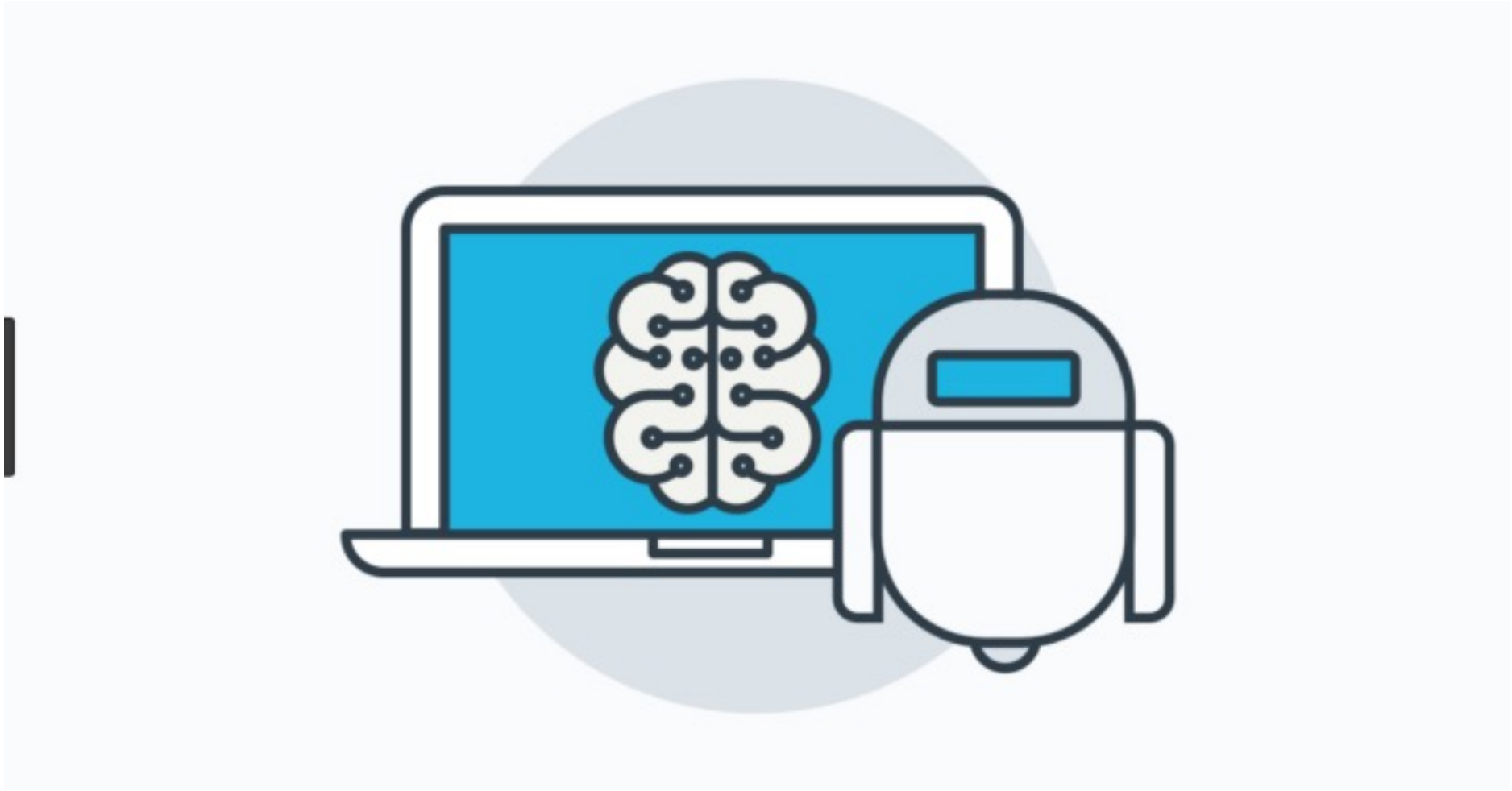
MACHINE LEARNING

Jingjing Yang, PhD
Jingjing.yang@emory.edu
Department of Human Genetics
Emory University School of Medicine

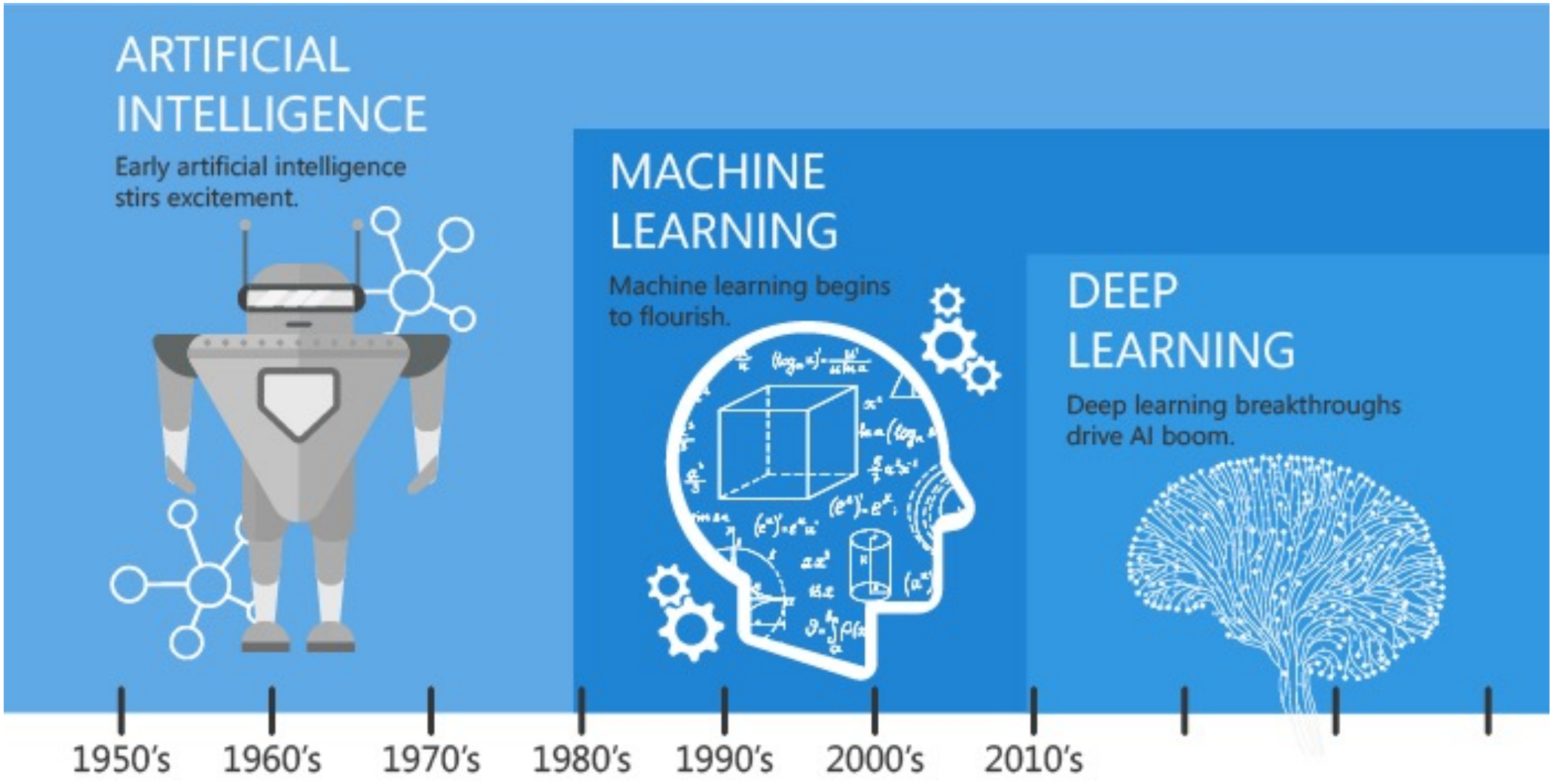
Overview about Machine Learning



Machine Learning

- Prediction
  - Product recommendation
- Image Recognition
  - Face ID
- Speech Recognition
  - Siri
- Medical Diagnoses
  - Risk score for Type 2 Diabetes
- Financial Trading

Quick History about Machine Learning



Since an early flush of optimism in the 1950's, smaller subsets of artificial intelligence - first machine learning, then deep learning, a subset of machine learning - have created ever larger disruptions.

Machine Learning vs. Statistical Learning

- According to **Arthur Samuel**, Machine Learning algorithms enable the computers to learn from data, and even improve themselves, without being explicitly programmed.

- According to "**The Elements of Statistical Learning**", the bible of Statistical Learning, Statistical Learning is referred to using statistical methods to extract important patterns and trends, and understand data that were generated in many fields.

- The intersection of **Computer Science** and **Statistics** gave birth to probabilistic approaches in **Artificial Intelligence**.

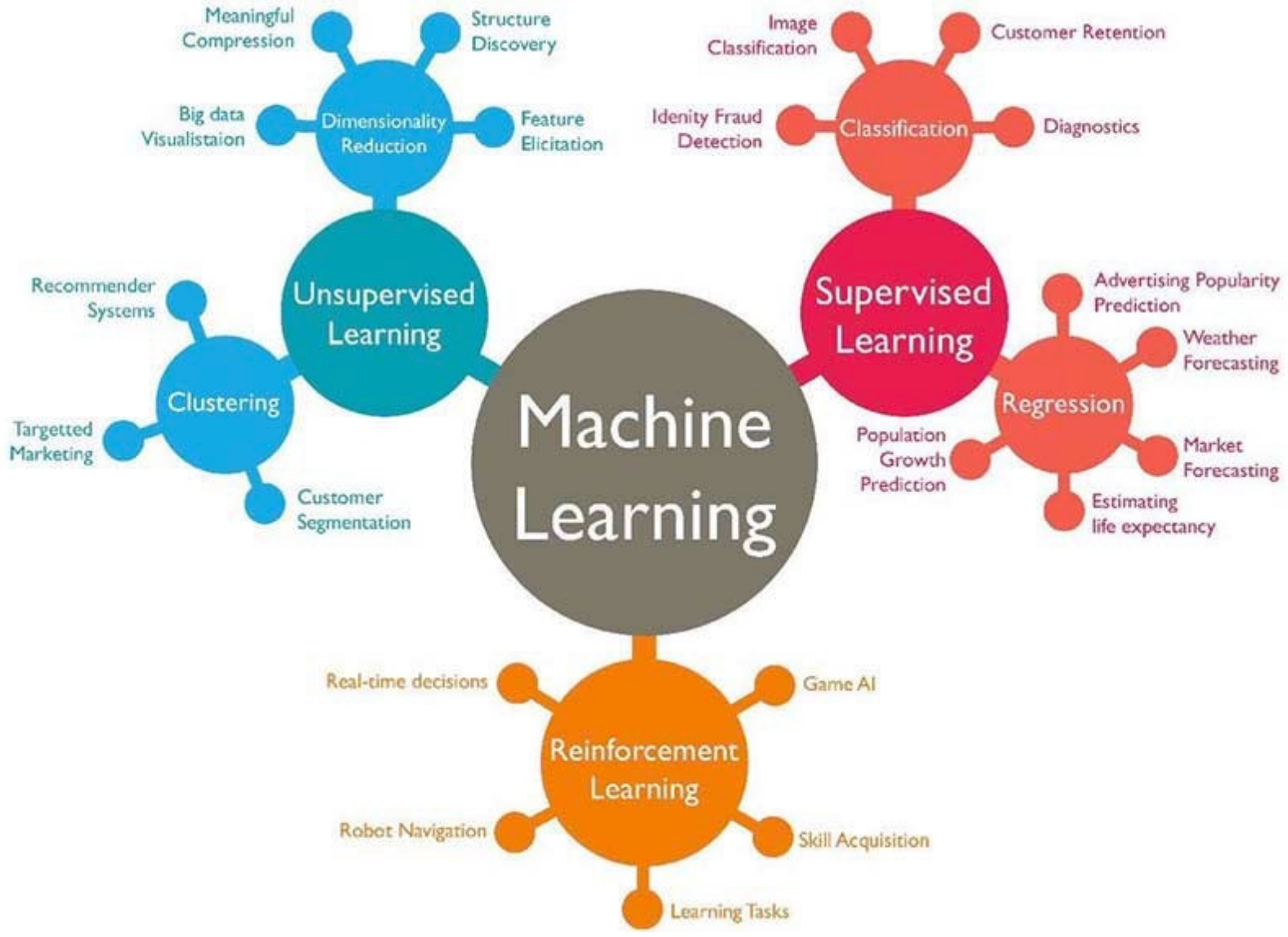
- **Keywords**: Learning from the DATA, Statistical Methods, Computational Algorithms

