# FIT2086 Lecture 10
# Simulation Based Statistical Methods

Daniel F. Schmidt

Faculty of Information Technology, Monash University

September 30, 2021

## Outline

## Revision from last week (1)

- Machine learning methods
- Cross validation for model selection
  - Withhold data to estimate prediction error
  - $K$-fold CV divides data up into $K$ equal sized groups
  - Train on $K - 1$ folds, predict on the remianing fold
- Decision Trees
  - Split the data up by asking questions of the predictors
  - Number of leaves determines complexity of tree
  - Easy to interpret, flexible
- Methods for learning trees
  - Greedy growing of trees – find best split at each step
  - Backwards pruning of large tree
  - Use CV to select number of leaves in the tree

## Revision from last week (2)

- Trees have low bias, high variance
- One solution: random forests
  - Grow many trees with guided random search
  - Aggregate predictions from the trees
  - Stable, low variance, but loses interpretability
- $k$ nearest neighbours (kNN) methods
  - Assume individuals similar in predictors are similar in targets
  - Find $k$ “most similar” individuals in data to new individual
  - Use their targets to predict target for new individual
- Use CV to select $k$, other tuning parameters

## Outline

## Simulation Based Statistical Methods (1)

- Statistics began as a discipline in early 1910s
- From 1910s through to 1970s, most focus was on what could be done by hand
- Most developed methods were easy/possible to do using pen and paper
- Limited scope of what could be done
- In 1970s, advent of cheap(ish) digital computers changed things

## Simulation Based Statistical Methods (2)

- The idea of simulation based methods to solve problems pre-dates computers
- Example problem from 18th century: Buffon's needle
  - Named after Georges-Louis Leclerc, Comte De Buffon
  - Asked the question:
    *"Suppose we have a floor made of parallel strips of wood, each the same width, and we drop a needle onto the floor. What is the probability that the needle will lie across a line between two strips?"*
- Solving this problem analytically is difficult; requires knowledge of integral geometry
- But a simple solution exists that requires no special knowledge

## Simulation Based Statistical Methods (3)

- Simulation solution to Buffon's needle problem
- Repeatedly drop a needle onto a floor made of parallel strips of wood, and record the proportion of times the needle lines across a line between the two strips
- By weak law of large numbers, if we repeat this sufficient times, we can estimate the probability with increasing accuracy
- This is an example of a Monte Carlo method
  - Why is it called this?

# Convergence of Empirical Probabilities (1)

- Previous problem an example of the following general problem
- Given RVs $X_1, \ldots, X_p \in \mathcal{X}$ from some population, find

$$\mathbb{P}(\{X_1, \ldots, X_p\} \in Z)$$

  where $Z \subset \mathcal{X}$
- That is, find the probability that the RVs take on values in some set $Z$
- Very general problem statement

## Convergence of Empirical Probabilities (2)

- For example, let $X_1, X_2 \sim N(0,1)$; what is

$$\mathbb{P}(\sqrt{|X_1|} + \sqrt{|X_2|} > 2)?$$

  is a problem of the previous form with

$$Z = \{x_1, x_2 : \sqrt{|x_1|} + \sqrt{|x_1|} > 2\}$$

- Or, let $X_1 \sim \mathrm{Poi}(\lambda_1)$ and $X_2 \sim \mathrm{Poi}(\lambda_2)$; what is

$$\mathbb{P}(X_1 - X_2^{3/2} > 0)?$$

  is another problem of previous form with

$$Z = \{x_1, x_2 : x_1 - x_2^{3/2} > 0\}$$

- These problems can be answered analytically, but it is hard

## Convergence of Empirical Probabilities (3)

- Trivial to propose problems that cannot be solved analytically
- Simulation based approach always provides a solution
- Algorithm:
  1. Randomly generate $X_1, \ldots, X_p$ from their distribution $m$ times
  2. Estimate the probability using the empirical probability

$$\mathbb{P}(X_1, \ldots, X_P \in Z) \approx \frac{1}{m} \sum_{i=1}^{m} I(X_1, \ldots, X_P \in Z)$$

- Here, $I(\cdot)$ is the indicator function
  - Returns a one if the condition is met
  - Returns a zero otherwise
- The larger $m$, the more accurate the probability

## Convergence of Empirical Probabilities (4)

- Proof that this approach works is straightforward
- Indicator function $I(\cdot)$ transforms our RVs into Bernoulli RVs
- Probability $\theta$ of success of this Bernoulli is

$$\mathbb{P}(X_1, \ldots, X_P \in Z)$$

- Then

$$\frac{1}{m} \sum_{i=1}^{m} I(X_1, \ldots, X_P \in Z)$$

  is the mean of our $m$ Bernoulli RVs
- By weak law of large numbers

$$\frac{1}{m} \sum_{i=1}^{m} I(X_1, \ldots, X_P \in Z) \to \theta$$

  as $m \to \infty$.

