Stochastic View of Linear Regression

Renato Assunção
ESRI and DCC/UFMG

The initial view of the linear regression model

- We studied the linear regression model in two different views.
- In the first view, we look at the problem in an abstract way and reduce it to a mathematical optimization problem.
  - We saw that the optimal function $g(\boldsymbol{X})$ of the features $\boldsymbol{X}$ to minimize $MSE(g) = \mathbb{E}(Y - g(X))^2$ is $g(\boldsymbol{x}) = \boldsymbol{\mu}(\boldsymbol{x}) = \mathbb{E}(Y|\boldsymbol{X} = \boldsymbol{x})$. Using Taylor expansion, we approximate this multivariate and probably non-linear optimal function, assuming that

$$\boldsymbol{\mu}(\boldsymbol{x}) = \mathbb{E}(Y|\boldsymbol{X} = \boldsymbol{x}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \varepsilon \, .$$

  - Then, we approximate $\mathbb{E}(Y - g(X))^2$ by the average empirical (squared) error

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}))^2$$

  where $(y_i, \boldsymbol{x}_i) = (y_i, x_{i1}, \ldots, x_{ip})$ is the i-th instance of the random vector $(Y, \boldsymbol{X})$.

The second view of the linear regression model

- In the second view, we look at the problem as a linear algebra problem.
  - We see the n instantiations or realizations of the random vector $(Y, \boldsymbol{X})$ as matrices and vectors in $\mathbb{R}^n$.
  - We looked for the linear combination $\boldsymbol{X}\beta$ living in the vector subspace $\mathcal{C}(\boldsymbol{X})$ that best approximates the vector $\boldsymbol{Y} \in \mathbb{R}^n$.
  - We found that the solution is $\boldsymbol{X}\hat{\beta}$, the orthogonal projection of $\boldsymbol{Y}$ into $\mathcal{C}(\boldsymbol{X})$.
- In both views, the solution was the same:

$$\hat{\beta} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{Y} \, .$$

- However, there is a third view that can take us much further to understand the properties of this solution
- This is obtained by introducing a stochastic (probabilistic) view of the data.

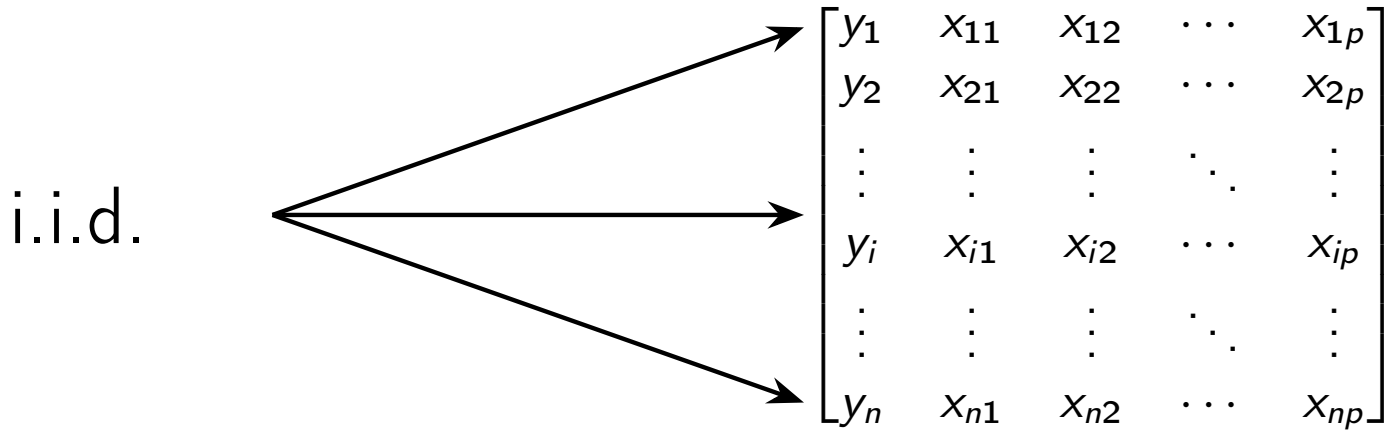
The third view is a stochastic model

- We assume a probabilistic mechanism generating the random data $(Y, \boldsymbol{X})$ with probability density $f(y, \boldsymbol{x}) = f(y, x_1, \ldots, x_p)$.
- We are allowed to observe a small sample of these data, the empirical sample with n training examples.
- The observed data are $(y_1, \boldsymbol{x}_1), (y_2, \boldsymbol{x}_2), \ldots, (y_n, \boldsymbol{x}_n)$.
- The feature vector is composed of p variables: $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})$.
- Organize the data into a matrix

$$\begin{bmatrix} y_1 & x_{11} & x_{12} & \cdots & x_{1p} \\ y_2 & x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_i & x_{i1} & x_{i2} & \cdots & x_{ip} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ y_n & x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}$$

i.i.d. rows

- The random rows of this matrix are considered instantiations (or realizations) independent and identically distributed (i.i.d.) from a certain joint distribution with density $f(y, \boldsymbol{x}) = f(y, x_1, \ldots, x_p)$.



- For example, consider the apartment price prediction problem.
- The i.i.d. assumption comes from assuming that the apartments are randomly selected as if all possible apartments (present and even future apartments, yet to be built under current conditions) were placed into a bag and, after being thoroughly shuffled or mixed, we randomly selected n of them.

i.i.d. rows

- This assumption may be violated if, instead of sampling apartments from across the entire city, we randomly select a single building and construct a sample of $n = 20$ apartments.
- In this setting, the assumption of independence between rows is no longer appropriate when the target population is the full set of apartments in the city. Apartments within the same building tend to share unobserved characteristics (e.g., location, construction quality, amenities), inducing dependence.
- As a consequence, the sample exhibits low variability: the apartments are very similar to one another. Knowing the prices and features of 19 units allows us to predict the remaining one with high accuracy.
- However, this apparent predictive performance is misleading. The prediction error estimated from this sample will be overly optimistic and much smaller than the error we would obtain when predicting the price of an apartment randomly drawn from the entire city.

Columns are not independent

- The rows can be viewed as independent vectors, but the random columns should not be assumed to be independent random vectors.
- Clearly, knowing the vector $(x_{11}, \ldots, x_{n1})$ of apartment areas helps substantially in predicting the vector $(y_1, \ldots, y_n)$ of prices.
- Moreover, the column of apartment areas is strongly associated with, and is a good predictor of, the column corresponding to the number of rooms.
- For example, a $50\,\mathrm{m}^2$ apartment is extremely unlikely to have 3 or 4 rooms, whereas a $300\,\mathrm{m}^2$ apartment having only one room would be highly surprising.
- Indeed, it is precisely because the columns are dependent that we use the feature columns $\boldsymbol{x}$ to predict the response vector y.
- The random columns are not identically distributed either. For example, the continuous probability distribution generating apartment prices is fundamentally different from the discrete probability distribution governing the (typically small integer) number of rooms.

Generative models

- In ML we distinguish between two types of models: generative and discriminative.
- Generative models aim to model the joint distribution $f(y, \boldsymbol{x})$
- Examples of generative models include:
  - Bayesian networks
  - Markov random fields
  - Naïive Bayes classifier
  - Gaussian mixture models
  - Hidden Markov models
  - Linear discriminant analysis
- This is in contrast to discriminative models, which focus on modeling $f(y \mid \boldsymbol{x})$.

Discriminative models

- The joint density can always be decomposed as:

$$\underbrace{f(y, \boldsymbol{x})}_{\text{joint}} = \underbrace{f(y|\boldsymbol{x})}_{\text{conditional}} \underbrace{f(\boldsymbol{x})}_{\text{marginal}}$$

- Discriminative models focus on the conditional distribution $f(y \mid \boldsymbol{x})$, ignoring the marginal distribution $f(\boldsymbol{x})$.
- Rather than modeling the joint behavior of many variables, we focus on predicting a single random variable Y given the features $\boldsymbol{x}$.
- Many ML models are discriminative:
  - Linear and Logistic regression
  - Supervised neural nets for prediction and classification
  - Classification and regression trees, Random forests
  - Support vector machines (SVM), Boosting (e.g., XGBoost)

Discriminative models

- In fact, the notation $f(y \mid \boldsymbol{x})$ hides a very large family of distributions: there is one conditional distribution for each specific value of $\boldsymbol{x}$.
- For example:

$$f(y^{\text{price}} | x_1^{\text{area}} = 100m^2, x_2^{\text{rooms}} = 2, x_3^{\text{restroom}} = 1, \ldots)$$

and

$$f(y^{\text{price}} | x_1^{\text{area}} = 150m^2, x_2^{\text{rooms}} = 2, x_3^{\text{restroom}} = 2, \ldots)$$

are two different probability distributions.

- Therefore, if $\boldsymbol{x}_i \neq \boldsymbol{x}_j$, the conditional distributions $(Y_i \mid \boldsymbol{x}_i)$ and $(Y_j \mid \boldsymbol{x}_j)$ are not the same.
- They are still independent, but not identically distributed.

Discriminative models

- Why some apartments have high prices while others are relatively inexpensive?
- We explain this variability by decomposing its causes into two components:
  - Causes determined by the observed attributes (features);
  - Other causes that are unmeasured or unknown.
- In addition, we decompose the random variable $(Y \mid \boldsymbol{x})$ as the sum of its conditional expectation and a deviation from that expectation.

