

Machine Learning in practice

Marta Sestelo & Nora M. Villanueva

GDG DevFest Galicia 2018

Santiago de Compostela, Spain. 17th November 2018

Who are we?



Nora M. Villanueva
Senior Data
Scientist
[@noramvillanueva](#)



Marta Sestelo, PhD
Senior Data
Scientist
[@MartaSestelo](#)



Summary

Machine Learning framework will be introduced by showing both supervised and unsupervised learning methods. We will describe some of the most well-known and widely used techniques: (1) regression models, (2) classification models, (3) generalized additive models, (4) tree-based models and (5) clustering. We will focus on understanding some mathematical details of these models and how to carry out a practical use of them. We will illustrate these techniques analyzing two different case studies, using to this end a free interactive computing environment, a Jupyter notebook including the R kernel.



Table of Contents

1. Installations Instructions
2. Intro to Machine Learning
3. Regression (LM)
4. Classification (GLM)
5. Generalized Additive Models (GAM)
6. Tree-Based Methods
7. Model evaluation
8. Clustering



Installation Instructions

Docker installation

We are going to install Docker Community Edition (CE). It is ideal for developers and small teams looking to get started with Docker and experimenting with container-based apps. You can find instructions in the following links:

- Ubuntu: <https://docs.docker.com/install/linux/docker-ce/ubuntu/>
- Mac OS: <https://store.docker.com/editions/community/docker-ce-desktop-mac>
- Windows: <https://store.docker.com/editions/community/docker-ce-desktop-windows>

Once it has been installed, the following step will be download and run the *Gradient Jupyter Notebook* image that contains the jupyter app with the IRKernel and the packages needed already installed.



Gradiant Jupyter R Image

- This image includes libraries for data analysis from the R community.

- Run the Gradiant Jupyter Notebook image in your Docker container:

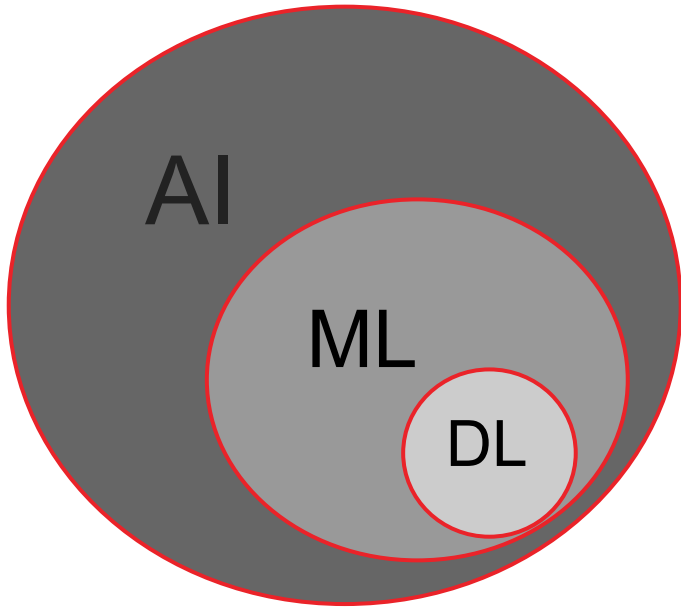
```
mkdir ml_notebook  
docker run -p 8888:8888 -v "$(pwd)/ml_notebook:/notebooks" -ti --rm gradiant/jupyter:5.7.0-gdg2018
```

- Browse to <http://localhost:8888>
- Download the *workshop_machine_learning.ipynb* file from https://github.com/sestelo/machine_learning_workshop_gdg2018



Intro to Machine Learning

Intro to Machine Learning - Categorization



AI attempts to understand and build intelligent entities

ML explores the study and construction of systems that can learn from and make predictions on data

DL uses **ML** techniques (mostly classical ones) with added levels of complexity relying on the computing capabilities we have nowadays



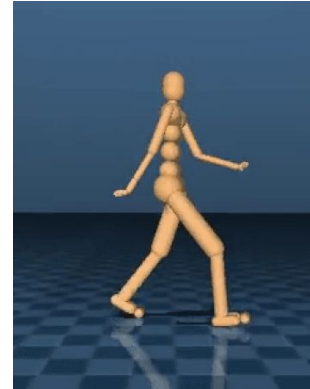
Intro to Machine Learning - Categorization

Based on the type of learning:

- Supervised Learning
- Unsupervised Learning
- Semi-supervised learning
- Reinforcement learning

Based on the desired type of response:

- **Regression:** the response is continuous. Type: supervised.
- **Classification:** the response is divided into two or more classes. Type: supervised.
- **Clustering:** there is not response. Type: unsupervised.



Some Notation

Notation

Before starting it is necessary establish some notation...

\mathbf{Y} : response or dependent variable (output, label, target, etc.)

$\mathbf{X} = X_1, \dots, X_p$: predictors or independent variables (input, features, etc.)

p : number of predictors or features

n : number of observations

$(X_i, Y_i)_{i=1}^n$: sample



Regression

Linear Regression

- **Supervised** learning approach that we used when the response Y is **quantitative**.
 - Ex.: Weather forecasting, population growth prediction, price estimation, etc.
- This technique allow us to know the **relationship between the response and the covariates**.
 - **Inference**
 - **Prediction**
- We are going to assume that there is some relationship between Y and $\mathbf{X} = X_1, \dots, X_p$, which can be written in the very general form

$$Y = m(\mathbf{X}) + \varepsilon$$

- Here, m is some fixed but unknown function of X_1, \dots, X_p , and ε is a random error term, which is independent of X and has mean zero.



Linear Regression - Estimation

In the more simple case (**Simple Linear Regression**) m will be

$$\text{intercept} \quad \text{---} \quad \text{slope}$$
$$m(X) = \beta_0 + \beta_1 X$$

β_0 and β_1 are two unknown constants that represent the intercept and slope terms in the linear model. Together, β_0 and β_1 are known as the model coefficients or parameters.

- How do we **estimate** m ? Using least square method.

Let $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$ be the prediction for Y based on the i th value of X. Then $e_i = y_i - \hat{y}_i$ represents the i th residual—this is the difference between the i th observed response value and the i th response value that is predicted by our linear model.