## Convergence of Empirical Probabilities (5)

- Example: Let $X \sim N(0,1)$, and find

$$\mathbb{P}(\sqrt{|X|} > 1)$$

- Let $Q = \sqrt{|X|}$; then the pdf of $Q = \sqrt{|X|}$ is

$$p(Q = q) = \left(\frac{2^3}{\pi}\right)^{\frac{1}{2}} q \exp\left(-\frac{q^4}{2}\right)$$

  and

$$\mathbb{P}(Q > 1) = \int_1^\infty p(q)dq = 1 - \frac{2}{\sqrt{\pi}} \int_0^x \exp\left(-t^2\right) dt = 0.3171$$

- The integral can be evaluated using MATLAB/R as it is a standard integral (erf)

## Convergence of Empirical Probabilities (6)

- To get an approximate answer using simulation is trivial

```
X = rnorm(m, 0, 1)
mean(sqrt(abs(X)) > 1)
```

- Results:
  - $m = 100$, estimate was $0.3300$
  - $m = 1,000$, estimate was $0.3310$
  - $m = 10,000$, estimate was $0.3179$
  - $m = 100,000$, estimate was $0.3176$
  - $m = 10^9$, estimate was $0.3172$
- Recall exact probability is $0.3171$

## Convergence of Empirical Probabilities (7)

- In theory, for large enough $m$, one can always estimate any probability
- In practice, this simple approach may not always work well
- If $m$ is the number of samples we have generated, then $1/m$ is the smallest non-zero probability we can estimate
  - This is the resolution of our estimate
- If the probability we are estimating is smaller than $1/m$, then we cannot estimate it well
- So for small probabilites $p$, we need a very large $m$, at least $10$ times greater than $1/p$ to get a good estimate

## More generally

- We can find any statistic of $X$ (variance, median, quantiles, etc.) to good accuracy if we have enough random samples
- Let `x` be an R vector containing $m$ samples drawn from $p(x)$; then

$$\mathbb{E}[X] \approx \texttt{mean(x)};$$

and

$$\mathbb{V}[X] \approx \texttt{var(x)};$$

and

$$Q(p = 0.2) \approx \texttt{quantile(x,p=0.2)};$$

and $p(x)$ itself can be approximated by

```
hist(x,Probability=T)
```

and so on, with better approximation as $m$ grows.

## Pseudo-Random Numbers (1)

- To use simulation we need to generate random numbers
- A computer is completely deterministic, so how to do that?
- We instead use pseudo random numbers
- A pseudo-random number generator is an algorithm that generates a sequence of numbers that
    - Are completely deterministic, assuming you know the process and the state of the algorithm
    - Are indistinguishable from true random numbers if you do not know the state

## Pseudo-Random Numbers (2)

- The most important random numbers we need to generate are uniformly distributed
- The pdf for a uniformly distributed RV with support $(0, 1)$ is

$$p(X = x) = 1$$

- We can use uniformly distributed RVs to get RVs from any distribution
- Generating uniform RVs is a well studied problem
  - The SIMD-oriented Fast Mersenne Twister is cutting edge
  - It is fast, and has a period of up to $2^{2216091} - 1$ (before it repeats)

## Pseudo-Random Numbers (3)

- Most uniform random number generators are based on

$$X_n \leftarrow f(X_{n-1}, X_{n-2}, \ldots, X_0)$$

  where $f(\cdot)$ is a transition function.
- The initial choice of $X_0$ is called the seed
  - Usually chosen using the current time, or sometimes a “true” random number generator
- Given the same seed, the sequence of pseudo-random numbers will be identical
- This is a disadvantage, and an advantage
  - Disadvantage: good seeding is crucial
  - Advantage: your simulations can be exactly repeatable

## Sampling Other Random Variables

- Armed with the ability to generate uniform RVs, how to simulate from arbitrary distributions?
- For example, how to simulate RVs following a normal distribution?
- There are many, many algorithms to attack this problem
- Some other methods for sampling RVs
  - Inverse CDF procedure
  - Metropolis-Hastings algorithm
  - Accept-Reject algorithms
  - Adaptive rejection sampling
  - Slice sampling
- To some degree, no need to worry as R/Python/Matlab have implementations of most standard RVs