Machine Learning

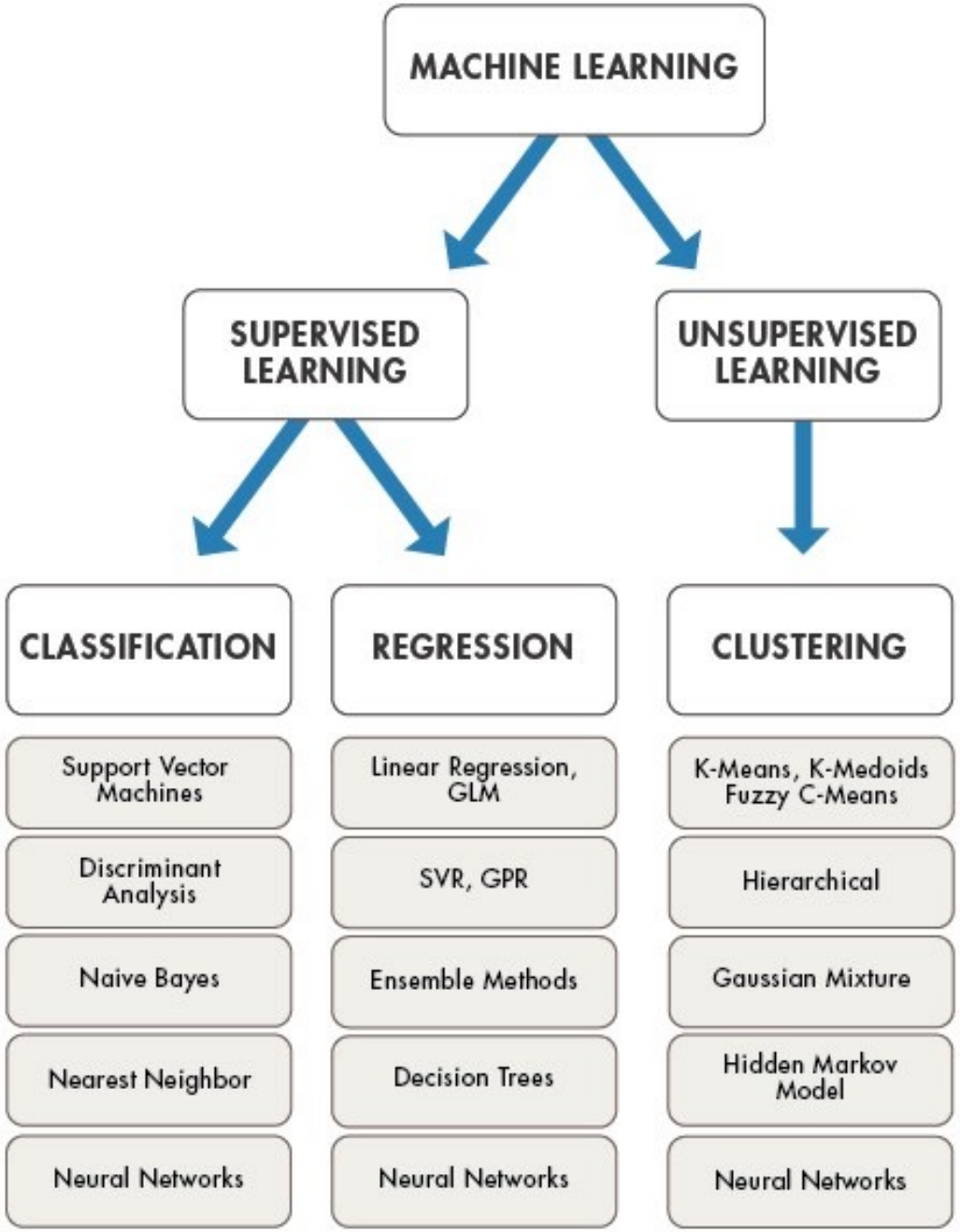
- **Machine Learning** is a category of algorithms that allow software applications to become more accurate in predicting outcomes without being explicitly programmed.

- Basic premise of **Machine Learning** is to build algorithms that can receive input data and use statistical analysis to predict an output while updating outputs as new data becomes available.

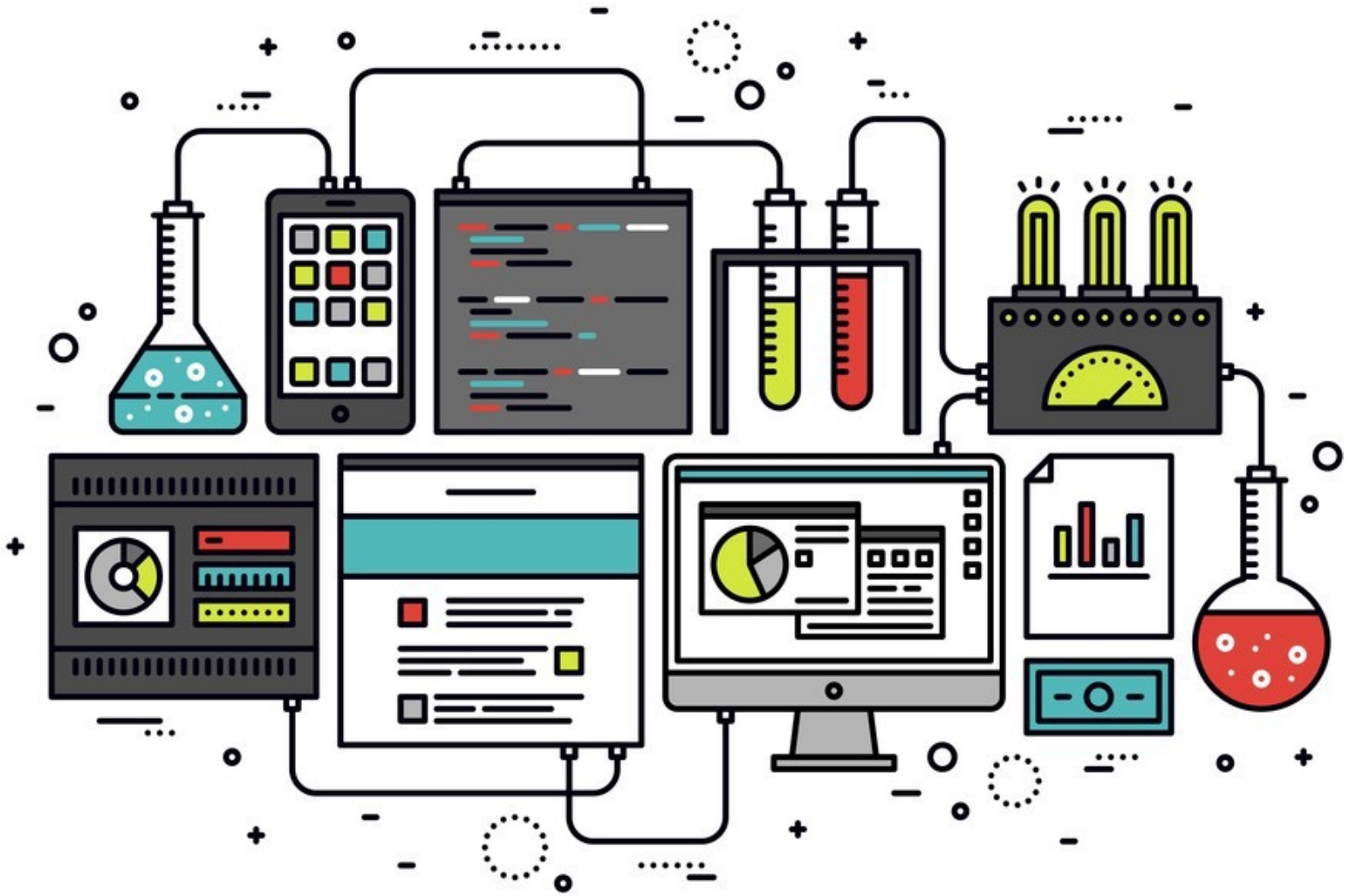
Types of Learning : Supervised, Unsupervised, Reinforcement



Machine Learning Methods



Machine Learning Workflow



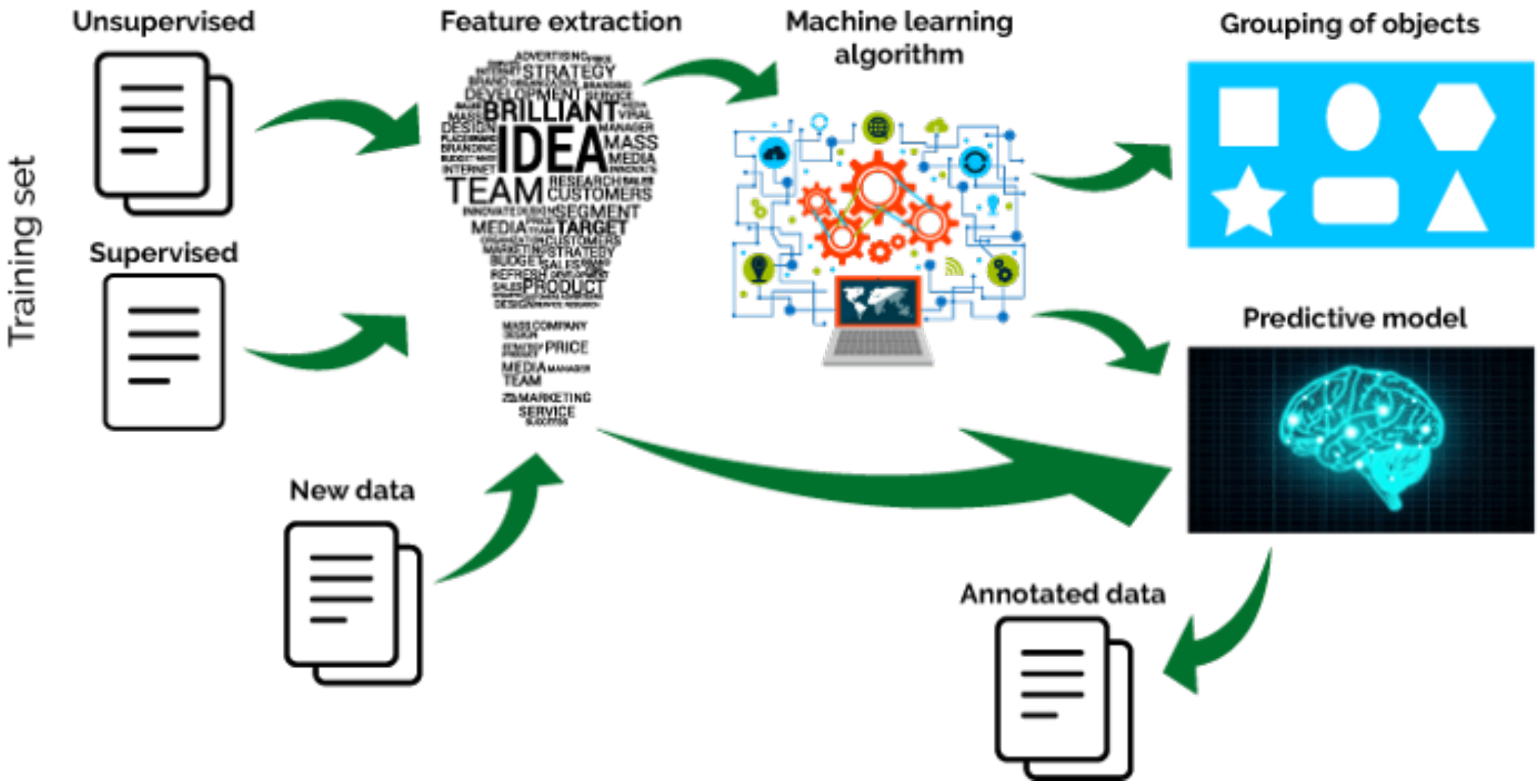
Machine Learning Workflow

- Data Collection
  - Scientific Problem / Hypothesis
  - Experiment Design
- Data Pre-processing
  - Data Cleaning
  - Feature Exploration
- Researching the model that will be best for the type of data
- Training and validating the model
- Evaluation / testing the model

