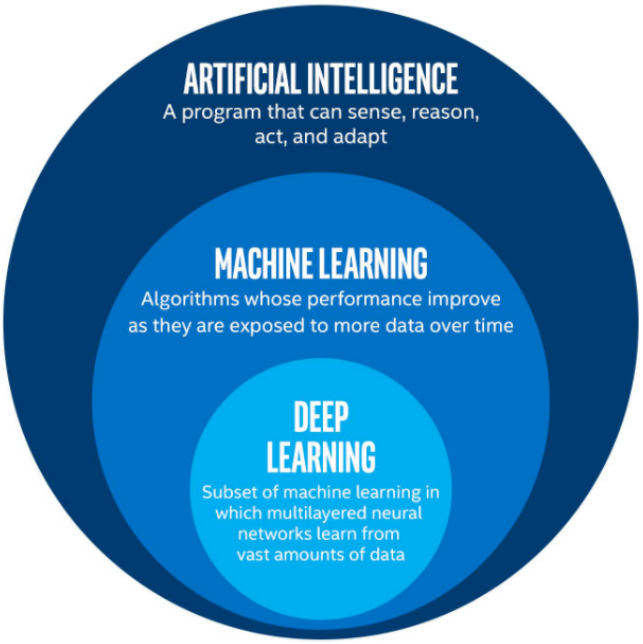
2. Machine Learning

“All models are wrong, but some are useful” George Box 1978

Welcome, this document is full of links to valuable resources, feel free to explore them in-depth click on everything and its subsequent links **(especially if you have some form of obsessive-compulsive disorder or Wikipedia syndrome)**

Intro:

Machine Learning and Artificial Intelligence:



2.1 Datasets

Different types of problems require different data and techniques to solve them.

See Classification and regression data sets (features and labels):

data set for classification iris data description

data set for regression wine data description

These are repositories of datasets:

UC Irvin https://archive.ics.uci.edu

Data set from https://index.okfn.org/place/

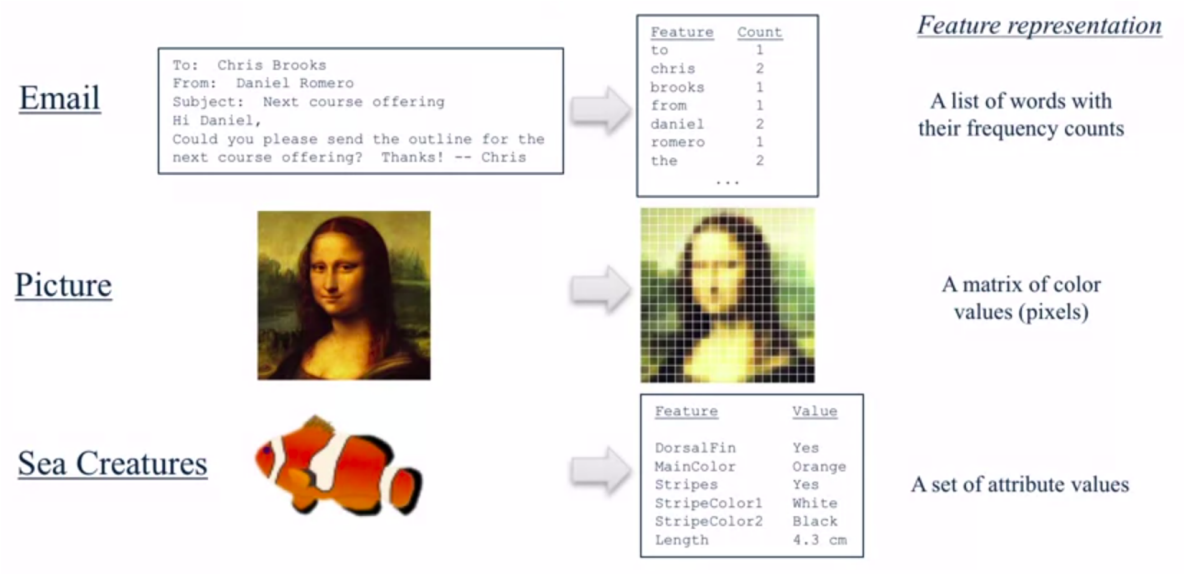
UN http://data.un.org/

World bank https://data.worldbank.org/

Load the dataset into pandas framework code example

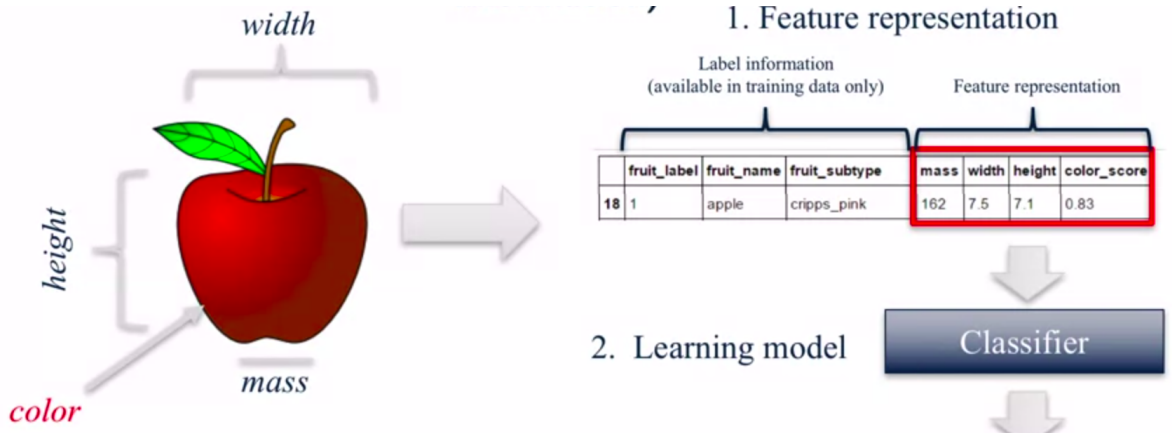
Activity: Go into one of the above links, download a dataset, open it in a plain text editor (notepad, nano, pico, etc...) and locate the instances, the features/attributes, the values of the attributes, the labels/classes. Now load it into pandas in your own notebook

2.1.1 Representing data:



source by Kevyn Collins-Thompson, University of Michigan

Representing data in data sets:



source by Kevyn Collins-Thompson, University of Michigan

How are they used?

We use **regression** to predict a numeric value given a set of inputs.

(Finance market prediction).

https://drive.google.com/open?id=1-q4M6vzZ4J85DN9uPDcID542oP7W9Wqp

https://drive.google.com/open?id=1c_EKNlB4fqGy-f0mo789zjqKUNmv1Ood

We use **classification** to predict a class or label given a set of inputs.

(Facebook face recognition tagging).

In this course, we will address the case of supervised machine learning.

(Facebook Talk Video) Talks about the key intuitions behind machine learning with NLP problems (1 to 40min). The relevance of data and labeling it (44 to min 49). *Holy grail explained in video full of insight which you are not going to watch, but I will post it anyway cause reasons:* *https://www.facebook.com/Engineering/videos/644326502463/*

Short version Reading (data):
https://machinelearningmastery.com/hands-on-big-data-by-peter-norvig/

makes notes on the following concepts:

Big Data	Algorithms	Trade between more data and better algorithms

2.2 Linear Regression:

Make a graph using a small data set and watch the relationship between the dots and the function:

CPUs (x)	Price (y)
2	100
3	200
4	300
5	400
6	500
7	600
8	????

