

# Introduction to Machine Learning

## Lecture 4 Linear Models I

Goker Erdogan  
26 – 30 November 2018  
Pontificia Universidad Javeriana

# Types of learning problems

| Supervised learning   | Unsupervised learning   | Reinforcement learning  |
|---|---|---|
| <p>Learning from supervision (i.e., teacher)</p> <p>Given inputs and <b>outputs</b>, learn a function that maps input → output</p>  | <p>Learning with no supervision</p> <p>Given only inputs, extract some pattern from the data.</p>   | <p>Learning from rewards and punishments.</p> <p>Inspired by (operant) conditioning in psychology.</p> <p>The agent acts in an environment over some <b>time</b>.</p> <p>It gets reward/punishment from time to time.</p> <p>Needs to learn a <i>policy</i> to maximize reward.</p> |
| <p>Examples:</p> <ul style="list-style-type: none"><li>- Recognizing faces</li><li>- Predicting the sales for a product</li><li>- Learning to rank search results</li></ul> | <p>Examples:</p> <ul style="list-style-type: none"><li>- Clustering (market segmentation)</li><li>- Dimensionality reduction (visualization)</li><li>- Generating celebrity faces</li></ul> | <p>Examples:</p> <ul style="list-style-type: none"><li>- Most robotics applications (e.g. learning to move around)</li><li>- Playing games (e.g., AlphaGo)</li></ul>  |

# Supervised learning

- Two main types of problems
  - **Regression**
    - Outputs are **real-valued** (quantitative)
    - Examples:
      - Predicting the sales of a product
      - Predicting the value of a home
  - **Classification**
    - Outputs are **discrete** (qualitative)
    - Examples:
      - Face/speech/object recognition
      - Predicting if a person will default on their loan
      - Predicting whether a stock will go up or down
- *Note inputs can be of any type (real-valued or discrete).*

# Linear regression

- Assume the relation between input and output is **linear**

$$y = \beta_0 + \sum_{d=1}^D x_d \beta_d$$

- $\beta$  are known as **parameters** or **coefficients**
- $\beta_0$  is known as **bias** or **intercept**

- Matrix notation

- $X_{N \times (D+1)}$ : Data matrix. N samples. D features (measurements)
  - Add a new feature (column) that is all 1s.
- $Y_N$ : Output (target) vector. N samples.
- $\beta_{D+1}$ : Parameter vector. D+1 coefficients (including intercept).
- Then, the above can be written as

$$Y = X\beta$$

# Linear regression

- Note that we can apply any (possibly **nonlinear**) preprocessing on  $x$

$$y = \beta_0 + \sum_{d=1}^D \phi_d(x) \beta_d$$

- This is still a **linear** model. Because it is **linear in the parameters**.
- Linearity

$$y(c_1\beta + c_2\gamma) = c_1y(\beta) + c_2y(\gamma)$$

- Why do we care about linear regression?
  - Easy to interpret
  - May outperform complex models if
    - Little data
    - High noise in data
  - Foundation for many techniques

# Solving linear regression

- Find the parameters that fit the training data  $\{x, y\}_{n=1,2,\dots,N}$ 
  - Pick the *line* that passes as close as possible to the training points
- How do we measure **closeness**?

$$E(\beta) = \frac{1}{2} \sum_{n=1}^N (y - (\beta_0 + \sum_{d=1}^D X_{nd}\beta_d))^2$$