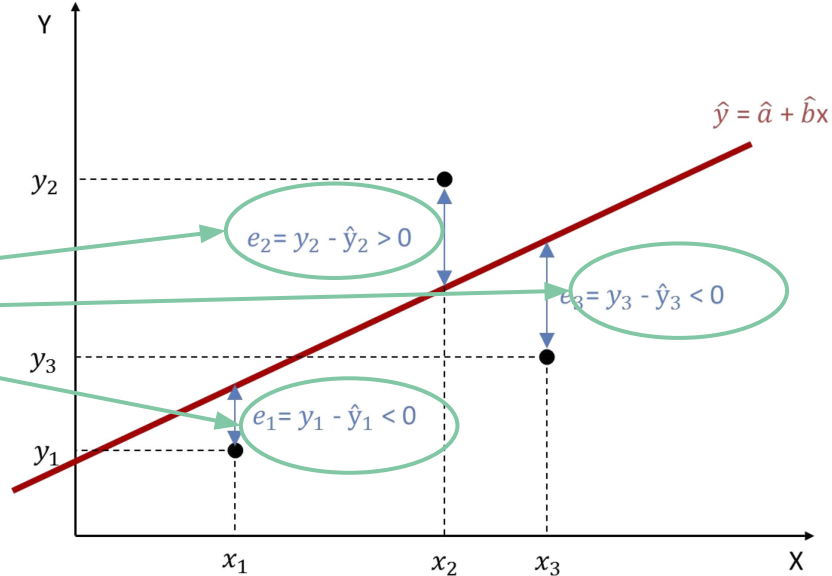


Linear Regression - Estimation

We will try to look for the values $\widehat{\beta}_0$ and $\widehat{\beta}_1$ that minimizes the following sum of squares (RSS):

$$\begin{aligned} \text{RSS} &= \sum_{i=1}^n \hat{e}_i^2 = \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 = \\ &= \sum_{i=1}^n (Y_i - (\hat{\beta}_0 + \hat{\beta}_1(X_i)))^2 \end{aligned}$$

Residuals



Linear Regression - Extensions

- Multiple Linear Regression

$$m(\mathbf{X}) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p = \beta_0 + \sum_{j=1}^p \beta_j (X_j)$$

- Basic extensions:

Non Linear Regression (Polynomial representation)

- Quadratic

$$m(\mathbf{X}) = \beta_0 + \beta_1 X + \beta_2 X^2$$

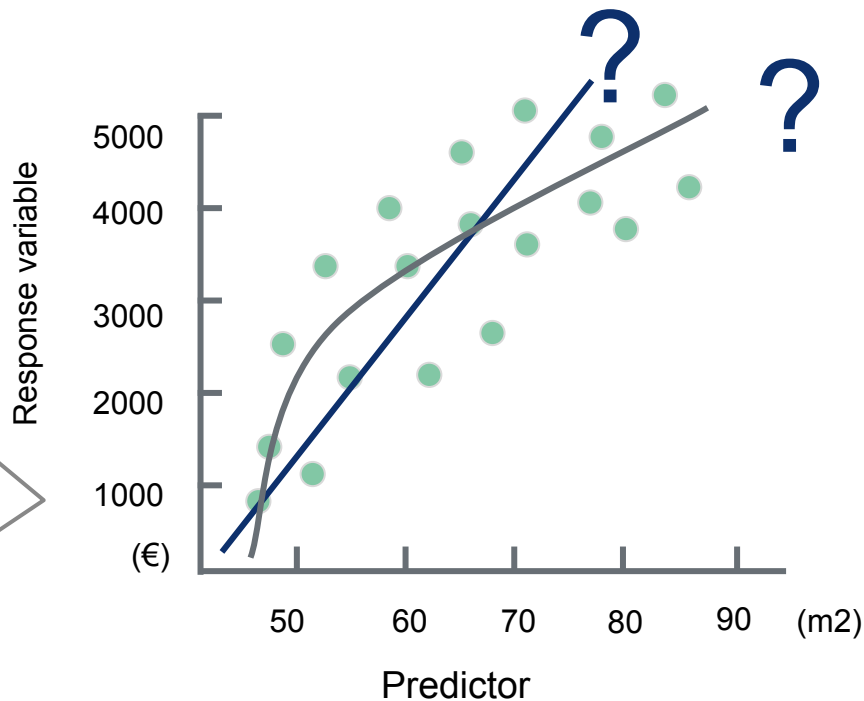
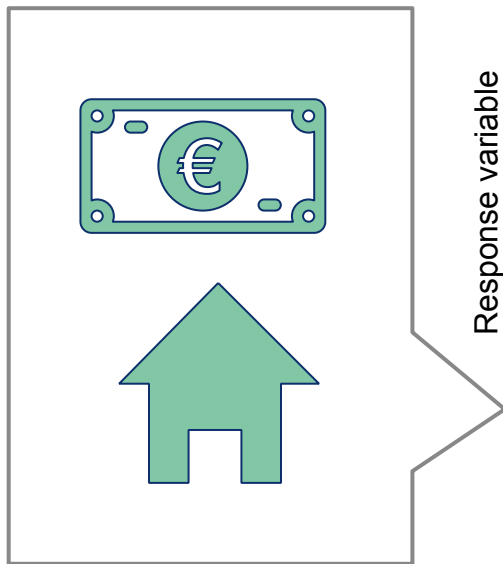
- Cubic

$$m(\mathbf{X}) = \beta_0 + \beta_1 X + \beta_2 X^2 + \beta_3 X^3$$



Linear Regression - Use Case

Predicting apartments prize



Classification

Classification

- In many situations, the response variable is **qualitative** or **categorical**.
- **Classification:** process of predicting qualitative responses.
 - Ex.: Fraud detection, image classification, diagnostic of a disease, customer retention, churn, etc.
- Often the methods used for classification first **predict** the **probability of each of the categories** of a qualitative variable, as the basis for making the classification. In this sense they also behave like regression methods.
- There are many possible classification techniques, or classifiers:
 - Logistic regression
 - Linear discriminant analysis (quadratic discriminant analysis)
 - Generalized Additive models (GAMs)
 - Decision Trees, Random forest



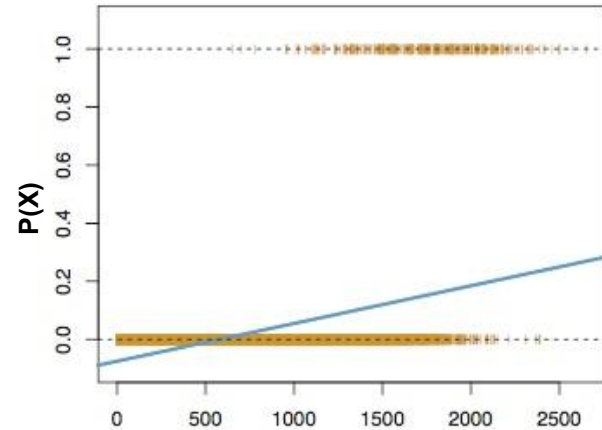
Generalized Linear Regression - Logistic Regression

- Response variable falls into one of **two categories** (Yes or No, 1 or 0, etc)
- Logistic regression **models the probability** that Y belongs to a particular category.

$$p(X) = P(Y = 1|X)$$

- How should we model the relationship between $p(X) = \Pr(Y = 1|X)$ and X ?

$$\begin{array}{ll} p(X) = P(Y = 1|X) & \beta_0 + \beta_1 X \\ [0, 1] & (-\infty, +\infty) \end{array}$$



Generalized Linear Regression - Logistic Regression

How should we model the relationship between $p(X) = \Pr(Y = 1|X)$ and X ?

