Data Science Stream 2021

Agenda for da day

- 10am: Intro and Setup
- 11am: Basics of Data (Cleaning & etc)
- 12pm: BREAK!
- 1pm: Foundation for model trainings
- 2pm: Your turn!
- 3pm: PowerBI
- 4pm: Conclusion
- Afterwards: Q&A, Networking and etc

Finding Friends

https://github.com/NZMSA/

What is Machine Learning?

	x		y
	1000kg	$y = mx + c$	40mpg
	3000kg		20mpg
	2000kg		?

What is Machine Learning?

	x		y
	1000kg	$y = mx + c$	40mpg
	3000kg		20mpg
	2000kg		30mpg

What are we setting up?

- Install Python
- Install Pip
- Setup Virtual Environment
- Installing Packages
- Create a Jupyter Notebook

Why Python?

- The most used programming language in the data science field.
- Has dedicated libraries to do data collection and cleaning, data exploration, data modeling, and data visualization.

Setup - Install python

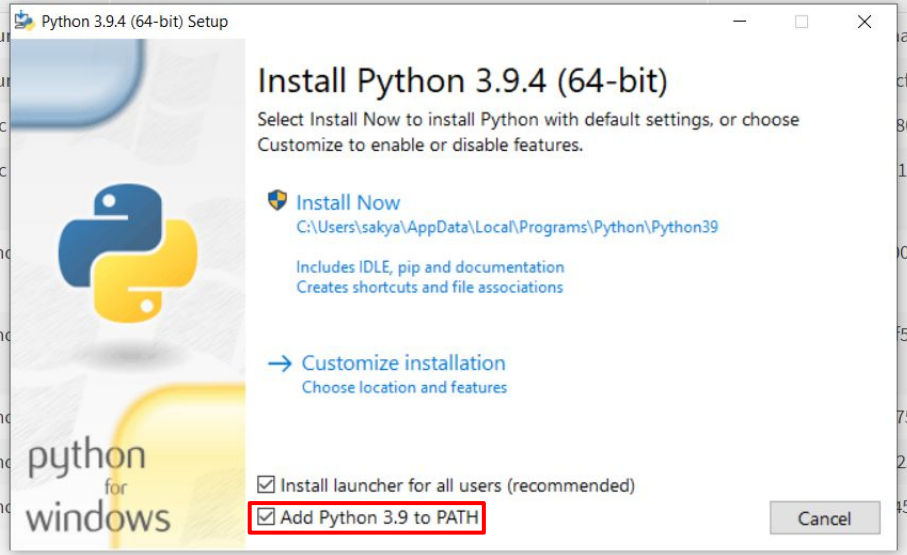
https://www.python.org/downloads/release/python-394/

Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		cc8507b3799ed4d8baa7534cd8d5b35f	25411523	SIG
XZ compressed source tarball	Source release		2a3dba5fc75b695c45cf1806156e1a97	18900304	SIG
macOS 64-bit Intel installer	Mac OS X	for macOS 10.9 and later	2b974bfd787f941fb8f80b5b8084e569	29866341	SIG
macOS 64-bit universal2 installer	Mac OS X	for macOS 10.9 and later, including macOS 11 Big Sur on Apple Silicon (experimental)	9aa68872b9582c6c71151d5dd4f5ebca	37648771	SIG
Windows embeddable package (32-bit)	Windows		b4bd8ec0891891158000c6844222014d	7580762	SIG
Windows embeddable package (64-bit)	Windows		5c34eb7e79cfe8a92bf56b5168a459f4	8419530	SIG
Windows help file	Windows		aaacfe224768b5e4aa7583c12af68fb0	8859759	SIG
Windows installer (32-bit)	Windows		b790fdaff648f757bf0f233e4d05c053	27222976	SIG
Windows installer (64-bit)	Windows	Recommended	ebc65aaa142b1d6de450ce241c50e61c	28323440	SIG

Setup - Install Python and Add to PATH

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Sour		a7534cd8d5b35f	25411523	SIG
XZ compressed source tarball	Sour		cf1806156e1a97	18900304	SIG
macOS 64-bit Intel installer	Mac		80b5b8084e569	29866341	SIG
macOS 64-bit universal2 installer	Mac		151d5dd4f5ebca	37648771	SIG
Windows embeddable package (32-bit)	Win		00c6844222014d	7580762	SIG
Windows embeddable package (64-bit)	Win		56b5168a459f4	8419530	SIG
Windows help file	Win		7583c12af68fb0	8859759	SIG
Windows installer (32-bit)	Win		233e4d05c053	27222976	SIG
Windows installer (64-bit)	Win		450ce241c50e61c	28323440	SIG



Why pip?

- A package-management system written in Python used to install and manage software packages.
- It makes it easier for you to install different libraries.
- Alternatively, you can use Anaconda.

Setup - Pip install !!

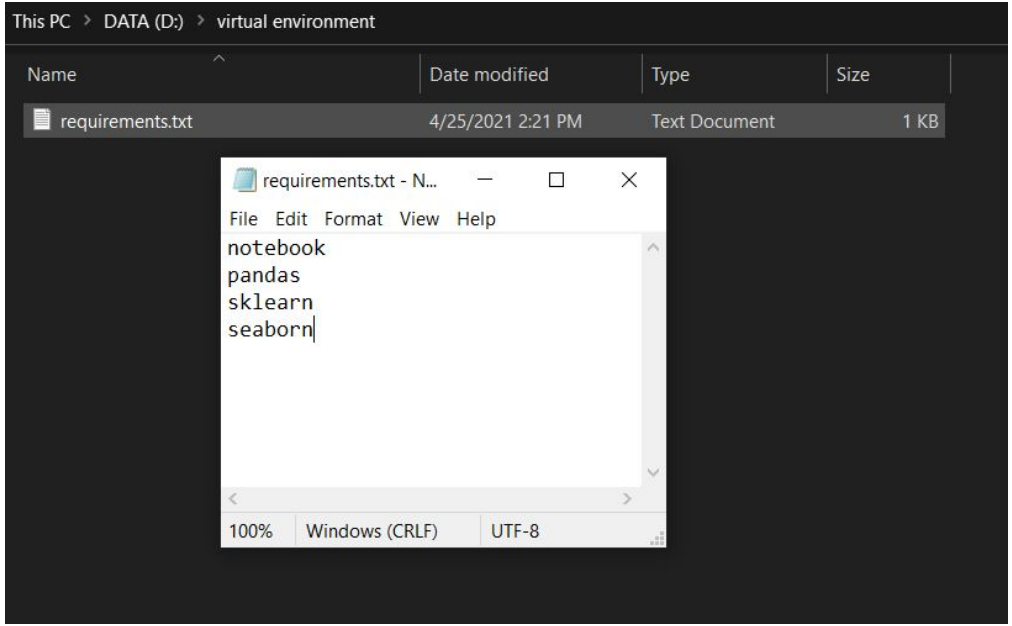
- Download the get-pip.py to your computer
  - `curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py`
- Open a command line
- Run the following command:
  - `python get-pip.py` (Windows)
  - `python3 get-pip.py` (MacOS)

Why use Virtual Environment?