2.2.1 Establishing hypothesis functions:

Write a function that you think would pass through most of the dots in the graph. (5 min.)

Linaer_reg_examples.docx

2.2.2 Establishing cost/loss functions:

To see how good your hypothesis is, compare your hypothesis function with the real values and calculate the error i.e. (the difference between the actual data and the predictions of your function).

Linaer_reg_examples.docx

Compensate for the negative sign (get the distance not the direction)

Integrate the error of every data point

Get the average error.

Mean Square Error

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

If we have a cost function and a model, we can treat this as an optimization problem.

Fitting a function with an Optimization method:

Math way: (Optimal) Matrix operations $\mathbf{b=(X'X)^{-1}\ X'y}$ but can be very expensive depending on the data set and has several restrictions (no sparsity, shape of the matrix, among others ...).

The Iterative way. (M.L.) (approximation through trial and error approach).

Write down the initial hypothesis with random parameters for a linear regression model for the following dataset, where you wish to predict **USD Based ISE**:

https://archive.ics.uci.edu/ml/datasets/ISTANBUL+STOCK+EXCHANGE#

Reading (Types of Gradient Descent):

Definitions: **https://builtin.com/data-science/gradient-descent**

Algorithms: **https://www.geeksforgeeks.org/gradient-descent-algorithm-and-its-variants/**

just the first 3 sections (Batch/Stochastic/Minibatch) This reading lets you tell the difference between each version of gradient descent, the changes concern the way data is fed and when parameters are updated.

Make notes on:

Gradient Descent **Batch** **Stochastic** **Minibatch**

2.2.3 Gradient descent

Optimizing to local minima**. (Example of graphic, behavior)** use gradient_examples.docx

Use of derivatives to move towards the local minimum. **(Show how the derivatives in the algorithm work)** see class notes.

A detailed explanation of the derivation of the cost function for gradient descent: https://math.stackexchange.com/a/1695446/522739

Works with several dimensions. see class notes.

Make a run of gradient descent (single dimension) with the info in file gradient_examples.docx

There are many types of optimization functions (such as gradient descent), here are some visualizations http://www.benfrederickson.com/numerical-optimization/

code for a simple linear regression model with gradient descent code

2.3 Classification Models : Logistic Regression

Reading (logistic Regression Tutorial):
https://machinelearningmastery.com/implement-logistic-regression-stochastic-gradient-descent-scratch-python/ This reading will give a basic familiar look of the steps needed to perform logistic regression, the why of each of those steps will be discussed in class.

2.3.1 Hypothesis function:

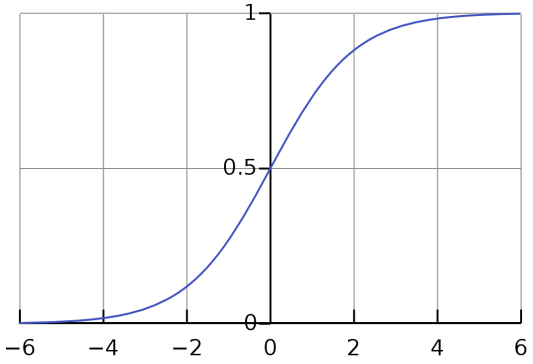
The reason why a linear regression algorithm is a bad classifier. Go through these examples with your instructor logistic_regression.docx

Activity: Make a manual run of logistic regression for the grades at the end of logistic_regression.

2.3.2 Activation functions:

Non-linear transformations

$$S(x) = \frac{1}{1 + e^{-x}}$$



2.3.3 Cost function:

Since we have a different model we now require a different cost function.

Go through these examples with your instructor Logistic_cost_function.docx

(if someone asks how this is so:
https://www.quora.com/Why-is-cost-function-of-logistic-regression-sigmoid-a-non-convex-function-But-natural-log-sigmoid-is-a-convex-function "the log of a sigmoid function is not convex")

Reading (Cross-Entropy explanation):
https://datascience.stackexchange.com/questions/9302/the-cross-entropy-error-function-in-neural-networks A nice discussion regarding the math behind cross-entropy and probability**.**

Reading (Other types of cost/loss functions):
http://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html Summary of several loss functions and what they are good for. This will be useful for when you try to make your projects and you don't know which loss function to use.

2.3.4 Optimization:

We use the same gradient descent for optimization.

Show how logistic regression looks in code, compare it with linear regression.

look at the code and try find each of the components of the algorithm in the code

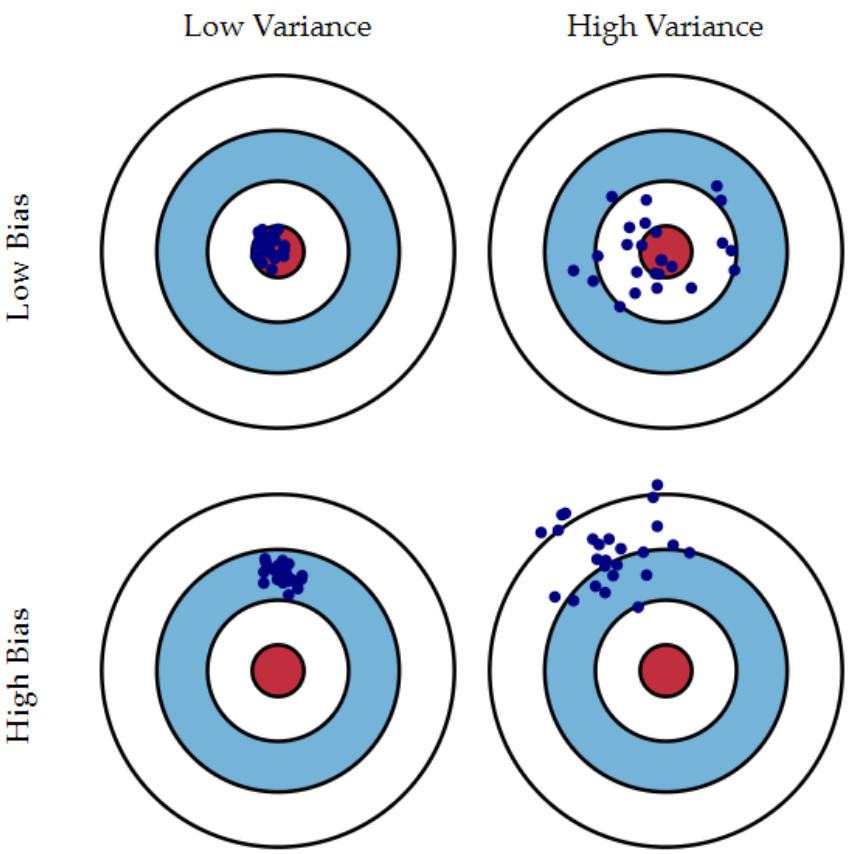
2.4 Diagnostics for Machine Learning Model (Problem detection and performance)

Reading (choosing tests):
https://machinelearningmastery.com/how-to-choose-the-right-test-options-when-evaluating-machine-learning-algorithms/
Make notes of the following concepts:
train/test split **cross validation**

Lets see how bias and variance look like when trying to make predictions. Example Variance vs Bias

2.4.1 Variance and 2.4.2 Bias

A nice metaphor of Bias and Variance from

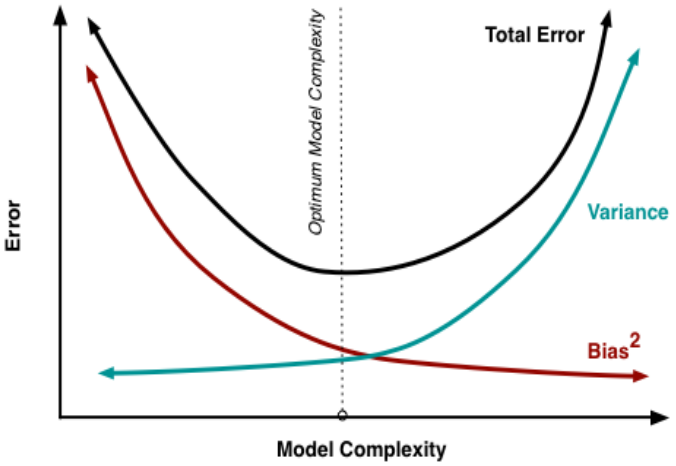


http://scott.fortmann-roe.com/docs/BiasVariance.html

2.4.3 Underfitting and 2.4.4 Overfitting

Learning to read the errors (Bias and Overfitting)

Training error



Cross-validation error "The purpose of cross-validation is model checking, not model building."

