Machine Learning
Lecture 2:
Linear Models

Young && Yandex ШАД

girafe ai

Radoslav Neychev

Outline

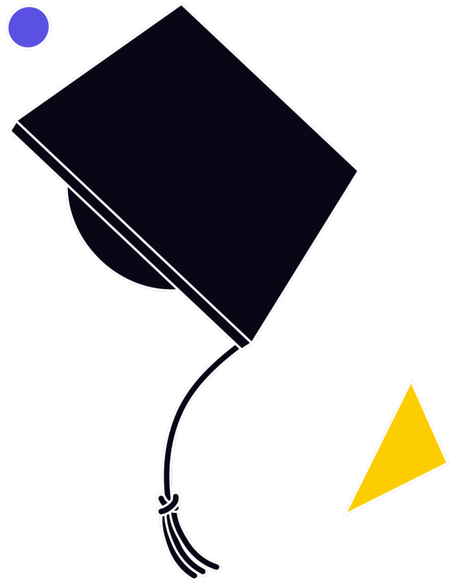
01 Linear models overview

02 Linear Regression under the hood

03 Gauss-Markov theorem

04 Regularization in Linear regression

05 Model validation and evaluation



Previous lecture recap

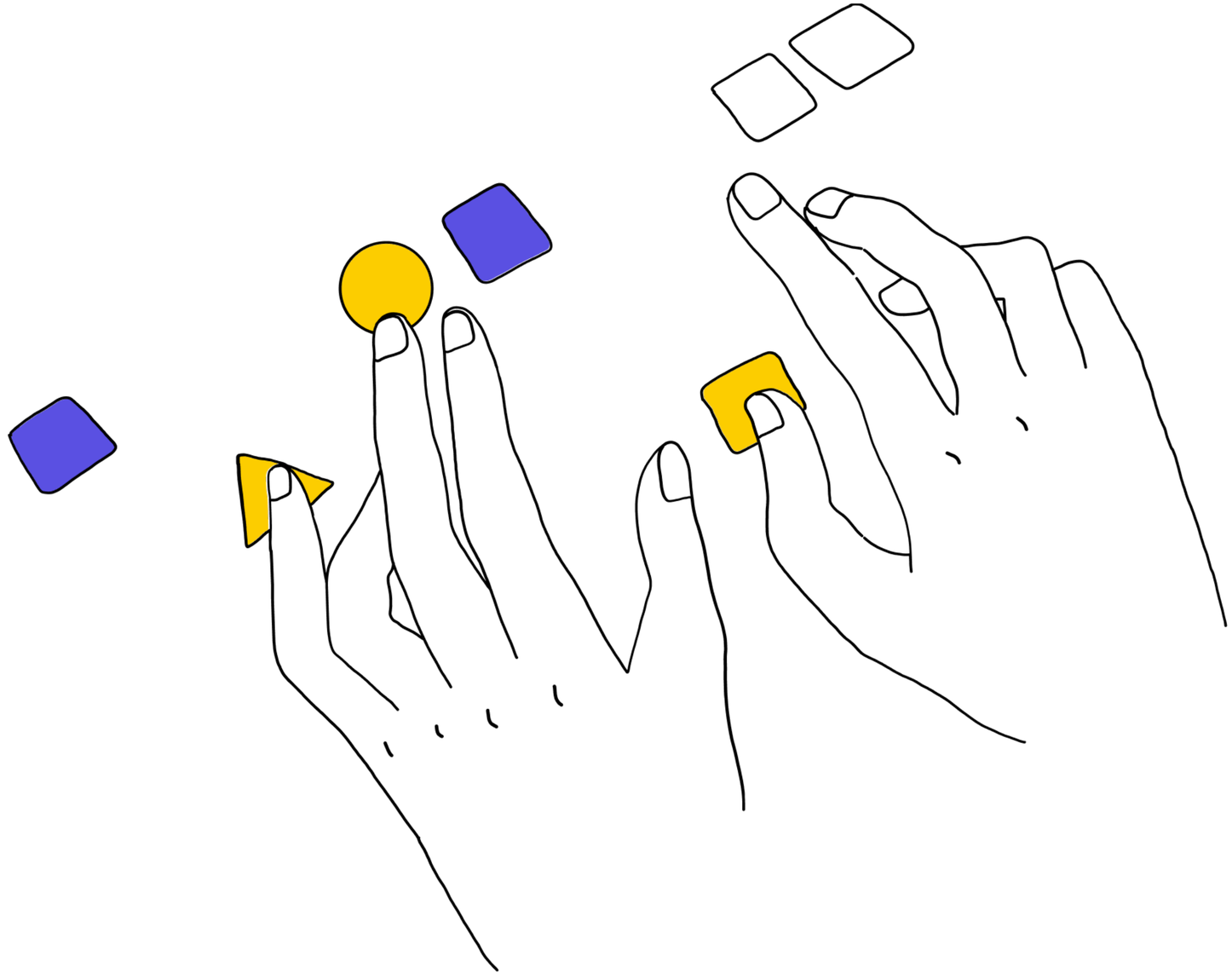
01 Dataset, observation, feature, design matrix, target

