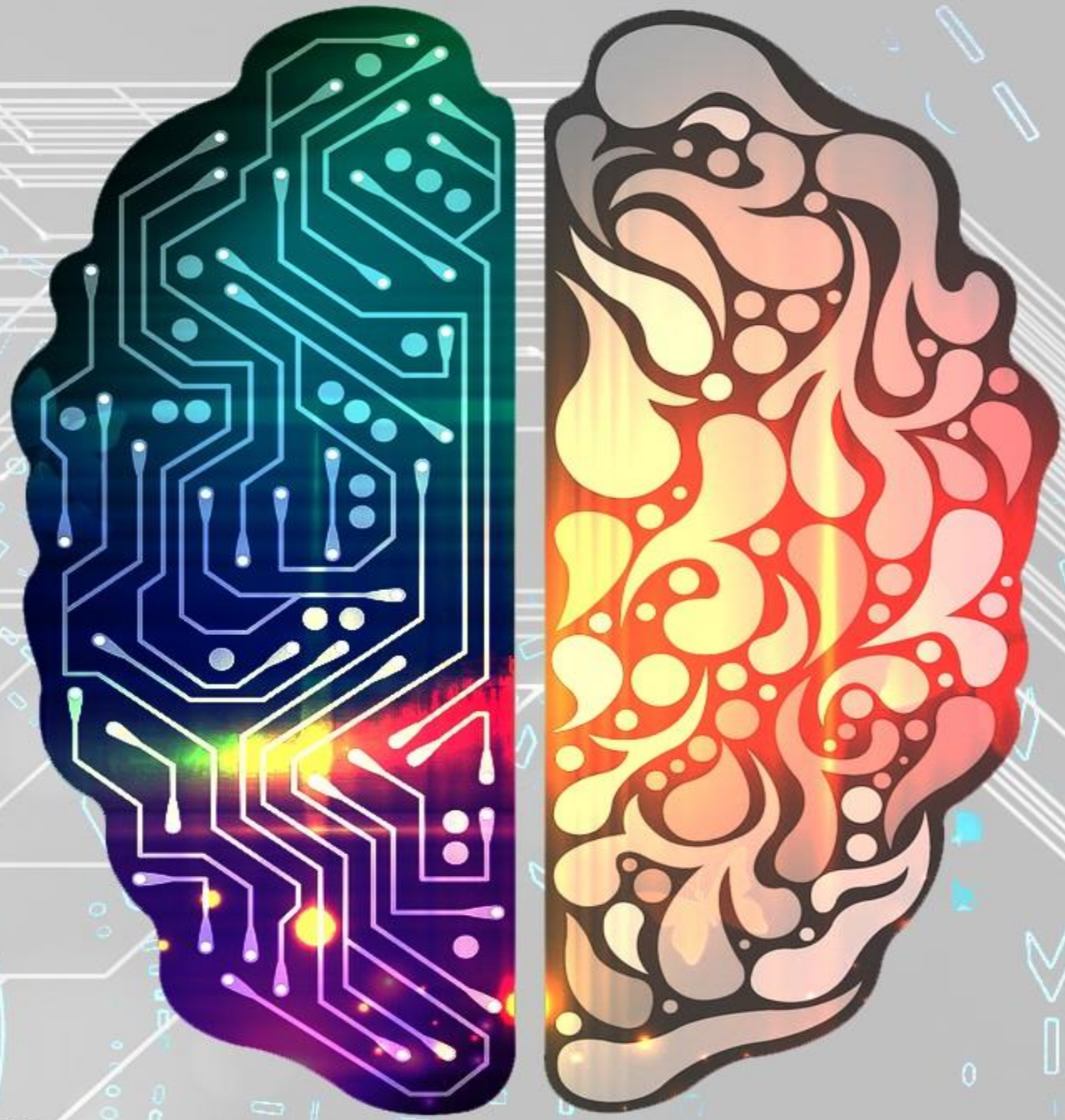
“ MEMES FOR TEACHING”

MACHINE LEARNING

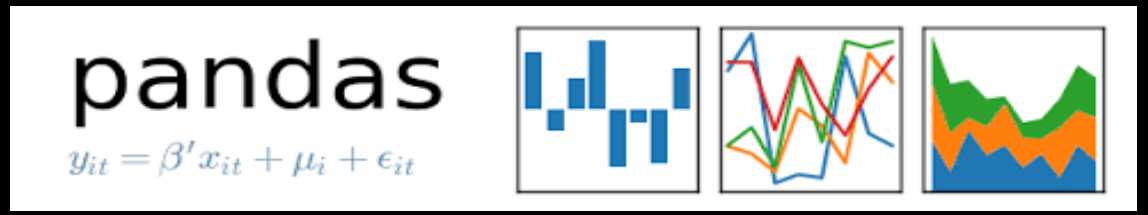


DAY1- IOSD DEVELOPMENT WEEK

AGENDA:

1. Introduction to Python 3.7/3.8
2. Introduction to Data Visualisation and Normalization
3. Introduction to Machine Learning
4. Project on Regression using MULTIVARIATE LINEAR REGRESSION (AIR POLLUTION PREDICTION)
5. NEW UPDATES IN INDUSTRY: JULIA VS PYTHON

TOOLS FOR ML: (DAY 1)



PYTHON 3.8

- INSTALLING PYTHON 3.8, SUBLIME TEXT 3, PIP, JUPYTER NOTEBOOK
- Introduction to JUPYTER NOTEBOOK
- WHY PYTHON? WHY NOT C/C++/JAVA? Python for Data Science
- Expressions, Variables and Operations in Python
- PRINT AND INPUT IN Python LOOPS AND IF-ELSE-ELIF In Python
- Functions in Python
- Libs/ Dependencies in Python with PIP
- DATA Types in Python Lists, Tuples and Dictionaries, SETS NOT REQUIRED☺
- Array In Python Numpy.ndarray VS LISTS
- Want to Learn More: https://www.youtube.com/channel/UCvObX-nmbN3zQzBVMFWgfdw

