BIOS 7747: Machine Learning for Biomedical Applications

Supervised learning: regression

Antonio R. Porras (antonio.porras@cuanschutz.edu)

Department of Biostatistics and Informatics
Colorado School of Public Health
University of Colorado Anschutz Medical Campus

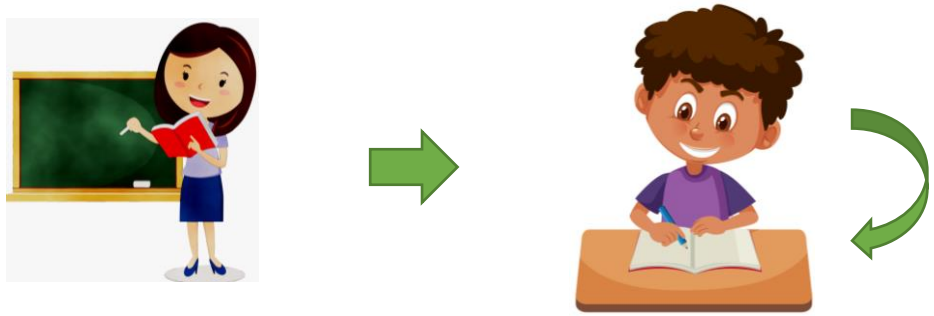
Outline

- Supervised learning: regression
- Gradient descent optimization to solve regression problems
- Linear regression
- Need for non-linear regression models with examples

Supervised learning

- Supervised learning
  - Learning from a dataset with known labels or outcomes

- Assumptions
  - The training dataset contains the “right” answers.
  - The right answers can be obtained from the available data



Training dataset
(the teacher)

(x, y)

Model
(the student)

$f?$

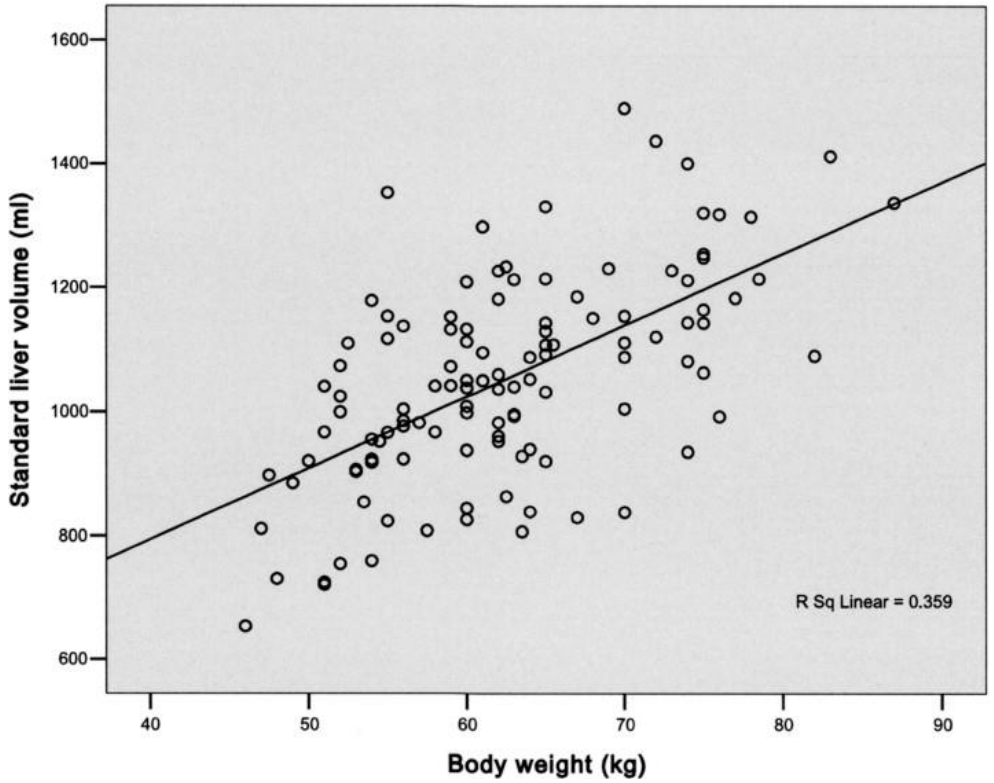
Supervised learning: Regression

- What is a (mathematical) model?
  - Set of **variables** and their **relationships** expressed using a mathematical language (equations)
    - **Variables**: known (input variables or features) vs. predicted (output variables or predictions)
    - **Relationships**: linear vs. non-linear
  - Regression models predict **continuous** variables
    - Note: known or input variables are not necessarily continuous.

Regression

- Example: standard liver volume for transplantation (SLW)

SLW = 218 + 12.3 x bodyWeight



[S.C. Chan, C.L. Liu, C.M. Lo, *et al.*Estimating liver weight of adults by body weight and gender. World J Gastroenterol, 12 (2006), p. 2217]

Regression

- How do we define a regression model?

  - Which are the input variables?
    Input variable: x

  - What is the predicted variable?
    Output variable: y

  - What types of relationship are there between variables?
    A hypothesis about relationships between variables is needed: f
    Most common: relationships are **linear** and variables are **independent**

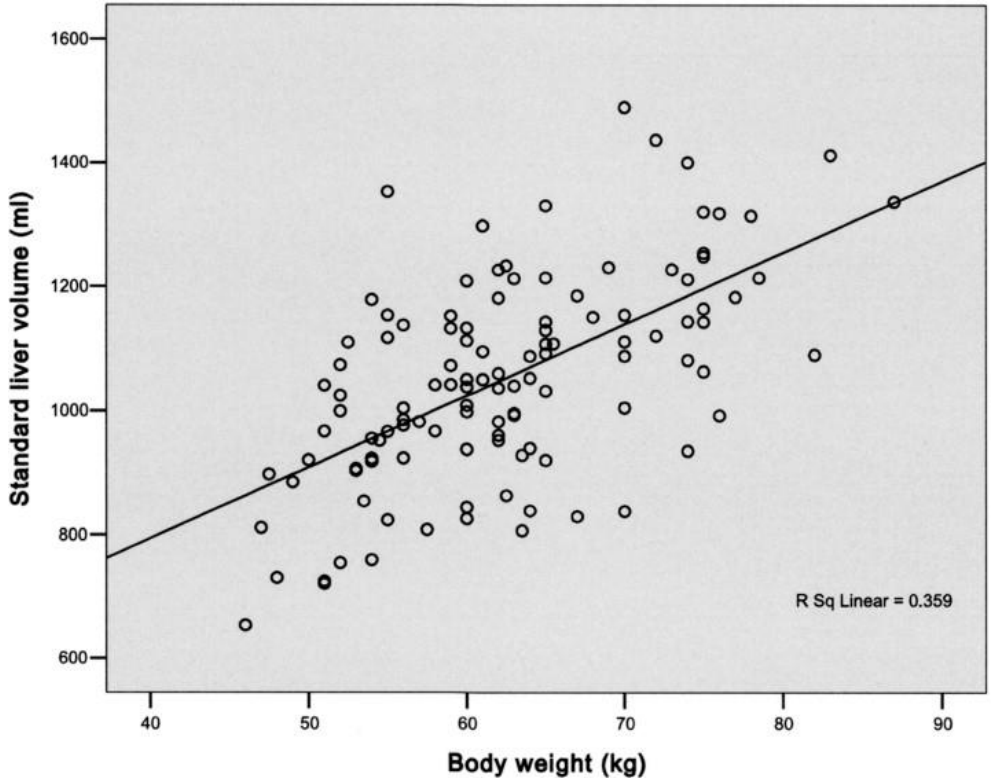
  - Example of linear regression with a single variable: $f(x) = \theta_o + \theta_1 x$
    - Parameters: $\boldsymbol{\theta} = \{\theta_0, \theta_1\}$