02 i.i.d. property

03 Model, prediction, loss/quality function

04 Parameter, Hyperparameter

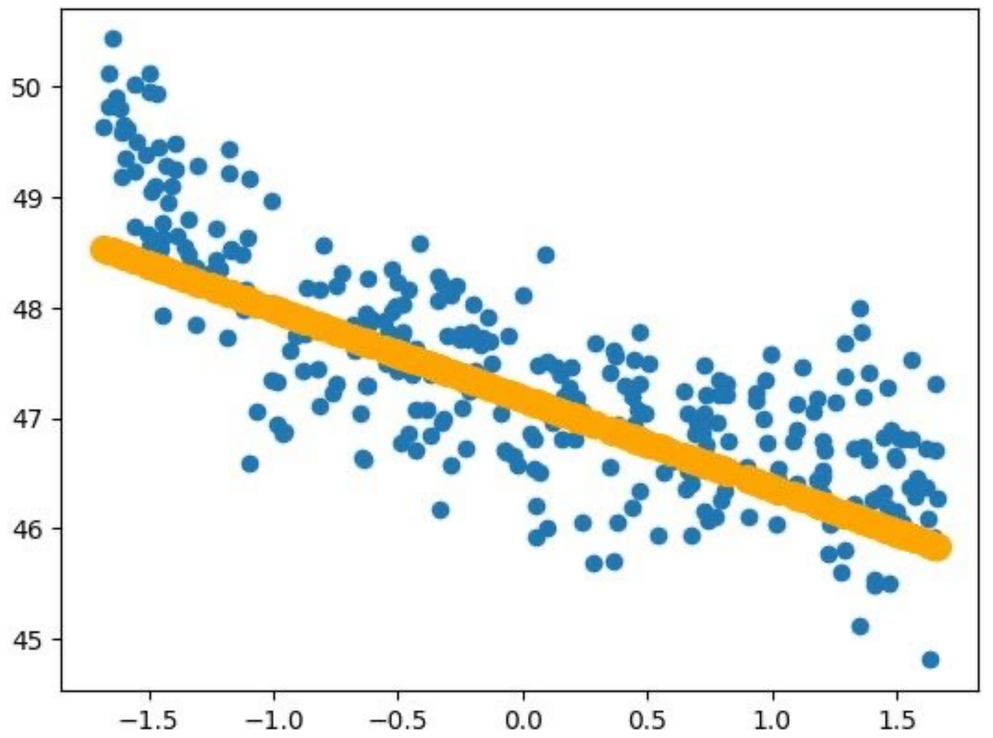
Linear Models



01

Linear models

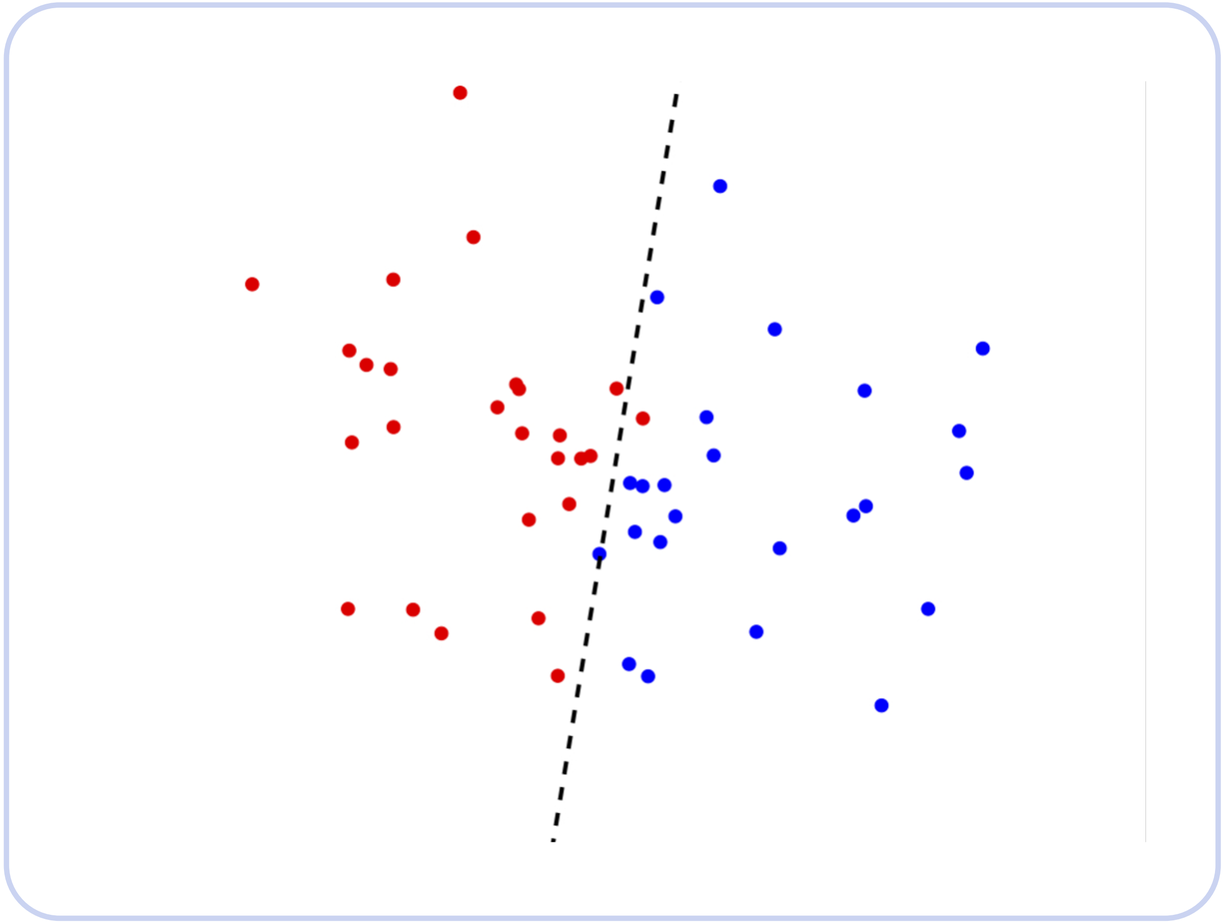
- Regression models



$$y = \boldsymbol{w}^\top \boldsymbol{x} + b = \sum_{i=1}^{p} w_i x_i + b$$

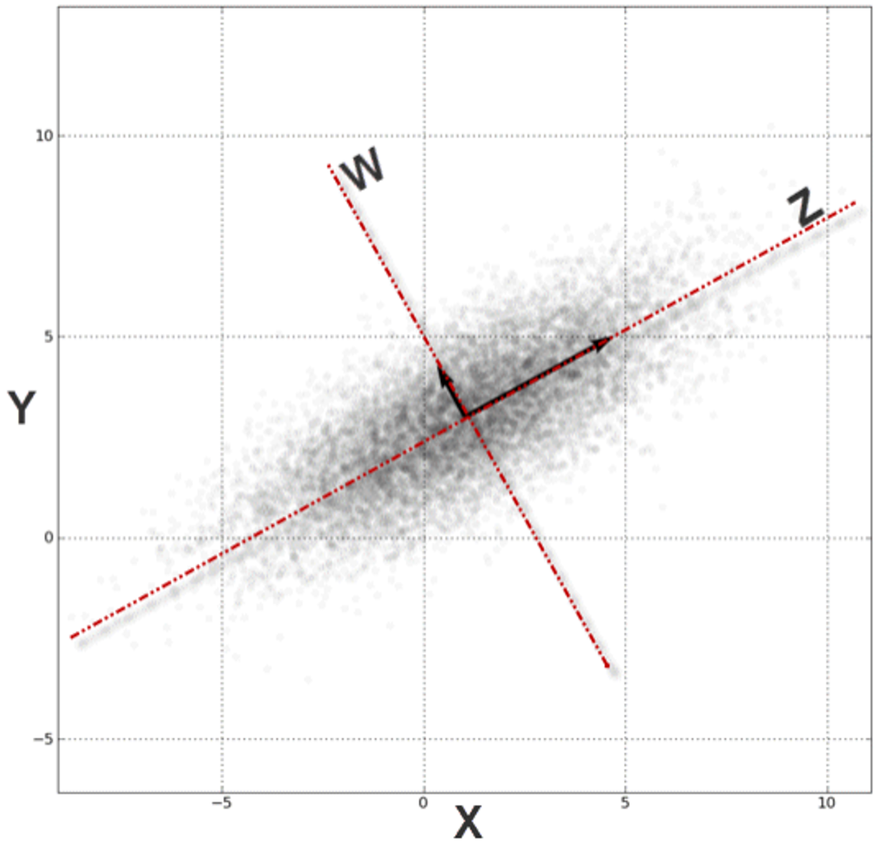
Linear models

- Regression models
- Classification models



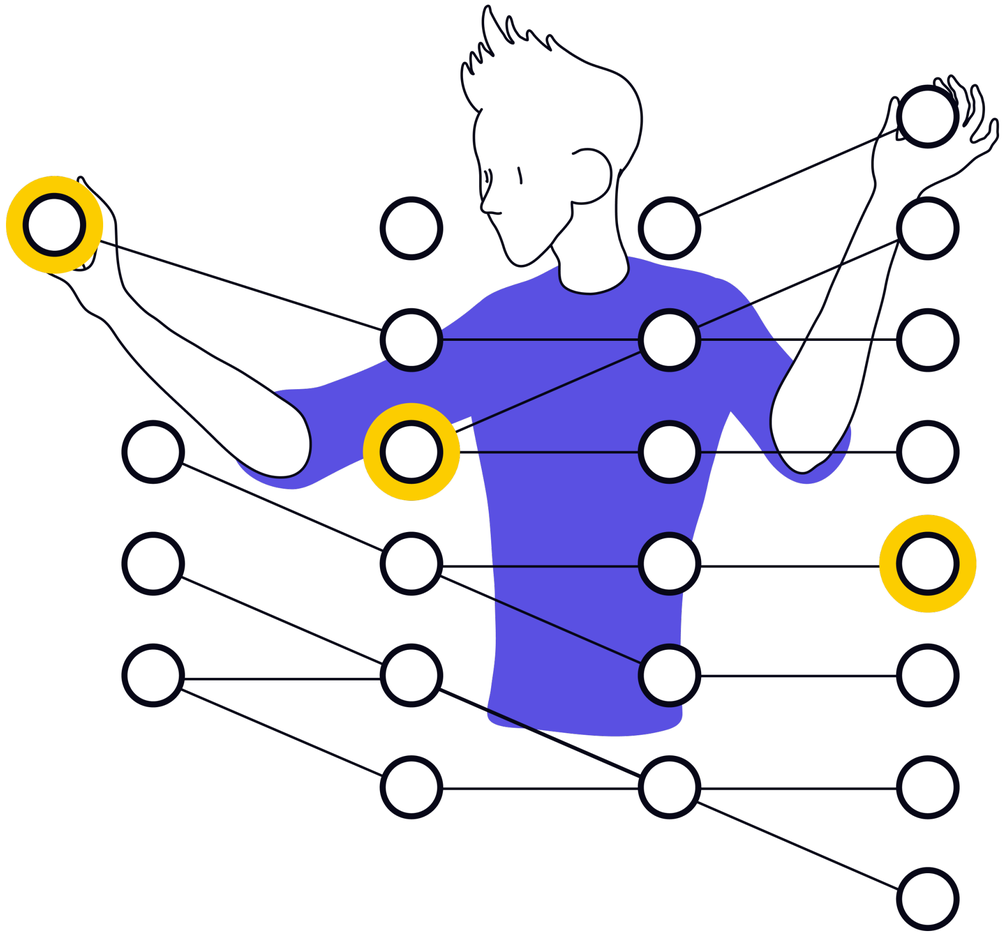
Linear models

- Regression models
- Classification models
- Unsupervised models
  (e.g. PCA analysis):

