# Regression

Kaiqi Zhao
The University of Auckland

*Slides are partially based on the materials from the University of British Columbia*

# Content

- Regression
- Linear Regression
- Least Squares
- Locally Weighted Regression
- Regression Tree
- Summary

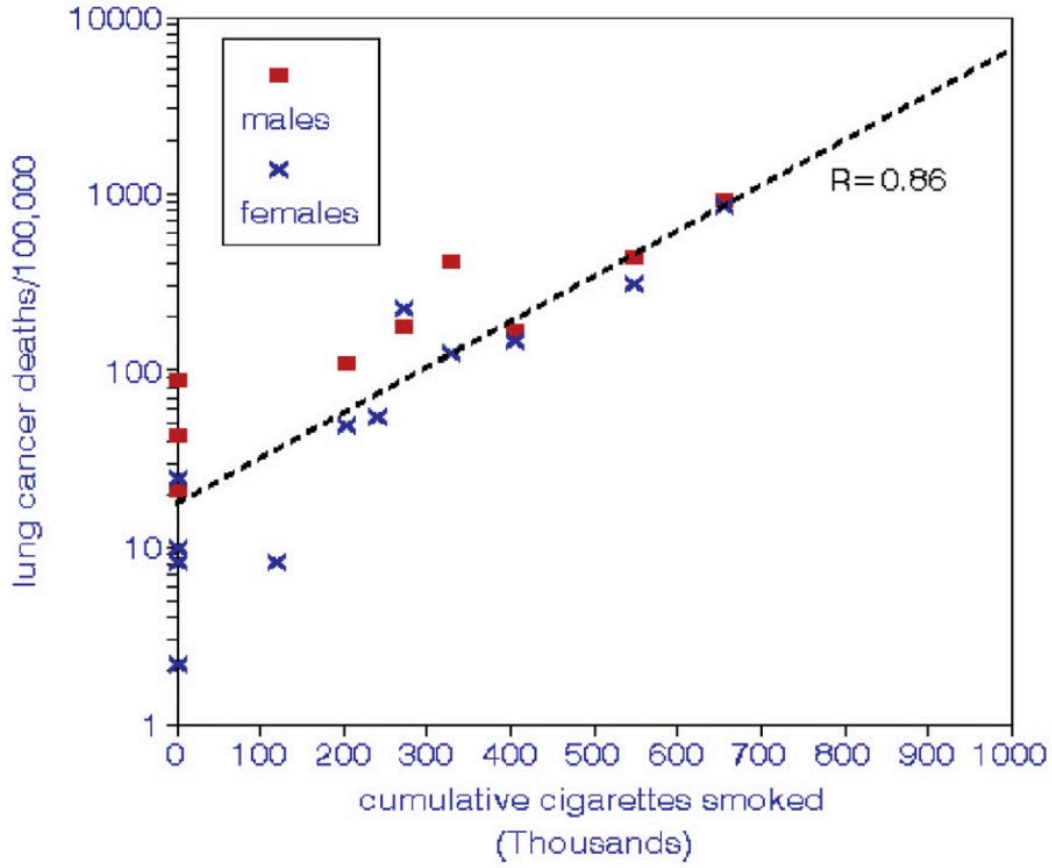
# Supervised Learning Round 2: Regression

- We are going to revisit supervised learning
- Previously, we considered classification
  - We assumed $y_i$ was discrete: $y_i = spam$ or $y_i = not\ spam$
- Now we are going to consider regression
  - We allow $y_i$ to be numerical, for example $y_3 = 10.34cm$

# Regression

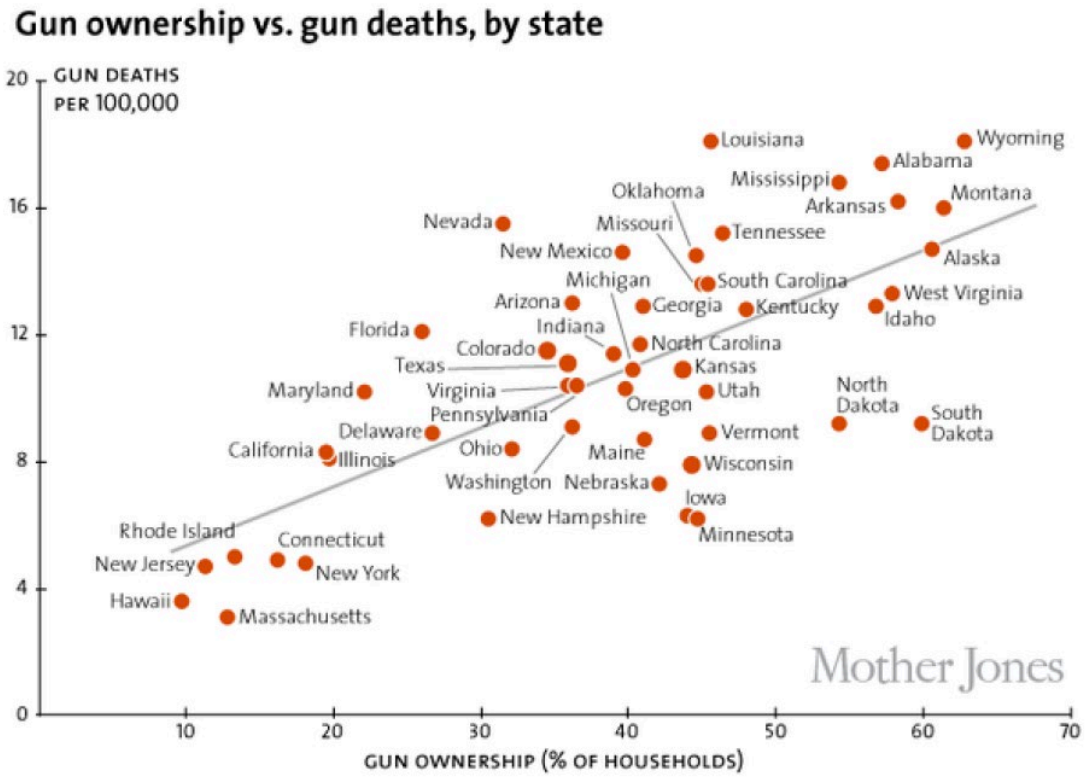
# Example: Dependent vs. Explanatory Variables

- We want to discover relationship between numerical variables
  - Does number of lung cancer deaths change with number of cigarettes?