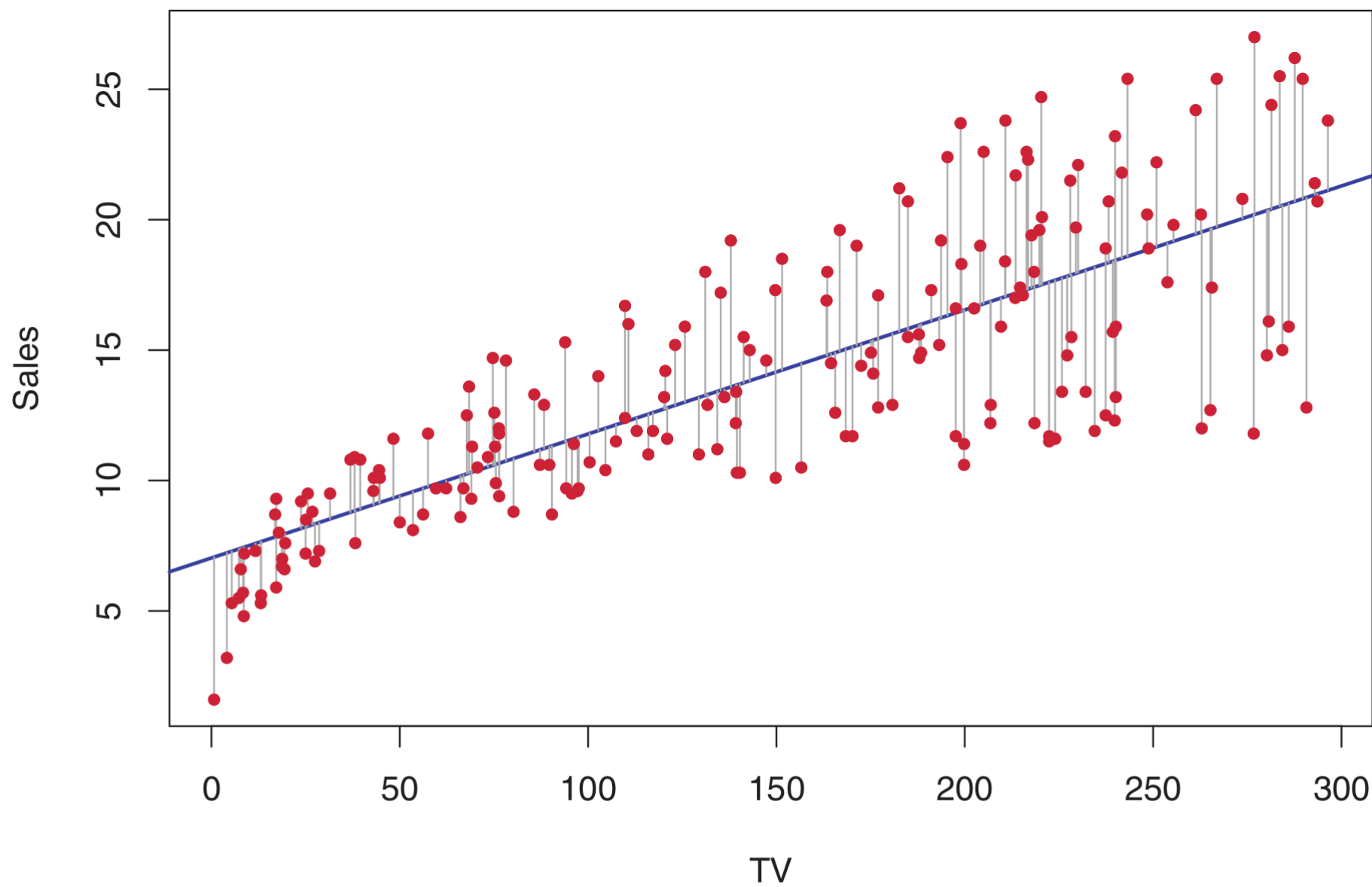
- Find parameters that minimize the **sum-of-squares** error (method of least squares)
  - Set derivative to 0 and solve for  $\beta$