FIREBLAZE AI SCHOOL

Data Scientist

Also known as Data Manager, Statistician

• Tools that need to be mastered •

Python

R Programming

SQL

• Skills that need to be mastered •

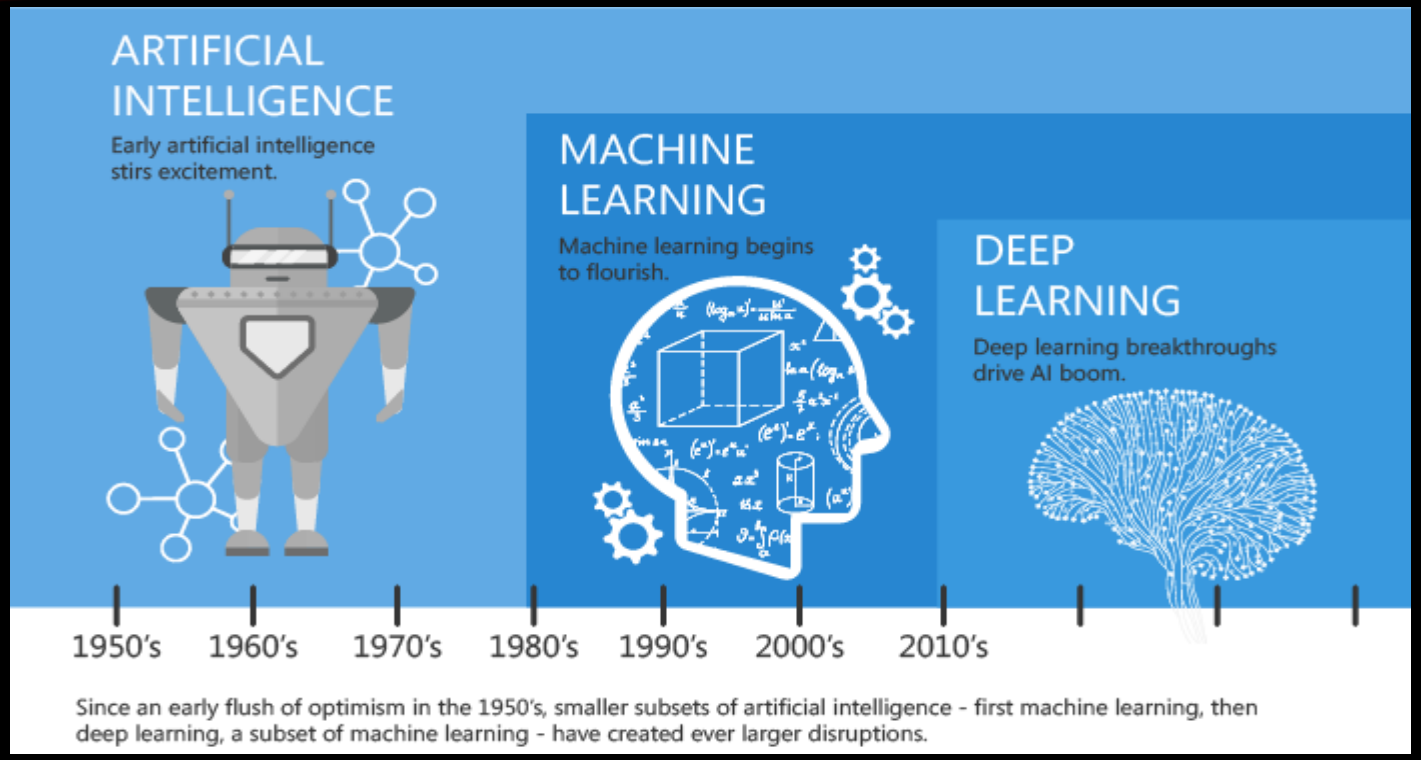
Programming

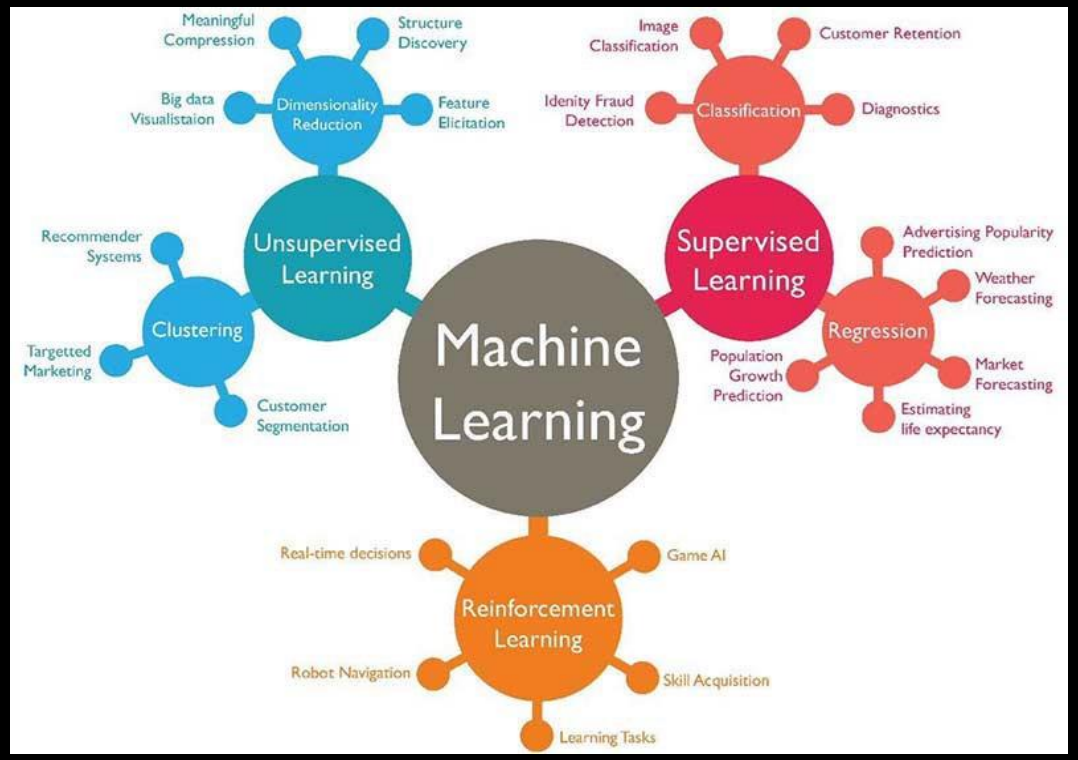
Statistics

Machine Learning

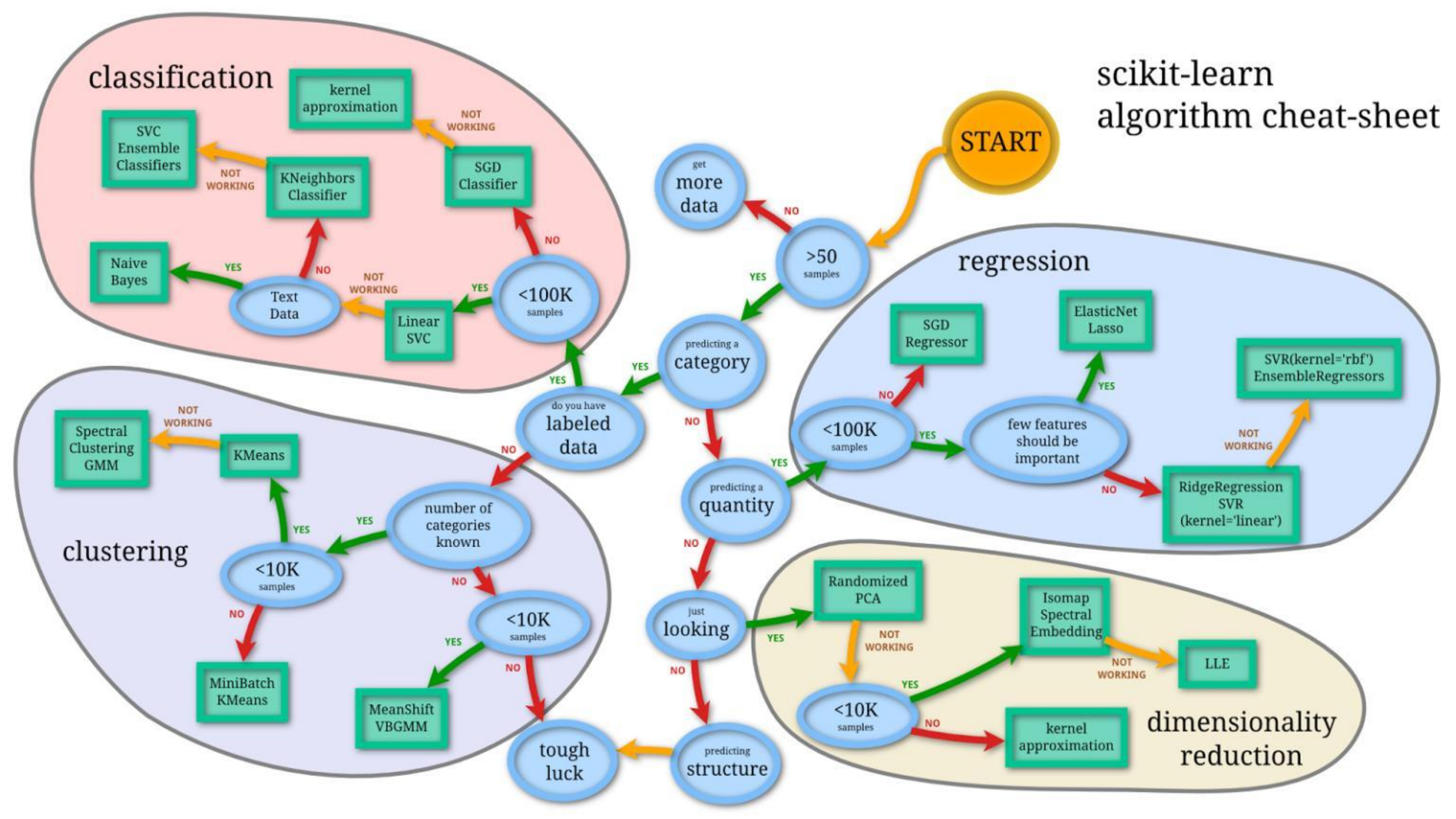
Data Visualization

INTRODUCTION TO ML





- What is DATA, Types of DATA based on Labelling
- AI vs ML: ML and IoT & Robotics
- What is ML? Focus on SUPERVISED LEARNING: SIMPLE OPERATIONS VS ML: INFERENCE MODEL
- What is NLP(Cosine-distances), Discussion of Use cases of NLP: Summary generator, Rap and music generator etc.
- Introduction to Regression model in Details for Project :Linear vs Polynomial


scikit-learn
algorithm cheat-sheet
START
>50 samples
NO
get more data
YES
predicting a category
YES
do you have labeled data
YES
<100K samples
NO
Linear SVC
NOT WORKING
Text Data
YES
Naive Bayes
NO
KNeighbors Classifier
NOT WORKING
SVC Ensemble Classifiers
NOT WORKING
kernel approximation
NOT WORKING
SGD Classifier
NO
classification
NO
predicting a quantity
YES
<100K samples
NO
SGD Regressor
YES
few features should be important
YES
ElasticNet Lasso
NO
RidgeRegression SVR (kernel='linear')
NOT WORKING
SVR(kernel='rbf') EnsembleRegressors
regression
NO
just looking
YES
Randomized PCA
NOT WORKING
<10K samples
YES
Isomap Spectral Embedding
NOT WORKING
LLE
NO
kernel approximation
dimensionality reduction
NO
predicting structure
tough luck
NO
number of categories known
YES
<10K samples
NO
MiniBatch KMeans
YES
<10K samples
NO
MeanShift VBGMM
YES
KMeans
NOT WORKING
Spectral Clustering GMM
clustering

MULTIVARIATE LINEAR REGRESSION

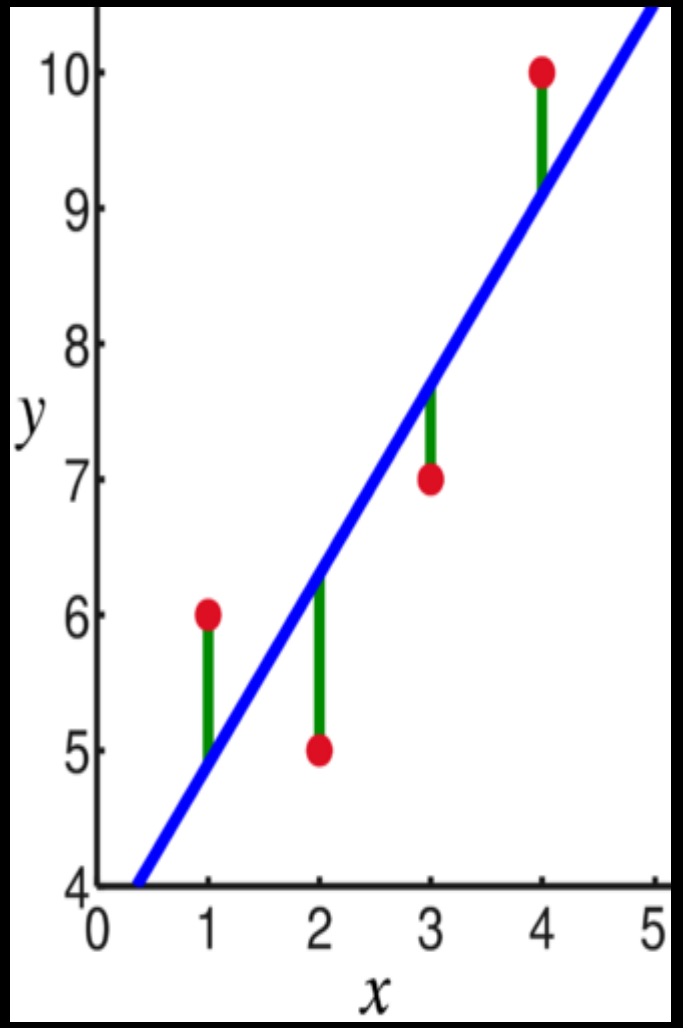
$$\mathbf{y} = X\boldsymbol{\beta} + \boldsymbol{\varepsilon},$$