- Keep your dependencies separate.
- Different projects might use different versions of libraries.

Setup - Create virtual environment

1. Optional: Create your Git repo
2. Create and Go to a folder where you want to store the venv
3. Create a .txt file called “requirements.txt”
4. Open the file and typed these in (these will be the libraries we will use) ->



Setup - Create Virtual environment

1. In the same folder, open terminal.

   On windows, type these:

```
python -m venv venv

pip install -r requirements.txt

jupyter notebook
```

2. On mac, type these

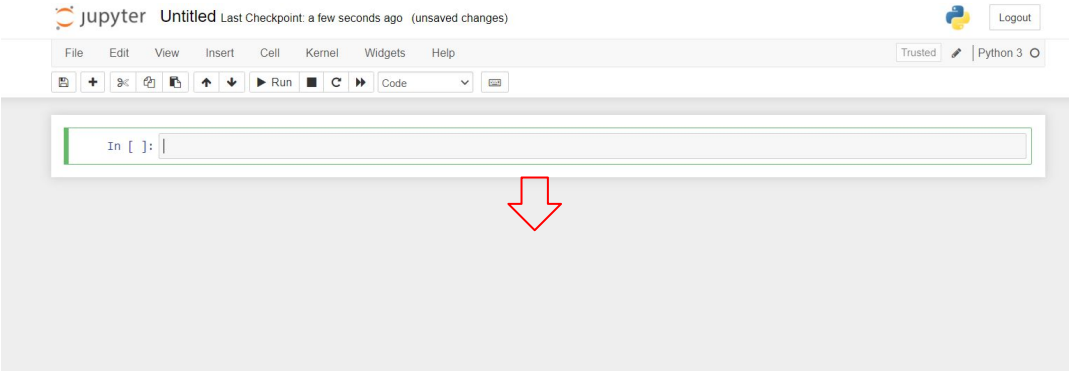
```
python3 -m venv venv

source ./venv/bin/activate

pip install -r requirements.txt

jupyter notebook
```


Why Jupyter Notebook?

- It’s good
- Allow us to see both code and results
- Run cells to cells for better understanding
- Easy and secure

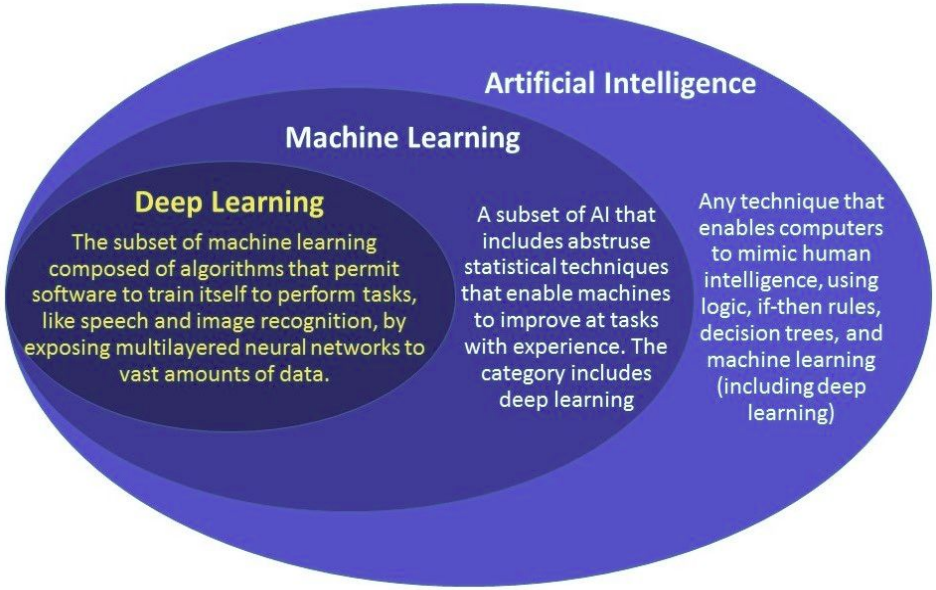
Setup - Create a notebook

1. New
2. Python 3
3. You are ready to do some Data Science!



Coming Next - Data & ML & AI & etc!!

AI vs ML vs DL


Artificial Intelligence
Any technique that enables computers to mimic human intelligence, using logic, if-then rules, decision trees, and machine learning (including deep learning)
Machine Learning
A subset of AI that includes abstruse statistical techniques that enable machines to improve at tasks with experience. The category includes deep learning
Deep Learning
The subset of machine learning composed of algorithms that permit software to train itself to perform tasks, like speech and image recognition, by exposing multilayered neural networks to vast amounts of data.

Machine Learning Basics

1. Supervised Learning - Neural Nets , SVM , Naive Bayes , Random Forest
2. Reinforcement Learning - Q Learning , A3C
3. Semi Supervised Learning - GAN
4. Unsupervised Learning - K Means Clustering (Might be replaced !)
5. **Self Supervised Learning - Transformer-based architectures like BERT**

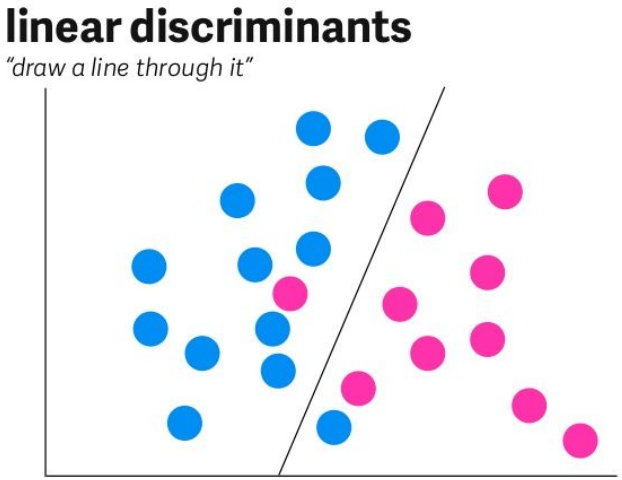
Yann LeCun Cake Analogy 1.0

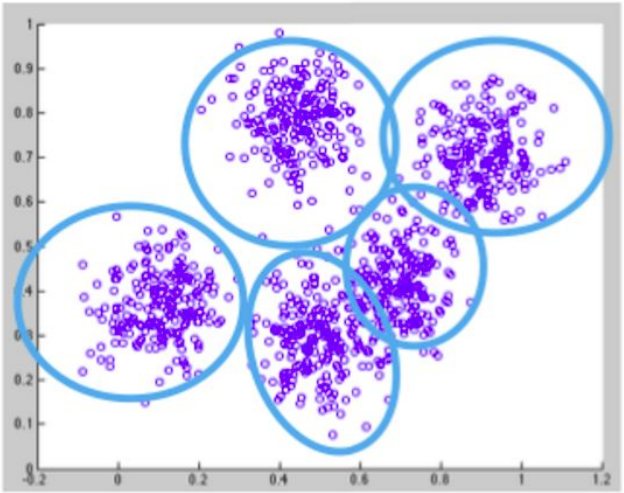
- "Pure" Reinforcement Learning (cherry)
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- Supervised Learning (icing)
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- Unsupervised/Predictive Learning (cake)
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

