

Linear Regression

Shusen Wang

Warm-up: Vector and Matrix

Vector and Matrix

Vector (n -dim)

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}$$

Matrix ($n \times d$)

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1d} \\ a_{21} & a_{22} & \cdots & a_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nd} \end{bmatrix}$$

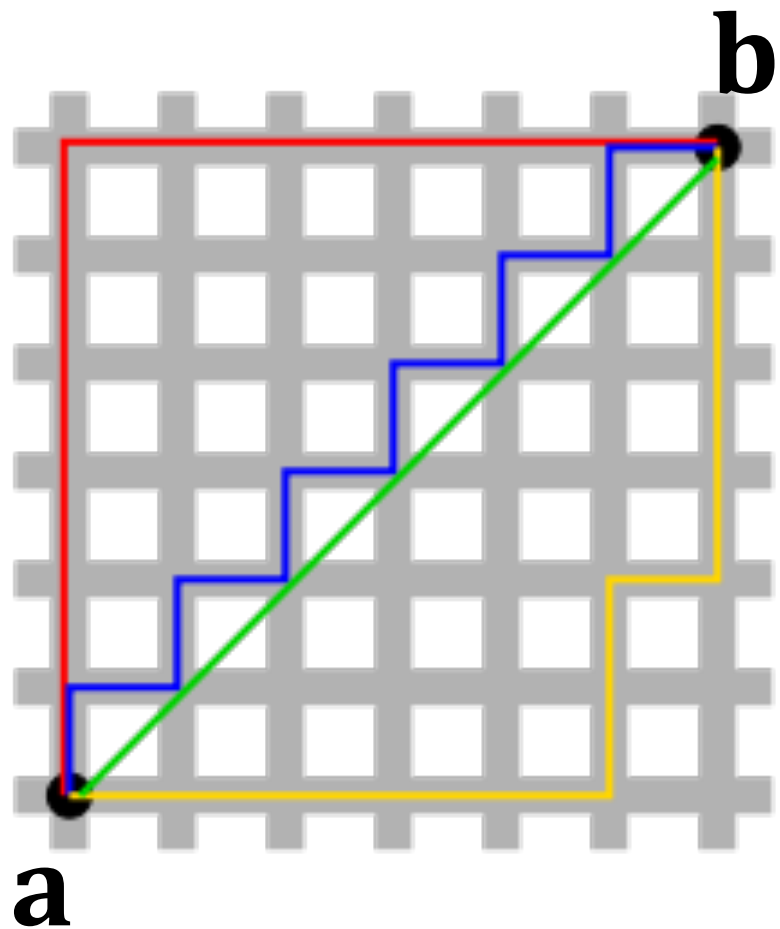
Row and columns

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_{:1} & \mathbf{a}_{:2} & \cdots & \mathbf{a}_{:d} \end{bmatrix} = \begin{bmatrix} \mathbf{a}_{1:} \\ \mathbf{a}_{2:} \\ \vdots \\ \mathbf{a}_{n:} \end{bmatrix}$$

Vector Norms

- The ℓ_p norm: $\|\mathbf{x}\|_p := \left(\sum_i |x_i|^p \right)^{1/p}$.
- The ℓ_2 norm: $\|\mathbf{x}\|_2 = \left(\sum_i x_i^2 \right)^{1/2}$ (the Euclidean norm).
- The ℓ_1 norm $\|\mathbf{x}\|_1 = \sum_i |x_i|$.
- The ℓ_∞ norm is defined by $\|\mathbf{x}\|_\infty = \max_i |x_i|$.

Vector Norms



- The ℓ_2 -distance (Euclidean distance):
 $||\mathbf{a} - \mathbf{b}||_2$ (green line)
- The ℓ_1 -distance (Manhattan distance):
 $||\mathbf{a} - \mathbf{b}||_1$ (red, blue, yellow lines)

Transpose and Rank

Transpose:

$$\begin{bmatrix} 6 & 4 & 24 \\ 1 & -9 & 8 \end{bmatrix}^T = \begin{bmatrix} 6 & 1 \\ 4 & -9 \\ 24 & 8 \end{bmatrix}$$

Square matrix: a matrix with the same number of rows and columns.

Symmetric: a square matrix \mathbf{A} is symmetric if $\mathbf{A}^T = \mathbf{A}$.

Rank: the number of linearly independent rows (or columns).

Full rank: a square matrix is full rank if the rank equals to #columns.