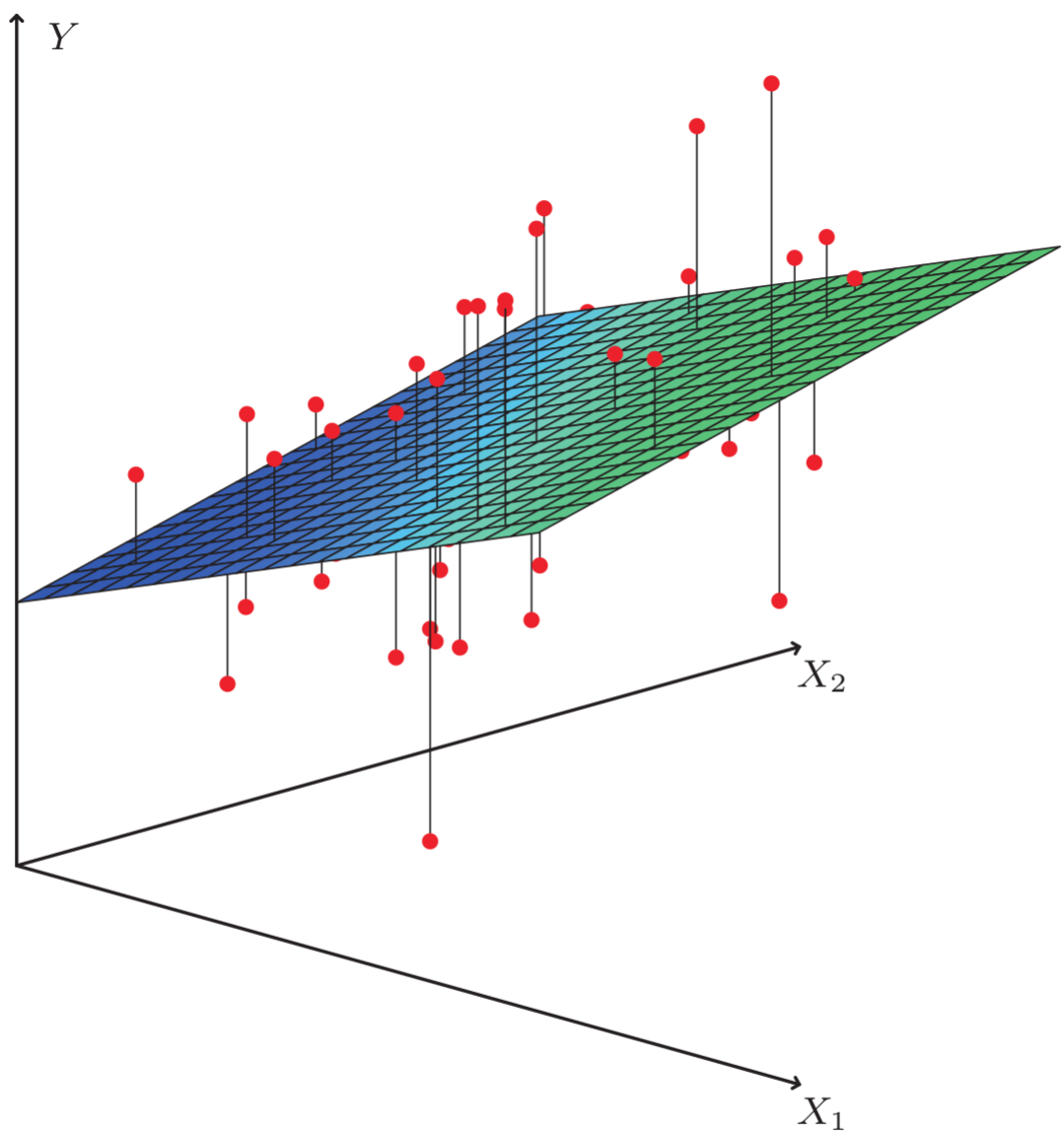
- For 1D case

$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2},$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x},$$