Yann LeCun Cake Analogy 2.0

- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Self-Supervised Learning (cake génoise)**
  - The machine predicts any part of its input for any observed part.

Supervised vs Unsupervised





Formulating the Problem

Feature Matrix (X) m x n

Contains m examples with n features

PassengerId	Pclass	Name	Sex	Age
1	3	Braund, Mr. Owen Harris	male	22
2	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38
3	3	Heikkinen, Miss. Laina	female	26
4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35
5	3	Allen, Mr. William Henry	male	35

Label set (y) n x 1

Each label y_i corresponds to an example x_i

Survived
0
1
1
1
0

Formulating the Problem (continued)

Given example x_i

Predict

Corresponding label $\hat{y}_i$

PassengerId	Pclass	Name	Sex	Age
12	1	Bonnell, Miss. Elizabeth	female	58

Survived
0 or 1

Goal: Learn patterns and predict better than random guess

Train-Test Split

Common splits:

Small datasets

- 30%/70%

Larger datasets

- 10%/10%/80%

Big Data

- 0.5%/0.5%/99%

https://en.wikipedia.org/wiki/Training,_validation,_and_test_sets

Golden Rule of ML

- Test data **should not** ever influence training of the algorithm
- Test data **should not** be edited

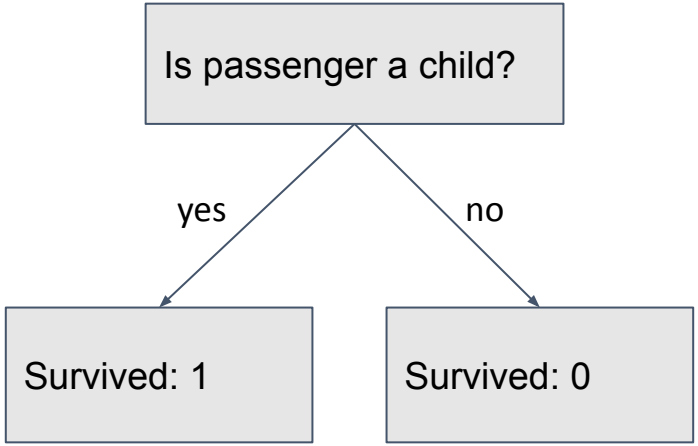
Possible violations

- Training more algorithms until one gets higher accuracy
- Cherry picking performance statistics
- Changing model / Hyperparameters after testing

Decision Stumps

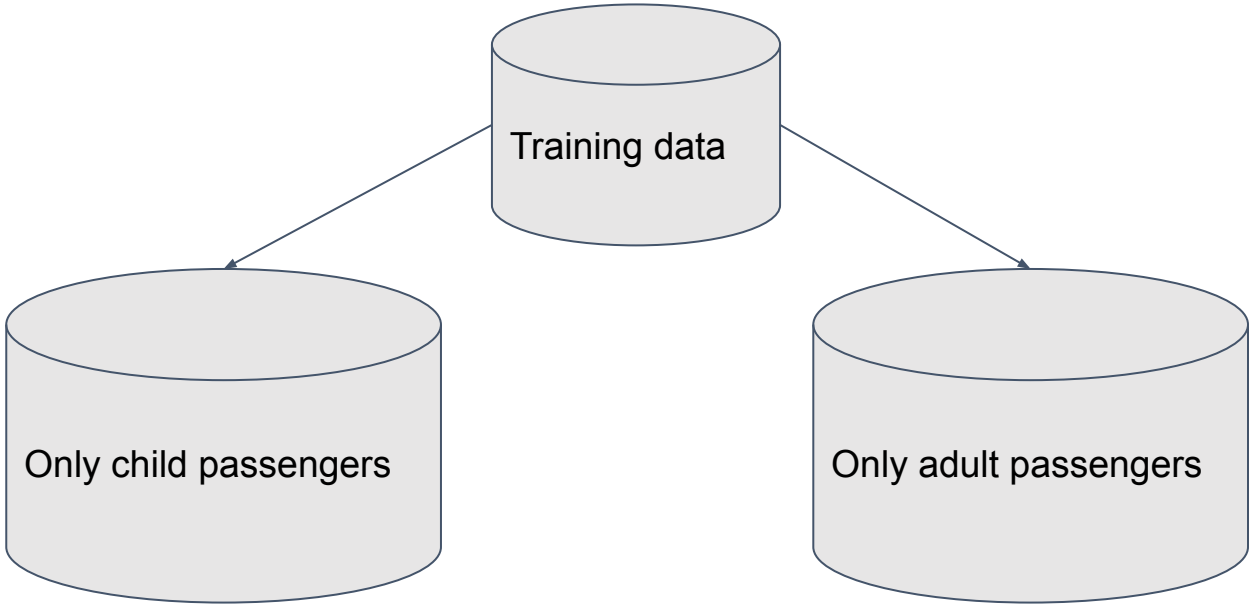
Choose only 1 rule based on 1 feature

Choose most accurate rule



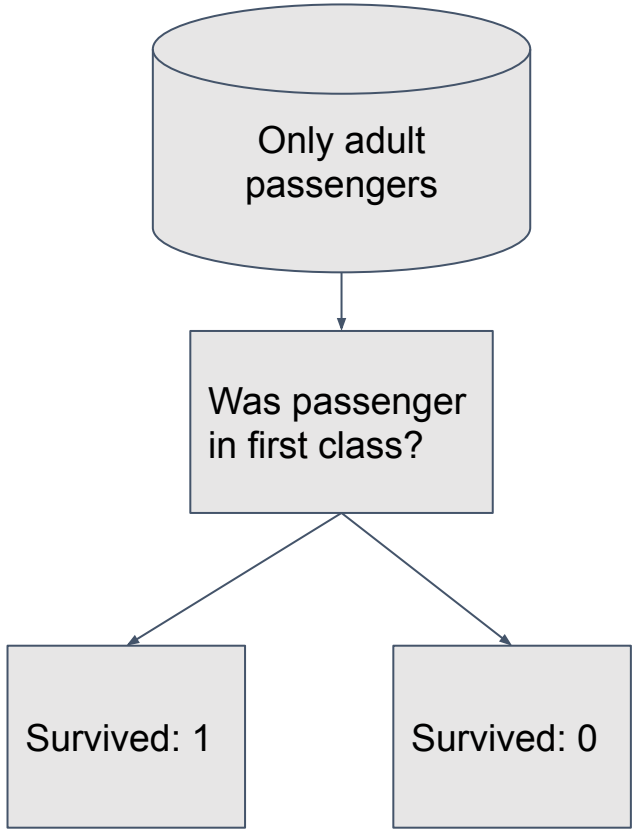
Decision Stumps (Continued 1)