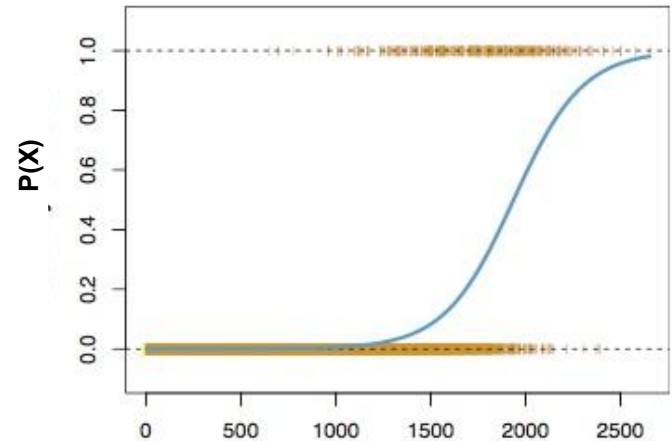
$$\frac{p(X)}{1-p(X)} \quad \beta_0 + \beta_1 X$$
$$[0, +\infty) \quad (-\infty, +\infty)$$

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X$$
$$(-\infty, +\infty) \quad (-\infty, +\infty)$$

Logit Function

$$P(Y = 1|X) = g^{-1}(\beta_0 + \beta_1 X) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}}$$

where g is the link function.



Note: To avoid this problem, we must model $p(X)$ using a function that gives outputs between 0 and 1 for all values of X .



Logistic Regression - Estimation

- How do we estimate the model?
 - Using **maximum likelihood**
 - Iterative algorithm such as Newton-Raphson
- We try to find $\widehat{\beta}_0$ and $\widehat{\beta}_1$ such that plugging these estimates into the model for $p(X)$:
 - yields a number close to one for all individuals who $Y = 1$
 - a number close to zero for all individuals who did not ($Y = 0$)
- Formalized using a likelihood function:

$$l(\beta_0, \beta_1) = \prod_{i:y=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'})).$$



Logistic Regression - Estimation

- Multiple Logistic Regression

$$p(X) = \frac{e^{\beta_0 + \beta_1 X + \dots + \beta_p X_p}}{1 + e^{\beta_0 + \beta_1 X + \dots + \beta_p X_p}}$$

- Basic extensions:

for > 2 Response Classes (Multinomial regression)



$$Y = \begin{cases} 1 & \text{if blue} \\ 2 & \text{if pink} \\ 3 & \text{if black} \end{cases}$$

$$p(X) = P(Y = k|X) = \frac{e^{\beta_{k0} + \beta_k X}}{1 + \sum_{l=1}^{K-1} e^{\beta_{l0} + \beta_l X}}$$



Logistic Regression - Use Case


Titanic data set

- ✓ Is there any effect of the ticket price in the probability of death during the titanic journey?
- ✓ Are the males less likely to survive than females?
- ✓ Can you know the probability of death for any given age?
- ✓ Can you predict if an individual will survive?



Generalized Additive Models

Generalized Additive Models

- In some circumstances, **Linear Models** (LM or GLM) can be very **restrictive**, since they assume **linearity in the effect of the covariates** and, in some circumstances, this is not supported by the data at hand.  If the parametric model fails, then the **conclusion will be erroneous**.
- **Nonparametric regression** techniques are involved in modeling the dependence between Y and X, but without specifying, in advance, the function m that links them.
- **Additive model** (AM), which is a generalization of the LM, by introducing one-dimensional, nonparametric functions instead of linear components. Specially, AMs express the conditional mean as

$$m(\mathbf{X}) = \alpha + f_1(X_1) + f_2(X_2) + \dots + f_p(X_p)$$

α : constant

f_j : is the unknown smooth partial function associated to each continuous covariate X_j .



Generalized Additive Models

- This models can also be used in situations where Y is **qualitative**.
- In such cases, the **Generalized Additive Models** (GAMs) extends the AM by allowing for different distributions of the response variable Y (binomial, poisson, etc.). Specially, GAMs express the conditional mean as

$$m(\mathbf{X}) = g(\alpha + f_1(X_1) + f_2(X_2) + \dots + f_p(X_p))$$

where g is a monotonic known function (the inverse of link function).



Generalized Additive Models - Estimation

In order to **estimate** the previous models, several approaches have been suggested in the literature, for example:

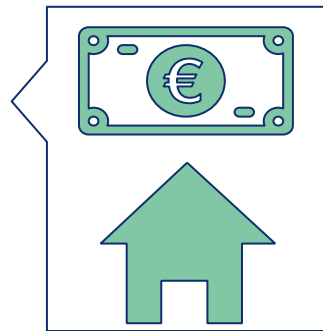
- local polynomial kernel smoothers - backfitting algorithm (KernSmooth, np packages)
- methods based on penalised regression splines (**mgcv** package)
- Bayesian versions of these



Generalized Additive Models - Use Case

Predicting apartments prize

- ✓ Can the relationship between the prize and each predictor be adequately summarized using a linear equation, or is the relationship more complicated?
- ✓ Given some specific values for the features, how much will be the prize of the apartment?