  - Example of linear regression with multiple variables: $f(\boldsymbol{x}) = \theta_o + \theta_1 x_0 + \theta_2 x_1 + \cdots + \theta_M x_{M-1}$
    - M independent variables: $\boldsymbol{x} = \{x_0, \ldots, x_{M-1}\}$
    - $M + 1$ parameters: $\boldsymbol{\theta} = \{\theta_0, \ldots, \theta_{M+1}\}$
    - Linear regression for M independent variables: $f(\boldsymbol{x}) = \theta_0 + \sum_{p=0}^{M-1} \theta_{p+1} x_p$

Regression

- Example: standard liver volume for transplantation (SLW)

SLW = 218 + 12.3 x bodyWeight + 51 x gender



[S.C. Chan, C.L. Liu, C.M. Lo, *et al.*Estimating liver weight of adults by body weight and gender. World J Gastroenterol, 12 (2006), p. 2217]

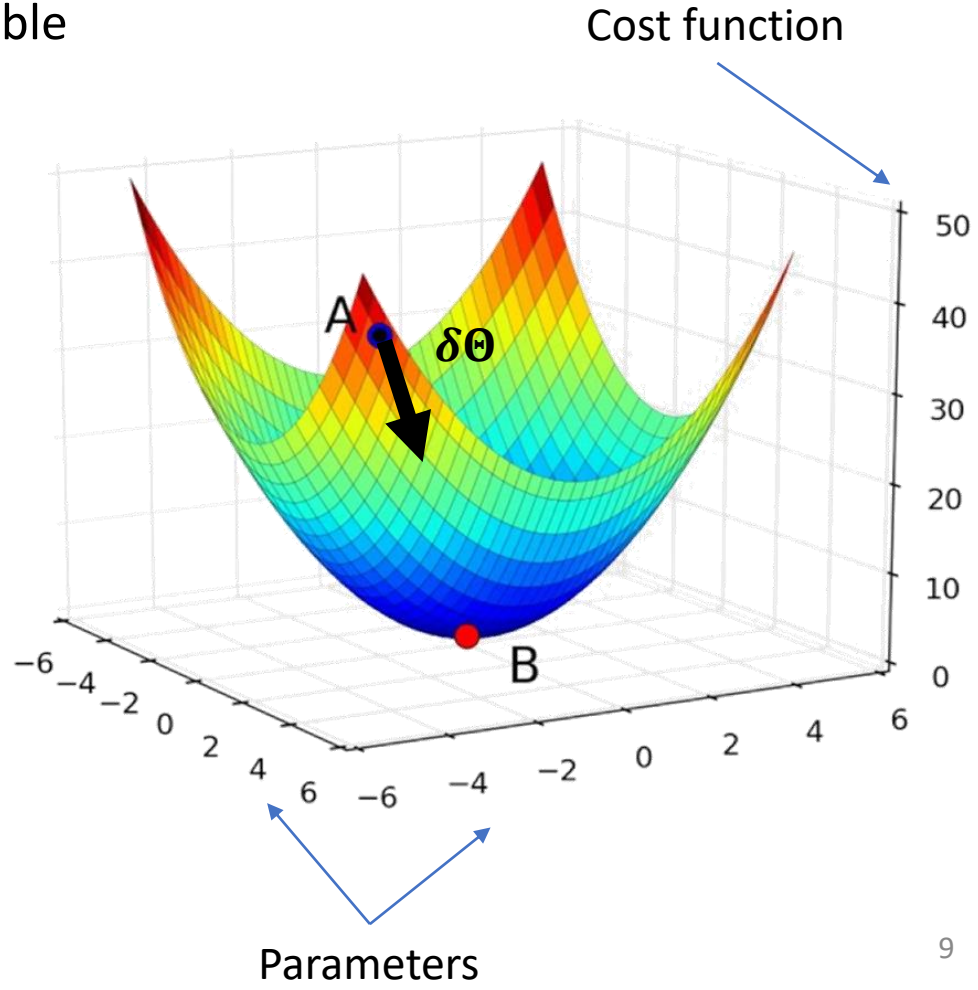
Regression

- How do we train a regression model?
  - Training dataset
    - Notation:
      - N: number of training samples
      - M: number of independent variables (features)
      - Training sample pair: $(\boldsymbol{x}, y)$
      - Training sample with index i: $\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)$
  - Learning goal: $f(\boldsymbol{x}) \rightarrow y$?
    - We know $(\boldsymbol{x}, y)$ for the training dataset, how do we estimate $\hat{y} = f(\boldsymbol{x})$ for new samples?
    - We hypothesize that $f(\boldsymbol{x}) \approx y$ in the training dataset [Assumption!!]
    - Model parameters: $\boldsymbol{\theta} = \arg\min \underbrace{\sum_{i=0}^{N-1}\left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)}\right)^2}_{\text{Cost function } J(\boldsymbol{\theta})}$

Regression

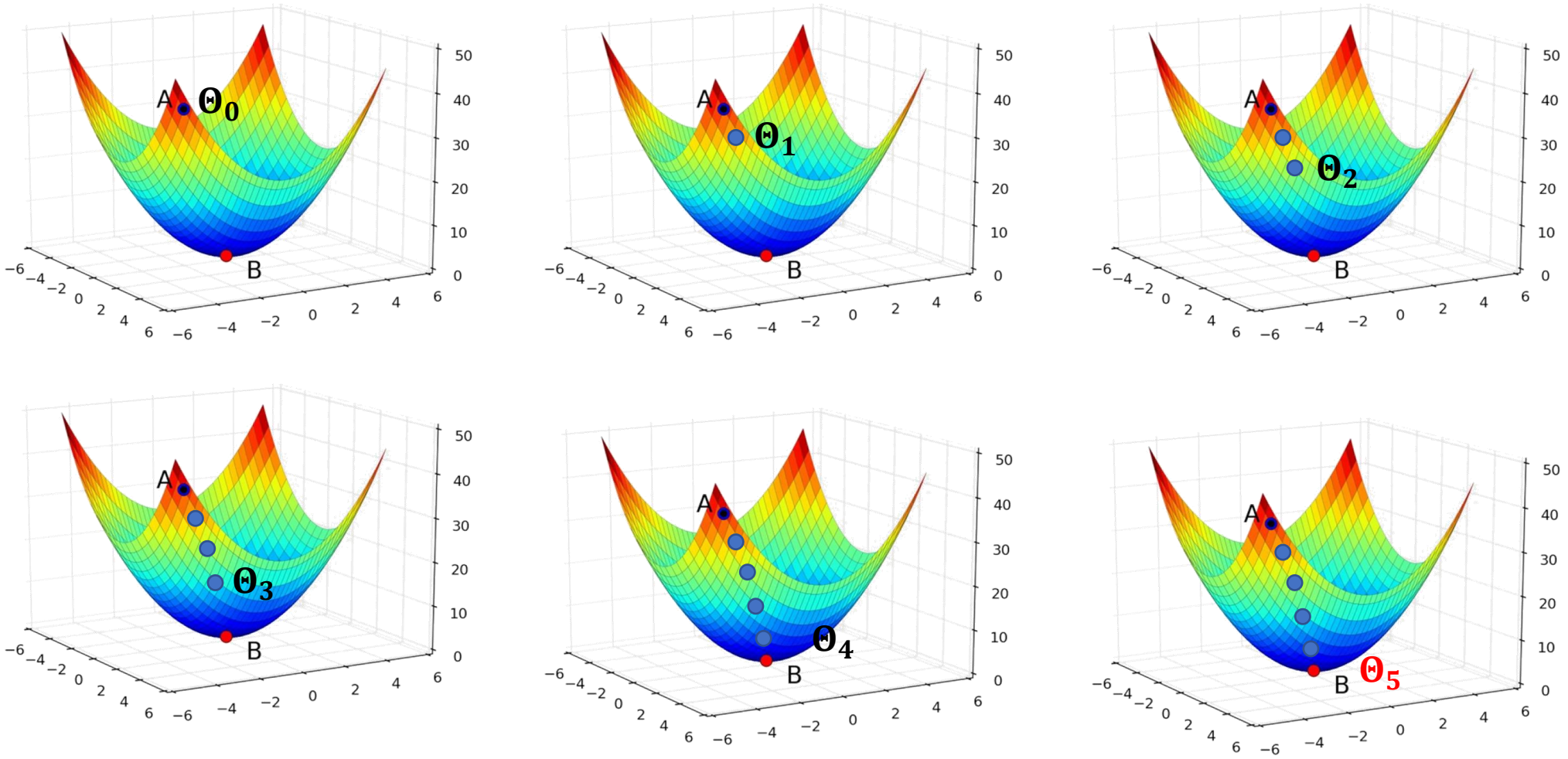
- How do we train a regression model?
  - Exhaustive parameter search is normally not feasible
  - Gradient descent optimization
    1. Set an initial value for $\boldsymbol{\theta}$
       - Based on prior hypothesis
       - $\boldsymbol{\theta_0} = \vec{0}$ is common when no prior hypotheses
    2. Do until convergence:
       - $\boldsymbol{\theta_{t+1}} = \boldsymbol{\theta_t} + \boldsymbol{\delta\theta_t}$ so $J(\boldsymbol{\theta_{t+1}}) < J(\boldsymbol{\theta_t})$