## Outline

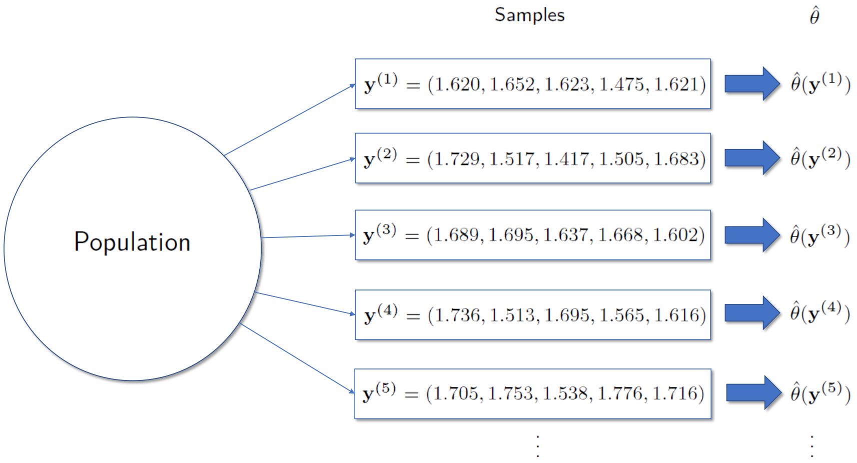
## Data Resampling Methods (1)

- Bootstrap methods are one of the most widely used simulation based statistical methods
- They are part of the larger family of “resampling” methods
- We have already met one – cross validation
- In CV, we estimate prediction error on future data
- The bootstrap is about quantifying how variable our estimates are
  - Standard errors, confidence intervals, etc.

## Data Resampling Methods (2)

- Recall the population-sample model of inference
- We have a (infinitely) large population
  - Characterised by some quantity $\theta$
- We draw a *finite* sample of size $n$ from our population
- Use this sample to estimate $\theta$ (call this $\hat{\theta}$)
- How much does this estimate vary if we draw a new sample from our population?

## Data Resampling Methods (2)


Samples
$\hat{\theta}$
Population
$\mathbf{y}^{(1)} = (1.620, 1.652, 1.623, 1.475, 1.621)$
$\hat{\theta}(\mathbf{y}^{(1)})$
$\mathbf{y}^{(2)} = (1.729, 1.517, 1.417, 1.505, 1.683)$
$\hat{\theta}(\mathbf{y}^{(2)})$
$\mathbf{y}^{(3)} = (1.689, 1.695, 1.637, 1.668, 1.602)$
$\hat{\theta}(\mathbf{y}^{(3)})$
$\mathbf{y}^{(4)} = (1.736, 1.513, 1.695, 1.565, 1.616)$
$\hat{\theta}(\mathbf{y}^{(4)})$
$\mathbf{y}^{(5)} = (1.705, 1.753, 1.538, 1.776, 1.716)$
$\hat{\theta}(\mathbf{y}^{(5)})$
$\vdots$
$\vdots$

## Data Resampling Methods (3)

- In general, to analyse the behaviour of our estimate under repeated sampling, we need the sampling distribution
- This involves us specifying a population distribution we believe is appropriate
    - We must then derive the distribution of the estimate $\hat{\theta}$
- This can be difficult (or impossible), and relies on specific assumptions
- If $\hat{\theta}$ is equal to sample mean we need weaker assumptions
- But for general statistic, the problems remain

## The Exact Bootstrap (1)

- In the 1970s several people proposed data resampling methods to solve this problem
- One of the most important is the bootstrap (Efron, 1970)
- The basic idea is that we should treat our sample as an estimate of our population
- We can then draw new samples from our this surrogate population
- Use this to define an empirical sampling distribution for statistic
- Relies on the fact that the as $n$ increases, the sample probabilities are a consistent estimate of the population

## The Exact Bootstrap (2)

- The bootstrap works by treating our sample as our population
- Form a distribution over all $M$ possible combinations of data points you could form from the sample (with replacement)
- Let $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}$ the combinations of data points
    - These represent all the samples you could draw from this "population"
- Compute our estimate of interest $\hat{\theta}$ for each combination
    - Call this quantity $\hat{\theta}^{(i)}$
- Let $\hat{\theta}$ denote the estimate from the original sample $\mathbf{y}$
    - If our data $\mathbf{y}$ was the complete population, this would be our "population value" of $\theta$
    - We can then use the bootstrap distribution to estimate quantities such as bias by comparing them against $\hat{\theta}$