Subset data based on last rule - **is passenger a child?**



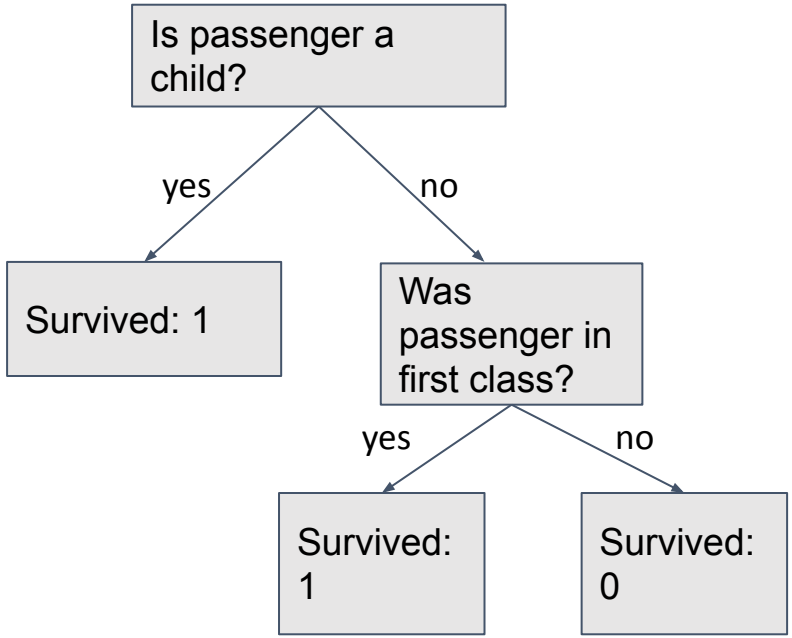
Decision Stumps (Continued 2)

- Take one of the subset datasets
- Find most accurate rule
- Make sure that rule uses a different feature



Decision Trees

- Use greedy recursive splitting to create decision stumps
- This creates a decision tree
- Internal nodes are rules, each rule is based on a different feature
- Leaves are corresponding classes
- The more rules you add the higher the accuracy



Score Function

How to choose the rule?

We choose the split that **decreases entropy** in the labels the most and **maximises information** gain:

$$information\ gain = \underbrace{entropy(y)}_{\text{entropy before split}} - \frac{\overbrace{n_{yes}}^{\text{number of examples satisfying rule}}}{n} \underbrace{entropy(y_{yes})}_{\text{entropy examples satisfying rule}} - \frac{n_{no}}{n} entropy(y_{no})$$

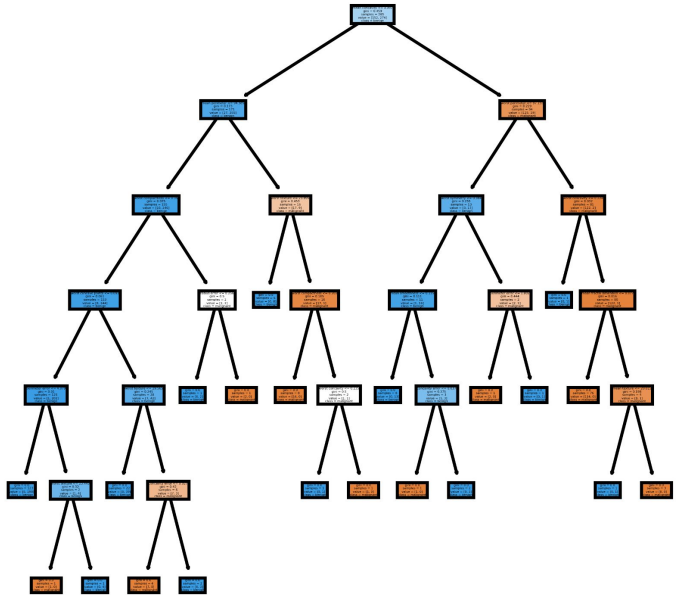
$$entropy(s) = -p_{sick} \log_2 p_{sick} - p_{not\ sick} \log_2 p_{not\ sick}$$

Hyperparameters

Hyperparameter is the parameter whose value is used to control the learning process.

For Decision Tree:

- Depth
- Splitting criterion
- Minimum number of leaves



https://towardsdatascience.com/visualizing-decision-trees-with-python-scikit-learn-graphviz-matplotlib-1c50b4aa68dc

Decision Tree On Numeric Spaces

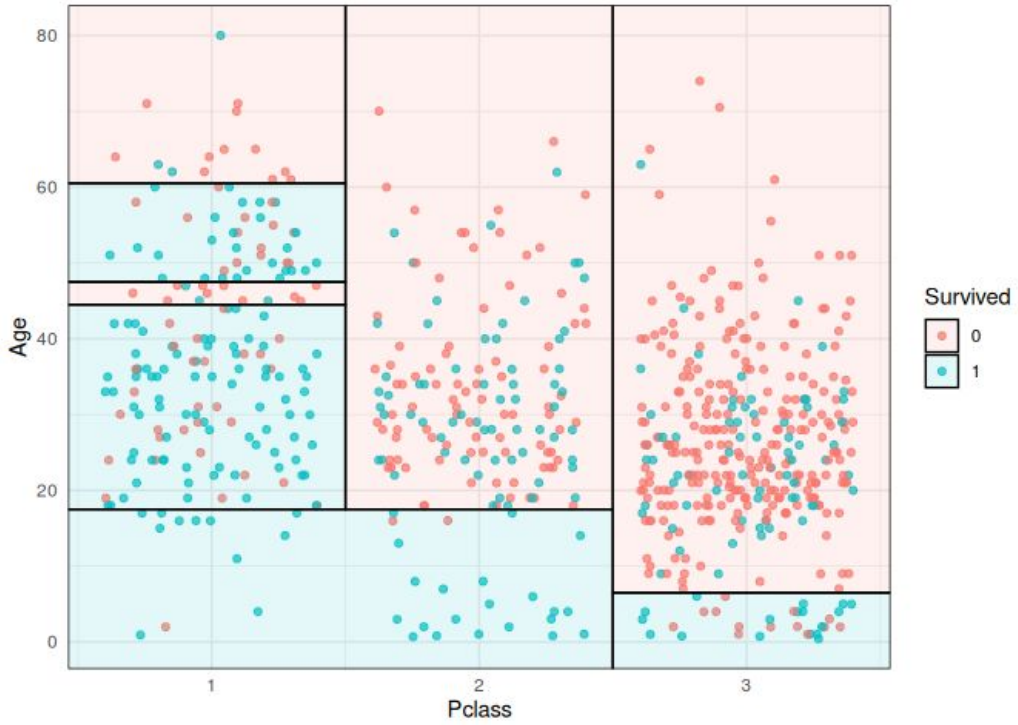
Find the split that gives largest information gain:

Pclass < 2

Repeat splits:

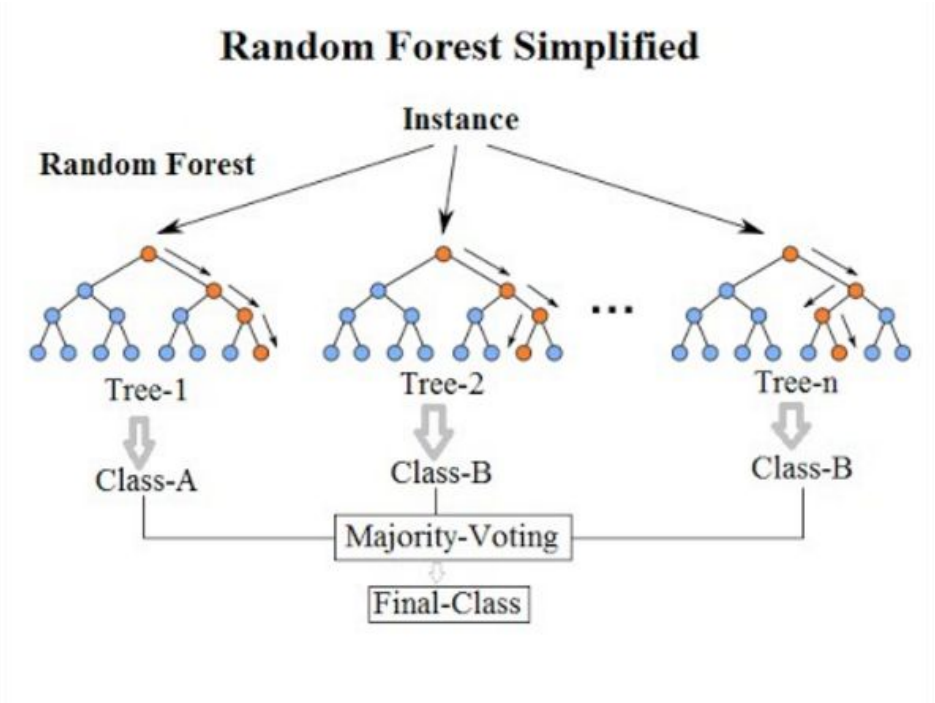
Age < 20

Partition whole space into subspaces using rules



https://paulvanderlaken.com/2020/03/31/visualizing-decision-tree-partition-and-decision-boundaries/

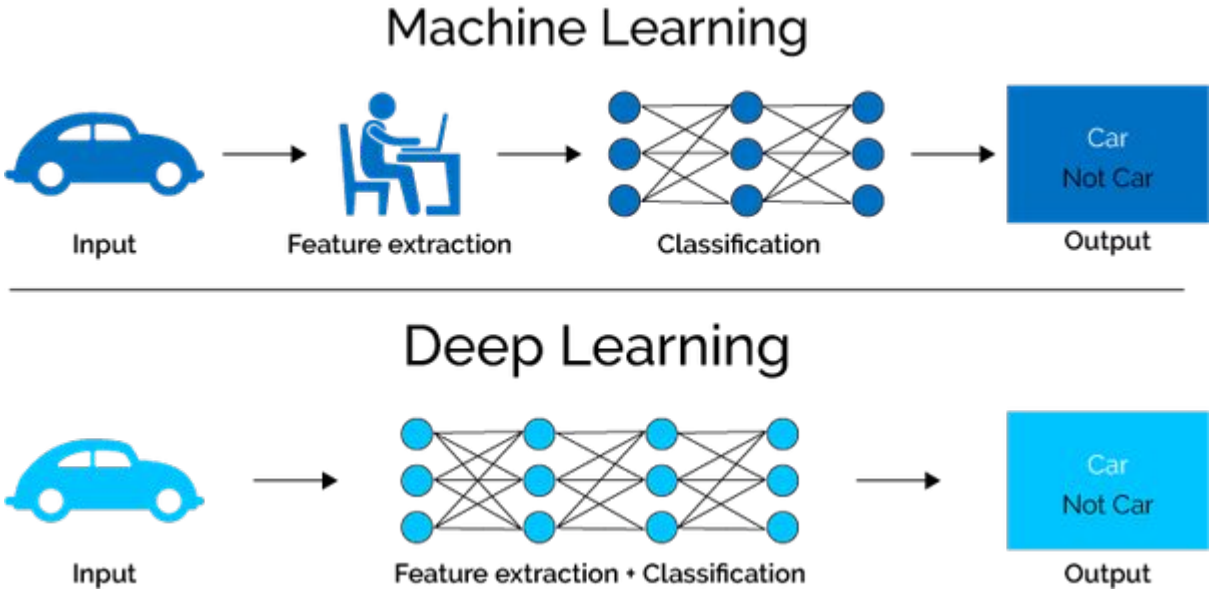
Random Forest Classifier



https://williamkoehrsen.medium.com/random-forest-simple-explanation-377895a60d2d

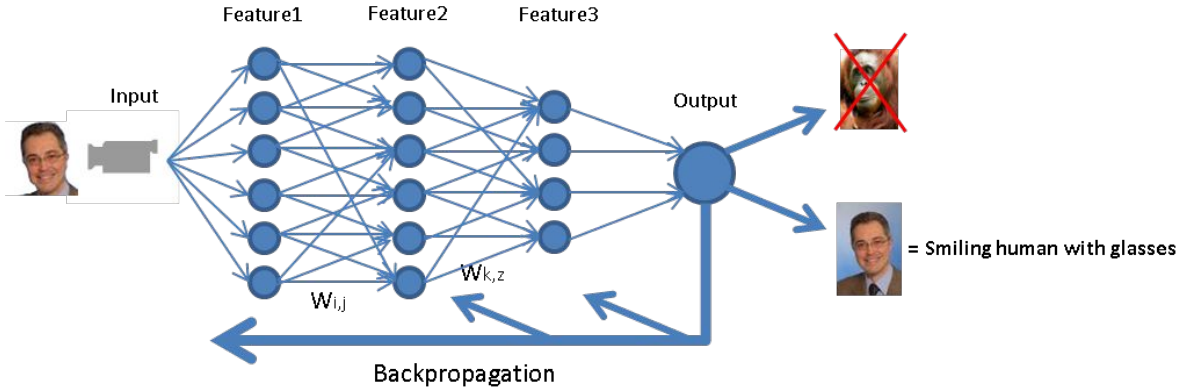
Where To Start ?