$\bar{x}$ : Mean of x  
 $\bar{y}$ : Mean of y

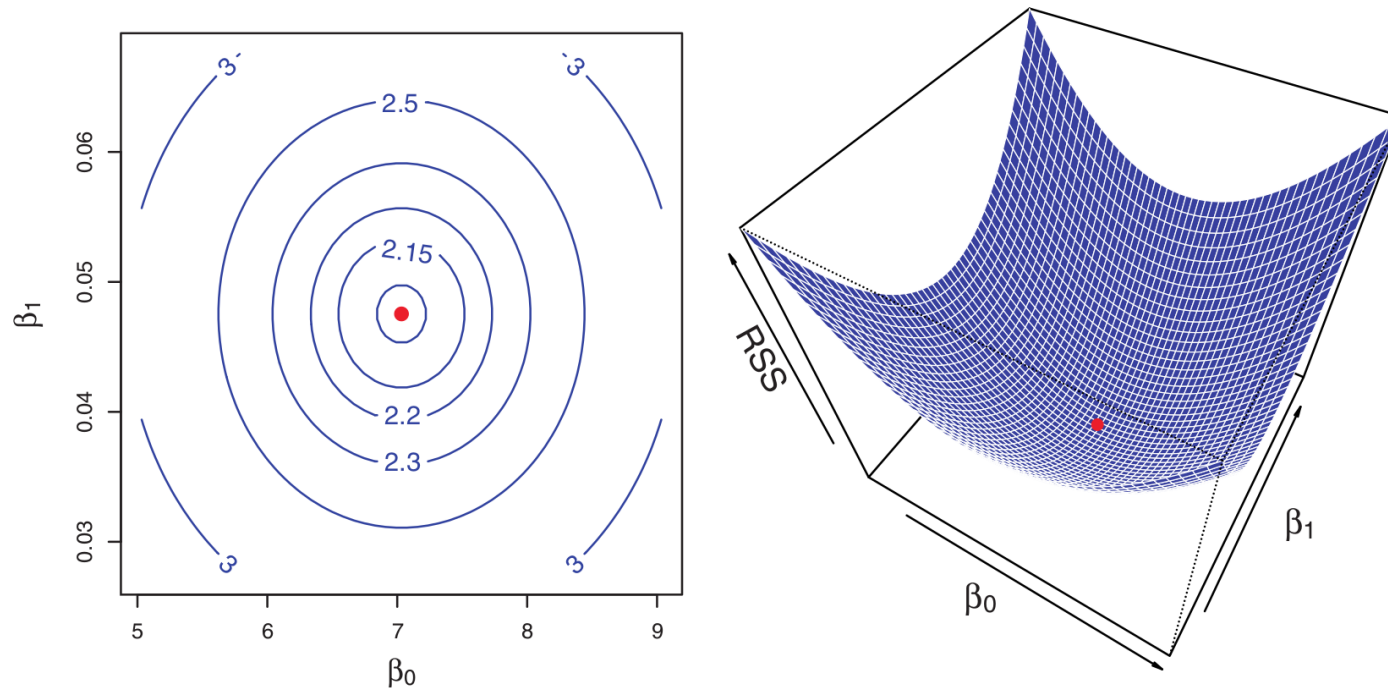






# Solving linear regression

- Linear regression is an easy problem
  - Convex (i.e., single global minimum)
  - Fast algorithms available



**FIGURE 3.2.** Contour and three-dimensional plots of the RSS on the Advertising data, using sales as the response and TV as the predictor. The red dots correspond to the least squares estimates  $\hat{\beta}_0$  and  $\hat{\beta}_1$ , given by (3.4).

# Solving the general case

- Minimize the following wrt to  $\beta$

$$\text{RSS}(\beta) = (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta).$$