## The Exact Bootstrap (3)

- For example, we can estimate the bias using:

$$\text{bias} = \frac{1}{M} \sum_{i=1}^{m} \hat{\theta} - \hat{\theta}^{(i)}$$

  which is the average difference between the estimate on the original sample (our “population value”) and the bootstrap estimates

- We can estimate the variance using

$$\text{Var} = \left(\frac{1}{M-1}\right) \sum_{i=1}^{M} \left(\hat{\theta}^{(i)} - \frac{1}{M} \sum_{j=1}^{M} \hat{\theta}^{(j)}\right)^2$$

  which is the empirical variance of our bootstrap samples of $\hat{\theta}$

- Confidence intervals can be obtained using percentiles of the bootstrap estimates

## The Exact Bootstrap: Example (1)

- As an example, consider the sample

$$\mathbf{y} = (2, 6, 3)$$

- Compute the bootstrap distribution for the sample mean $\bar{Y}$

$$\begin{array}{ccc}
(2,2,2) & & 2 \\
(2,2,6) & & 3.3333 \\
(2,2,3) & & 2.3333 \\
(2,6,2) & & 3.3333 \\
(2,6,6) & \overset{\bar{Y}}{\Longrightarrow} & 4.6667 \\
\vdots & & \vdots \\
(3,3,2) & & 2.6667 \\
(3,3,6) & & 4 \\
(3,3,3) & & 3
\end{array}$$

## The Exact Bootstrap: Example (1)

- Using this exact bootstrap distribution we find

$$\text{bias} = 0$$

  and

$$\text{Var} = 1$$

- Note that we made no assumptions about the distribution of the population
  - The exact bootstrap distribution, and therefore estimates from bootstrap, become more accurate as $n$ increases
- But the number of different bootstrap samples $M = n^n$
  $\Rightarrow$ the exact bootstrap rapidly becomes infeasible

## The Bootstrap Algorithm (1)

- Let $\mathbf{y} = (y_1, \ldots, y_n)$ be a sample of size $n$ from an unknown population
- Let $\hat{\theta}$ be the estimate of our statistic on the full sample $\mathbf{y}$
- The bootstrap algorithm: For $i = 1$ to $m$ ($m \ll M$)
  1. Create a new sample of size $n$ by sampling with replacement from $\mathbf{y}$
  2. Compute the estimate $\hat{\theta}^{(i)}$ of our statistic based on this new sample
- This is called a resampling procedure
- $\hat{\theta}^{(1)}, \ldots, \hat{\theta}^{(m)}$ is as estimate of bootstrap distribution
- The larger the $m$, the closer the bootstrap is to the exact bootstrap
  - Often works well even for values of $m = 1,000$

## The Bootstrap Algorithm (2)

- The bootstrap is very general
- For example, imagine we have a linear regression model
- We can calculate the mean-squared prediction error on the data we used to fit the model
    - This will be optimistic, as we have used the training data twice
- It would be useful to get a confidence interval on this statistic
    - Will give us a range of plausible values of error for future data
- To do this using bootstrap
    - Resample $m$ bootstrap samples
    - Fit models using these samples and calculate mean-squared error of our fit
    - Use set of different MSE values to get a confidence interval

## Example: Supervised Learning

- Example: predicting `BP` (our $y$) using `Age` and `Weight`
- Example, forming a single bootstrap sample

Original Data

| BP | Age | Weight |
|---|---|---|
| 105 | 47 | 85.4 |
| 115 | 49 | 94.2 |
| 116 | 49 | 95.3 |
| 117 | 50 | 94.7 |
| 112 | 51 | 89.4 |
| 121 | 48 | 99.5 |
| 110 | 47 | 90.9 |
| 115 | 49 | 94.1 |
| 125 | 52 | 101.3 |

$\Longrightarrow$

Bootstrap Sample

| BP | Age | Weight |
|---|---|---|

## Example: Supervised Learning

- Example: predicting `BP` (our $y$) using `Age` and `Weight`
- Example, forming a single bootstrap sample

Original Data

| `BP` | `Age` | `Weight` |
|---|---|---|
| 105 | 47 | 85.4 |
| 115 | 49 | 94.2 |
| 116 | 49 | 95.3 |
| 117 | 50 | 94.7 |
| 112 | 51 | 89.4 |
| 121 | 48 | 99.5 |
| 110 | 47 | 90.9 |
| 115 | 49 | 94.1 |
| 125 | 52 | 101.3 |