Data Pre-processing (80% time)

- Possible data problems
  - Missing data: Ignoring or Imputing?
  - Noisy data: Excluding or Smoothing?
  - Inconsistent data: Excluding or Correcting?
  - Outliers : Excluding?
- Data types
  - Numeric, e.g., age, height, weight
  - Categorical, e.g., gender, ethnicity; generally coded as 0/1
  - Ordinal, e.g., low/medium/high; generally coded as consecutive numbers such as 0/1/2

Machine Learning



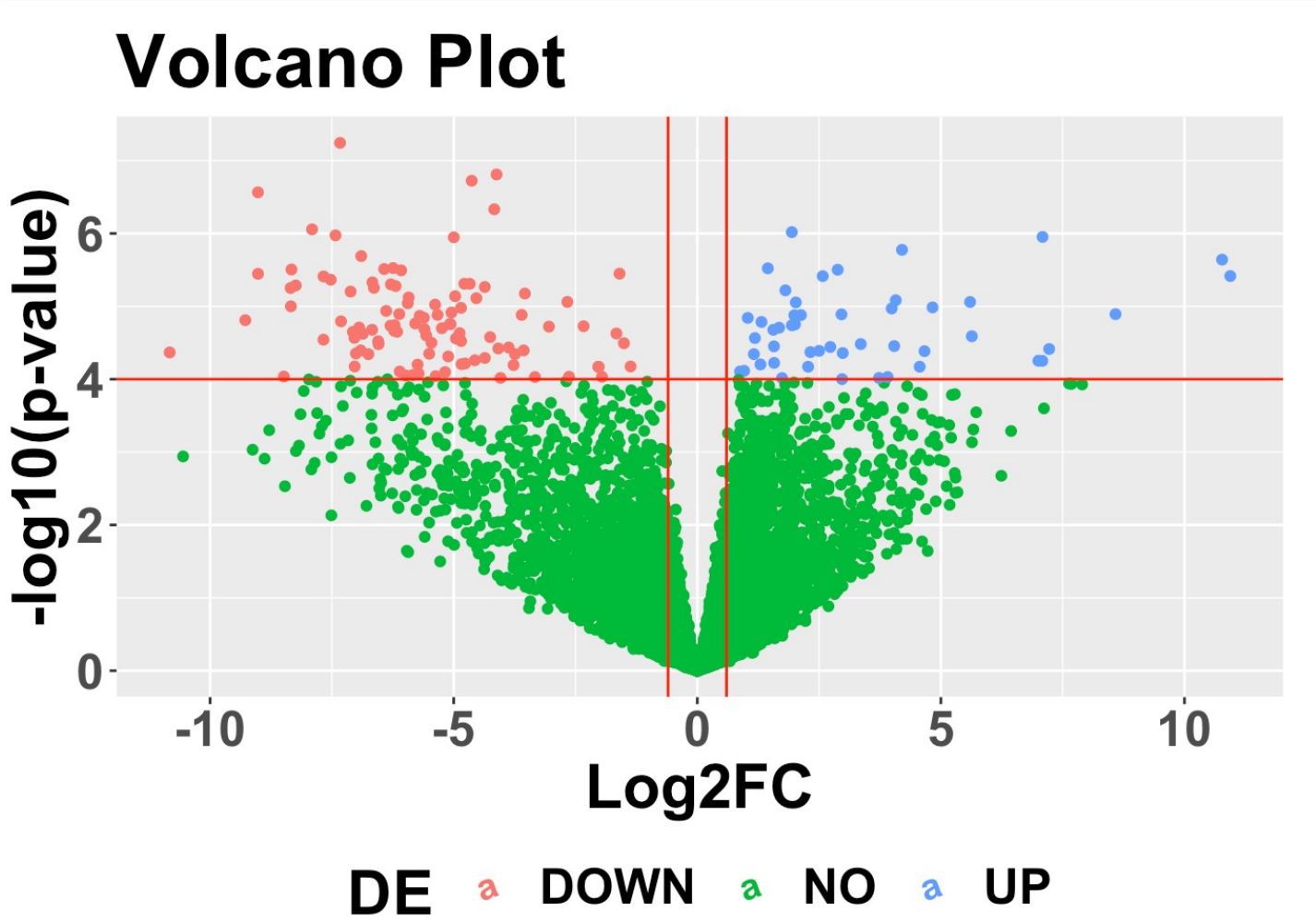
Clustering : Unsupervised Learning

Unsupervised Learning

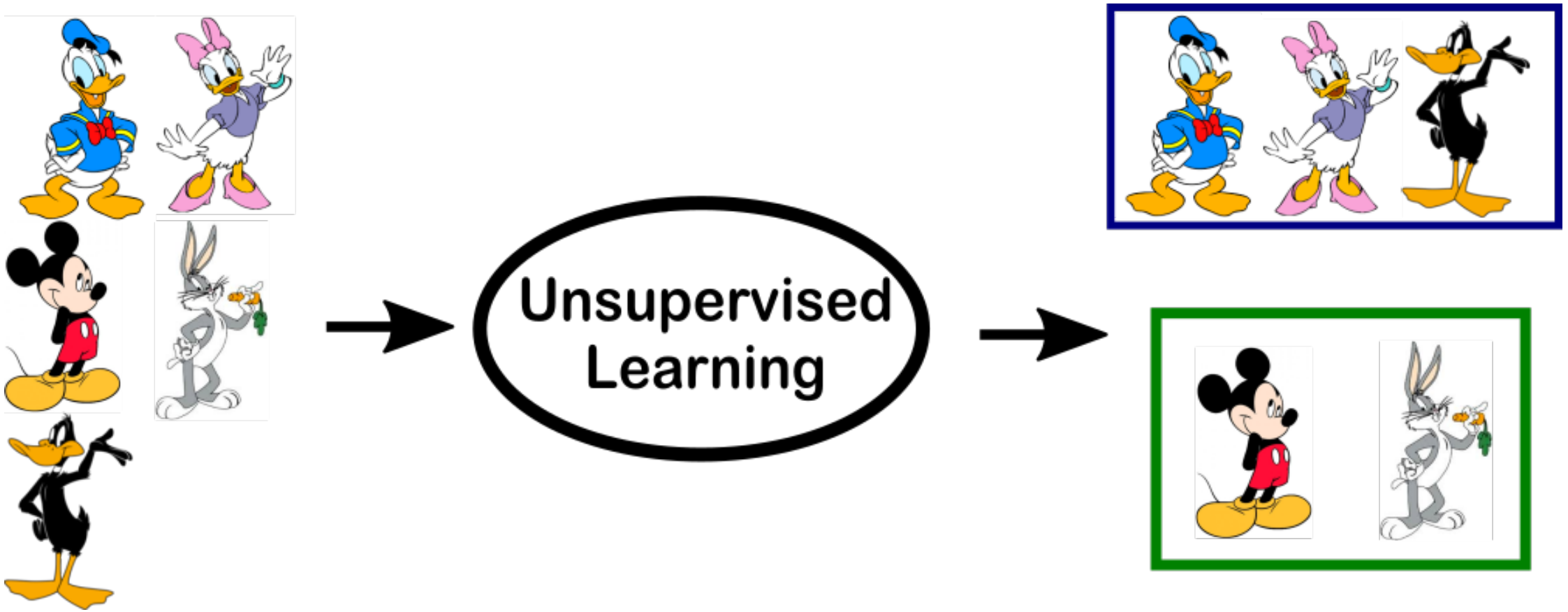
- **Association**: An association rule learning problem is where you want to discover rules that describe large portions of your data
  - Test if people with genetic variation X are more likely to have disease Y
  - Test if a treatment will be effective in clinical trials

- **Clustering**: A clustering problem is where you want to discover the inherent groupings in the data
  - Grouping cells with respect different characteristics

Association Study: Differential Gene Expression Analysis



Clustering : Ducks vs. Not Ducks



Biomedical Research Problems

- Data Quality Assessment
  - Clustering samples according to their ancestry
  - Clustering sequence samples

- Identifying Cell Types
  - Clustering single cells

Clustering Methods

- Uniform Manifold Approximation and Projection (UMAP)
  - Dimension reduction. Projecting high dimensional features to 2-dimension
  - UMAP preserves more of the global structure with superior run time performance.
  - Widely used in RNAseq studies
- Hierarchical Clustering
  - Build a hierarchy from the bottom-up, and doesn't require us to specify the number of clusters beforehand.
  - Put each data point in its own cluster.
  - Identify the closest two clusters and combine them into one cluster.
  - Repeat the above step till all the data points are in a single cluster.