Eigenvalue Decomposition

- Let \mathbf{A} be any $n \times n$ symmetric matrix.
- Eigenvalue decomposition: $\mathbf{A} = \sum_{i=1}^n \lambda_i \mathbf{v}_i \mathbf{v}_i^T$.
- Eigenvalues satisfy $|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n|$.
- Eigenvectors satisfy $\mathbf{v}_i^T \mathbf{v}_j = 0$ for all $i \neq j$.
- \mathbf{A} is full rank \iff all the eigenvalues are nonzero.

Warm-up: Optimization

Optimization: Basics

Optimization problem: $\min_{\mathbf{w}} f(\mathbf{w}); \quad \text{s. t. } \mathbf{w} \in \mathcal{C}.$

- $\mathbf{w} = [w_1, \dots, w_d]$: optimization variables
- $f : \mathbb{R}^d \mapsto \mathbb{R}$: objective function
- \mathcal{C} (a subset of \mathbb{R}^d) : feasible set

Optimization: Basics

Optimization problem: $\min_{\mathbf{w}} f(\mathbf{w}) ;$ s. t. $\mathbf{w} \in \mathcal{C} .$

- $\mathbf{w} = [w_1, \dots, w_d]$: optimization variables
- $f : \mathbb{R}^d \mapsto \mathbb{R}$: objective function
- \mathcal{C} (a subset of \mathbb{R}^d) : feasible set


Constraint

Optimization: Basics

Optimization problem: $\min_{\mathbf{w}} f(\mathbf{w}); \quad \text{s. t. } \mathbf{w} \in \mathcal{C}.$

Optimal solution: $\mathbf{w}^* = \operatorname{argmin}_{\mathbf{w} \in \mathcal{C}} f(\mathbf{w}).$

- $f(\mathbf{w}^*) \leq f(\mathbf{w})$ for all the vectors \mathbf{w} in the set \mathcal{C} .
- \mathbf{w}^* may not exist, e.g., \mathcal{C} is the empty set.
- If \mathbf{w}^* exists, it may not be unique.

Least Squares Regression

Linear Regression

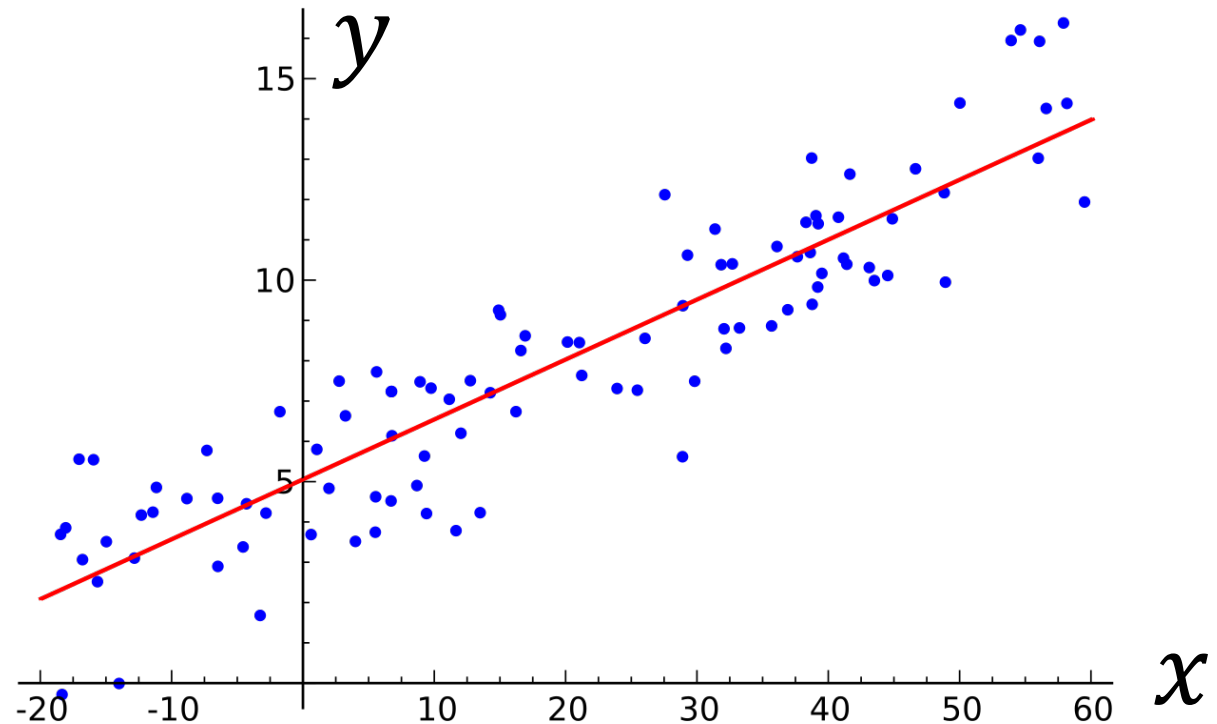
Input: vectors $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \dots, y_n \in \mathbb{R}$