Read in class or after class to help you understand how to grade or rank your models:

Reading (Confusion Matrix):
https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/ This reading gives an overall summary of how to use the information provided by a confusion matrix.

Makes notes on concepts:

Confusion matrix Accuracy Precision

Video optional (ROC curves): https://www.dataschool.io/roc-curves-and-auc-explained/ This reading gives an overall summary of what is a ROC curve. Used to compare two or more classifiers

Makes notes on concepts:

ROC curve

2.5. Improving models:

If the data has already been cleaned and transformed then there are several tricks or hacks you can perform so that the model will improve its performance.

2.5.1 Adjusting Hyper Parameters

Using the **Learning rate**. Run Linear Reg. with different learning rates and compares times and results.

Exercises: **Implement scaling functions** and learning rate functions for your algorithms, compare the run time between the normal versions and the more complex ones.

Batch Size, there are different ways of adjusting the sizes of the instances provided to train, basically this refers to how many samples do you see before you update the model, i.e. begin a new epoch.

2.5.2 regression 1 hot encoding:

from source

One-Hot encoding.

id	color
1	red
2	blue
3	green
4	blue

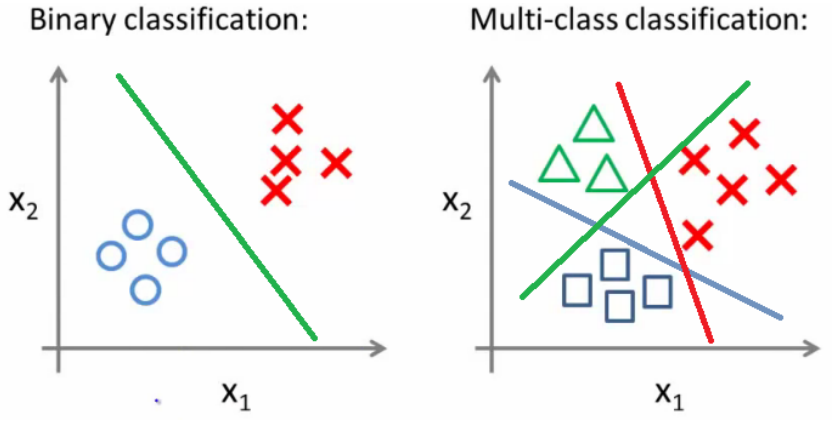
One Hot Encoding

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

2.5.3 Hacks for logistic regression:

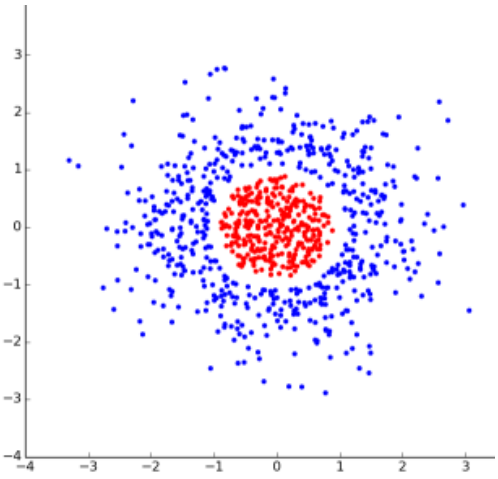
Using more complex functions to generate more accurate decision boundaries.

1 vs all strategy.



from source

Simulating Nonlinearities into a linear model



from source

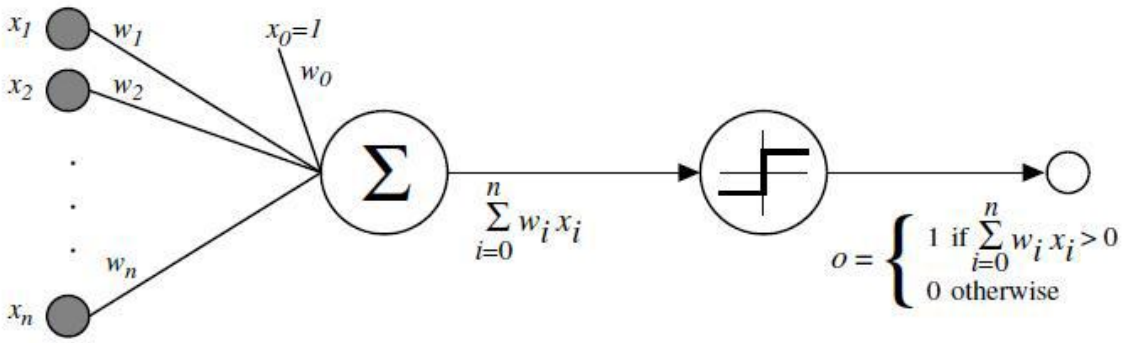
Explicar entregable para final de semana 3 (link de repo en canvas)

Entregable: Implementación de una técnica de aprendizaje máquina sin el uso de un framework.

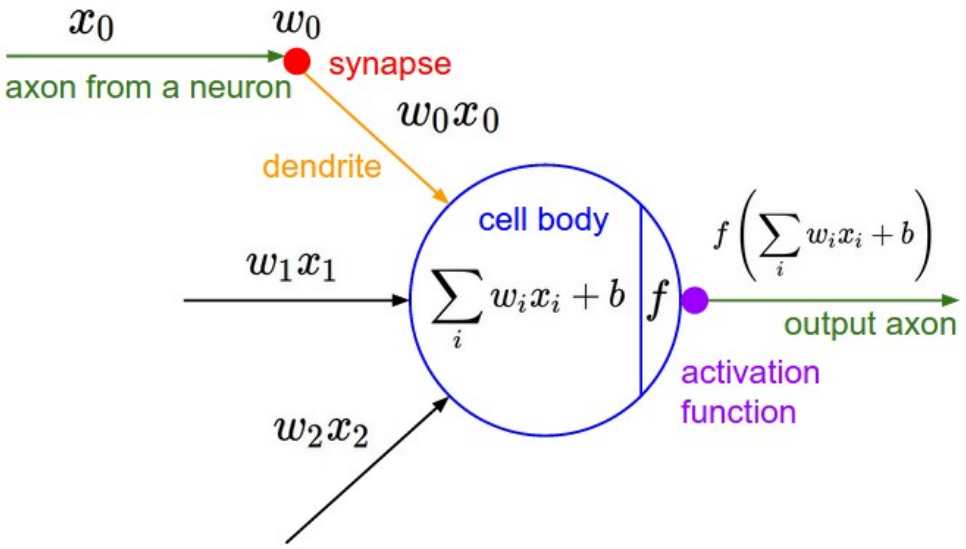
1. **Crea un repositorio de GitHub para este proyecto.**
2. **Programa uno de los algoritmos vistos en el módulo (o que tu profesor de módulo autorice) sin usar ninguna biblioteca o framework de aprendizaje máquina, ni de estadística avanzada. Lo que se busca es que implementes manualmente el algoritmo, no que importes un algoritmo ya implementado.**
3. **Prueba tu implementación con un set de datos y realiza algunas predicciones. Las predicciones las puedes correr en consola o las puedes implementar con una interfaz gráfica apoyándote en los visto en otros módulos.**
4. **Tu implementación debe de poder correr por separado solamente con un compilador, no debe de depender de un IDE o de un "notebook". Por ejemplo, si programas en Python, tu implementación final se espera que esté en un archivo .py no en un Jupyter Notebook.**
5. **Después de la entrega intermedia se te darán correcciones que puedes incluir en tu entrega final.**