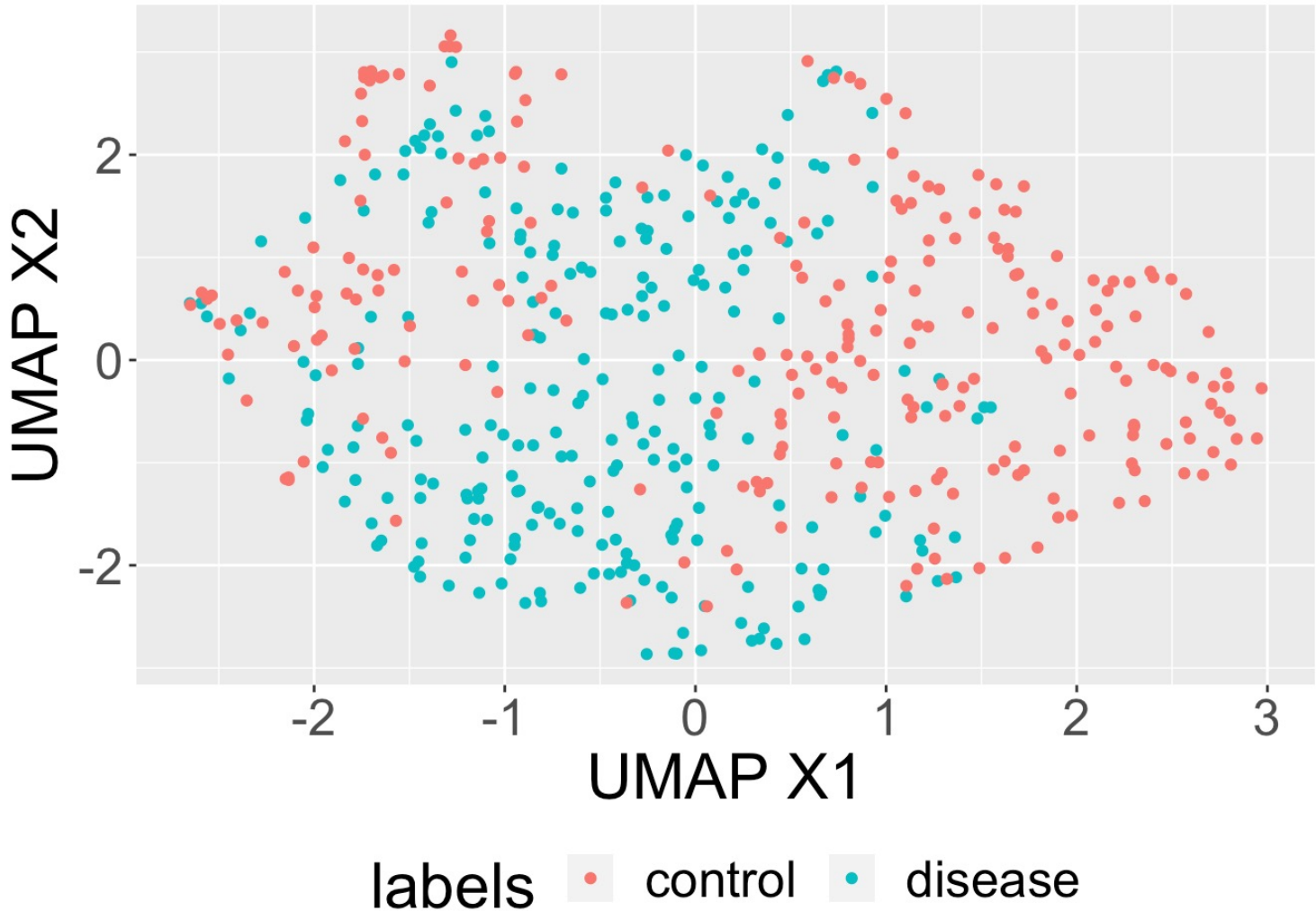
Example RNA-Seq Data

```
head(mRNAexp_data)
```

```
##         Sample_ID Sample_Label   KRAS   HRAS   BRAF   RAF1 MAP3K1 MAP3K2 MAP3K3
## 1 TCGA-05-4244-01      disease -0.248  0.515  1.083  0.846  1.050  1.010  1.851
## 2 TCGA-05-4249-01      disease -0.786  1.400  0.479 -0.353  0.366  0.299  0.402
## 3 TCGA-05-4250-01      disease  2.431 -0.607 -0.231  0.881  0.454 -2.143  1.014
## 4 TCGA-05-4382-01      disease -1.848 -0.476 -1.437  0.328  0.872  0.877  1.252
## 5 TCGA-05-4384-01      disease -1.457  0.991  0.511  0.525  1.421 -3.411 -0.647
## 6 TCGA-05-4389-01      disease  1.571  0.837  0.943 -0.185  2.569  0.036  0.638
##   MAP3K4 MAP3K5 MAP2K1  MAP2K2  MAP2K3  MAP2K4  MAP2K5   MAPK1   MAPK3    MAPK4
## 1  0.416  0.461  0.078 -0.0933 -1.5615  0.1916 -1.9091 -2.3858 -1.0441  -8.0643
## 2 -1.193 -0.234  0.752  0.7079 -0.6895  0.3511 -1.0612 -1.5409 -2.7780  -0.3124
## 3  0.126 -1.673  0.549 -1.1156 -1.8050 -2.1712 -2.2206 -3.2568 -0.3711  -7.1490
## 4 -0.936 -0.603  0.560  0.5942 -0.6349 -1.2071 -2.8627 -1.1870 -2.8349 -10.4816
## 5 -0.230  0.532  1.098 -1.9015 -0.8264  2.2097  1.0364 -0.0773 -0.9921  -4.2445
## 6  0.680  0.212 -1.723  1.2524 -1.5176 -1.9543  0.9599 -1.6332 -0.1284  -4.3858
##    MAPK6   MAPK7  MAPK8   MAPK9  MAPK12 MAPK14    DAB2  RASSF1  RAB25
## 1 0.4812 -2.1086 1.5096  2.1246 -0.0214 0.9591 -4.1024 -2.3154 3.4180
## 2 1.4195 -1.4247 3.5736 -0.4379 -0.6093 1.9235 -3.4054 -2.5505 4.0763
## 3 2.6576 -0.2633 3.5895 -1.1063  1.7538 1.4769 -3.1768 -2.0182 3.7324
## 4 1.4538  2.2449 0.9217 -2.4182 -0.3514 0.1317  0.2466 -0.3413 1.9215
## 5 0.3312 -0.3324 0.6528 -0.6446  0.8338 0.2478 -1.4205 -2.8088 4.4442
## 6 1.2589 -1.9748 3.2084 -0.9054 -0.4110 2.3225 -2.0756 -2.4175 5.4549
```

- Z-scores relative to normal samples
- 500 tumor samples
- 26 genes
- 2 groups: disease, control (simulated Sample_Label)

Clustering by UMAP



Hierarchical Clustering

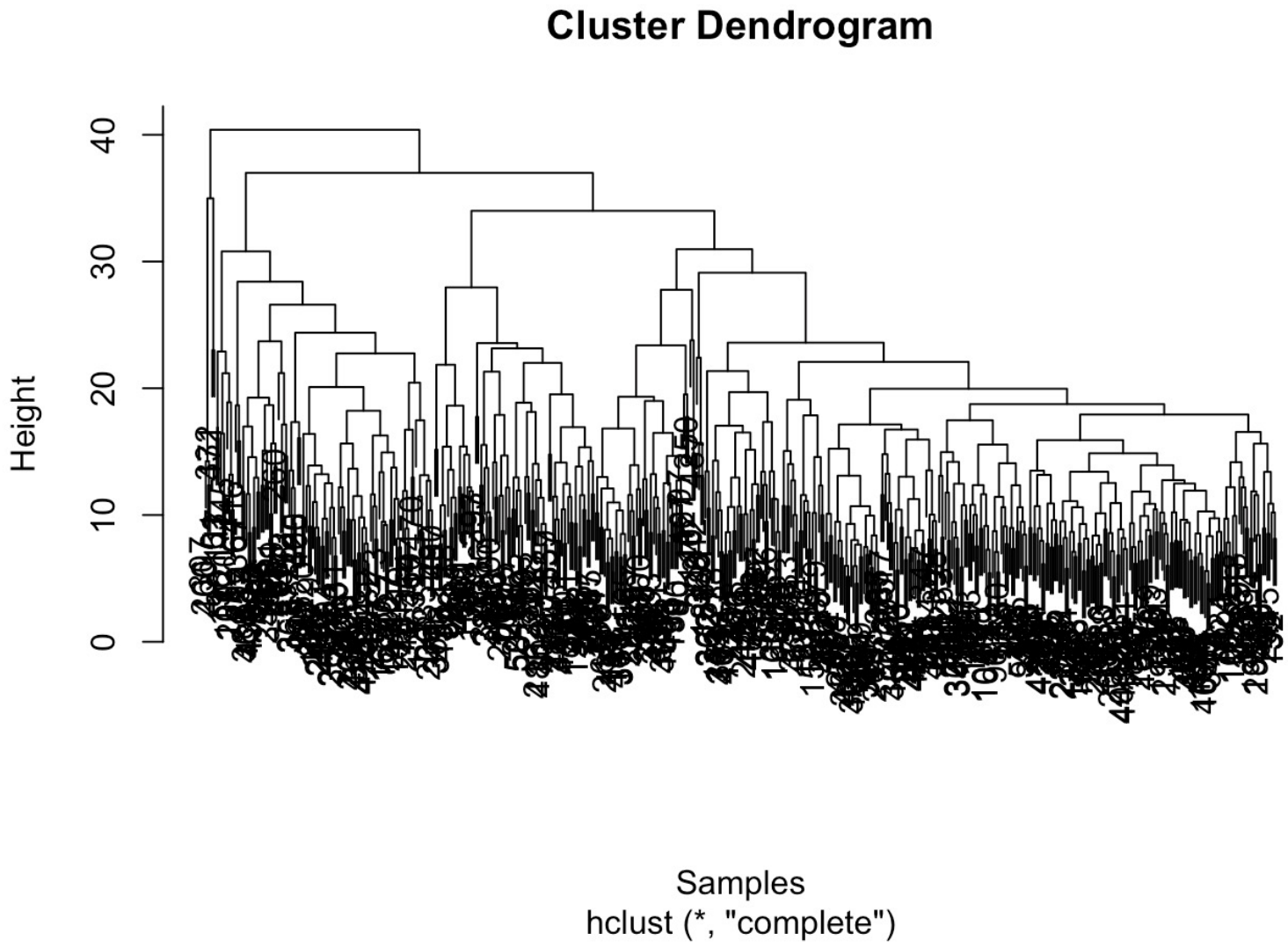
There are a few ways to determine how close two clusters are:

- Complete linkage clustering: Find the maximum possible distance between points belonging to two different clusters.

- Mean linkage clustering: Find all possible pairwise distances for points belonging to two different clusters and then calculate the average.

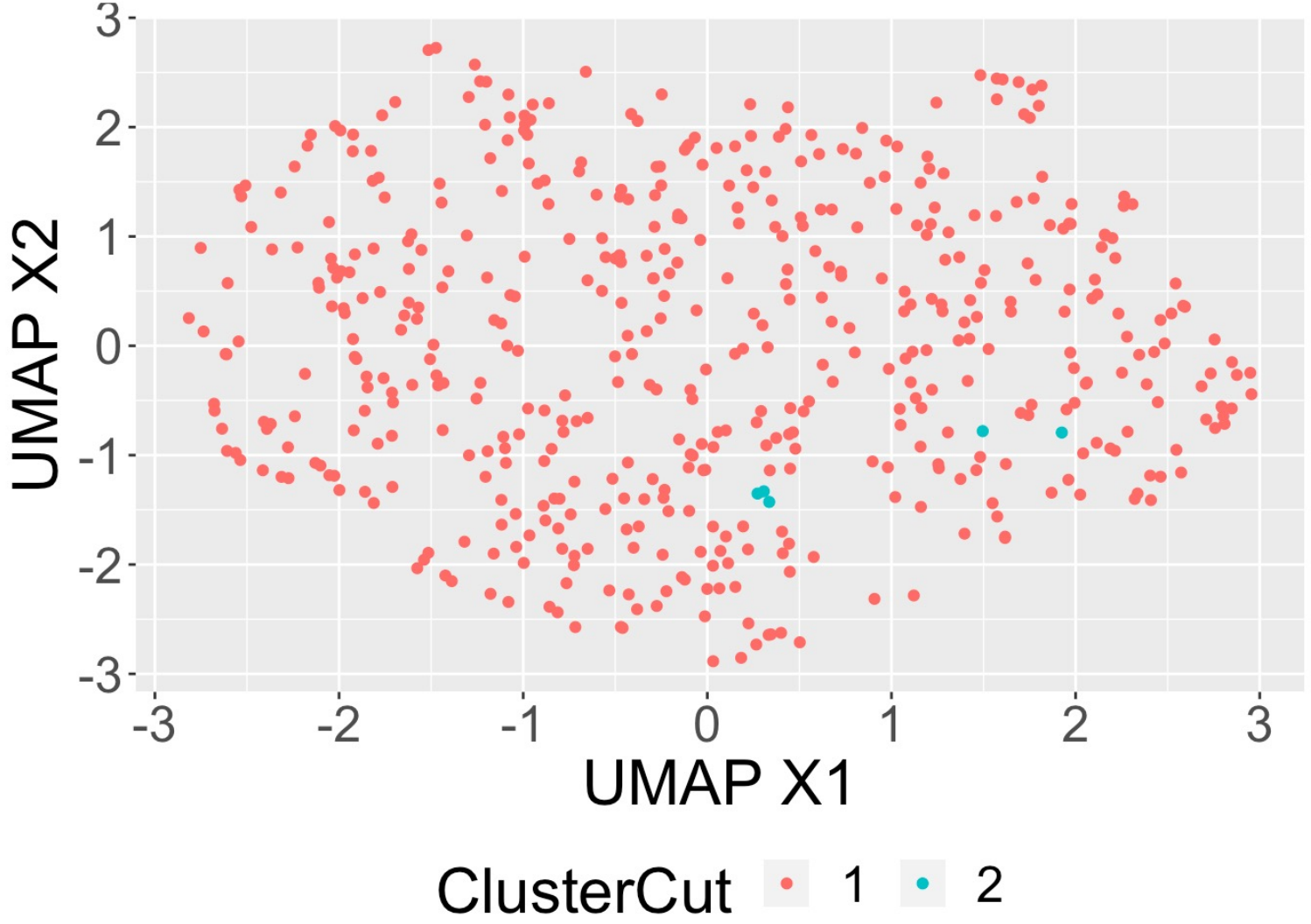
Complete linkage and **mean linkage** clustering are the ones used most often.