Step:
- Direction
- Magnitude



Gradient descent optimization

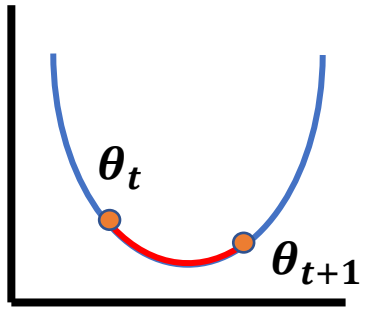
- Gradient descent algorithm



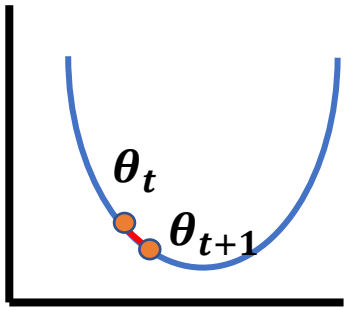
Gradient descent optimization

- Gradient descent algorithm: steps
  - $\frac{\partial}{\partial \Theta_t} J(\Theta_t)$ provides the direction and magnitude of change, but the magnitude converges to zero as the parameters approach their optimal value.
  - Introduction of a learning rate: $\theta_{t+1} = \theta_t + \delta\theta_t = \theta_t - \alpha \frac{\partial}{\partial \theta_t} J(\theta_t)$

Large steps:
- Fewer iterations to converge
- May never reach optimal solution



Small steps:
- More iterations to converge
- Often gets closer to optimal solution (in convex problems)



Note: In non-linear, non-convex regression problems, small steps will be more likely to get caught in local minima. But large steps may get lost in the parameter space and produce unstable behaviors

Gradient descent optimization

- Gradient descent algorithm: updating parameters

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \frac{\partial}{\partial \boldsymbol{\theta}_t} J(\boldsymbol{\theta}_t) \longrightarrow \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)} \right)^2$$

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \left(2 \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)} \right) \frac{\partial}{\partial \boldsymbol{\theta}_t} f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) \right)$$

$$\text{where } f(\boldsymbol{x}; \boldsymbol{\Theta}) = \theta_o + \theta_1 x_0 + \theta_2 x_1 + \cdots + \theta_M x_{M-1}$$

- In practice, we define the cost function as $J(\boldsymbol{\Theta}) = \frac{1}{2} \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\Theta}\right) - y^{(i)} \right)^2$ to eliminate the constant in the derivation:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)} \right) \frac{\partial}{\partial \boldsymbol{\theta}_t} f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right)$$

Gradient descent optimization

- Gradient descent algorithm: updating parameters

$$\theta_0 \leftarrow \theta_0 - \alpha \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)} \right)$$

$$\theta_1 \leftarrow \theta_1 - \alpha \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)} \right) \boldsymbol{x}_0^{(i)}$$

$$\theta_2 \leftarrow \theta_2 - \alpha \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)} \right) \boldsymbol{x}_1^{(i)}$$

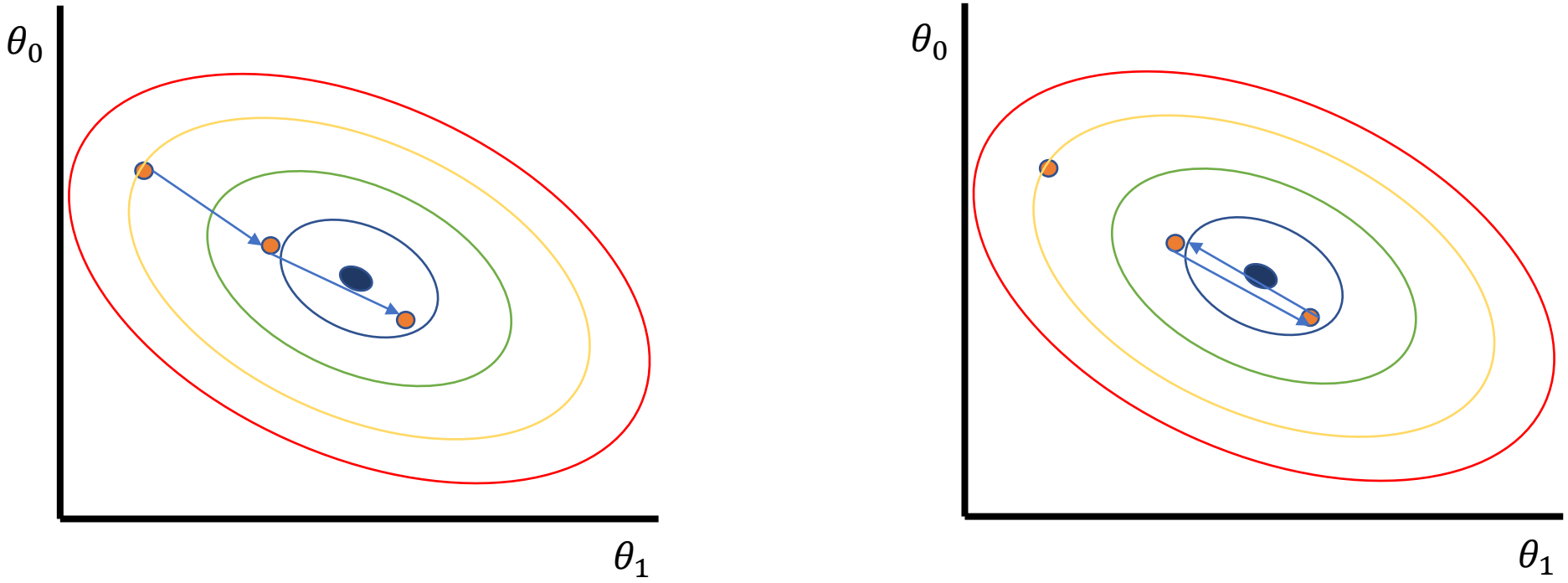
.
.
.

Note: the values to update each parameter only depend on one single input variable or feature

Note: only need to be calculated once!

Gradient descent optimization

- Gradient descent algorithm: when to stop?