# Example: Dependent vs. Explanatory Variables

- We want to discover relationship between numerical variables
  - Does number of lung cancer deaths change with number of cigarettes?
  - Does number of gun deaths change with gun ownership?

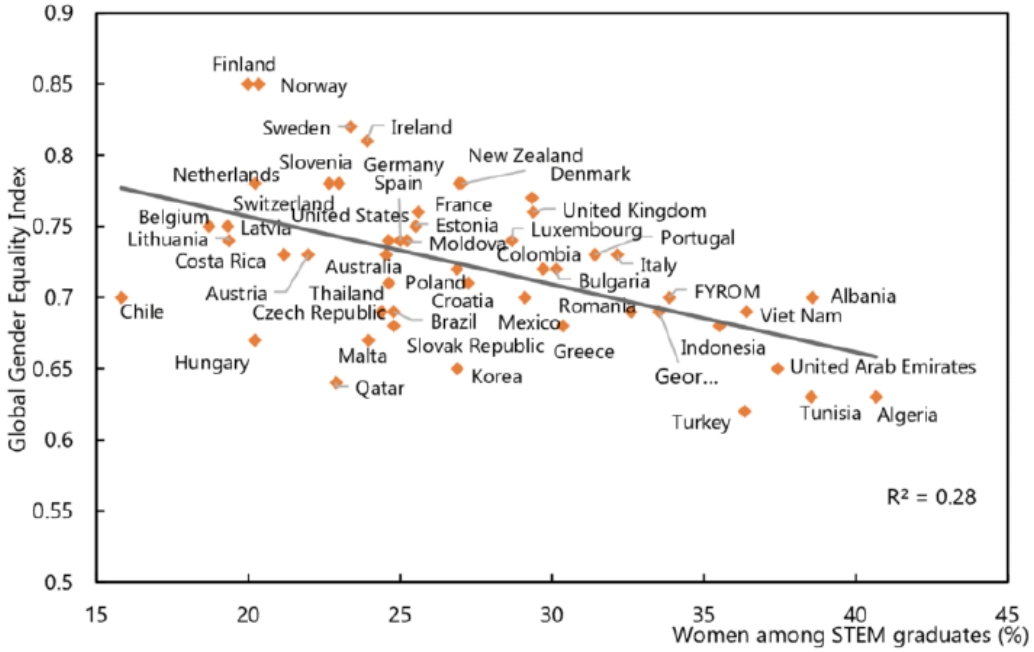


# Example: Dependent vs. Explanatory Variables

- We want to discover relationship between numerical variables
  - Does number of lung cancer deaths change with number of cigarettes?
  - Does number of gun deaths change with gun ownership?
  - Does number violent crimes change with violent video games?
  - Do people in big cities walk faster?
  - Is the universe expanding or shrinking or staying the same size?

# Example: Dependent vs. Explanatory Variables

- We're doing supervised learning
  - Trying to predict value of 1 variable (the $y_i$ values) – instead of measuring correlation between 2 variables
- Supervised learning does not give causality
  - OK: Higher gender equality index is correlated with lower graduation rate
  - OK: Higher gender equality index helps predict lower graduation rate
  - BAD: Higher gender equality index leads to lower graduation rate



# Example: Dependent vs. Explanatory Variables

- One way to handle numerical $y_i$ : discretize
  - E.g., for 'age' could we use $age \leq 20$, $20 < age \leq 30$, $age > 30$
  - Now we can apply methods for classification to do regression
  - Coarse discretization loses resolution
  - Fine discretization requires lots of data
- There exist regression versions of classification methods:
  - Regression trees, non-parametric models, probabilistic models
- We start with one of oldest, but still most popular/important methods
  - Linear regression based on squared error
  - Interpretable and the building block for more-complex methods

