Regression

Data Base and Data Mining Group of Politecnico di Torino

Flavio Giobergia

Politecnico di Torino

Introduction to regression

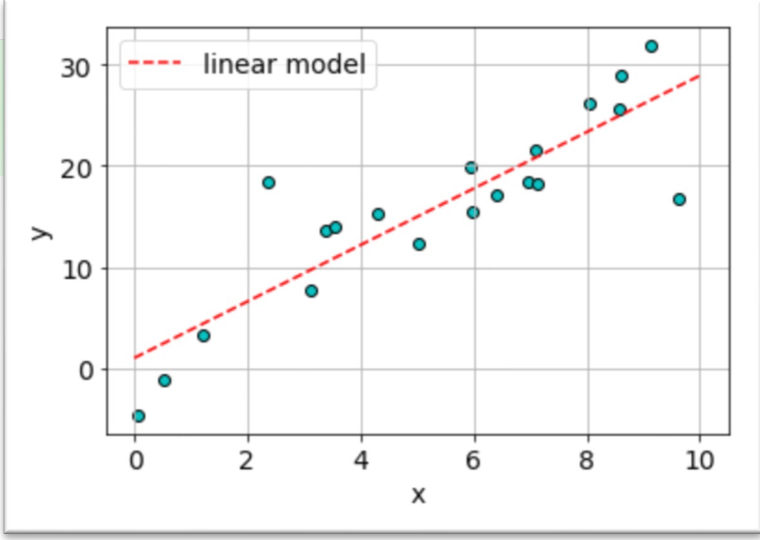
- **Objective**: Predict a *continuous outcome variable* based on *one or more predictor variables*
  - i.e., learn a function $f : \mathcal{X} \rightarrow \mathbb{R}$
  - We refer to the outcome as the *dependent variable*, and to the predictors as the *independent variables*

- Useful for:
  - Making predictions
  - Understanding relationships between variables
  - Identifying significant predictors