- This is different from **traditional Machine Learning (**SVM , Decision Trees ...)



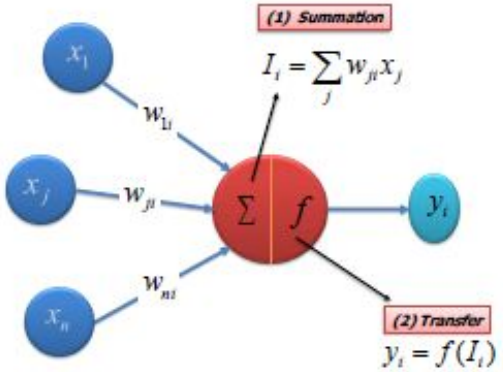
Neural Architecture - Basics

- Trained on a loss/objective function
- Trainable parameters are called weights
- Need lot of data

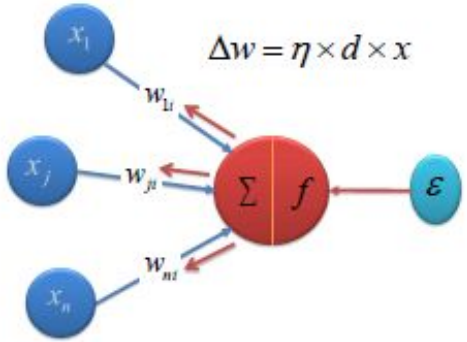


Perceptron is the building box !

Feedforward Input Data

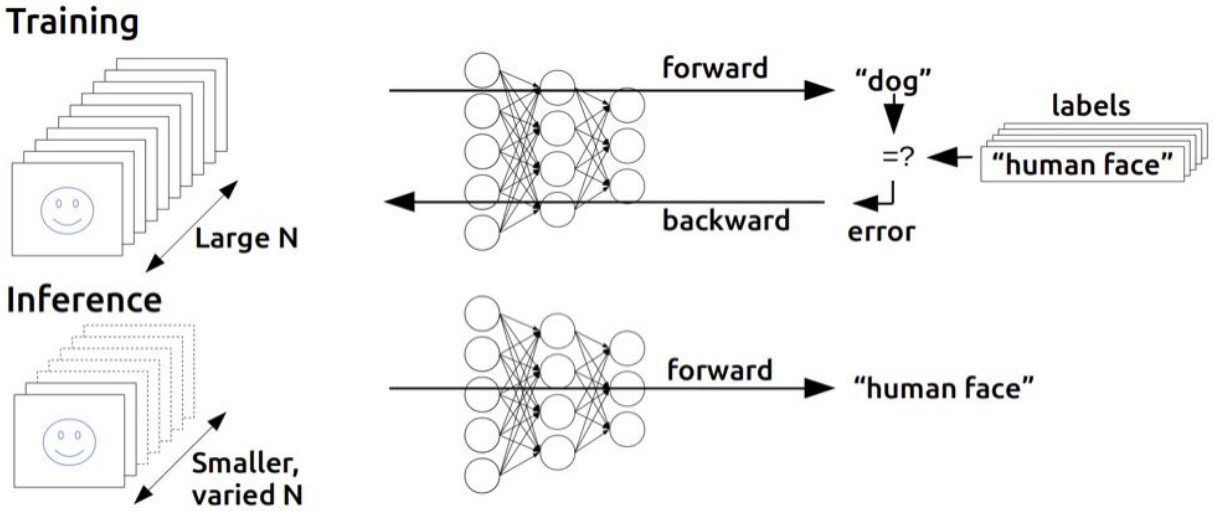


Backward Error Propagation



Nonlinearity applied inside a node

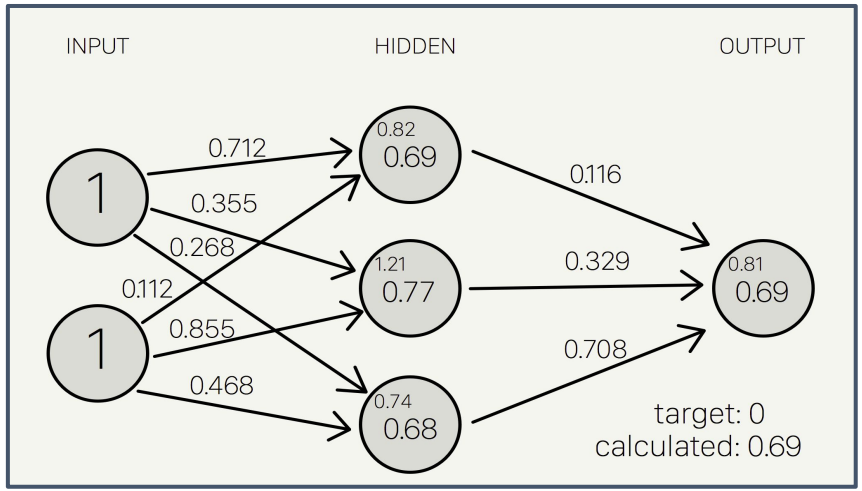
Training VS Inference



Training Phase (3 steps)

