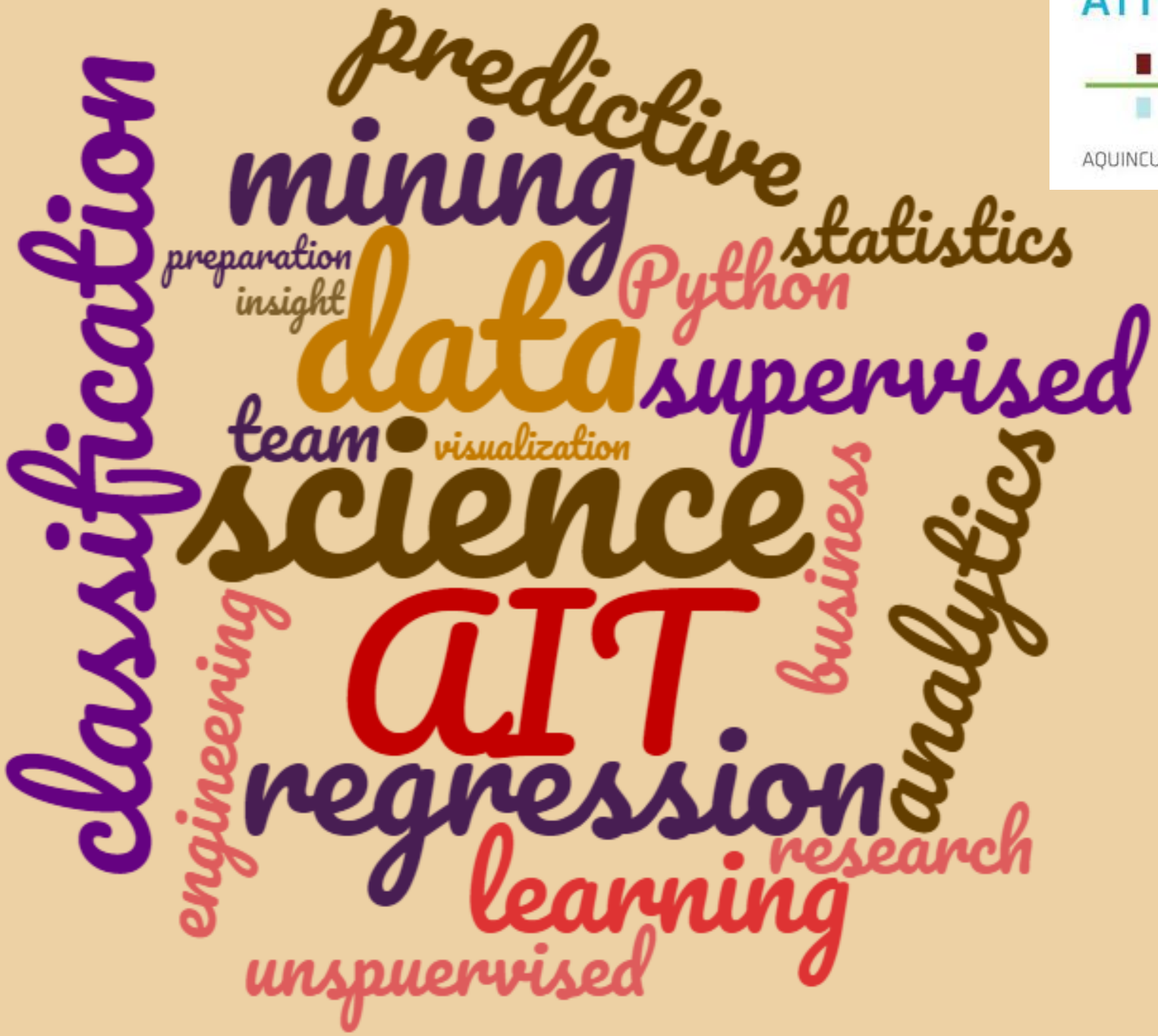
Data Science

March 31, 2020.
Linear regression



AIT-BUDAPEST
AQUINCUM INSTITUTE OF TECHNOLOGY

Roland Molontay

New schedule of the semester

	Tuesday class	*Thursday*	*Friday class*	*Sunday*
W1 (02/03)				
W2 (02/10)			HW1 out	
W3 (02/17)				
W4 (02/24)		HW1 deadline	HW2 out	Forming teams
W5 (03/02)				
W6 (03/09)			CANCELLED	
BREAK	BREAK	HW2 deadline	BREAK	
W7 (13/23)	HW3 out		MIDTERM / Project plan	
W8 (03/30)				HW3 deadline
W9 (04/06)			GOOD FRIDAY	
W10 (04/13)	HW4 out	MILESTONE 1		
W11 (04/20)				HW4 deadline
W12 (04/27)			LABOR DAY	
W13 (05/04)		MILESTONE 2		
W14 (05/11)	FINAL		PROJECT presentations	

Grading

- MIDTERM (25%) + FINAL (25%) + HOMEWORK (25%) + PROJECT (25%)
- Cutoffs for letter grades:
  - 90% - A+
  - 85% - A
  - 80% - A-
  - 75% - B+
  - 70% - B
  - 65% - B-
  - 60% - C+
  - 55% - C
  - And so on…

Regression

- We try to predict continuous valued output based on the values of other variables (via supervised learning)
- Examples:

Explanatory (input) variables	Target (output)
Number of rooms, size, location (ZIP code), …	Market value of a house
Movie budget, film genre, popularity of the actors (based on their IMDB pages)	Box office result of a movie
Major, admission point score, gender, age, GMAT scores, ….	GPA

- Challenges: finding the right explanatory variables, the most suitable functional form/modelling approach

Fundamental task of regression

Let $X = (X_1, X_2, \dots, X_p)$ be the feature vector and Y is the target variable.

Regression: we suppose that there is a relationship between X and Y, in general: $Y = f(X) + \epsilon$, where ϵ (the random error) is independent from X and has zero mean

Aim: giving prediction: $\hat{Y} = \hat{f}(X)$

In reality $\hat{f}$ is sometimes considered to be a black-box, we are not interested in the functional form, but to give accurate enough prediction to Y

Learning: On the labeled data of the training set we estimate the function f, minimizing the „prediction error” on the training set

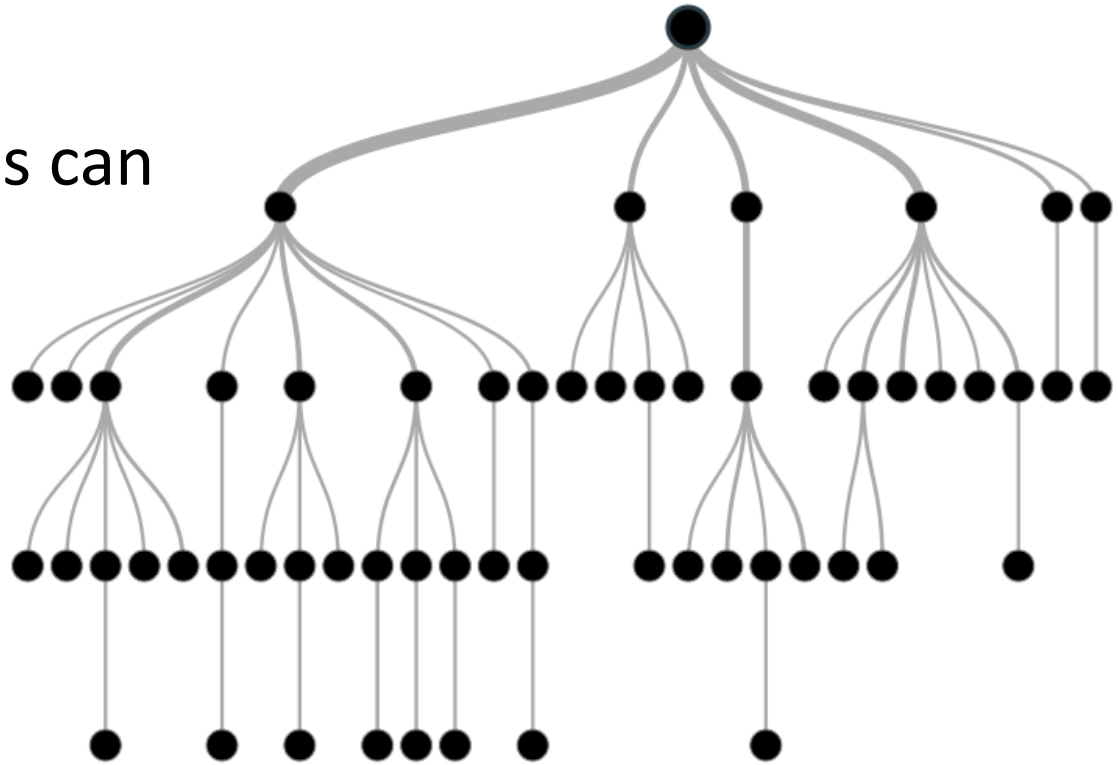
Prediction: using $\hat{f}$ for data that we have not seen before $\hat{Y} = \hat{f}(X)$