Linear regression

Data Base and Data Mining Group of Politecnico di Torino

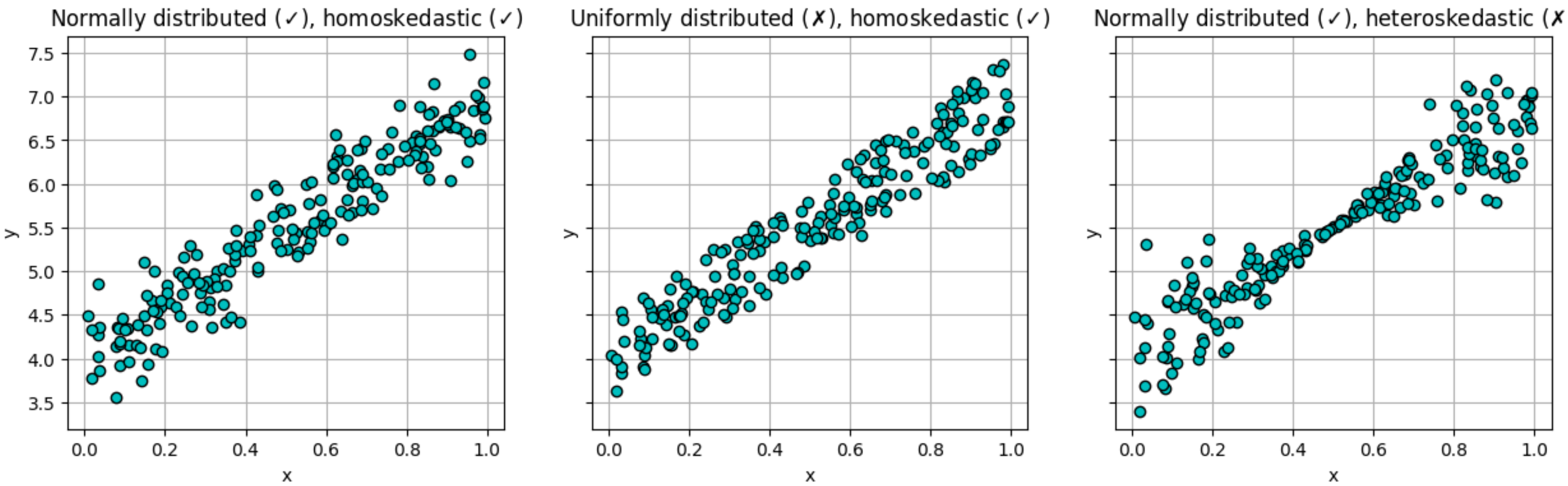
Linear regression



- Used to model linear relationships between predictors and outcome
- Assumption:
  - There is a linear relation between the independent (x) and dependent (y) variables
  - $y = \theta_0 + \theta_1 x + \varepsilon$ (observation)
  - ε represents a stochasticity that we cannot model
- Simple linear regression:
  - Goal: estimate θ_0, θ_1 so that we can build our own model!
  - $\hat{y} = \hat{\theta}_0 + \hat{\theta}_1 x$ (prediction)
- ε: residual (difference between predictions and observations)