$\Longrightarrow$

Bootstrap Sample

| `BP` | `Age` | `Weight` | $I$ |
|---|---|---|---|
| 115 | 49 | 94.1 | 8 |

## Example: Supervised Learning

- Example: predicting `BP` (our $y$) using `Age` and `Weight`
- Example, forming a single bootstrap sample

Original Data

| `BP` | `Age` | `Weight` |
|---|---|---|
| 105 | 47 | 85.4 |
| 115 | 49 | 94.2 |
| 116 | 49 | 95.3 |
| 117 | 50 | 94.7 |
| 112 | 51 | 89.4 |
| 121 | 48 | 99.5 |
| 110 | 47 | 90.9 |
| 115 | 49 | 94.1 |
| 125 | 52 | 101.3 |

$\Longrightarrow$

Bootstrap Sample

| `BP` | `Age` | `Weight` | $I$ |
|---|---|---|---|
| 115 | 49 | 94.1 | 8 |
| 125 | 52 | 101.3 | 9 |

## Example: Supervised Learning

- Example: predicting `BP` (our $y$) using `Age` and `Weight`
- Example, forming a single bootstrap sample

Original Data

| BP | Age | Weight |
|---|---|---|
| 105 | 47 | 85.4 |
| 115 | 49 | 94.2 |
| 116 | 49 | 95.3 |
| 117 | 50 | 94.7 |
| 112 | 51 | 89.4 |
| 121 | 48 | 99.5 |
| 110 | 47 | 90.9 |
| 115 | 49 | 94.1 |
| 125 | 52 | 101.3 |

$\Longrightarrow$

Bootstrap Sample

| BP | Age | Weight | $I$ |
|---|---|---|---|
| 115 | 49 | 94.1 | 8 |
| 125 | 52 | 101.3 | 9 |
| 115 | 49 | 94.2 | 2 |

## Example: Supervised Learning

- Example: predicting `BP` (our $y$) using `Age` and `Weight`
- Example, forming a single bootstrap sample

Original Data

| BP | Age | Weight |
|---|---|---|
| 105 | 47 | 85.4 |
| 115 | 49 | 94.2 |
| 116 | 49 | 95.3 |
| 117 | 50 | 94.7 |
| 112 | 51 | 89.4 |
| 121 | 48 | 99.5 |
| 110 | 47 | 90.9 |
| 115 | 49 | 94.1 |
| 125 | 52 | 101.3 |

$\Longrightarrow$

Bootstrap Sample

| BP | Age | Weight | $I$ |
|---|---|---|---|
| 115 | 49 | 94.1 | 8 |
| 125 | 52 | 101.3 | 9 |
| 115 | 49 | 94.2 | 2 |
| 125 | 52 | 101.3 | 9 |

## Example: Supervised Learning

- Example: predicting `BP` (our $y$) using `Age` and `Weight`
- Example, forming a single bootstrap sample

Original Data

| `BP` | `Age` | `Weight` |
|---|---|---|
| 105 | 47 | 85.4 |
| 115 | 49 | 94.2 |
| 116 | 49 | 95.3 |
| 117 | 50 | 94.7 |
| 112 | 51 | 89.4 |
| 121 | 48 | 99.5 |
| 110 | 47 | 90.9 |
| 115 | 49 | 94.1 |
| 125 | 52 | 101.3 |

$\Longrightarrow$

Bootstrap Sample

| `BP` | `Age` | `Weight` | $I$ |
|---|---|---|---|
| 115 | 49 | 94.1 | 8 |
| 125 | 52 | 101.3 | 9 |
| 115 | 49 | 94.2 | 2 |
| 125 | 52 | 101.3 | 9 |
| 121 | 48 | 99.5 | 6 |
| 105 | 47 | 85.4 | 1 |
| 116 | 49 | 95.3 | 3 |
| 112 | 51 | 89.4 | 5 |
| 125 | 52 | 101.3 | 9 |

## Example: Bagging

- Bootstrapping can be used to improve prediction error
- The bagging (bootstrap aggregration algorithm) does just this
- Procedure:
  - Generate $m$ bootstrap samples
  - For each bootstrap sample, fit a model
  - Predict onto new data by averaging over predictions from all $m$ models
- Can improve prediction errors if model is low bias/high variance
  - Trees are a prime example
- Similar to random forests – inspired them, in fact