# Linear Regression

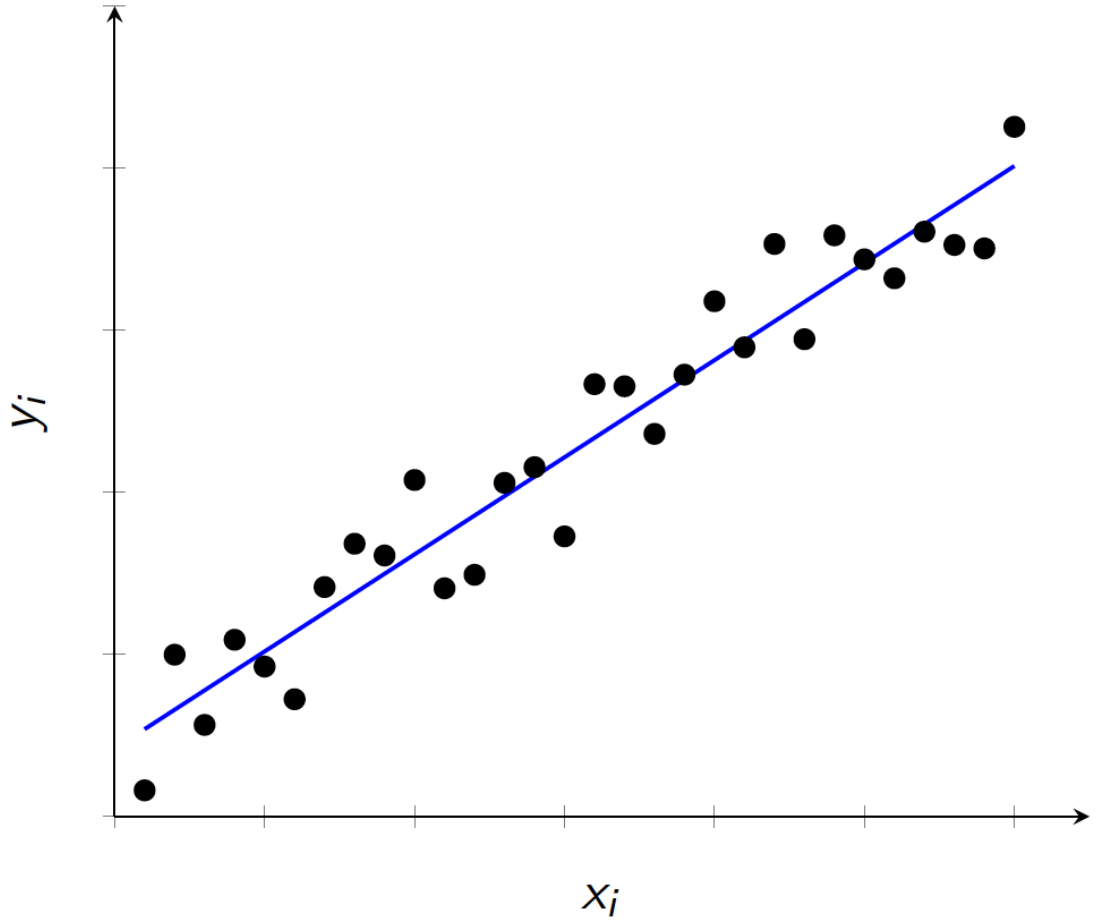
# Linear Regression in 1 Dimension

- Assume we only have 1 feature ($d = 1$)
  - E.g., $x_i$ is number of cigarettes and $y_i$ is number of lung cancer deaths
- Linear regression makes predictions $\hat{y}_i$ using a linear function of $x_i$

$$\hat{y}_i = wx_i$$

- The parameter $w$ is the weight or regression coefficient of $x_i$,
  - For explicity, we ignore the $y$-intercept
- As $x_i$ changes, slope $w$ affects the rate that $\hat{y}_i$ increases/decreases
- Positive $w$: $\hat{y}_i$ increase as $x_i$ increases
- Negative $w$: $\hat{y}_i$ decreases as $x_i$ increases

# Linear Regression in 1 Dimension




line $\hat{y}_i = wx_i$ for a particular slope $w$

# Least Squares

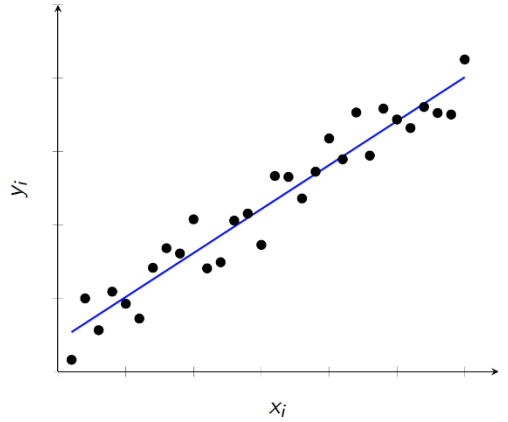
# Least Squares Objective

- Our linear model is given by

$$\hat{y}_i = wx_i$$

- So we make predictions for a new example $\tilde{x}_i$ by using

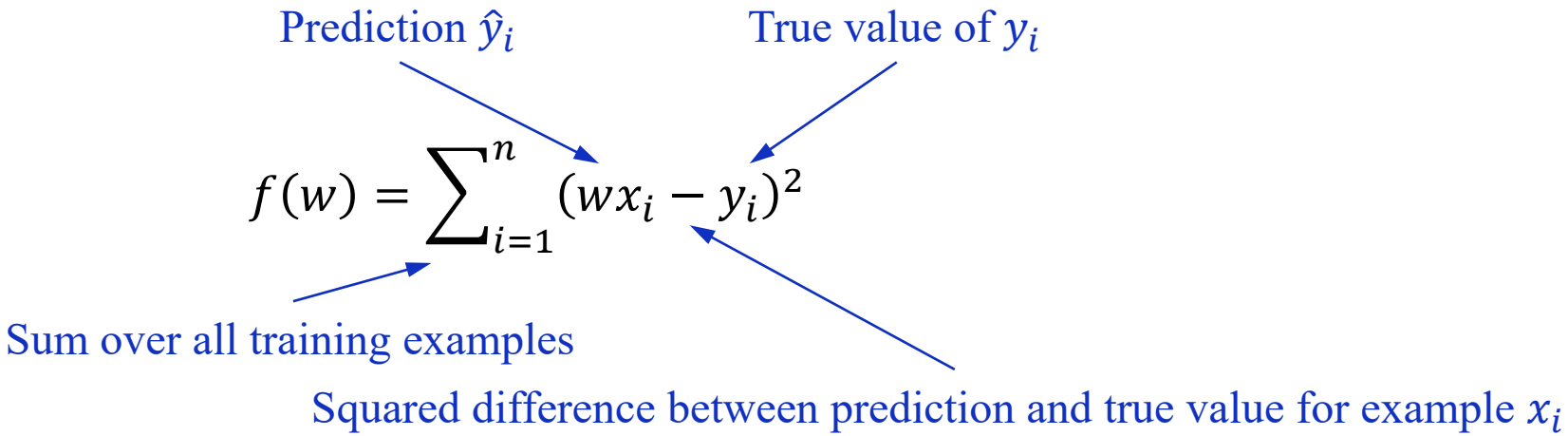
$$\hat{y}_i = w\tilde{x}_i$$




- But we can't use the same error as the one we used in classification
  - It is unlikely to find a line where $\hat{y}_i = y_i$ exactly for many points
    - Due to noise, relationship not being quite linear or just floating-point issues
  - Best model may have $|\hat{y}_i - y_i|$ is small but not exactly 0