Decomposition of $(Y \mid \boldsymbol{x})$

- Given that $\boldsymbol{X} = \boldsymbol{x}$, define

$$\varepsilon = Y - \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x}) = Y - \mu(\boldsymbol{x})$$

- Hence, we can always write

$$\underbrace{(Y \mid \boldsymbol{X} = \boldsymbol{x})}_{\text{random}} = \underbrace{\mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x})}_{\text{deterministic given } \boldsymbol{x}} + \underbrace{\varepsilon}_{\text{random}} = \mu(\boldsymbol{x}) + \varepsilon.$$

- The function $\mu(\boldsymbol{x}) = \mathbb{E}(Y \mid \boldsymbol{X} = \boldsymbol{x})$ is the optimal predictor of Y under squared error loss.

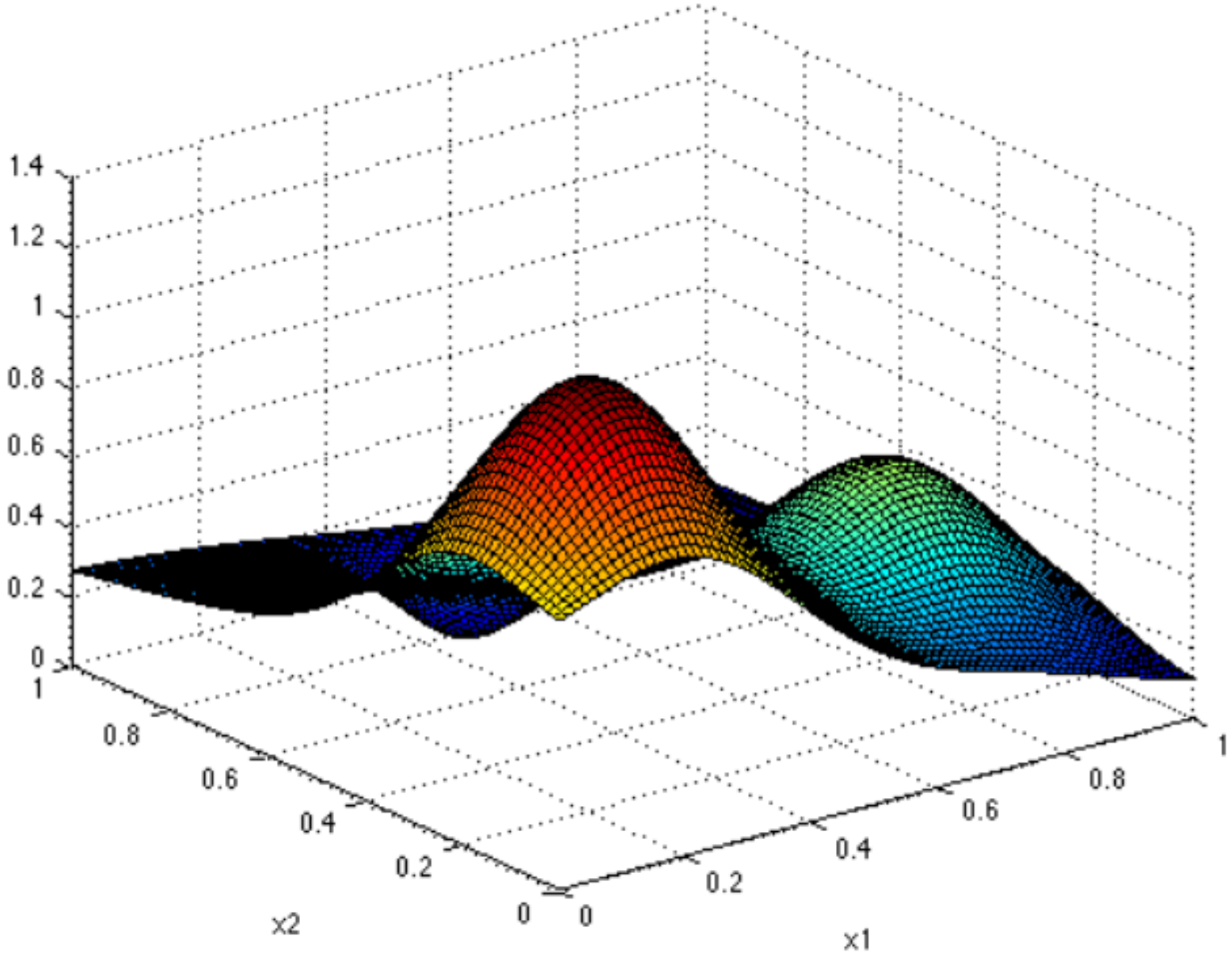


Figure: Surface $\mu(\boldsymbol{x}) = \mu(x_1, x_2)$

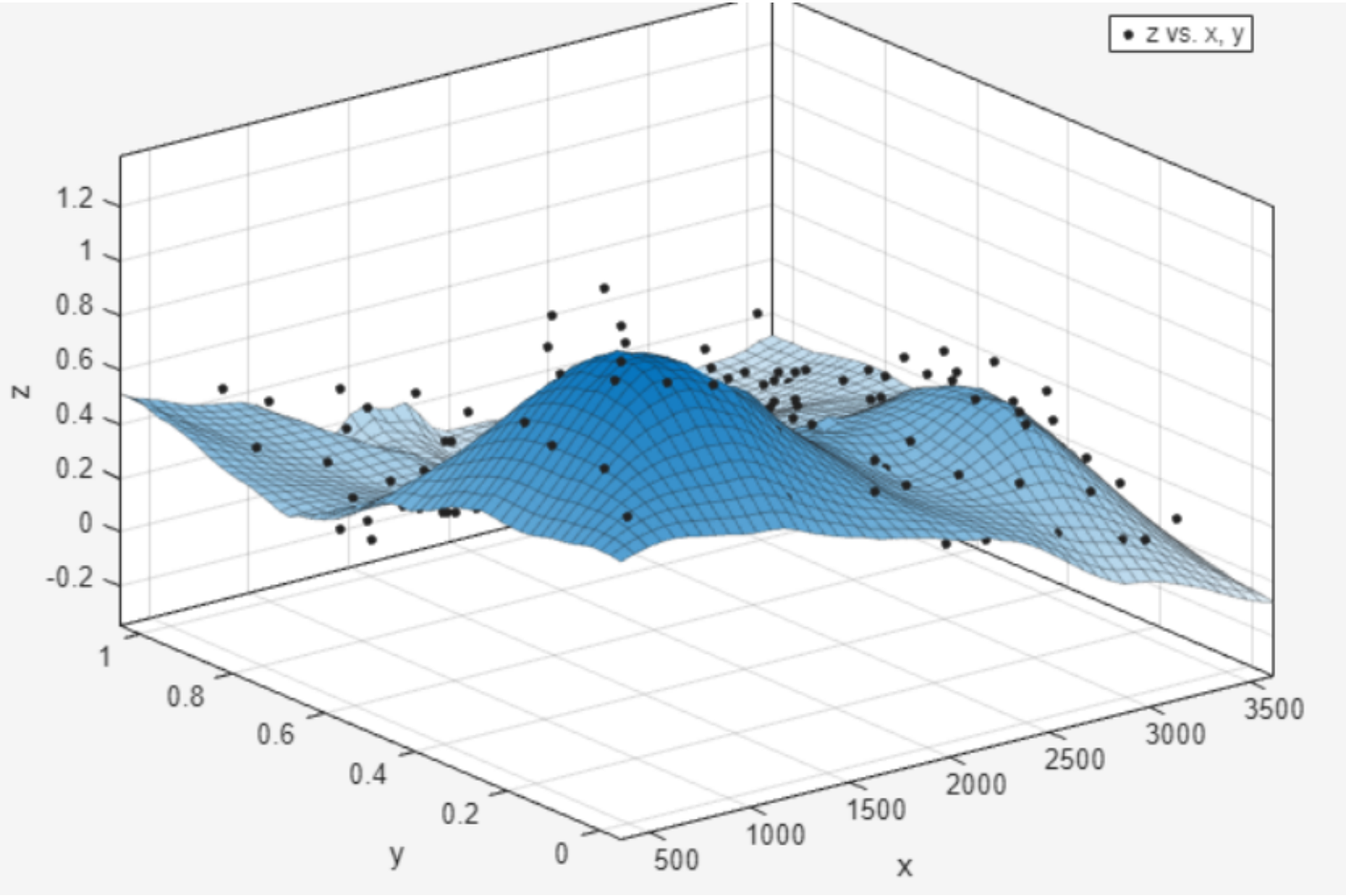


Figure: The dots represent observations $Y = \mu(\boldsymbol{x}) + \varepsilon$, scattered around the regression surface $\mu(\boldsymbol{x}) = \mu(x_1, x_2)$.