Hierarchical Clustering with Complete Linkage




```
clusters <- hclust(dist(mRNAexp_data[, 3:28]), method = "complete")
plot(clusters, xlab = "Samples")
```

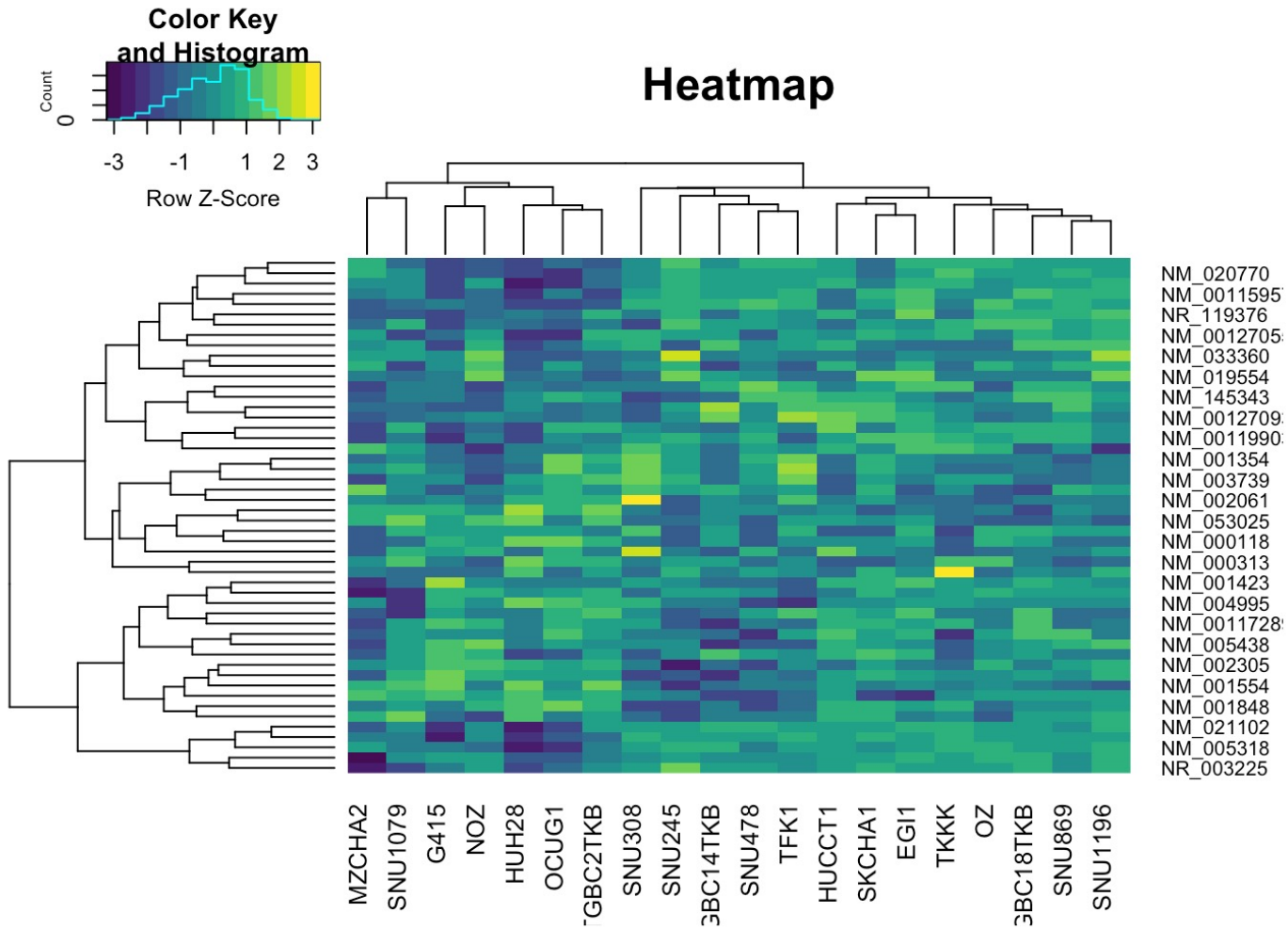
Hierarchical Clustering with Complete Linkage




```
ggplot(data.frame(mRNA.umap$layout, ClusterCut = factor(clusterCut) ),
       aes(x = X1, y = X2, colour = ClusterCut)) +
    geom_point() + labs(x = "UMAP X1", y = "UMAP X2")
```

Heatmap by *hvlcpm()*

```
hvlcpm <- RNAseq_CPM.keep.log2[select_genes, ]
gplots::heatmap.2(hvlcpm,
          col=viridis,
          trace="none",
          main="Heatmap",
          scale="row")
```

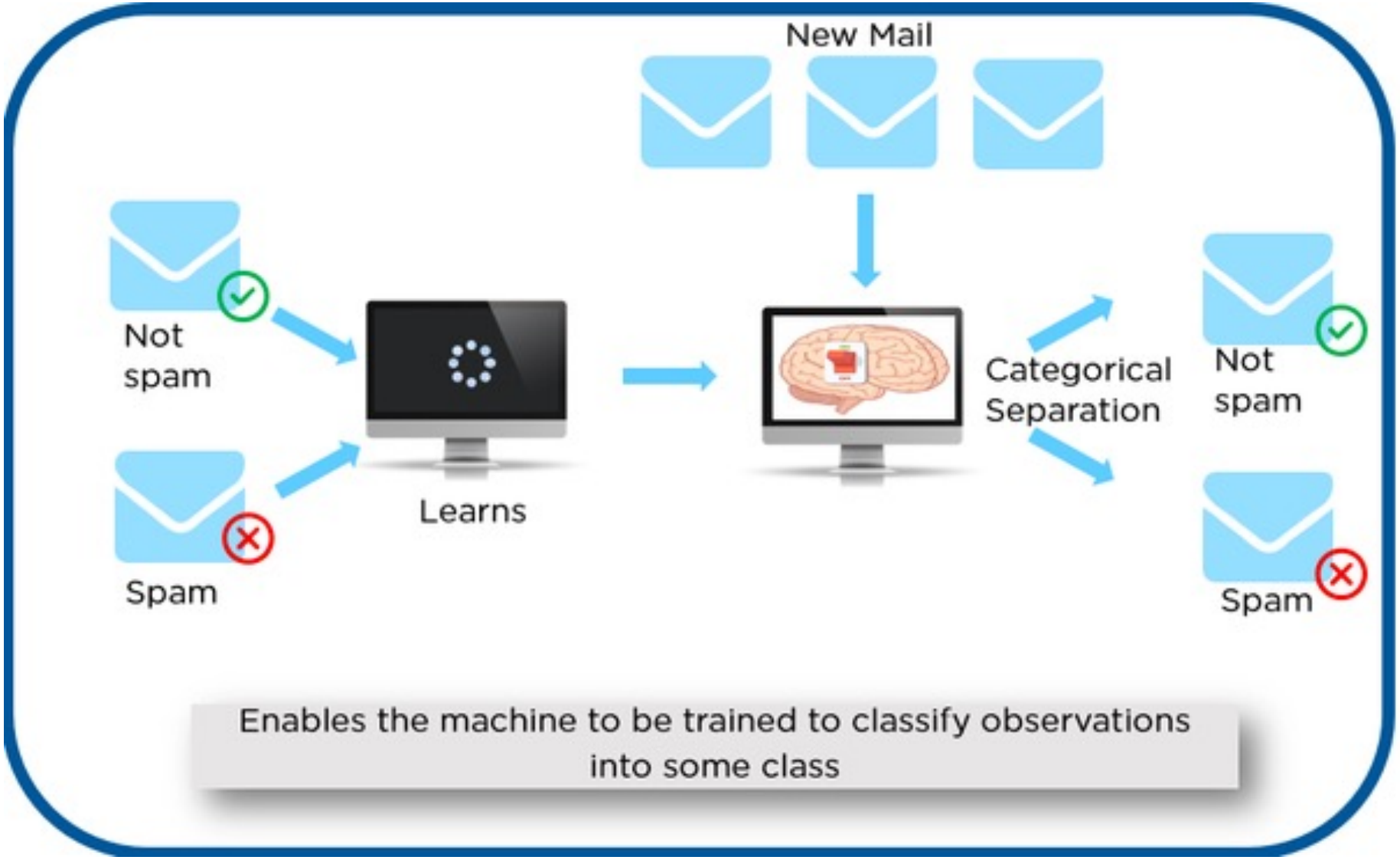


Supervised Learning

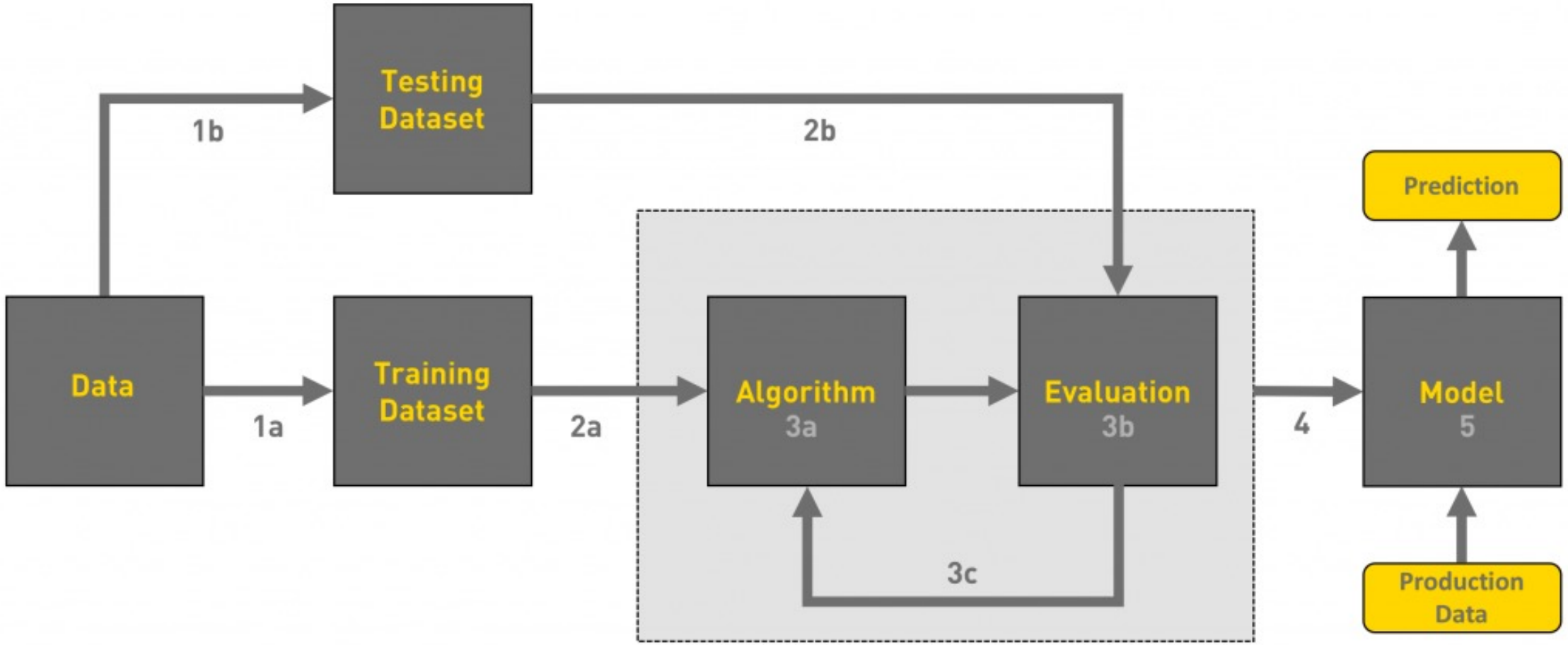
- **Classification**
  - A classification problem is when the output variable is a category, such as “disease” or “no disease”.
- **Regression**
  - A regression problem is when the output variable is a real measured value, such as “weight”, “BMI”, “blood pressure”.
  - Regression analysis is a form of predictive modelling technique which investigates the relationship between dependent and independent variables.