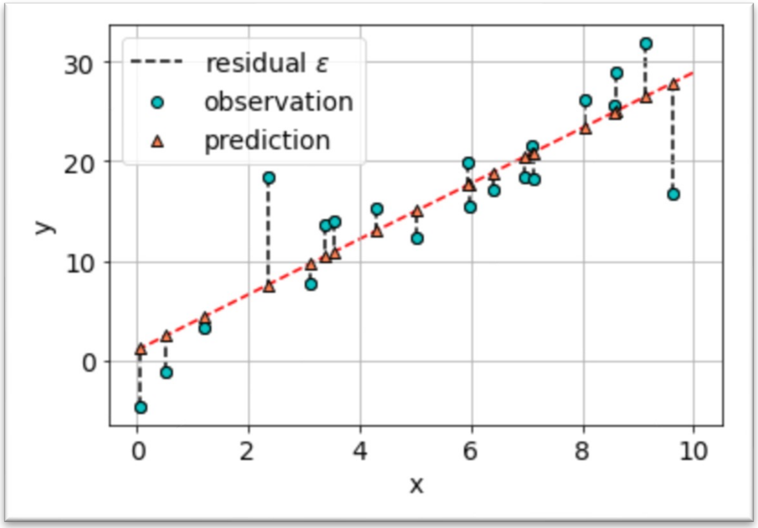
Residuals

- Residuals are expected to be:
  - Normally distributed
  - Homoskedastic



Residuals, error

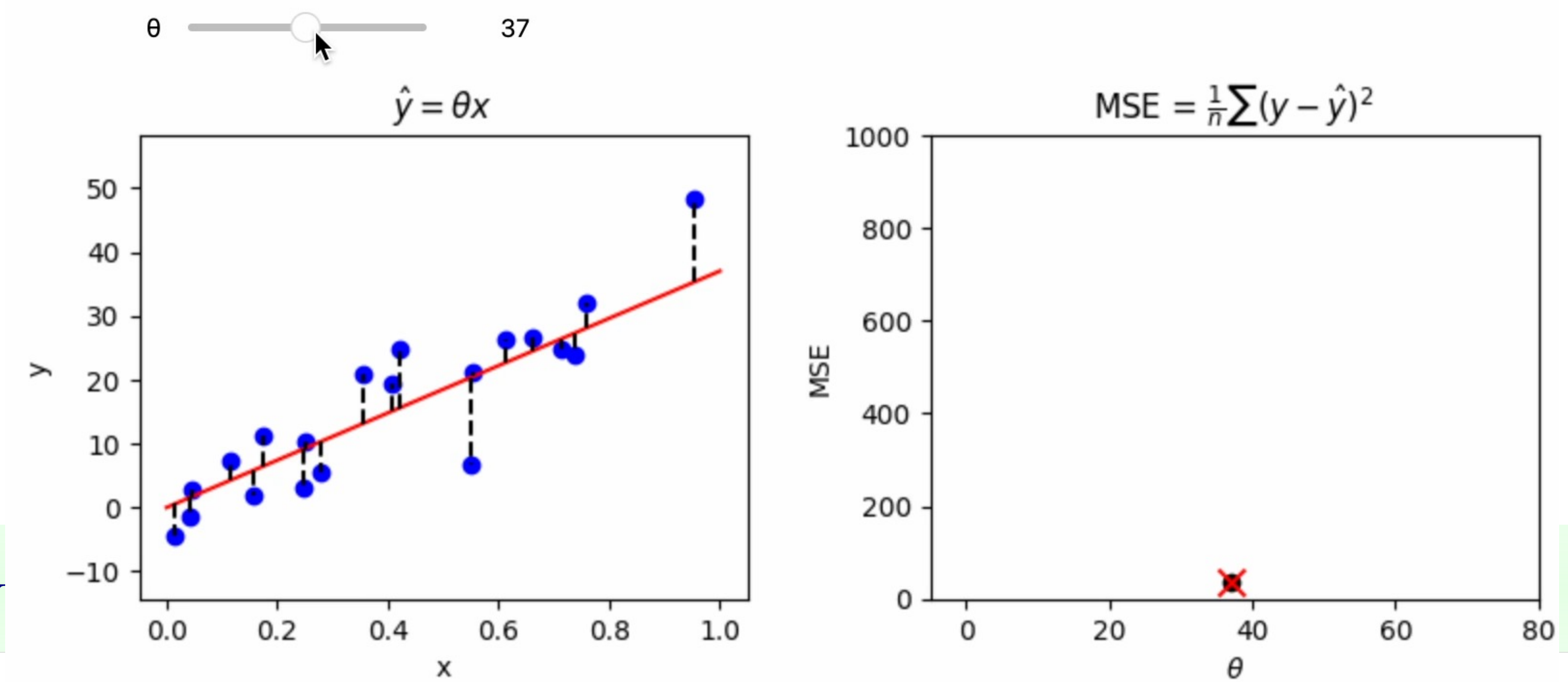
- We can compute the squared error for x_i
  - $(y_i - \hat{y}_i)^2 = \varepsilon_i{}^2$
- Properties of squared errors:
  - Quantify quality of prediction
    - The smaller the better!
  - Always positive
  - “Stretches” error:
    - $(\text{Large error})^2$ = even larger error
    - $(\text{Small error})^2$ = smaller error



- Error over the entire dataset: mean squared error (MSE)
  - $MSE = \frac{1}{n}\sum_i (y_i - \hat{y}_i)^2$

Residuals, error

- The MSE, $\frac{1}{n}\sum_i(y_i - \theta_0 - \theta_1 x_i)^2$, is a quadratic function of the parameters θ
- So, it has a single minimum, which are the "best" values for θ



Error minimization

- $MSE(\theta_0, \theta_1) = \frac{1}{n}\sum_i (y_i - \theta_0 - \theta_1 x_i)^2$
  - "Cost function" to be minimized
- We want to find θ_0, θ_1 that minimize the MSE
- MSE is a quadratic function of θ_0, θ_1
  - Minimum for $\frac{\partial MSE}{\partial \theta_0} = 0$, $\frac{\partial \mathrm{MSE}}{\partial \theta_1} = 0$
- Linear regression chooses the parameters θ_0, θ_1 that minimize the SSE
  - $\theta_0 = \bar{y} - \theta_1 \bar{x}$
  - $\theta_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}$