Output: a vector $\mathbf{w} \in \mathbb{R}^d$ and scalar $b \in \mathbb{R}$ such that $\mathbf{x}_i^T \mathbf{w} + b \approx y_i$.

1-dim ($d = 1$) example:

Solution:

$$y_i \approx 0.15 x_i + 5.0$$

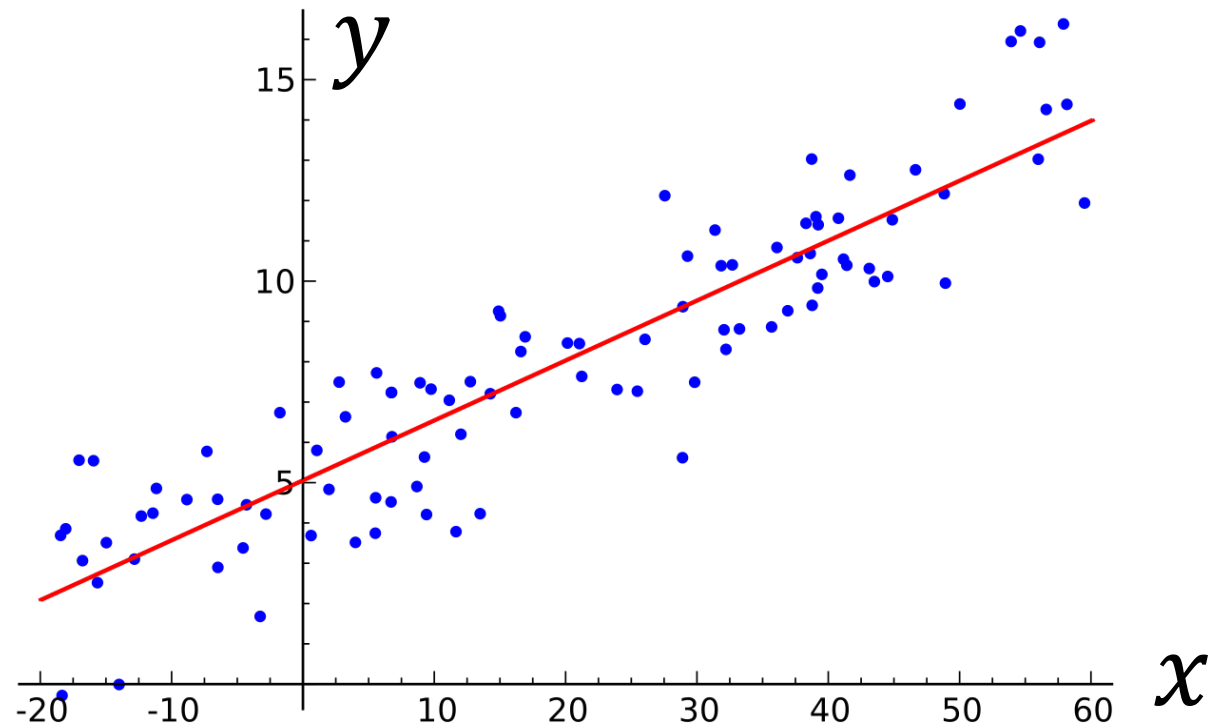


Linear Regression

Input: vectors $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \dots, y_n \in \mathbb{R}$

Output: a vector $\mathbf{w} \in \mathbb{R}^d$ and scalar $b \in \mathbb{R}$ such that $\mathbf{x}_i^T \mathbf{w} + b \approx y_i$.

Question (regard training):
how to compute \mathbf{w} and b ?



Least Squares Regression

Input: vectors $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ and labels $y_1, \dots, y_n \in \mathbb{R}$

Output: a vector $\mathbf{w} \in \mathbb{R}^d$ and scalar $b \in \mathbb{R}$ such that $\mathbf{x}_i^T \mathbf{w} + b \approx y_i$.

Method: least squares regression.