We assume that $\mu(\boldsymbol{x})$ is linear

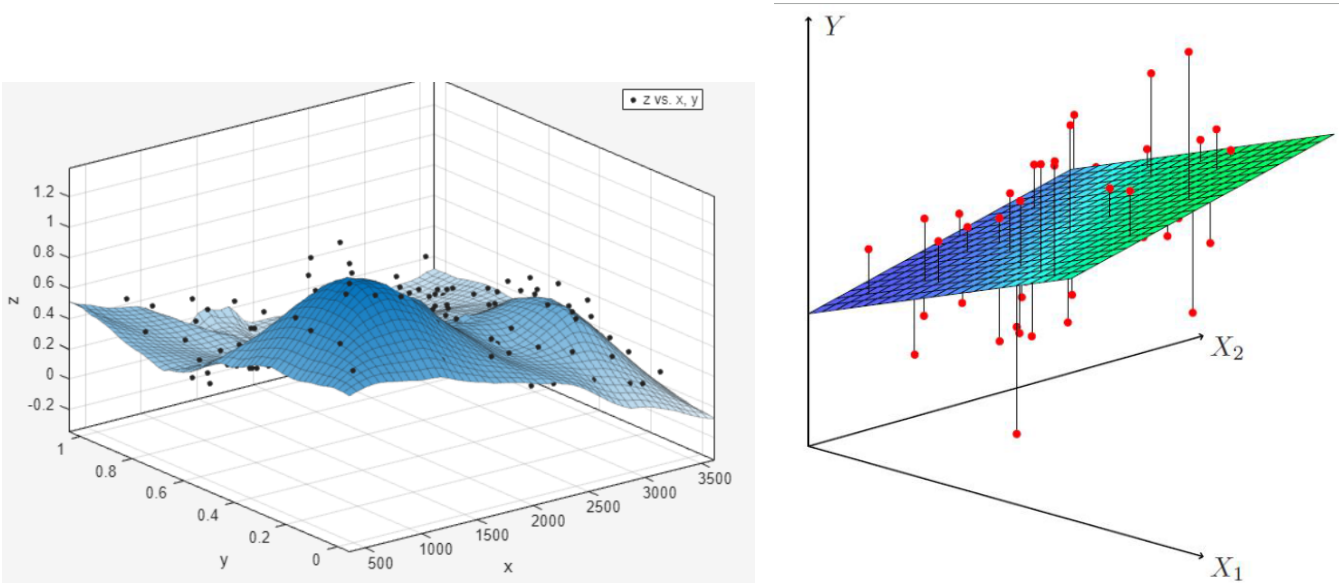


Figure: We assume that $\mu(\boldsymbol{x})$ is a linear function of the features $\mu(\boldsymbol{x}) = \mu(x_1, x_2) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$.

The stochastic model

- We have

$$
\begin{aligned}
(Y \mid \boldsymbol{X} = \boldsymbol{x}) &= \mu(\boldsymbol{x}) + \varepsilon \\
&= \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p + \varepsilon \\
&= [1, x_1, x_2, \ldots, x_p] \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix} + \varepsilon \\
&= \boldsymbol{x}'\boldsymbol{\beta} + \varepsilon
\end{aligned}
$$

- Consider now the stochastic error ε.

The error ε

- We can always assume that $\mathbb{E}(\varepsilon) = 0$. To see this, suppose that $\mathbb{E}(\varepsilon) = \alpha \neq 0$. Then,

$$\begin{aligned} Y &= \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon \\ &= \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon - \alpha + \alpha \\ &= (\beta_0 + \alpha) + \beta_1 x_1 + \cdots + \beta_p x_p + (\varepsilon - \alpha) \\ &= \beta_0^* + \beta_1 x_1 + \cdots + \beta_p x_p + \varepsilon^*. \end{aligned}$$

- The new error satisfies

$$\mathbb{E}(\varepsilon^*) = \mathbb{E}(\varepsilon - \alpha) = \mathbb{E}(\varepsilon) - \alpha = \alpha - \alpha = 0.$$

- Thus, the same variable Y can be represented either with an error ε such that $\mathbb{E}(\varepsilon) = \alpha$ and intercept β_0, or with a shifted intercept β_0^* and an error ε^* such that $\mathbb{E}(\varepsilon^*) = 0$.
- Therefore, without loss of generality, we can always assume $\mathbb{E}(\varepsilon) = 0$.

How about the variance $\mathbb{V}(\varepsilon)$

- We assume that $\mathbb{V}(\varepsilon) = \sigma^2$, constant across values of $\boldsymbol{x}$.
- Thus, the typical deviation from the mean $\mu(\boldsymbol{x})$ is the same for all values of $\boldsymbol{x}$.
- This assumption is called **homoscedasticity**.

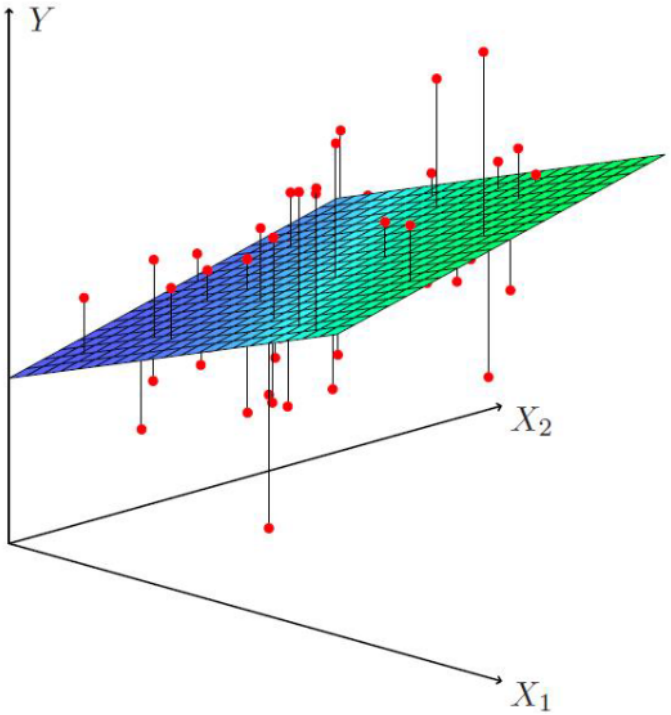


Figure: Homoscedastic errors: constant standard deviation around the surface $\mu(\boldsymbol{x})$.

Homoscedastic versus Heteroscesdastic errors

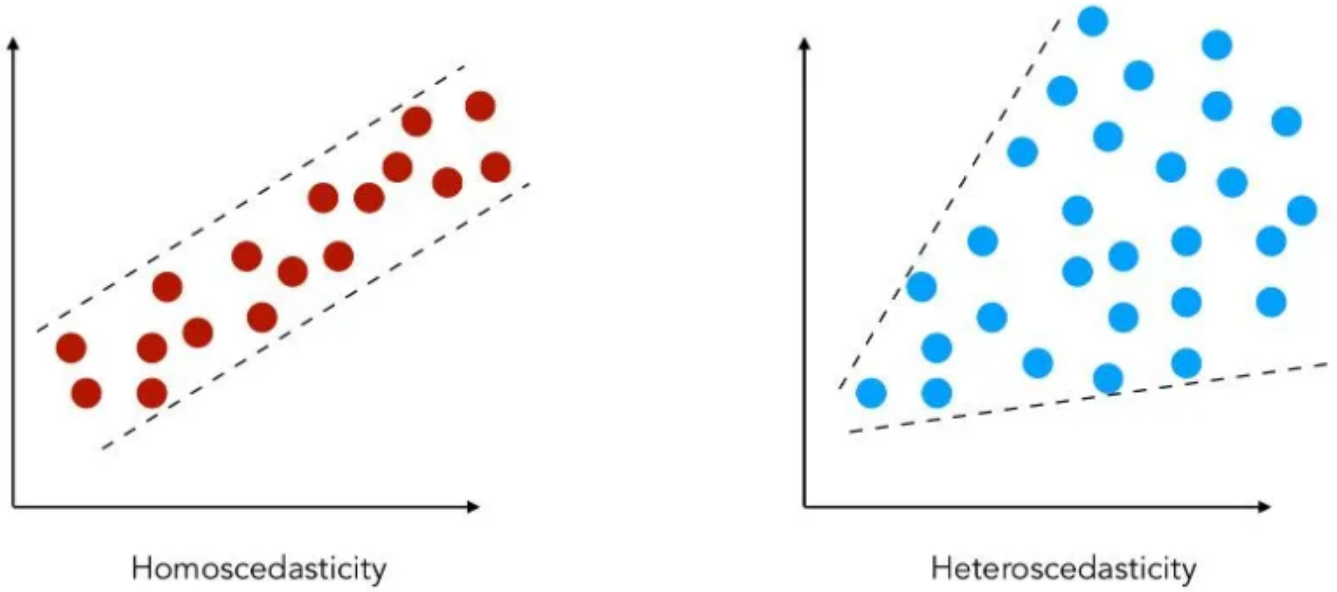


Figure: Homoscedastic versus Heteroscesdastic errors with data following a simple linear regression model $y = \beta_0 + \beta_1 x + \varepsilon$.