Multivariate case

- Similarly, we can define a problem with *n* independent variables
- $x = (x_1, x_2, \ldots x_n)$
- Multiple linear regression:
  - $\hat{y} = \theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n$
  - $\hat{y} = \boldsymbol{\theta}^T \boldsymbol{x}$
    - as a scalar product of $\boldsymbol{x} = (1, x_1, x_2 \ldots x_n)$ and $\boldsymbol{\theta} = (\theta_0, \theta_1 \ldots \theta_n)$
- Solution:
  - $\boldsymbol{\theta} = (X^T X)^{-1} X^T Y$
- The coefficients help understand the relationship between the independent and dependent variables
  - E.g. θ_1 indicates the change in the predicted y for a one-unit increase in x_1, all else being equal

Non-linear relationships

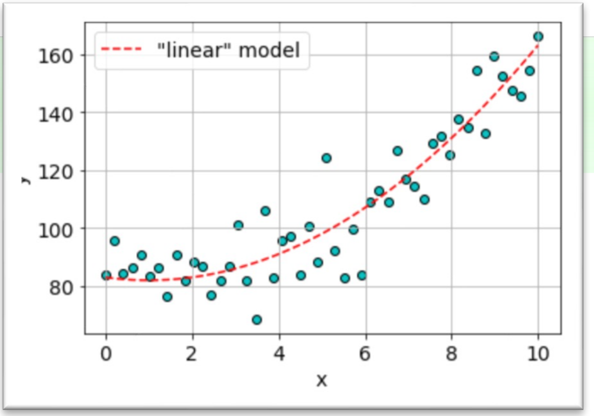
- We may want to model non-linear relationships
- We can *add new features*, non-linear transformations of the original one(s)
  - E.g., if we expect an inverse quadratic relationships between x and y, we introduce a new feature, $\frac{1}{x^2}$
- Then, we use a "classic" linear regression
  - The model learns a separate coefficient for each feature
  - $y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \leftrightarrow \theta_0 + \theta_1 x + \theta_2 \frac{1}{x^2}$

x	y
0.4	105
0.3	84
0.2	210

⇩

$x_1 = x$	$x_2 = \frac{1}{x^2}$
0.4	2.5
0.5	2
0.2	5

Polynomial regression



- We can introduce more flexibility in representing relationships with a polynomial regression
  - i.e., add new polynomial features up to degree *n*
  - Increases model capacity
  - Univariate: $\hat{y} = \theta_0 + \theta_1 x + \theta_2 x^2 \ldots + \theta_n x^n$
- For multivariate problems, we can add either powers, or interactions (or both!)
  - Powers ($x_1^2, x_2^2, x_1^3 \ldots$)
  - Interactions ($x_1 x_2, x_1 x_2^2, \ldots$), capturing relations between variables at different polynomial degrees
  - E.g., $x_1, x_2, x_3, n = 2 \rightarrow x_1^2, x_2^2, x_3^2, x_1 x_2, x_1 x_3, x_2 x_3$
  - ! The total number of features increases <u>combinatorially</u>!

Evaluation

Data Base and Data Mining Group of Politecnico di Torino

MSE, RMSE, MAE

- Mean squared error (MSE)
  - $MSE = \frac{1}{n}\sum_i (y_i - \hat{y}_i)^2$
  - Sometimes not normalized by # points
    - SSE (Sum of SE)
- RMSE (root MSE)
  - $RMSE = \sqrt{MSE}$
  - Same unit of measurement as the dependent var.
- Mean absolute error (MAE)
  - $MAE = \frac{1}{n}\sum_i | y_i - \hat{y}_i |$
  - Penalizes more «small» errors (w.r.t. MSE)

R-squared (R^2)