# Least Squares Objective

- Instead of exact $y_i$, we evaluate size of the error in prediction
- Classic way is setting slope $w$ to minimize Sum of Squared Errors (SSE)

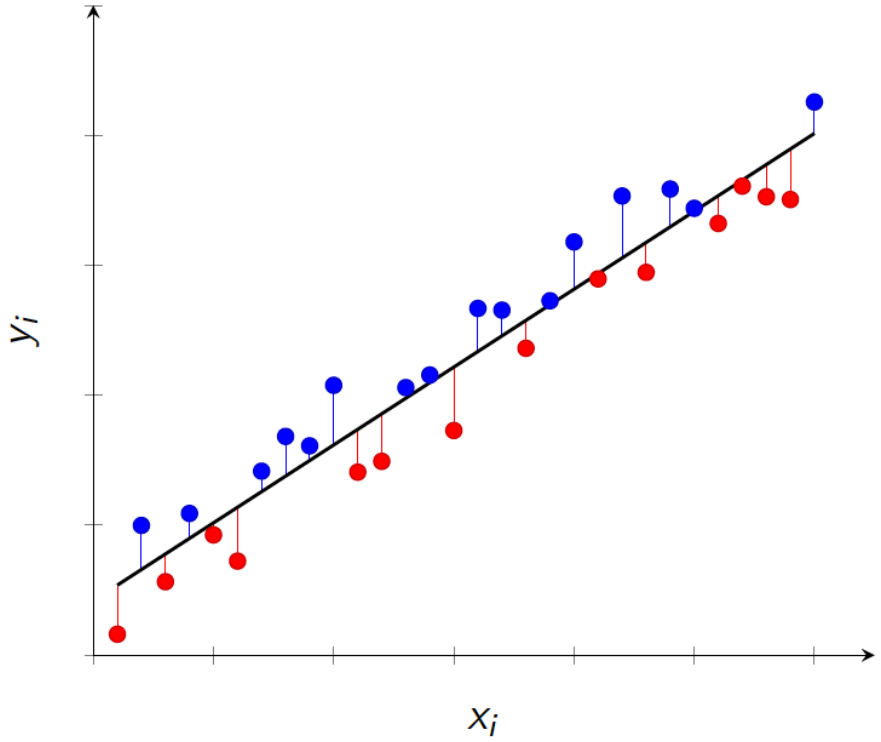



- There are some justifications for this choice, e.g., probabilistic models
- But usually, it is done because it is easy to minimize
  - Minimizing/maximizing a function often needs to compute the first derivative.
  - It is hard to compute the first derivative for the sum of absolute error $\sum_{i=1}^{n} |wx_i - y_i|$

# Least Squares Objective

- Classic way to set slope $w$ is minimizing sum of squared errors

$$f(w) = \sum_{i=1}^{n} (wx_i - y_i)^2$$




- "Error" is the sum of the squared values of these vertical distances between the line ($wx_i$) and the targets ($y_i$)
- If this error is small then our predictions are close to the target

# Minimizing a Differential Function

- Simple approach to minimizing a differentiable function $f$
  1. Take the derivative of $f$
  2. Find points $w$ where the derivative $f'(w)$ is equal to 0
  3. Choose the smallest one (and check that $f''(w)$ is positive).
- Note that this problem: $f(w) = \sum_{i=1}^{n}(wx_i - y_i)^2$, has the same set of minimizers as this problem:
  - E.g., $f(w) = \frac{1}{2}\sum_{i=1}^{n}(wx_i - y_i)^2 + 1000$
- We can multiply $f$ by any positive constant without changing the solution
  - Derivative will still be zero at the same locations
  - We will use this trick a lot!

# Finding Least Squares Solution

- Finding $w$ that minimizes the sum of squared errors

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(wx_i - y_i)^2 = \frac{1}{2}\sum_{i=1}^{n}[w^2x_i^2 - 2wx_iy_i + y_i^2]$$

$$= \frac{w^2}{2}\sum_{i=1}^{n}x_i^2 - w\sum_{i=1}^{n}x_iy_i + \frac{1}{2}\sum_{i=1}^{n}y_i^2$$

$$= \frac{w^2}{2}a - wb + c$$

Take derivative $f'(w) = wa - b + 0$

Setting $f'(w) = 0$ and solving gives

$$w = \frac{b}{a} = \frac{\sum_{i=1}^{n}x_iy_i}{\sum_{i=1}^{n}x_i^2}$$

# Finding Least Squares Solution

- Finding $w$ that minimizes sum of squared errors

$$w = \frac{b}{a} = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}$$

- Let's check that this is a minimizer by checking the second derivative

$$f'(w) = w \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i y_i$$

$$f''(w) = \sum_{i=1}^{n} x_i^2$$

  - Since $(anything)^2$ is non-negative and $(anything\ non-zero)^2 > 0$, if we have one $non-zero$ feature then $f''(w) > 0$ and this is a minimizer