Tree-Based Methods

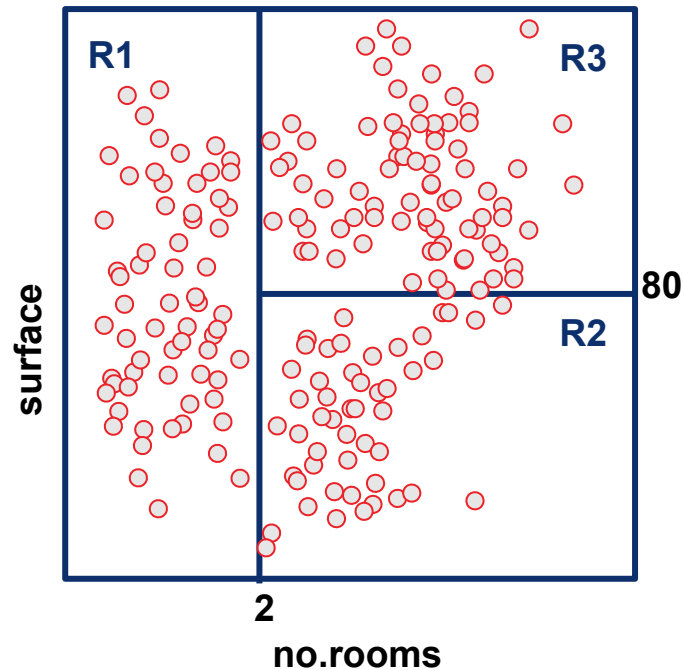
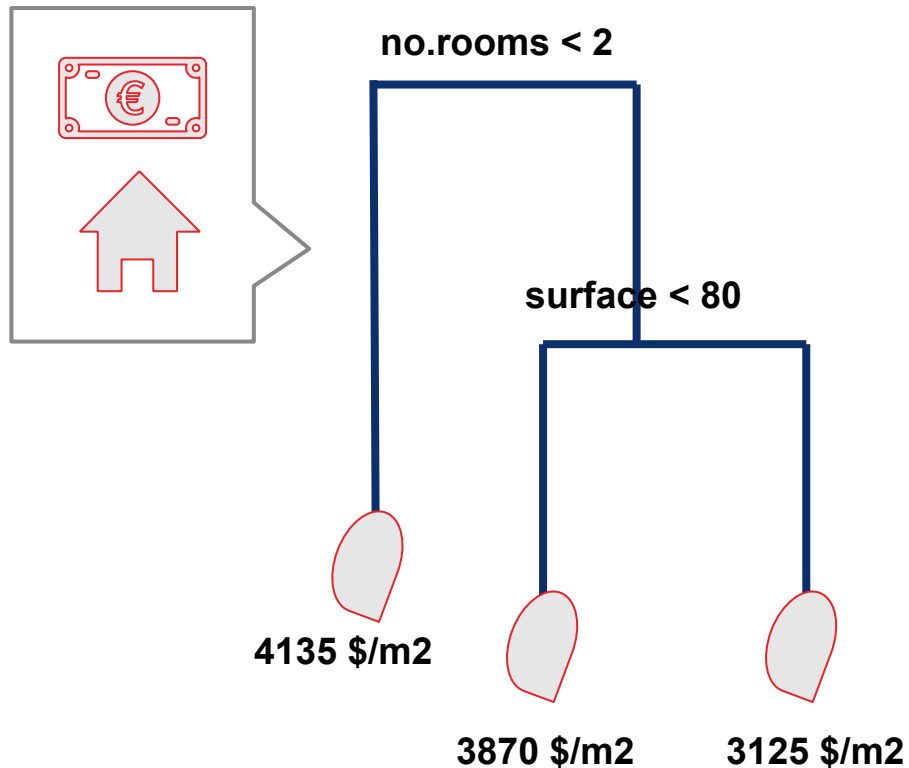
Tree - Based Methods

- They are valid for **regression** and **classification**.
- How do they work?
 - **Stratifying** or segmenting the predictor space into a number of simple regions.
 - In order to make a **prediction** for a given observation: we typically use the mean or the mode of the training observations in the region to which it belongs.
- Tree-based methods like decision tree are **simple** and **useful** for **interpretation**, but not competitive compared to the best supervised learning approaches, such as GAMs, in terms of prediction accuracy.

- ✓ But..., we also introduce bagging, random forests, and boosting.
- ✓ To combine a large number of trees can often result in dramatic improvements in prediction accuracy, at the expense of some loss in interpretation.



Decision trees

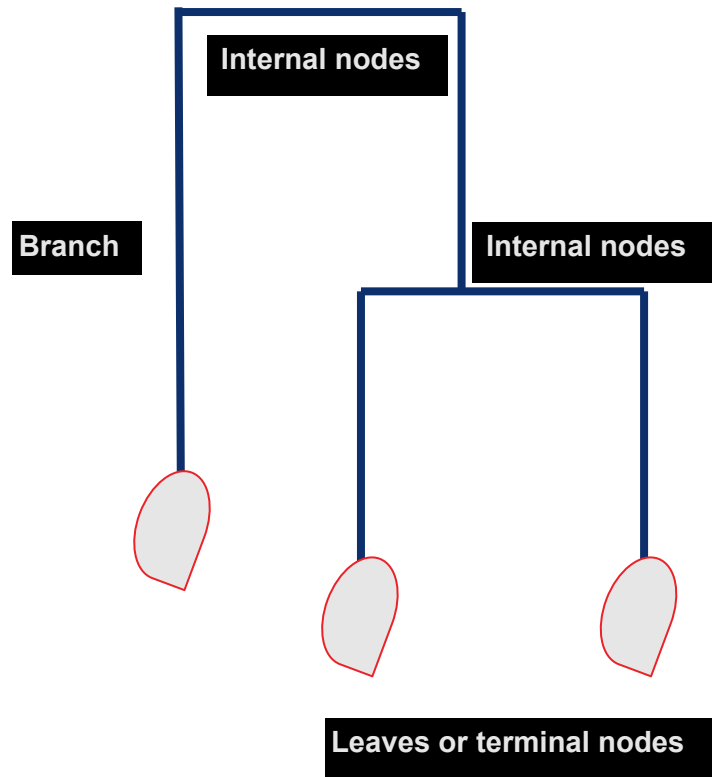


Decision trees

- Tree analogy:
 - ✓ **Terminal nodes** or **leaves** of the tree: the regions R1, R2, and R3
 - ✓ **internal nodes**: the points along the tree where the predictor space is split
 - ✓ **Branches**: segments of the trees that connect the nodes

Note: the decision tree is an **over-simplification** of the true relationship between price, no.rooms and surface.

Advantages over other types of regression models: 1) easy to interpret 2) nice graphical representation.



Decision trees - Prediction

- Two important steps:
 1. Divide the predictor space: the set of possible values for X_1, X_2, \dots, X_p into J distinct and non-overlapping regions, R_1, R_2, \dots, R_J .
 2. Make the prediction: for every observation that falls into the region R_j , it will be the same, that is simply the mean of the response values for the training observations in R_j .
- How do we build these regions?

Goal: to find boxes R_1, \dots, R_J that minimize the RSS

$$\sum_{j=1}^J \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$


where \hat{y}_{R_j} is the mean response for the training observations within the j th box.