## The Bootstrap

- Strengths of the bootstrap
  - Only assumption made is that our sample is representative of our population
  - Easy to code, easy to apply
  - Accuracy of estimates increases with increasing sample size
- Weaknesses of the bootstrap
  - Can be slow if estimates are slow to compute
  - For small $n$ accuracy is not necessarily good
  - Might need large $m$ to get “smooth” distribution
  - Cannot be applied to some problems without modifications, e.g., lasso

## Permutation Tests (1)

- The bootstrap is usually used to estimate sampling statistics
  - Bias
  - Standard errors
  - Confidence intervals
- Another important statistic based on sampling distributions are $p$-values
- We can use another resampling technique called permutation tests to estimate these
- These are particular useful for testing hypotheses of association
  - E.g., supervised learning of $y$ given predictors

## Permutation Tests (2)

- Consider $n$ pairs of targets $y_i$ and predictor values $x_i$
  - Is $y$ associated with $x$ using a linear model?
- We want to test

$$H_0 : \beta = 0 \text{ vs } H_A : \beta \neq 0$$

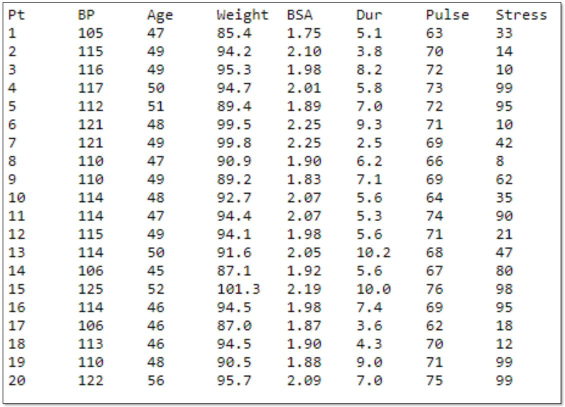
- The usual procedure is to
  1. Specify a distribution for the population
  2. Calculate a test statistic (i.e., the least-squares estimate $\hat{\beta}$)
  3. See how likely our observed statistic would be under the null distribution
- The $p$-value depends crucially on
  - The choice of assumed distribution for the population
  - The ability to derive the null distribution for our statistic accurately (often CLT is used)

## Permutation Tests (3)

- The permutation test approach resamples the $y$ values to approximate the (unknown) null distribution
- The permutation test algorithm: for $i = 1$ to $m$
  1. Randomly permute the values of the targets
  2. Calculate our association statistic of interest; call it $\hat{\theta}^{(i)}$
- Then we calculate the association statistic on our data
- And see how likely it would be to arise under our permutation distribution $\hat{\theta}^{(1)}, \ldots, \hat{\theta}^{(m)}$
  - This is our $p$-value of association
- The key idea is that permutations of the $y$'s will not be associated with the predictors, so we can see how the estimates vary under our null distribution

## Permutation Test: Example (1)

- Consider the following dataset (we examined in Lecture 6)



| Pt | BP | Age | Weight | BSA | Dur | Pulse | Stress |
|---|---|---|---|---|---|---|---|
| 1 | 105 | 47 | 85.4 | 1.75 | 5.1 | 63 | 33 |
| 2 | 115 | 49 | 94.2 | 2.10 | 3.8 | 70 | 14 |
| 3 | 116 | 49 | 95.3 | 1.98 | 8.2 | 72 | 10 |
| 4 | 117 | 50 | 94.7 | 2.01 | 5.8 | 73 | 99 |
| 5 | 112 | 51 | 89.4 | 1.89 | 7.0 | 72 | 95 |
| 6 | 121 | 48 | 99.5 | 2.25 | 9.3 | 71 | 10 |
| 7 | 121 | 49 | 99.8 | 2.25 | 2.5 | 69 | 42 |
| 8 | 110 | 47 | 90.9 | 1.90 | 6.2 | 66 | 8 |
| 9 | 110 | 49 | 89.2 | 1.83 | 7.1 | 69 | 62 |
| 10 | 114 | 48 | 92.7 | 2.07 | 5.6 | 64 | 35 |
| 11 | 114 | 47 | 94.4 | 2.07 | 5.3 | 74 | 90 |
| 12 | 115 | 49 | 94.1 | 1.98 | 5.6 | 71 | 21 |
| 13 | 114 | 50 | 91.6 | 2.05 | 10.2 | 68 | 47 |
| 14 | 106 | 45 | 87.1 | 1.92 | 5.6 | 67 | 80 |
| 15 | 125 | 52 | 101.3 | 2.19 | 10.0 | 76 | 98 |
| 16 | 114 | 46 | 94.5 | 1.98 | 7.4 | 69 | 95 |
| 17 | 106 | 46 | 87.0 | 1.87 | 3.6 | 62 | 18 |
| 18 | 113 | 46 | 94.5 | 1.90 | 4.3 | 70 | 12 |
| 19 | 110 | 48 | 90.5 | 1.88 | 9.0 | 71 | 99 |
| 20 | 122 | 56 | 95.7 | 2.09 | 7.0 | 75 | 99 |