Stop criteria:

- Cost function does not improve more than a specific threshold after a fixed number of consecutive iterations.
- Fixed number of iterations

Gradient descent optimization

- Regular gradient descent algorithm:

  - What if we our dataset is “too” large?
    - Can’t load everything on memory simultaneously: inefficient I/O bottleneck problem
    - “Too many” evaluations are required to calculate the gradient at one single iteration

- Stochastic gradient descent algorithm

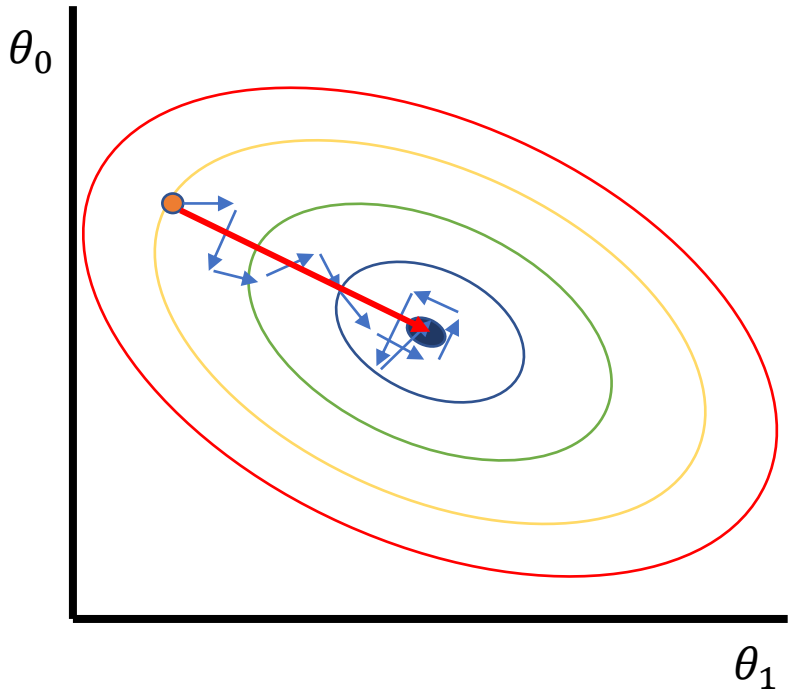
  - Gradients are only calculated on a reduced number of samples grouped in batches
    - Gradient calculations are faster
    - Fewer evaluations are needed for gradient “estimation” (the gradient is only approximated)

Gradient descent optimization

- Stochastic gradient descent algorithm: adding or averaging?

Real gradient

Batch gradient



Large batch size

- Batch gradient approximates the real gradient
- Higher trust in the direction potentially allows for higher learning rates

Small batch size

- Batch gradient may not represent the real gradient well
- Lower trust in the direction usually allow for smaller learning rates

Gradient descent optimization

- Stochastic gradient descent algorithm: adding or averaging?

Addition: $J(\boldsymbol{\Theta}) = \frac{1}{2}\sum_{i=0}^{N-1}\left(f\left(\boldsymbol{x}^{(i)};\boldsymbol{\Theta}\right) - y^{(i)}\right)^2$

Magnitude of cost function and its gradient depends on the number of samples

Simpler implementation of regular gradient descent algorithm

Average: $J(\boldsymbol{\Theta}) = \frac{1}{2N}\sum_{i=0}^{N-1}\left(f\left(\boldsymbol{x}^{(i)};\boldsymbol{\Theta}\right) - y^{(i)}\right)^2$

Magnitude of cost function and its gradient is independent from the number of samples

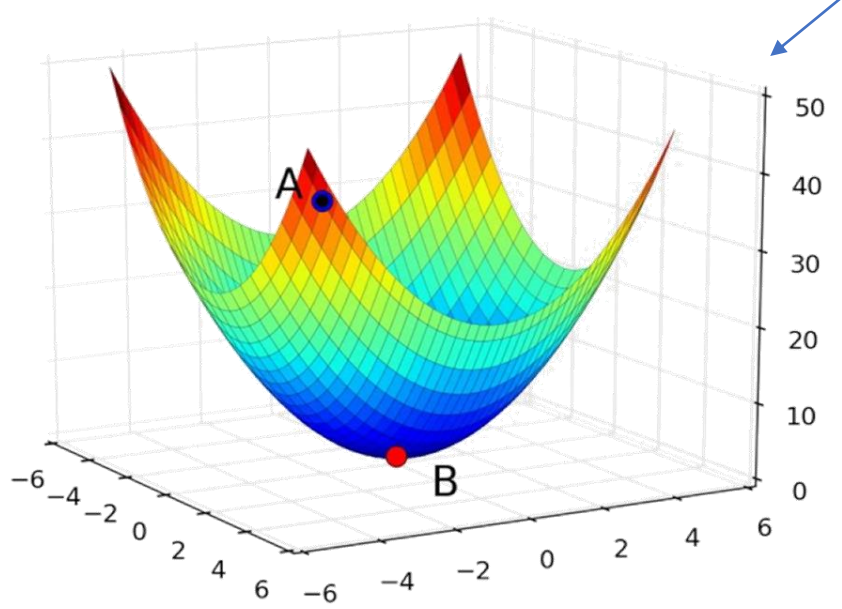
Provides stability when evaluating different batch sizes

But careful with different number of samples between batches!!!!

Linear regression

- **Linear** regression has an analytical solution (if computationally feasible)

$$\boldsymbol{\theta} = \arg\min J(\boldsymbol{\theta}) = \arg\min \frac{1}{2}\sum_{i=0}^{N-1}\left(f\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right) - y^{(i)}\right)^2$$



In linear regression, $J(\boldsymbol{\theta})$ is convex so:

(1) The only local minimum is the global minimum

(2) At the minimum: $\frac{\partial}{\partial\boldsymbol{\theta}}J(\boldsymbol{\theta}) = \frac{\partial}{\partial\boldsymbol{\theta}}\left(\frac{1}{2}(\boldsymbol{X\theta}-\mathbf{y})^T(\boldsymbol{X\theta}-\mathbf{y})\right) = 0$

$$\boldsymbol{X} = \begin{pmatrix} 1 & x_0^{(0)} & \dots & x_{M-1}^{(0)} \\ \vdots & & & \vdots \\ 1 & x_0^{(N-1)} & \dots & x_{M-1}^{(N-1)} \end{pmatrix} \qquad \boldsymbol{y} = \begin{pmatrix} y_0 \\ \vdots \\ y_{N-1} \end{pmatrix}$$

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_0 \\ \vdots \\ \theta_M \end{pmatrix}$$

Linear regression

- **Linear** regression has an analytical solution (if computationally feasible)

$$\frac{\partial}{\partial \boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} \left(\frac{1}{2} (\boldsymbol{X\theta} - \boldsymbol{y})^T (\boldsymbol{X\theta} - \boldsymbol{y}) \right) = \mathbf{0}$$

$$\frac{\partial}{\partial \boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{2} \frac{\partial}{\partial \boldsymbol{\theta}} \left((\boldsymbol{\theta}^T \boldsymbol{X}^T - \boldsymbol{y}^T) (\boldsymbol{X\theta} - \boldsymbol{y}) \right) = \mathbf{0}$$

$$\frac{\partial}{\partial \boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{2} \frac{\partial}{\partial \boldsymbol{\theta}} (\boldsymbol{\theta}^T \boldsymbol{X}^T \boldsymbol{X\theta} - \boldsymbol{\theta}^T \boldsymbol{X}^T y - \boldsymbol{y}^T \boldsymbol{X\theta} + \boldsymbol{y}^T \boldsymbol{y}) = \mathbf{0}$$

$$\frac{\partial}{\partial \boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{2} [\boldsymbol{X}^T \boldsymbol{X\theta} + \boldsymbol{X}^T \boldsymbol{X\theta} - \boldsymbol{X}^T y - \boldsymbol{X}^T y] = 0$$

$$\frac{\partial}{\partial \boldsymbol{\theta}} J(\boldsymbol{\theta}) = \boldsymbol{X}^T \boldsymbol{X\theta} - \boldsymbol{X}^T y = 0$$

$\boldsymbol{X\theta} = \boldsymbol{y}$

Normal equation

$\boldsymbol{X}^T \boldsymbol{X\theta} = \boldsymbol{X}^T y$

$\boldsymbol{\theta} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{y}$

Pseudo-inverse of X!