Parametric and non-parametric models

- Parametric model
  - We find $f(X)$ in a predefined function family (functional form)
    - E.g. linear, polynomial form
  - The models can be described using a finite number of parameters
    - We optimize for these parameters
- Non-parametric model
  - No functional form is supposed
  - No assumption is needed for the functional form, there is no restriction, the model can better fit the data, the model is more flexible
  - „Infinite dimensional parameter space”
  - To achieve an accurate estimation, more data points are needed than in the parametric case
  - Usually more complex and slower that the parametric case
  - E.g: kNN, decision tree, SVM

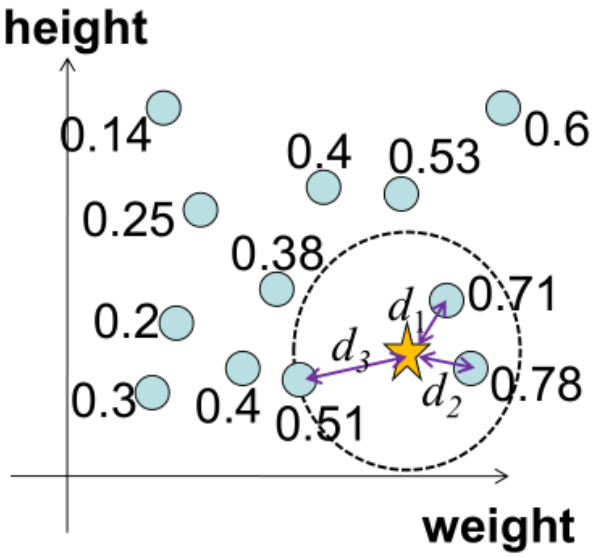
Previously covered methods for regression

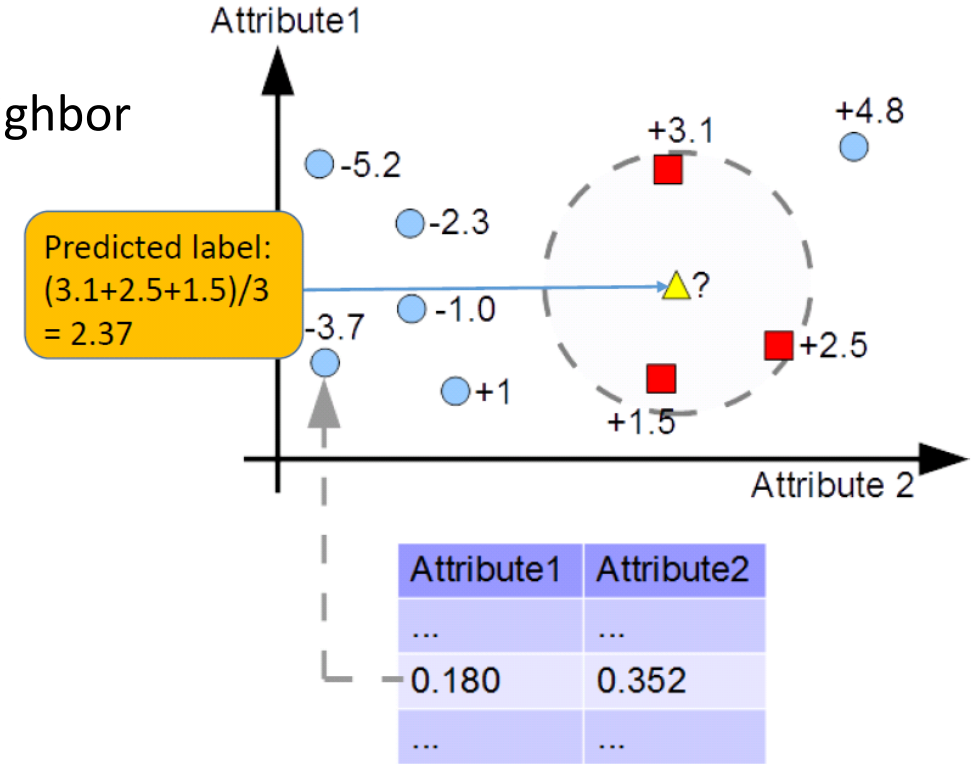
- Until know we have focused on classification problems
- We emphasized that some models can also be used to solve regression problems
  - kNN
  - Decision tree



kNN algorithm for regression

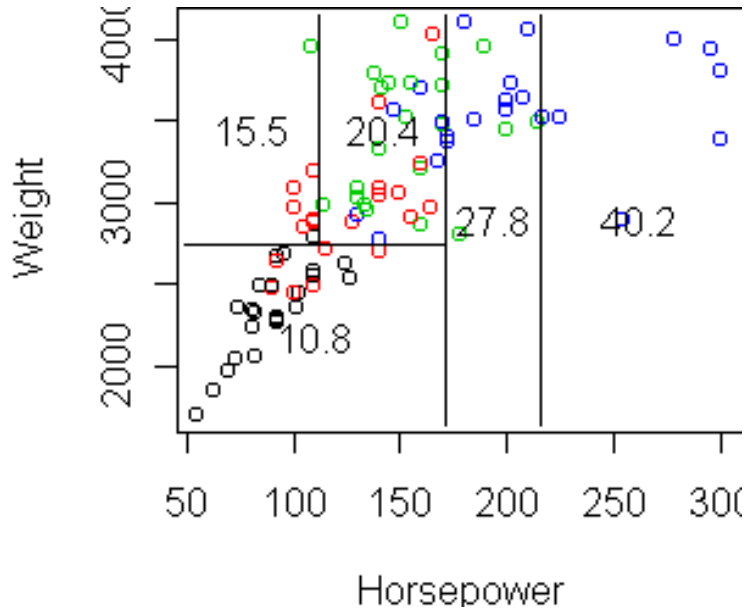
- For regression problems:
  - Averaging the target variable of the *k* nearest neighbors
  - Weighted averaging
    - $w_i = \frac{1}{d_i^2}$ where d_i is the distance of the *ith* neighbor