Unfortunately to consider every possible partition of the feature space into J boxes is computationally infeasible

Solution: top-down, greedy approach
Recursive Binary Splitting



Decision trees - Recursive Binary Split

1. Consider all predictors X_1, X_2, \dots, X_p and all possible values of the cutpoints for each of the predictors
 choose the predictor and cutpoint such that the resulting tree has the lowest RSS.
2. Repeat the process, with other predictors (keeping fixed the previous predictors)
3. The process continues until a stopping criterion is reached (continue until no region contains more than five observations)



Classification trees

- The prediction for each region will be the **value of the majority class**.
So, the **class proportions** among the training observations will be important.

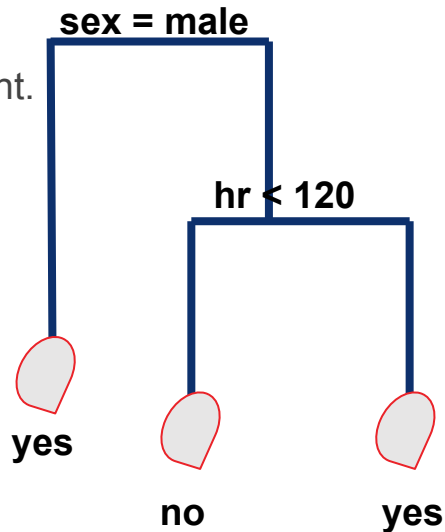
- Alternative to RSS is the classification error rate (not sufficiently sensitive)

$$E = 1 - \max_k(\hat{p}_{mk})$$

- Better option: Gini Index (a measure of total variance across the K classes)

$$G = \sum_{k=1}^K \hat{p}_{mk}(1 - \hat{p}_{mk})$$

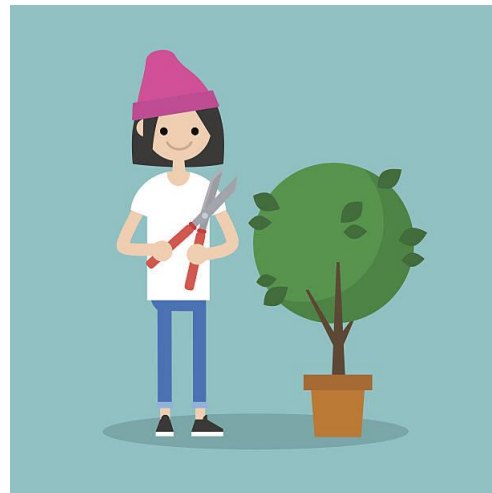
\hat{p}_{mk} represents the proportion of training observations in the mth region that are from the kth class.



Tree - Based Methods - Tree pruning

- The previous process may produce good predictions on the training set, but is likely to overfit the data, leading to poor test set performance \longrightarrow resulting tree might be too complex.
- A smaller tree with fewer splits (that is, fewer regions R_1, R_2, \dots, R_J) might lead to lower variance and better interpretation at the cost of a little bias.
- **How can I prune the tree?**
 - ✓ Prune a large tree from the leaves to the root and form subtrees T_1, T_2, \dots, T_m .
 - ✓ Select the optimal tree T_i by cross validation.

$$\frac{RSS(T_1) - RSS(T_0)}{|T_0| - |T_1|}$$



Bagging

Training data in halves: maybe two quite different results

- Decision trees have **high variance**.
- **Solution**: averaging a set of observations reduces variance.
- Bootstrap aggregation, or **bagging**, is a general-purpose procedure for reducing the variance of a statistical learning method.
- ✓ We generate B different bootstrapped training data sets
- ✓ We then train our method on the b-th bootstrapped training set in order to get $\hat{f}^{*b}(x)$
- ✓ Finally average all the predictions, to obtain

$$\hat{f}_{bag}(x) = \frac{1}{B} \sum_{b=1}^B \hat{f}^{*b}(x) \quad \text{with}$$

$$\begin{aligned} \hat{f}(x) &= \sum_{m=1}^M \hat{c}_m \cdot \mathbf{1}_{x \in R_m} \\ \hat{c}_m &= \text{ave}(y_i | x_i \in R_m) \end{aligned}$$



Record the total amount that the RSS or Gini Index is decreased due to splits over a given predictor, averaged over all B trees.

Bagging

- NOTE: These trees are grown deep, and are not pruned. Hence each individual tree has high variance, but low bias. Averaging these B trees reduces the variance.
- **Impressive improvements in accuracy** by combining together hundreds or thousands of trees.
- How can **bagging** be extended to a **classification problem** where Y is qualitative?
 - ✓ Record the class predicted by each of the B trees, and take a majority vote.
- Number B of trees? High enough.
- × **Disadvantage**: It improves prediction accuracy at the expense of interpretability.
- Overall summary of the importance of each predictor using the RSS or the Gini index

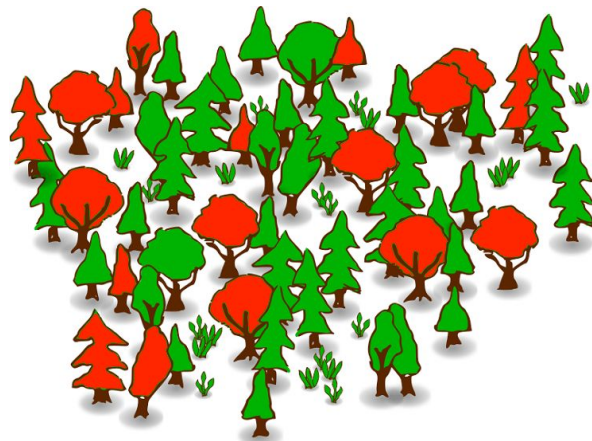


Random Forest

- **Bagging** suffers **tree correlation** → reducing the overall performance of the model.
Trees are not completely independent, all predictors are considered at every split of every tree
- **RF reduces even more the variance** → minimizing correlation between the trees
Trees are completely independent, a random subset of m predictors are considered at every split of every tree

$$m \approx \sqrt{p}$$

- **RF remarks**
 - Modification of bagging
 - Very popular learning algorithm
 - Builds a large collection of de-correlated trees
 - Good predictive performance



Boosting

- It is an **ensemble meta-algorithm** for **reducing bias and variance**.
- It converts **weak learners** to **strong ones**.
- Boosting works in a similar way than bagging, except that the trees are grown sequentially:
 - Each tree is grown using information from previously grown trees.
 - Boosting does not involve bootstrap sampling; instead each tree is fit on a modified version of the original data set (with the **residuals**).
- Type of algorithms:
 - AdaBoost (adabag package)
 - Gradient Boosting (gbm package)
 - Extreme Gradient Boosting - XGboost (xgboost package)