- The optimization model:

$$\min_{\mathbf{w}, b} \sum_{i=1}^n (\mathbf{x}_i^T \mathbf{w} + b - y_i)^2$$

Least Squares Regression

- The optimization model:

$$\min_{\mathbf{w}, b} \sum_{i=1}^n (\mathbf{x}_i^T \mathbf{w} + \boxed{b} - y_i)^2$$



Intercept (or bias)

Least Squares Regression

- The optimization model:

$$\min_{\mathbf{w}, b} \sum_{i=1}^n (\mathbf{x}_i^T \mathbf{w} + b - y_i)^2$$

$$\bar{\mathbf{x}}_i = \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix}$$

Least Squares Regression

- The optimization model:

$$\min_{\mathbf{w}, b} \sum_{i=1}^n (\mathbf{x}_i^T \mathbf{w} + b - y_i)^2$$

$$\bar{\mathbf{x}}_i = \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix}$$

$$\bar{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$$

Least Squares Regression

- The optimization model:

$$\min_{\mathbf{w}, b} \sum_{i=1}^n (\mathbf{x}_i^T \mathbf{w} + b - y_i)^2$$

$$\bar{\mathbf{x}}_i = \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix}$$

$$\bar{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$$

$$\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} = \mathbf{x}_i^T \mathbf{w} + b$$

Least Squares Regression

- The optimization model:

$$\min_{\mathbf{w}, b} \sum_{i=1}^n \underbrace{(\mathbf{x}_i^T \mathbf{w} + b - y_i)^2}_{= \bar{\mathbf{x}}_i^T \bar{\mathbf{w}}}$$

$$\bar{\mathbf{x}}_i = \begin{bmatrix} \mathbf{x}_i \\ 1 \end{bmatrix}$$

$$\bar{\mathbf{w}} = \begin{bmatrix} \mathbf{w} \\ b \end{bmatrix}$$

$$\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} = \mathbf{x}_i^T \mathbf{w} + b$$

Least Squares Regression

- The optimization model:

$$\min_{\mathbf{w}, b} \sum_{i=1}^n \left(\underbrace{\mathbf{x}_i^T \mathbf{w} + b}_{= \bar{\mathbf{x}}_i^T \bar{\mathbf{w}}} - y_i \right)^2$$



$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \sum_{i=1}^n \left(\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} - y_i \right)^2$$

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \sum_{i=1}^n (\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} - y_i)^2$$

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \sum_{i=1}^n (\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} - y_i)^2$$

$$\bar{\mathbf{X}} = \begin{bmatrix} \mathbf{x}_1^T & 1 \\ \mathbf{x}_2^T & 1 \\ \mathbf{x}_3^T & 1 \\ \vdots & \vdots \\ \mathbf{x}_n^T & 1 \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix}$$

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \sum_{i=1}^n (\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} - y_i)^2$$

$$\bar{\mathbf{X}} \bar{\mathbf{w}} = \begin{bmatrix} \bar{\mathbf{x}}_1^T \bar{\mathbf{w}} \\ \bar{\mathbf{x}}_2^T \bar{\mathbf{w}} \\ \bar{\mathbf{x}}_3^T \bar{\mathbf{w}} \\ \vdots \\ \bar{\mathbf{x}}_n^T \bar{\mathbf{w}} \end{bmatrix}$$

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \sum_{i=1}^n (\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} - y_i)^2$$

$$\bar{\mathbf{X}} \bar{\mathbf{w}} = \begin{bmatrix} \bar{\mathbf{x}}_1^T \bar{\mathbf{w}} \\ \bar{\mathbf{x}}_2^T \bar{\mathbf{w}} \\ \bar{\mathbf{x}}_3^T \bar{\mathbf{w}} \\ \vdots \\ \bar{\mathbf{x}}_n^T \bar{\mathbf{w}} \end{bmatrix}$$

$$\bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} = \begin{bmatrix} \bar{\mathbf{x}}_1^T \bar{\mathbf{w}} - y_1 \\ \bar{\mathbf{x}}_2^T \bar{\mathbf{w}} - y_2 \\ \bar{\mathbf{x}}_3^T \bar{\mathbf{w}} - y_3 \\ \vdots \\ \bar{\mathbf{x}}_n^T \bar{\mathbf{w}} - y_n \end{bmatrix}$$