2.6 Neural Networks

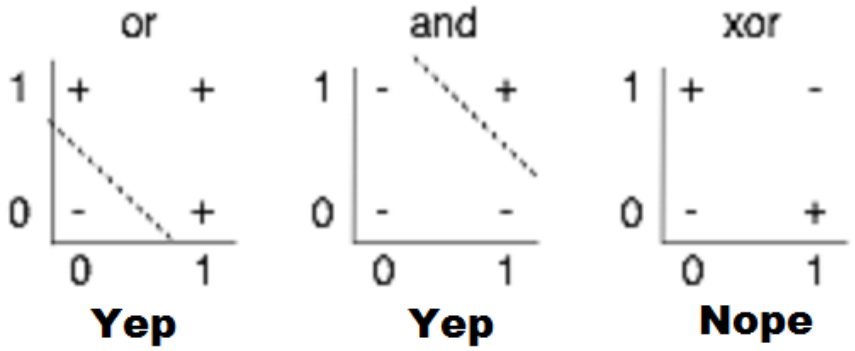
The historical roots. (Circuits, perceptron, multilayered, Deep Learning)



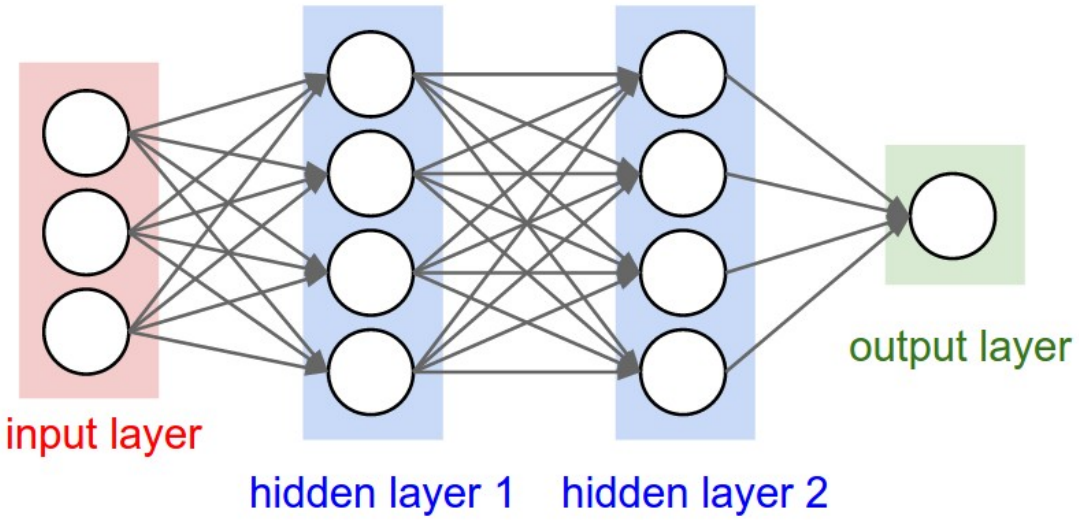
The biological inspiration. (The metaphor with neurons and the brain)



The problem with linearity (why are linear models not enough):



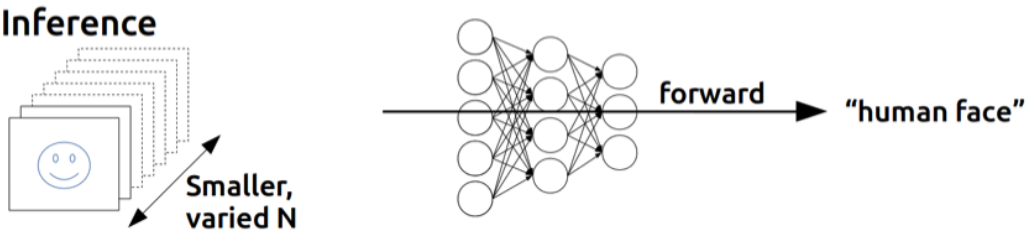
The structure (neurons/nodes, weights/params, activation values) see visualization



2.6.1 Feed forward (The prediction model)

Activity: do Feed Forward run by hand, choose weights AND, OR, and XNOR perceptron. Feed forward by hand

The hypothesis (Feed Forward Propagation)



Visualization

http://playground.tensorflow.org/ A more realistic (complex) visualization with interactive controls

2.6.2 Activation functions

Reading Activation functions:				
https://www.analyticssteps.com/blogs/7-types-activation-functions-neural-network make notes of:				
ReLu	Sigmoid	Tanh	Leaky ReLu	Softmax

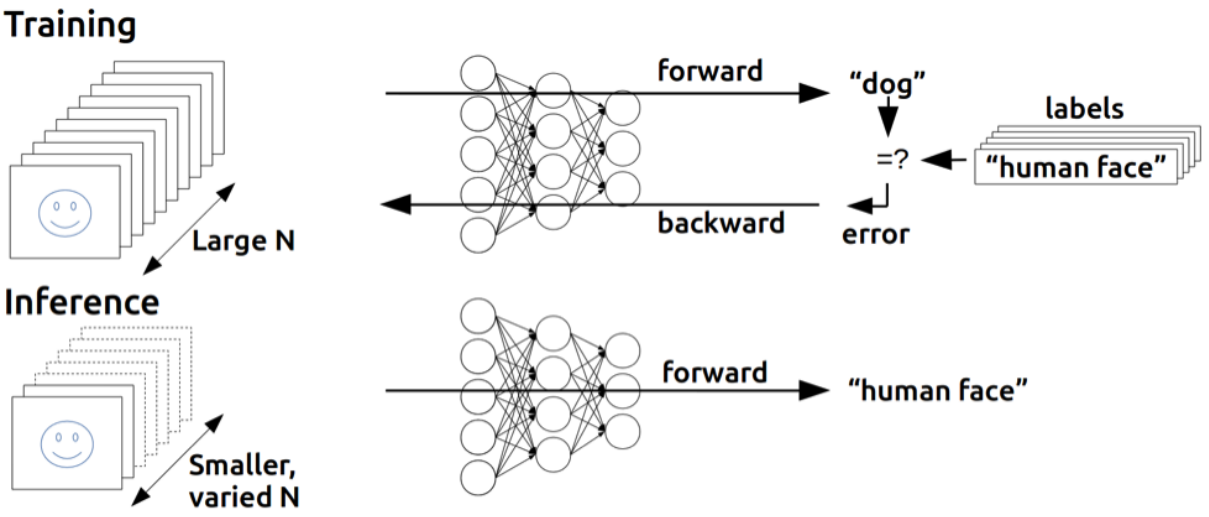
Activity: Program Feed forward function, remember to store the activation values in an accessible matrix. Here are some full networks use them as example just for the feedforward functions [example of simple nn] [vectorized implementation]

2.6.3. Cost Function:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

2.6.4 Back propagation

> **Reading Back propagation, step by step)**
> https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/ This post is a very nice explicit example of 1 run of forward and backpropagation by hand.