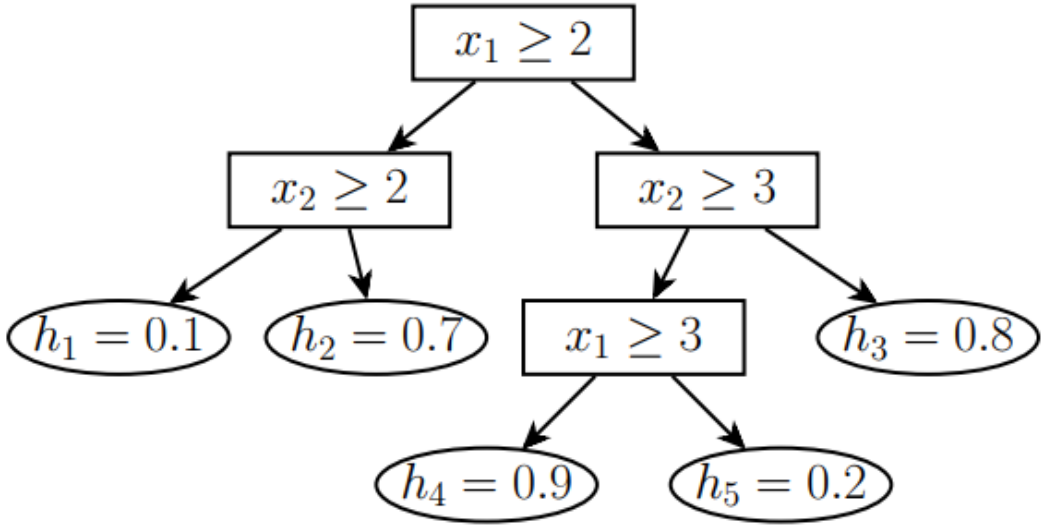




Decision tree for regression

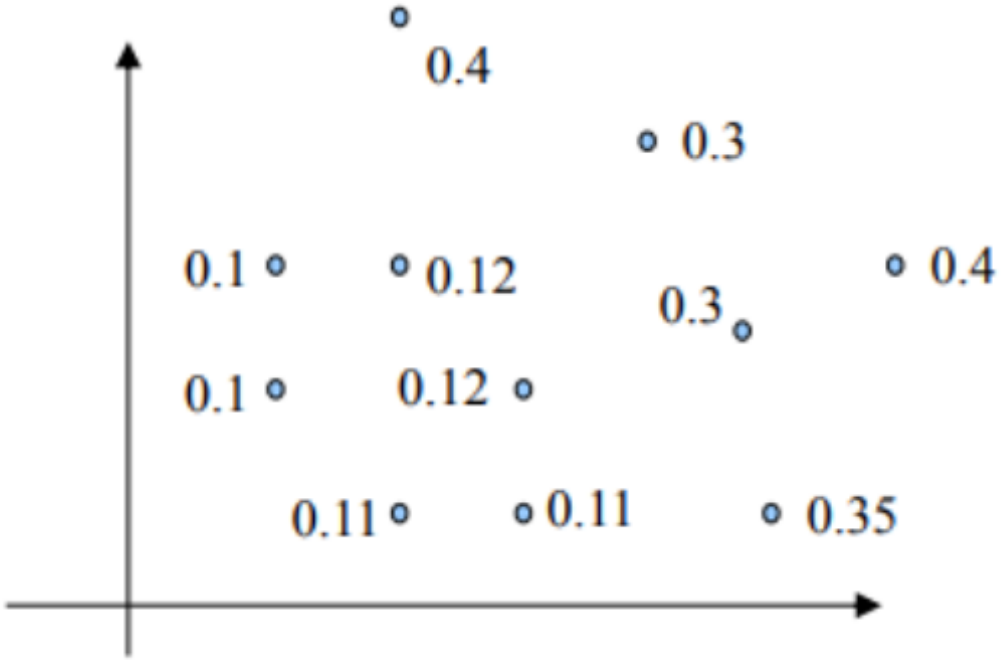
- Prediction: the average of the target variables in each leaf
- Splitting criteria: reducing the variance of the child nodes with respect to the target variable





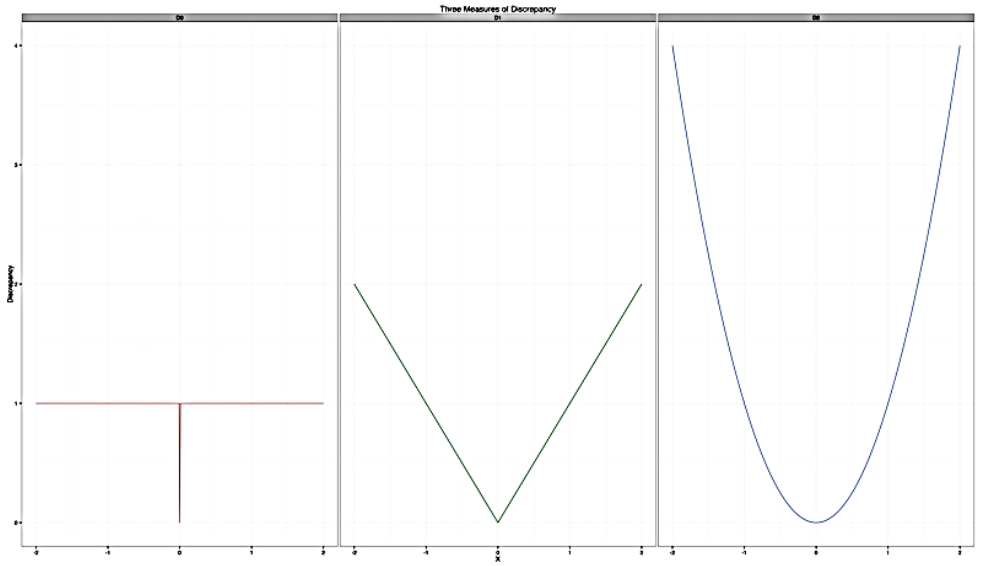
Problem

We would like to solve a regression problem using a decision tree. How would it split the data given in the coordinate system below? The maximum number of leaves is set to 3. Sketch the splits on the figure. (No precise calculation is required.)



Outlook – What is the best guess?

- We have a sample (some observations, real numbers in an ascending order):
  - E.g.: 1.3, 1.8, 2, 2, 2, 2.5, 4.4, 4.6, 5, 5.6, 5.6, 5.9, 6.3, 6.7, 7.1, 8.5, 8.9, 9.3, 9.3, 9.6, 9.8
- What would be the best guess?
- What minimizes the error?
  - It depends on which error?
- Mean, median, mode?
- Mean minimizes the squared error (loss):
  $\bar{x} = \operatorname{argmin}_s \sum_i (x_i - s)^2 \approx 5.44$
- Median minimizes the absolute error (loss):
  $\operatorname{median}(x) = \operatorname{argmin}_s \sum_i |x_i - s| = 5.6$
- Mode minimizes the „equivalence error”:
  $\operatorname{mode}(x) = \operatorname{argmin}_s \sum_i \chi(x_i \neq s) = 2$



Squared error

- For regression problems the aim is usually to minimize the squared error
  - Least square regression, ordinary least square - OLS
- Mean squared error –MSE
  Training data: $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)$

  Predicted function: $\hat{f}$

$$\mathrm{MSE}(\hat{f}) = \mathrm{E}\left(\left(\mathrm{y} - \hat{f}(x)\right)^2\right)$$

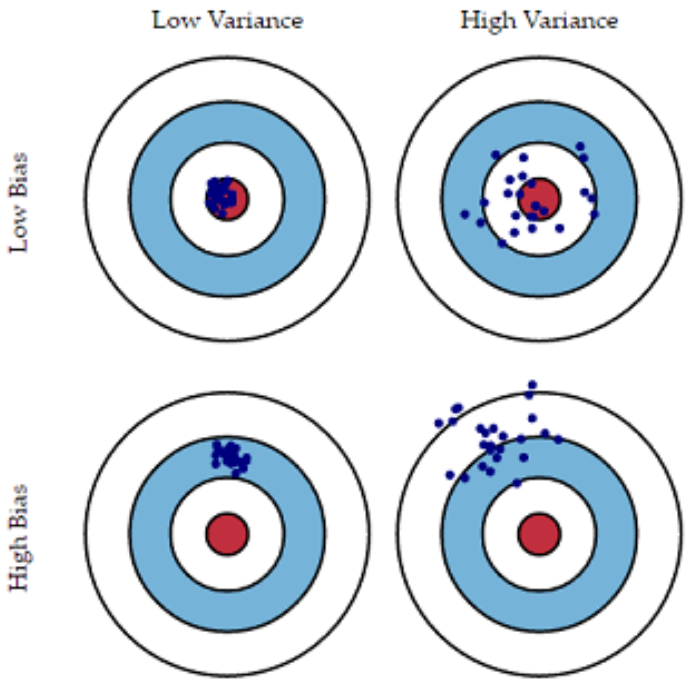
- MSE of the training set

$$\mathrm{MSE}_{train}(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

  - MSE of the test set can be calculated similarly

Bias and variance

- **Bias:** The difference between the average prediction of our model and the correct value which we are trying to predict.
- **Variance**: The variability of model prediction for a given data point
  - Usually a model has only one prediction for a given data point but think of it as we can repeat our process of model building to get separate hits on the target



Bias and variance II.