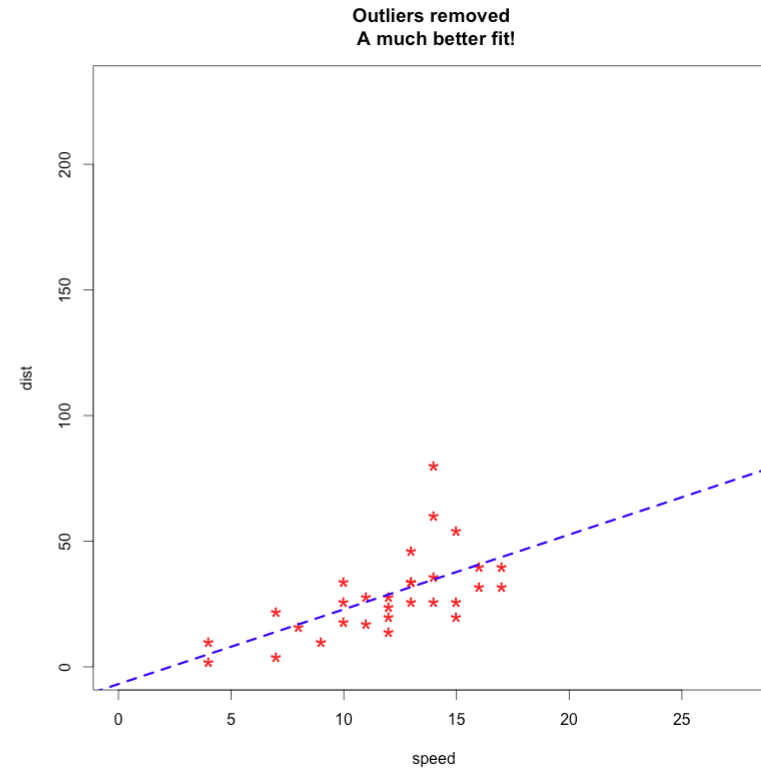
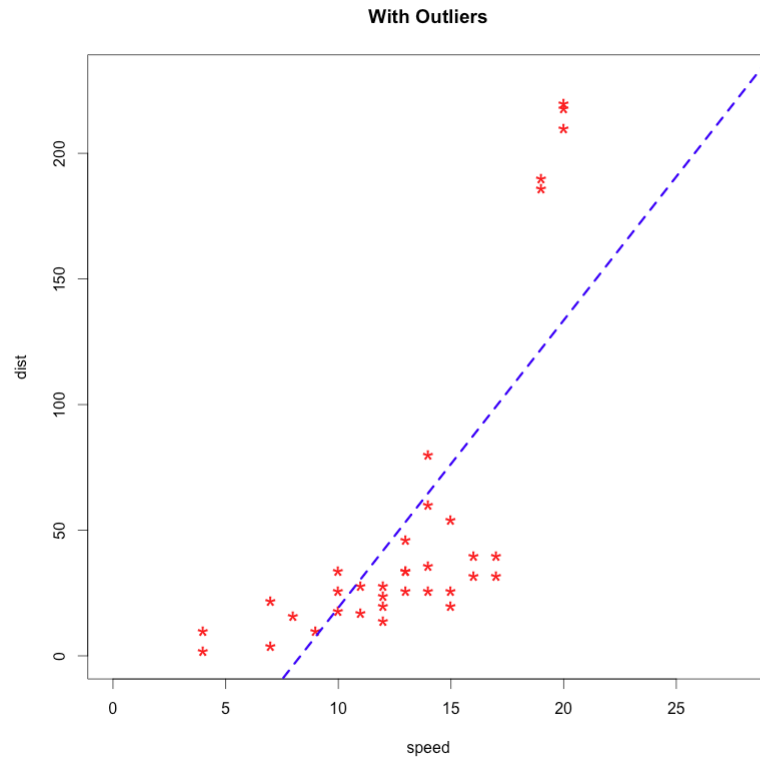
- Need to assume that  $\mathbf{X}$  is full rank (no linearly dependent columns)
- The solution is

$$\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}.$$

- In python, `numpy.linalg.lstsq`
- In R, `fit <- lm(y ~ x1 + x2 + x3 ...)`
- Good resources for matrix calculus
  - <http://www.matrixcalculus.org/>
  - <https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf>