# Least Squares in d-Dimensions

- Motivation: Smoking is not the only contributor to lung cancer
  - For example, there environmental factors like exposure to asbestos
- How can we model the combined effect of smoking and asbestos?
- A simple way is with a 2-dimensional linear function

$$\hat{y}_i = w_1 x_{i1} + w_2 x_{i2}$$

- We have a weight $w_1$ for feature 1 and $w_2$ for feature 2

$$\hat{y}_i = 10(\#cigarettes) + 25(\#asbestos)$$

# Least Squares in d-Dimensions

- If we have d features, the d-dimensional linear model is

$$\hat{y}_i = w_0 + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id}$$

  - In words, the output of our model is a weighted sum of the inputs
- We can re-write this in summation notation (setting $x_{i0} = 1$)

$$\hat{y}_i = \sum_{j=0}^{d} w_j x_{ij}$$

- We can also re-write this in vector notation

$$\hat{y}_i = w^T x_i$$

# Notation

- In the lectures, all vectors are assumed to be column-vectors

$$w = \begin{bmatrix} w_0 \\ w_1 \\ \dots \\ w_d \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_n \end{bmatrix} \qquad x_i = \begin{bmatrix} 1 \\ x_{i1} \\ \dots \\ x_{id} \end{bmatrix}$$

- So $w^T x_i$ is a scalar

$$w^T x_i = [w_0 \quad w_1 \quad \dots \quad w_d] \begin{bmatrix} 1 \\ x_{i1} \\ \dots \\ x_{id} \end{bmatrix} = w_0 + w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id} = w_0 + \sum_{j=1}^{d} w_j x_{ij}$$

- Rows of $X$ are transpose of column-vector $x$

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \dots \\ x_3^T \end{bmatrix}$$

# Least Squares in d-Dimensions

- The linear least squares model in d-dimensions minimizes

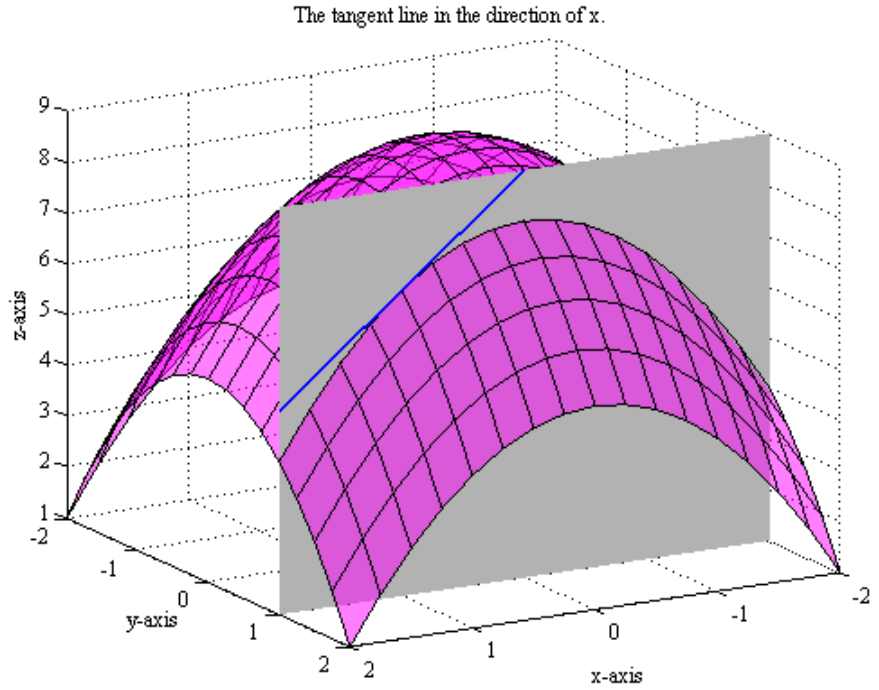
$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^T x_i - y_i)^2$$

  - $w$ is now a vector
  - $w^T x_i$ (prediction) is inner product of $w$ and $x_i$ (linear combination of features)
  - $\sum_{i=1}^{n}(w^T x_i - y_i)^2$ (error) is still the sum of squared differences between true $y_i$ and our prediction $w^T x_i$
- Dates back to 1801: Gauss used it to predict location of Ceres
- How do we find the best vector $w$ in $d$ dimensions?
  - Can we set the partial derivative of each variable to 0?

# Gradient and Critical Points in d-Dimensions

- Generalizing "set the derivative to 0 and solve" in d-dimensions:
  - Find 'w' where the gradient vector equals the zero vector
- Gradient is vector with partial derivatives w.r.t. $w_j$ at position $j$:

$$\nabla f(w) = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_2} \\ \frac{\partial f}{\partial w_3} \\ \vdots \\ \frac{\partial f}{\partial w_d} \end{bmatrix}$$



# Gradient and Critical Points in d-Dimensions

- Generalizing "set the derivative to 0 and solve" in d-dimensions:
  - Find 'w' where the gradient vector equals the zero vector
- Gradient is vector with partial derivatives w.r.t. $w_j$ at position $j$:

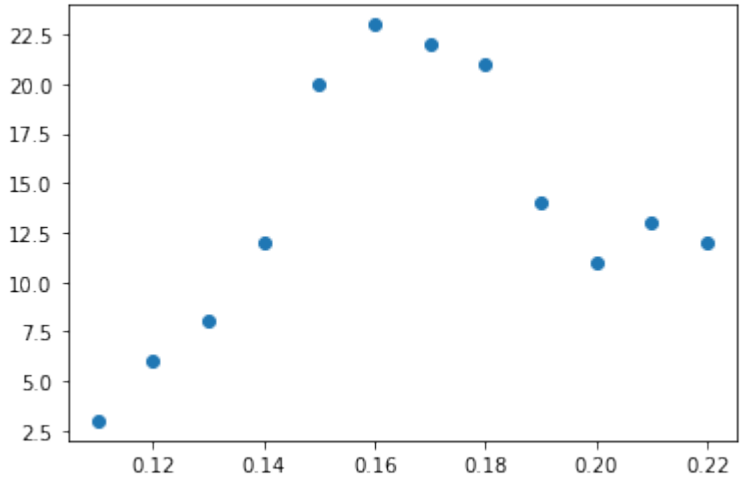
$$\nabla f(w) = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_2} \\ \frac{\partial f}{\partial w_3} \\ \vdots \\ \frac{\partial f}{\partial w_d} \end{bmatrix}$$