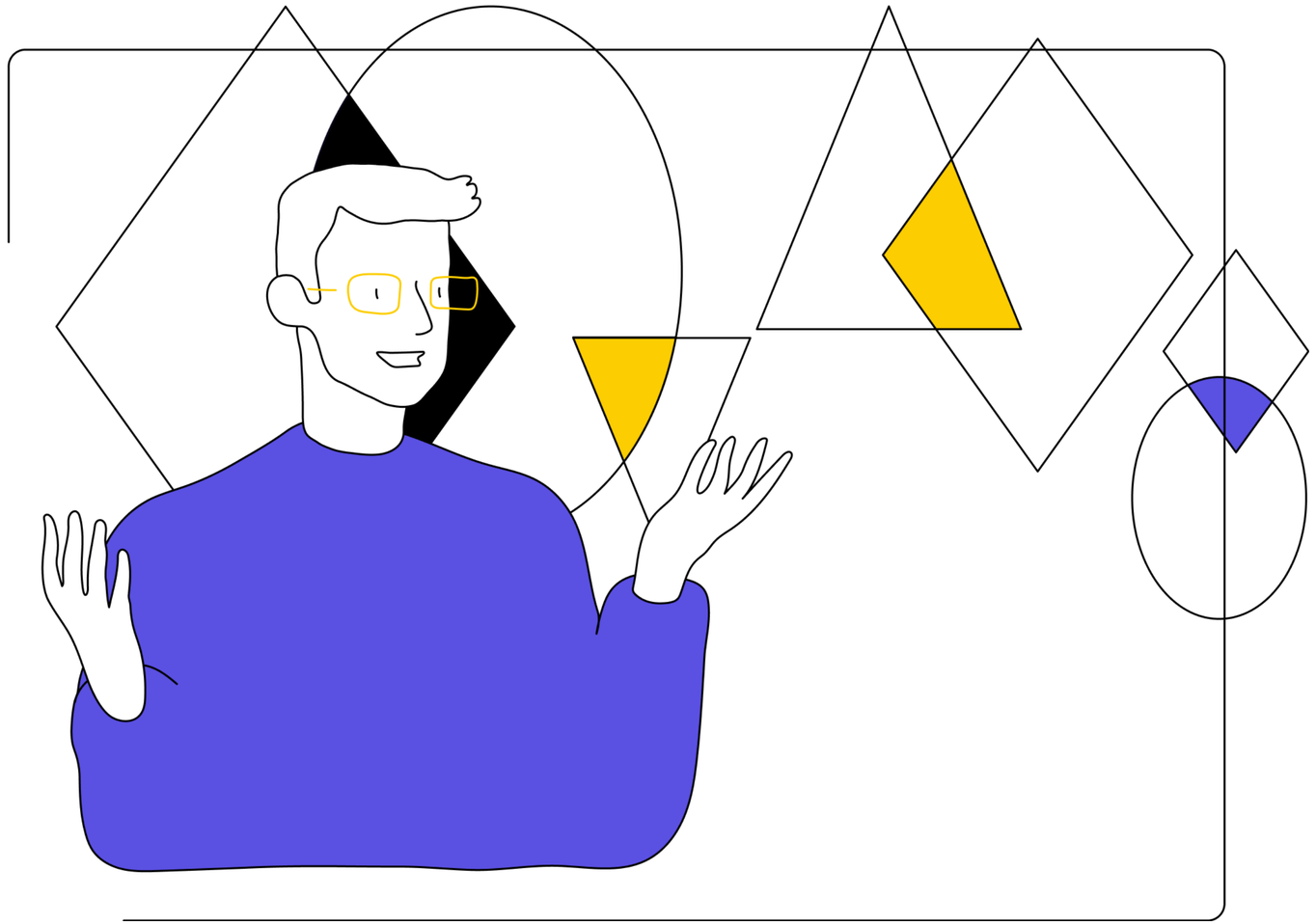


Linear models

- Regression models
- Classification models
- Unsupervised models
  (e.g. PCA analysis):
- Building block of other models (ensembles, NNs, etc.):



Linear Regression



02

Linear regression

Linear regression problem statement:

- Dataset $\{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$, where $\boldsymbol{x}^{(i)} \in \mathbb{R}^p, y^{(i)} \in \mathbb{R}$.

Linear regression

Linear regression problem statement:

- Dataset $\{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$, where $\boldsymbol{x}^{(i)} \in \mathbb{R}^p, y^{(i)} \in \mathbb{R}$.

- The model is linear:

$$\hat{y} = w_0 + \sum_{k=1}^{p} x_k w_k = //\boldsymbol{x} = [1; x_1, \ldots, x_p]// = \boldsymbol{x}^\top \boldsymbol{w}$$

we added an additional column of 1's to the design matrix to simplify the formulas

Linear regression

Linear regression problem statement:

- Dataset $\{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$, where $\boldsymbol{x}^{(i)} \in \mathbb{R}^p, y^{(i)} \in \mathbb{R}$.

- The model is linear:

$$\hat{y} = w_0 + \sum_{k=1}^{p} x_k w_k = //\boldsymbol{x} = [1; x_1, \ldots, x_p]// = \boldsymbol{x}^\top \boldsymbol{w}$$

where $\boldsymbol{w} = [w_0; w_1, \ldots, w_p]$ is bias term.

we added an additional column of 1's to the design matrix to simplify the formulas

Linear regression

Linear regression problem statement:

- Dataset $\{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$, where $\boldsymbol{x}^{(i)} \in \mathbb{R}^p, y^{(i)} \in \mathbb{R}$.

- The model is linear:

$$\hat{y} = w_0 + \sum_{k=1}^{p} x_k w_k = //\boldsymbol{x} = [1; x_1, \ldots, x_p]// = \boldsymbol{x}^\top \boldsymbol{w}$$

where $\boldsymbol{w} = [w_0; w_1, \ldots, w_p]$ is bias term.

- Least squares method (MSE minimization) provides a solution:

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} ||\mathbf{Y} - \mathbf{X}\boldsymbol{w}||_2^2$$

Analytical solution

Denote quadratic loss function:

$$L(\mathbf{X}, \mathbf{Y}, \boldsymbol{w}) = (\mathbf{Y} - \mathbf{X}\boldsymbol{w})^\top (\mathbf{Y} - \mathbf{X}\boldsymbol{w}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{w}\|_2^2 ,$$

where $\mathbf{X} = [\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}]^\top, \quad \boldsymbol{x}^{(i)} \in \mathbb{R}^p.$

To find optimal solution let's equal to zero the derivative of the equation above:

$$\nabla_{\boldsymbol{w}} Q(\boldsymbol{w}) = \nabla_{\boldsymbol{w}} [\mathbf{Y}^\top \mathbf{Y} - \mathbf{Y}^\top \mathbf{X}\boldsymbol{w} - \boldsymbol{w}^\top \mathbf{X}^\top \mathbf{Y} + \boldsymbol{w}^\top \mathbf{X}^\top \mathbf{X}\boldsymbol{w}] = 0$$

Analytical solution

Denote quadratic loss function:

$$L(\mathbf{X}, \mathbf{Y}, \boldsymbol{w}) = (\mathbf{Y} - \mathbf{X}\boldsymbol{w})^\top (\mathbf{Y} - \mathbf{X}\boldsymbol{w}) = \|\mathbf{Y} - \mathbf{X}\boldsymbol{w}\|_2^2 ,$$

where $\mathbf{X} = [\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}]^\top, \quad \boldsymbol{x}^{(i)} \in \mathbb{R}^p.$

To find optimal solution let's equal to zero the derivative of the equation above:

$$\nabla_{\boldsymbol{w}} Q(\boldsymbol{w}) = \nabla_{\boldsymbol{w}} [\mathbf{Y}^\top \mathbf{Y} - \mathbf{Y}^\top \mathbf{X}\boldsymbol{w} - \boldsymbol{w}^\top \mathbf{X}^\top \mathbf{Y} + \boldsymbol{w}^\top \mathbf{X}^\top \mathbf{X}\boldsymbol{w}] = 0$$

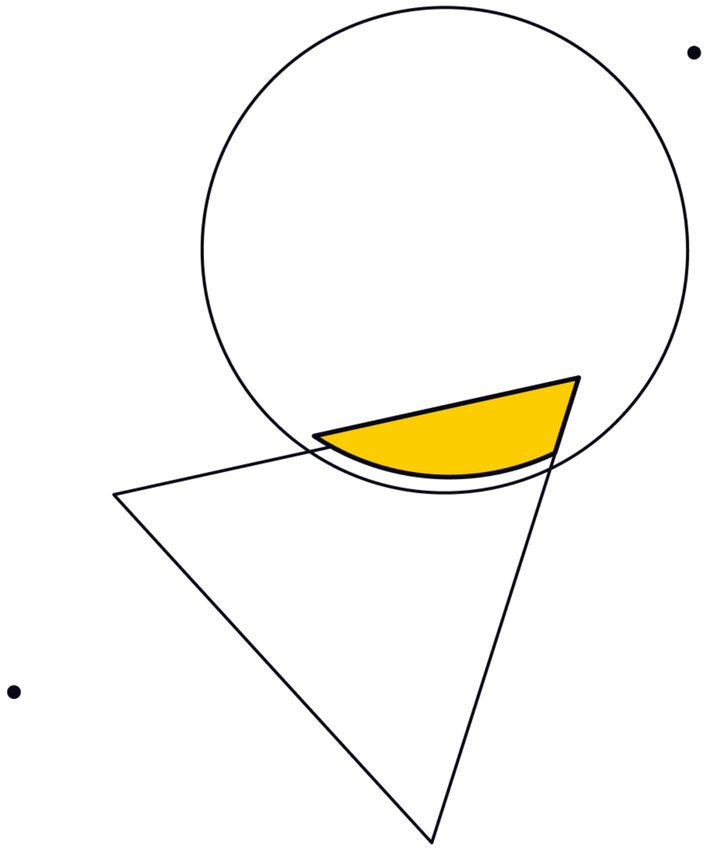
$$\hat{\boldsymbol{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$$

what if this matrix is singular?

Analytical solution

$$\hat{\boldsymbol{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$$

what if this matrix is singular?



Unstable solution

In case of multicollinear features the matrix $\mathbf{X}^\top \mathbf{X}$ is almost singular.

It leads to unstable solution:

```
w_true
```

```
array([ 2.68647887, -0.52184084, -1.12776533])
```

```
w_star = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(Y)
w_star
```

```
array([   2.68027723, -186.0552577 ,  184.41701118])
```

Unstable solution

In case of multicollinear features the matrix $\mathbf{X}^\top \mathbf{X}$ is almost singular.

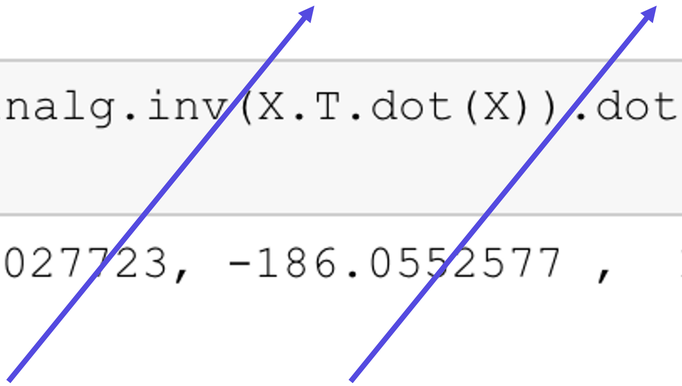
It leads to unstable solution:

```
w_true
```

```
array([ 2.68647887, -0.52184084, -1.12776533])
```

```
w_star = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(Y)
w_star
```

```
array([   2.68027723, -186.0552577 ,  184.41701118])
```



corresponding features
are almost collinear

Unstable solution

In case of multicollinear features the matrix $\mathbf{X}^\top \mathbf{X}$ is almost singular.