Boosting

Algorithm 8.2 *Boosting for Regression Trees*

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.
2. For $b = 1, 2, \dots, B$, repeat:
 - (a) Fit a tree \hat{f}^b with d splits ($d + 1$ terminal nodes) to the training data (X, r) .
 - (b) Update \hat{f} by adding in a shrunk version of the new tree:

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x). \quad (8.10)$$

- (c) Update the residuals,

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i). \quad (8.11)$$

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^B \lambda \hat{f}^b(x). \quad (8.12)$$



Boosting

Tuning parameters:

1. **The number of trees B** . Unlike bagging and random forests, boosting can overfit if B is too large, although this overfitting tends to occur slowly if at all. Use **cross-validation** to select B .
2. **The shrinkage parameter λ** , a small positive number. This controls the rate at which boosting learns. Typical values are 0.01 or 0.001, and the right choice can depend on the problem.
3. **The number d of splits in each tree**, which controls the complexity of the boosted ensemble. Often $d = 1$ works well, in which case each tree is a stump, consisting of a single split. In this case, the boosted ensemble is fitting an additive model, since each term involves only a single variable. More generally d is the interaction depth, and controls the interaction order of the boosted model, since d splits can involve at most d variables.



Model Evaluation

Model Evaluation - Regression

- Why are there so many different statistical learning approaches, instead of a single best method?

No one method dominates all others over all possible data sets

- What is model evaluation? It is a measure of the **quality of the model**.
- How can we carry it out? Measuring how well the predictions actually match the observed data, i.e., estimating the prediction error on new data.
- For calculating this measure you **MUST** split data into **training** and **test**. And this measure must be obtained into the testing data.

Low MSE training and High MSE test



Overfitting



Model Evaluation - Regression

Metrics:

- Mean Squared Error (MSE)

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

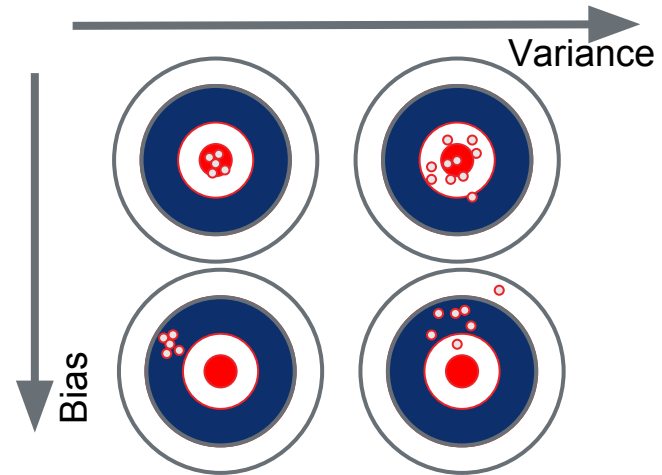
- The MSE is the same that

$$E(y_0 - \hat{f}(x_0))^2 = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2$$

As a general rule, as we use more flexible methods the variance will be increase and the bias will be decrease.

- Mean Absolute Error (MAE)

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{f}(x_i)|$$



Model Evaluation - Classification

Contingency table or Confusion matrix

➤ Example for 2 classes: 2x2 dimension

➤ Metrics

✓ Sensitivity/Recall/True Positive Rate

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

✓ Specificity/Selectivity/TNR

$$TNR = \frac{TN}{TN+FP}$$

✓ Precision

$$PPV = \frac{TP}{TP+FP}$$

✓ Accuracy

$$ACC = \frac{TP+TN}{TP+TN+FP+FN}$$

✓ F1score

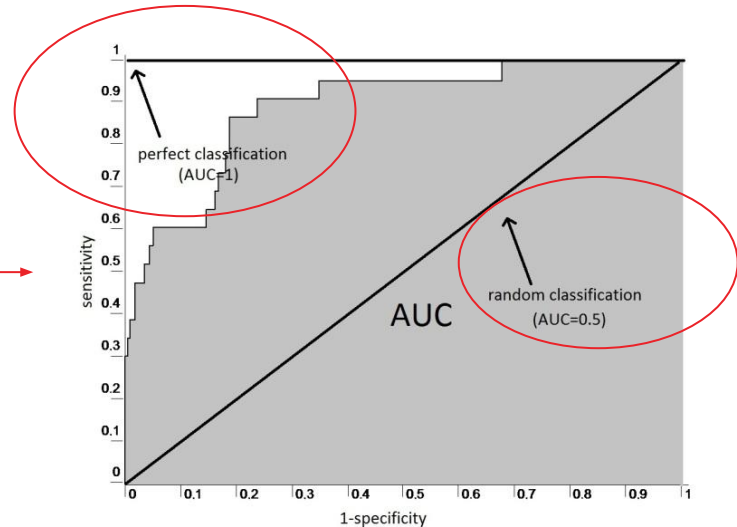
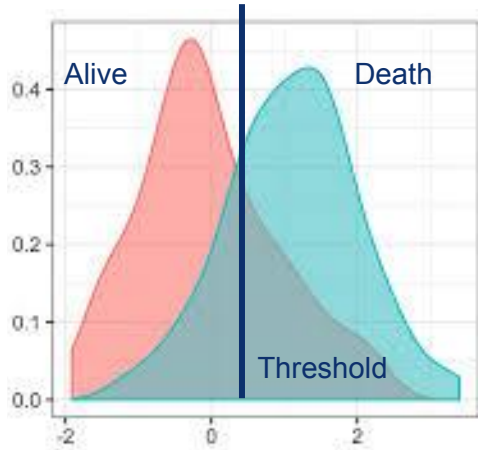
$$F1 = 2 \cdot \frac{PPV \cdot TPR}{PPV+TPR}$$