- We want to model blood pressure using Age

## Permutation Test: Example (2)

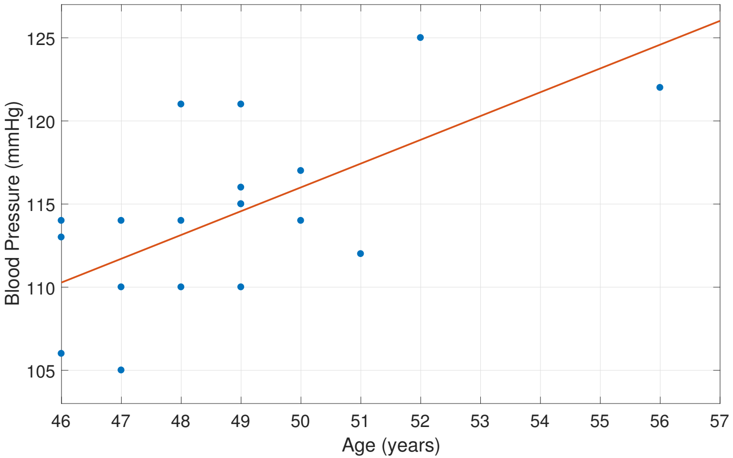
- Fit `Age` to `BP` using least-squares regression
  - $\hat{\beta} = 1.4310$
  - $p$-value of association ($t$-test) $= 0.00157$
- $p$-value was calculated under assumption population is normal
- To calculate a $p$-value using permutation test
  - Permute $y$ (`BP`) values $m$ times, then fit `Age` to `BP`
  - Store each permutation estimate as $\hat{\beta}^{(i)}$
- Calculate the $p$-value using

$$p \approx \frac{1}{m} \sum_{i=1}^{m} I\left(|\hat{\beta}^{(i)}| > |\hat{\beta}|\right)$$

i.e., the proportion of times the absolute value of the estimates obtained from our permutations exceeds the absolute value of estimate $\hat{\beta}$ obtained on our data

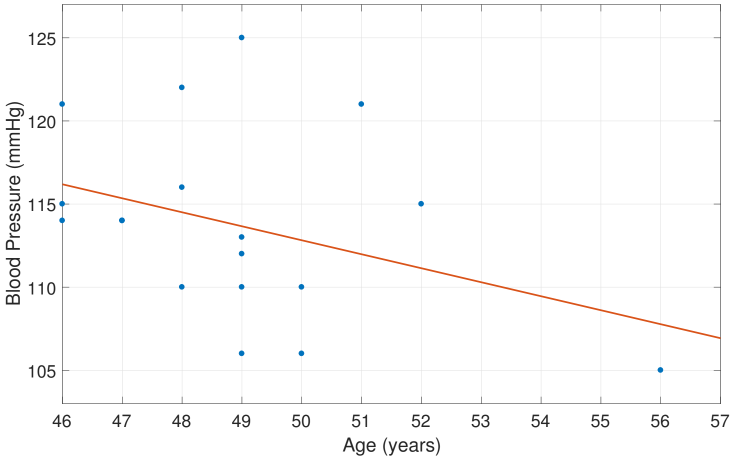
## Permutation Test: Example (3)