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(w^T x_i - y_i)^2 = \frac{1}{2}\sum_{i=1}^{n}\left[\left(\sum_{j=1}^{d} w_j x_{ij}\right)^2 - 2y_i\left(\sum_{j=1}^{d} w_j x_{ij}\right) + y_i^2\right]$$

$$\frac{\partial f}{\partial w_1} = \sum_{i=1}^{n}\left[\left(\sum_{j=1}^{d} w_j x_{ij}\right)x_{i1} - y_i x_{i1}\right]$$

# Non-Linear Dataset

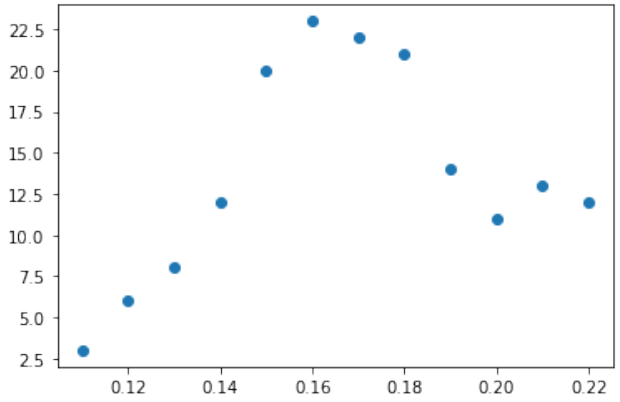
- The linear least squares model finds a straight line to fit the training data.
- It doesn't work when data points are not distributed along a line.




- What should we do?

# Non-Linear Feature Transforms

- If the data is quadratic, can we use linear regression to fit a quadratic function?
  - $y = w_0 + w_1 x + w_2 x^2$
- Yes, if we make a simple feature transformation, e.g.,
  - Let $z_1 = x, z_2 = x^2$
  - The original quadratic function becomes $y = w_0 + w_1 z_1 + w_2 z_2$
  - This transformation can be extended to d-dimension
- Limitation: you may not know which function fits the best in many cases



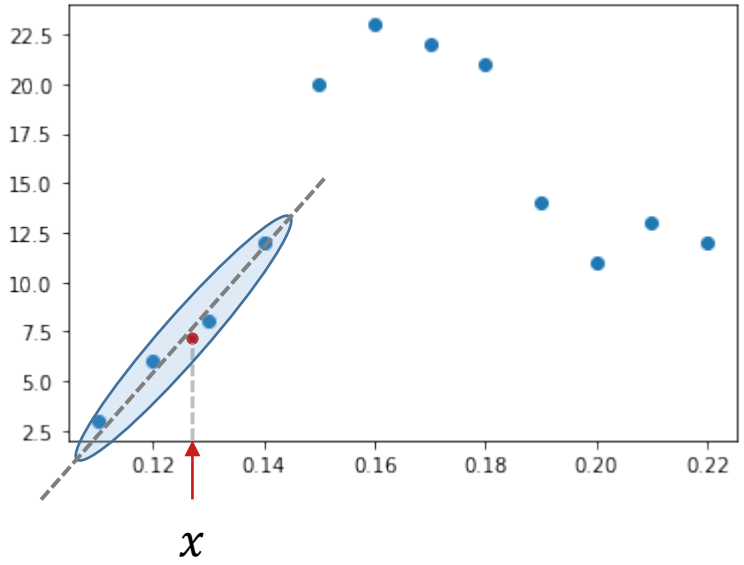
# Locally Weighted Regression

# Locally Weighted Regression

- The prediction problem of linear regression:
  - Input: a query data point $x$
  - Output: the target value $\hat{y} = w^T x$

- If we fit a linear regression model on the full dataset, the estimation error might be high

- If we fit a linear regression model from the nearby data points, the error might be lower



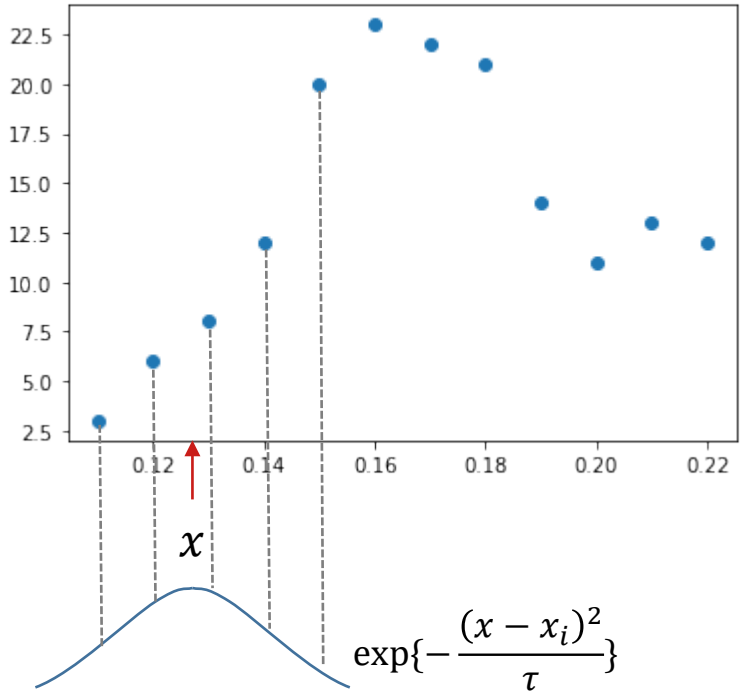
# Locally Weighted Regression

- Least square error for linear regression:
  - Find $w$ to minimize $\frac{1}{2}\sum_{i=1}^{n}(w^T x_i - y_i)^2$

- Locally Weighted Regression:
  - Give higher weights to the nearby examples
  - Find $w$ to minimize $\frac{1}{2}\sum_{i=1}^{n}\theta_i(w^T x_i - y_i)^2$, where $\theta_i$ is the weight for the i-th example in the dataset.
  - $\theta_i$ is often set as a Gaussian-like function:

$$\theta_i = \exp\{-\frac{(x - x_i)^2}{\tau}\}$$

  - The closer $x_i$ is to the query $x$, the higher $\theta_i$
  - The more distant $x_i$ is to the query $x$, the lower $\theta_i$




$\tau$ is a hyperparameter that you can set with validation error

# Locally Weighted Regression

- Locally weighted regression is an instance-based learning method
  - We will discuss more instance-based learning in Week 8.