- Bias:

$$\mathrm{Bias}\left(\hat{f}(x)\right) = E(\hat{f}(x) - f(x))$$

  - Sometimes the squared bias is used

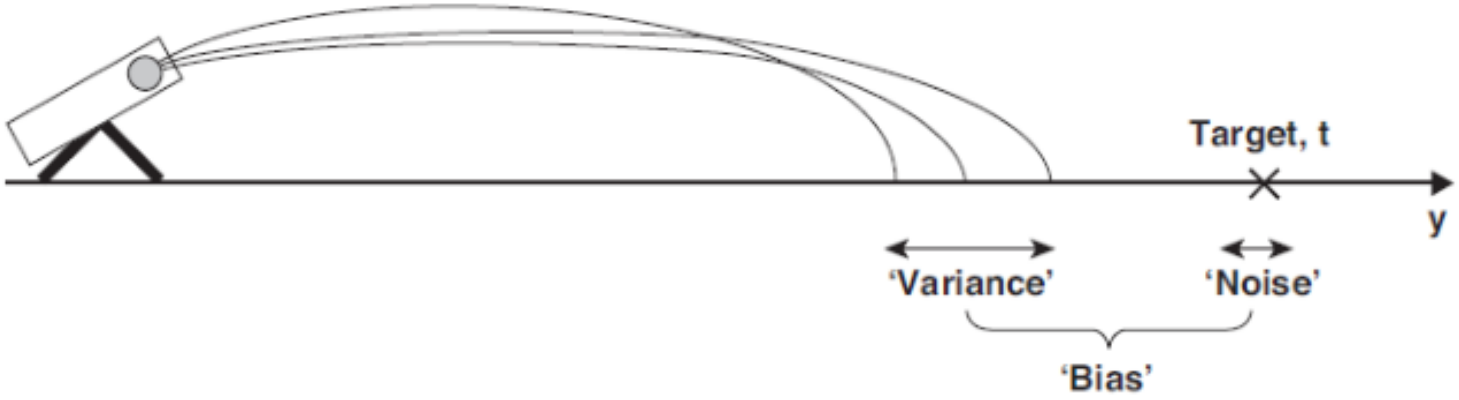
- Variance:

$$\mathrm{Var}\left(\hat{f}(x)\right) = \mathrm{E}\left(\hat{f}(\mathrm{x}) - \mathrm{E}\left(\hat{f}(\mathrm{x})\right)\right)^2 = \mathrm{E}\left(\hat{f}(\mathrm{x})^2\right) - \mathrm{E}\left(\hat{f}(x)\right)^2$$

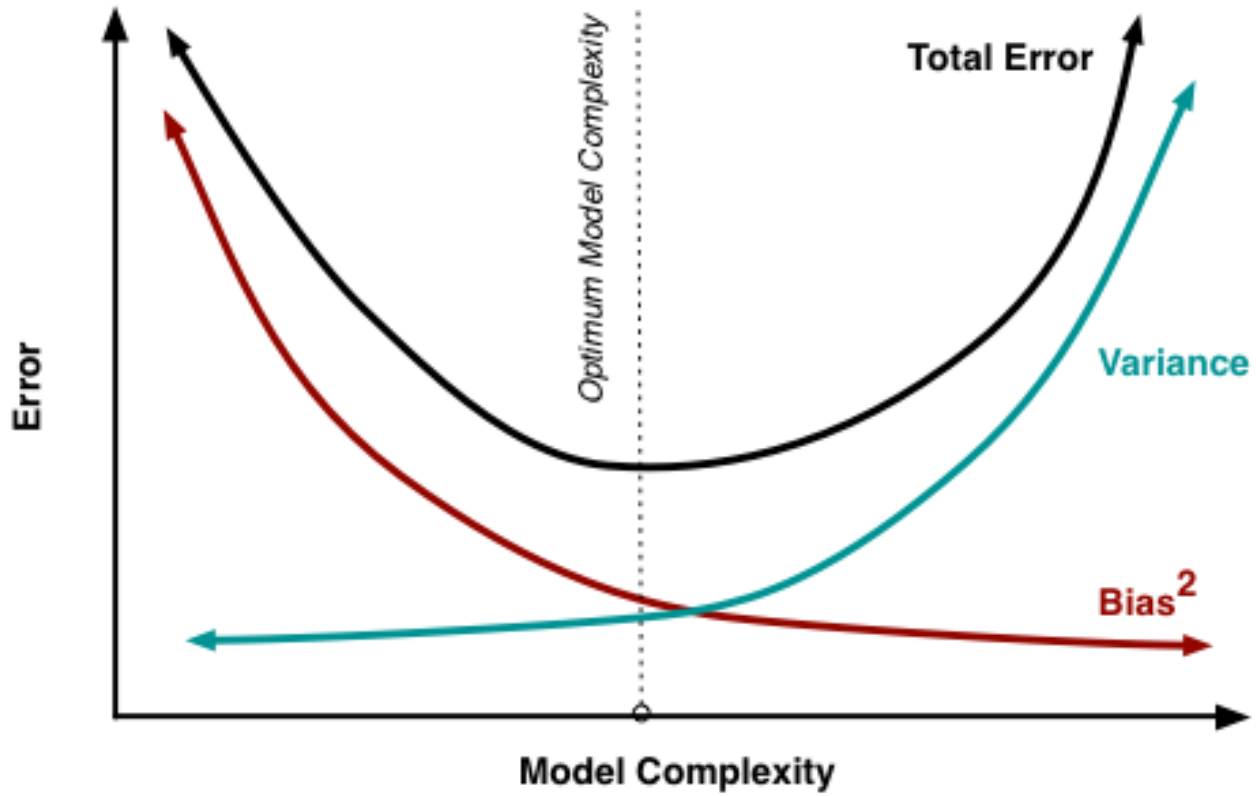
Decomposition of mean squared error

- Claim: For a new record the mean squared error of the predicted value (predicted by $\hat{f}$ function) can be decomposed in the following way:

$$\mathrm{MSE}\left(\hat{f}(x)\right) = \mathrm{Var}\left(\hat{f}(x)\right) + \mathrm{Bias}^2\left(\hat{f}(x)\right) + \mathrm{Var}(\epsilon)$$



Bias-variance tradeoff



Optimal solution: the conditional expected value

- If we aim to minimize the expected (mean) squared error $E\left(\left(Y - f\left(X_1, X_2, \ldots, X_p\right)\right)^2\right)$, then the optimal f function is the conditional expected value:

$$\mathrm{f_{opt}}(\mathrm{x}_1, \ldots, x_p) = E\left(Y \middle| X_1 = x_1, X_2 = x_2, \ldots, X_p = x_p\right)$$

  - The problem is theoretically solved but to calculate the conditional expected value we should know the joint density function $g\left(y, x_1, x_2, \ldots, x_p\right)$
    - If the joint distribution is the *(p+1)*-dimensional normal distribution then $\mathrm{f_{opt}}$ is a linear function
      - Due to the multi-dimensional central limit theorem normality is a valid assumption in many cases
- It is reasonable to look for *f* in the family of linear functions

Linear regression

- We assume that *f* is linear, i.e. $f(\mathrm{x}_1, \ldots, x_p) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_p x_p$
- We are looking for the parameters $w_0, w_1, \ldots, w_p \in R$ such that $E\left(Y - (w_0 + w_1 X_1 + w_2 X_2 + \cdots + w_p X_p)\right)^2$ is minimal
- On a training set with n data points we have to solve the following minimization problem

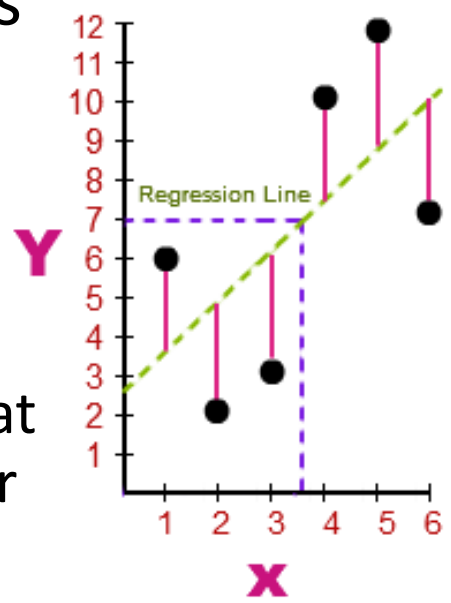
$$\mathrm{argmin}_{w_0, w_1, \ldots, w_p} \frac{1}{n} \sum_{i=1}^{n} (y_i - (w_0 + \boldsymbol{w}^T \boldsymbol{x_i}))^2$$