Heteroscesdastic errors with a univariate x

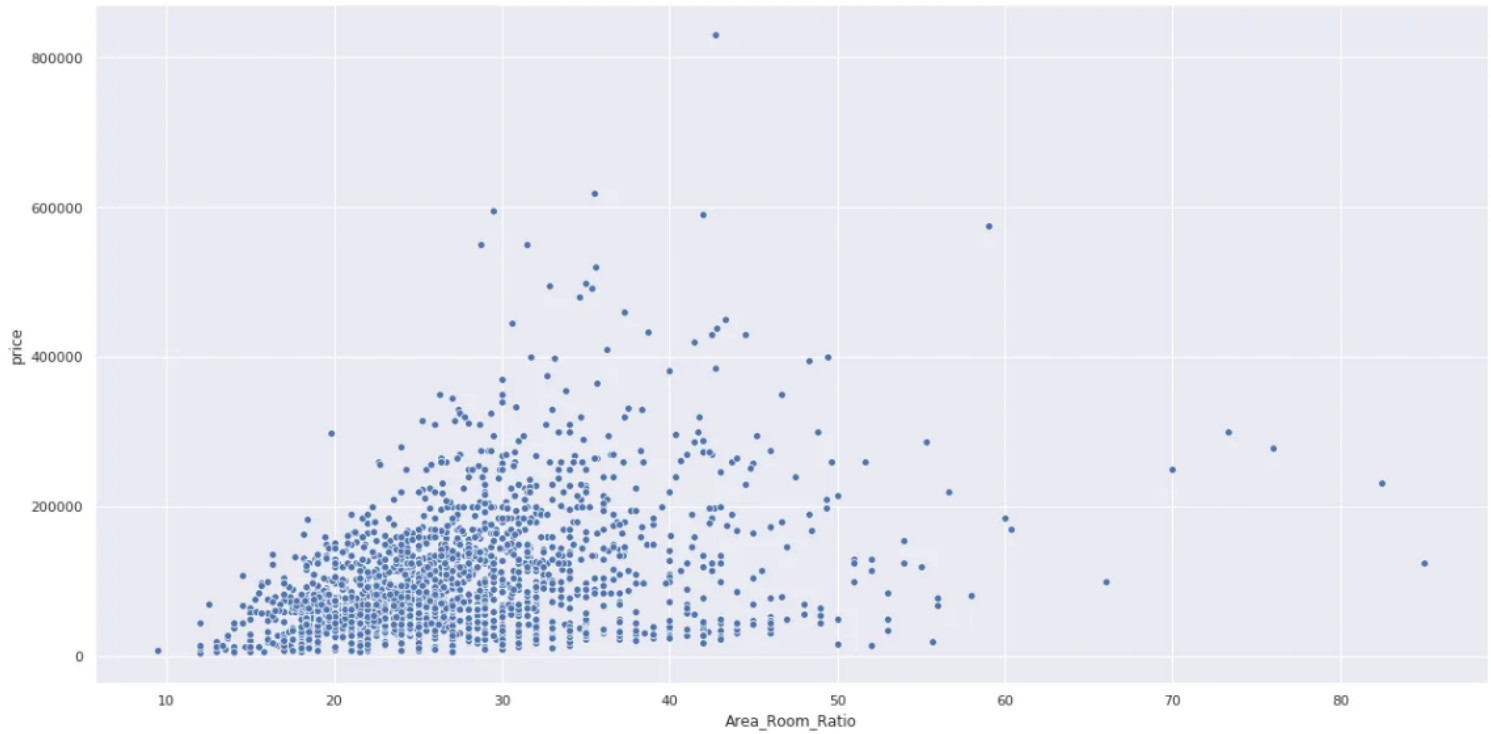


Figure: Illustration of heteroscesdastic errors with real data following a simple linear regression model $y = \beta_0 + \beta_1 x + \varepsilon$.

Heteroscesdastic errors: another example

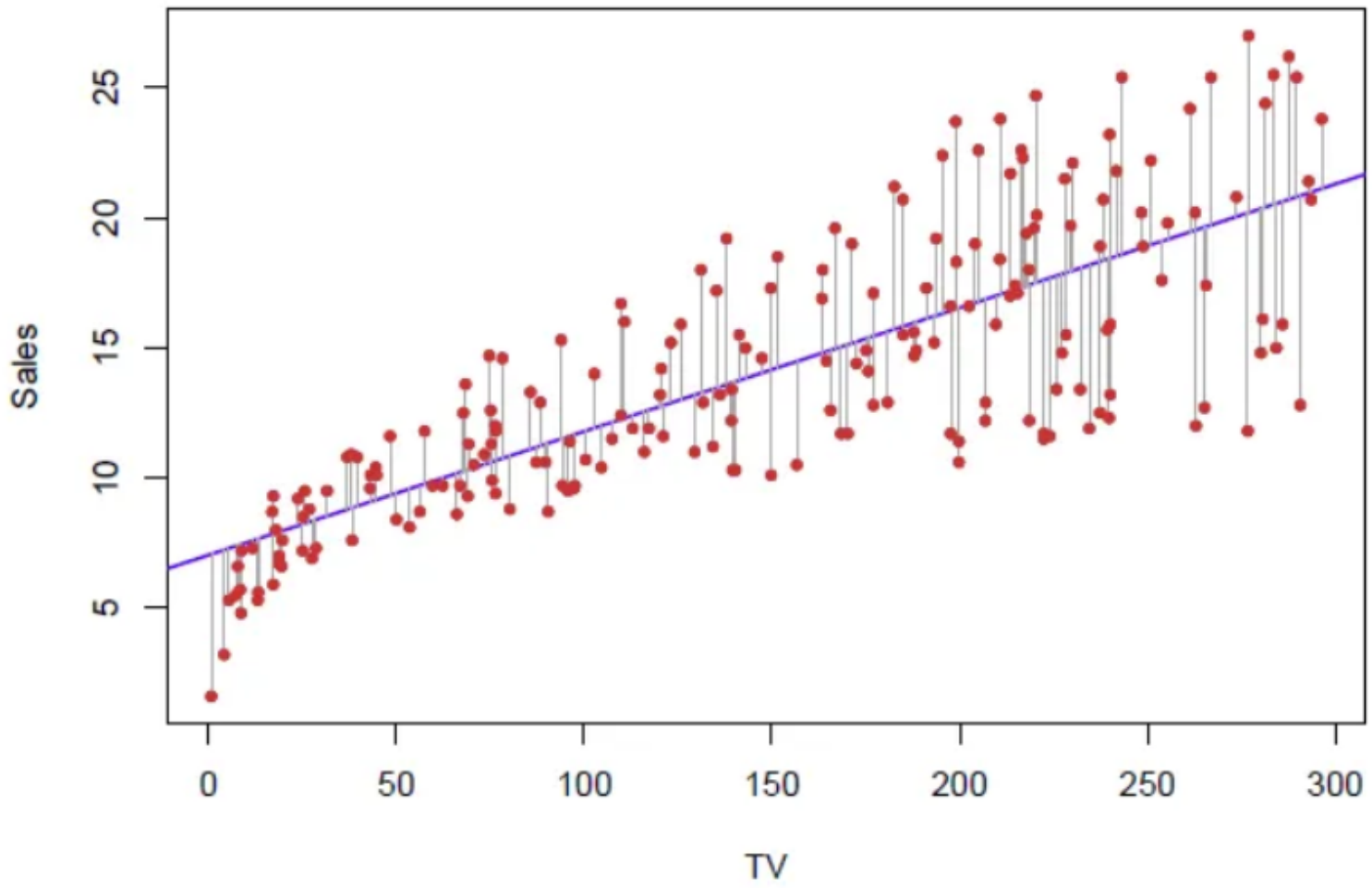


Figure: Another real data illustration of heteroscesdastic errors.

Probability distribution of ε_i

- For the i-th observation: $Y_i = \boldsymbol{x}_i'\boldsymbol{\beta} + \varepsilon_i$
- We set $\mathbb{E}(\varepsilon_i) = 0$ and $\mathbb{V}(\varepsilon_i) = \sigma^2$.
- Additionally, we assume that the probability distribution of these errors is Gaussian and that they are independent.
- Looking at the entire vector of random errors:

$$\begin{bmatrix} \varepsilon_0 \\ \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{bmatrix} \sim \mathcal{N}_n \left(\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \sigma^2 \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \right)$$

- That is,

$$\boldsymbol{\varepsilon} \sim \mathcal{N}_n(\mathbf{0}, \sigma^2 \ \boldsymbol{I}_n)$$

Probability distribution of ε_i

- This implies that $\varepsilon_1, \ldots \varepsilon_n$ are i.i.d. with distribution $\mathcal{N}(0, \sigma^2)$.
- Therefore,

$$(Y_i \mid \boldsymbol{X}_i = \boldsymbol{x}_i) = \boldsymbol{x}_i'\boldsymbol{\beta} + \underbrace{\varepsilon_i}_{\mathcal{N}(0,\sigma^2)}$$

- Adding a constant to a Gaussian random variable alters only its expectation and hence, conditioned on the features $\boldsymbol{X}$, the observations are independent and

$$(Y_i|\boldsymbol{X}_i = \boldsymbol{x}_i) \sim \mathcal{N}(\boldsymbol{x}_i'\boldsymbol{\beta}, \sigma^2)$$

The probability distribution $\mathcal{N}(\boldsymbol{x}_i'\boldsymbol{\beta}, \sigma^2)$