- Unpermutated data: $\hat{\beta} = 1.4310$



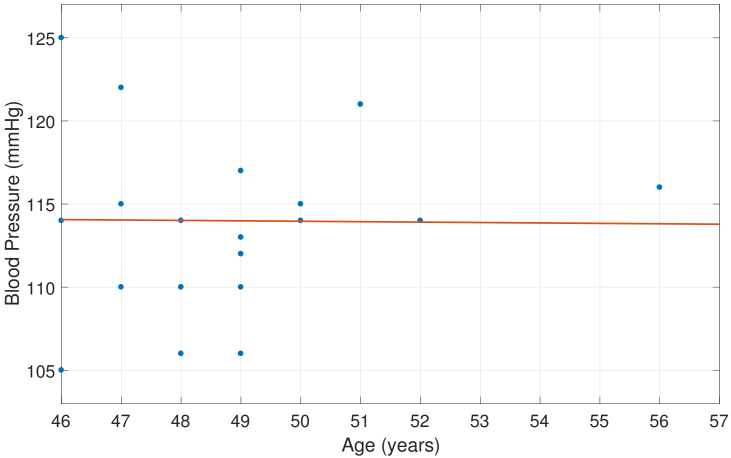
## Permutation Test: Example (4)

- Permuation: $\hat{\beta} = -0.8418$



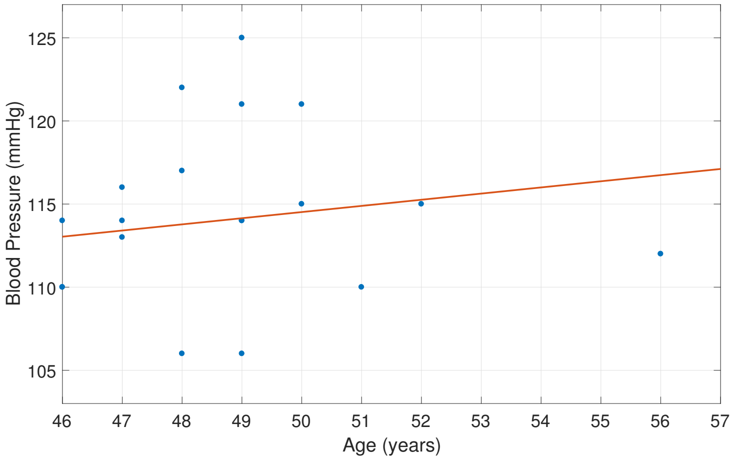
## Permutation Test: Example (5)

- Permuation: $\hat{\beta} = -0.0253$



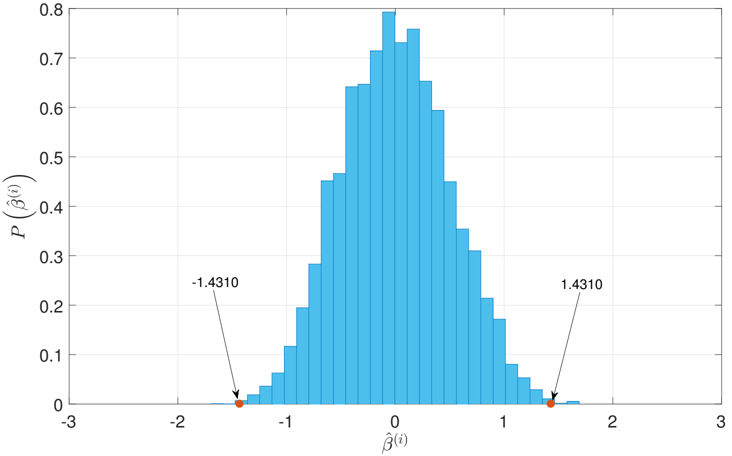
## Permutation Test: Example (6)

- Permuation: $\hat{\beta} = 0.3704$



## Permutation Test: Example (7)

- Distribution of $m = 10,000$ permutation samples



- $\hat{\beta} = 1.4310$, $\mathbb{P}\left(|\hat{\beta}^{(i)}| > |\hat{\beta}|\right) \approx 0.0013$, $t$-test $= 0.00157$

## Permutation Tests

- Strengths of permutation tests:
  - Makes few assumptions about population distribution
  - Potentially more accurate $p$-values for non-normal models (logistic reg, etc.)
  - Easy to code, easy to apply
- Weaknesses of permutation tests:
  - Resolution is determined by $m$; smallest non-zero value is $1/m$
  - Can be slow to run if fitting model is very slow
  - Assumes that individuals in population are independent

## Reading/Terms to Revise

- For those who want to know more about random number generation/sampling and bootstrap methods, I recommend the two books:
  - *An Introduction to the Bootstrap*, B. Efron and R. Tibshirani
  - *Monte Carlo Statistical Methods*, C. Robert

- Terms to know:
  - Simulation
  - Pseudo-random numbers
  - Bootstrap distribution
  - Bootstrap algorithm
  - Permutation test

- Next week: machine learning algorithms for unsupervised data discovery (clustering, mixture modelling, matrix completion)