Linear regression

- Assumptions in **linear** regression:
  - Linearity
  - Independence of predictors (no multicollinearity)
  - Homoscedasticity: error (residual) variance does not depend on the independent variables
  - Data are normally distributed

Linear regression

- Evaluation of a regression model:

Residuals

- Mean absolute error: $\frac{1}{N}\sum_0^{N-1}\left|f\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right)-y^{(i)}\right|$
- Mean squared error (MSE) : $\frac{1}{N}\sum_0^{N-1}\left(f\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right)-y^{(i)}\right)^2$ → Penalizes large errors or residuals
- Coefficient of determination: $R^2 = 1 - \frac{\sum_0^{N-1}\left(f\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right)-y^{(i)}\right)^2}{\sum_0^{N-1}\left(y^{(i)}-\bar{y}\right)^2}$ → MSE; → Variance

$R^2 = 1$: Good model
$R^2 = 0$: Model is not better than mean value
$R^2 < 0$: Worse predictor than mean value

No universal rule for interpretation

Does not decrease when adding features

- Adjusted R^2: $R^2_{adjusted} = \frac{\left(1-R^2\right)(N-1)}{N-M-1}$ → Better suited for model comparison

Penalized when adding more features

Linear regression

- Statistical significance in linear regression:

  - T-test on individual coefficients:
    - Coefficients: $H_0: \theta_p = 0$ (i.e., the coefficient has no significant effect)
    - Not meaningful for the intercept

  - F-test on regression model:
    $H_0: \theta_p = 0 \; \forall p > 0$ (i.e., the regression model is not significantly better than the average value or intercept)

    Is it really meaningful?

Linear regression

- Confidence and prediction intervals:

  - Confidence interval of the coefficients:

Note: For the intercept, $x_0^{(i)} = 1$

$$\theta_p \pm t_{\alpha/2,N-M-1} * \sigma_{\theta_p}$$

Since: $\boldsymbol{\theta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{Y}$, then $\boldsymbol{\sigma_\theta} = \sqrt{\text{diag}\{\sigma_y^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}\}}$

where $\sigma_y^2 = \frac{\sum_{\forall i}\left(y^{(i)} - \hat{y}^{(i)}\right)^2}{\text{N}-\text{M}-1}$ is the variance of the residuals

  - Confidence interval for the prediction given $\boldsymbol{x}^{(i)}$:

$$\hat{y} \pm t_{\alpha/2,N-M-1} * \hat{\sigma}$$

where $\hat{\sigma} = \sqrt{\sigma_y^2 \boldsymbol{x}^{(i)^T}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}^{(i)}}$

Linear regression

- Other measurements:

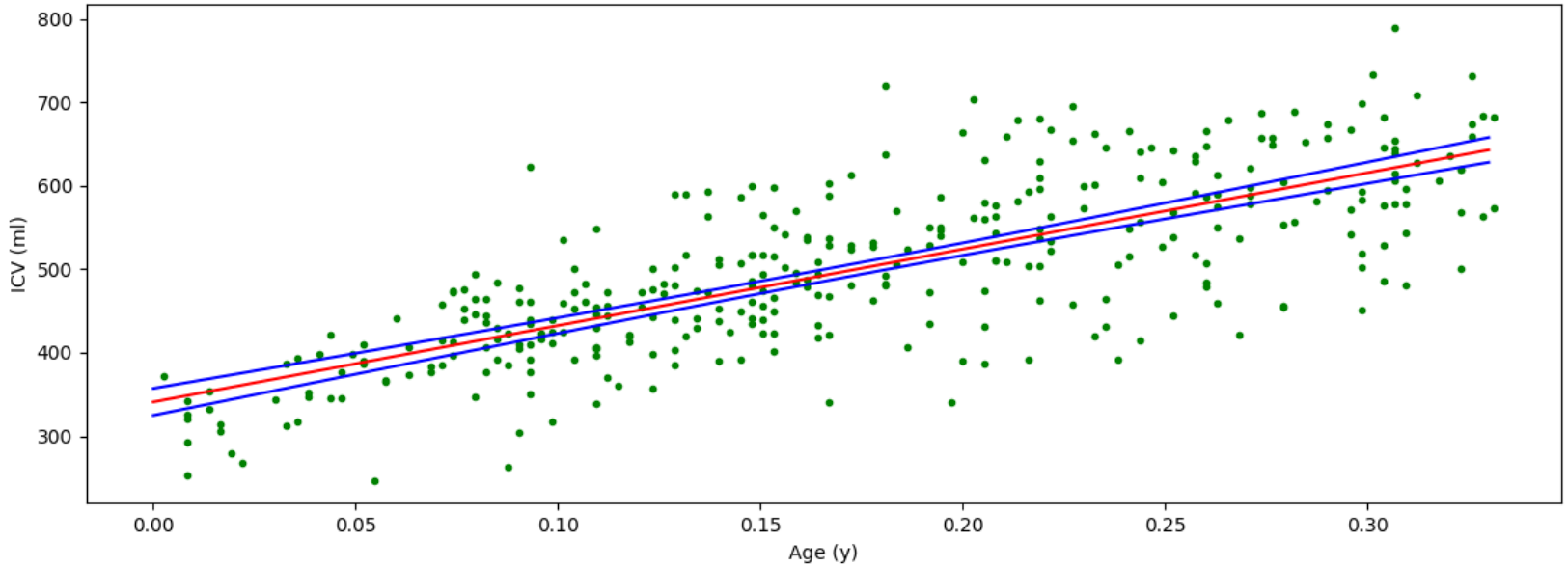
  - Condition number: measures the sensitivity of the coefficient estimates to small changes in the data matrix.

$$k(X) = \|X\|\|X^{-1}\| = \frac{eig_{max}(X)}{eig_{min}(X)}$$

  - Large numbers are related to ill-conditioned problems.
  - Although thresholds have been proposed (e.g., 20, 30), this number varies when centering data or when the problem dimension change.

Linear regression

- Example: intra-cranial volume vs. age (age range 0-4 months)



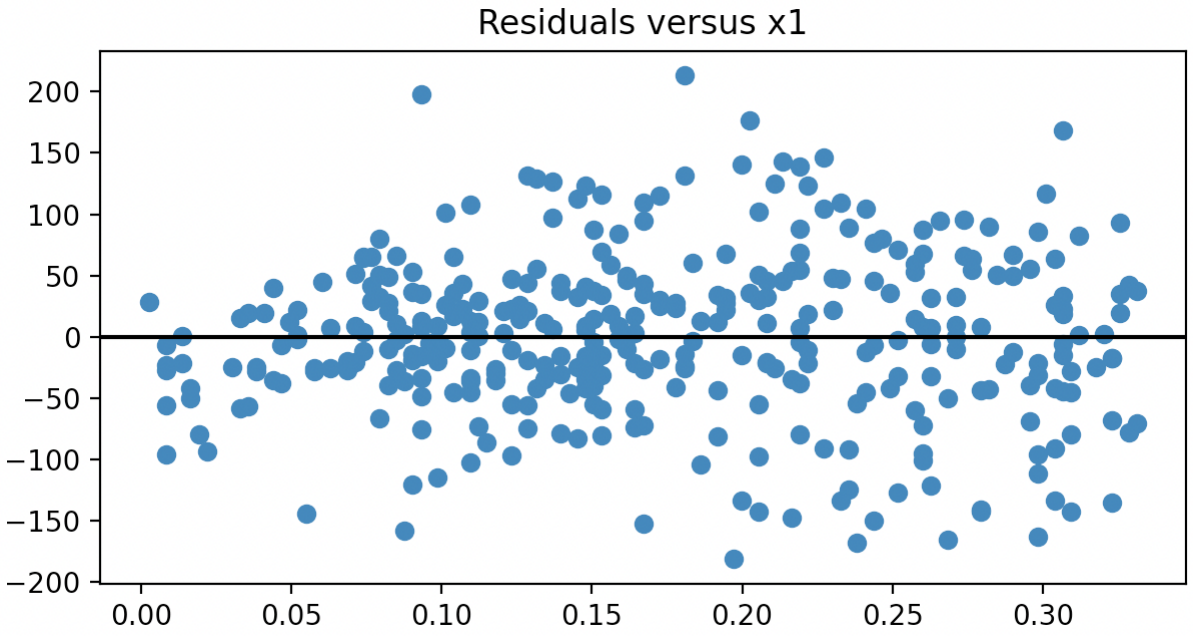
R-squared: 0.572
Adj. R-squared: **0.570**
F-statistic: 461.5
Prob (F-statistic): **1.19e65**