		Real Class	
		Alive	Death
Predicted Class	Alive	5 TP	2 FP
	Death	3 FN	17 TN



Model Evaluation - Classification

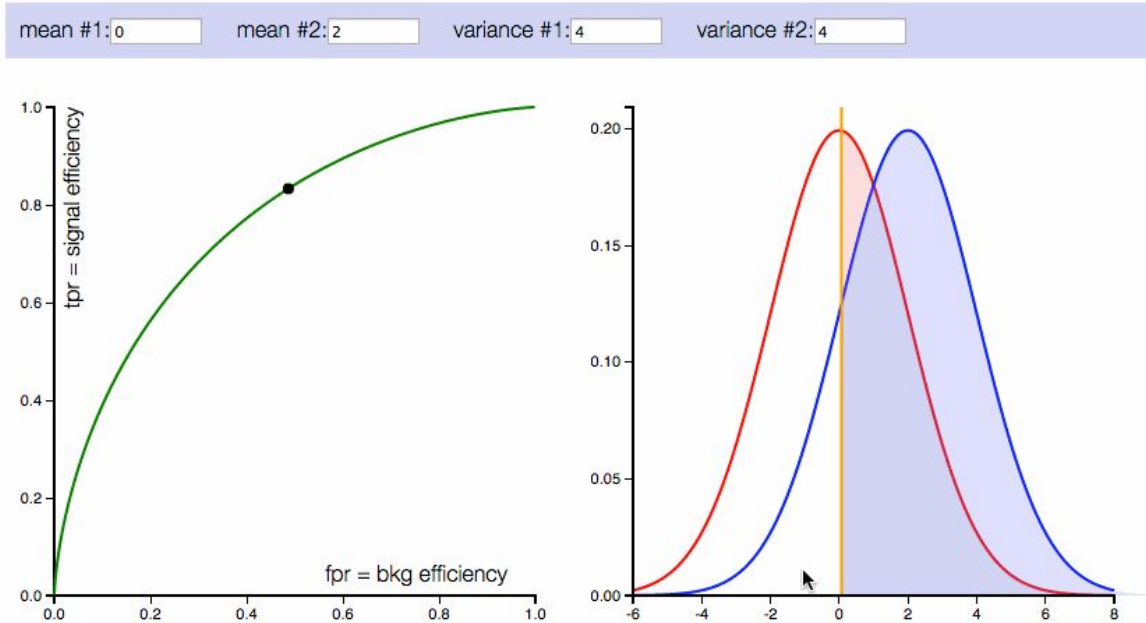
- Receiver Operating Characteristic curve (ROC curve)
 - A plot for sensitivity (TPR) against 1-specificity (FPR) at various threshold settings.
- Area under the curve (AUC) $[0, 1]$



Model Evaluation - Classification

- Receiver Operating Characteristic curve (ROC curve)
 - A plot for sensitivity (TPR) against 1-specificity (FPR) at various threshold settings.

ROC curve demo



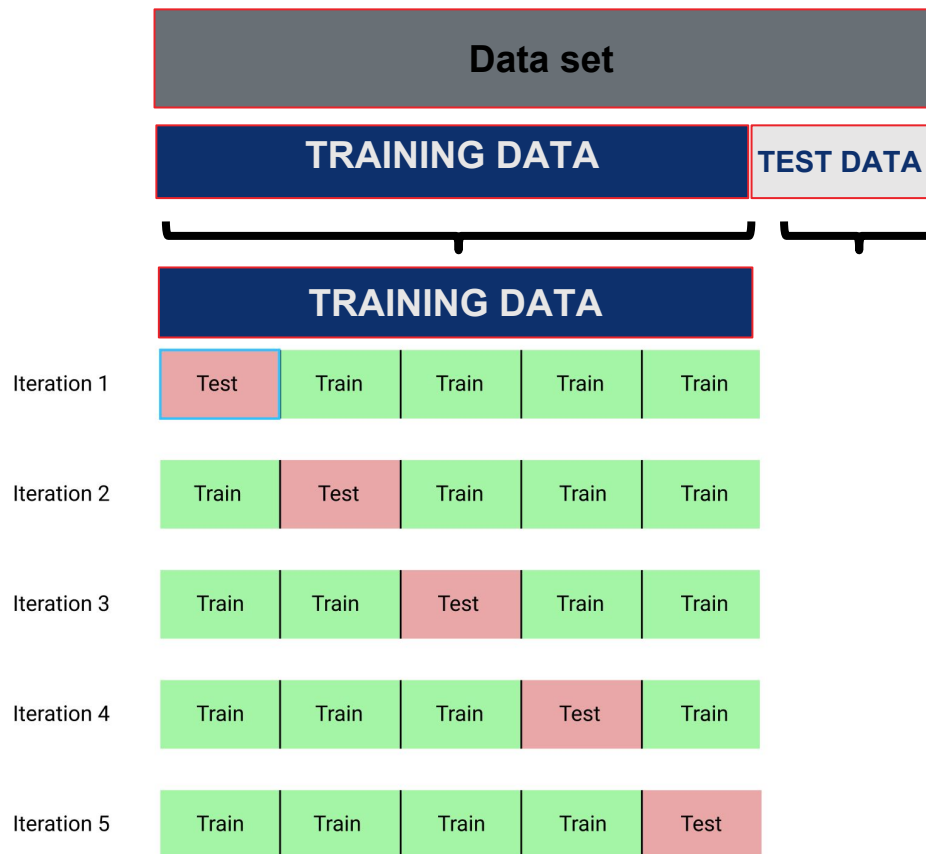
Model Evaluation

- Split Data Train and Test

✓ 80% vs 20%

- **k-Fold Cross-validation** Technique:

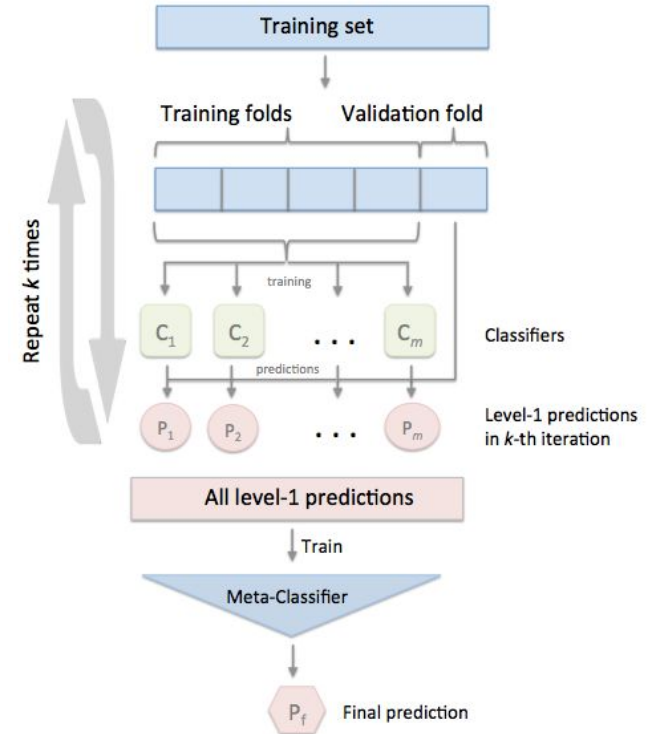
➤ A method for estimating test MSE using the training data