It leads to unstable solution:

```
w_true
```

```
array([ 2.68647887, -0.52184084, -1.12776533])
```

```
w_star = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(Y)
w_star
```

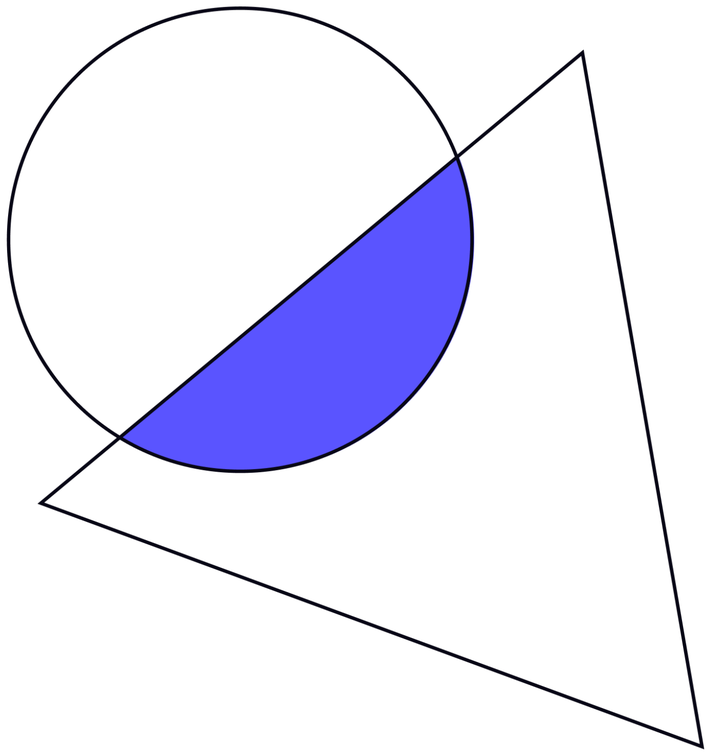
```
array([   2.68027723, -186.0552577 ,  184.41701118])
```

the coefficients are huge
and sum up to almost 0

Regularization

To make the matrix nonsingular, we can add a diagonal matrix:

$$\hat{\boldsymbol{w}} = (\mathbf{X}^\top \mathbf{X} + \lambda \boldsymbol{I})^{-1} \mathbf{X}^\top \mathbf{Y},$$

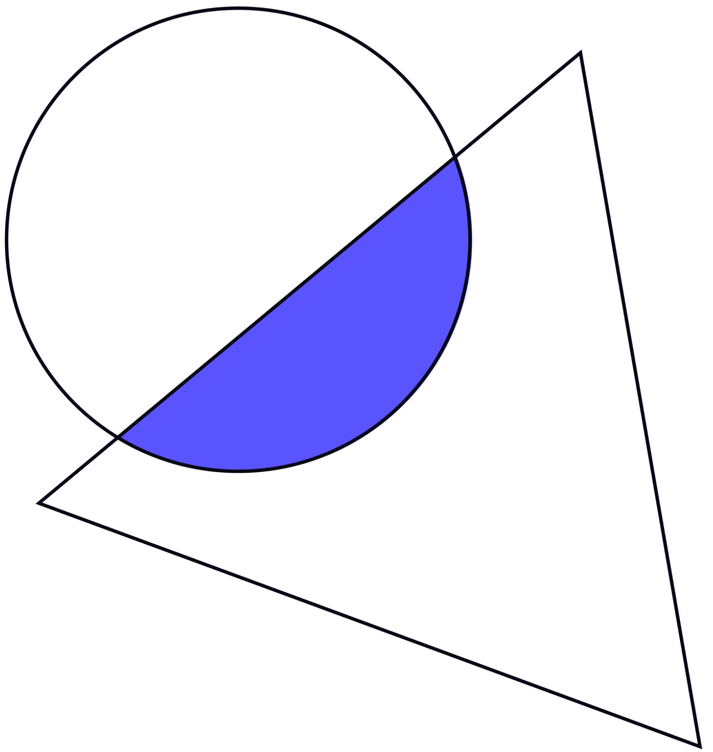


Regularization

To make the matrix nonsingular, we can add a diagonal matrix:

$$\hat{\boldsymbol{w}} = (\mathbf{X}^\top \mathbf{X} + \lambda \boldsymbol{I})^{-1} \mathbf{X}^\top \mathbf{Y},$$

where $\boldsymbol{I} = \mathrm{diag}[1_i, \ldots, 1_p]$.

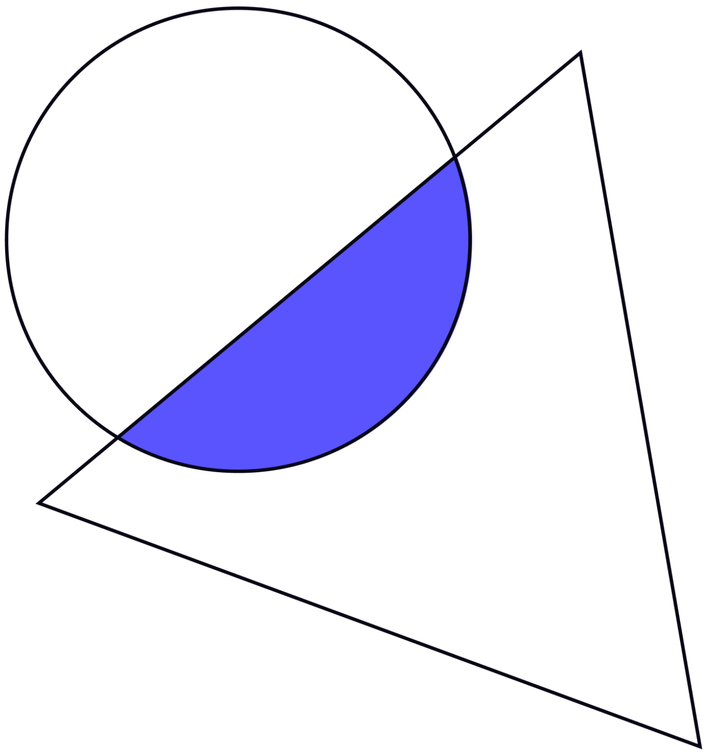


Regularization

To make the matrix nonsingular, we can add a diagonal matrix:

$$\hat{\boldsymbol{w}} = (\mathbf{X}^\top \mathbf{X} + \lambda \boldsymbol{I})^{-1} \mathbf{X}^\top \mathbf{Y},$$

where $\boldsymbol{I} = \mathrm{diag}[1_i, \ldots, 1_p]$.



Actually, it's a solution for the following loss function:

$$Q(\boldsymbol{w}) = \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{w}\|_2^2 + \lambda \|\boldsymbol{w}\|_2^2,$$

exercise: derive it by yourself

Gauss-Markov theorem

03

Gauss-Markov theorem

Suppose target values are expressed in following form:

$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{w} + \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon} = [\varepsilon_1, \ldots, \varepsilon_N]$ are random variables

Gauss–Markov assumptions:

- $\mathbb{E}(\varepsilon_i) = 0 \quad \forall i$
- $\mathrm{Var}(\varepsilon_i) = \sigma^2 < \inf \quad \forall i$
- $\mathrm{Cov}(\varepsilon_i, \varepsilon_j) = 0 \quad \forall i \neq j$

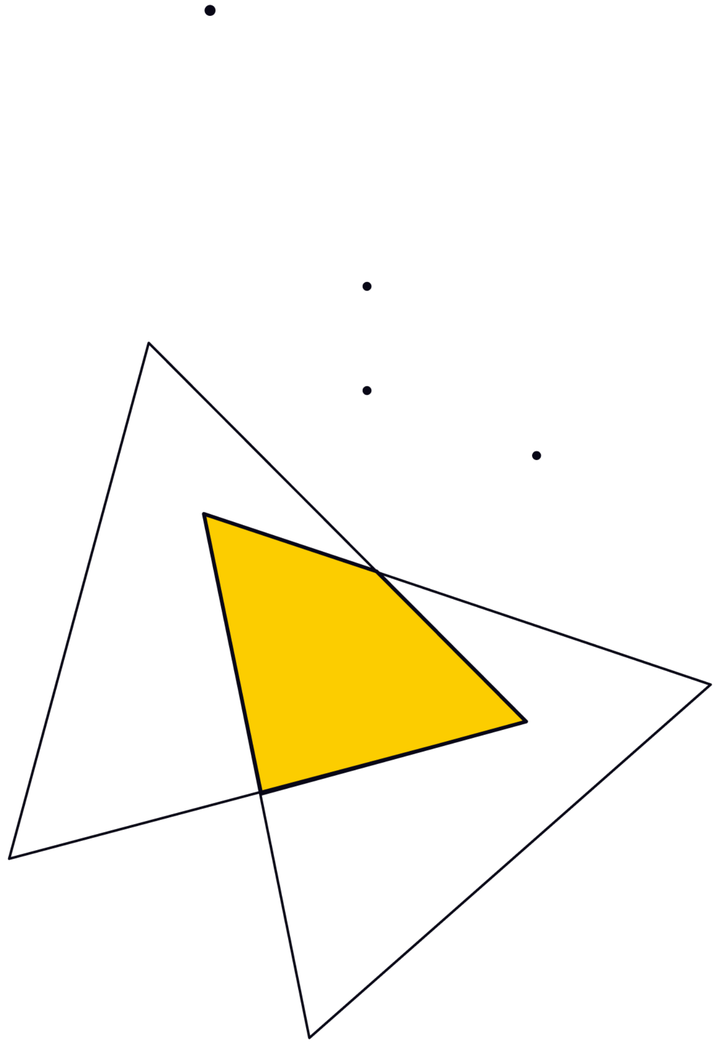
Gauss-Markov theorem

Gauss–Markov assumptions:

- $\mathbb{E}(\varepsilon_i) = 0 \quad \forall i$
- $\mathrm{Var}(\varepsilon_i) = \sigma^2 < \inf \quad \forall i$
- $\mathrm{Cov}(\varepsilon_i, \varepsilon_j) = 0 \quad \forall i \neq j$

$$\hat{\boldsymbol{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$$

delivers **B**est **L**inear **U**nbiased **E**stimator



Loss functions in regression

$$\mathrm{MSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{n}\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|_2^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\mathrm{MAE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{n}\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|_1 = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Different norms

Once more: loss functions:

$$\mathrm{MSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{n}\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|_2^2$$

$$\mathrm{MAE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{n}\|\boldsymbol{y} - \hat{\boldsymbol{y}}\|_1$$

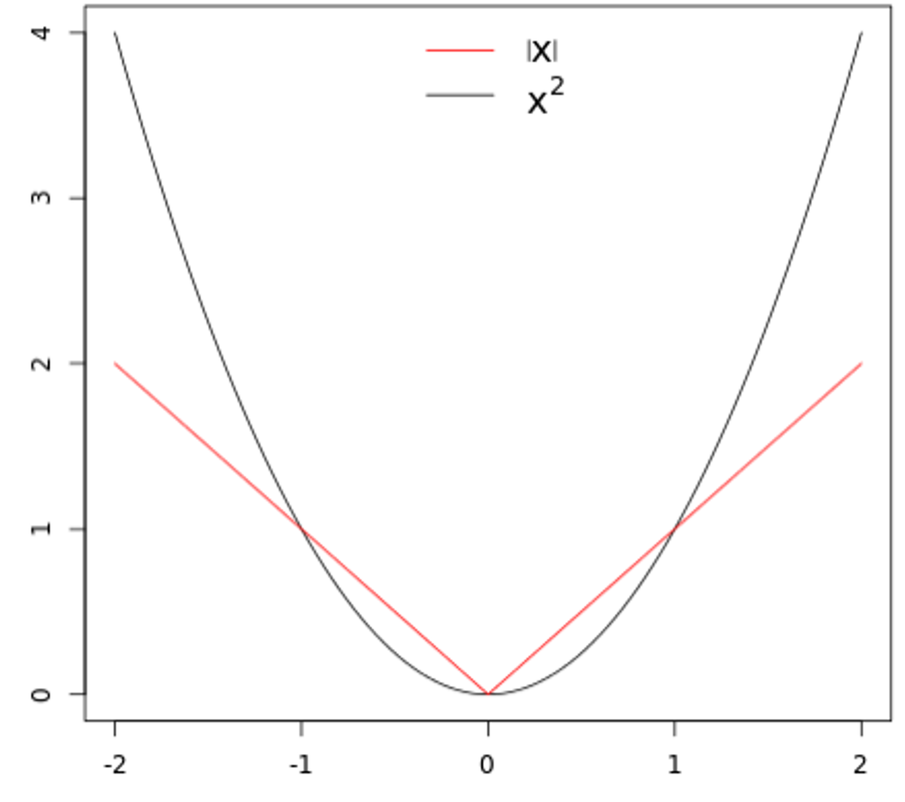
only works for Gauss-Markov theorem

Regularization terms:

- L2 $\|\boldsymbol{w}\|_2^2$
- L1 $\|\boldsymbol{w}\|_1$

What’s the difference?

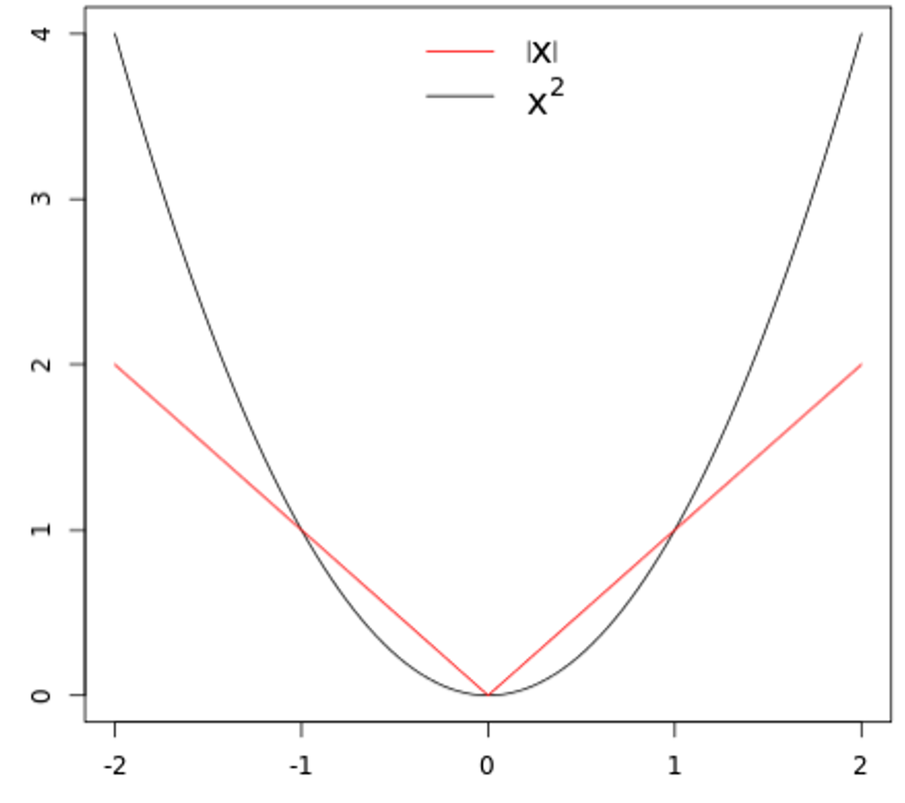
- MSE (L_2)
  - delivers BLUE according to Gauss-Markov theorem
  - differentiable
  - sensitive to noise
- MAE (L_1)
  - non-differentiable (not a problem)
  - much more prone to noise



What’s the difference?

- L_2 regularization
  - constraints weights
  - delivers more stable solution
  - differentiable

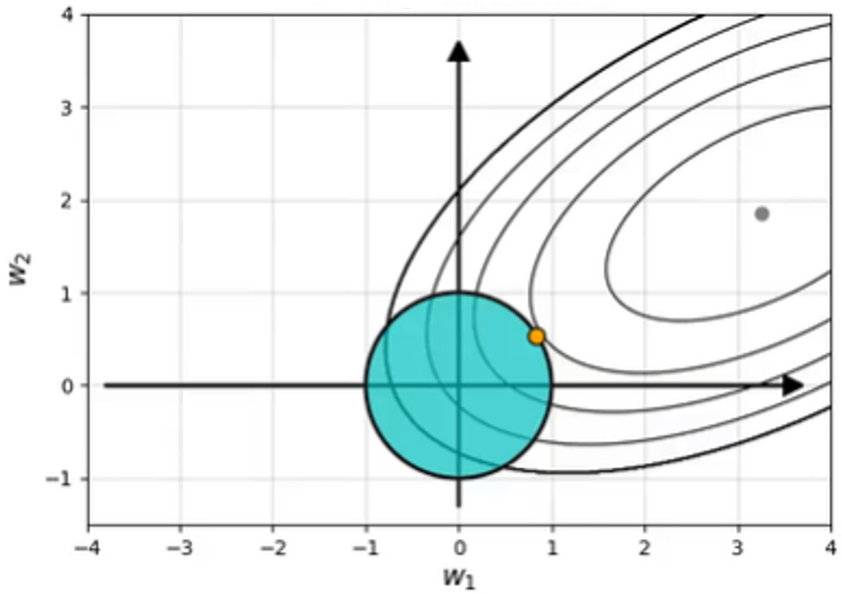
- L_1 regularization
  - non-differentiable
  - not a problem
  - selects features



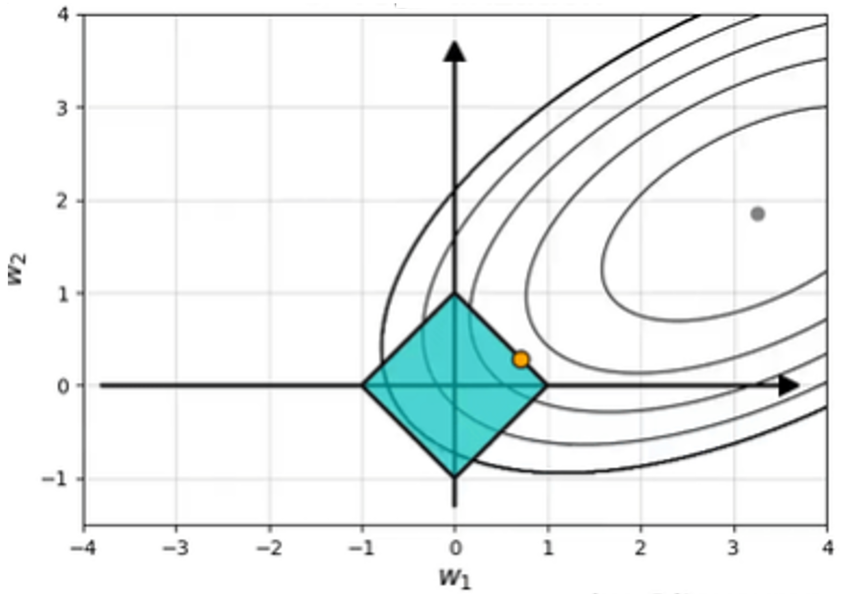
What’s the difference?

