# Machine Learning with R Oct 2020

Ulaş Işıldak
Middle East Technical University
Biological Sciences
isildak.ulas@gmail.com

# Workshop Material

All the workshop materials are available in the GitHub repository:

https://github.com/rsgturkey/Workshop2020

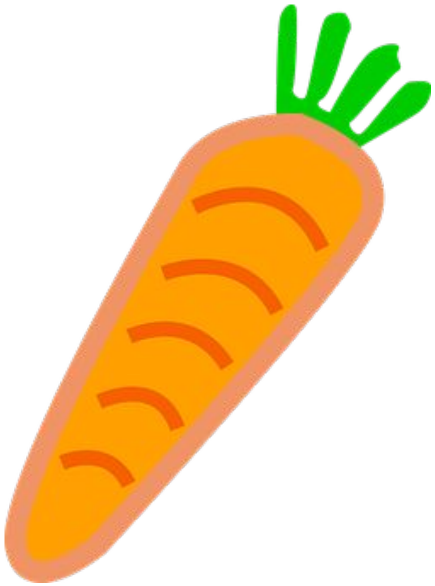
# Aim

The aim of this workshop is to introduce you to the main concepts and some important models in machine learning, and to enable you to implement basic machine learning models in R.

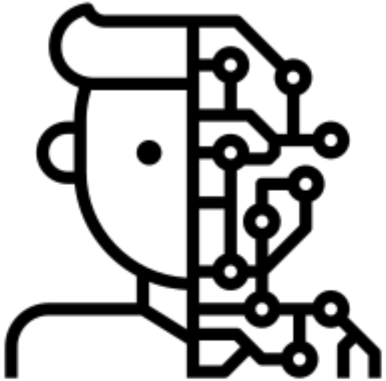
We will not cover in depth, advanced treatment of machine learning.

# What we will cover?

- What is Machine Learning?
- Supervised Learning Algorithms
  - Regression
  - Classification
- Model selection & evaluation
- ML Tools in R: caret
- Application on real data



# What is Machine Learning?



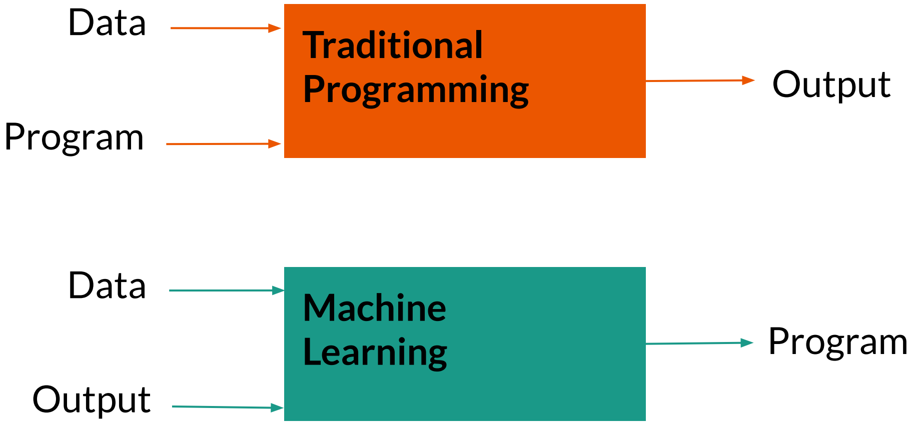
**Machine Learning ...**

- is a field of **artificial intelligence**,
- uses **statistical techniques**,
- allows computers to **learn without explicitly programmed**.

The **goal** of ML is to discover structure/pattern in data or improve decision making and predictions.

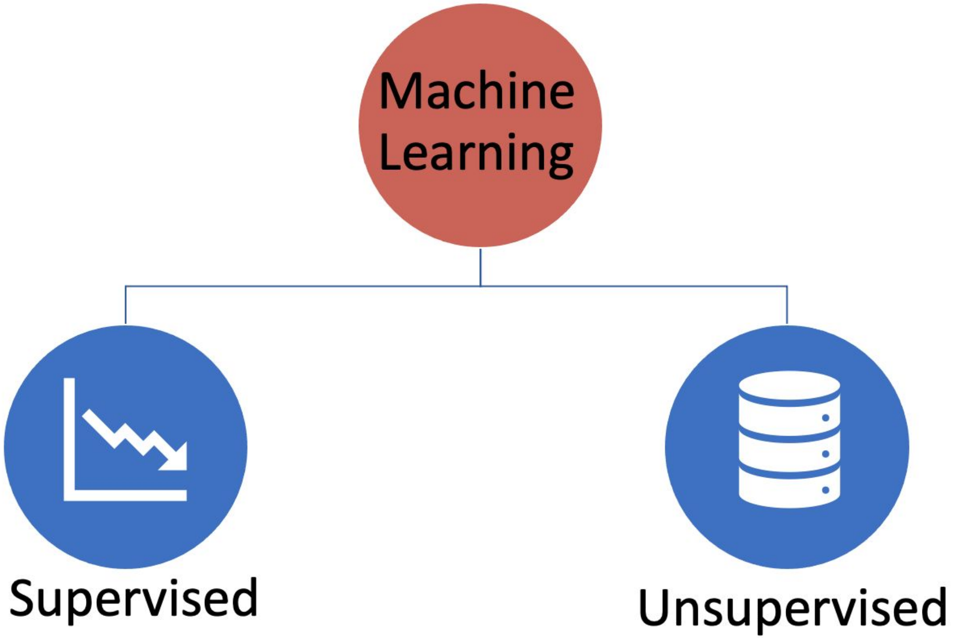
# ML vs Traditional Programming

- In **traditional programming**, a person manually formulates or codes rules.
- In **machine learning**, algorithm automatically formulates the rules from the data.



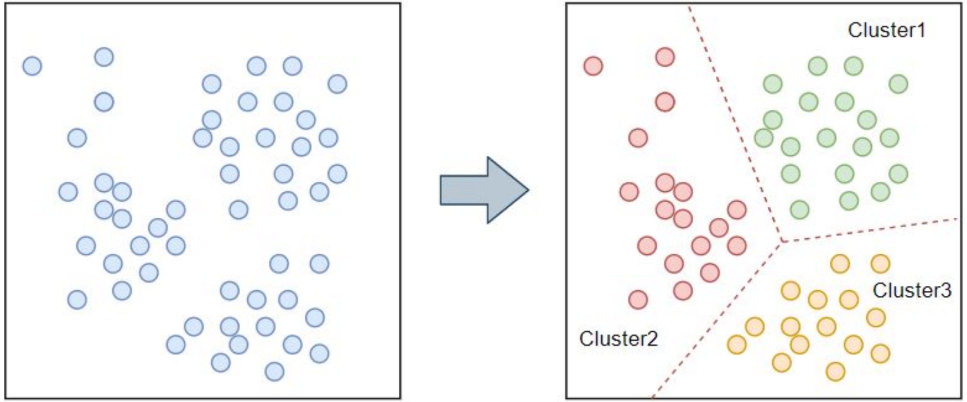
# ML Types

- Two main types of machine learning.
- We will focus on supervised machine learning



# Unsupervised Learning

- Analyzes the relationships to discover structures, trends, or patterns in the data.
- Typically used for **dimensionality reduction** or **clustering** analysis.
- Common algorithms:
  - Hierarchical clustering
  - k-Means clustering
  - Principal Component Analysis (PCA)