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \sum_{i=1}^n (\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} - y_i)^2$$

$$\|\bar{\mathbf{X}}\bar{\mathbf{w}} - \mathbf{y}\|_2^2 = \left\| \begin{bmatrix} \bar{\mathbf{x}}_1^T \bar{\mathbf{w}} - y_1 \\ \bar{\mathbf{x}}_2^T \bar{\mathbf{w}} - y_2 \\ \bar{\mathbf{x}}_3^T \bar{\mathbf{w}} - y_3 \\ \vdots \\ \bar{\mathbf{x}}_n^T \bar{\mathbf{w}} - y_n \end{bmatrix} \right\|_2^2$$

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \sum_{i=1}^n (\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} - y_i)^2$$

$$\|\bar{\mathbf{X}}\bar{\mathbf{w}} - \mathbf{y}\|_2^2 = \left\| \begin{bmatrix} \bar{\mathbf{x}}_1^T \bar{\mathbf{w}} - y_1 \\ \bar{\mathbf{x}}_2^T \bar{\mathbf{w}} - y_2 \\ \bar{\mathbf{x}}_3^T \bar{\mathbf{w}} - y_3 \\ \vdots \\ \bar{\mathbf{x}}_n^T \bar{\mathbf{w}} - y_n \end{bmatrix} \right\|_2^2 = \sum_{i=1}^n (\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} - y_i)^2.$$

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \sum_{i=1}^n (\bar{\mathbf{x}}_i^T \bar{\mathbf{w}} - y_i)^2$$



Matrix form:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \|\bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y}\|_2^2$$

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Tasks

Linear
Regression

Methods

Least Squares Regression

LASSO

Least Absolute Deviations

Algorithms

?

Least Squares Regression

- The optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Tasks

Linear
Regression

Methods

Least Squares Regression

LASSO

Least Absolute Deviations

Algorithms

Analytical Solution

Gradient Descent (GD)

Conjugate Gradient (CG)

Least Squares Regression

- Solve the optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Gradient: $\frac{\partial \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2}{\partial \bar{\mathbf{w}}} = 2(\bar{\mathbf{X}}^T \bar{\mathbf{X}} \bar{\mathbf{w}} - \bar{\mathbf{X}}^T \mathbf{y})$

Algorithms

Analytical Solution

Gradient Descent (GD)

Conjugate Gradient (CG)

Least Squares Regression

- Solve the optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Gradient: $\frac{\partial \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2}{\partial \bar{\mathbf{w}}} = 2(\bar{\mathbf{X}}^T \bar{\mathbf{X}} \bar{\mathbf{w}} - \bar{\mathbf{X}}^T \mathbf{y}) = \mathbf{0}$



1st-order optimality condition

Algorithms

Analytical Solution

Gradient Descent (GD)

Conjugate Gradient (CG)

Least Squares Regression

- Solve the optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Gradient: $\frac{\partial \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2}{\partial \bar{\mathbf{w}}} = 2(\bar{\mathbf{X}}^T \bar{\mathbf{X}} \bar{\mathbf{w}} - \bar{\mathbf{X}}^T \mathbf{y}) = \mathbf{0}$



Normal equation: $\bar{\mathbf{X}}^T \bar{\mathbf{X}} \bar{\mathbf{w}}^* = \bar{\mathbf{X}}^T \mathbf{y}$

Assume $\bar{\mathbf{X}}^T \bar{\mathbf{X}}$ is full rank.



Analytical solution: $\bar{\mathbf{w}}^* = (\bar{\mathbf{X}}^T \bar{\mathbf{X}})^{-1} \bar{\mathbf{X}}^T \mathbf{y}$

Algorithms

Analytical Solution

Gradient Descent (GD)

Conjugate Gradient (CG)

Least Squares Regression

- Solve the optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Gradient: $\frac{\partial \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2}{\partial \bar{\mathbf{w}}} = 2(\bar{\mathbf{X}}^T \bar{\mathbf{X}} \bar{\mathbf{w}} - \bar{\mathbf{X}}^T \mathbf{y}) = \mathbf{0}$

Gradient descent repeats:

1. Compute gradient: $\mathbf{g}_t = \bar{\mathbf{X}}^T \bar{\mathbf{X}} \bar{\mathbf{w}}_t - \bar{\mathbf{X}}^T \mathbf{y}$
2. Update: $\bar{\mathbf{w}}_{t+1} = \bar{\mathbf{w}}_t - \alpha_t \mathbf{g}_t$

Algorithms

Analytical Solution

Gradient Descent (GD)

Conjugate Gradient (CG)