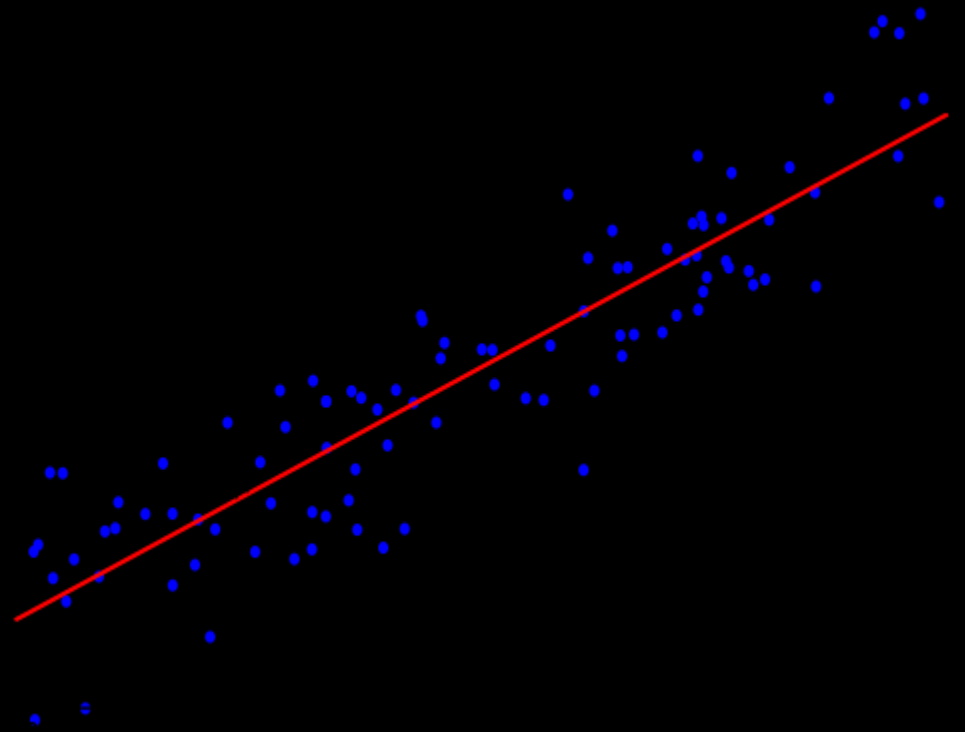
where

$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix},$$

$$X = \begin{pmatrix} \mathbf{x}_1^{\mathsf{T}} \\ \mathbf{x}_2^{\mathsf{T}} \\ \vdots \\ \mathbf{x}_n^{\mathsf{T}} \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix},$$

$$\boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}, \quad \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}.$$





- Stock Price Prediction
- Housing Prices Prediction
- Cricket Score Prediction
- Traffic Volume Prediction
- Tourist Gathering Prediction
- Air Pollution Level Prediction

METHOD OF LEAST SQUARE OPTIMIZATION AND GRADIENT DESCENT & HYPER PARAMETERS

Least-squares estimation and related techniques [edit]

Main article: Linear least squares

Let's assume that the independent variable is $\vec{x} = [x_1, x_2, \ldots, x_m]$ and the model's parameters are $\vec{\beta} = [\beta_0, \beta_1, \cdots, \beta_m]$, then the model's prediction would be $y \approx \beta_0 + \sum_{i=1}^{m} \beta_i \times x_i$. If $\vec{x}$ is extended to $\vec{x} = [1, x_1, x_2, \ldots, x_m]$ then y would become a dot product of the parameter and the independent variable, i.e. $y \approx \sum_{i=0}^{m} \beta_i \times x_i = \vec{\beta} \ . \ \vec{x}$. In the least-squares setting, the optimum parameter is defined as such that minimizes the sum of mean squared loss:

$$\vec{\hat{\beta}} = \arg\min_{\vec{\beta}} L(D, \vec{\beta}) = \arg\min_{\vec{\beta}} \sum_{i=1}^{n} (\vec{\beta} \ . \ \vec{x_i} - y_i)^2$$

Now putting the independent and dependent variables in matrices X and Y respectively, the loss function can be rewritten as:

$$\begin{aligned} L(D, \vec{\beta}) &= ||X\vec{\beta} - Y||^2 \\ &= (X\vec{\beta} - Y)^T (X\vec{\beta} - Y) \\ &= Y^T Y - Y^T X\vec{\beta} - \vec{\beta}^T X^T Y + \vec{\beta}^T X^T X\vec{\beta} \end{aligned}$$

As the loss is convex the optimum solution lies at gradient zero. The gradient of the loss function is (using Denominator layout convention):

$$\begin{aligned} \frac{\partial L(D, \vec{\beta})}{\partial \vec{\beta}} &= \frac{\partial \left(Y^T Y - Y^T X\vec{\beta} - \vec{\beta}^T X^T Y + \vec{\beta}^T X^T X\vec{\beta} \right)}{\partial \vec{\beta}} \\ &= -2Y^T X + 2\vec{\beta}^T X^T X \end{aligned}$$

Setting the gradient to zero produces the optimum parameter:

$$\begin{aligned} -2Y^T X + 2\vec{\beta}^T X^T X = 0 \\ \Rightarrow Y^T X = \vec{\beta}^T X^T X \\ \Rightarrow X^T Y = X^T X\vec{\beta} \\ \Rightarrow \vec{\hat{\beta}} = (X^T X)^{-1} X^T Y \end{aligned}$$

Learning rate

The size of these steps is called the *learning rate*. With a high learning rate we can cover more ground each step, but we risk overshooting the lowest point since the slope of the hill is constantly changing. With a very low learning rate, we can confidently move in the direction of the negative gradient since we are recalculating it so frequently. A low learning rate is more precise, but calculating the gradient is time-consuming, so it will take us a very long time to get to the bottom.

Cost function

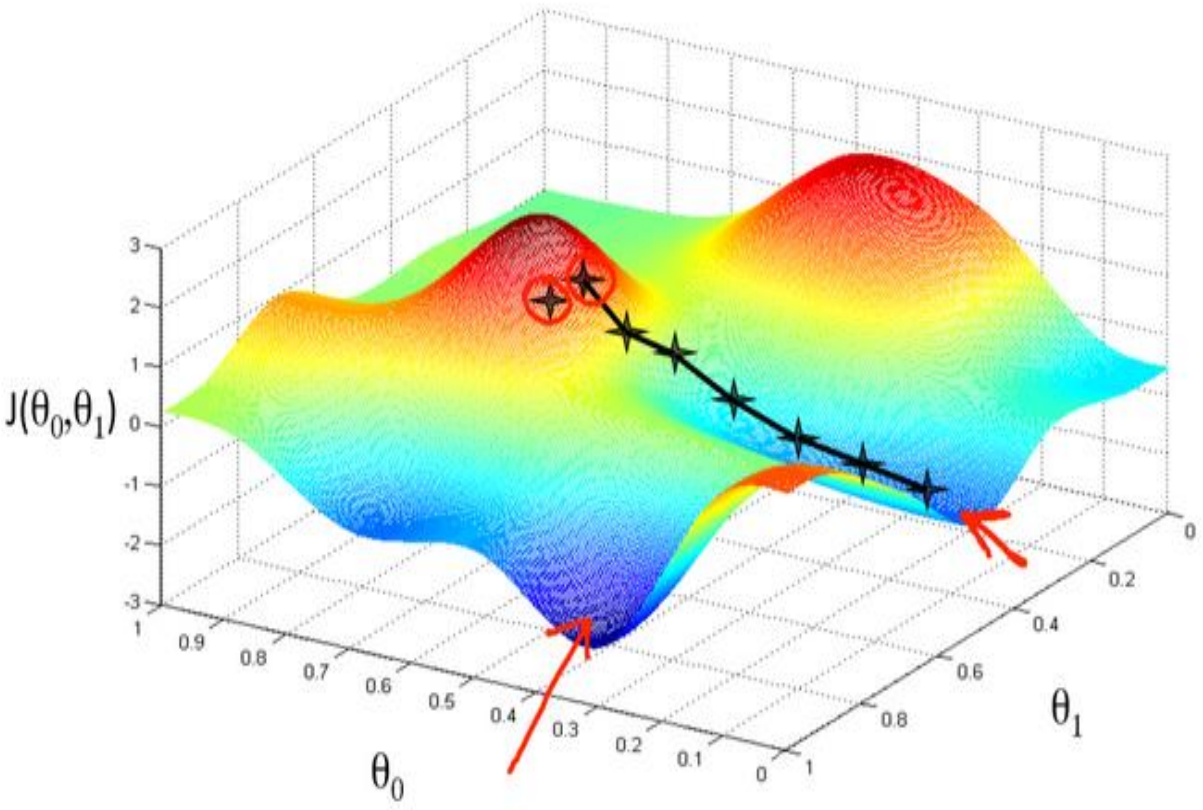
A Loss Functions tells us "how good" our model is at making predictions for a given set of parameters. The cost function has its own curve and its own gradients. The slope of this curve tells us how to update our parameters to make the model more accurate.

Given the cost function:

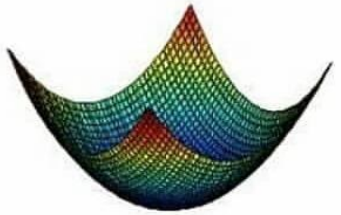
$$f(m, b) = \frac{1}{N} \sum_{i=1}^{n} (y_i - (mx_i + b))^2$$

The gradient can be calculated as:

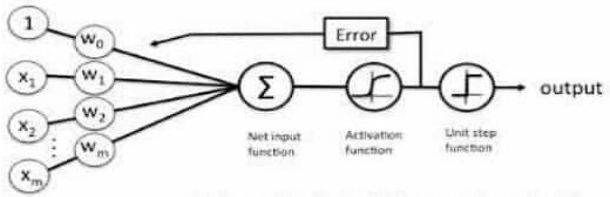
$$f'(m, b) = \begin{bmatrix} \frac{df}{dm} \\ \frac{df}{db} \end{bmatrix} = \begin{bmatrix} \frac{1}{N} \sum -2x_i(y_i - (mx_i + b)) \\ \frac{1}{N} \sum -2(y_i - (mx_i + b)) \end{bmatrix}$$



You

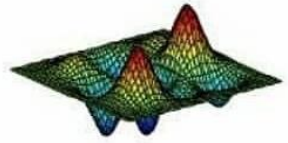


- Unique optimum: global/local.