Go through the following example step by step with your teacher Back propagation summary

Integrating the whole network.

Iterating until convergence, # of epochs or until error is low enough.

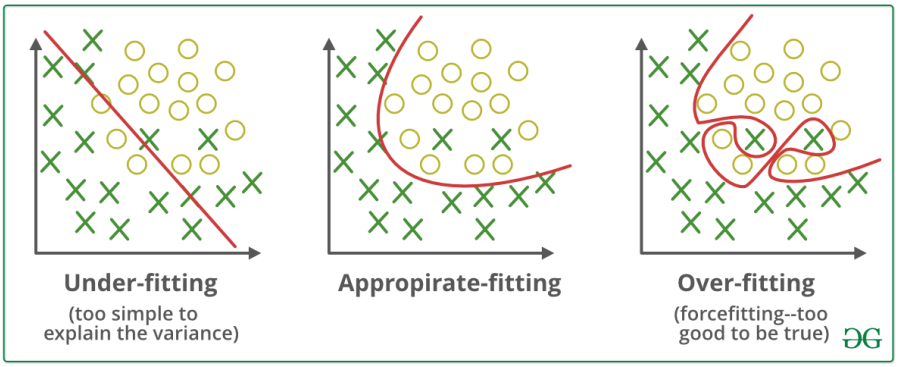
Activity: Go through the code full implementation of the example and try doing your own [example of simple nn] [vectorized implementation]

Activity: Go through the implementation of a Neural Network in tensor flow 2 [mnist fashion example]

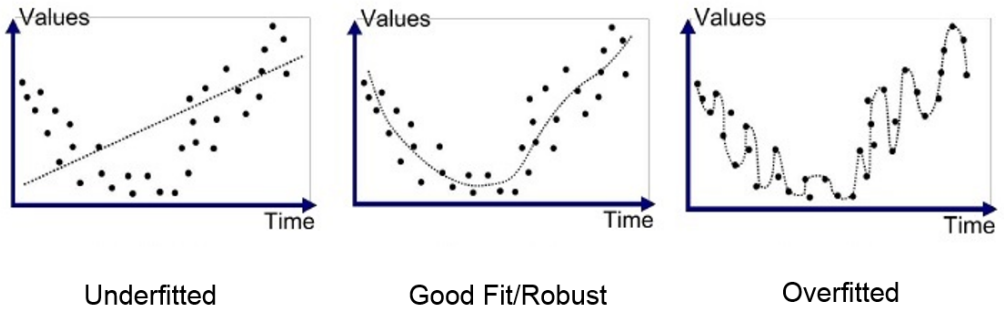
Diagnostics for Neural Networks:

Overfitting and Underfitting (How does it look in each of the different graphs)

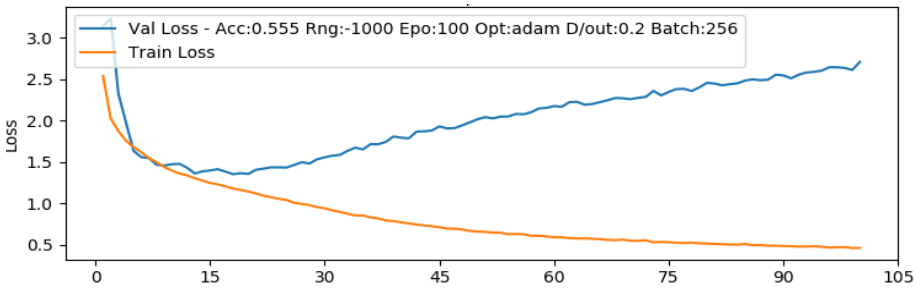
classification:

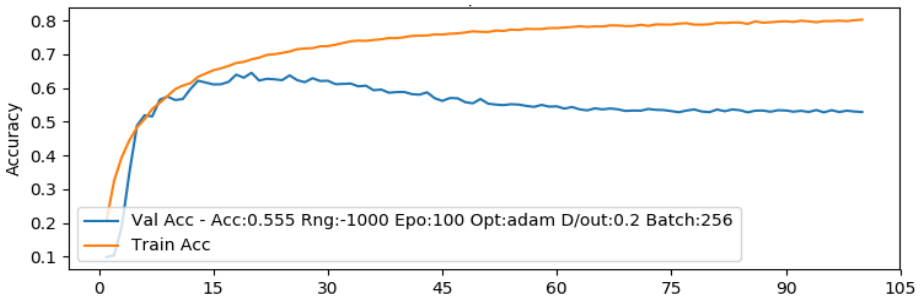


regression:



Error:





Cross-Validation**: use for detecting overfitting**

2.6.5 Neural Network Hyper-Parameters:

learning rate

number of layers

number of neurons

number of out-puts neurons

play with these factor and see how they affect the performance of the network, see f they make the network overfit or underfit visualization

2.6.6 Regularization: use for reducing overfitting

L2 Regularization or Weight Decay

$w = w - \alpha * \text{gradient} + \lambda/(2*m)||w||_2^2$

$w = (1- \alpha*\lambda/m)*w - \alpha * \text{gradient}$

Drop out**: use for reducing overfitting**

Batch normalization: **use to train big models faster**

Activity: run your own neural network. or use the visualization and make notes of what configurations work and test the ones you have seen before.

See the most common errors using: wrong weights in BP, inadequate cost functions, type of gradient descent is used.

Testing different configurations of the networks (layers, inputs, outputs)