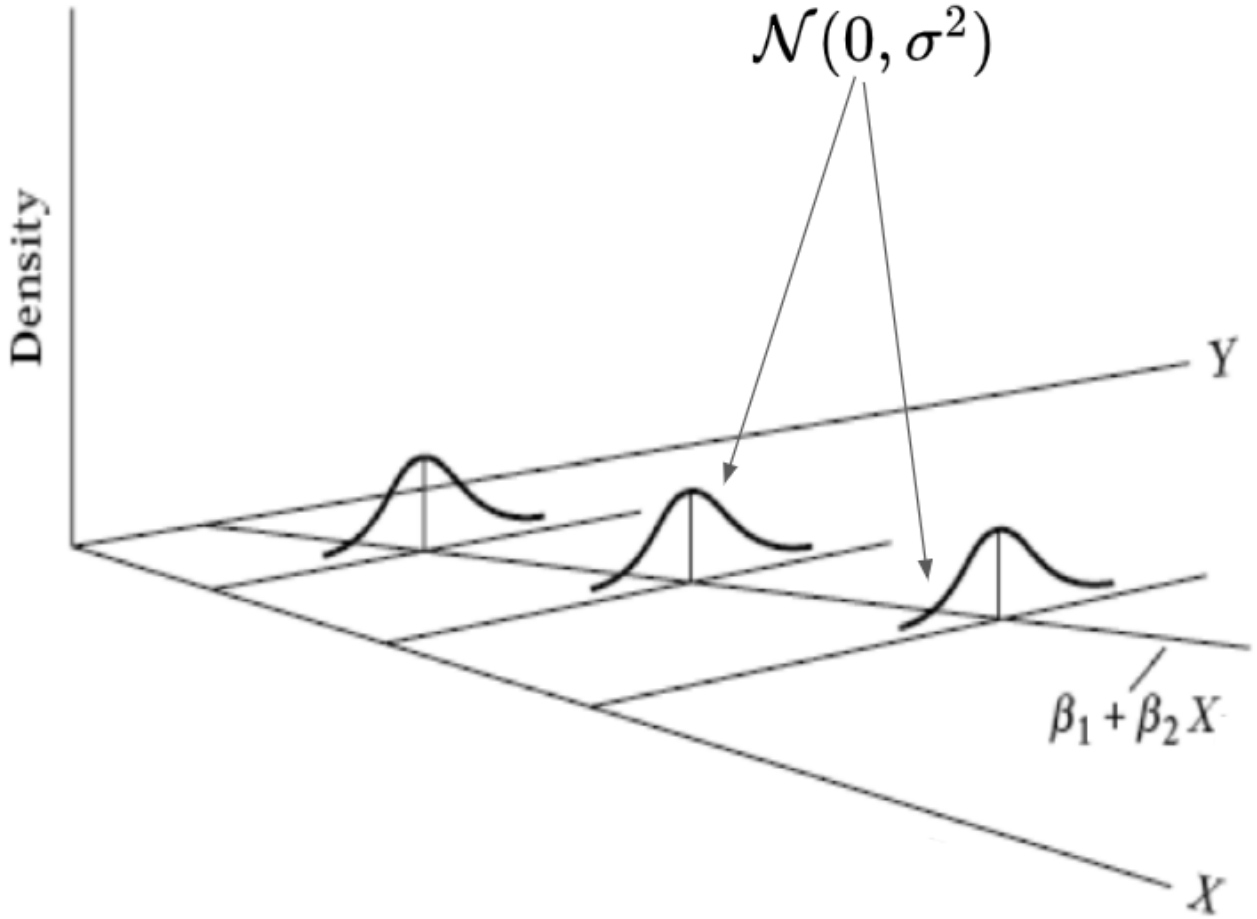


Figure: A simple linear regression model $y = \beta_0 + \beta_1 x + \varepsilon$.

Stochastic matrix notation

- For n observations:

$$\underbrace{\begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ \vdots \\ Y_n \end{pmatrix}}_{\boldsymbol{Y},\ n\times 1} = \underbrace{\begin{pmatrix} 1 & X_{11} & \dots & \dots & X_{1p} \\ 1 & X_{21} & \dots & \dots & X_{2p} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 1 & X_{n1} & \dots & \dots & X_{np} \end{pmatrix}}_{\boldsymbol{X},\ n\times(p+1)} \underbrace{\begin{pmatrix} \beta_0 \\ \vdots \\ \beta_p \end{pmatrix}}_{\boldsymbol{\beta},\ (p+1)\times 1} + \underbrace{\begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \vdots \\ \epsilon_n \end{pmatrix}}_{\boldsymbol{\epsilon},\ n\times 1}$$

- The deterministic expectation:

$$\boldsymbol{\mu}(\boldsymbol{X}) = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \vdots \\ \mu_n \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_1'\boldsymbol{\beta} \\ \boldsymbol{x}_2'\boldsymbol{\beta} \\ \vdots \\ \vdots \\ \boldsymbol{x}_n'\boldsymbol{\beta} \end{pmatrix} = \begin{pmatrix} 1 & X_{11} & \dots & \dots & X_{1p} \\ 1 & X_{21} & \dots & \dots & X_{2p} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 1 & X_{n1} & \dots & \dots & X_{np} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_p \end{pmatrix}$$

Stochastic matrix notation

- Linear regression model:

$$(\boldsymbol{Y} \mid \boldsymbol{X}) \sim \mathcal{N}_n(\boldsymbol{\mu}(\boldsymbol{X}), \sigma^2 \boldsymbol{I}_n)$$

- Equivalently,

$$(\boldsymbol{Y} \mid \boldsymbol{X}) \sim \mathcal{N}_n \left(\begin{pmatrix} 1 & X_{11} & \dots & \dots & X_{1p} \\ 1 & X_{21} & \dots & \dots & X_{2p} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 1 & X_{n1} & \dots & \dots & X_{np} \end{pmatrix} \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_p \end{pmatrix}, \sigma^2 \begin{pmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 0 & 1 \end{pmatrix} \right)$$

Gaussian distribution: a property

Let $\boldsymbol{Y}$ be a Gaussian random vector:

$$\underbrace{\boldsymbol{Y}}_{n\times 1} \sim \mathcal{N}_n(\underbrace{\boldsymbol{\mu}}_{n\times 1}, \underbrace{\boldsymbol{\Sigma}}_{n\times n})$$

and $\boldsymbol{A}$ be a $k \times n$ matrix of real constants. Then, the random vector $\boldsymbol{AY}$ has distribution

$$\underbrace{\boldsymbol{AY}}_{k\times 1} \sim \mathcal{N}_k(\underbrace{\boldsymbol{A\mu}}_{k\times 1}, \underbrace{\boldsymbol{A\Sigma A'}}_{k\times k})$$

Distribution of the LS Estimator

- We already learned that the least squares estimator of the unknown β in the linear regression model is

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{Y}$$

  where the constant matrix $\boldsymbol{A}$ is given by

$$\underbrace{\boldsymbol{A}}_{(p+1)\times n} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'$$

- As $\boldsymbol{Y} \sim \mathcal{N}_n(\boldsymbol{X}\beta, \sigma^2\boldsymbol{I}_n)$ is a random vector and $\hat{\boldsymbol{\beta}} = \boldsymbol{A}\boldsymbol{Y}$, then $\hat{\boldsymbol{\beta}}$ is also a random vector.
- The previous property gives us the distribution of this random vector

$$\hat{\boldsymbol{\beta}} \sim \mathcal{N}_{p+1}(\boldsymbol{A}\boldsymbol{X}\beta, \sigma^2\boldsymbol{A}\ \boldsymbol{I}_n\ \boldsymbol{A}')\,.$$

Distribution of the LS Estimator

Fundamental result

$$\hat{\boldsymbol{\beta}} \sim \mathcal{N}_{p+1}(\boldsymbol{AX}\boldsymbol{\beta}, \sigma^2 \boldsymbol{AA}')$$

- For the expected value, we have

$$\boldsymbol{AX}\boldsymbol{\beta} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{\beta}$$

- For the covariance matrix:

$$\begin{aligned}
\sigma^2 \boldsymbol{AA}' &= \sigma^2 (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}' \left((\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\right)' \\
&= \sigma^2 (\boldsymbol{X}'\boldsymbol{X})^{-1} \left(\boldsymbol{X}'\boldsymbol{X}\right) \left(\boldsymbol{X}'\boldsymbol{X}\right)^{-1})' \\
&= \sigma^2 ((\boldsymbol{X}'\boldsymbol{X})^{-1})' \\
&= \sigma^2 ((\boldsymbol{X}'\boldsymbol{X})')^{-1} \\
&= \sigma^2 (\boldsymbol{X}'\boldsymbol{X})^{-1}
\end{aligned}$$

Distribution of the LS Estimator

Fundamental result

$$\hat{\boldsymbol{\beta}} \sim \mathcal{N}_{p+1}(\boldsymbol{\beta}, \sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1})$$

- We have $\mathbb{E}(\hat{\boldsymbol{\beta}}) = \boldsymbol{\beta}$.
- Thus, the estimator $\hat{\boldsymbol{\beta}}$ varies randomly across samples but is centered at the true (unknown) parameter $\boldsymbol{\beta}$.
- Each component of $\hat{\boldsymbol{\beta}}$ is a random variable centered at the corresponding component of $\boldsymbol{\beta}$.
- Therefore, each component of $\hat{\boldsymbol{\beta}}$ does not systematically overestimate or underestimate the corresponding component of $\boldsymbol{\beta}$.
- We say that $\hat{\boldsymbol{\beta}}$ is an **unbiased estimator** of $\boldsymbol{\beta}$.
- We will discuss the covariance matrix after an example.

A very simple example

- The objective of this very simple example is to illustrate the random nature of the LS estimator.
- We will simulate 8 data points that follows the linear regression model below:

$$\begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \\ Y_6 \\ Y_7 \\ Y_8 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 2 \\ 1 & 2 \\ 1 & 3 \\ 1 & 3 \\ 1 & 4 \\ 1 & 4 \end{pmatrix} \begin{pmatrix} 0.1 \\ 0.7 \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \varepsilon_5 \\ \varepsilon_6 \\ \varepsilon_7 \\ \varepsilon_8 \end{pmatrix}$$

  where $\varepsilon \sim \mathcal{N}_8(\mathbf{0}, \sigma^2 I_8)$ with $\sigma = 0.3$.
- That is, the 8×2 design matrix $\boldsymbol{X}$ is fixed, the coefficient vector is $(\beta_0, \beta_1)' = (0.1, 0.7)$ and the random errors are i.i.d. $\mathcal{N}(0, \sigma^2)$ with $\sigma = 0.3$.

Simulation

```
# Install statsmodels (if not already installed)
!pip install -q statsmodels

# Import required libraries
import numpy as np
import pandas as pd
import statsmodels.api as sm
import matplotlib.pyplot as plt

# Set seed for reproducibility
np.random.seed(432)
# Create predictor variable x
x = np.repeat([1, 2, 3, 4], 2)
# Generate noise ~ N(0, 0.3^2)
noise = np.random.normal(0, 0.3, size=8)
# Compute response y = 0.1 + 0.7 * x + noise
y = 0.1 + 0.7 * x + noise
# Create DataFrame
df = pd.DataFrame({'x': x, 'y': y})
# Fit linear regression model
X = sm.add_constant(df['x'])  # Add intercept
model = sm.OLS(df['y'], X).fit()
# Print model summary
print(model.summary())

# Optional: Plot the data and fitted line
plt.scatter(df['x'], df['y'], color='blue', label='Data')
plt.plot(df['x'], model.predict(X), color='red', label='Fitted line')
mu_true = 0.1 + 0.7 * x
plt.plot(x, mu_true, color='darkblue', linewidth=3, label='True regression line')
plt.xlabel('x'); plt.ylabel('y')
plt.title('Linear Regression with 8 Observations')
plt.legend() ; plt.grid(True) ; plt.show()
```

Figure: Generating data, fitting the model, showing the true model, the data and the fitted model.