Forward Pass

- Calculating the loss

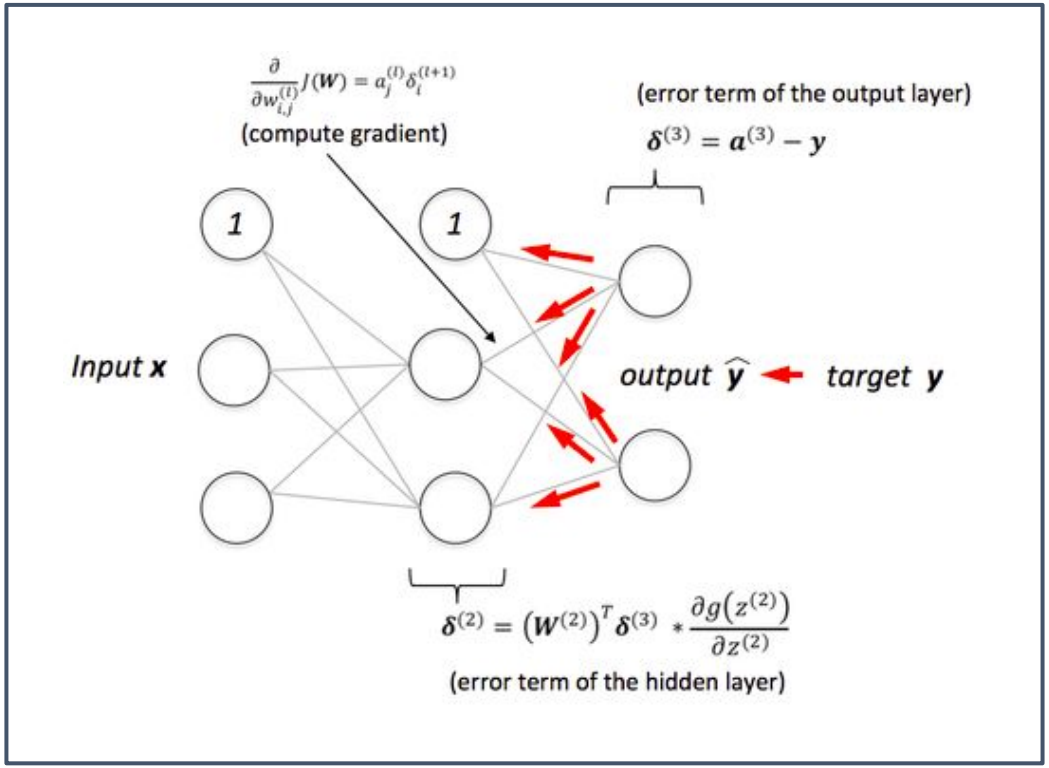


$$E(\mathbf{w}) = \sum_{i=1}^{N} \left(y_i - f_{\mathbf{w}}(\mathbf{x}_i)\right)^2$$

$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right)$$

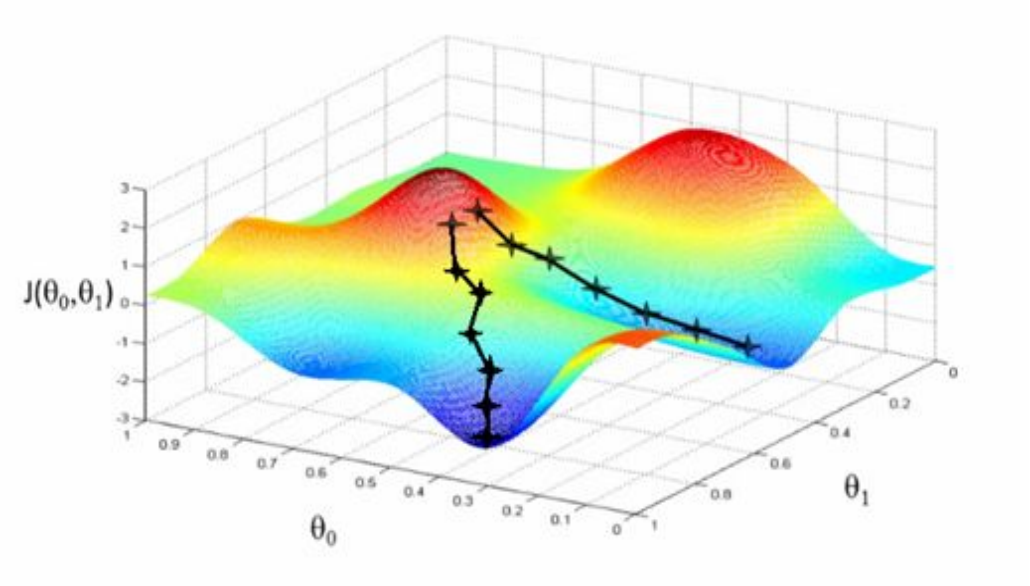
Backward Pass

- Backpropagation Algorithm
- Distributing Gradients



Optimization

- Reducing the loss
- Updating the weight matrix



Updating weights - SGD

Gradient descent algorithm

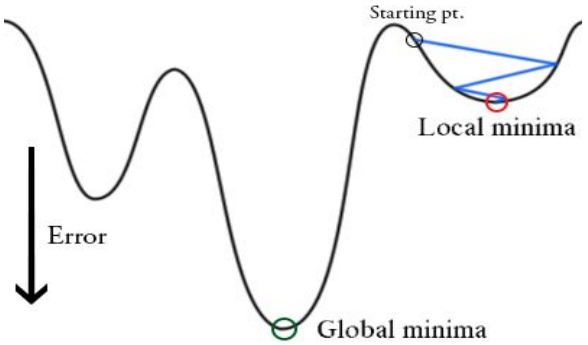
repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

(for $j = 1$ and $j = 0$)