Architectures for multiclass networks

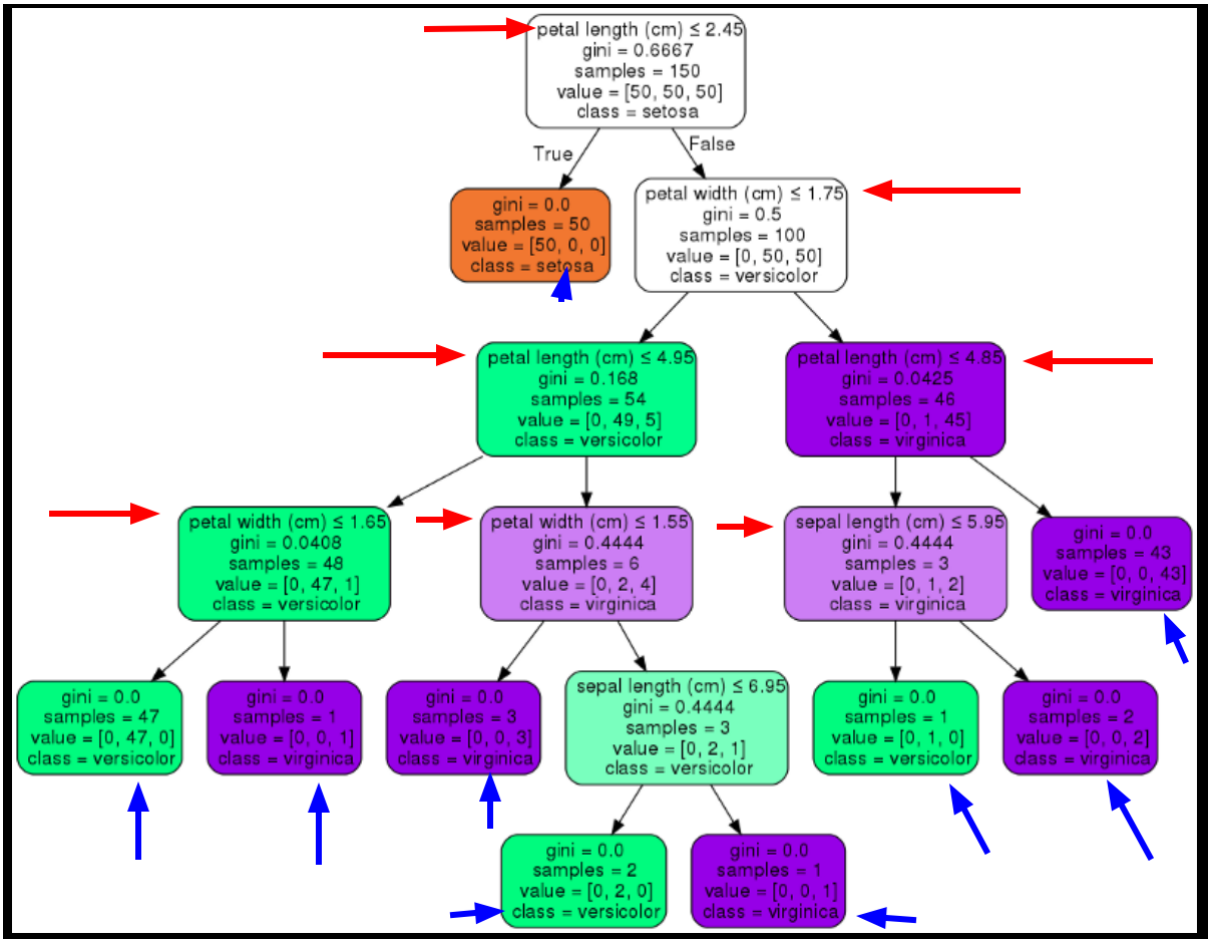
Activity : Go through the implementation of a Neural Network with L2 regularization for the Iris dataset in tensorflow 2

Activity : Go through the implementation of a Neural Network with batch normalization and dropout diabetes example in tensorflow 2

2.7 Supervised Learning for Classification:

2.7.1 Decision Trees

A decision tree is a supervised learning algorithm by Ross Quinlan that uses the data set to create a conditional tree structure that checks for specific values of attributes in each level of the true to quickly rule out other possible classes and reach the target class as quickly as possible. The following is the example of a decision tree for the iris data set.



Where do they fit in?

Shannon's information theory:

To build the tree we need to find a way to quickly rule out big branches. Enter **Entropy** in information by Claud Shannon.

We need to quantify what we know and what we don't know. A way to do this is by calculating the probabilities of other possible states within a restricted space. This in turn gives us an indicator of how certain we are that a variable has a given value or in this case that a specific class will be in that branch.

What is entropy in math?

$$H(X) = -\sum_{i=0}^{N-1} p_i \log_2 p_i$$

The negatory summation of the logarithm of the probabilities of a variable having each specific value. Multiplied by its own probability to keep the proportion.

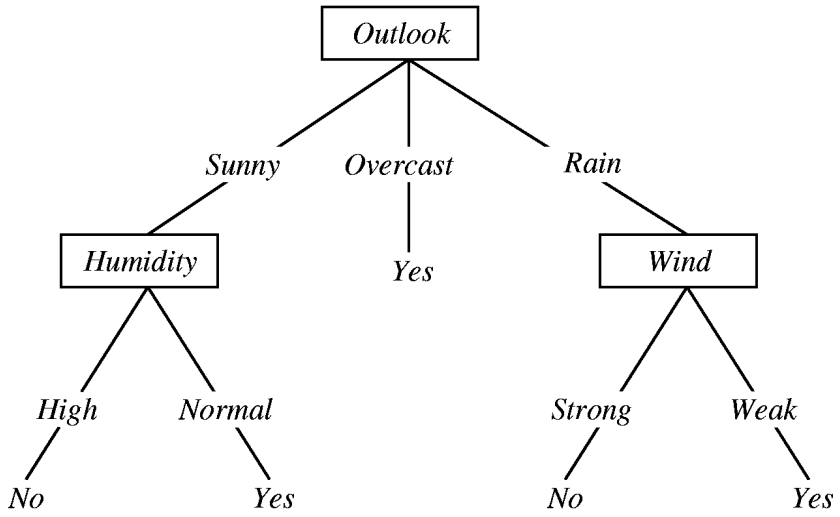
We use this measurement of probabilities (**entropy**) to calculate the **Information Gain.** If you have the entropy of a system and the you calculate the difference between the System entropy and the entropy of a part of that system, the resulting value is the information gain.

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

Decision trees calculate the entropy of each attribute in a dataset with respect to the class, and then they use that to get the information gain each would provide if they were selected as the current classifying node. The attribute with the highest information gain is the node you should always choose, and then you need to recursively repeat the procedure until the entropy for the next level is 0, which means that all of the members of a branch will have the same value in the class.

see the example in Should we play champ? of how to create an ID3 tree by hand. You can use the following empty template to make your own.

This is the tree you are trying to build:



Here is an implementation of a decision tree using sklearn

2.7.2 Random Forests

Random Forests are collections of small decision trees working together through an ensemble method. To understand random forest we must first have a basic grasp of ensemble methods.

Ensemble methods

ensemble methods are techniques for integrating several models of machine learning algorithms into a single application or model.

There are different approaches as to how to integrate the ensembles, the most popular ones are:

- Bagging
- Boosting
- Stacking

We will focus on baggin since it is the one used for Random Forests.

Bagging (AKA bootstrap aggregating)

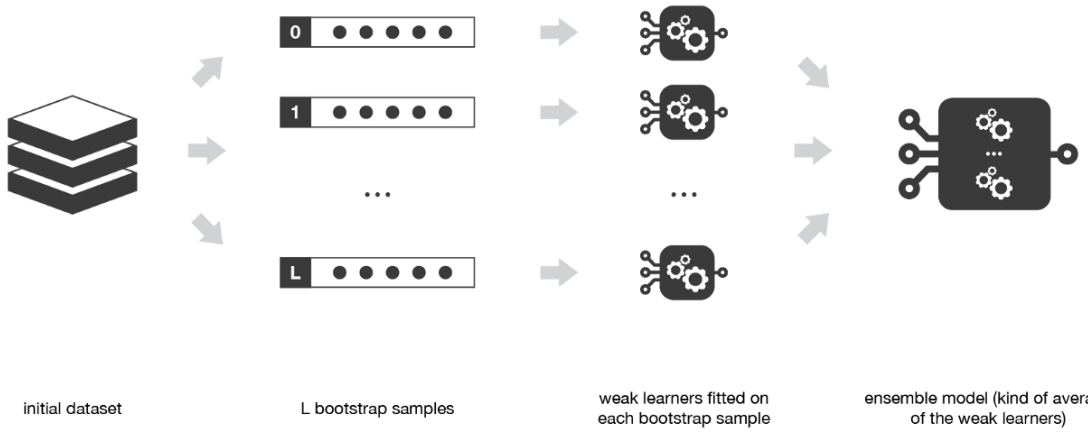
"When training a model, no matter if we are dealing with a classification or a regression problem, we obtain a function that takes an input, returns an output and that is defined with respect to the training dataset. Due to the theoretical variance of the training dataset (we remind that a dataset is an observed sample coming from a true unknown underlying distribution), the fitted model is also subject to variability: if another dataset had been observed, we would have obtained a different model.