Stacked Ensembles

Stacking - Super learning - Stacked Regression

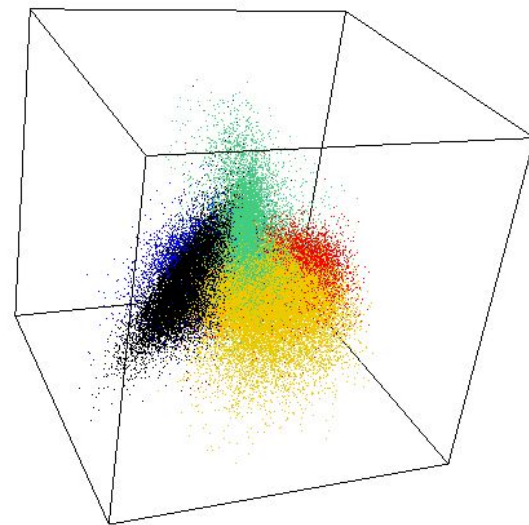
- Ensemble machine learning methods (RF, boosting) use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms.
- Stacking Involves training a second-level “metalearner” to find the optimal combination of the base learners.
- Unlike bagging and boosting, the goal in stacking is to ensemble strong, diverse sets of learners together.



Clustering

Clustering

- This is an **unsupervised** technique (no response or label).
 - Ex: Customer segmentation, Grouping Students, etc.
- It seek to simplify the data trying to find **homogeneous subgroups** amon the observations.
- The resulted groups are called **clusters**.
- There are several algorithm for clustering:
 - k-means (Centroid-based clustering)
 - Gaussian Mixture Models (Distribution-based clustering)
 - DBSCAN (Density-based clustering)



Clustering - K-means

- It is one of the most **famous** algorithm for clustering.
- It is an **heuristic method** that resolve an optimization problem:
 - aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.
- With notation... given a set of observations (X_1, X_2, \dots, X_n) , where each observation is a d-dimensional real vector, k-means clustering aims to partition the n observations into k ($\leq n$) sets $S = \{S_1, S_2, \dots, S_n\}$ so as to minimize the within-cluster sum of squares (WCSS) (i.e. variance

$$\arg \min_{\mathbf{S}} \sum_{i=1}^k \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2 = \arg \min_{\mathbf{S}} \sum_{i=1}^k |S_i| \text{Var } S_i$$

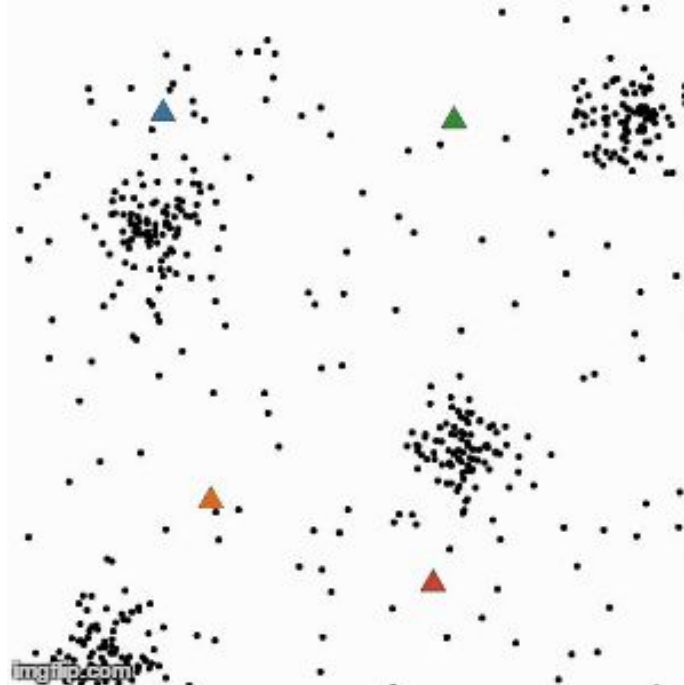
where $\boldsymbol{\mu}_i$ is the mean of points in S_i .

- There exist other similar algorithms: K-medians, K-medoids, etc.



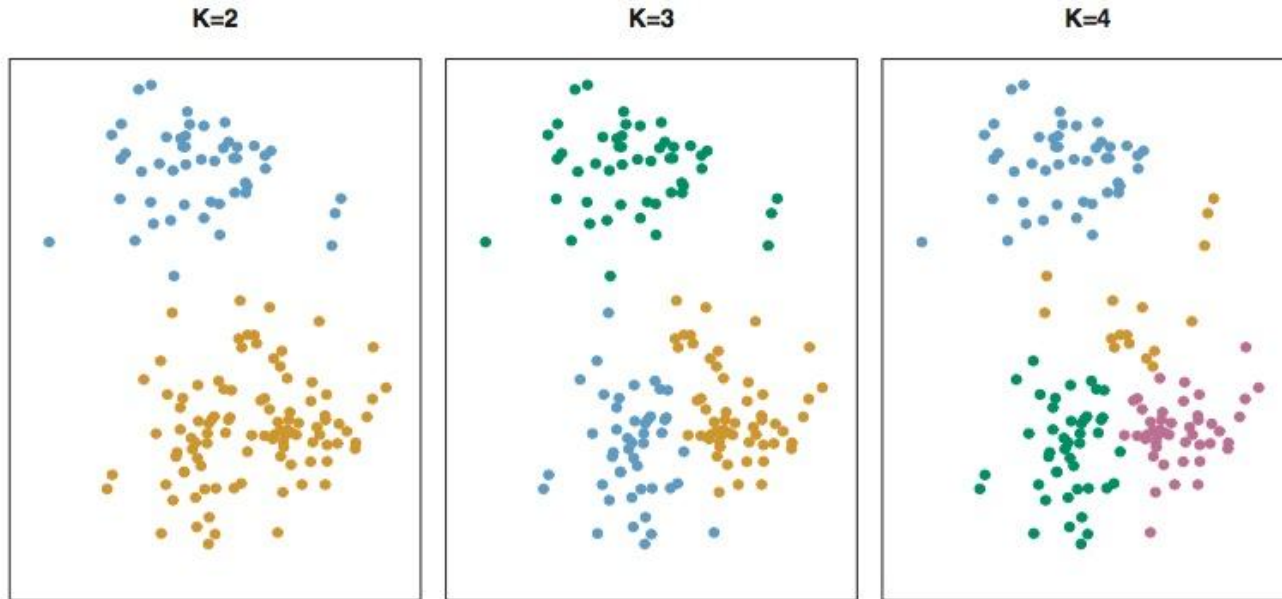
Clustering - kmeans

- What are the steps to get the results?



Clustering - kmeans

- How many clusters should we look for in the data?





Oriented to Industry requirements

Marta Sestelo msestelo@gradient.org

Nora M. Villanueva nmvillanueva@gradient.org



Google Developers