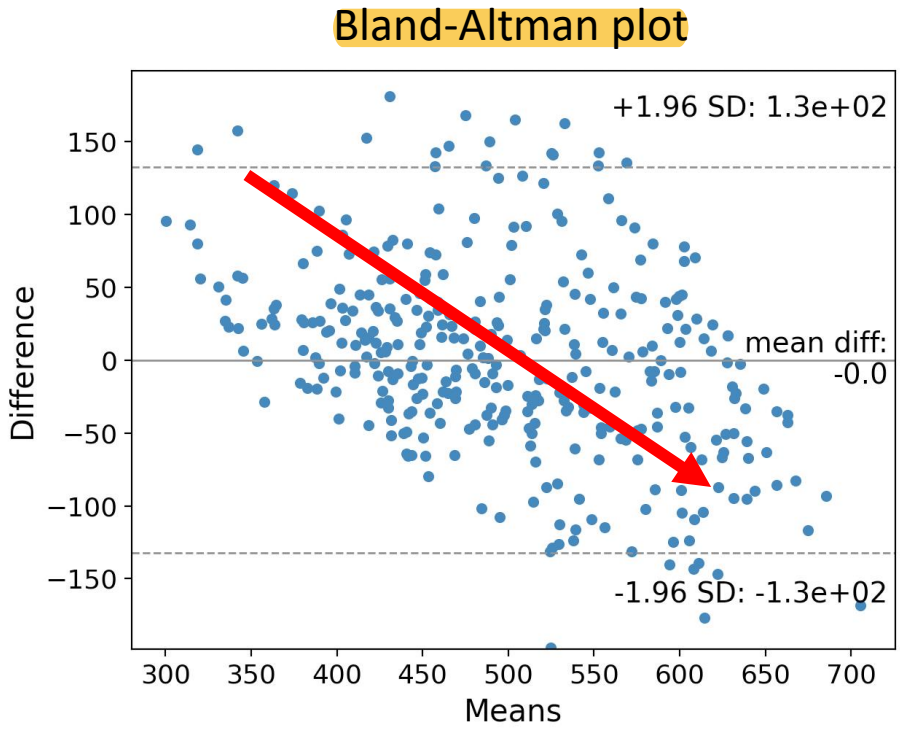
	coef	std err	t	P>|t|	[0.025	0.975]
const	340.8132	8.235	41.384	0.000	324.616	357.011
x1	915.3257	42.608	21.483	0.000	831.523	999.129

Cond. No. **12.1**

Linear regression

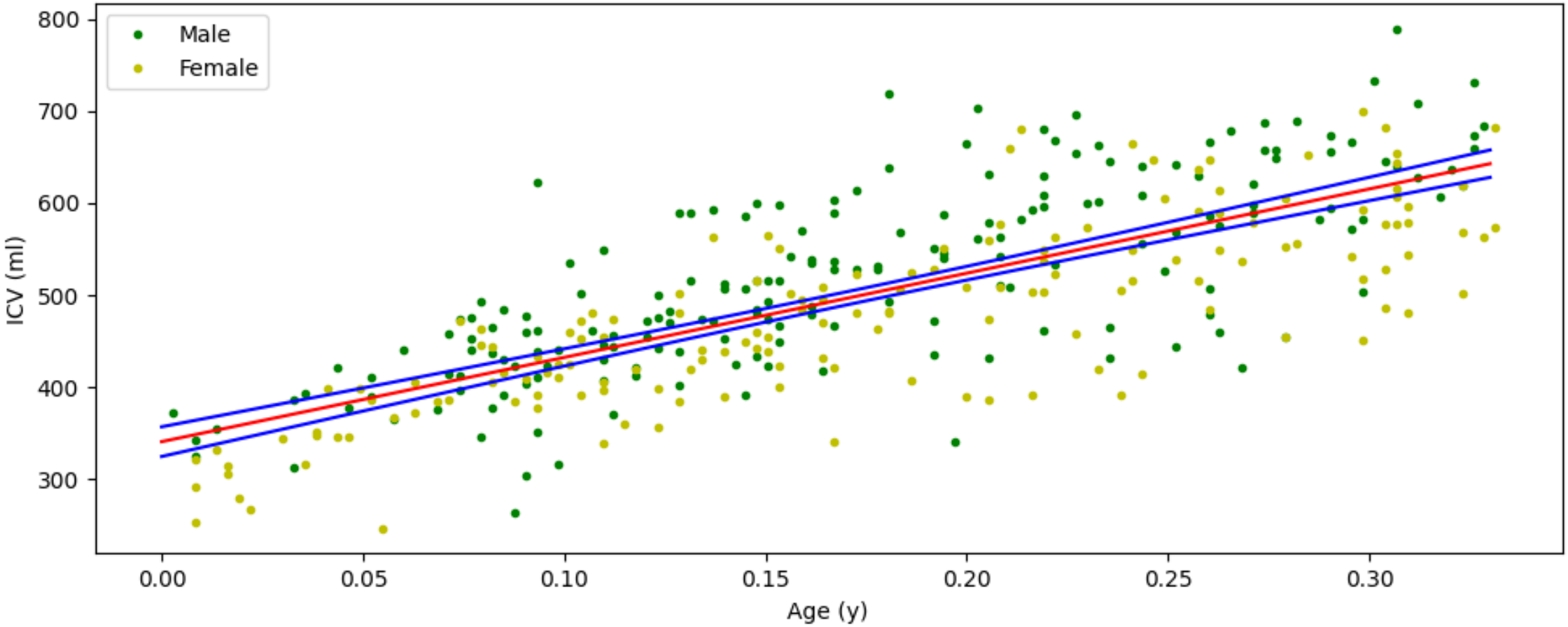
- Example: intra-cranial volume vs. age (age range 0-4 months)





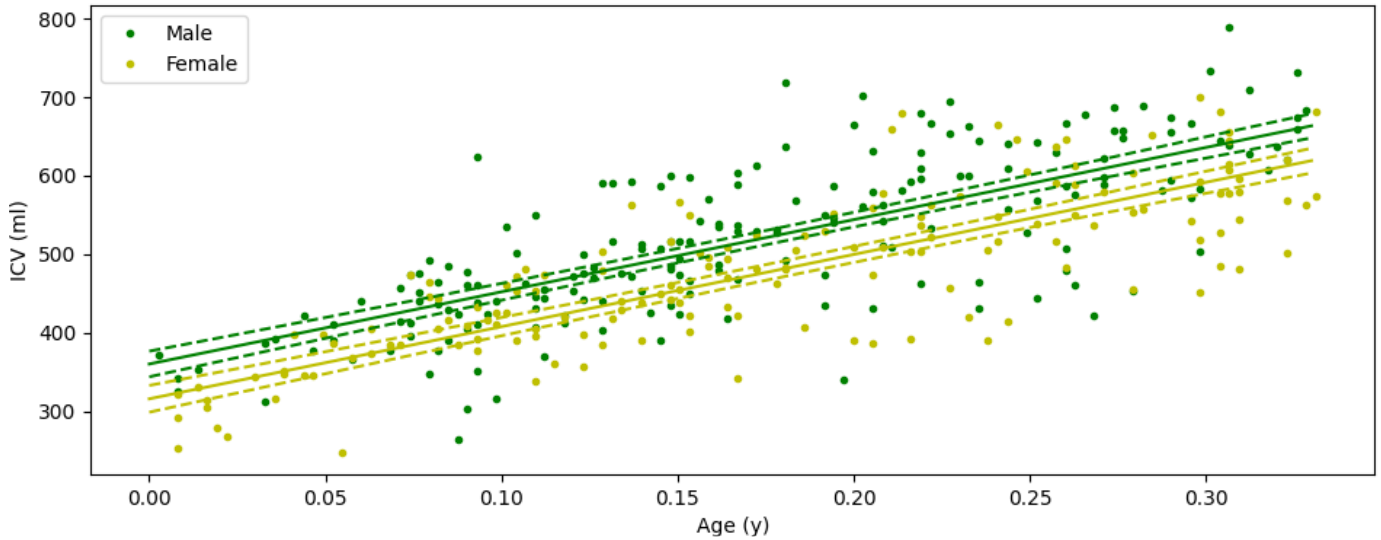
Linear regression

- Example: intra-cranial volume vs. age (age range 0-4 months)