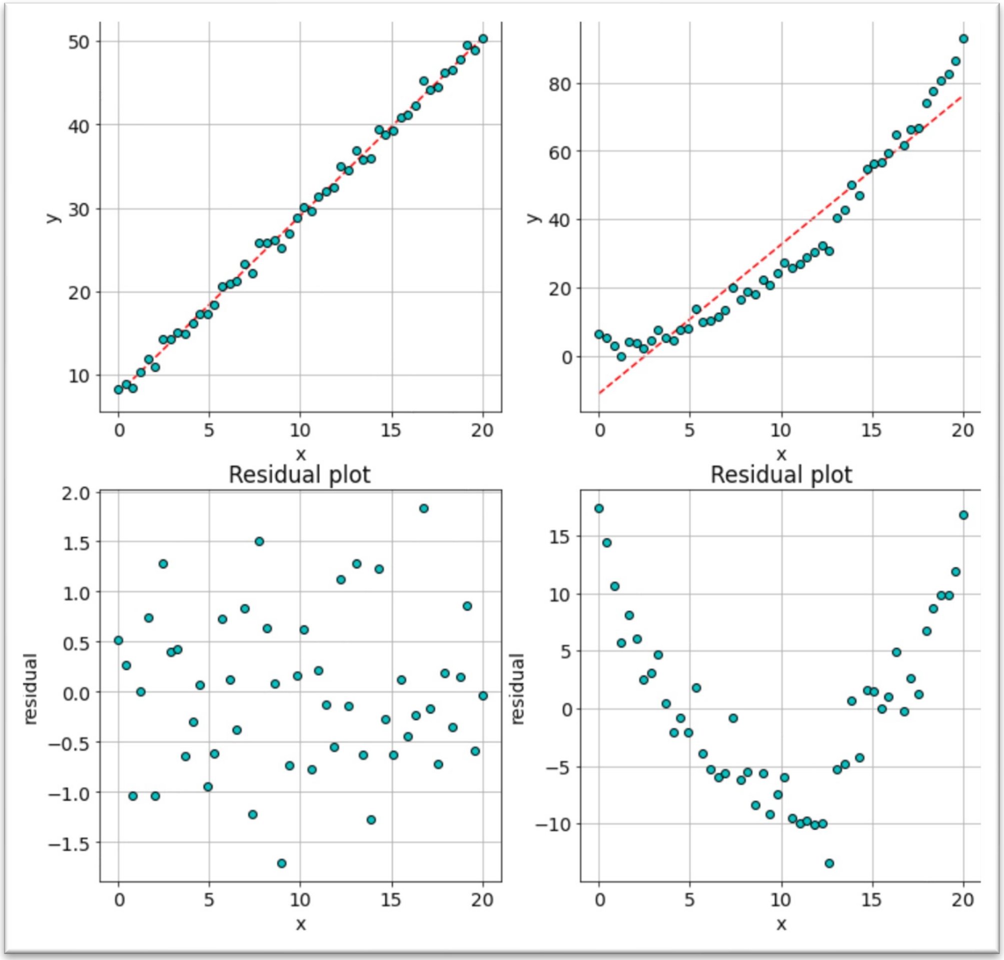
- R^2: proportion of the variance in the dependent variable that is explained by the independent variables

$$R^2 = 1 - \frac{MSE}{\sigma^2}$$

- Edge cases:
  - Model predicts everything perfectly
    - $MSE = 0$, $R^2 = 1$ (upper bound)
  - Model is no better than predicting mean value of y
    - $MSE = \sigma^2$, $R^2 = 0$
  - Model is worse than predicting mean value
    - $MSE < \sigma^2$, $R^2 < 0$

Residual plots

- Residual plots: visual assessment of the goodness of fit of a regression model
  - Expecting residuals to be random scattered around zero, with constant variance, and no patterns



