot \underbrace{\text{measure of magnitude of coefficients}}_{\lambda \cdot \|\theta\|_2^2}$$

$$\implies \min \|\Phi\theta - y\|_2^2 + \lambda\|\theta\|_2^2 \quad \text{or} \quad \min \|\Phi\theta - y\|_2^2 + \lambda\|\theta\|_1$$

$L_2$ regularization — $L_1$ regularization

  - multi-objective optimization
  - $\lambda$ is a tuning parameter

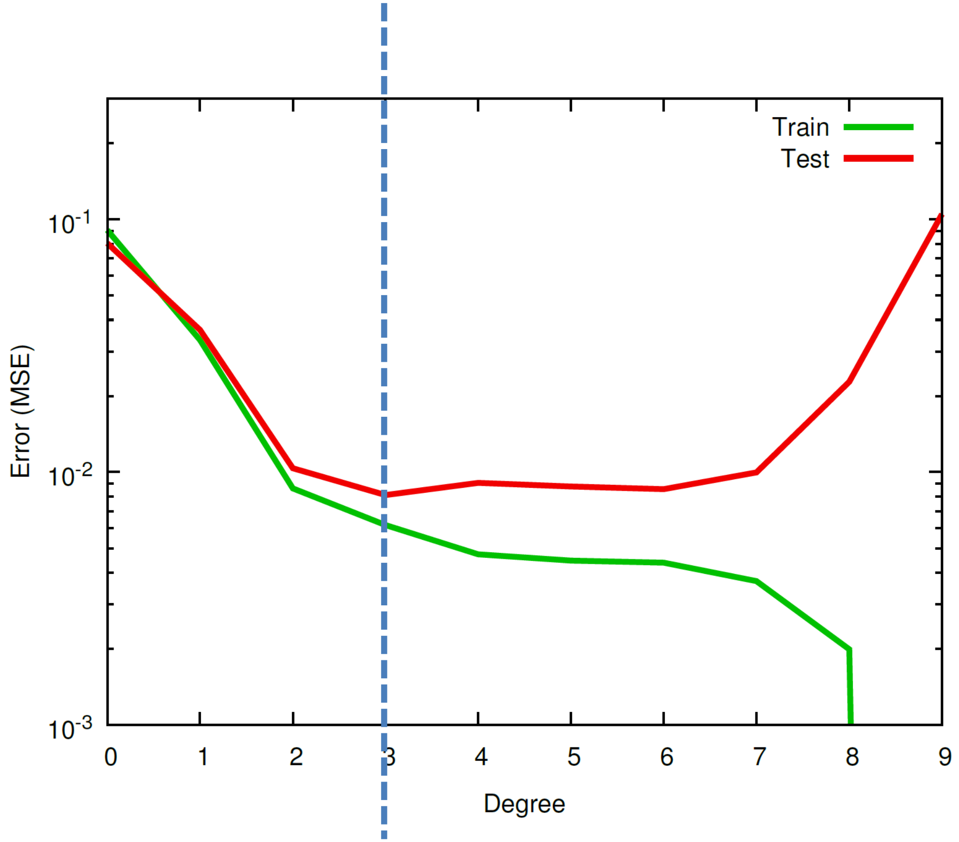
# Different Regularization Techniques in Deep Learning

- $L_2$ and $L_1$ regularization

- Data augmentation
  - The simplest way to reduce overfitting is to increase the size of the training data.

# Different Regularization Techniques in Deep Learning

- Early stopping
  - When we see that the performance on the validation set is getting worse, we immediately stop the training on the model.




Early stopping

# Different Regularization Techniques in Deep Learning

- Dropout
  - This is the one of the most interesting types of regularization techniques.
  - It also produces very good results and is consequently the most frequently used regularization technique in the field of deep learning.
  - At every iteration, it randomly selects some nodes and removes them.
  - It can also be thought of as an ensemble technique in machine learning.
  - (will discuss later)