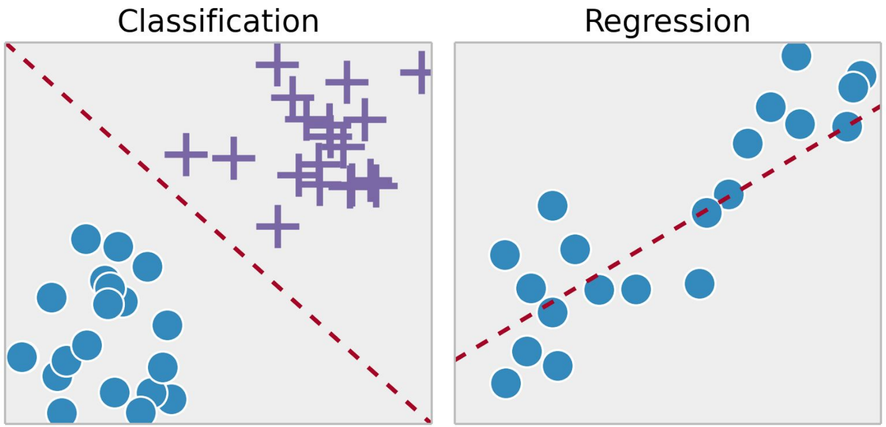



from ecloudvalley.com

# Supervised Learning

Two types of supervised learning problems:

- **Regression**: prediction of a quantitative (continuous) feature
  - Linear regression, polynomial regression.
  - e.g. predict blood sugar level
- **Classification**: prediction of a qualitative (discrete) feature
  - Logistic regression, decision tree, random forest.
  - e.g. predict type of cancer




from towardsdatascience.com

# Linear Regression

- Simple linear regression is a simple way of evaluating the relationship between two variables.

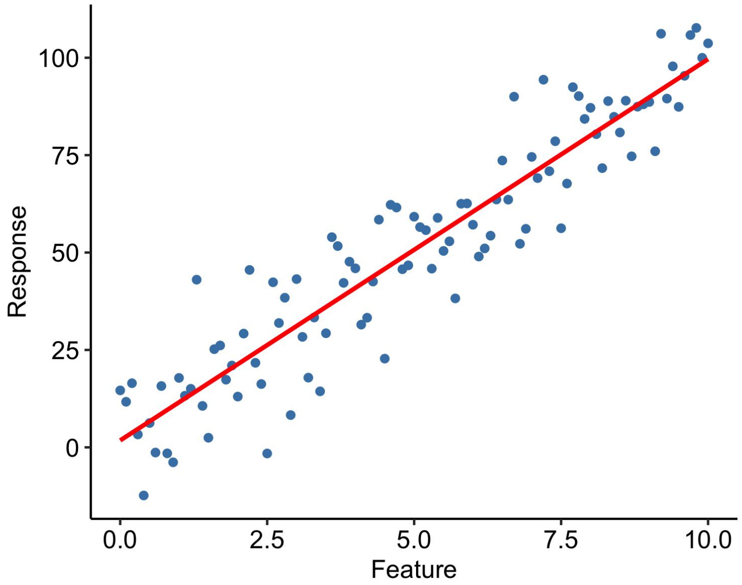
$$Y = \beta_0 + \beta_1 X$$

X - *independent variable* (*feature, predictor*)

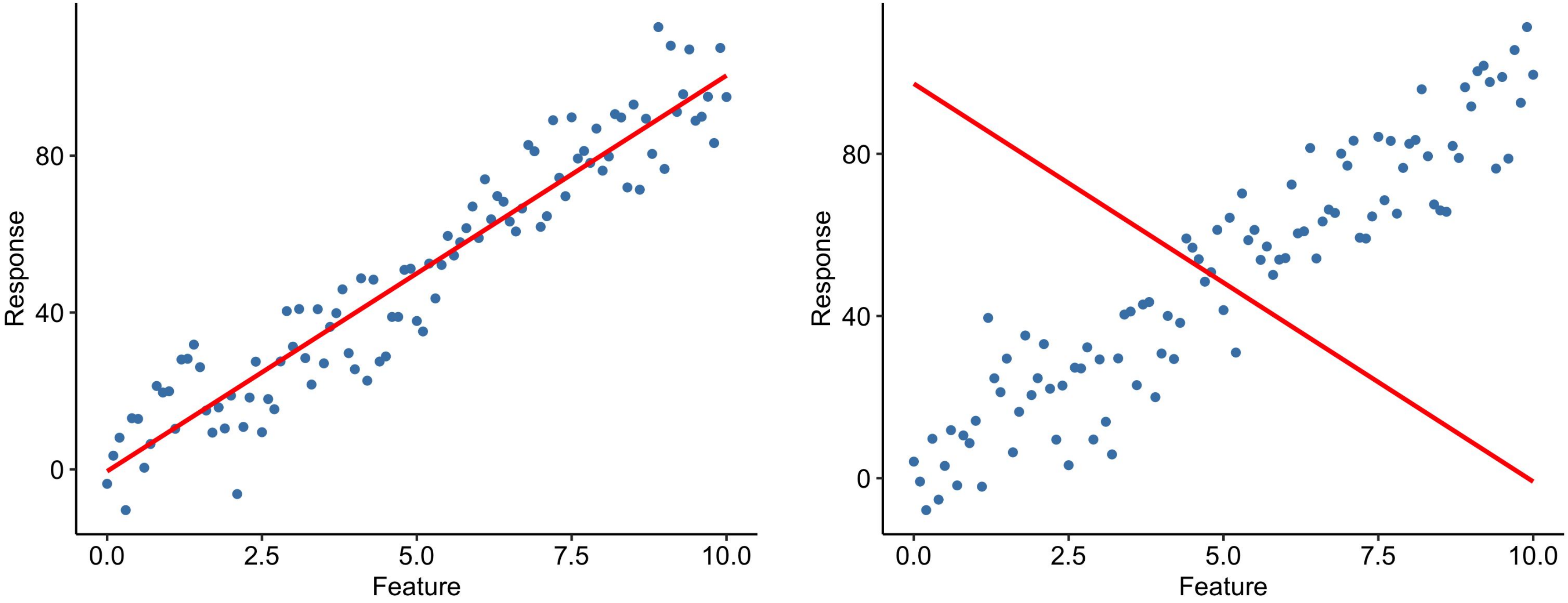
Y - *dependent variable* (*response, outcome*)

$\beta_0$ - intercept

$\beta_1$ - slope coefficient



# Which line fits best?



# Loss Function

- It defines a metric of the errors committed by the model.

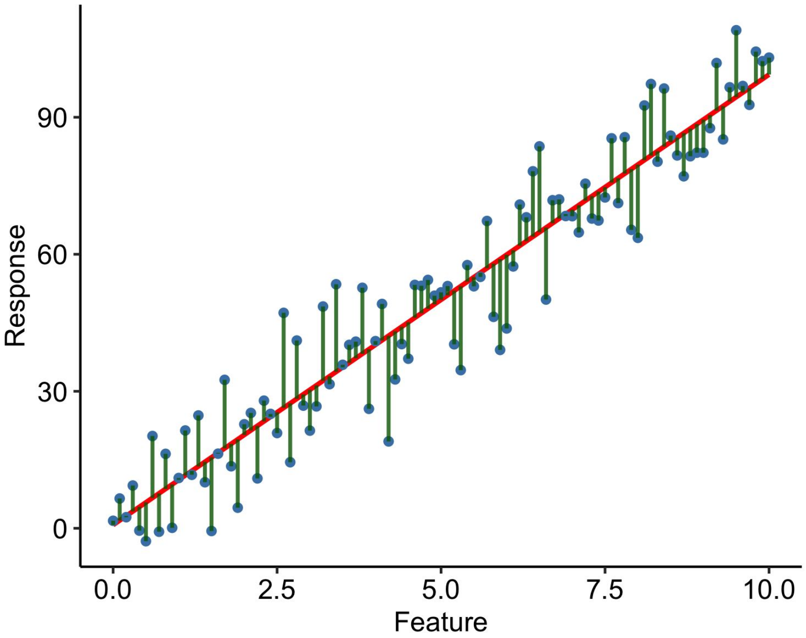
$$Loss = f(Error)$$

- Important in **fitting** and **evaluating:**
    - In finding optimum parameters (fitting)
    - Evaluating the model performance

# Regression Loss

- **Residual -** difference between the observed value and predicted value (i.e. error).
- **Mean Squared Error (MSE)** - average squared error

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Actual_i - Predicted_i)^2$$



# Multiple Regression

- Multiple regression maps the relationship between a response variable and multiple independent variables.