Least Squares Regression

- Solve the optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Convergence: after $O\left(\kappa \log \frac{1}{\epsilon}\right)$ iterations,

$$\left\| \bar{\mathbf{X}} (\bar{\mathbf{w}}_t - \bar{\mathbf{w}}^*) \right\|_2 \leq \epsilon \left\| \bar{\mathbf{X}} (\bar{\mathbf{w}}_0 - \bar{\mathbf{w}}^*) \right\|_2.$$

$$\kappa = \frac{\lambda_{\max}(\bar{\mathbf{X}}^T \bar{\mathbf{X}})}{\lambda_{\min}(\bar{\mathbf{X}}^T \bar{\mathbf{X}})} \text{ is the condition number.}$$

Algorithms

Analytical Solution

Gradient Descent (GD)

Conjugate Gradient (CG)

Least Squares Regression

- Solve the optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Convergence: after $O\left(\sqrt{\kappa} \log \frac{1}{\epsilon}\right)$ iterations,

$$\left\| \bar{\mathbf{X}} (\bar{\mathbf{w}}_t - \bar{\mathbf{w}}^*) \right\|_2 \leq \epsilon \left\| \bar{\mathbf{X}} (\bar{\mathbf{w}}_0 - \bar{\mathbf{w}}^*) \right\|_2.$$

$$\kappa = \frac{\lambda_{\max}(\bar{\mathbf{X}}^T \bar{\mathbf{X}})}{\lambda_{\min}(\bar{\mathbf{X}}^T \bar{\mathbf{X}})} \text{ is the condition number.}$$

The pseudo-code of CG is available at the [Wikipedia](#).

Algorithms

Analytical Solution

Gradient Descent (GD)

Conjugate Gradient (CG)

Least Squares Regression

- Solve the optimization model:

$$\min_{\bar{\mathbf{w}} \in \mathbb{R}^{d+1}} \left\| \bar{\mathbf{X}} \bar{\mathbf{w}} - \mathbf{y} \right\|_2^2$$

Tasks

Linear
Regression

Methods

Least Squares Regression

LASSO

Least Absolute Deviations

Algorithms

Analytical Solution

Gradient Descent (GD)

Conjugate Gradient (CG)

Solve Least Squares in Python

1. Load Data

```
from keras.datasets import boston_housing

(x_train, y_train), (x_test, y_test) = boston_housing.load_data()

print('shape of x_train: ' + str(x_train.shape))
print('shape of x_test: ' + str(x_test.shape))
print('shape of y_train: ' + str(y_train.shape))
print('shape of y_test: ' + str(y_test.shape))
```

```
shape of x_train: (404, 13)
shape of x_test: (102, 13)
shape of y_train: (404,)
shape of y_test: (102,)
```


2. Add A Feature

```
import numpy
```

```
n, d = x_train.shape
```

```
xbar_train = numpy.concatenate((x_train, numpy.ones((n, 1))),  
                                axis=1)
```

```
print('shape of x_train: ' + str(x_train.shape))
```

```
print('shape of xbar_train: ' + str(xbar_train.shape))
```

```
shape of x_train: (404, 13)
```

```
shape of xbar_train: (404, 14)
```

3. Solve the Least Squares

Analytical solution: $\bar{\mathbf{w}} = (\bar{\mathbf{X}}^T \bar{\mathbf{X}})^{-1} \bar{\mathbf{X}}^T \mathbf{y}$

```
xx = numpy.dot(xbar_train.T, xbar_train)
xx_inv = numpy.linalg.pinv(xx)
xy = numpy.dot(xbar_train.T, y_train)
w = numpy.dot(xx_inv, xy)
```

3. Solve the Least Squares

Analytical solution: $\bar{\mathbf{w}} = (\bar{\mathbf{X}}^T \bar{\mathbf{X}})^{-1} \bar{\mathbf{X}}^T \mathbf{y}$

```
xx = numpy.dot(xbar_train.T, xbar_train)
xx_inv = numpy.linalg.pinv(xx)
xy = numpy.dot(xbar_train.T, y_train)
w = numpy.dot(xx_inv, xy)
```

3. Solve the Least Squares

Analytical solution: $\bar{\mathbf{w}} = (\bar{\mathbf{X}}^T \bar{\mathbf{X}})^{-1} \bar{\mathbf{X}}^T \mathbf{y}$

```
xx = numpy.dot(xbar_train.T, xbar_train)
xx_inv = numpy.linalg.pinv(xx)
xy = numpy.dot(xbar_train.T, y_train)
w = numpy.dot(xx_inv, xy)
```