Linear regression

- Example: intra-cranial volume vs. age (age range 0-4 months)



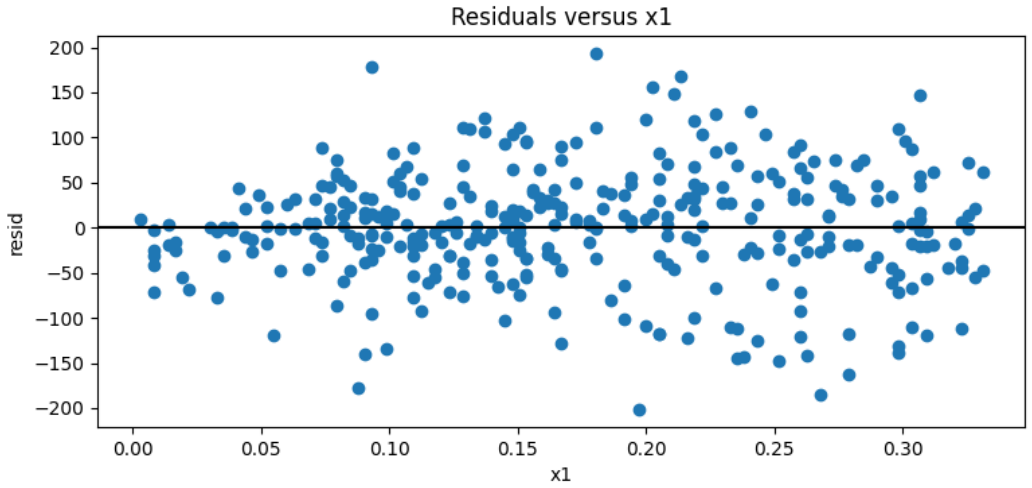
R-squared: 0.617
Adj. R-squared: **0.615**
F-statistic: 278.3
Prob (F-statistic): **1.10e-72**

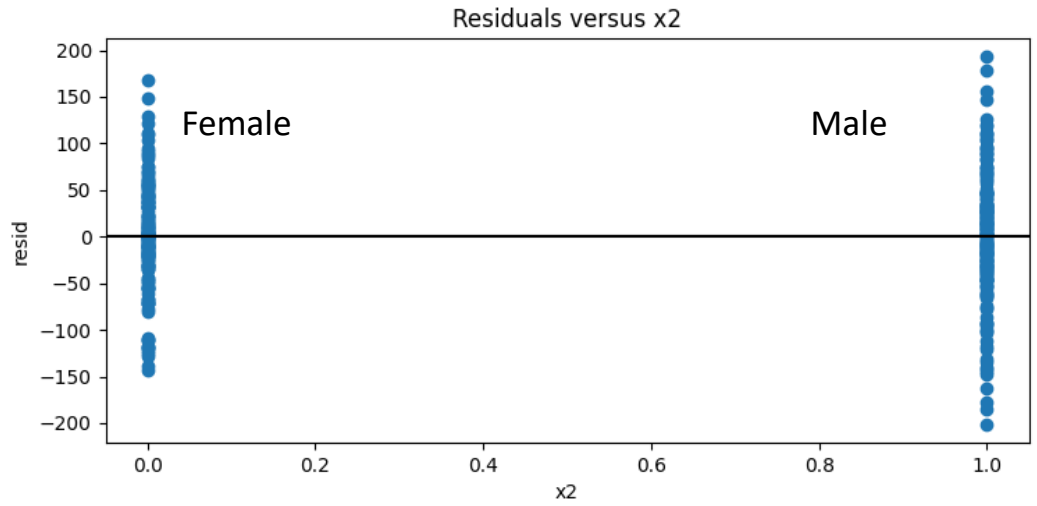
		coef	std err	t	P>|t|	[0.025	0.975]
	const	315.7497	8.716	36.228	0.000	298.607	332.892
Age	x1	919.4257	40.330	22.798	0.000	840.103	998.749
Sex	x2	44.3694	6.905	6.426	0.000	30.788	57.950

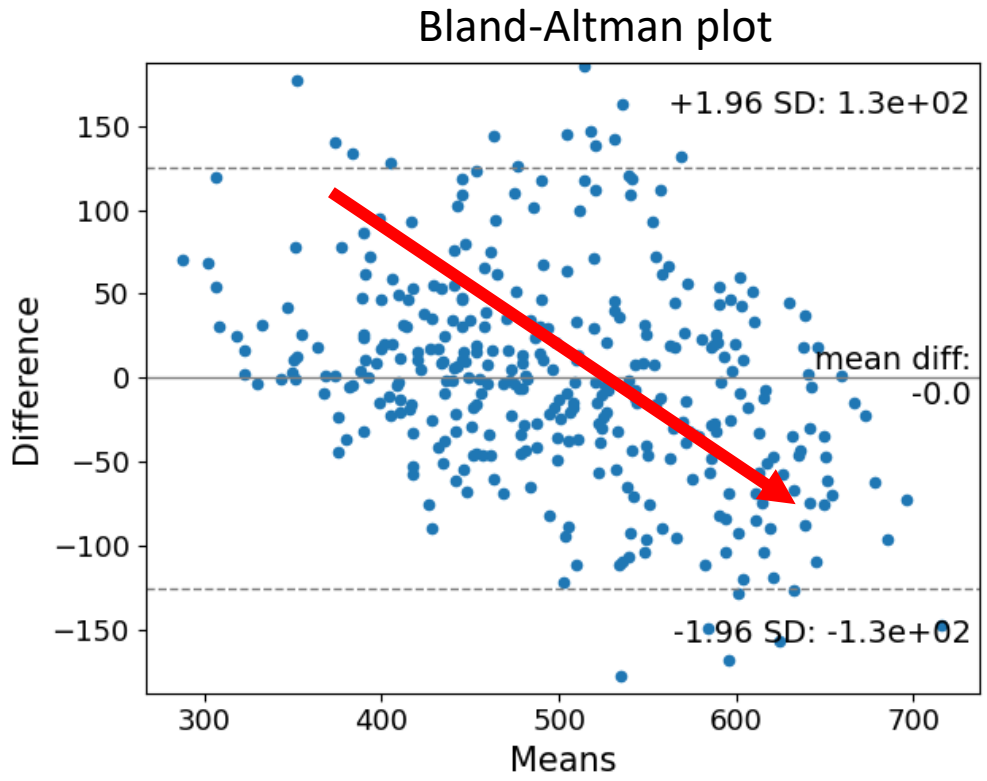
Cond. No. **14.1**

Linear regression

- Example: intra-cranial volume vs. age (age range 0-4 months)