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p$$

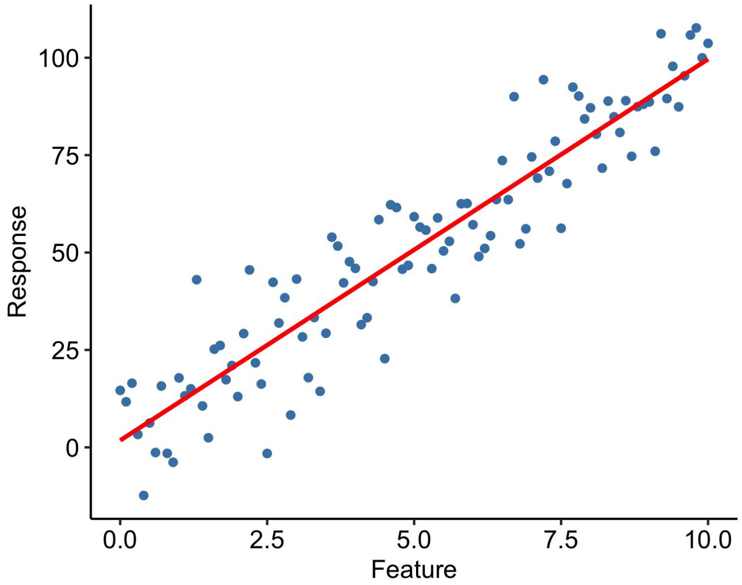
$X_1, X_2, \ldots, X_3$ - *independent variables*

Y - *dependent variable* (*response*)

$\beta_0$ - intercept

$\beta_1, \beta_2, \ldots, \beta_p$ - slope coefficients for each variable

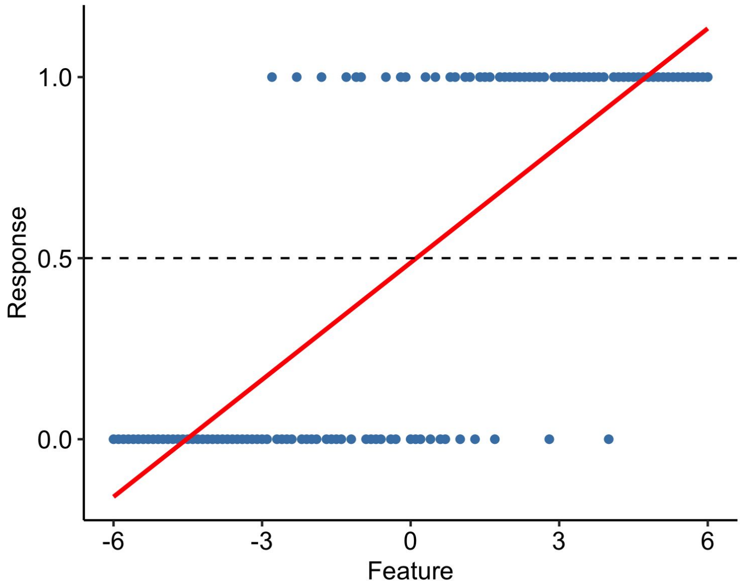
- In essence, it is a simple linear regression that uses multiple features to predict response



# Classification

- Suppose we have a binary classification task:
  - ***Y = f(X)***,

    where the response is Y binary (0 or 1).
- Can we use linear regression for this classification task?
  - i.e., Y = 1, if Y > 0.5
- The problem is that linear regression produces probabilities (outputs) less than 0, or bigger than 1.



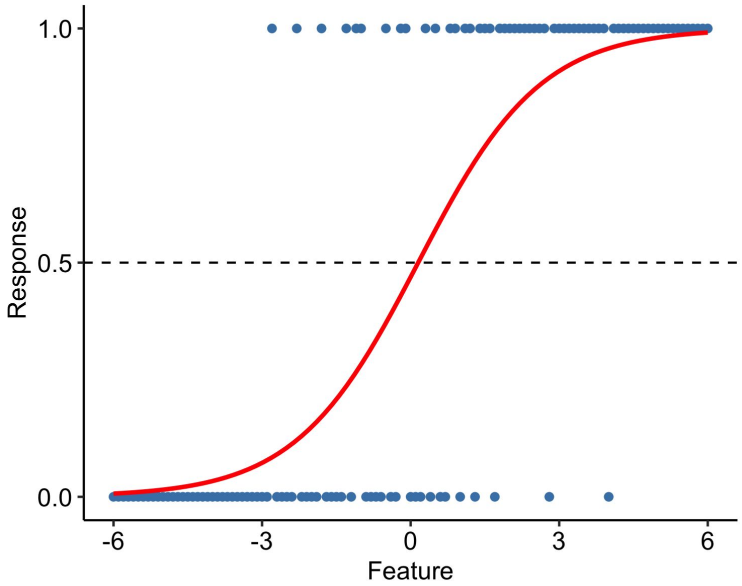
# Logistic Regression

- Logistic regression is similar to linear regression.

$$Y = Logistic(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p)$$

- In logistic regression, the prediction is transformed using a logistic function.
- The logistic function maps predictions to the range of 0 and 1.

$$Logistic(x) = \frac{1}{1 + exp(-x)}$$



# Confusion Matrix

Confusion matrix used to visualize the performance of a classification model.

**Accuracy**: Out of all the cases, what percent of predictions are correct?

$$Acc. = \frac{TP + TN}{TP + TN + FP + FN}$$

Predicted Values

| True Values | 1 | 0 |
|---|---|---|
| 1 | True Positives (TP) | False Negatives (FN) |
| 0 | False Positives (FP) | True Negatives (TN) |