Classification : Supervised Learning

Classification



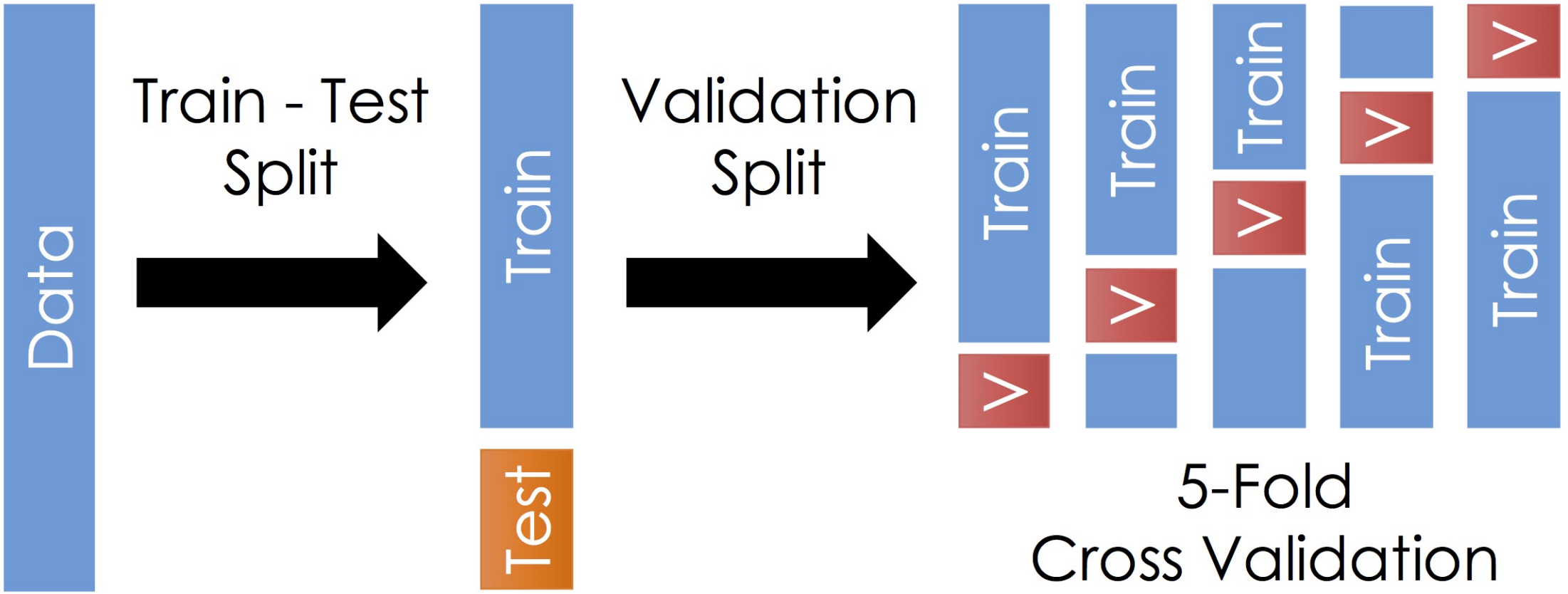
Workflow


Testing Dataset
1b
2b
Prediction
Data
Training Dataset
1a
2a
Algorithm
3a
Evaluation
3b
4
Model
5
3c
Production Data

Machine Learning Data Structure

- Training set: Data used for learning, that is to fit the parameters of the classifier/model.
- Validation set: Independent data (different from the training data) used to tune the parameters of a classifier/model (cross-validation is primarily used).
- Test set: Independent data (different from the training and validation data) used only to assess the performance of a fully-trained classifier.

Cross Validation



Tunning Parameters & Model Selection

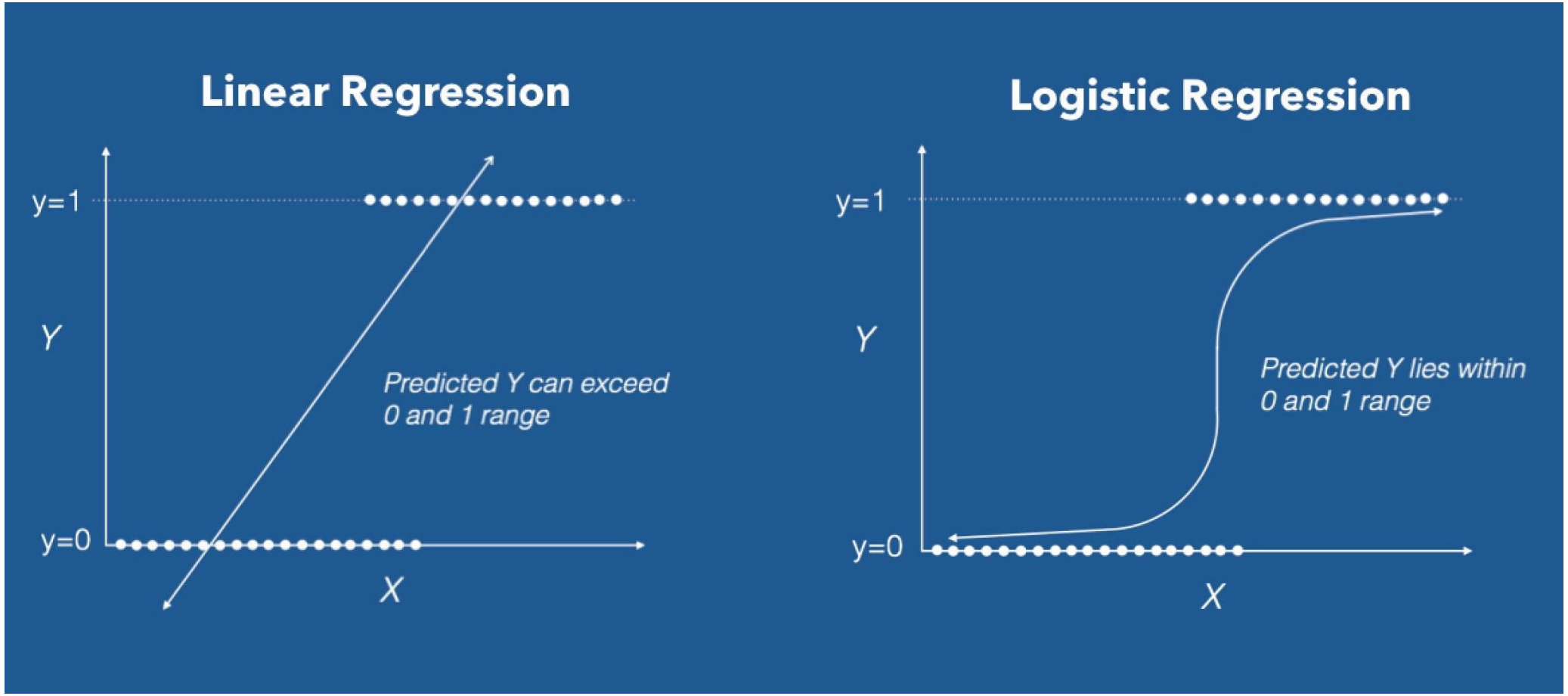
- Tunning Parameters
  - Train a set of models with respect to a range of parameters
  - Use validation data to select best parameters leading to the best performance

- Model Selection
  - Train multiple models with respect to different settings
    - For example, different sets of predictive features might be considered
    - Different methods/models might be considered
  - Use test data to select a best model with best performance

Classification Method

- **Logistic Regression** (Generalized linear regression model with binary responses)
  - https://en.wikipedia.org/wiki/Logistic_regression

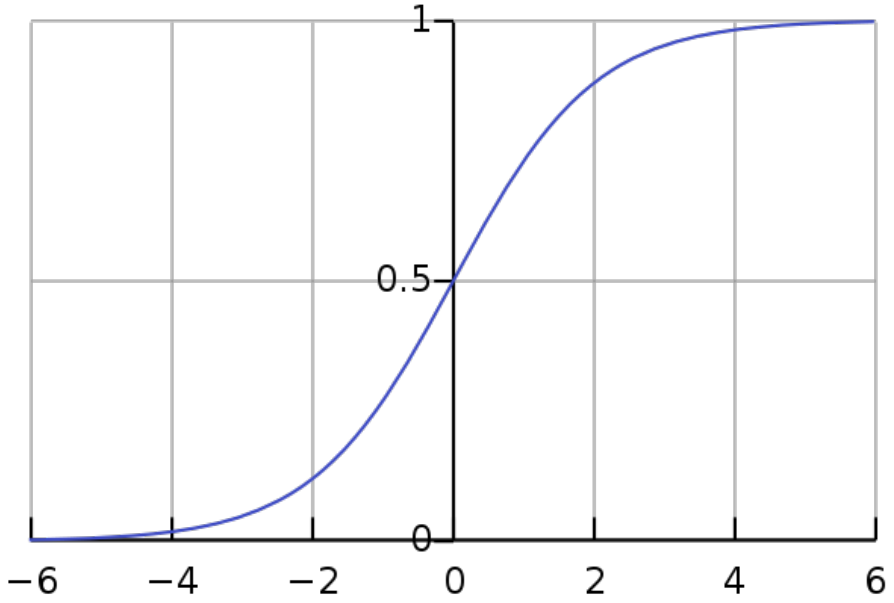
Logistic Regression



Logistic Regression

$$l_{\text{LogOdds}} = \log\left(\frac{p}{1-p}\right) = X\beta$$

- $p = Prob(Y = 1)$
- $p = \frac{1}{1+e^{-X\beta}} = \sigma(X\beta)$, Sigmoid function of $X\beta$



Elastic-Net Penalized Regression

- Penalized regression with a combined L1 penalty (LASSO) and L2 penalty (Ridge) on coefficients

$$\min_{\beta_0,\beta} \frac{1}{N} \sum_{i=1}^{N} w_i l(y_i, \beta_0 + \beta^T x_i) + \lambda \left[(1-\alpha)||\beta||_2^2/2 + \alpha||\beta||_1 \right],$$

- Variable Selection for using L1 penalty (LASSO)
- Account for Highly Correlated variables (co-linearility) for using L2 penalty (Ridge)
- Need to tune penalty parameters $\boldsymbol{\lambda}, \boldsymbol{\alpha}$ by Cross Validation
- β_0, β will be estimated by using the above objective function for each unique pair of parameter values of $\boldsymbol{\lambda}, \boldsymbol{\alpha}$

Elastic-Net Penalized Regression

- R package “glmnet”
  - Fits a generalized linear model via penalized maximum likelihood. The regularization path is computed for the lasso or elastic-net penalty at a grid of values for the regularization parameter lambda.
  - The algorithm is extremely fast, and can exploit sparsity in the input matrix x.
  - It fits linear, logistic and multinomial, Poisson, and Cox regression models.
  - https://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html#top

Model Evaluation

- Test model performance using a test data set that is independent of the training/validation data sets

- Evaluation criteria
  - Classification
    - Misclassification Rate:

$$Misclassification\_Rate = \frac{FalsePositives + FalseNegatives}{N}$$

    - ROC/AUC

  - Regression
    - Mean Square Error:

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

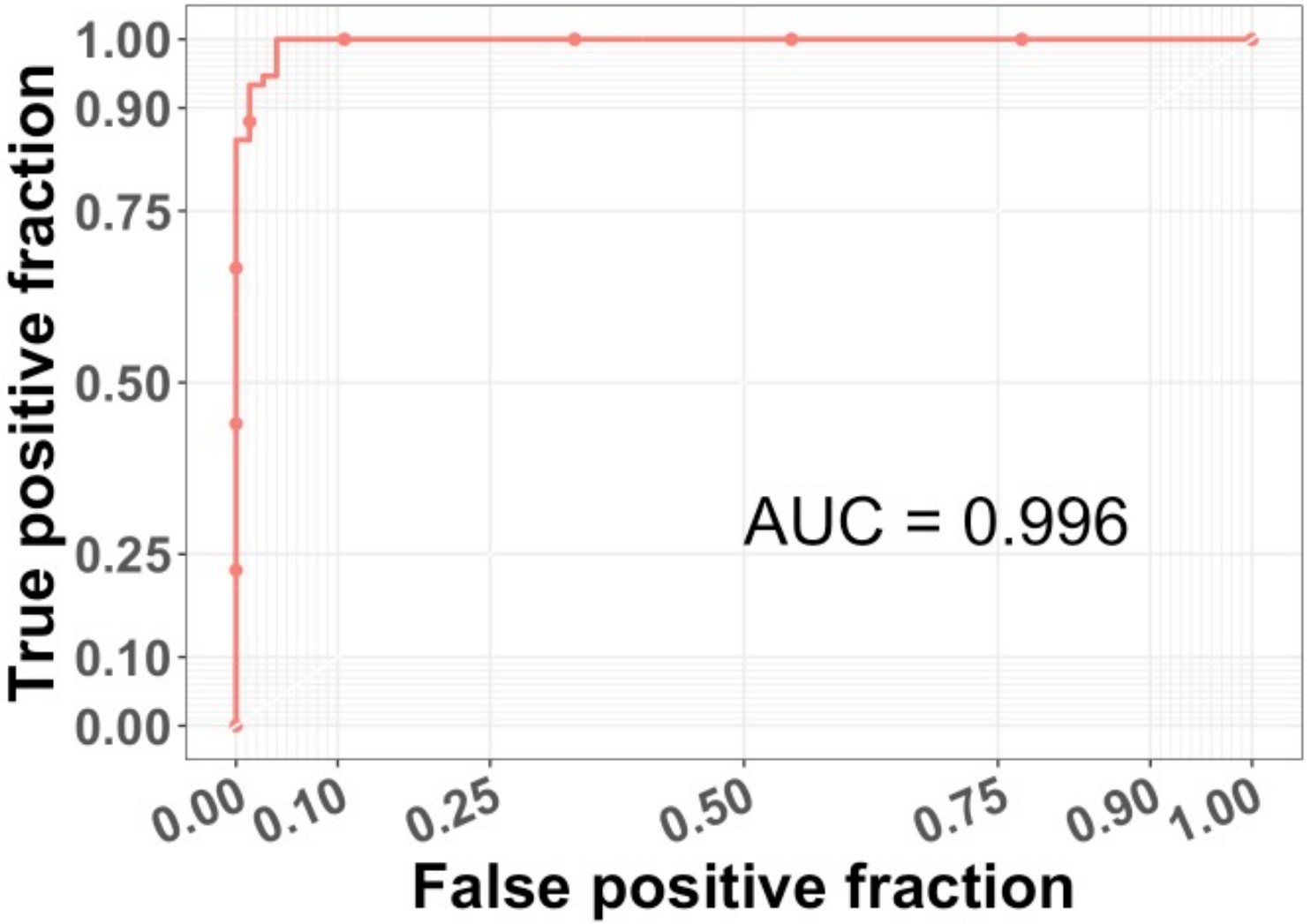
Confusion Matrix for Two-group Classification

		True condition				
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$	F_1 score $= 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$		

ROC Curve

- **Receiver operating characteristic (ROC)** curve is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.
- Plot **True Positive Rate** (TPR, sensitivity, recall rate, probability of detection, power) against the **False Positive Rate** (FPR, 1-specificity, probability of false alarm, type I error) **at various threshold settings**.
- **Area under the curve (AUC, C statistic),** the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative').
- https://en.wikipedia.org/wiki/Receiver_operating_characteristic

Example ROC Plot



Example using R package *caret*

R package *caret*

- The **caret** package (**C**lassification **A**nd **RE**gression **T**raining) is a set of functions that attempt to streamline the process for creating predictive models.
- Integrates almost all Machine Learning models
- The package contains tools for:
  - data splitting (training vs. test)
  - pre-processing (quality control, imputing missing values)
  - feature selection
  - model tuning using resampling
  - variable importance estimation (R function “varImp()”)
- https://topepo.github.io/caret/index.html

Partition Training and Test Data

```
set.seed(2022)
trainIndex_2class <- createDataPartition(mRNAexp_data$Sample_Label, p = .7,
                                         list = FALSE,
                                         times = 1)
head(trainIndex_2class)
```

```
##      Resample1
## [1,]         1
## [2,]         4
## [3,]         6
## [4,]         7
## [5,]         8
## [6,]        10
```

Setup Arguments for Model Training

```
## set model training parameters
fitControl <- trainControl(## 10-fold CV
                           method = "cv",
                           number = 10,
                           ## Estimate class probabilities
                           classProbs = TRUE,
                           ## Evaluate performance using the following function
                           summaryFunction = twoClassSummary)

### Set `sample_labels` variable as factors
mRNAexp_data$Sample_Label = as.factor(mRNAexp_data$Sample_Label)
```

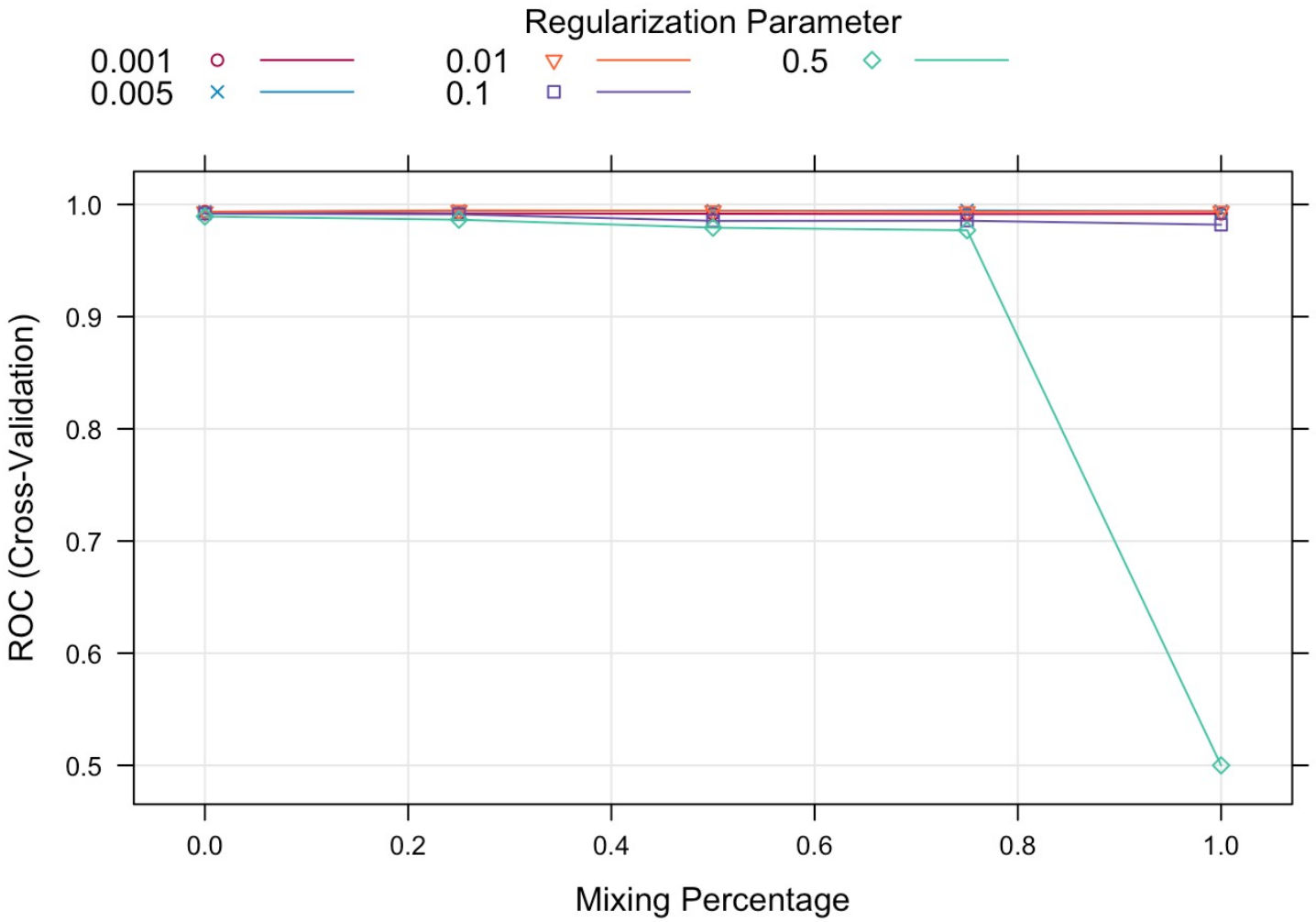
Train the classification model by "glmnet" method

```
## Train the classification model by "glmnet" method
glmnet.fit <- train(Sample_Label ~ ., data = mRNAexp_data[trainIndex_2class, c(2, 3:28)],
                    method = "glmnet",
                    trControl = fitControl,
                    preProc = c("center", "scale"),
                    ## set tuning parameter grid
                    tuneGrid = expand.grid(alpha = seq(0, 1, length.out = 5),
                           lambda = c(0.005, 0.001, 0.01, 0.1, 0.5)),
                    ## Specify which metric to optimize
                    metric = "ROC")
print(glmnet.fit)
```

Trained classification model by "glmnet" method

```
## glmnet
##
## 350 samples
##  26 predictor
##   2 classes: 'control', 'disease'
##
## Pre-processing: centered (26), scaled (26)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 314, 316, 316, 315, 314, 315, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda  ROC        Sens       Spec
##   0.00   0.001   0.9935751  0.9709150  0.9771242
##   0.00   0.005   0.9935751  0.9709150  0.9771242
##   0.00   0.010   0.9935751  0.9709150  0.9771242
##   0.00   0.100   0.9919198  0.9535948  0.9771242
##   0.00   0.500   0.9893011  0.9310458  0.9771242
##   0.25   0.001   0.9920693  0.9598039  0.9604575
##   0.25   0.005   0.9943793  0.9653595  0.9715686
##   0.25   0.010   0.9947627  0.9712418  0.9826797
##   0.25   0.100   0.9912651  0.9248366  0.9885621
##   0.25   0.500   0.9864603  0.8957516  0.9888889
##   0.50   0.001   0.9917606  0.9653595  0.9660131
##   0.50   0.005   0.9944167  0.9712418  0.9660131
##   0.50   0.010   0.9943793  0.9653595  0.9826797
##   0.50   0.100   0.9854212  0.9137255  0.9888889
##   0.50   0.500   0.9793189  0.9019608  0.9826797
##   0.75   0.001   0.9914146  0.9653595  0.9601307
##   0.75   0.005   0.9947253  0.9712418  0.9826797
##   0.75   0.010   0.9936873  0.9653595  0.9830065
##   0.75   0.100   0.9855344  0.9133987  0.9888889
##   0.75   0.500   0.9770291  0.8790850  0.9826797
##   1.00   0.001   0.9916859  0.9653595  0.9715686
##   1.00   0.005   0.9940333  0.9712418  0.9885621
##   1.00   0.010   0.9939959  0.9594771  0.9830065
##   1.00   0.100   0.9819920  0.9133987  0.9771242
##   1.00   0.500   0.5000000  0.5000000  0.5000000
##
## ROC was used to select the optimal model using the largest value.
## The final values used for the model were alpha = 0.25 and lambda = 0.01.
```