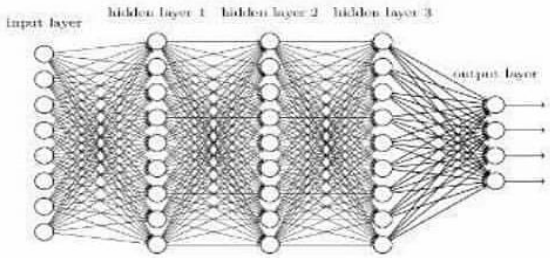
Schematic of a logistic regression classifier.

The guy she tells you not to worry about



- Multiple local optima
- In high dimensions possibly

Deep neural network



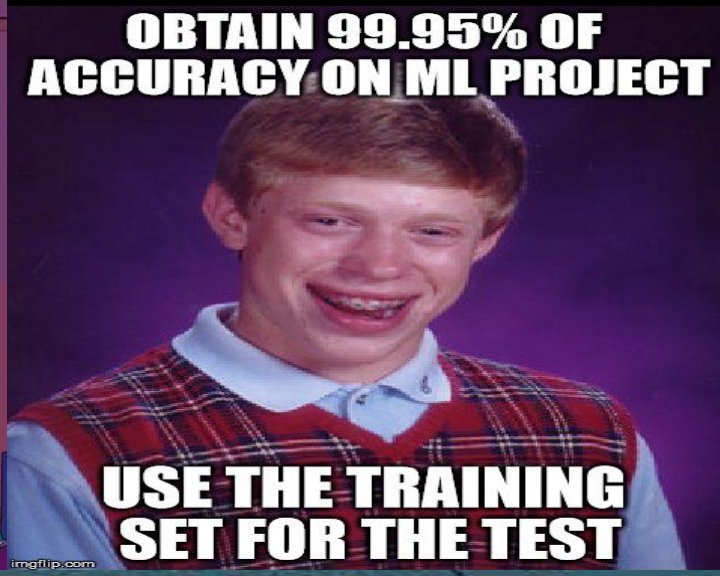
Via: Machine Learning India - @ml.india

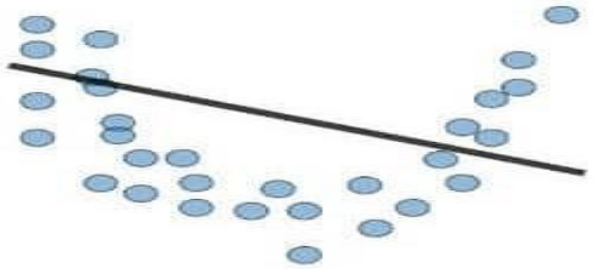
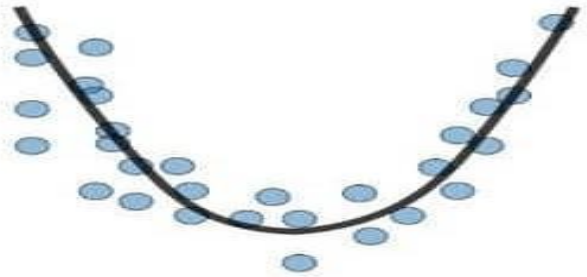
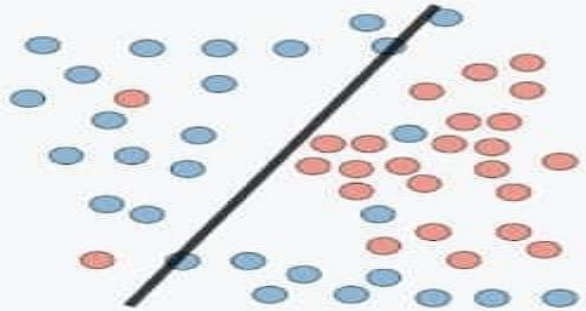
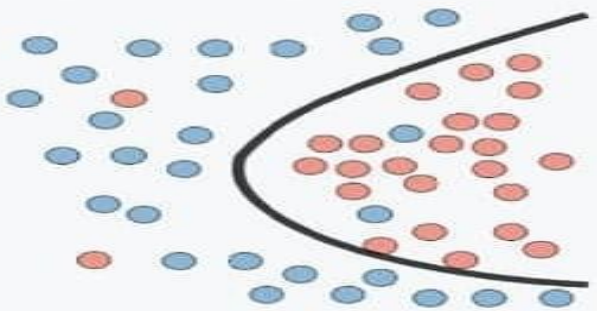
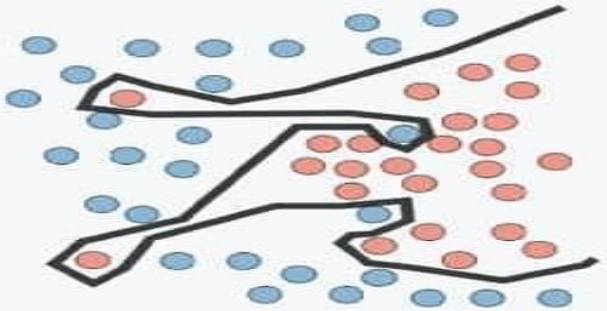
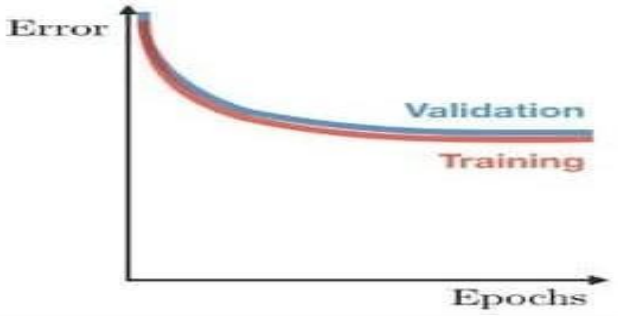
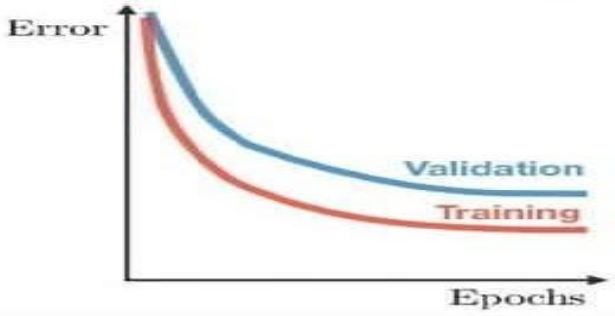
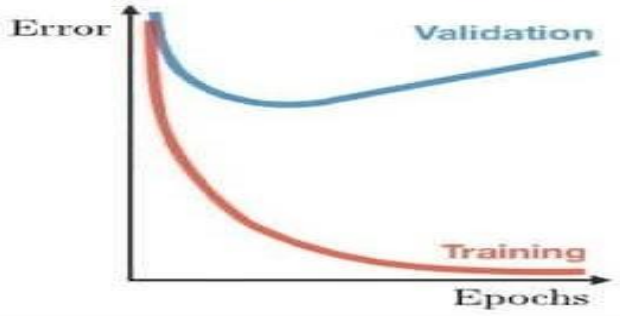
ACCURACY CHECKING

Confusion matrix — The confusion matrix is used to have a more complete picture when assessing the performance of a model. It is defined as follows:

		Predicted class +	**Predicted** class -
Actual class	+	**TP** True Positives	**FN** False Negatives Type II error
	-	**FP** False Positives Type I error	**TN** True Negatives

Metric	Formula	Interpretation
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	Overall performance of model
Precision	$\frac{TP}{TP + FP}$	How accurate the positive predictions are
Recall Sensitivity	$\frac{TP}{TP + FN}$	Coverage of actual positive sample
Specificity	$\frac{TN}{TN + FP}$	Coverage of actual negative sample
F1 score	$\frac{2TP}{2TP + FP + FN}$	Hybrid metric useful for unbalanced classes



	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	• Complexify model • Add more features • Train longer		• Perform regularization • Get more data