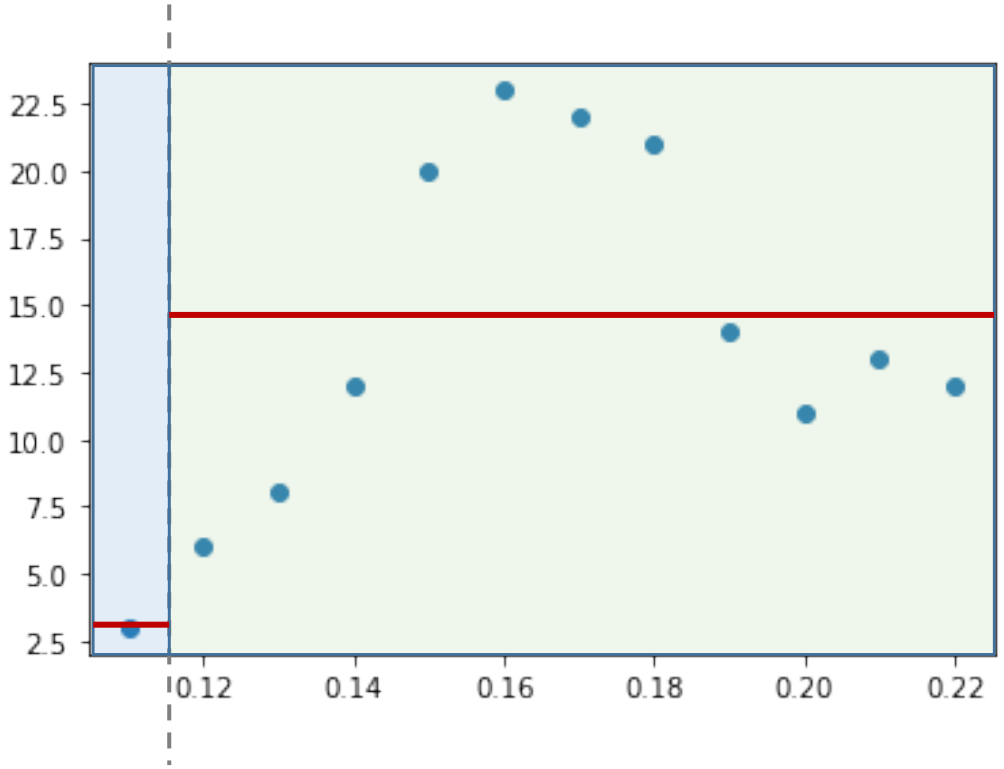
- This method needs to fit a linear regression model for an arbitrary input data point. The training is done in the test phase. This might be time consuming

- This method is called "Non-parametric" in machine learning:
  - Parametric model – The number of parameters is fixed w.r.t. the size of datasets
  - Non-parametric model – The number of parameters grow with the size of datasets

# Regression Trees

# Regression Trees

- Regression tree is similar to decision tree:
  - Each time, we select an axis to split the space based on some criteria (e.g., squared error)
  - On the leaf nodes, use the mean of the data points or fit a linear regressor

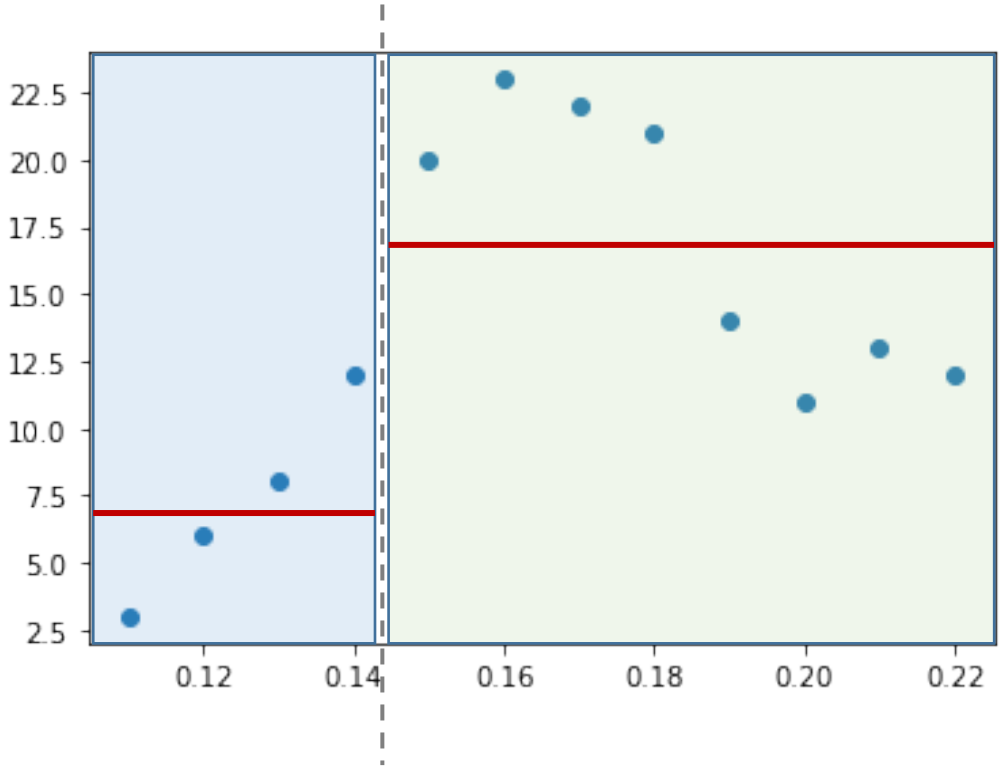



Splitting at $x > 0.11$

- Square error for the left part: 0
- Square error for the right part: 342.18

# Regression Trees

- Regression tree is similar to decision tree:
  - Each time, we select an axis to split the space based on some criteria (e.g., squared error)
  - On the leaf nodes, use the mean of the data points or fit a linear regressor