The output

OLS Regression Results

Dep. Variable:	y	R-squared:	0.930
Model:	OLS	Adj. R-squared:	0.918
Method:	Least Squares	F-statistic:	79.36
Date:	Mon, 31 Mar 2025	Prob (F-statistic):	0.000111
Time:	13:50:01	Log-Likelihood:	1.2695
No. Observations:	8	AIC:	1.461
Df Residuals:	6	BIC:	1.620
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.1395	0.206	0.676	0.524	-0.366	0.645
x	0.6716	0.075	8.909	0.000	0.487	0.856

Omnibus:	7.386	Durbin-Watson:	2.389
Prob(Omnibus):	0.025	Jarque-Bera (JB):	2.297
Skew:	-1.252	Prob(JB):	0.317
Kurtosis:	3.786	Cond. No.	7.47

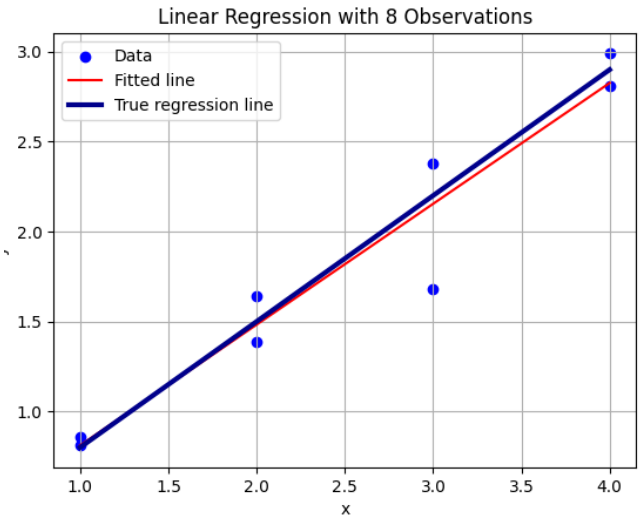


Figure: Output from statsmodel library

What we obtained

- With these 8 data points, we obtain the Ls estimate: $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1) = (0.14, 0.67)'$
- The true (and and supposedly unknown) parameter vector is: $\boldsymbol{\beta} = (0.1, 0.7)'$.
- Clearly, $\hat{\boldsymbol{\beta}} \neq \boldsymbol{\beta}$.
- The estimation error is:

$$\hat{\boldsymbol{\beta}} - \boldsymbol{\beta} = \begin{pmatrix} 0.14 \\ 0.67 \end{pmatrix} - \begin{pmatrix} 0.1 \\ 0.7 \end{pmatrix} = \begin{pmatrix} 0.04 \\ -0.03 \end{pmatrix}$$

- Next, we simulate a second dataset from the same linear regression model using a different random seed. Thus, the only change is in the vector of errors $\boldsymbol{\varepsilon}$.

The second simulation

```
# A second set of observations follwoing the same linear regression model.

# New sedd
np.random.seed(1234)
# New noise ~ N(0, 0.3^2)
noise2 = np.random.normal(0, 0.3, size=8)
# New response
y2 = 0.1 + 0.7 * x + noise2
# Create DataFrame
df2 = pd.DataFrame({'x': x, 'y': y2})
# Fit linear regression model
X = sm.add_constant(df2['x'])  # Add intercept
model2 = sm.OLS(df2['y'], X).fit()
# Print model summary
print(model2.summary())

# Optional: Plot the data and fitted line
plt.scatter(df2['x'], df2['y'], color='blue', label='Data')
plt.plot(df2['x'], model2.predict(X), color='red', label='Fitted line')
plt.plot(x, mu_true, color='darkblue', linewidth=3, label='True regression line')
plt.xlabel('x'); plt.ylabel('y')
plt.title('Linear Regression with 8 Observations')
plt.legend() ; plt.grid(True) ; plt.show()
```

Figure: Code for the second simulation.

The second simulation output

OLS Regression Results

Dep. Variable:	y	R-squared:	0.907
Model:	OLS	Adj. R-squared:	0.891
Method:	Least Squares	F-statistic:	58.17
Date:	Mon, 31 Mar 2025	Prob (F-statistic):	0.000265
Time:	13:53:45	Log-Likelihood:	-0.61906
No. Observations:	8	AIC:	5.238
Df Residuals:	6	BIC:	5.397
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.0593	0.261	0.227	0.828	-0.580	0.699
x	0.7281	0.095	7.627	0.000	0.495	0.962

Omnibus:	1.813	Durbin-Watson:	2.848
Prob(Omnibus):	0.404	Jarque-Bera (JB):	0.732
Skew:	0.099	Prob(JB):	0.694
Kurtosis:	1.532	Cond. No.	7.47

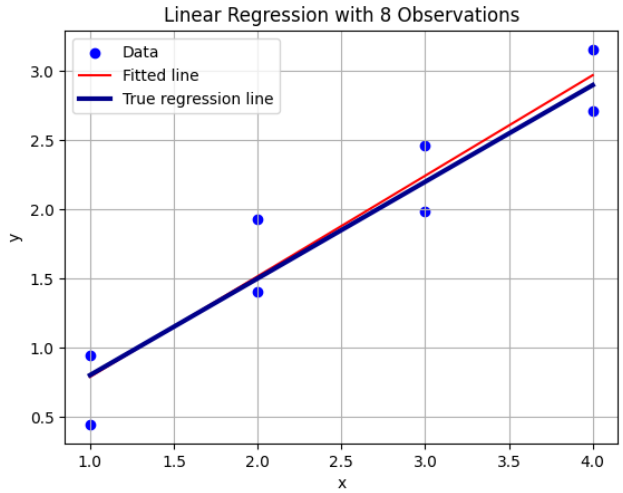


Figure: Output from the second simulation

Results of the second simulation

- With these new 8 data points, we obtain a different Ls estimate: $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1) = (0.06, 0.73)'$
- The true parameter vector remains the same: $\boldsymbol{\beta} = (0.1, 0.7)'$.
- Again, $\hat{\boldsymbol{\beta}} \neq \boldsymbol{\beta}$.
- The estimation error for this second sample from the same model is:

$$\hat{\boldsymbol{\beta}} - \boldsymbol{\beta} = \begin{pmatrix} 0.06 \\ 0.73 \end{pmatrix} - \begin{pmatrix} 0.1 \\ 0.7 \end{pmatrix} = \begin{pmatrix} -0.04 \\ 0.03 \end{pmatrix}$$

- This new sample leads to a different estimate $\hat{\boldsymbol{\beta}}$ and a different estimation error.
- What is the ontological status (i.e., the nature) of the objects $\hat{\boldsymbol{\beta}}$ and $\boldsymbol{\beta}$? Are they fixed vectors, random vectors, or mathematical functions?

Learning the random nature of $\hat{\beta}$

- Repeat the data generation of the random errors ε 50 times holding fixed $\boldsymbol{X}$ and the parameter vector β.

```
# Set parameters
np.random.seed(432); n_simulations = 50; x = np.repeat([1, 2, 3, 4], 2)
true_intercept = 0.1 ; true_slope = 0.7 ; sigma = 0.3

intercepts = [] ; slopes = [] # Store estimated parameters
plt.figure(figsize=(10, 6)) # Set up the figure for regression lines

# Run simulations
for i in range(n_simulations):
    noise = np.random.normal(0, sigma, size=len(x)) # Generate noise and Y
    y = true_intercept + true_slope * x + noise
    X = sm.add_constant(x) ; model = sm.OLS(y, X).fit() # Fit OLS model
    # Save estimates
    intercepts.append(model.params[0]) ; slopes.append(model.params[1])
    # Plot regression line
    y_fit = model.predict(X); plt.plot(x, y_fit, color='lightgray', linewidth=1)

# Plot the true regression line
X_true = sm.add_constant(x) ; y_true = true_intercept + true_slope * x
plt.plot(x, y_true, color='darkblue', linewidth=3, label='True regression line')
plt.title("Monte Carlo Simulation: 50 Fitted Regression Lines")
plt.xlabel("x"); plt.ylabel("y"); plt.legend(); plt.grid(True); plt.show()
```