AIR POLLUTION DATASET: PROJECT

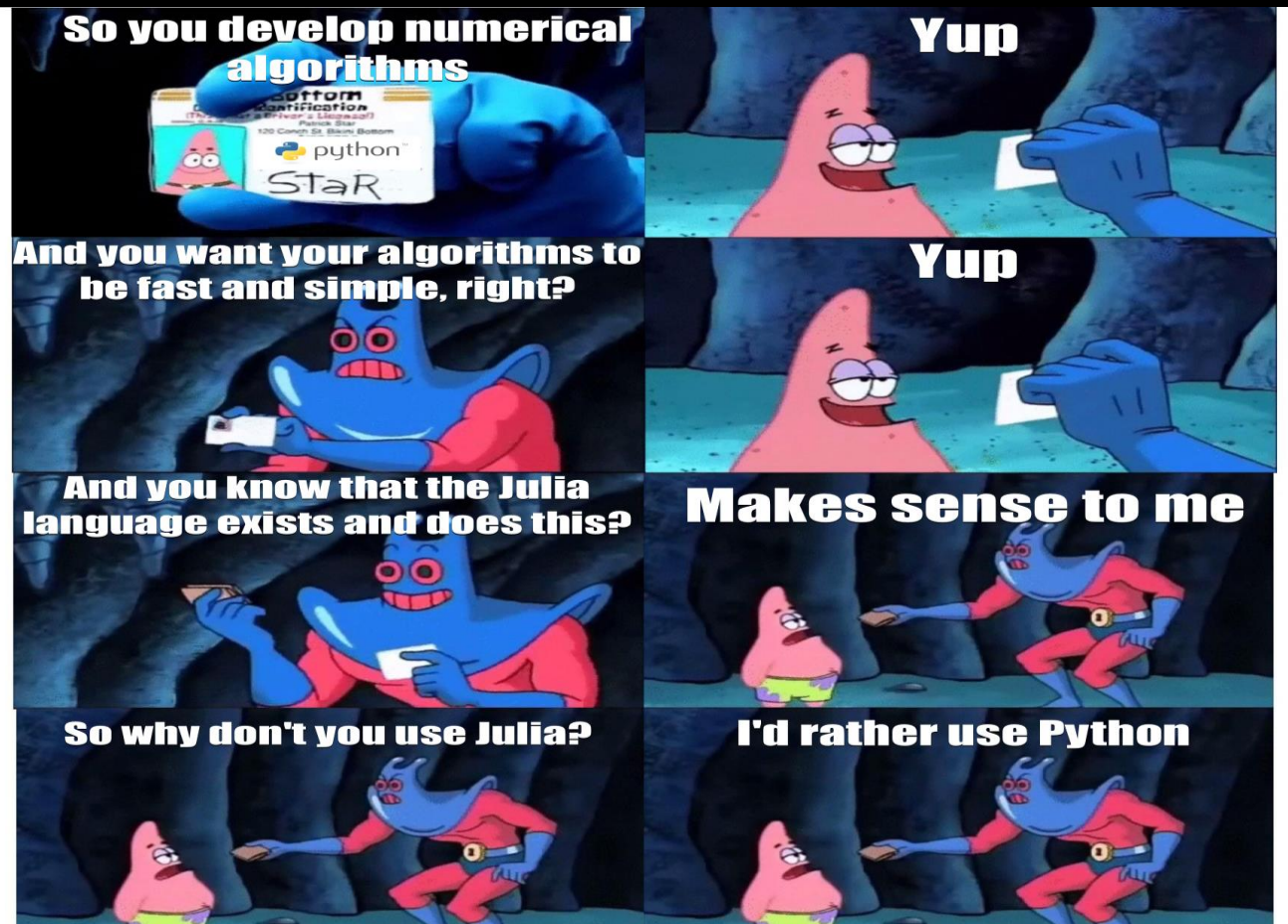
- 5-Features :1 Label
- Regression Model to Predict Air Pollution in City with Given params.
- CHOICE OF MODEL: MVR with Sklearn (Multivariate Regression)
- Visualisation suggests that Data is Linear in terms of Labels
- AIM: TO PREDICT MOST OPTIMIZED EQUATION IN FORM $y = C + \sum_{i=1}^{5} \beta_i * x_i$
- RESULTS: VALIDATION ACC. >90%
- GET GIT BASH HERE: https://gitforwindows.org/
- GIT Clone the repo: https://github.com/Ankuraxz/ML-DL-OPENCV_IOSD_WORKSHOP.git

  Follow These Steps in ~rf:

  1. $mkdir iosd_workshop
  2. $ git clone https://github.com/Ankuraxz/ML-DL-OPENCV_IOSD_WORKSHOP.git in local Machine
  3. Upload the cd to Google Drive for Collab or use Jupyter Notebook
  4. In collab: %cd contents/Mydrive
  5. Change %cd at ./iosd_workshop/AIR POLLUTION PROJECT
  6. Else !git clone <repo_name> in collab : 1 time use only: follow steps 4,5 as it is

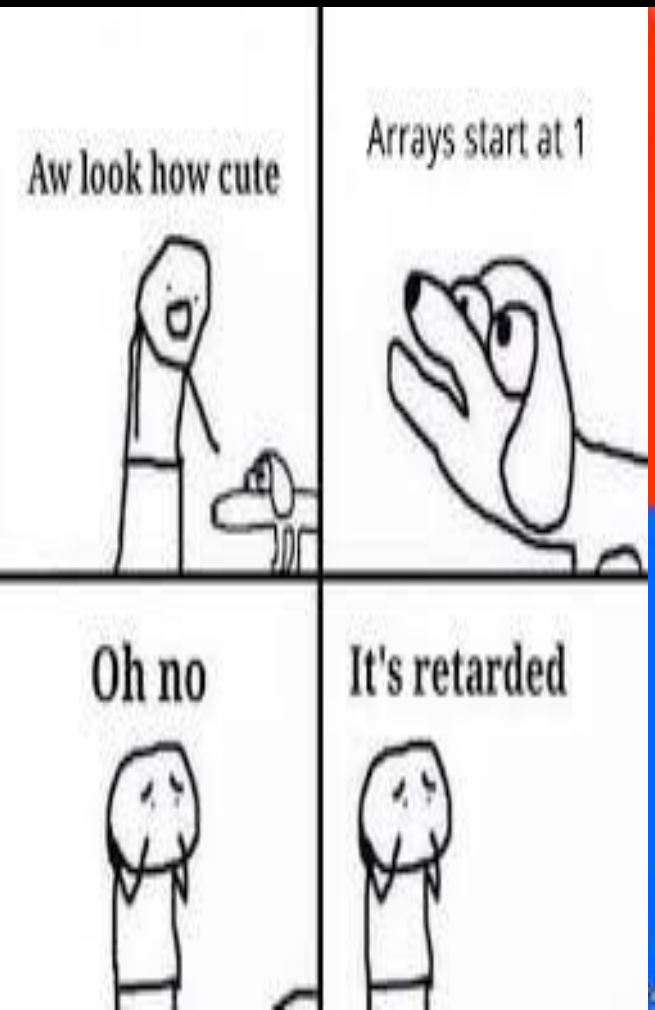
UPDATES: JULIA VS PYTHON

- https://julialang.org/



- Julia, Created in 2012 in MIT (VIRAL SHAH A co-creator) is from India
- Designed for HIGH PERFORMANCE
- It is as easy to learn as Python with all similar feature like high level syntax, Code Completion etc.
- It is Open Source and Multi Paradigm with Debugging facilities
- It is Complied like C/C++ and is much faster than Python/R/Matlab. Although Complied, It still has Interpreted lang features
- It Combines the good of Python, R, Matlab, Lisp, Swift, C/C++ to form a language which is Perfect for any Application From Data Science to App Development, AI to Competitive coding.
- It fells like Scripting Lang and is Dynamic with Optionally typing facilities

A FEW PROBLEMS(FEATURES:}) OF JULIA



- JULIA Does Not has much Community/ Libs Support.
- Julia has Indexing from 1 rather than 0, Thanks to MATLAB
- Less Popular than python for at least next 5-10 years.

ANKUR {STUDENT MENTOR AT IOSD UIET KUK}

https://www.linkedin.com/in/ankur-verma-0bb747154

https://github.com/Ankuraxz

ankurvermaaxz@gmail.com


DEATH BY POWERPOINT
memecenter.com
MemeCenter


DON'T PPT THEM TO DEATH!
MAKE IT INTERESTING WITH MEMES

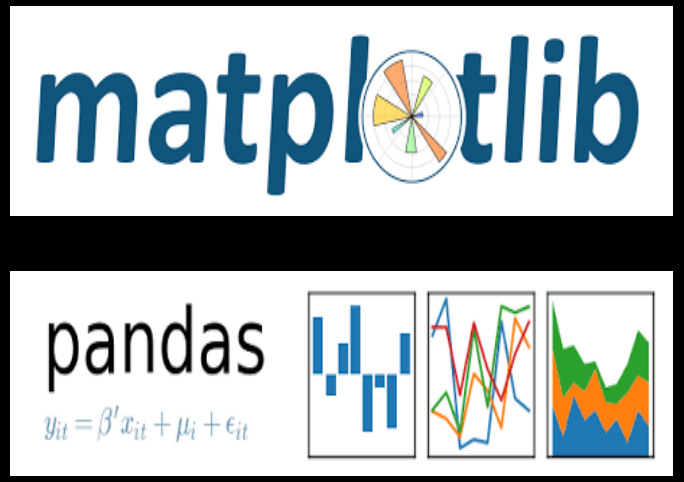

00101
11001
DEEP
LEARNING
WWW

DAY2- IOSD DEVELOPMENT WEEK

AGENDA:

1. Introduction to Deep Learning
2. Working of Neural Networks, Multi layer Perceptrons
3. Introduction to Tensorflow 2.0
4. Project on Image Classification using Dense Neural Network(Hand written digit predictor)
5. Discussion of Deeper Networks (OPTIONAL)
6. Introduction to Transfer learning and GAN's(OPTIONAL)
7. Creating a Custom network in Tensorflow from Scratch/ Transfer Learning(OPTIONAL)
8. NEW UPDATES IN INDUSTRY: ML/DL on Rpi, ASIC's-FPGA's and GPU's (tf-lite)

TOOLS FOR DL: (DAY 2)



Artificial Intelligence

Any technique that enables computers to mimic human intelligence. It includes *machine learning*