**Sensitivity**: Out of all the positives, what percent of predictions are correct?

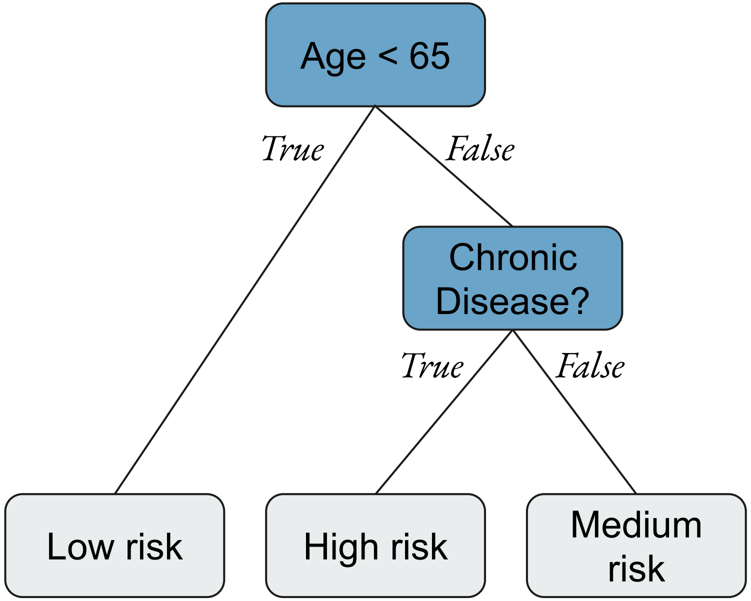
$$Sensitivity = \frac{TP}{TP + FN}$$

**Specificity**: Out of all the negatives, what percent of predictions are correct?

$$Specificity = \frac{TN}{TN + FN}$$

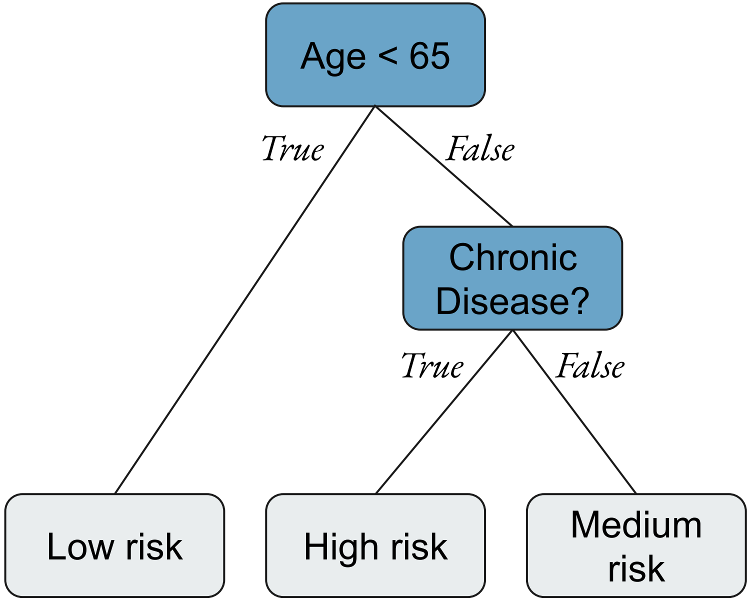
# Decision Tree

- Decision tree models response as a sequence of TRUE or FALSE questions.
- Also called CART: Classification And Regression Trees.
- A decision tree is drawn upside down with its root at the top.
- Simple to understand, and visualize



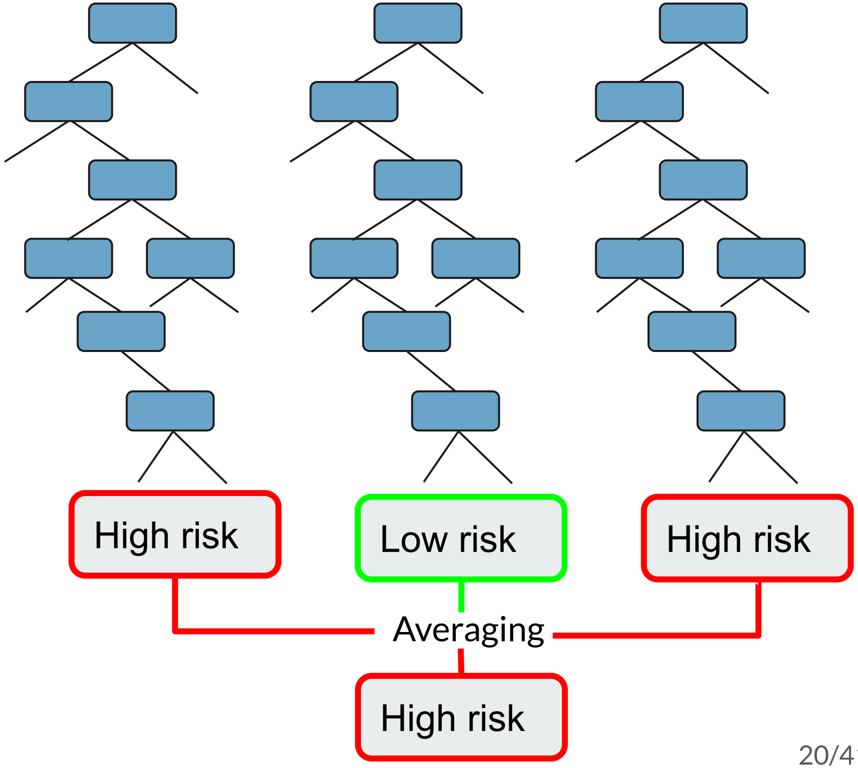
# Decision Tree

- Constructing a decision tree:
    - At each step, choose a feature that best splits the items.
    - Repeat splitting nodes until a predefined threshold is reached (e.g. minsplit).
    - Prune the tree.



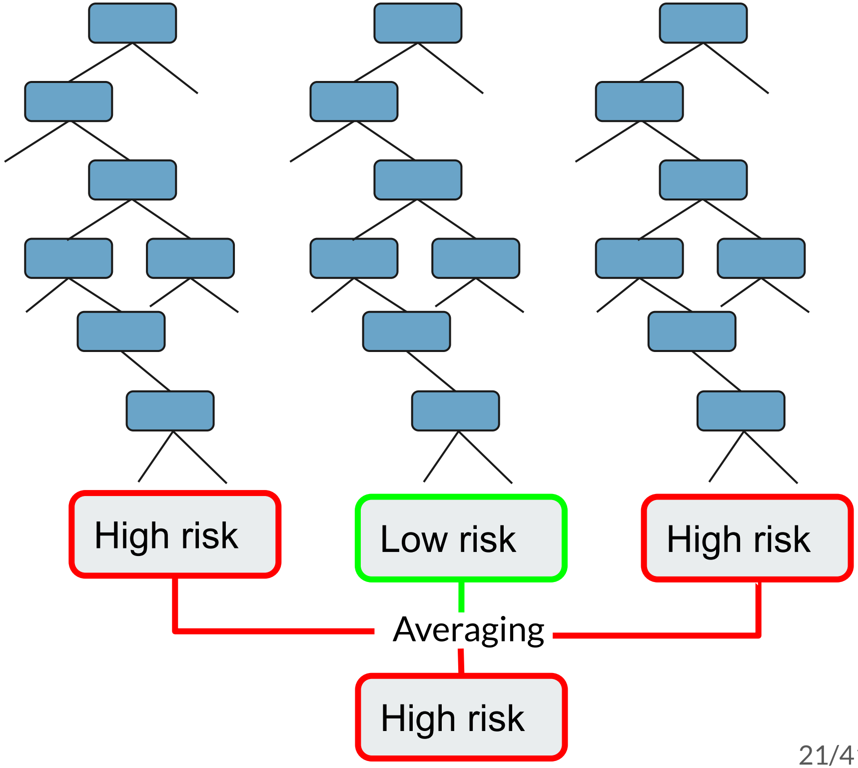
# Random Forest

- Random forests consist of a large number of decision trees, each based on a different feature.
- Each tree performs a prediction and the final prediction is determined by majority vote.
- Harder to interpret, and visualize.