# What can go wrong?

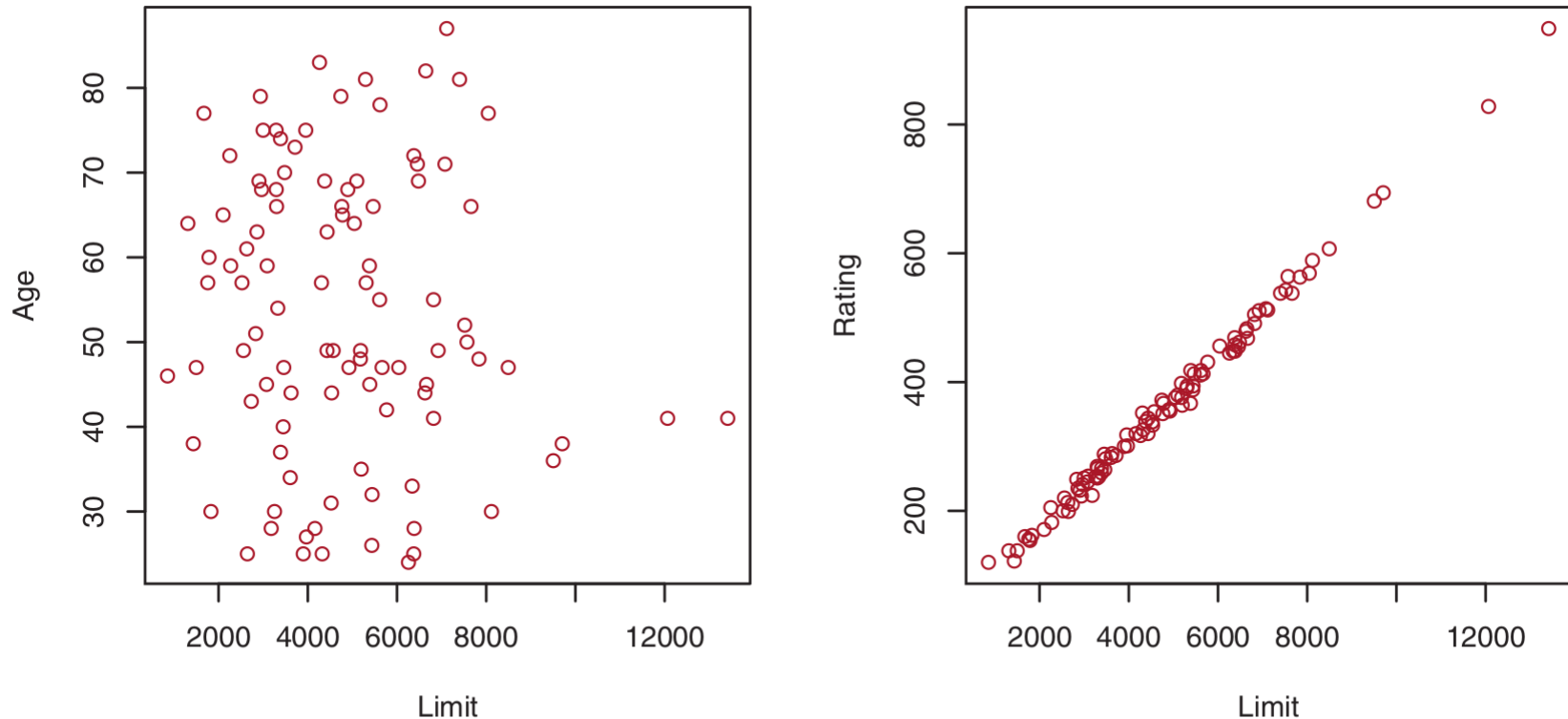
- Outliers



- Robust alternatives
  - Use absolute value instead of squared error
  - Harder to solve

# What can go wrong?

- Correlated inputs (**colinearity**)