L1 induces sparse solutions for least squares

L2 regularization



L1 regularization

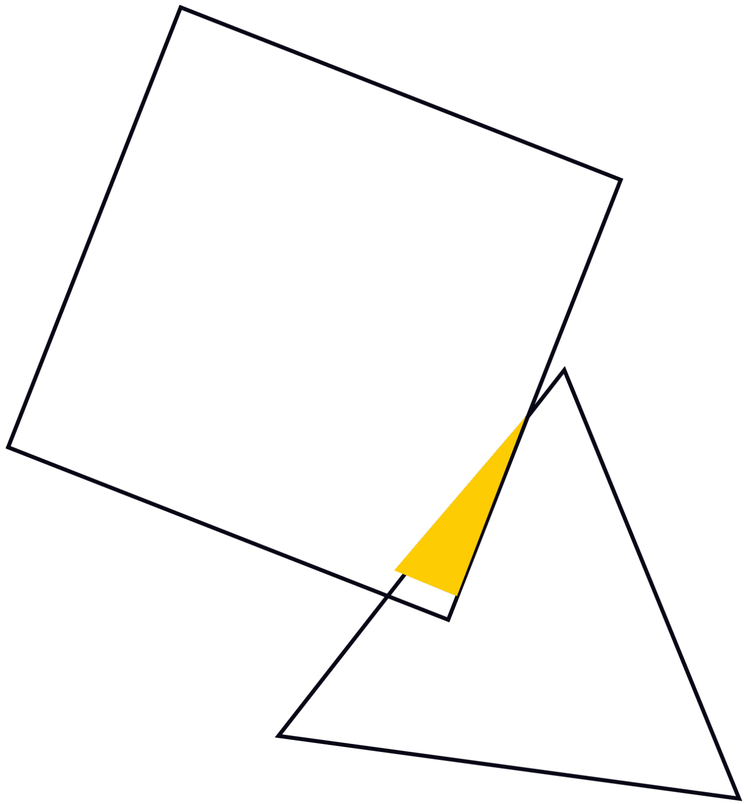


source: https://people.eecs.berkeley.edu/~jrs/189/

Loss functions in regression

Other functions to measure the quality in regression:

- R2 score
- MAPE
- SMAPE
- …



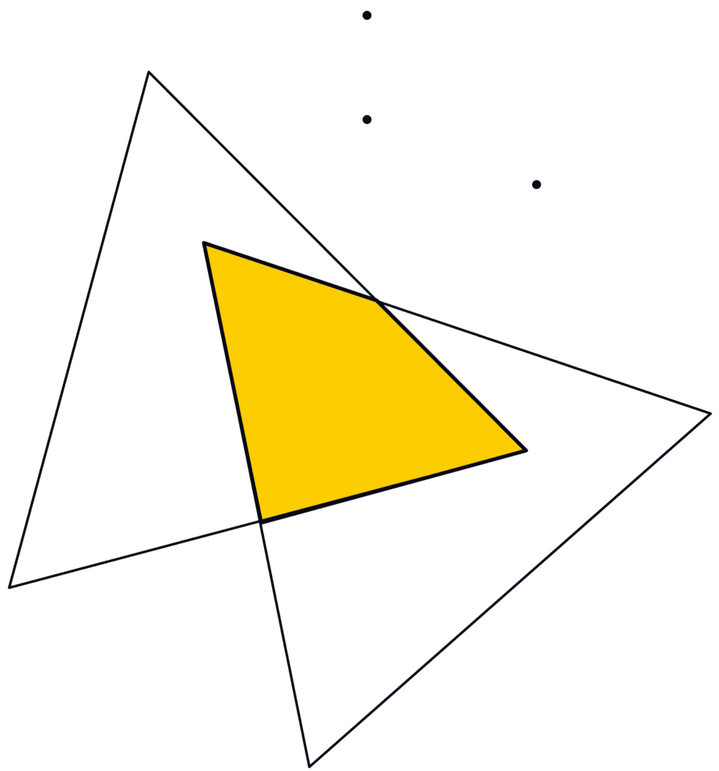
Model validation and evaluation

04

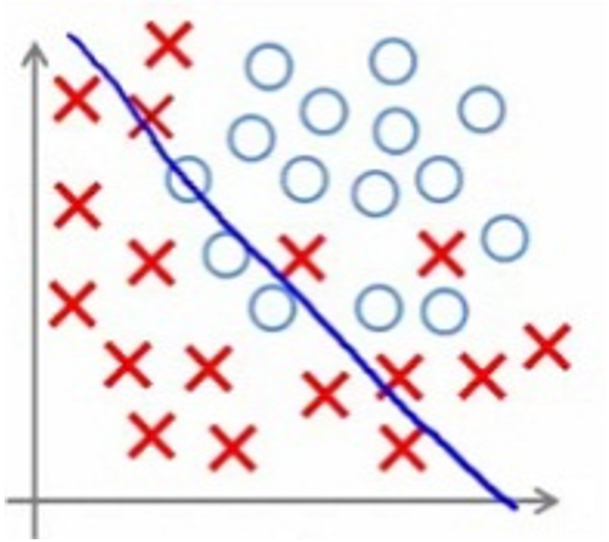
Supervised learning problem statement

Let’s denote:

- Training set $\{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$,where
  - $\boldsymbol{x}^{(i)} \in \mathbb{R}^p, y^{(i)} \in \mathbb{R}$ for regression
  - $\boldsymbol{x}^{(i)} \in \mathbb{R}^p, y^{(i)} \in \{C_1, \ldots, C_K\}$ for classification
- Model $f(\boldsymbol{x})$ predicts some value for every object
- Loss function $L(\boldsymbol{x}, y, f)$ that should be minimized

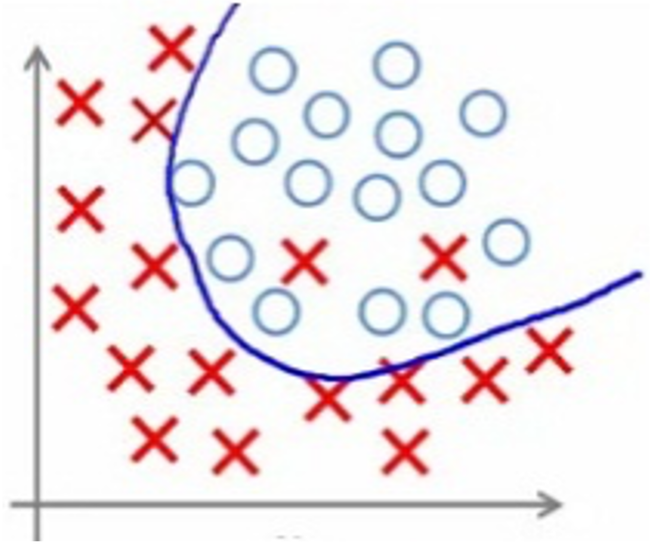


Overfitting vs. underfitting

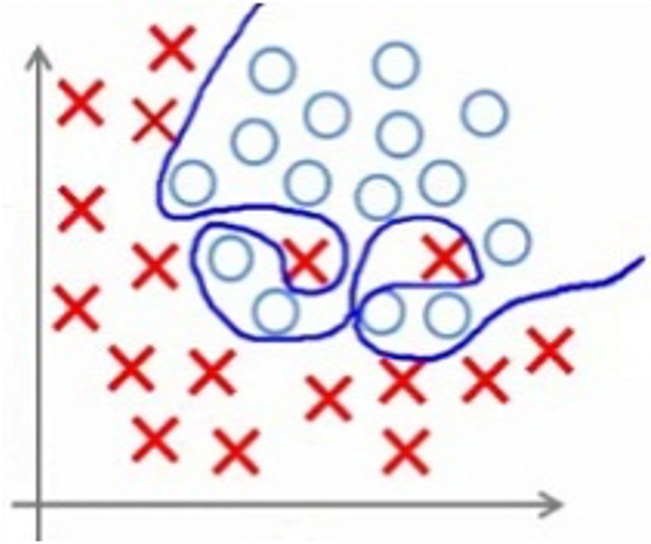


Under-fitting

(too simple to explain the variance)