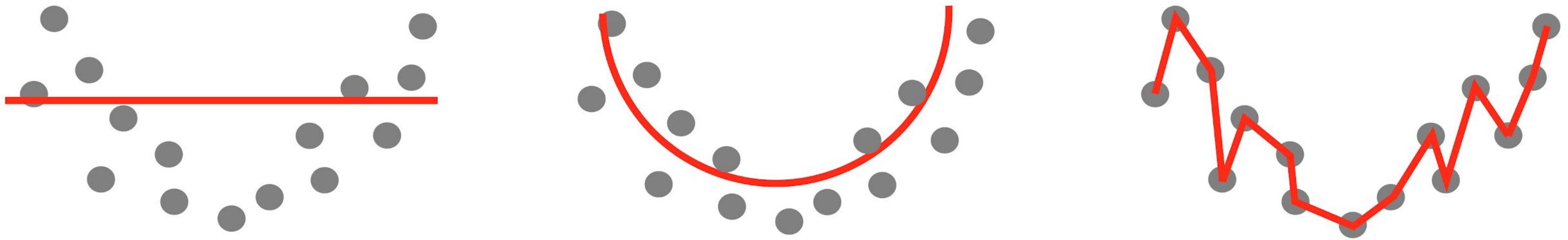


# Random Forest

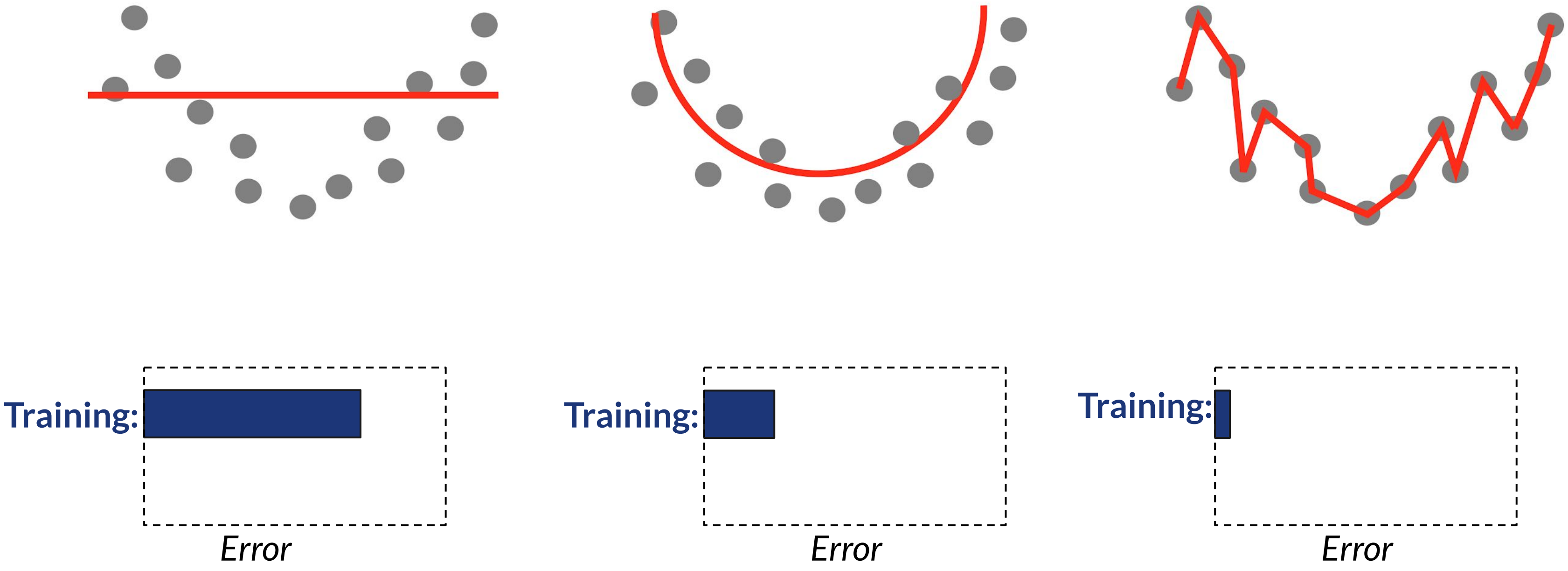
- Constructing a random forest of decision trees:
  - For n times {
    - Resample data
    - Create non-pruned decision tree}
  - Average the fitted values.