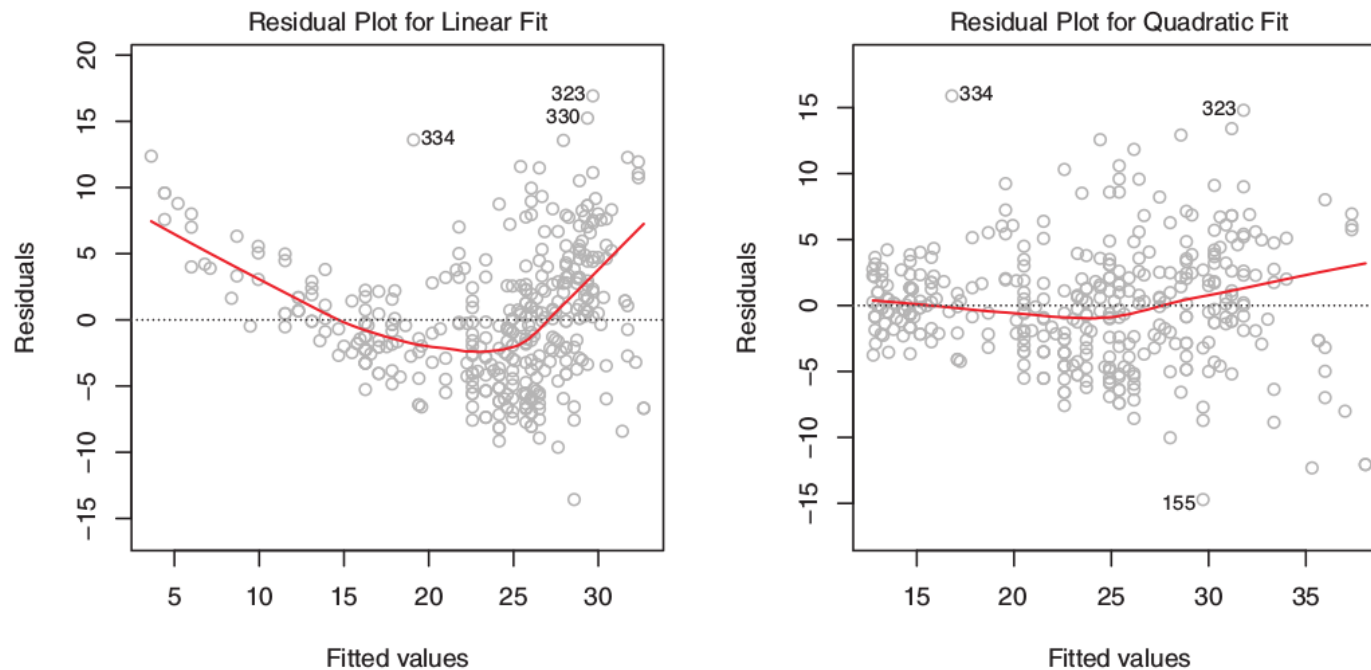
**FIGURE 3.14.** Scatterplots of the observations from the **Credit** data set. Left: A plot of **age** versus **limit**. These two variables are not collinear. Right: A plot of **rating** versus **limit**. There is high collinearity.

# What can go wrong?

- Correlated inputs (**colinearity**)
  - Solution unstable
  - Hard to interpret
  - What can we do?
    - Look at the correlation matrix
    - Whiten input data (PCA)
- Correlation of error-terms
  - Violates the independence assumption
  - e.g., Time-series data

# What can go wrong?

- Nonlinear input – output relation
  - Interaction terms
  - Residual (error) plot



**FIGURE 3.9.** Plots of residuals versus predicted (or fitted) values for the **Auto** data set. In each plot, the red line is a smooth fit to the residuals, intended to make it easier to identify a trend. Left: A linear regression of **mpg** on **horsepower**. A strong pattern in the residuals indicates non-linearity in the data. Right: A linear regression of **mpg** on **horsepower** and **horsepower**<sup>2</sup>. There is little pattern in the residuals.

# Summary

- Linear regression
  - Formulation
  - How to solve it
  - What can go wrong
- Exercises
  - Solve 1D case
  - Do the lab in Section 3.6 of ISLR

# References

- [1] James, Witten, Hastie, and Tibshirani. An Introduction to Statistical Learning with Applications in R. Chapter 3.
- [2] Hastie, Tibshirani, and Friedman. The Elements of Statistical Learning. Chapter 3.
- [3] <http://r-statistics.co/Outlier-Treatment-With-R.html>