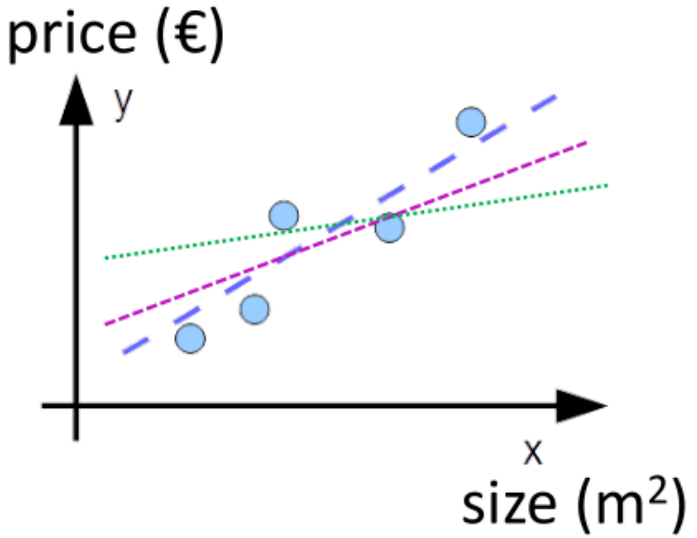
where *n*: number of training data points, $\mathbf{w} = (w_1, w_2, \ldots, w_p)$**:** parameter vector, $\boldsymbol{x}_i$: the feature vector of the *i*th record

- The minimization problem can be solved analytically
  - Huge matrices should be inverted, multiplied with each other (computationally very expensive in practice)
  - Solution: using other optimization methods, e.g. gradient descent

Regression line

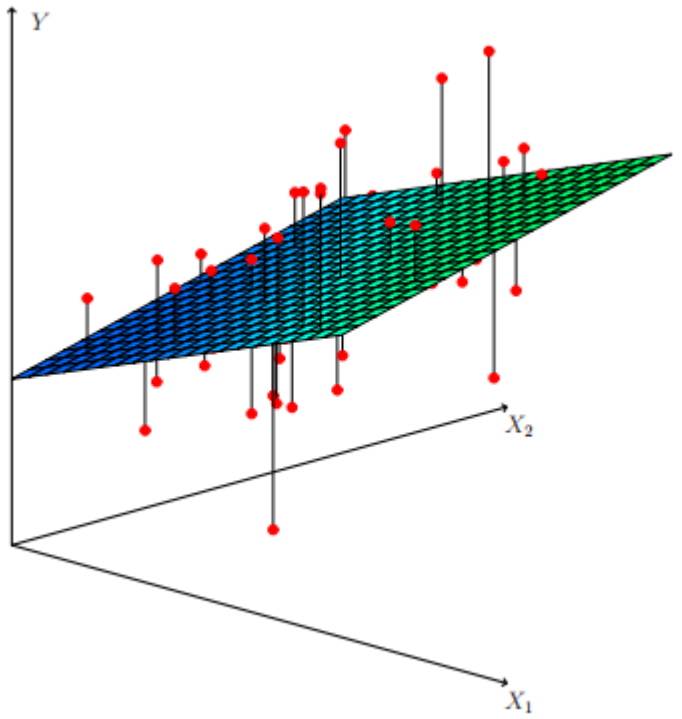
- In two dimension linear regression corresponds to a line
- If *n>p+1,* then the system is overdetermined, there is no such a regression that satisfies all the training data
  - We want to find the one that minimizes the squared error





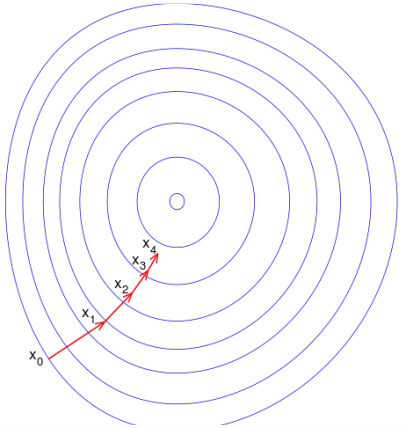
Linear regression for more variables

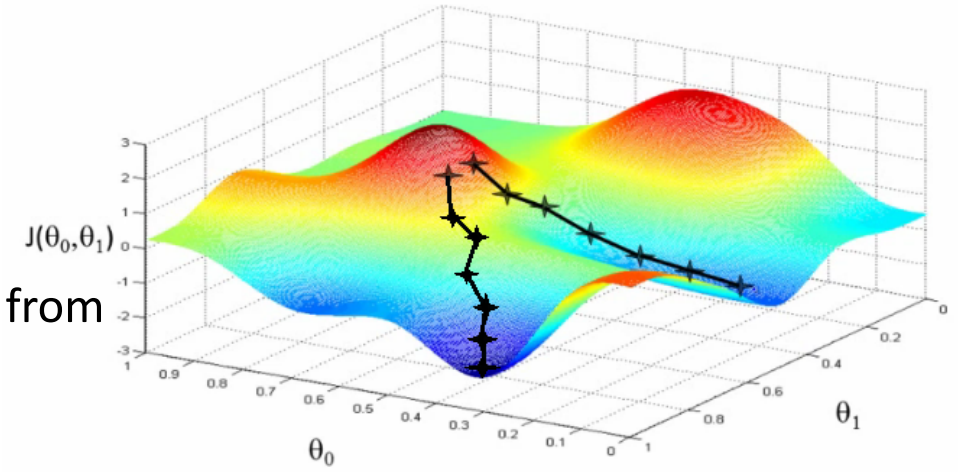
- Multi-dimensional linear regression defines a hyperplane
  - Red dots are the real observations
  - The plane is the fitted linear regression
    - It minimizes the squared error



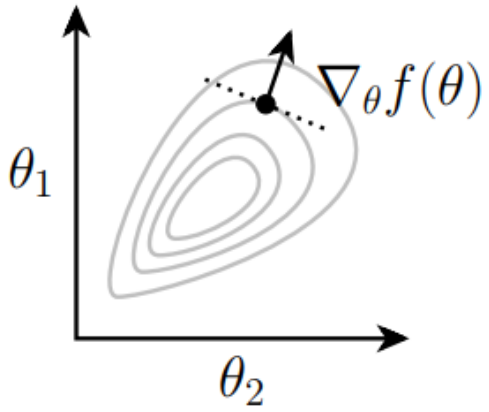
Gradient method (gradient descent/ascent)

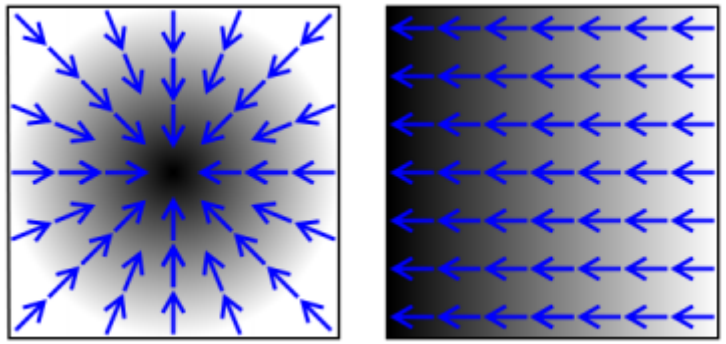
- Goal: to find the local minima (maxima) of an *f* function
- Idea: one takes steps proportional to the negative (positive) of the gradient of the function at the current point
  - Gradient ($\text{grad}\ f$) is the direction of steepest ascent
  - Negative gradient ($-\text{grad}\ f$) is the direction of steepest descent
  - The gradient stores all the partial derivatives of a multivariable function
- It will converge to a local minima (maxima), the convergence is quite fast under some regularity assumption
  - It is not necessarily a global minima (maxima)
  - To avoid getting stuck in a local minima we can initialize from more starting points (random restart hill climbing)

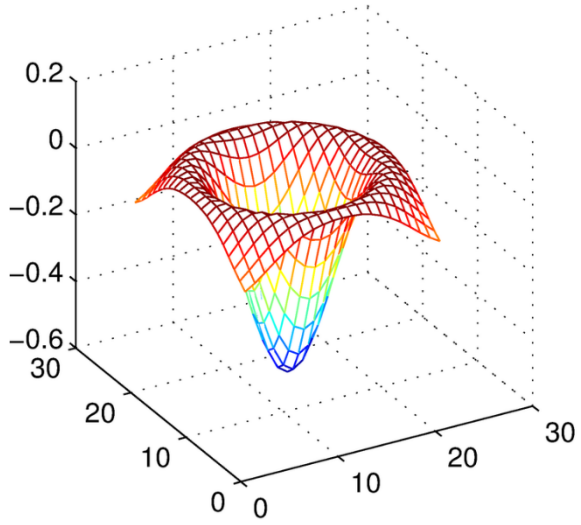


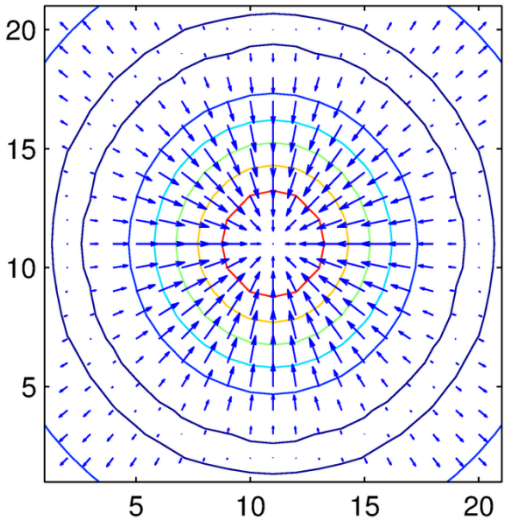


Gradient method









Example

Let $y=w_0+w_1x_1+w_2x_2$ be an equation for regression. We have an observation with $x_1=2$ and $x_2=-1$, its target variable is 3. Write the squared error for this data point as a function of w_0, w_1, w_2! Calculate the gradient of the squared error in this point!