The 50 fitted regression lines with the true line

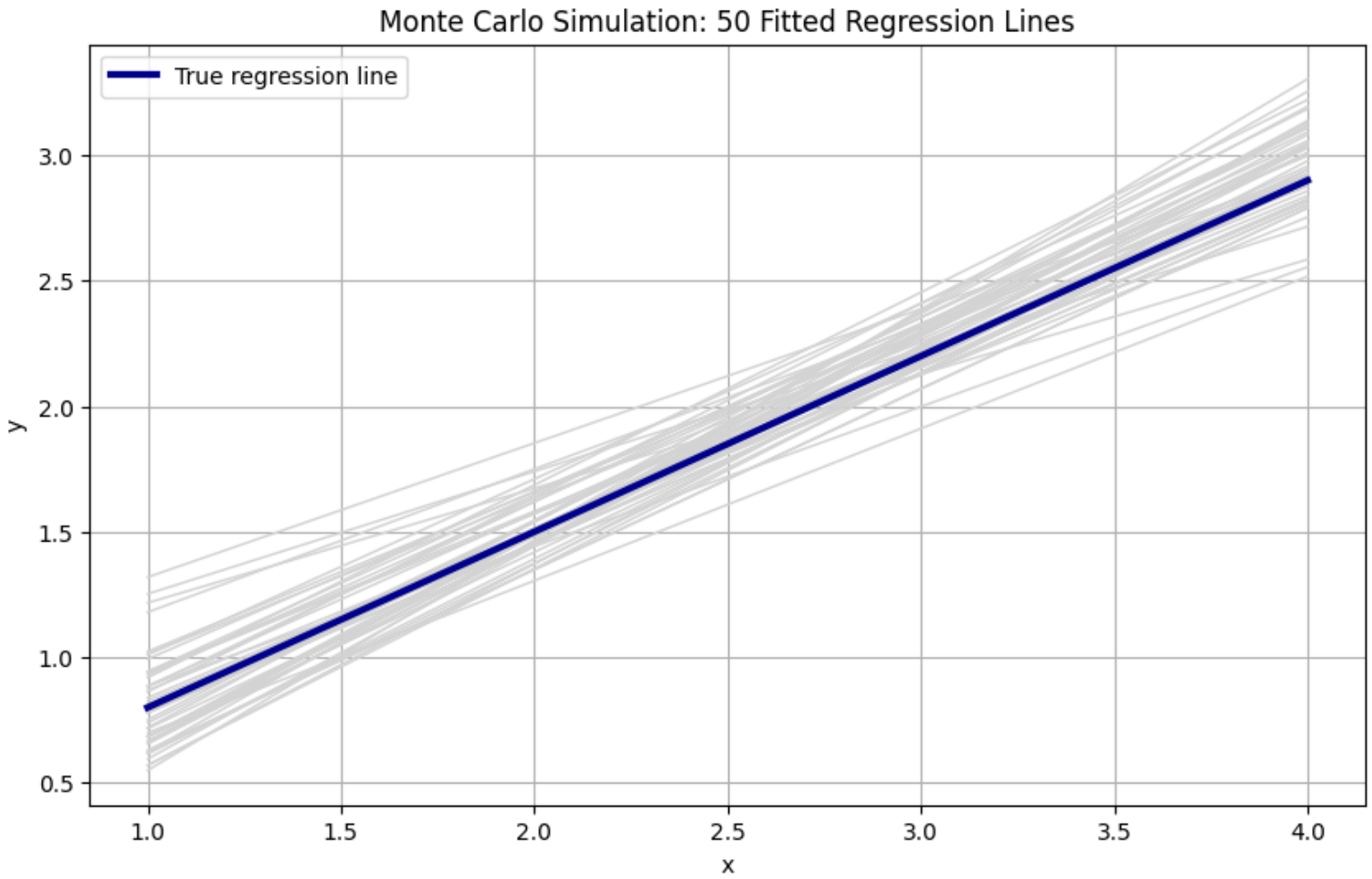


Figure: The true (blue) and the 50 (gray) estimated regression lines.

Code for the histograms of the 50 fitted coefficients

```
# Histograms of the parameter estimates
fig, axes = plt.subplots(1, 2, figsize=(14, 5))

# Intercept
axes[0].hist(intercepts, bins=10, color='lightgray', edgecolor='black')
axes[0].axvline(true_intercept, color='blue', linestyle='dashed', linewidth=2)
axes[0].set_title("Distribution of Intercept Estimates")
axes[0].set_xlabel("Intercept")
axes[0].set_ylabel("Frequency")

# Slope
axes[1].hist(slopes, bins=10, color='lightgray', edgecolor='black')
axes[1].axvline(true_slope, color='blue', linestyle='dashed', linewidth=2)
axes[1].set_title("Distribution of Slope Estimates")
axes[1].set_xlabel("Slope")
axes[1].set_ylabel("Frequency")

plt.tight_layout()
plt.show()
```

Figure: Code for histograms

The histograms of the 50 fitted coefficients

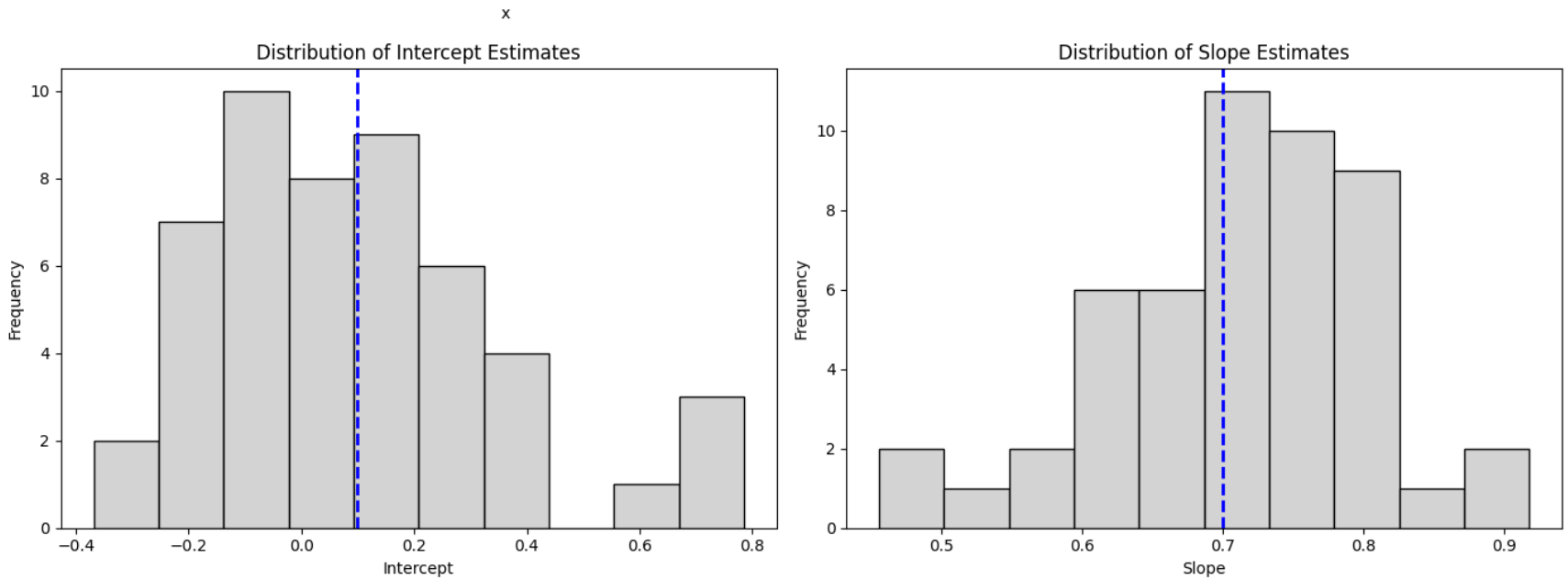
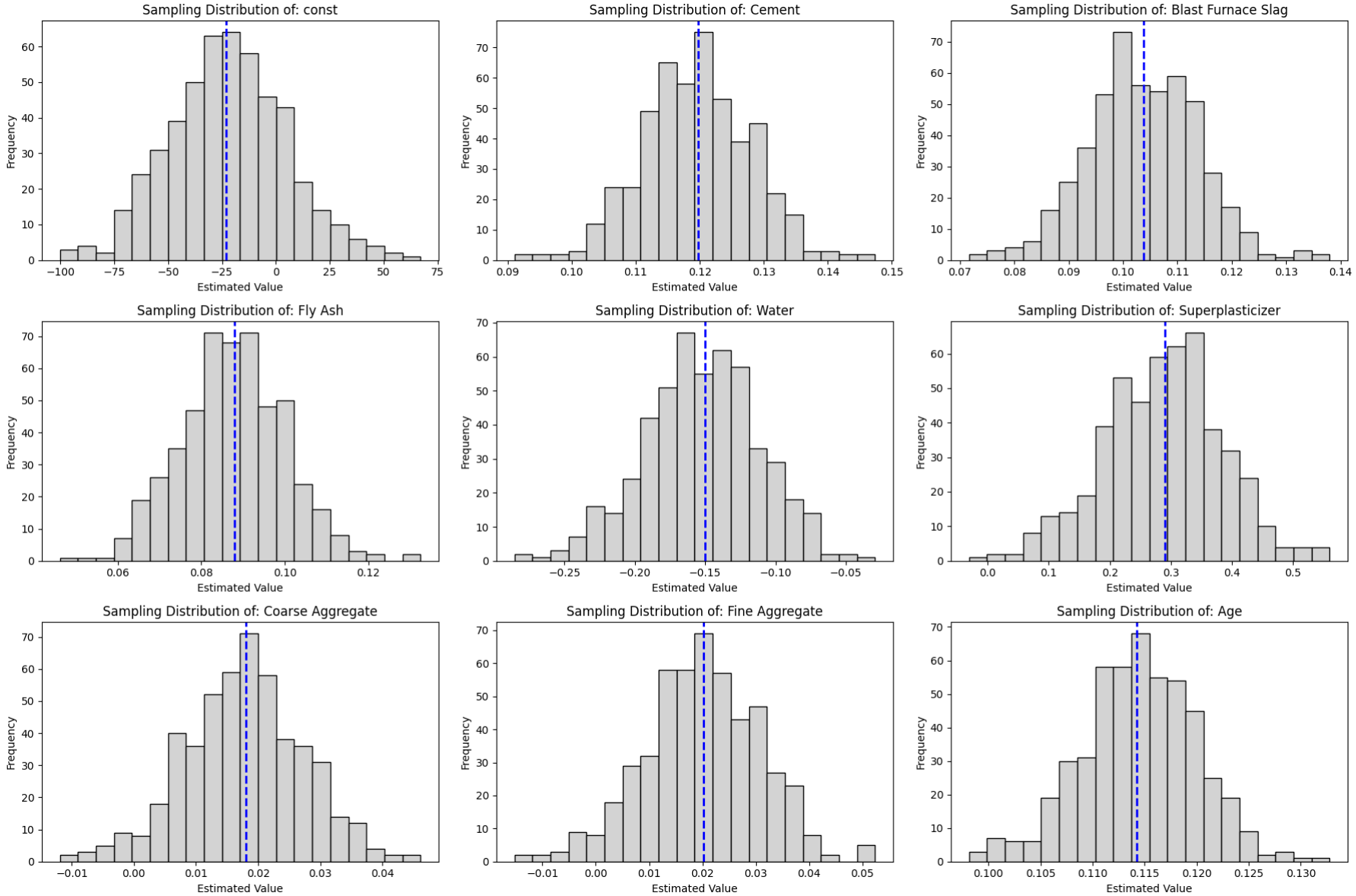


Figure: The histograms of $\hat{\beta}_1$ and $\hat{\beta}_2$ showing their random nature. The blue vertical line shows the true coefficient values.