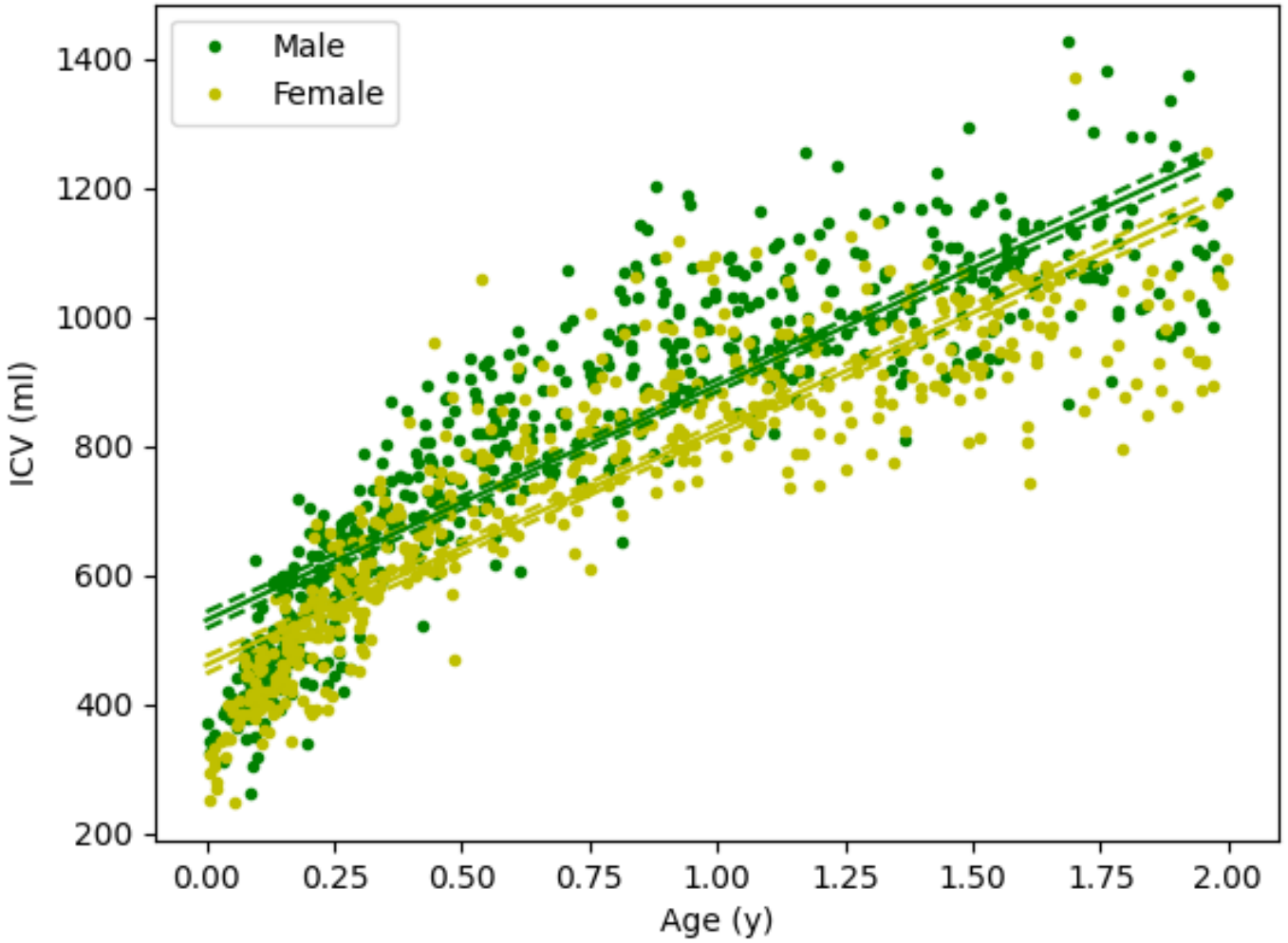






Linear regression

- Example: intra-cranial volume vs. age (age range 0-2 years)



R-squared: 0.769
Adj. R-squared: **0.768**
F-statistic: 1832.
Prob (F-statistic): **0.00**

	coef	std err	t	P>|t|	[0.025	0.975]
const	460.7480	6.839	67.369	0.000	447.329	474.167
x1	363.8412	6.090	59.744	0.000	351.892	375.790
x2	69.4469	6.830	10.167	0.000	56.045	82.849

Cond. No. **3.78**

$$R^2 = 1 - \frac{\sum_0^{N-1}\left(f\left(\boldsymbol{x}^{(i)};\boldsymbol{\theta}\right) - y^{(i)}\right)^2}{\sum_0^{N-1}\left(y^{(i)} - \bar{y}\right)^2}$$

Non-linear regression

- Linear regression is a powerful tool to investigate relationships between independent and dependent variables

- Linear regression most often cannot model accurately relationships between biological processes because they are normally not linear
  - One must be skeptical about performance evaluation metrics and statistical tests

- Non-linear functions often need to be considered:
  - Higher-order polynomials
  - Logarithmic and exponential functions
  - Sinusoidal functions
  - Radial basis functions (Gaussian kernels, splines…)

Non-linear regression

- Gradient descent algorithm: updating parameters

$$\theta_0 \leftarrow \theta_0 - \alpha \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)} \right)$$

$$\theta_1 \leftarrow \theta_1 - \alpha \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)} \right) \boldsymbol{x}_0^{(i)}$$

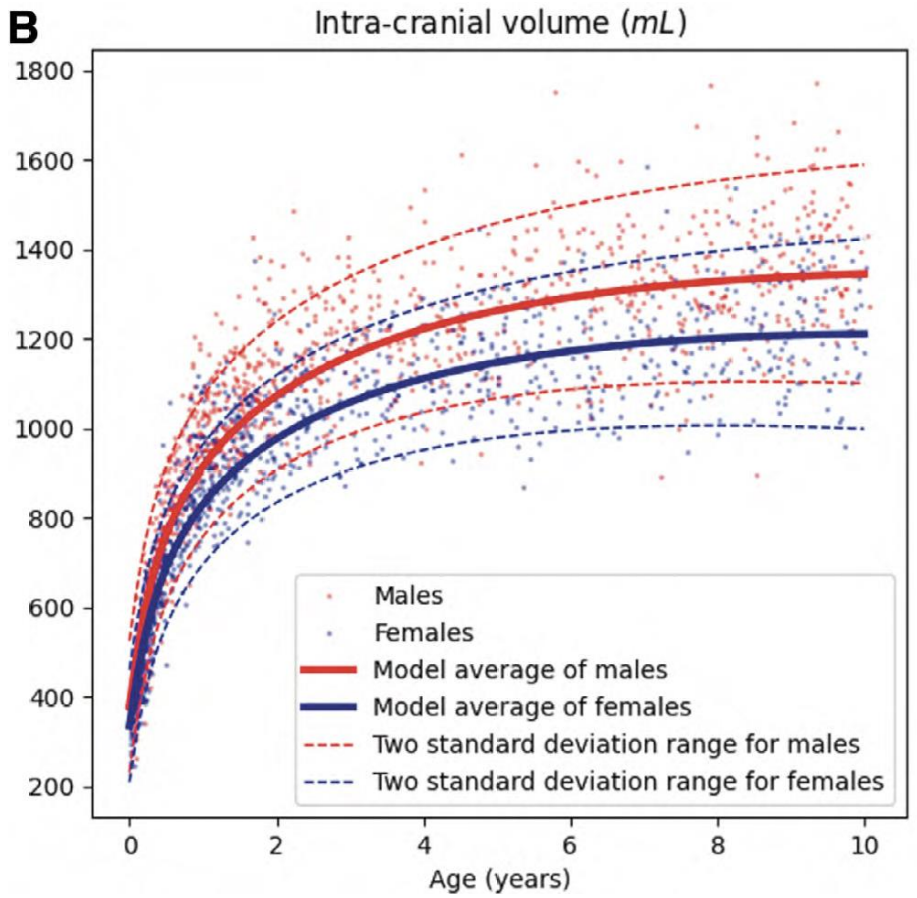
$$\theta_2 \leftarrow \theta_2 - \alpha \sum_{i=0}^{N-1} \left(f\left(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}\right) - y^{(i)} \right) \boldsymbol{x}_1^{(i)}$$

.
.
.

Only change in non-linear regression

Non-linear regression

- Example: intra-cranial volume vs. age (age range 0-2 years)



[Liu et al., *Data-driven Normative Reference of Pediatric Cranial Bone Development. Plast. Reconst. Surg. Global Open, 2022]*

Next class

- Numpy
- Statsmodels
- Curve fit using Scipy