# Overfitting

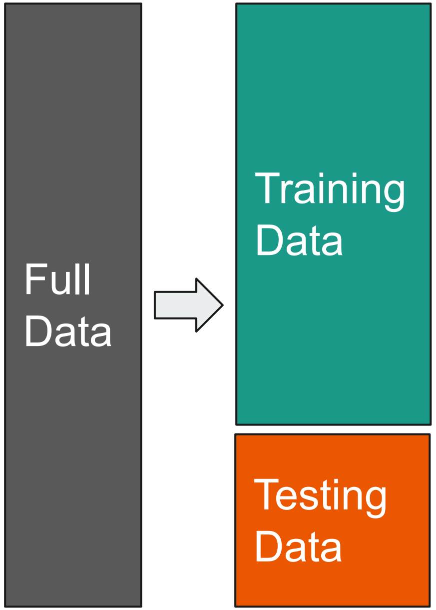


# Overfitting

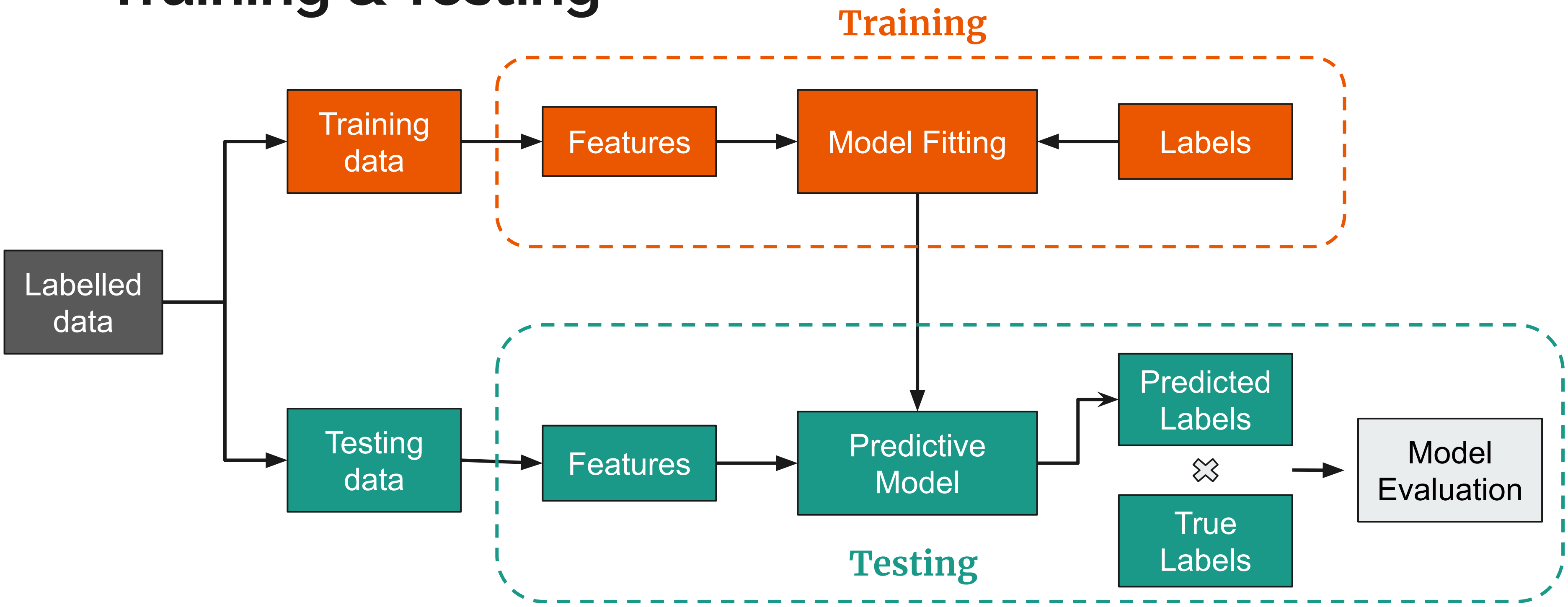


# Hold-out Data

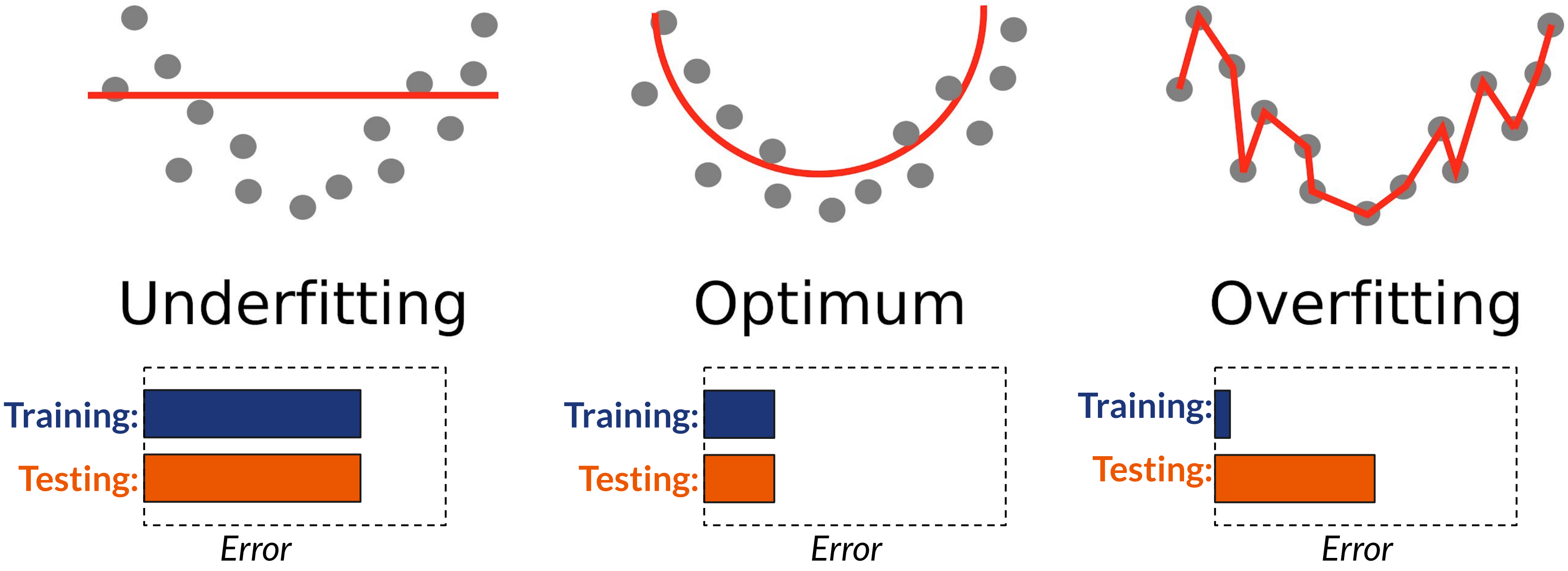
- **Overfitting -** fitting a model to the data too closely and thus failing to predict future observations
  - Therefore, the model accuracy should be evaluated on an unseen data set.
- **Training data** is the set of data used to fit (generate) the model.
- **Testing data** is the set of data used to evaluate model performance after training.
  - Testing set typically created by partitioning the all available data.



# Training & Testing

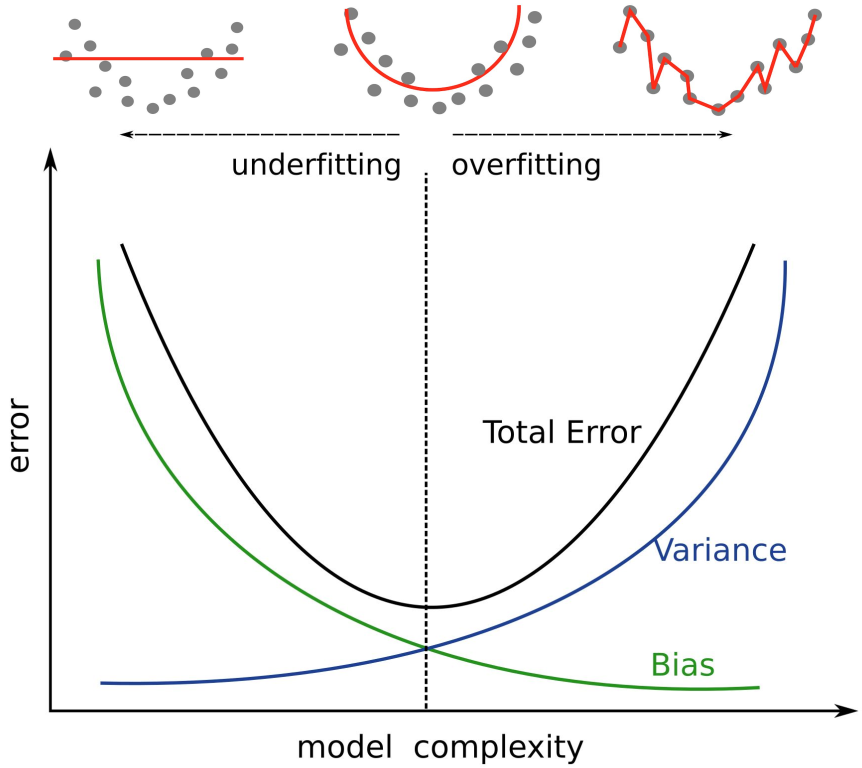


# Overfitting



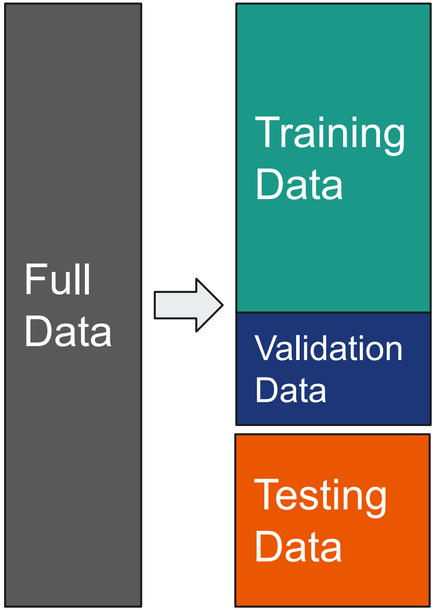
# Bias-Variance Tradeoff

- **Bias** - occurs when a model has limited flexibility.
    - Simply, it is the difference between predictions and true values.
- **Variance** - the sensitivity of a model to a specific set of training data.
    - Reflects how "over-specialized" is the classifier to a particular training set (i.e. overfitting).