Real case: Cement strength

- $\hat{\boldsymbol{\beta}}$ is a random vector.
- Let us see this by simulating data using the cement strength dataset.
- Fit the linear regression model to the dataset and obtain $\hat{\boldsymbol{\beta}}$.
- Take the residuals and obtain the $\widehat{\sigma^2} = \frac{1}{1030-(8+1)} \sum_i (y_i - \hat{y}_i)^2$.
- Simulate a large number of times *new* data vectors $\boldsymbol{Y}^*$ with the model $\boldsymbol{Y}^* = \boldsymbol{X}\hat{\boldsymbol{\beta}} + \boldsymbol{\varepsilon}$ with $\boldsymbol{\varepsilon} \sim \mathcal{N}_{1030}(\mathbf{0}, \widehat{\sigma^2}\boldsymbol{I}_{1030})$.
- Fit the linear regression to these simulated data and save the fitted parameter vectors.

Real case: Cement strength



The random nature of $\hat{\boldsymbol{\beta}}$

- You know from FECD-A that a histogram mimics the true probability distribution of the random variable it represents.
- Return to the histograms of the components $\hat{\beta}_j$ of the vector $\hat{\boldsymbol{\beta}}$.
- They illustrate the rigorous result we obtained:

$$\hat{\boldsymbol{\beta}} \sim \mathcal{N}_{p+1}(\boldsymbol{\beta}, \sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1})$$

  - The histograms look bell-shaped as a Gaussian density. This illustrates that $\mathbb{E}(\hat{\boldsymbol{\beta}}) = \boldsymbol{\beta}$ (the LS estimator is unbiased to estimate $\boldsymbol{\beta}$).
  - The histograms of $\hat{\beta}_j$ are centered on the true values β_j of the individual coefficients. This illustrates that $\mathbb{E}(\hat{\boldsymbol{\beta}})$ has a Gaussian distribution.
  - Let us focus now on the covariance matrix $\sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1}$ of $\hat{\boldsymbol{\beta}}$

The covariance matrix $\sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1}$ of $\hat{\boldsymbol{\beta}}$

- We have

$$\hat{\boldsymbol{\beta}} \sim \mathcal{N}_{p+1}(\boldsymbol{\beta}, \sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1})$$

- What does the matrix $(\boldsymbol{X}'\boldsymbol{X})$ mean?
- Let us consider only the previous simple example of 8 data points with only one feature x.
- Then,

$$\boldsymbol{X}'\boldsymbol{X} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 & 3 & 3 & 4 & 4 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 2 \\ 1 & 2 \\ 1 & 3 \\ 1 & 3 \\ 1 & 4 \\ 1 & 4 \end{pmatrix} = \begin{pmatrix} 8 & 20 \\ 20 & 30 \end{pmatrix}$$

The covariance matrix $\sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1}$ of $\hat{\boldsymbol{\beta}}$

- We have

$$\hat{\boldsymbol{\beta}} \sim \mathcal{N}_{p+1}(\boldsymbol{\beta}, \sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1})$$

- Then,

$$\sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1}) = (0.3)^2 \begin{pmatrix} 8 & 20 \\ 20 & 30 \end{pmatrix}^{-1} = (0.3)^2 \begin{pmatrix} 0.75 & -0.25 \\ -0.25 & 0.10 \end{pmatrix}$$

$$= \begin{pmatrix} \mathbb{V}(\hat{\beta}_1) & \text{Cov}(\hat{\beta}_1, \hat{\beta}_2) \\ \text{Cov}(\hat{\beta}_1, \hat{\beta}_2) & \mathbb{V}(\hat{\beta}_1) \end{pmatrix}$$

- We can easily extract the correlation between $\hat{\beta}_1$ and $\hat{\beta}_2$:

$$\text{Corr}(\hat{\beta}_1, \hat{\beta}_2) = \frac{\text{Cov}(\hat{\beta}_1, \hat{\beta}_2)}{\sqrt{\mathbb{V}(\hat{\beta}_1)}\sqrt{\mathbb{V}(\hat{\beta}_2)}} = \frac{-0.25(0.3)^2}{\sqrt{0.75}\sqrt{0.10}(0.3)^2} = -0.91$$

- Why such a high negative correlation? Intuition...

The covariance matrix $\sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1}$ of $\hat{\boldsymbol{\beta}}$

- Generally, with 2 features, we have

$$\boldsymbol{X}'\boldsymbol{X} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ x_{11} & x_{21} & x_{31} & \dots & x_{n1} \\ x_{12} & x_{22} & x_{32} & \dots & x_{n2} \end{pmatrix} \begin{pmatrix} 1 & x_{11} & x_{12} \\ 1 & x_{21} & x_{22} \\ \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} \end{pmatrix}$$

$$= \begin{pmatrix} n & \sum_i x_{i1} & \sum_i x_{i2} \\ \sum_i x_{i1} & \sum_i x_{i1}^2 & \sum_i x_{i1}x_{i2} \\ \sum_i x_{i2} & \sum_i x_{i1}x_{i2} & \sum_i x_{i2}^2 \end{pmatrix}$$

- The entry (i, j) in the matrix $(\boldsymbol{X}'\boldsymbol{X})$ is the inner product of the i-th and j-th features.

Real case: Cement strength

- Aim: To predict the compressive strength of concrete based on material composition.

Target Variable (Response Variable)

Feature Name	Description	Units	Typical Range
Compressive Strength	The maximum compressive stress the concrete can withstand.	MPa (MegaPascals)	2.33 - 82.6

- Number of Samples: 1,030 observations
- Number of Features: 8 predictors

Features (Predictor Variables)

Feature Name	Description	Units	Typical Range
Cement	The amount of cement used in the mix.	kg/m^3	102 - 540
Blast Furnace Slag	By-product of steel production, often used as a cement substitute.	kg/m^3	0 - 359.4
Fly Ash	A by-product of coal combustion, used as a partial cement replacement.	kg/m^3	0 - 200.1
Water	The amount of water used in the mix.	kg/m^3	121.8 - 247
Superplasticizer	Chemical additive to enhance workability and strength.	kg/m^3	0 - 32.2
Coarse Aggregate	Gravel or crushed stone used as a filler material.	kg/m^3	801 - 1145
Fine Aggregate	Sand used as a filler material.	kg/m^3	594 - 992.6
Age	Age of the concrete sample when tested.	days	1 - 365

Figure: Cement strength dataset

```python
#generate OLS regression results for all features
import statsmodels.api as sm

X_sm = sm.add_constant(X)
model = sm.OLS(y,X_sm)
print(model.fit().summary())
```

OLS Regression Results

Dep. Variable:	csMPa	R-squared:	0.616
Model:	OLS	Adj. R-squared:	0.613
Method:	Least Squares	F-statistic:	204.3
Date:	Fri, 15 Oct 2021	Prob (F-statistic):	6.29e-206
Time:	16:43:15	Log-Likelihood:	-3869.0
No. Observations:	1030	AIC:	7756.
Df Residuals:	1021	BIC:	7800.
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-23.3312	26.586	-0.878	0.380	-75.500	28.837
cement	0.1198	0.008	14.113	0.000	0.103	0.136
slag	0.1039	0.010	10.247	0.000	0.084	0.124
flyash	0.0879	0.013	6.988	0.000	0.063	0.113
water	-0.1499	0.040	-3.731	0.000	-0.229	-0.071
superplasticizer	0.2922	0.093	3.128	0.002	0.109	0.476
coarseaggregate	0.0181	0.009	1.926	0.054	-0.000	0.037
fineaggregate	0.0202	0.011	1.887	0.059	-0.001	0.041
age	0.1142	0.005	21.046	0.000	0.104	0.125

Figure: $\hat{\beta}_j$ and its standard deviation $\sqrt{\mathbb{V}(\hat{\beta}_j)} = \sqrt{\sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1}[j,j]}$.

The standard deviation of $\hat{\beta}_j$

- The standard deviation of $\hat{\beta}_j$ is the square root of the $[j, j]$-th element in the diagonal of the covariance matrix of $\hat{\beta}$:

$$\sqrt{\mathbb{V}(\hat{\beta}_j)} = \sqrt{\sigma^2(\boldsymbol{X}'\boldsymbol{X})^{-1}[j, j]}$$

- This element depends of σ^2, also an unknown parameter.
- We estimate this parameter σ^2 using the residuals:

$$\widehat{\sigma^2} = \frac{1}{n - (p + 1)} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

- Next, we look more closely at this estimator of σ^2