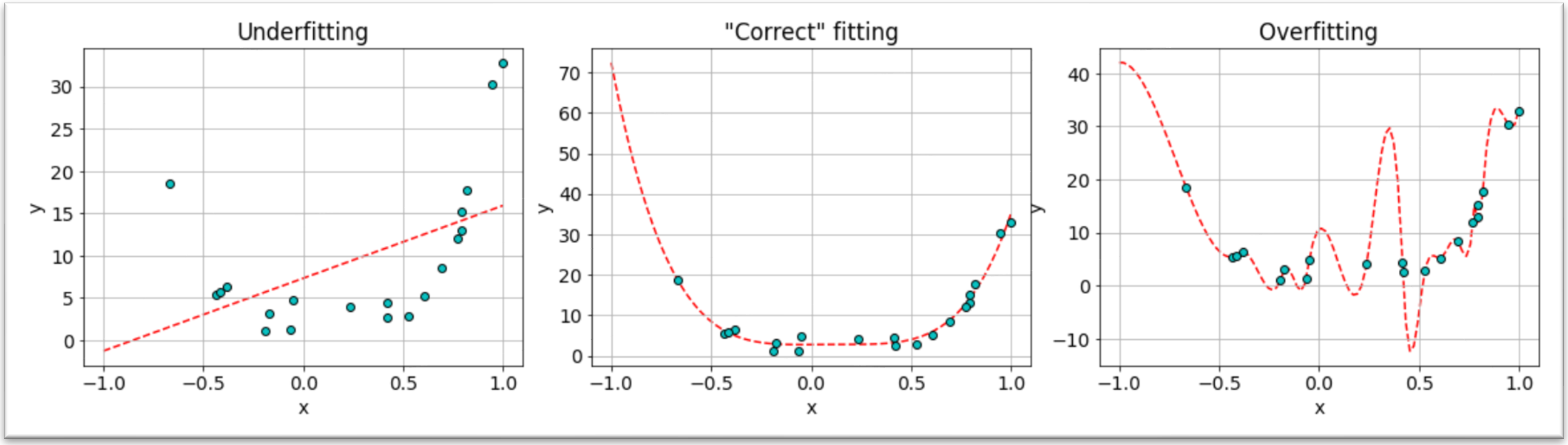
Regularization

Data Base and Data Mining Group of Politecnico di Torino

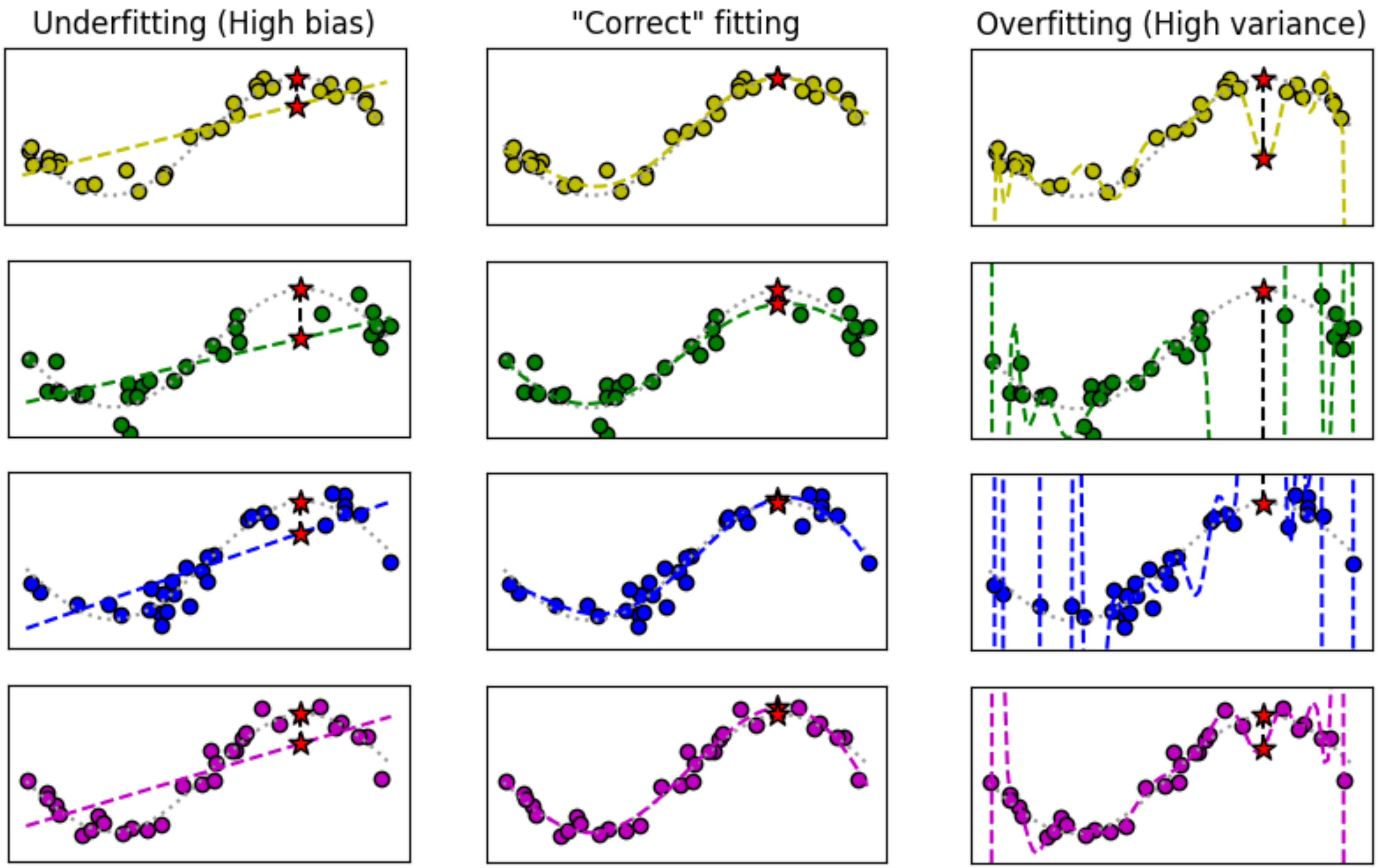
Overfitting and underfitting

- *Overfitting*: the model is too complex and fits the training data too closely (high variance)
  - Poor performance on test data

- *Underfitting*: the model is too simple and does not capture the underlying relationships (high bias)