The idea of bagging is then simple: we want to fit several independent models and "average" their predictions in order to obtain a model with a lower variance. However, we can't, in practice, fit fully independent models because it would require too much data. So, we rely on the good "approximate properties" of bootstrap samples (representativity and independence) to fit models that are almost independent.

First, we create multiple bootstrap samples so that each new bootstrap sample will act as another (almost) independent dataset drawn from true distribution. Then, we can fit a weak learner for each of these samples and finally aggregate them such that we kind of "average" their outputs and, so, obtain an ensemble model with less variance than its components. Roughly speaking, as the bootstrap samples are approximately independent and identically distributed (i.i.d.), so are the learned base models. Then, "averaging" weak learners outputs do not change the expected answer but reduce its variance" taken from Ensemble methods: bagging, boosting and stacking by Joseph Rocca.

Bootstrapping or Sampling with replacement

When you sample with replacement, you would choose one sample, use it for any purpose and then you put it back into the dataset instead of excluding it, and then choose another sample. This means that you can get repeated samples, though it is unlikely.

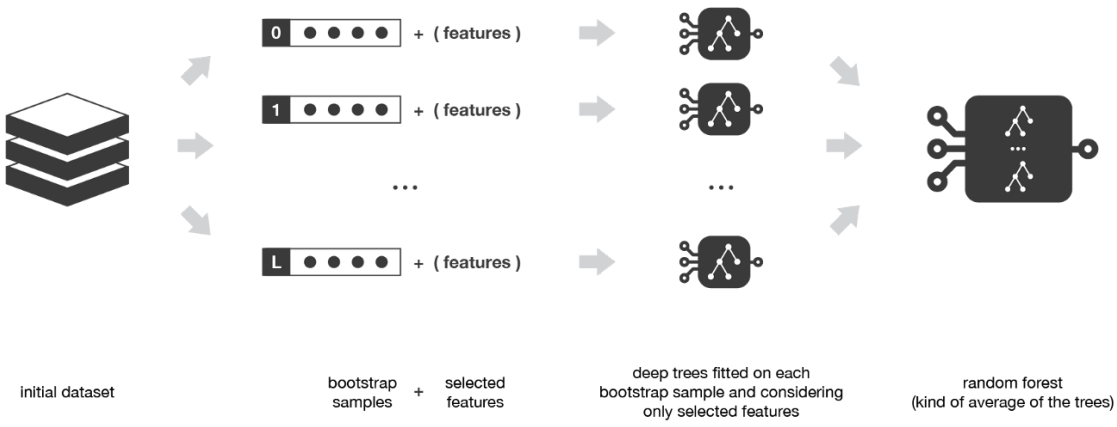


taken from Ensemble methods: bagging, boosting and stacking by Joseph Rocca.

Go through the example implementation of ensemble methods using scikit.

Random Forests

Random forests are an integration of the baggings method with decision trees. So instead of having one big expensive tree prone to overfitting, we can have several small trees which can be trained in parallel integrated through the baggins method. The selection of features is also randomized for each tree. This means that each tree is trained with an independent sub sample of the data set and on randomly selected features. This will create very diverse trees which will help our ensemble to generalize better.



taken from Ensemble methods: bagging, boosting and stacking by Joseph Rocca.

Go through the example of implementation of tree and random forest in sklearn.

Explicar entregable para final de semana 5 (link de repo en canvas)

Entregable: Uso de framework o biblioteca de aprendizaje máquina para la implementación de una solución.

1. Crea un repositorio de GitHub para este proyecto.
2. Programa uno de los algoritmos vistos en el módulo (o que tu profesor de módulo autorice) haciendo uso de una biblioteca o framework de aprendizaje máquina. **Lo que se busca es que demuestres tu conocimiento sobre el framework y como configurar el algoritmo.**
3. Prueba tu implementación con un set de datos y realiza algunas predicciones. Las predicciones las puedes correr en consola o las puedes implementar con una interfaz gráfica apoyándote en los visto en otros módulos.
4. Tu implementación debe de poder correr por separado solamente con un compilador, no debe de depender de un IDE o de un “notebook”. Por ejemplo, si programas en Python, tu implementación final se espera que esté en un archivo .py no en un Jupyter Notebook.
5. Después de la entrega intermedia se te darán correcciones que puedes incluir en tu entrega final.

Explicar entregable para final de semana 5 (link de repo en canvas)

Entregable: Análisis y Reporte sobre el desempeño del modelo.

1. Escoge una de las 2 implementaciones que tengas y genera un análisis sobre su desempeño en un set de datos. Este análisis lo deberás documentar en un reporte con indicadores claros y gráficas comparativas que respalden tu análisis.
2. El análisis debe de contener los siguientes elementos:
    1. Separación y evaluación del modelo con un conjunto de prueba y un conjunto de validación (Train/Test/Validation).
    2. Diagnóstico y explicación el grado de bias o sesgo: bajo medio alto
    3. Diagnóstico y explicación el grado de varianza: bajo medio alto
    4. Diagnóstico y explicación el nivel de ajuste del modelo: underfitt fitt overfitt
3. Basándote en lo encontrado en tu análisis utiliza técnicas de regularización o ajuste de parámetros para mejorar el desempeño de tu modelo y documenta en tu reporte cómo mejoró este.

Ejemplos de reportes de análisis:
https://drive.google.com/drive/folders/1qHvsKGRkDCf022DxOgc5BXSeZW1648EW?usp=sharing

2.7.3 Improving Trees and Random Forests

The hyper parameters of a tree

Max Depth

Entropy vs Purity

Instances related

The hyper parameters of a Random Forest

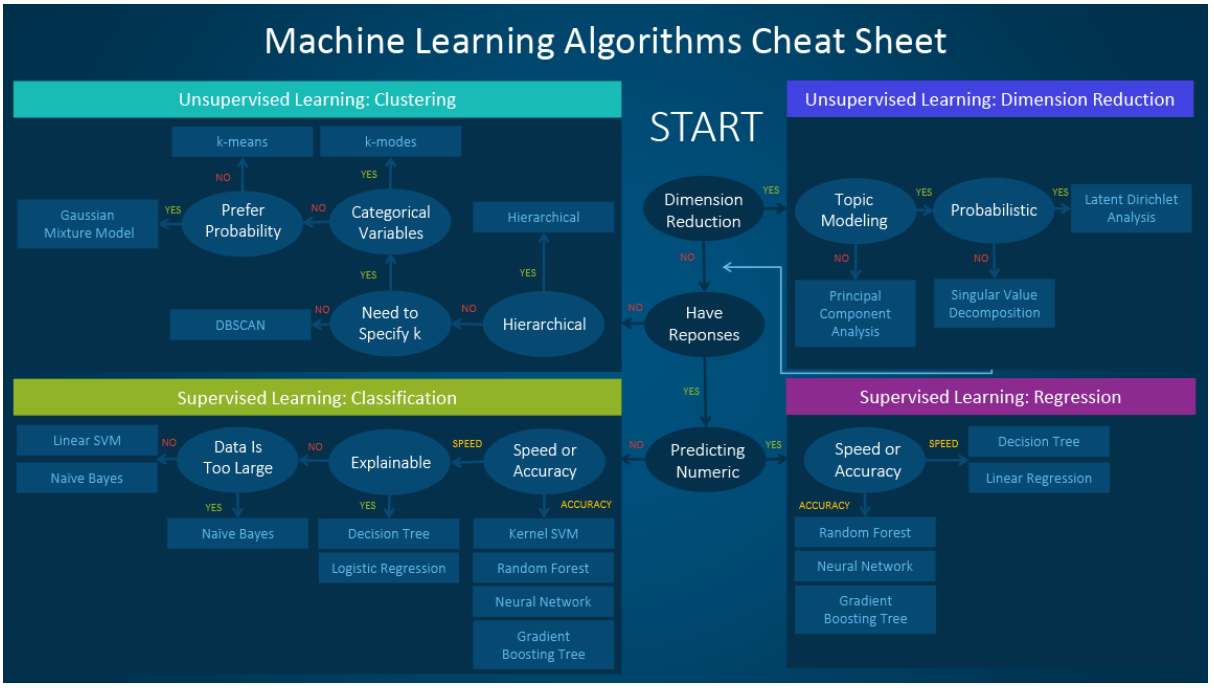
Forest related

Tree related

Instances related

2.8 Unsupervised Learning:

Unsupervised learning refers to a group of techniques that can make inference from unlabeled data. The way to use techniques varies widely but here is a very nice map to provide context