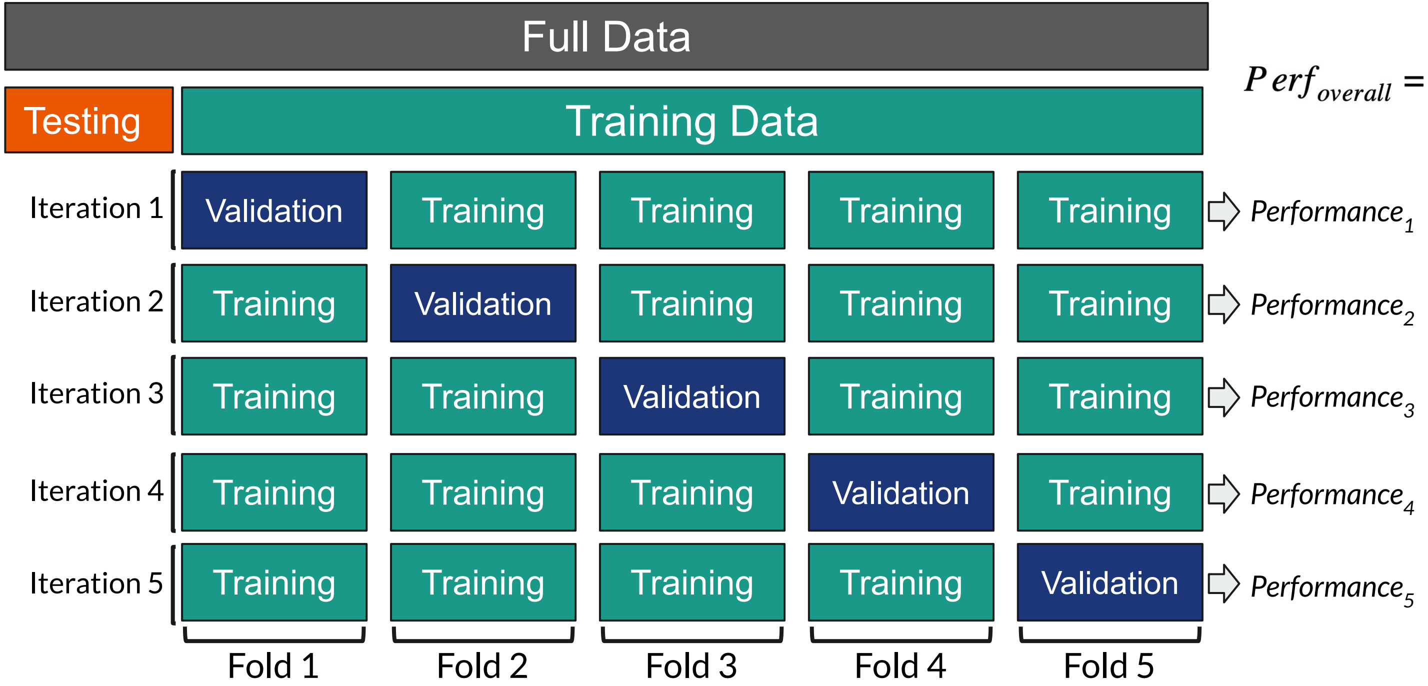


# Validation

- **Validation data** - a set of data used to tune the parameters of a model.
    - In contrast, **testing data** used only to assess the final performance of a fully-specified model.
- However, using a validation set in addition to the training and testing set may present some problems:
    - ending up overfitting to the validation set,
    - having less training data.
- A smarter implementation of the validation concept is **k-fold cross-validation**.



# 5-Fold Cross-Validation



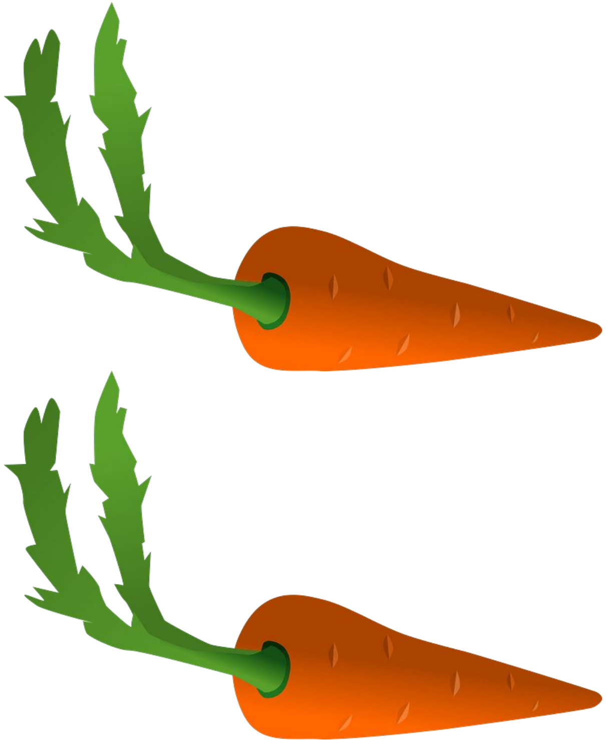

$$Perf_{overall} = \frac{1}{5}\sum_{i=1}^{5} performance_i$$

# Application on COVID-19 Dataset

Can we predict COVID-19 outcome with machine learning and R?

# Machine Learning in R : `caret`

- Acronym for **[C]**lassification **[A]**nd **[RE]**gression **[T]**raining.
- Includes many models:
    - Regression
    - Decision tree
    - Random forest
    - etc.
- It facilitates:
    - Data preprocessing (splitting, sampling, etc.)
    - Feature selection
    - Fitting & prediction.
    - Etc.



# Key `caret` functions

| Function | Description |
|---|---|
| `createDataPartition()` | Creates test/training partitions. |
| `trainControl()` | Controls the settings of model fitting |
| `train()` | Fits the model to the training data. |
| `confusionMatrix()` | Evaluate model performance for classification model |
| `varImp()` | Calculates model importance |

# `createDataPartition()`

- Used to split a dataset into separate training and testing set.
- Returns a vector position integers corresponding to the training data.

| Argument | Description |
|---|---|
| `y` | A vector of outcomes (reponses). |
| `p` | The proportion of data that goes to training. |

```r
# Get indices for training set
train_index =
  createDataPartition(y = mydata$outcomes,
                      p = 0.8,
                      list = FALSE)
```

# `trainControl()`

- Controls how `caret` fits a machine learning model.

```r
# Specify 10 fold cross-validation
ctrl = trainControl(method = "cv",
                    number = 10)
```

| **Argument** | **Description** |
|---|---|
| `method` | The resampling method. Use "cv" for cross-validation. |
| `number` | The number of folds. |

# `train()`

- The fitting workhouse of `caret`.
- Offers more than 200 models.
  - Just change the `method` argument!
  - Find all available models here

| Argument | Description |
|---|---|
| `x` | The features |
| `y` | The outcomes |
| `method()` | The model (algorithm) |
| `trControl()` | Control parameters for fitting |

```r
# Fit a random forest model
rf_model = train(x = features,
                 y = outcomes,
                 method = "rf",
                 trControl =  ctrl)
```

# `confusionMatrix()`

- Creates an confusion matrix.

```r
# Predict on testing set
pred_classes = predict(rf_model, newdata = test_x)

# Evaluate the performance
cm = confusionMatrix(data = pred_classes,
                     reference = factor(test_y))
```

| Argument | Description |
|---|---|
| `data` | Predicted classes. |
| `reference` | Actual classes. |

# Dataset

Data Descriptor | Open Access | Published: 24 March 2020

## Epidemiological data from the COVID-19 outbreak, real-time case information

Bo Xu, Bernardo Gutierrez, Sumiko Mekaru, Kara Sewalk, Lauren Goodwin, Alyssa Loskill, Emily L. Cohn, Yulin Hswen, Sarah C. Hill, Maria M. Cobo, Alexander E. Zarebski, Sabrina Li, Chieh-Hsi Wu, Erin Hulland, Julia D. Morgan, Lin Wang, Katelynn O'Brien, Samuel V. Scarpino, John S. Brownstein, Oliver G. Pybus, David M. Pigott & Moritz U. G. Kraemer