  - Poor performance on training and test data



High bias vs high variance

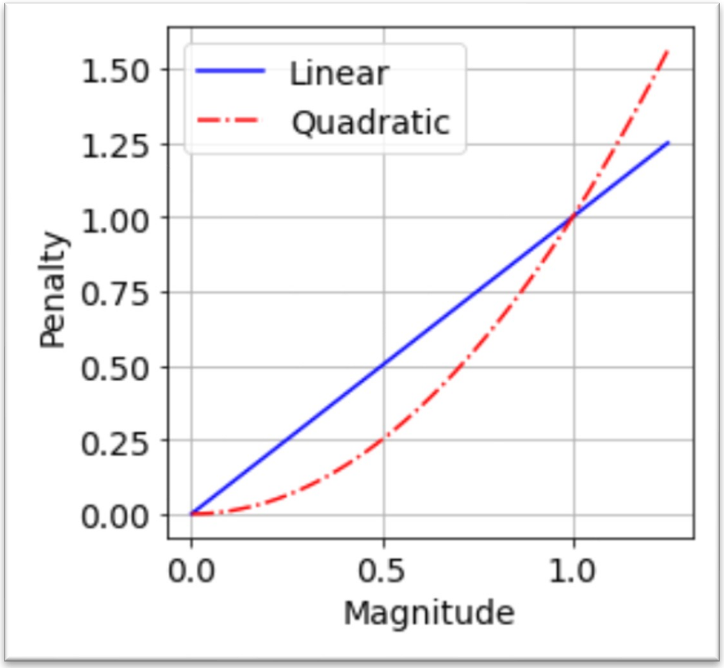


Preventing overfitting

- We can generally prevent overfitting by:
  - Reducing model capacity
    - (e.g., reduce the polynomial degree used)
  - Increasing the dataset size
  - Introducing regularization techniques