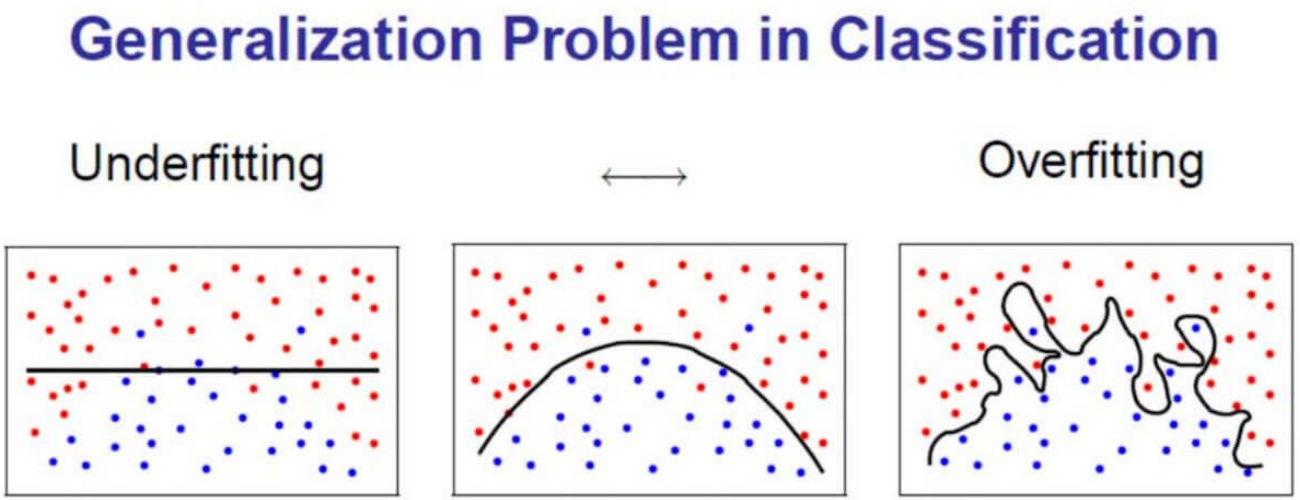
}

Updating weights

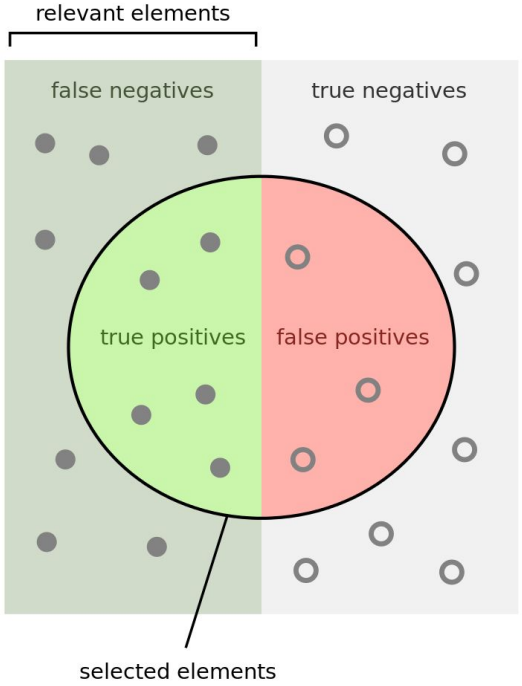


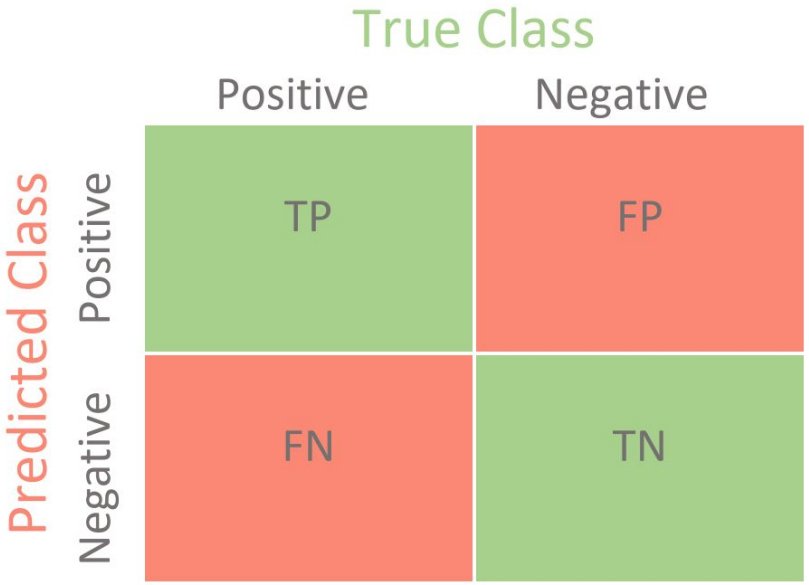
Loss Function

Overfitting (Main Issue)



Evaluation





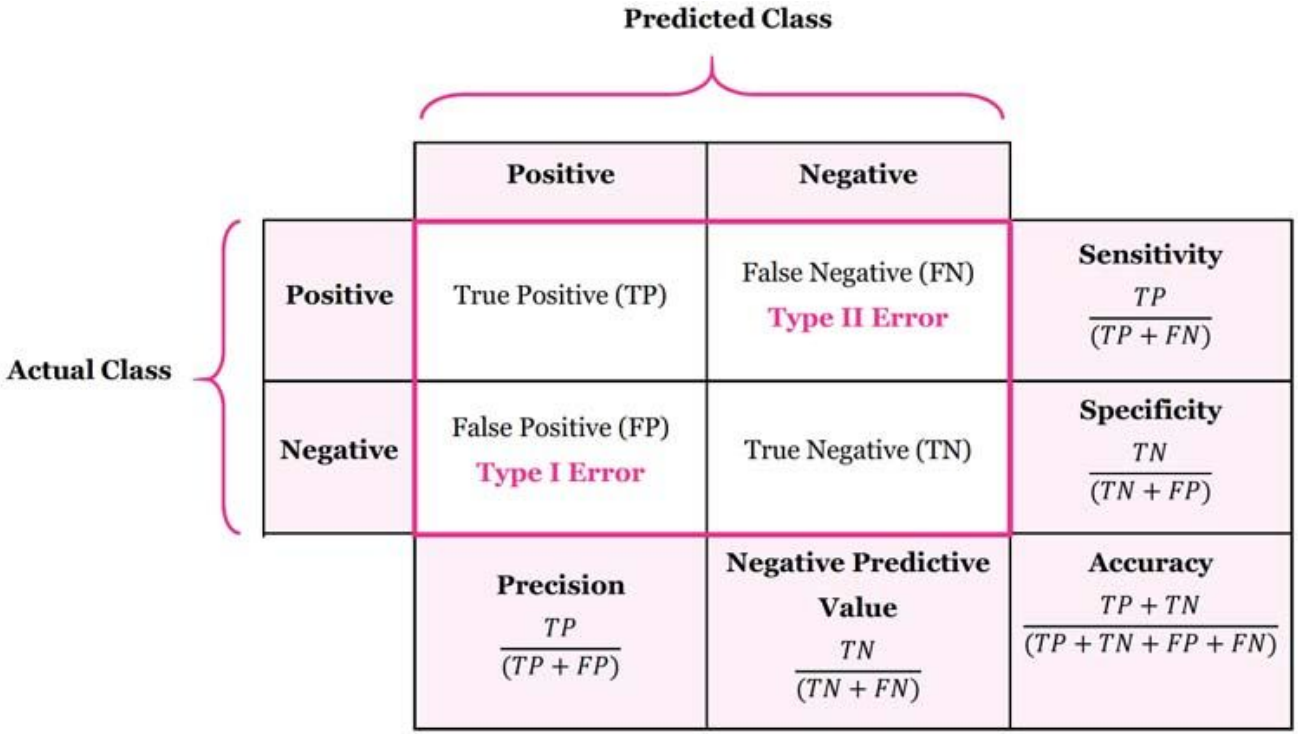
https://www.r-bloggers.com/2020/12/weighting-confusion-matrices-by-outcomes-and-observations/

Evaluation

Accuracy:
Beware, accuracy can be misleading for imbalanced data

F-1 Score:

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$



https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html

Final Steps

Once you are satisfied with the model:

- Retrain model on the whole dataset