Splitting at $x > 0.11$

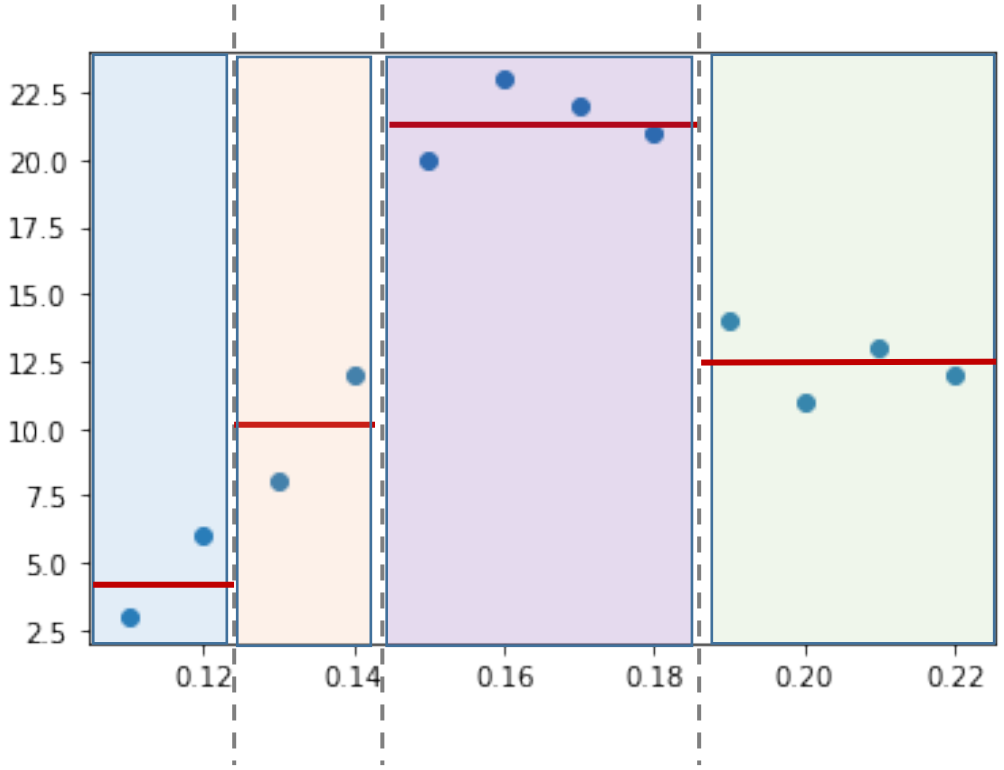
- Square error for the left part: 0
- Square error for the right part: 342.18

Splitting at $x > 0.14$

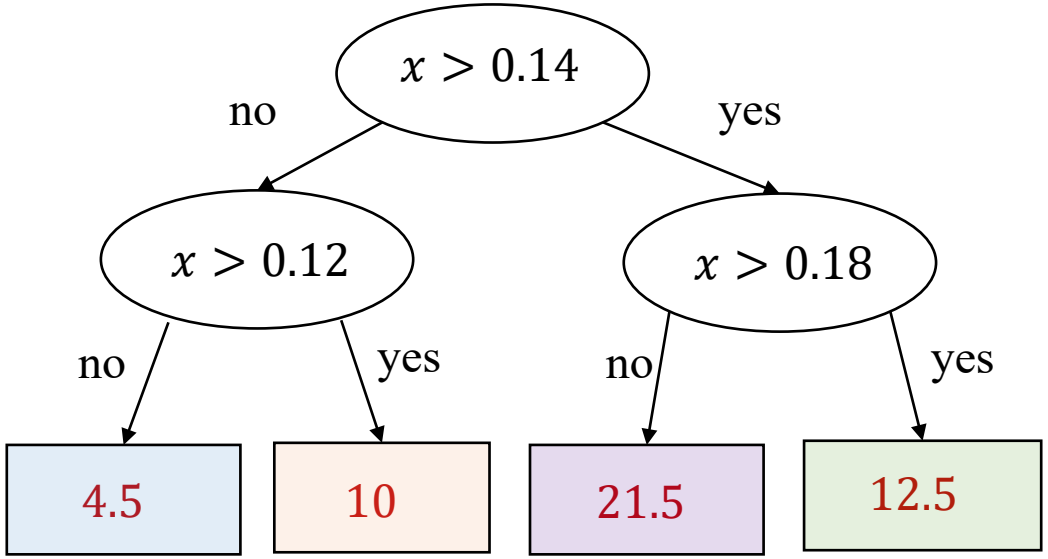
- Square error for the left part: 42.75
- Square error for the right part: 172

# Regression Trees

- Regression tree is similar to decision tree:
  - Each time, we select an axis to split the space based on some criteria (e.g., squared error)
  - On the leaf nodes, use the mean of the data points or fit a linear regressor




We can split further on the two parts



# Summary

- Regression considers the case of a numerical $y_i$
- Least squares is a classic method for fitting linear models
- With 1 feature, it has a simple closed-form solution
- Can be generalized to $d$ features
  - What does the regression look like in 2 dimensions?
- Non-linear data
  - Locally weighted regression
  - Regression trees