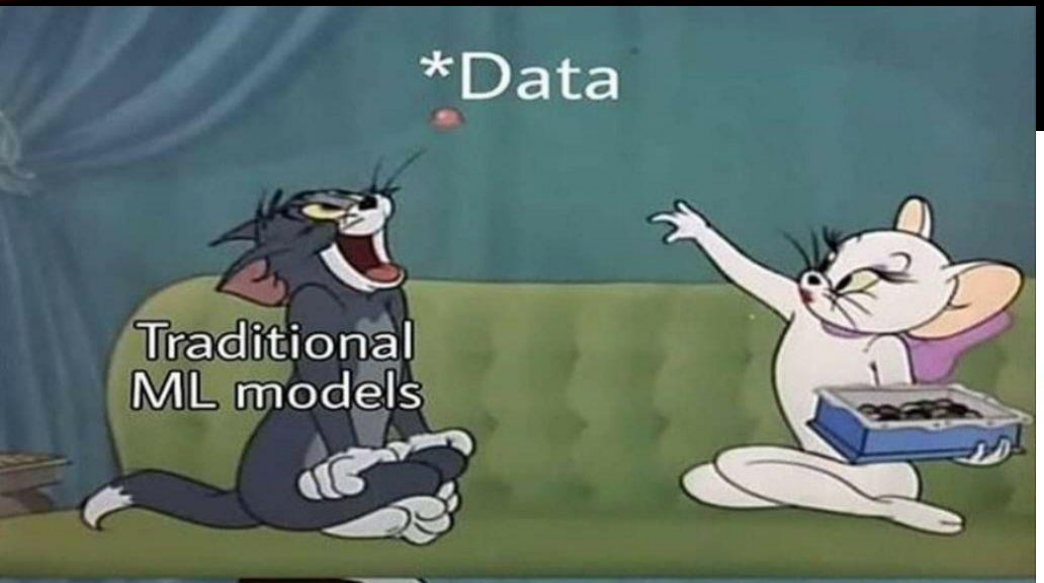
Machine Learning

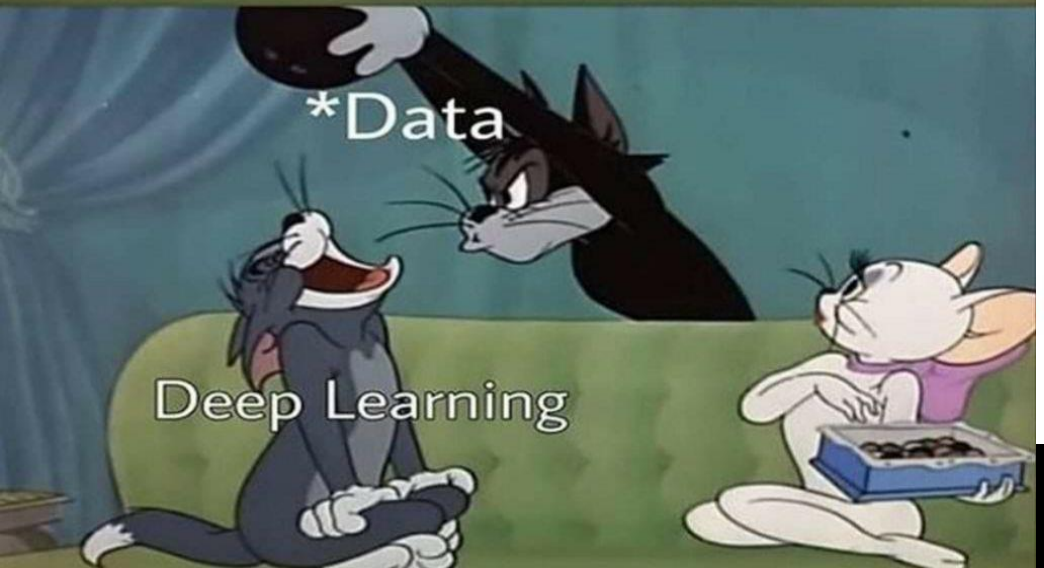
A subset of AI that includes techniques that enable machines to improve at tasks with experience. It includes *deep learning*

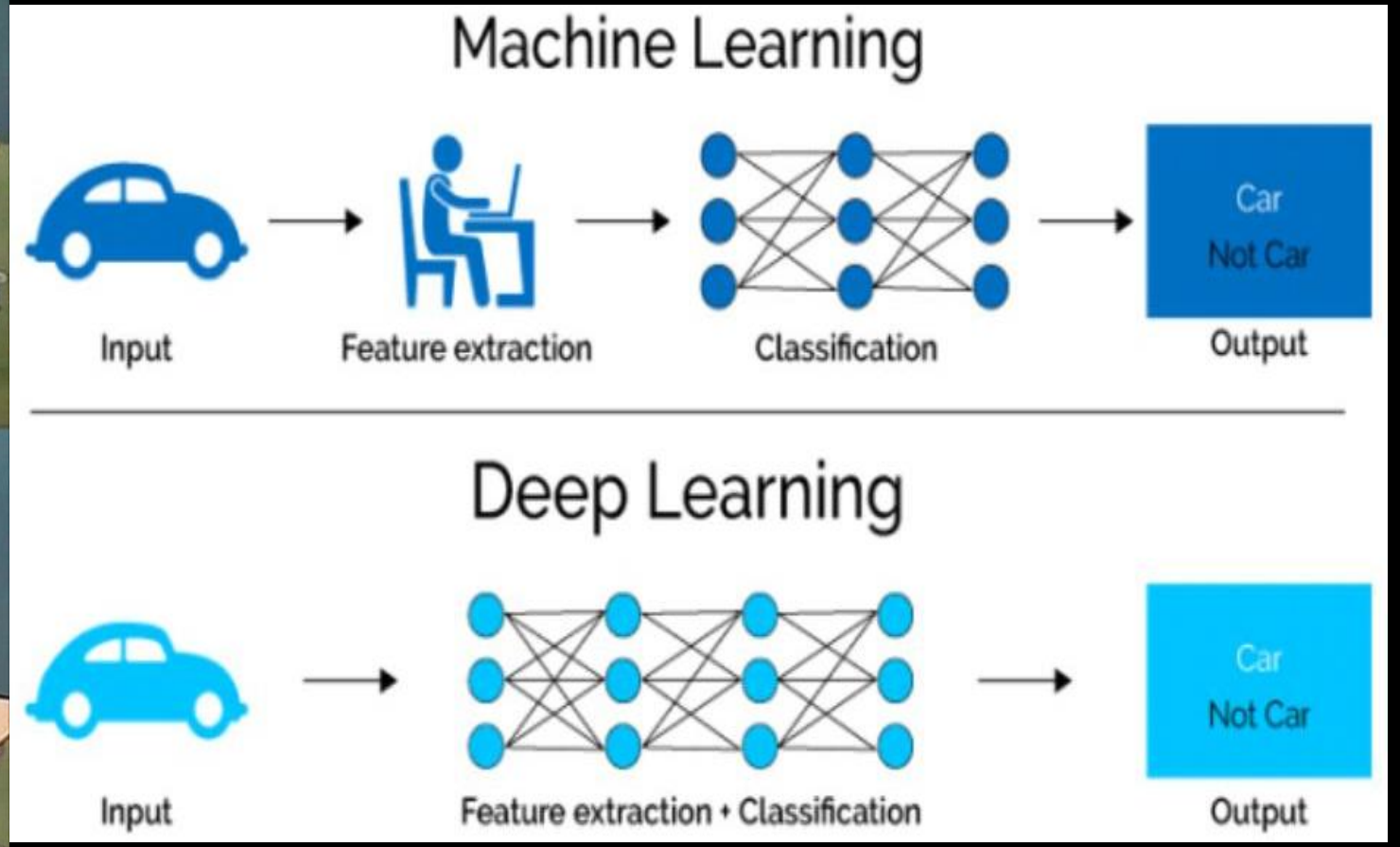
Deep Learning

A subset of machine learning based on neural networks that permit a machine to train itself to perform a task.