$$err = w_0 + w_1x_1 + w_2x_2 - 3 = w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3$$

$$\frac{d}{dw_1}(err)^2 = \frac{d}{dw_1}(w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)^2 = 2 \cdot 2 \cdot (w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)$$

$$\frac{d}{dw_2}(err)^2 = \frac{d}{dw_2}(w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)^2 = (-1) \cdot 2 \cdot (w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)$$

$$\frac{d}{dw_0}(err)^2 = \frac{d}{dw_0}(w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)^2 = 2 \cdot (w_0 + w_1 \cdot 2 + w_2 \cdot (-1) - 3)$$

Stochastic gradient descent (SGD) – for linear regression

- The function to minimize:
  $Err^2 = \left((w_0 + \boldsymbol{w}^T \boldsymbol{x}\right) - y)^2$
- The gradient is not calculated using the entire dataset but only for one randomly chosen record in each step
- In each step for one randomly chosen training data point
  - Calculate the gradient vector (partial derivatives) of the mean-squared error using that single point
  - We step to the direction of the negative gradient, i.e. we improve the parameters with the following term:
    (-1) · gradient · learning_rate

$$\frac{\partial \mathrm{Err}^2}{\partial w_0} = 2 \cdot \mathrm{Err}$$

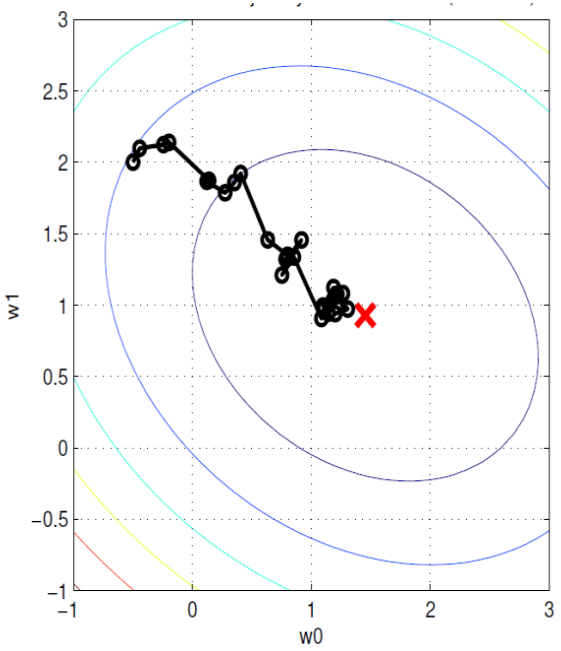
$$\frac{\partial \mathrm{Err}^2}{\partial w_i} = 2 \cdot \mathrm{Err} \cdot x_i$$

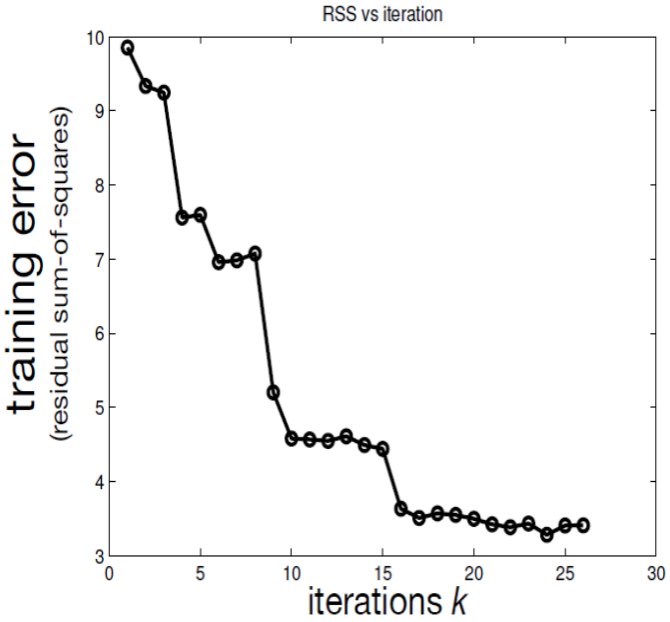
$$w_0 \leftarrow w_0 - \mathrm{lratre} \cdot 2\mathrm{Err}$$

$$w_i \leftarrow w_i - \mathrm{lratre} \cdot 2\mathrm{Err} \cdot x_i$$

Stochastic gradient descent on training data

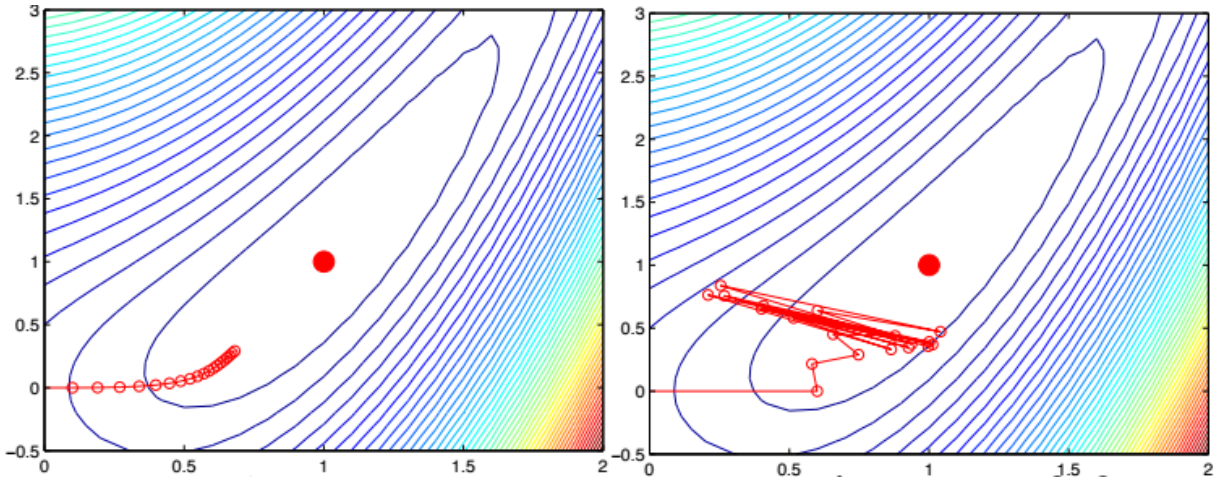
- Initialize the parameters (coefficients, weights) of the model randomly
- Iterate the following steps on the training data on randomly chosen data points
  - Calculate the squared error with respect to the chosen data point
  - Calculate the gradient
  - Improve the weights according to the gradient (and the learning rate)





Role of the learning rate

- Learning rate
  - Usual notation: λ
- It regulates the amount by which the weights will change at every step, i..e. how fast the optimization algorithm learns
  - If the rate is too large: the step size is too large, it can (plausibly) „jump over" the minima we are trying to reach
  - If the rate is too small: too many steps are needed to reach the minima (too slow)



Problem

Let $(0, 0, -2)$; $(0, 1, 1)$; $(1, 0, 2)$ be three records on the x_1–x_2 plane, where the third coordinate is the y target variable. Determine the coefficients of the following linear regression that minimizes the squared error: $y = w_1 x_1 + w_2 x_2 + w_0$.

a) Determine the optimal coefficients analytically!

b) Approximate the optimal coefficients using the gradient descent method (few steps enough).

c) Approximate the optimal coefficients using the stochastic gradient descent (few steps enough).

For gradient methods use the following initialization of the weights: $w_1 = w_2 = w_0 = 1$. Let the learning rate be 0.25.