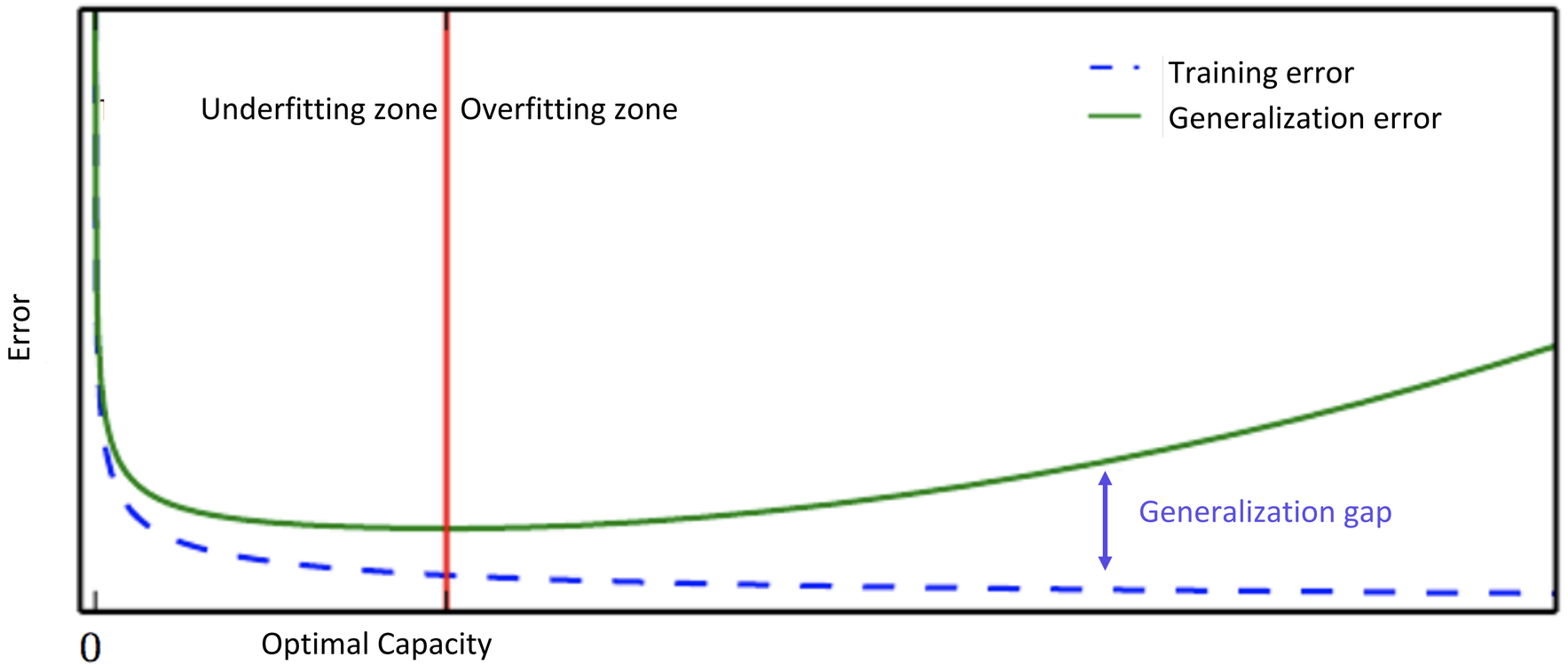
Appropriate-fitting



Over-fitting

(forcefitting — too good to be true)

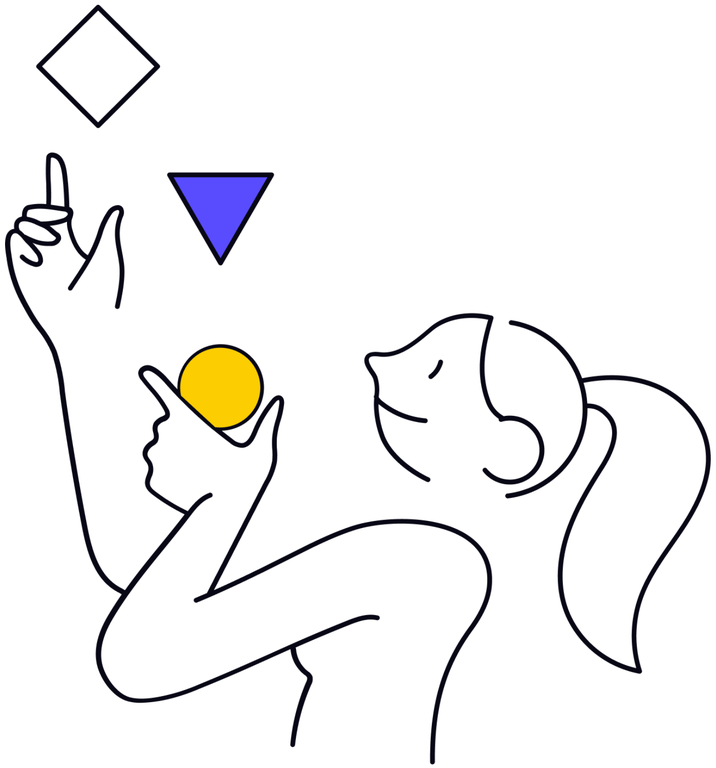
Overfitting vs. underfitting



Overfitting vs. underfitting

We can control overfitting / underfitting by altering model's capacity (ability to fit a wide variety of functions):

- select appropriate hypothesis space
- learning algorithm's effective capacity may be less than the representational capacity of the model family



Evaluating the quality

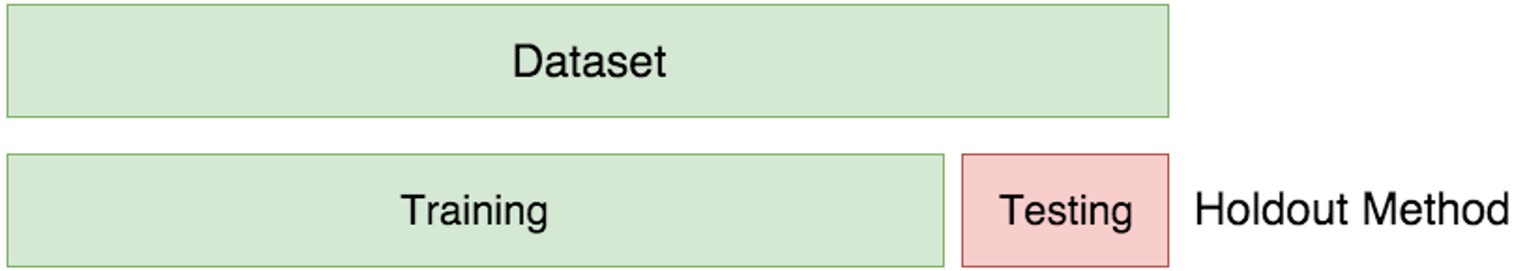
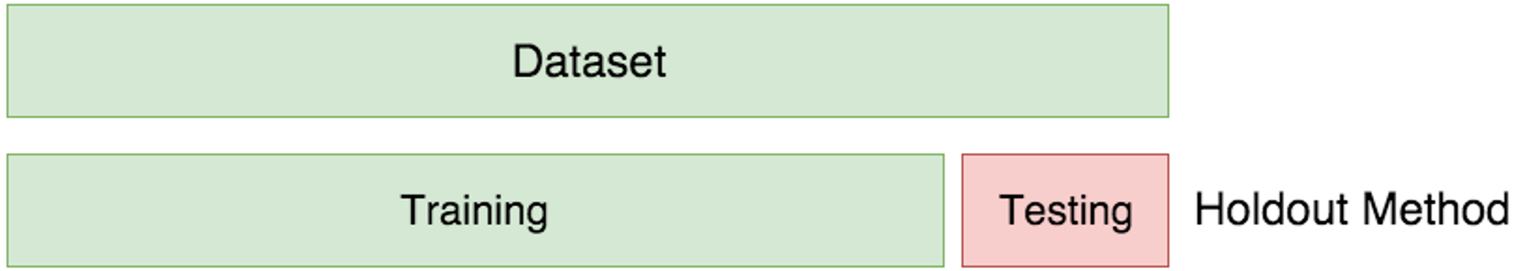


Image credit: Joseph Nelson @josephofiowa

Evaluating the quality



Holdout Method

Is it good enough?