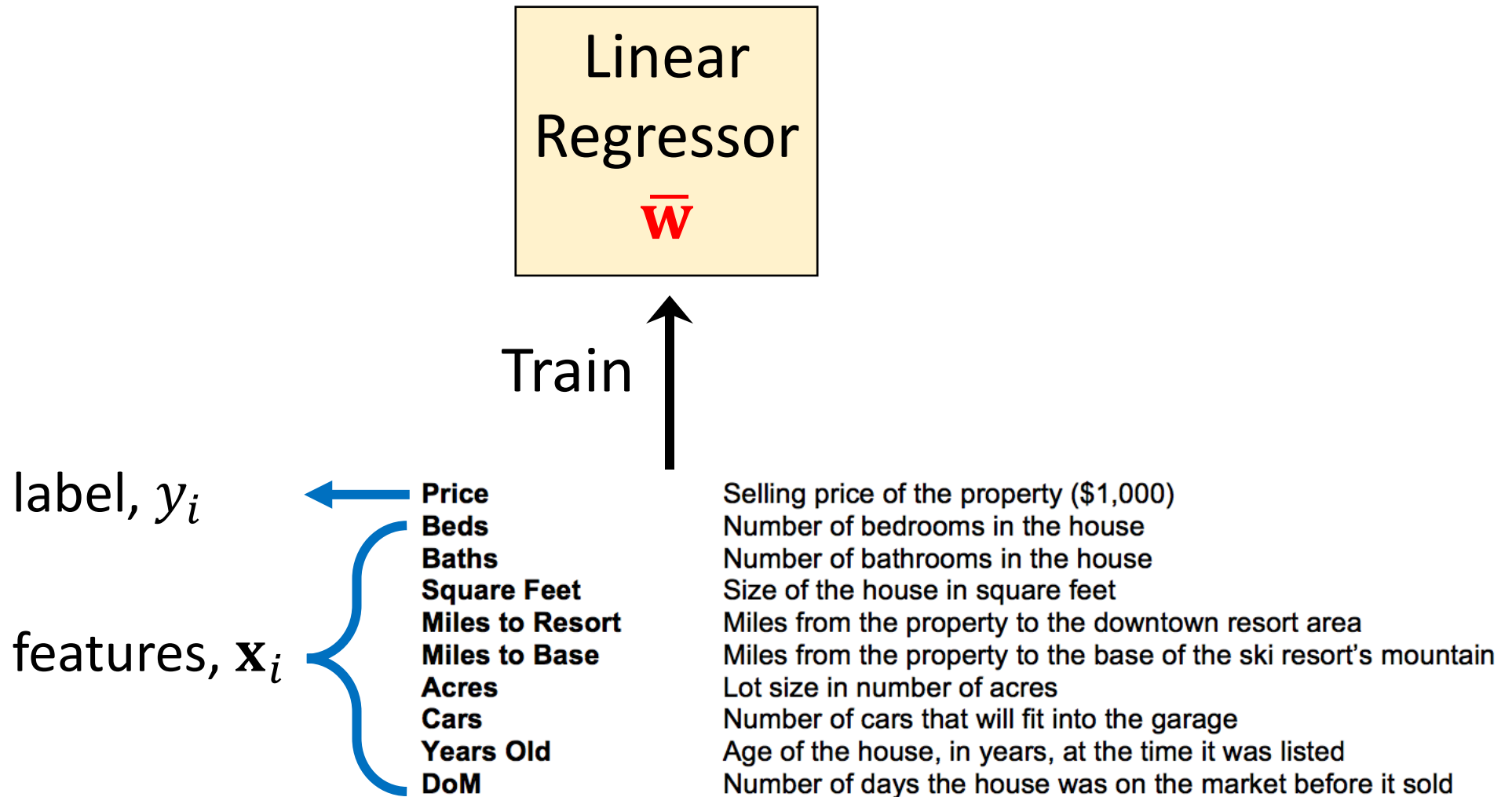
3. Solve the Least Squares

Training Mean Squared Error (MSE): $\frac{1}{n} \|\mathbf{y} - \bar{\mathbf{X}}\bar{\mathbf{w}}\|_2^2$

```
y_lsr = numpy.dot(xbar_train, w)
diff = y_lsr - y_train
mse = numpy.mean(diff * diff)
print('Train MSE: ' + str(mse))
```