Regularization techniques

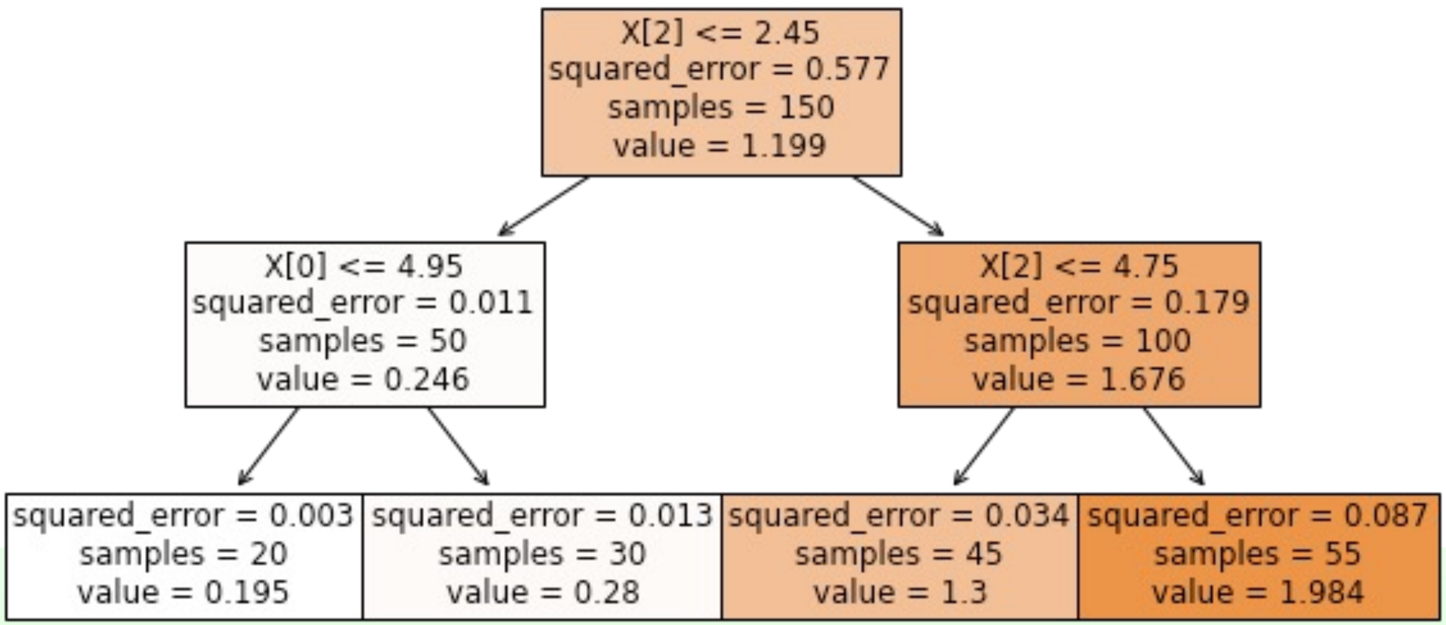
- Allow model to use high capacity, but penalize it if used unnecessarily
- Penalty term in the cost function
- L1 (Lasso) penalizes all non-zero weights linearly
  - $Cost = MSE + \lambda ||\boldsymbol{\theta}||_1$
  - Bring θ values to 0 if not strictly needed
- L2 (Ridge) penalizes ≈ 0 values less than ≈ 0 values
  - $Cost = MSE + \lambda ||\boldsymbol{\theta}||_2$
  - Allows θ values to be ≈ 0 for small contributions



Other regressors

Tree-based regression

- We can build decision trees for regression
  - Real values used as targets instead of classes
  - Node impurity computed as variance
  - Each leaf assigns average value of points in it



Other techniques

- Random forests can be obtained by aggregating the output of decision tree regressors (e.g., by averaging them)
- In KNN, we can produce the predicted outcome as the (possibly weighted) average of the neighbors' "votes"
- Neural networks natively produce continuous outputs