*Scientific Data* **7**, Article number: 106 (2020) | Cite this article



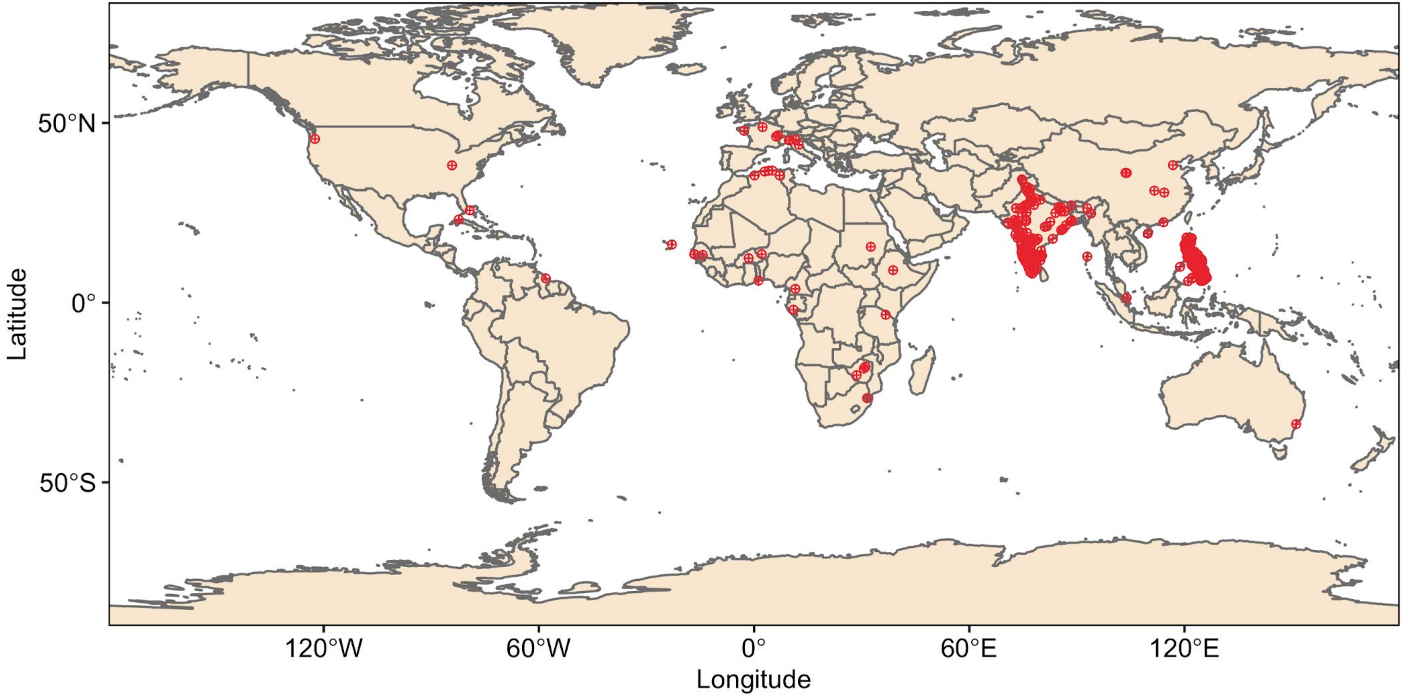
https://doi.org/10.1038/s41597-020-0448-0

# Processed data

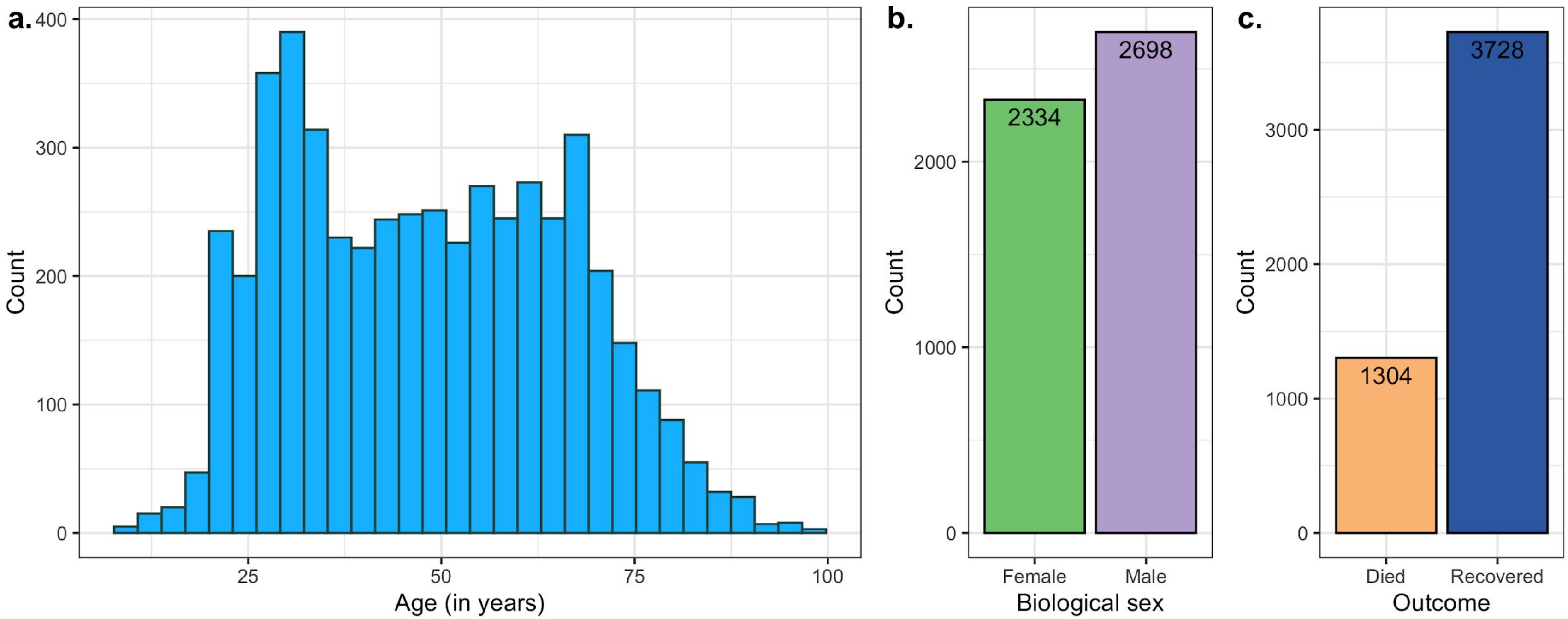
- Each of the rows represents a single individual case.
- Description of fields:
  - **outcome** - patients outcome, either "Died" or "Recovered".
  - **age** - age of the case (in years).
  - **sex** - biological sex of the case, either "Female" or "Male" .
  - **latitude** - the latitude of the location where the case was reported.
  - **longitude** - the longitude of the location where the case was reported
- The processed data is available in: [https://github.com/rsgturkey/Workshop2020](https://github.com/rsgturkey/Workshop2020)

| Response (Y) | Features (X) | | | |
|---|---|---|---|---|
| outcome | age | sex | latitude | longitude |
| Recovered | 28 | Female | 13.469730 | -16.696190 |
| Died | 70 | Male | 13.453056 | -16.577500 |
| Died | 79 | Male | -20.170000 | 28.580000 |
| Died | 56 | Male | 9.030000 | 38.740000 |
| Died | 30 | Male | -17.850000 | 31.050000 |
| Recovered | 52 | Male | -17.850000 | 31.050000 |
| Died | 44 | Male | -17.824390 | 31.049950 |
| Recovered | 77 | Male | 11.816130 | 122.848400 |
| Recovered | 32 | Female | 14.450000 | 120.980000 |
| Recovered | 34 | Female | 14.450000 | 120.980000 |

# The data overview



# The data overview



# Practical