Regularization

- Goal: to avoid overfitting, to have better generalization ability
- How?
  A regularization term is added to the loss function
  - The regularization term penalizes for too complex models
  - We have already seen this at decision trees
- For linear regression we add the following regularization term to the squared error: $Err^2 + Reg = (y - (w_0 + \boldsymbol{w}^T \boldsymbol{x}))^2 + \mu(w_0^2 + |\boldsymbol{w}|^2)$
  - It penalizes for two large weights (in absolute value) to avoid too large weights and overfitting (μ is the parameter of the regularization term)

Squared error

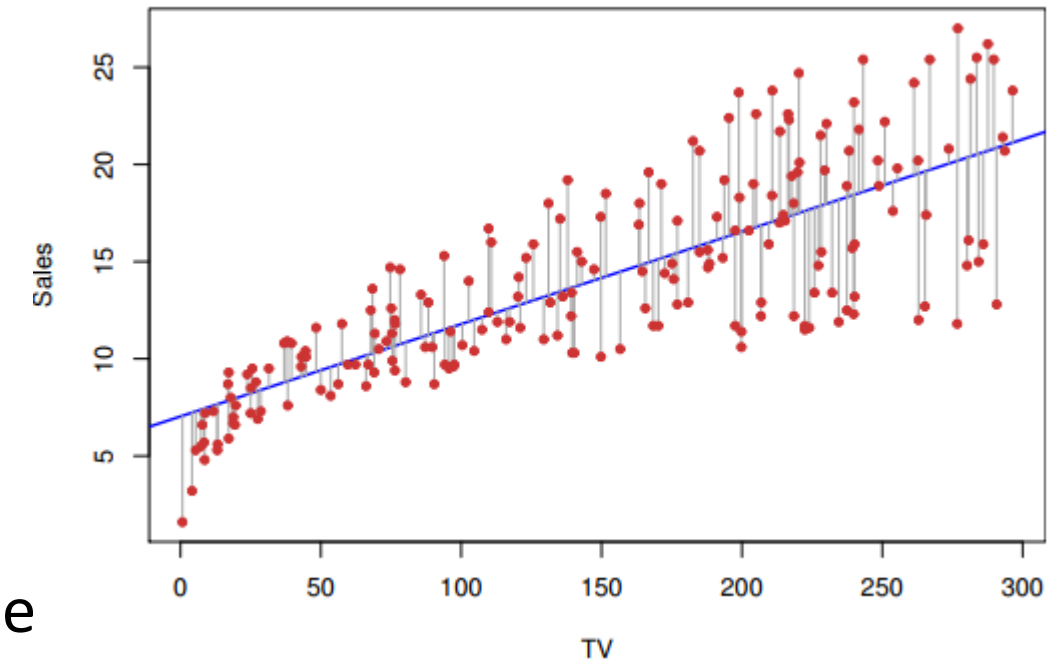
- Residual sum of squares (RSS) - sum of squared errors of prediction (SSE)

$$\text{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- Root-mean-square error (RMSE)

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}{n}}$$

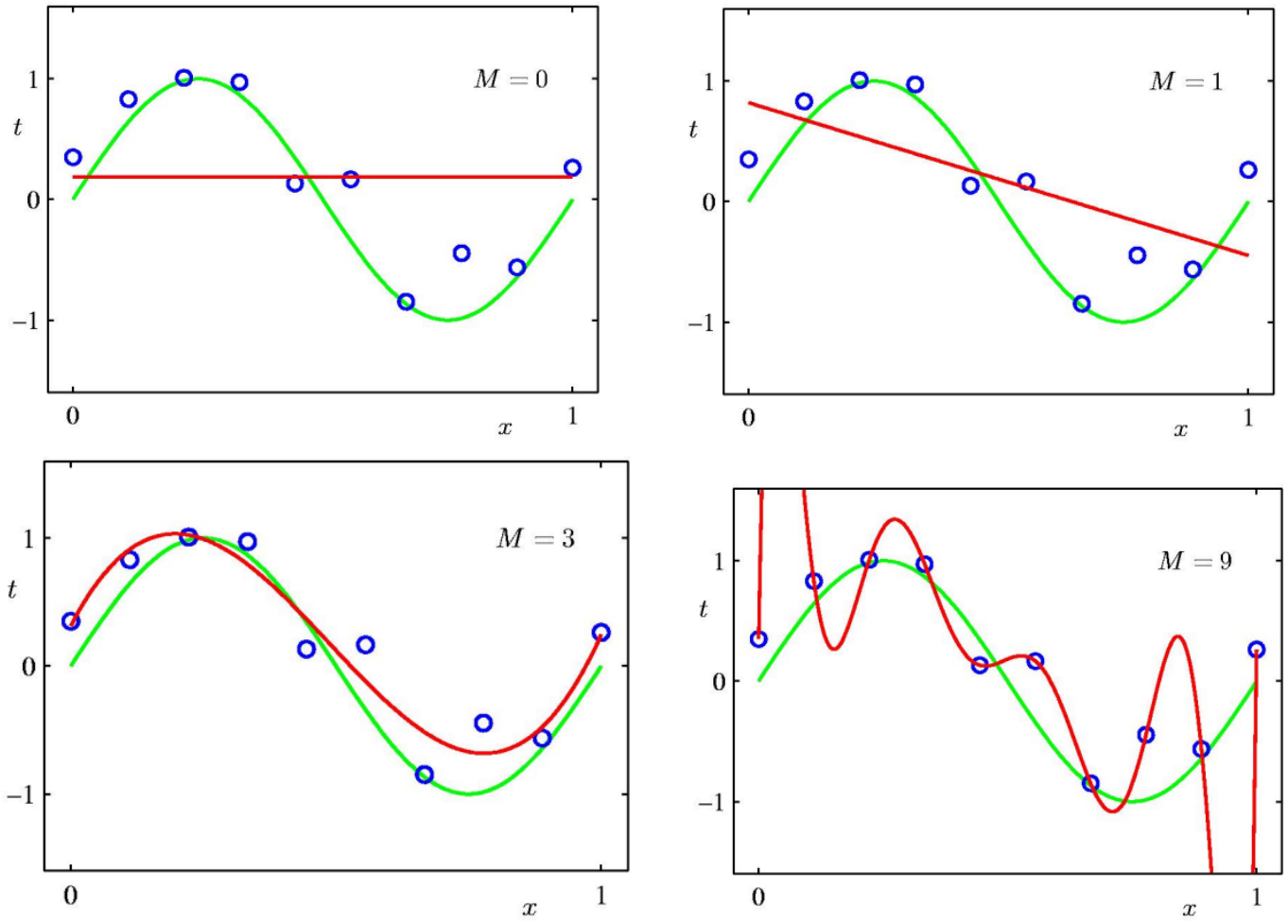
- Minimizing any of them leads to the same model