Train MSE: 22.00480083834814

Linear Regression for Housing Price



Linear Regression for Housing Price



Features of a House, \mathbf{x}'
→ Extend it to $\bar{\mathbf{x}}'$



Linear
Regressor
 $\bar{\mathbf{w}}$

→
Predict

Price:
 $\bar{\mathbf{w}}^T \bar{\mathbf{x}}' = \500K

4. Make Prediction for Test Samples

- Add a feature to the test feature matrix: $\mathbf{X}_{\text{test}} \rightarrow \bar{\mathbf{X}}_{\text{test}}$.
- Make prediction by: $\mathbf{y}_{\text{pred}} = \bar{\mathbf{X}}_{\text{test}} \bar{\mathbf{w}}$.

```
n_test, _ = x_test.shape
xbar_test = numpy.concatenate((x_test, numpy.ones((n_test, 1))), axis=1)
y_pred = numpy.dot(xbar_test, w)
```


4. Make Prediction for Test Samples

- Add a feature to the test feature matrix: $\mathbf{X}_{\text{test}} \rightarrow \bar{\mathbf{X}}_{\text{test}}$.
- Make prediction by: $\mathbf{y}_{\text{pred}} = \bar{\mathbf{X}}_{\text{test}} \bar{\mathbf{w}}$.
- MSE (test): $\frac{1}{n_{\text{test}}} \left\| \mathbf{y}_{\text{pred}} - \mathbf{y}_{\text{test}} \right\|_2^2$

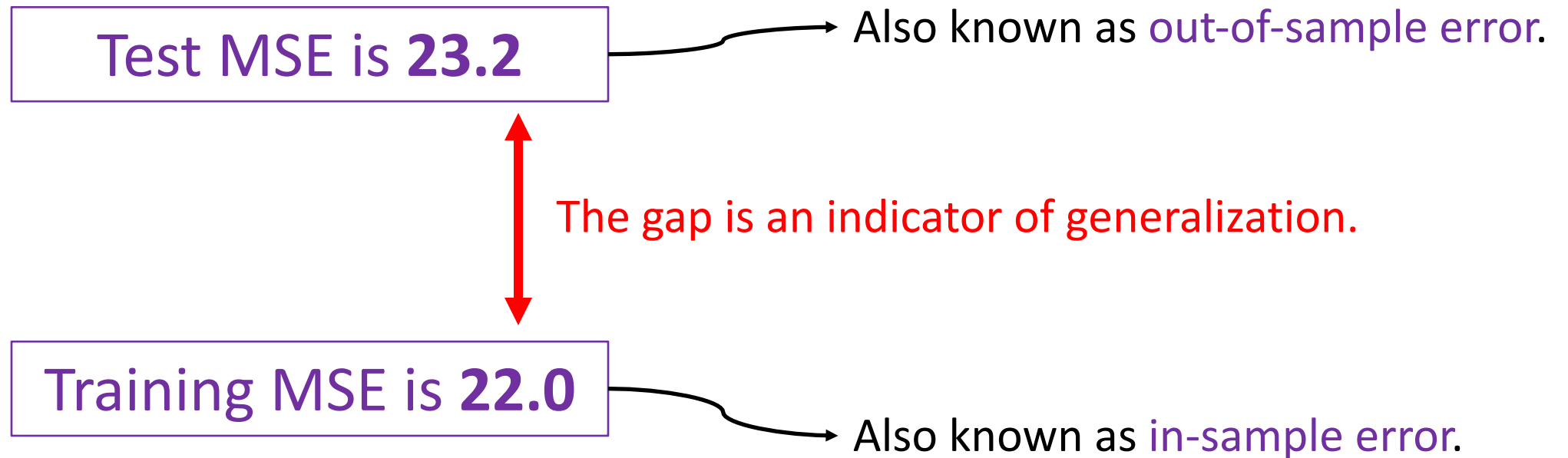
```
# mean squared error (testing)
```

```
diff = y_pred - y_test  
mse = numpy.mean(diff * diff)  
print('Test MSE: ' + str(mse))
```

Test MSE: 23.195599256409857

Training MSE is **22.0**

4. Make Prediction for Test Samples



5. Compare with Baseline

Trivial baseline:

- whatever the features are, the prediction is $\text{mean}(\mathbf{y})$.

```
y_mean = numpy.mean(y_train)

diff = y_pred - y_mean
mse = numpy.mean(diff * diff)
print('Test MSE: ' + str(mse))
```

Test MSE: 57.38297638530044

Test MSE of least
squares is **23.19**

Summary

- Linear regression problem.
- Least squares model.
- 3 algorithms for solving the model.
- Make predictions for never-seen-before test data.
- Evaluation of the model (training MSE and test MSE).
- Compare with baselines.