source

The main branch we will be working with is **Clustering**

> "Clustering is a set of techniques used to partition data into groups, or clusters. Clusters are loosely defined as groups of data objects that are more similar to other objects in their cluster than they are to data objects in other clusters. In practice, clustering helps identify two qualities of data:
>
> **Meaningful** clusters expand domain knowledge. For example, in the medical field, researchers applied clustering to gene expression experiments. The clustering results identified groups of patients who respond differently to medical treatments.
>
> **Useful** clusters, on the other hand, serve as an intermediate step in a data pipeline. For example, businesses use clustering for customer segmentation. The clustering results segment customers into groups with similar purchase histories, which businesses can then use to create targeted advertising campaigns." taken form https://realpython.com/k-means-clustering-python/

2.8.1 K-means

Partitional Clustering

Partitional clustering methods have several strengths:

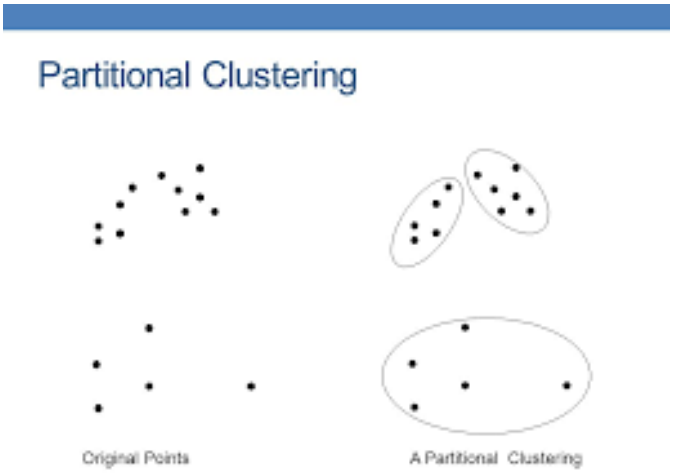
They work well when clusters have a spherical shape.

They're scalable with respect to algorithm complexity.

They also have several weaknesses:

They're not well suited for clusters with complex shapes and different sizes.

They break down when used with clusters of different densities.



source

Open **KMeans_sklearn_blobs.ipynb** (Partitional Clustering) run the code and make your own comments on it.

2.8.2 DBScan

The strengths of density-based clustering methods include the following:

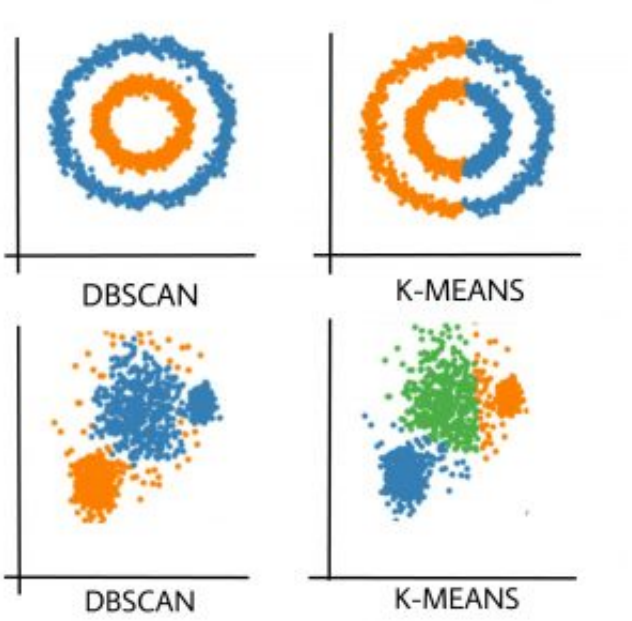
They excel at identifying clusters of nonspherical shapes.

They're resistant to outliers.

The weaknesses of density-based clustering methods include the following:

They aren't well suited for clustering in high-dimensional spaces.

They have trouble identifying clusters of varying densities.



source

Open **KMeans vs DBSCAN sklearn moons evaluation ari metric** (Density Clustering) run the code and make your own comments on it.

2.8.3 Hierarchical Clusters

Hierarchical clustering methods have several strengths:

They often reveal the finer details about the relationships between data objects.

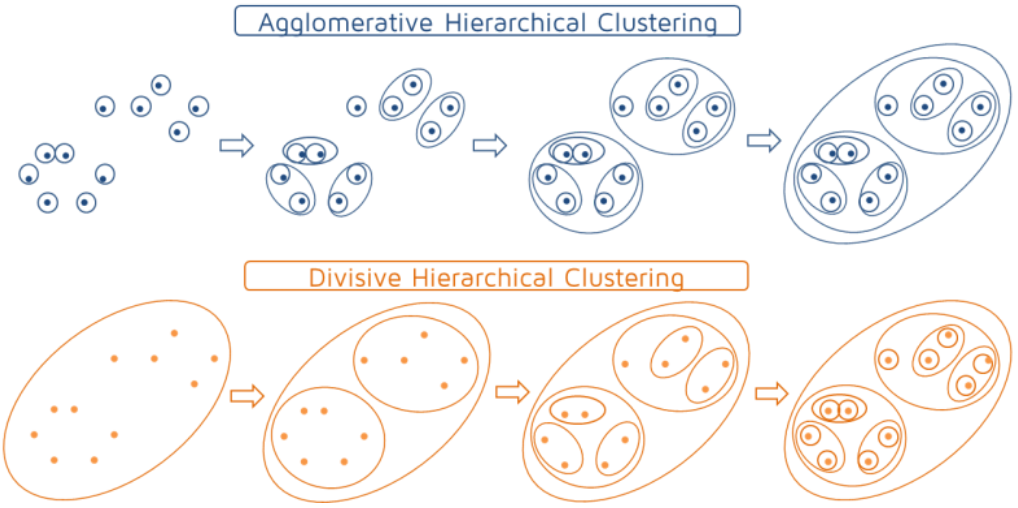
They provide an interpretable dendrogram.

The weaknesses of hierarchical clustering methods include the following:

They're computationally expensive with respect to algorithm complexity.

They're sensitive to noise and outliers.

Break down when used with clusters of different densities.



Source

Open **Hierarchical Clustering** (Hierarchical) run the code and make your own comments on it.

Activity complete the following notebook, where you should analyze a Dow Jones Index Data Set. Clusters in finance example

Go through this example of how to use pipelines and pca for preprocessing data in sklearn clusters and PCA to detect groups of cancer.

2.9 Improving different ML models

work on doubts of any of the models covered during the module.