Image credit: Joseph Nelson @josephofiowa

Evaluating the quality

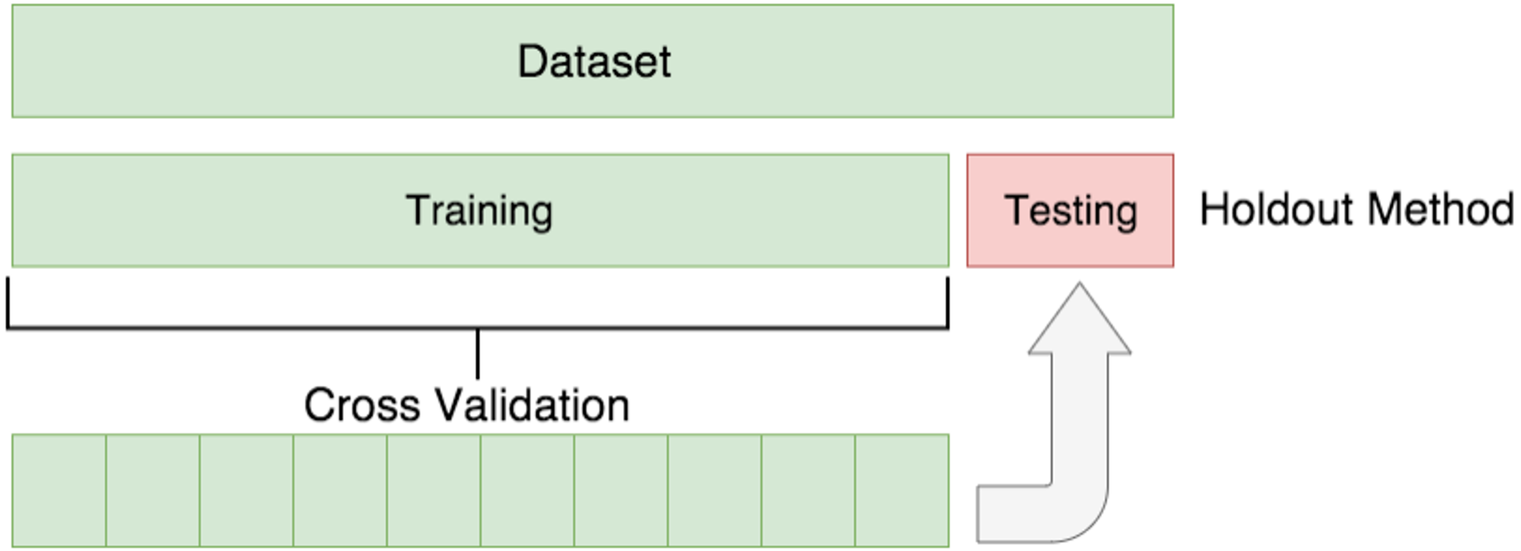


Image credit: Joseph Nelson @josephofiowa

Evaluating the quality

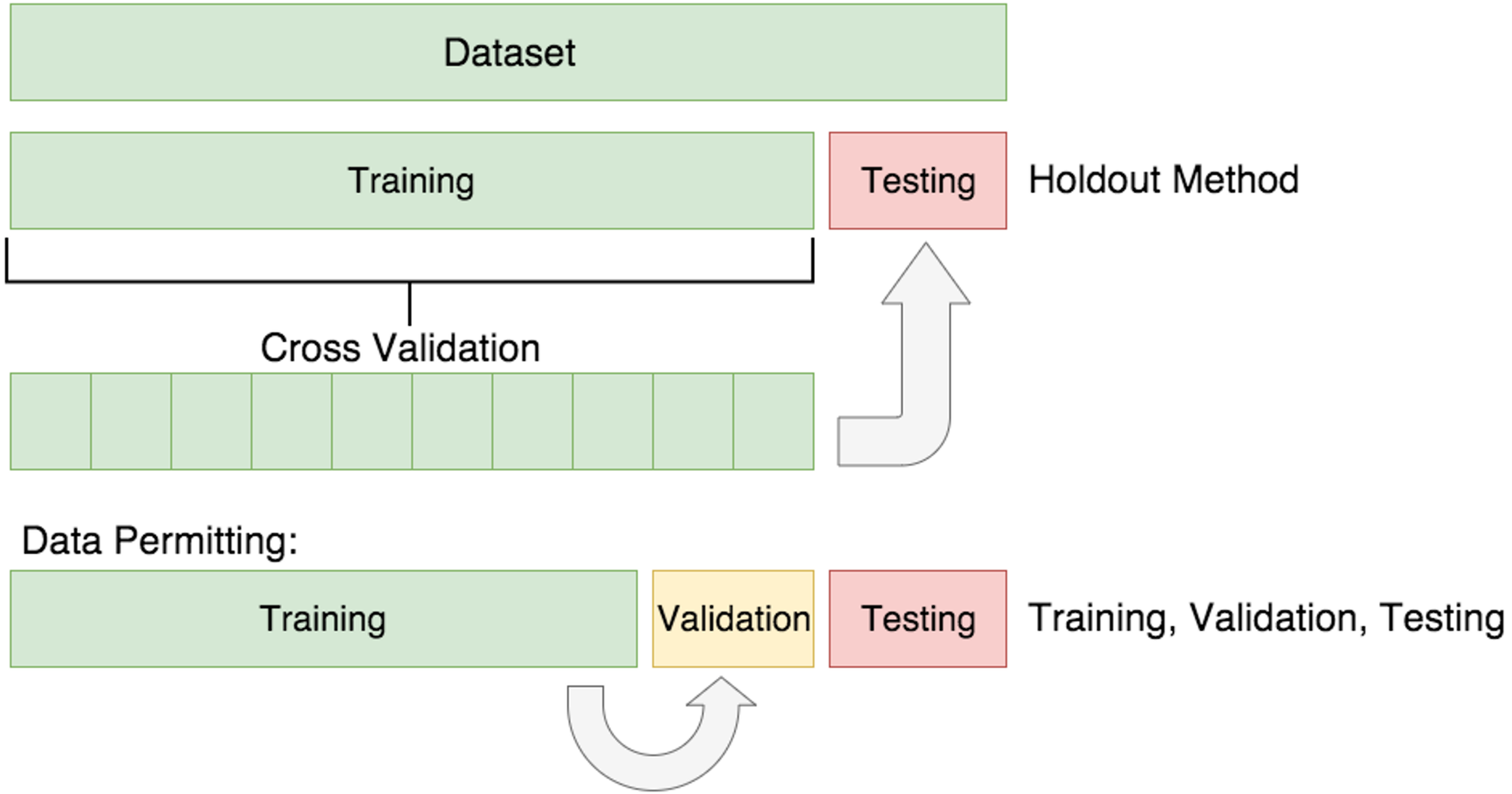
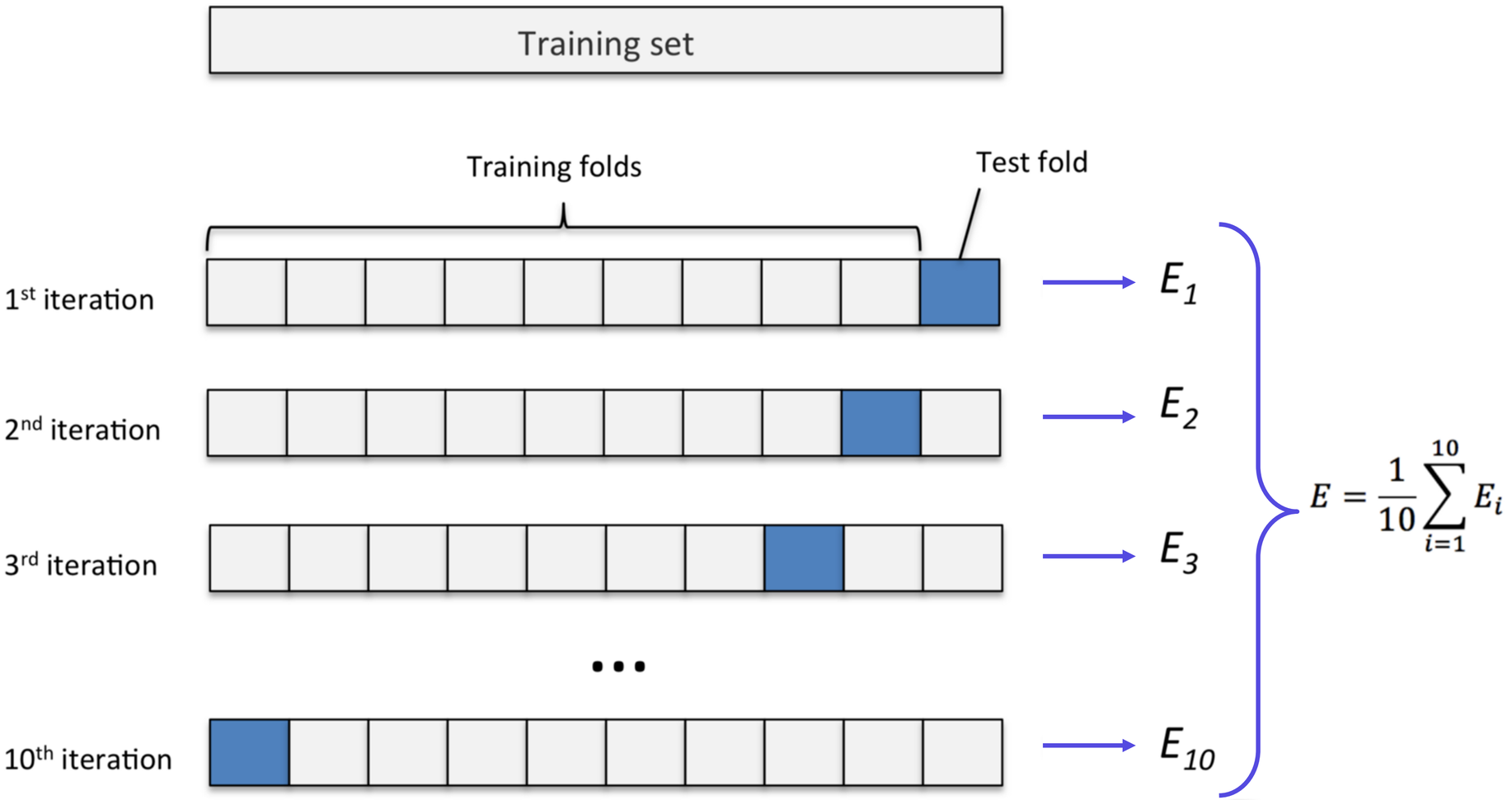


Image credit: Joseph Nelson @josephofiowa

Cross-validation



Outro

- Linear models are simple yet quite effective models
- Regularization incorporates some prior assumptions/additional constraints
- Trust your validation