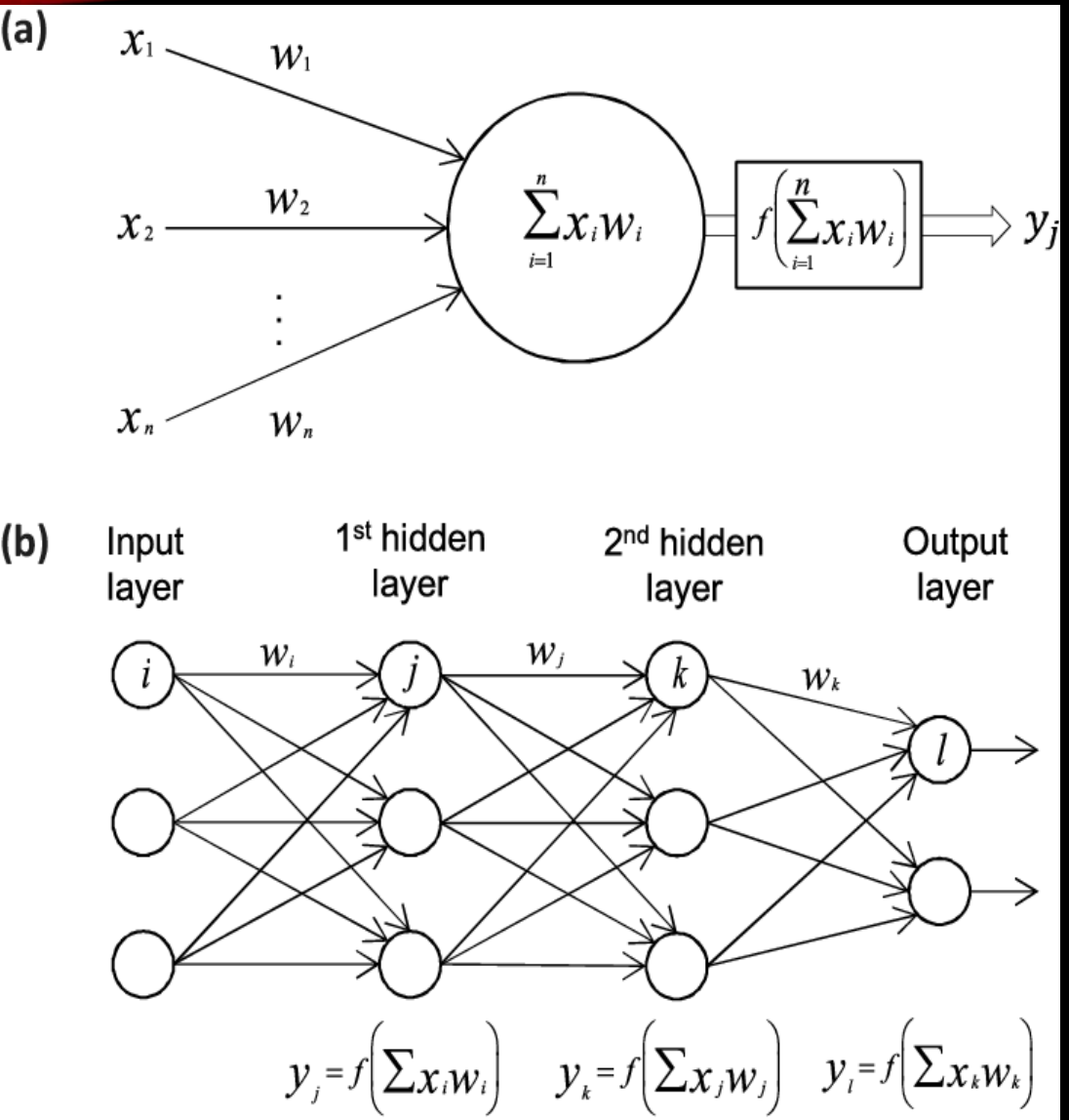
DL VS ML


*Data
Traditional
ML models


*Data
Deep Learning

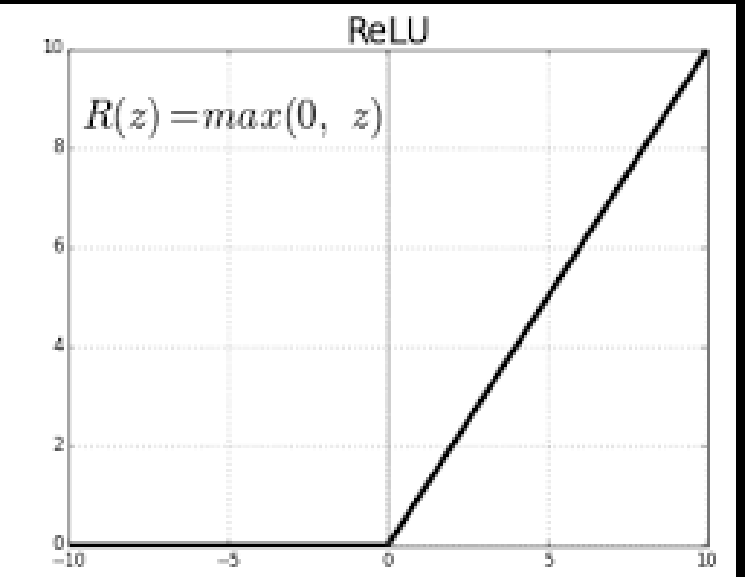

Machine Learning
Input
Feature extraction
Classification
Car
Not Car
Output
Deep Learning
Input
Feature extraction + Classification
Car
Not Car
Output

NEURAL NETWORKS

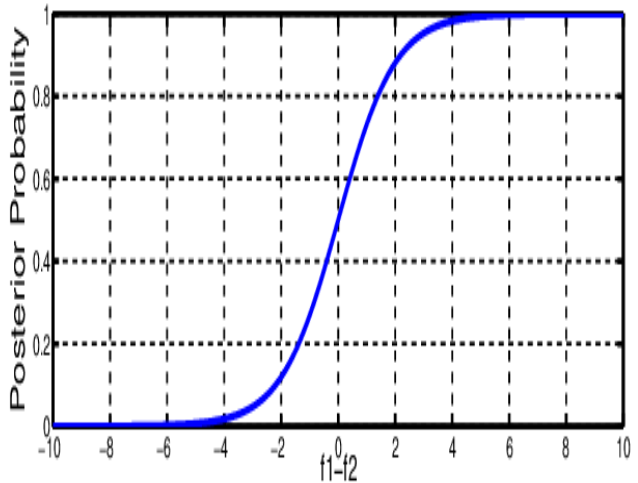


- **Artificial neural networks** (**ANN**) or **connectionist systems** are computing systems vaguely inspired by the biological neural networks that constitute animal brains.
- Such systems "learn" to perform tasks by considering examples, generally without being programmed with task-specific rules. For example, in image recognition, they might learn to identify images that contain cats by analyzing example images that have been manually labeled as "cat" or "no cat" and using the results to identify cats in other images. They do this without any prior knowledge of cats, for example, that they have fur, tails, whiskers and cat-like faces. Instead, they automatically generate identifying characteristics from the examples that they process.
- An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a signal then processes it and can signal neurons connected to it.
- Intro to fwd and back prop.

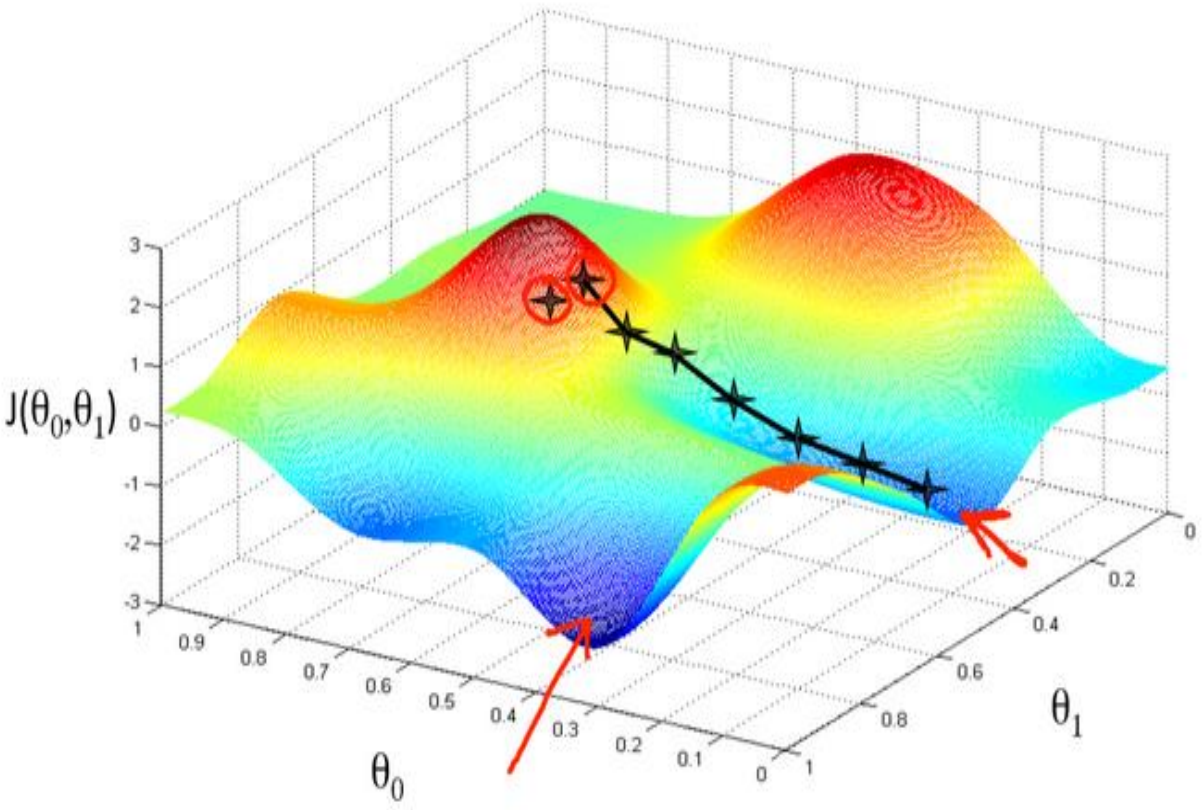
ACTIVATION FUNCTIONS (FOR PROJECT)



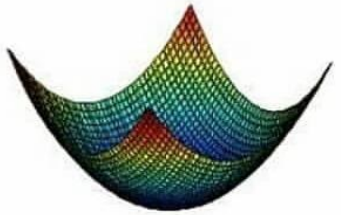
ReLU stands for rectified linear unit, and is a type of **activation function**. Mathematically, it is defined as y = max(0, x). ... **ReLU** is the most commonly used **activation function** in neural networks, especially in CNNs. If you are unsure what **activation function** to use in your network, **ReLU** is usually a good first choice. Advanced form Leaky-Relu



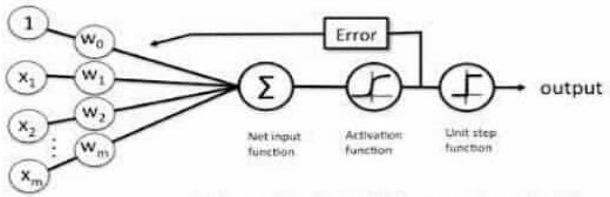
In mathematics, the **Softmax** function, also known as soft-argmax or normalized exponential function, is a function that takes as input a vector of K real numbers, and normalizes it into a probability distribution consisting of K probabilities proportional to the exponentials of the input numbers.



You

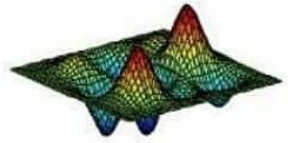


- Unique optimum: global/local.



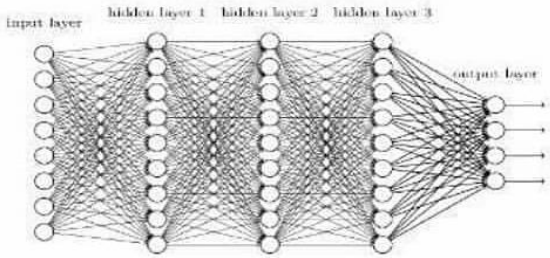
Schematic of a logistic regression classifier.