Parameter Tuning Results

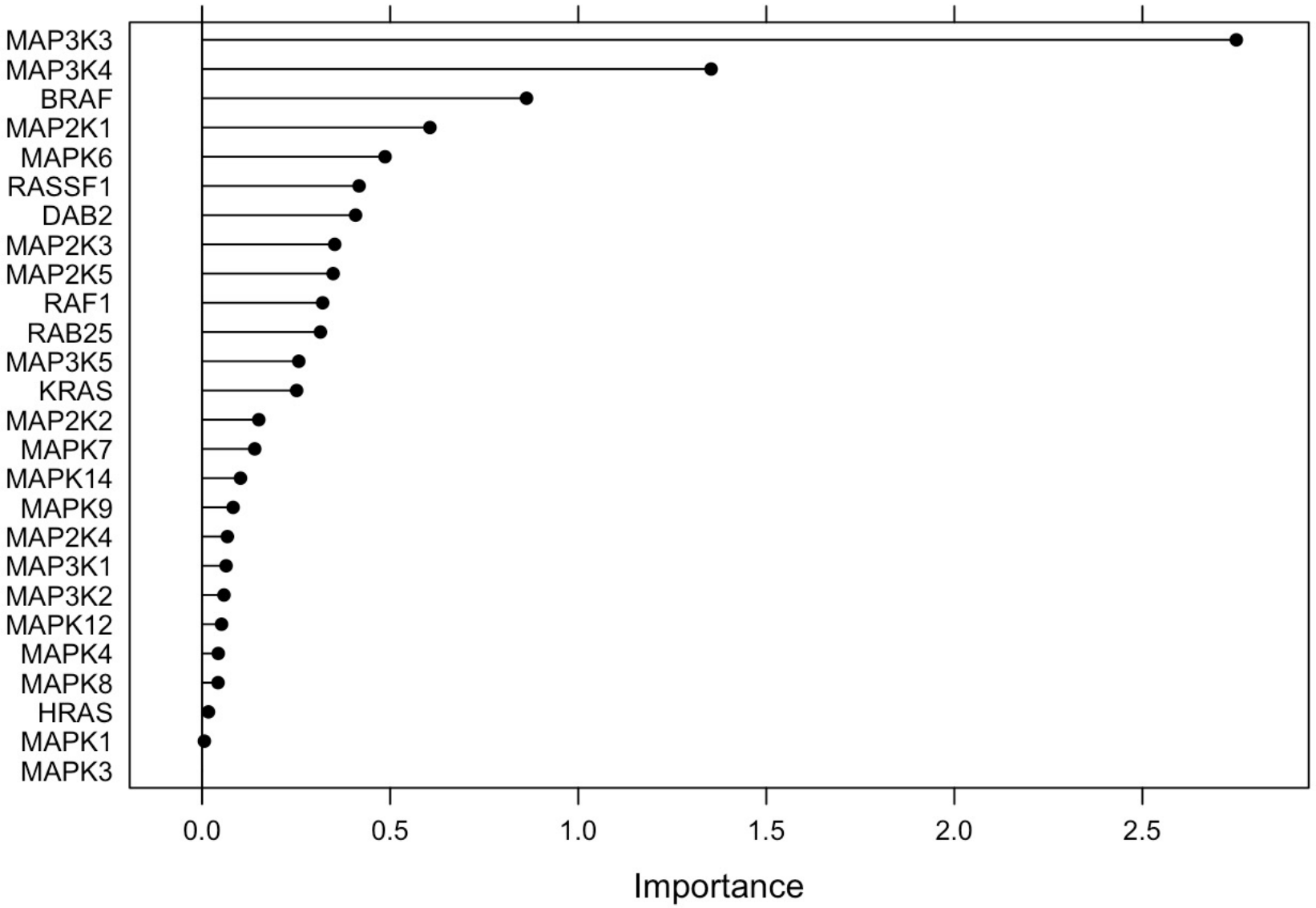


Predictor Importance

```
IMP_glmnet <- varImp(glmnet.fit, scale = FALSE)
print(IMP_glmnet)
```

```
## glmnet variable importance
##
##   only 20 most important variables shown (out of 26)
##
##         Overall
## MAP3K3 2.74960
## MAP3K4 1.35334
## BRAF   0.86255
## MAP2K1 0.60575
## MAPK6  0.48642
## RASSF1 0.41740
## DAB2   0.40803
## MAP2K3 0.35265
## MAP2K5 0.34842
## RAF1   0.32053
## RAB25  0.31465
## MAP3K5 0.25709
## KRAS   0.25140
## MAP2K2 0.15083
## MAPK7  0.14000
## MAPK14 0.10208
## MAPK9  0.08250
## MAP2K4 0.06724
## MAP3K1 0.06391
## MAP3K2 0.05818
```

Predictor Importance

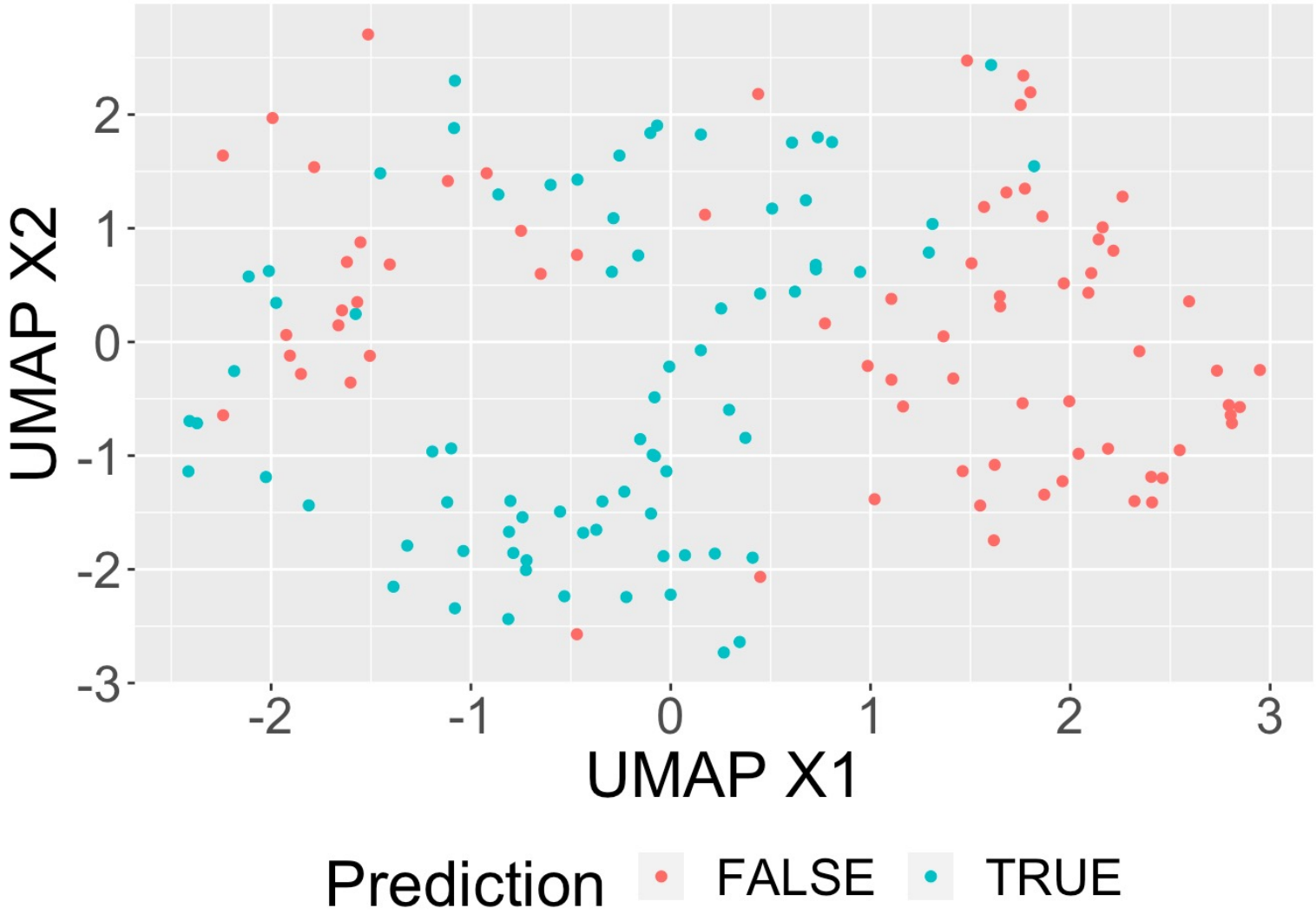


Prediction in Test Data

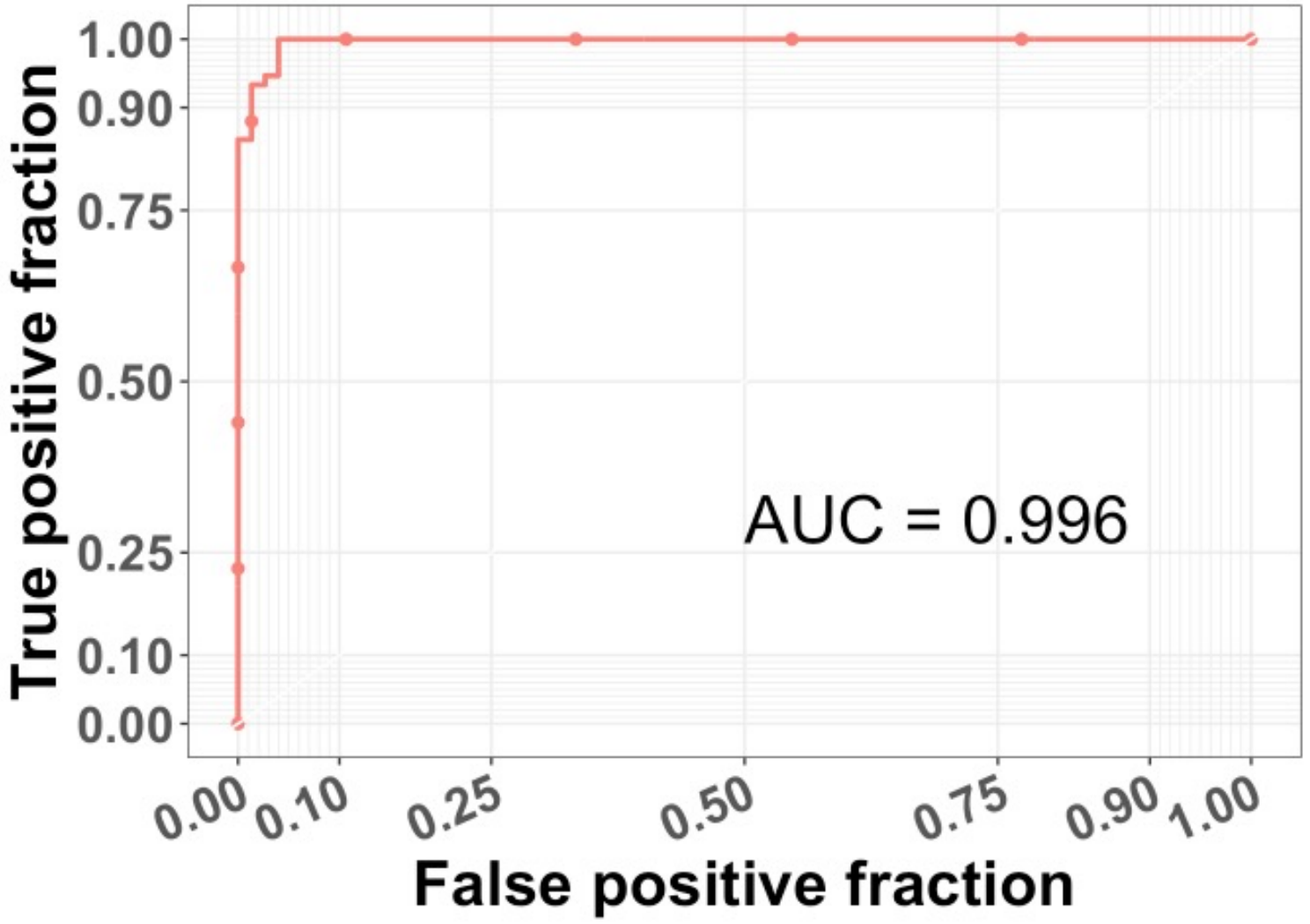
```r
# True labels for test samples
true.class <- mRNAexp_data$Sample_Label[-trainIndex_2class]

### Predict labels using trained models
pred.class.glmnet <- predict(glmnet.fit, newdata = mRNAexp_data[-trainIndex_2class, ])
confusionMatrix(pred.class.glmnet, true.class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction control disease
##    control      72       2
##    disease       3      73
##
##                Accuracy : 0.9667
##                  95% CI : (0.9239, 0.9891)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9333
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9600
##             Specificity : 0.9733
##          Pos Pred Value : 0.9730
##          Neg Pred Value : 0.9605
##              Prevalence : 0.5000
##          Detection Rate : 0.4800
##    Detection Prevalence : 0.4933
##       Balanced Accuracy : 0.9667
##
##        'Positive' Class : control
##
```

Prediction Results



ROC Plot for Prediction Results



References

- Towards Data Science Blogs: https://medium.com/@NotAyushXD
- Kaggle: https://www.kaggle.com/
- Introduction to R library “caret”
  - https://topepo.github.io/caret/index.html
- The Elements of Statistical Learning
  - https://web.stanford.edu/~hastie/ElemStatLearn/