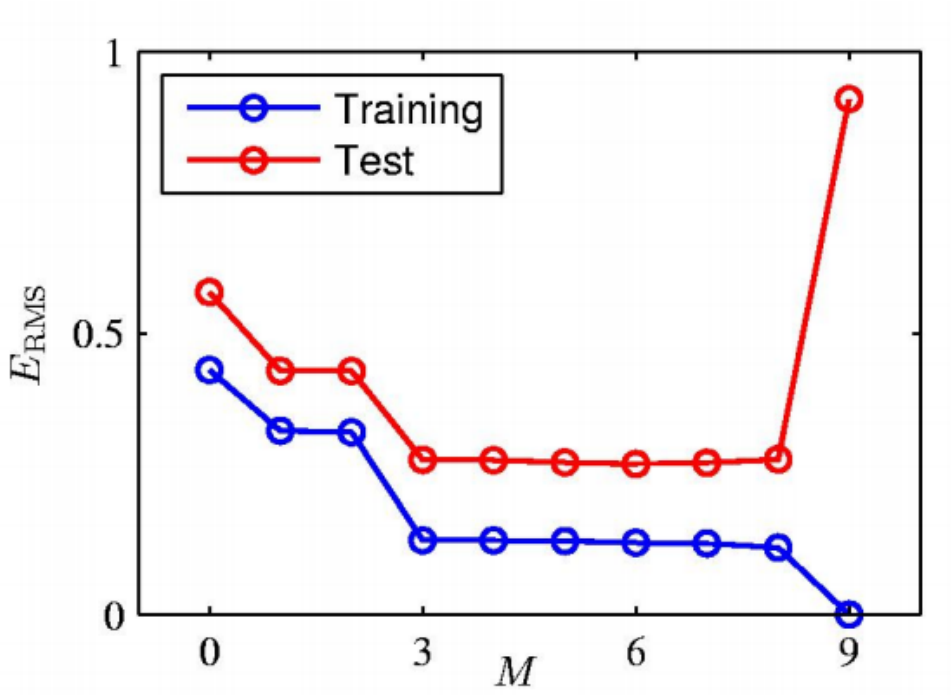
Polynomial regression

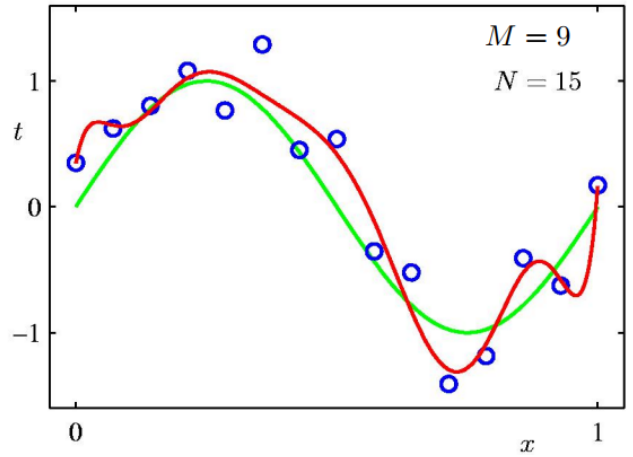
- Higher degree polynomials of the attributes are also present in the regression equation
  - Other functions of the attributes may be also considered: log, exp etc.

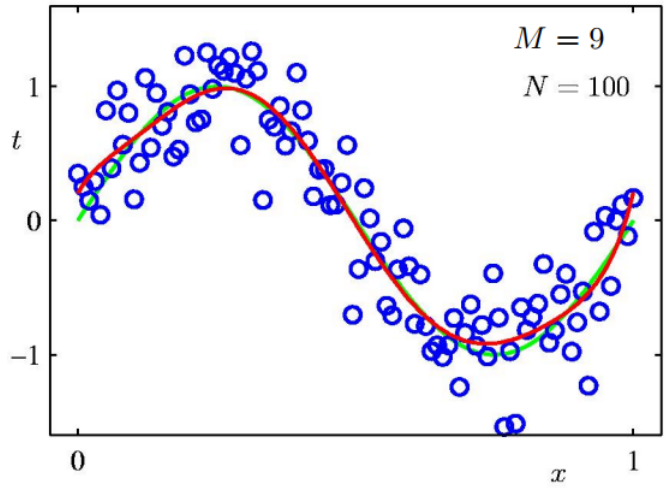
M is the degree of the fitted polynomial



Overfitting





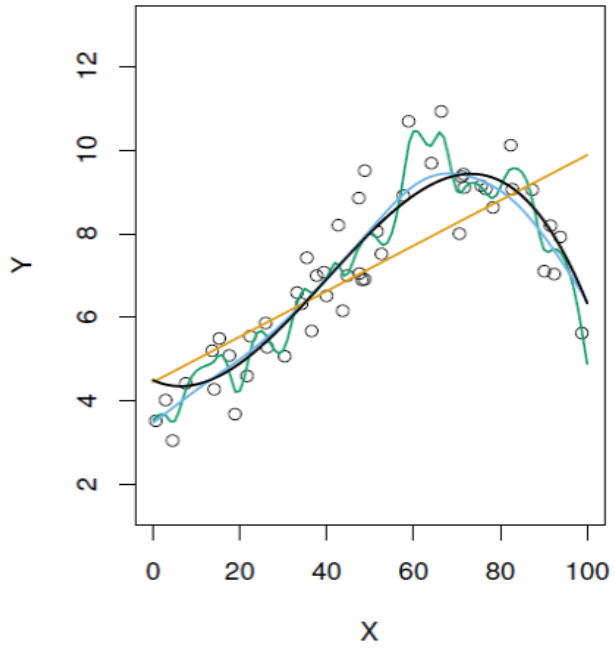


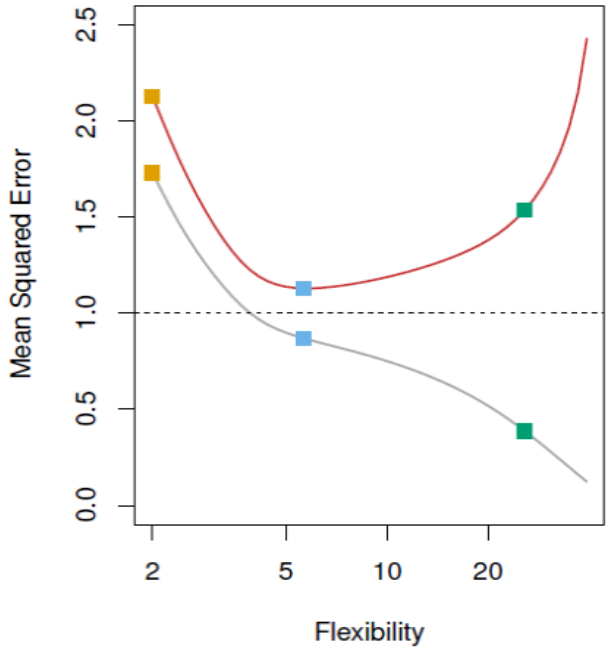
M is the degree of the fitted polynomial, N is the number of data points

Overfitting II.

Three estimates $\hat{f}$ are shown:

1. Linear regression.
2. Splines (very smooth).
3. Splines (quite rough).





Red line: Test MSE.

Gray line: Training MSE.

The circles are simulated data from the black curve. In this artificial example, we *know* what f is.

Interpretation of coefficients

- We have the following model: $y_i = w_0 + w_1 x_{1i} + w_2 x_{2i} + \mathrm{u_i}$
  - u_i is the error term
  - How can w_1 be interpreted?
  - $\frac{\partial y_i}{\partial x_{1i}} = w_1$, i.e. increasing x_{1i} by a unit increases the predicted value of y_i by w_1 units in expectation(if x_{2i} remains constant, i.e. ceteris paribus – other things held constant)
- If the model is: $y_i = w_0 + w_1 x_{1i} + w_2 x_{1i}^2 + \mathrm{u_i}$
  - $\frac{\partial y_i}{\partial x_{1i}} = w_1 + 2w_2 x_{1i}$ i.e., the expected effect of increasing x_{1i} on y_i is not constant in x_{1i}

Linear regression with interaction terms

- We can choose a model specification where there are interactions between the explanatory variables
  - It is mainly used if one of the variable is binary (dummy variable)
  - Let the model be: $y_i = w_0 + w_1 x_{1i} + w_2 d_i x_{1i} + \mathrm{u_i}$,
  - Where d_i is a binary (dummy) variable
  - $\frac{\partial y_i}{\partial x_{1i}} = w_1 + w_2 d_i$, i.e. the effect also depends on the value of d_i

Stepwise regression

- Automatic procedures to decide which features (explanatory variables) include in the regression
- **Forward selection**: starting with no variables in the model, adding the variable whose inclusion gives the most improvement of the fit, repeating the process until none improves the model to a statistically significant extent
- **Backward elimination**: starting with all candidate variables, deleting the variable whose loss gives the most insignificant decline of the model fit, repeating the process until no further variables can be deleted without a significant loss of fit
- **Bidirectional elimination**: a combination of the above, testing at each step for variables to be included or excluded

Statistical tests

- Using a t-test we can decide if a coefficient (weight) is significantly different from 0, i.e. the corresponding variable has significant explanatory power in the model
- Using an F-test it can be determined whether the full regression model has significant explanatory variable regarding the target variable

Acknowledgement

- András Benczúr, Róbert Pálovics, SZTAKI-AIT, DM1-2
- Krisztián Buza, MTA-BME, VISZJV68
- Bálint Daróczy, SZTAKI-BME, VISZAMA01
- Judit Csima, BME, VISZM185
- Gábor Horváth, Péter Antal, BME, VIMMD294, VIMIA313
- Lukács András, ELTE, MM1C1AB6E
- Tim Kraska, Brown University, CS195
- Dan Potter, Carsten Binnig, Eli Upfal, Brown University, CS1951A
- Erik Sudderth, Brown University, CS142
- Joe Blitzstein, Hanspeter Pfister, Verena Kaynig-Fittkau, Harvard University, CS109
- Rajan Patel, Stanford University, STAT202
- Andrew Ng, John Duchi, Stanford University, CS229