The guy she tells you not to worry about



- Multiple local optima
- In high dimensions possibly

Deep neural network



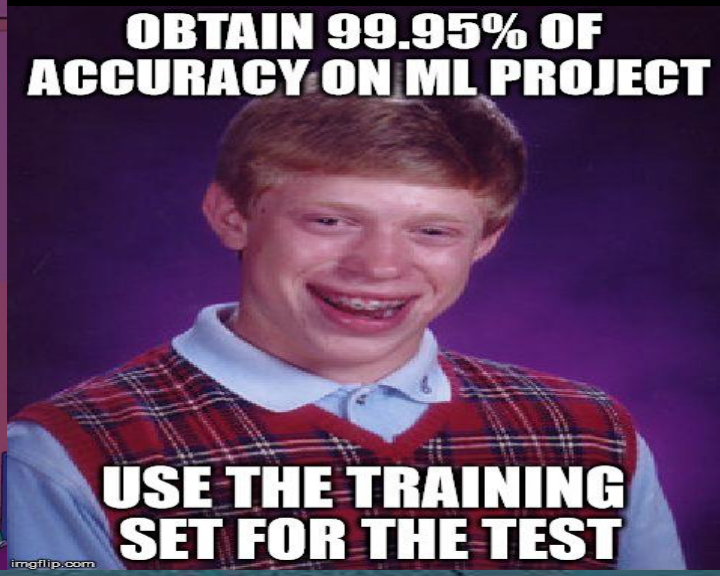
Via: Machine Learning India - @ml.india

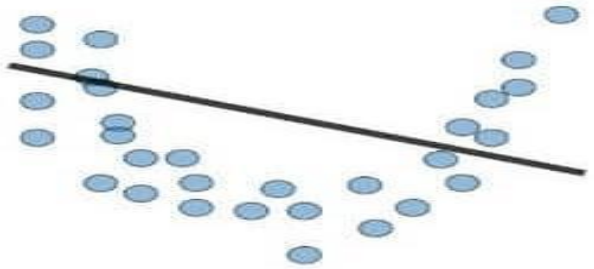
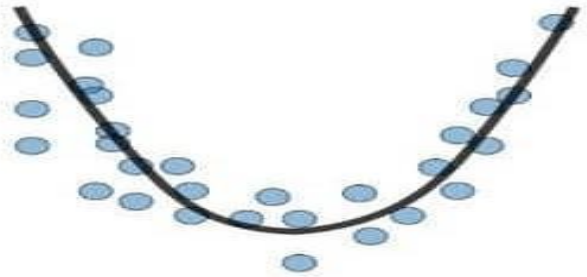
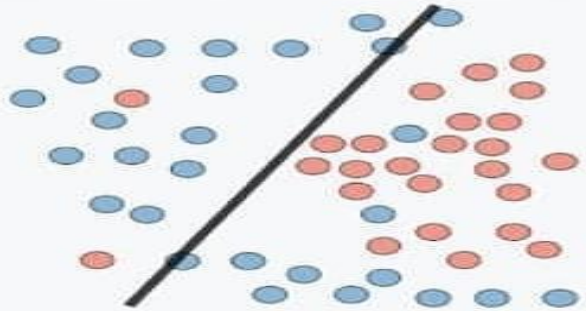
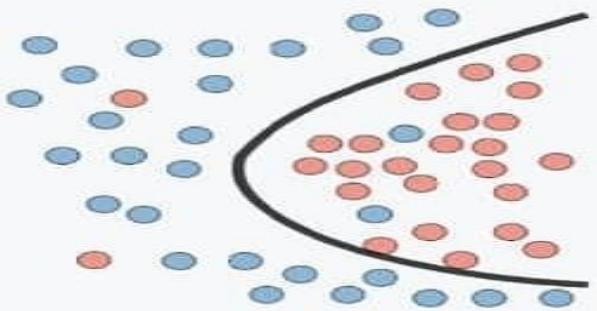
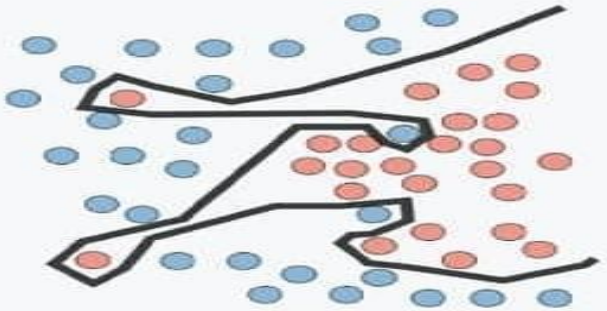
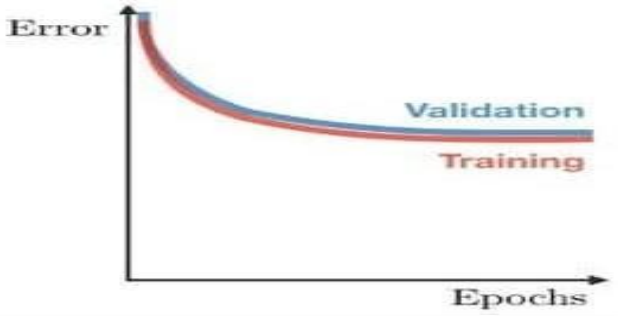
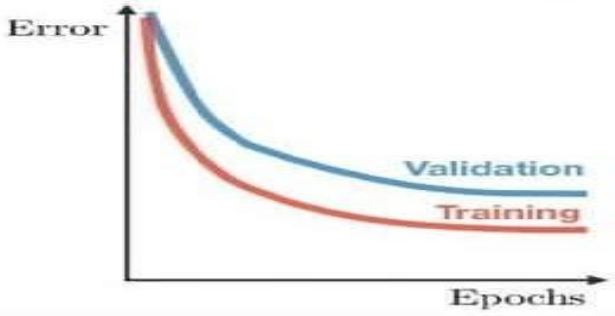
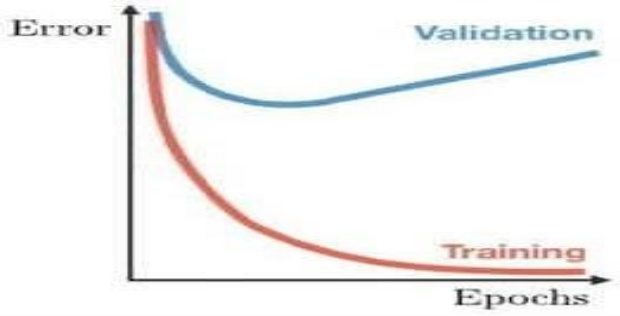
ACCURACY CHECKING

Confusion matrix — The confusion matrix is used to have a more complete picture when assessing the performance of a model. It is defined as follows:

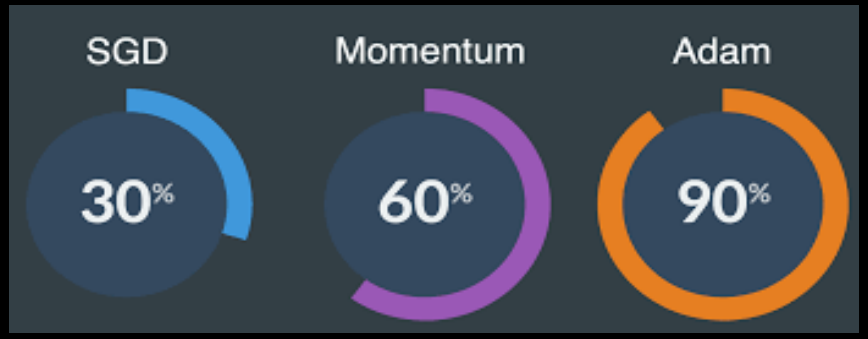
		Predicted class +	**Predicted** class -
Actual class	+	**TP** True Positives	**FN** False Negatives Type II error
	-	**FP** False Positives Type I error	**TN** True Negatives

Metric	Formula	Interpretation
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	Overall performance of model
Precision	$\frac{TP}{TP + FP}$	How accurate the positive predictions are
Recall Sensitivity	$\frac{TP}{TP + FN}$	Coverage of actual positive sample
Specificity	$\frac{TN}{TN + FP}$	Coverage of actual negative sample
F1 score	$\frac{2TP}{2TP + FP + FN}$	Hybrid metric useful for unbalanced classes



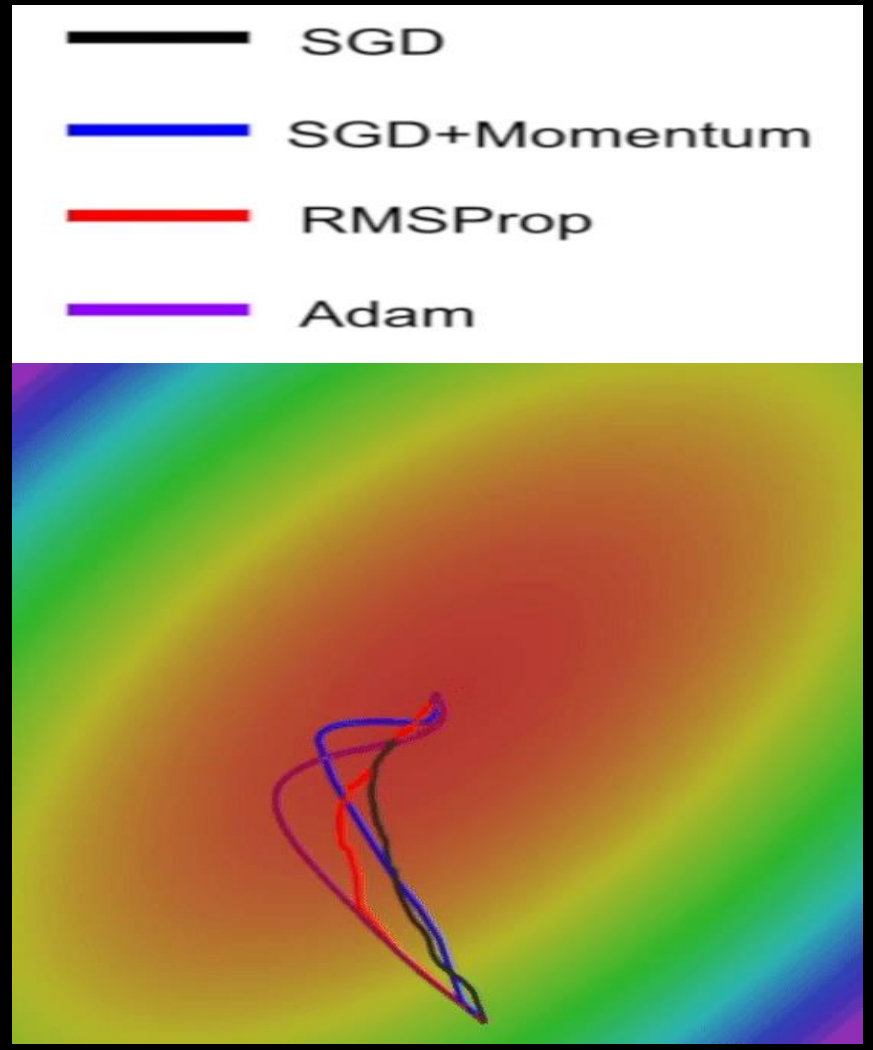
	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	• Complexify model • Add more features • Train longer		• Perform regularization • Get more data

OPTIMIZATION FUNCTION(FOR PROJECT) ~ADAM

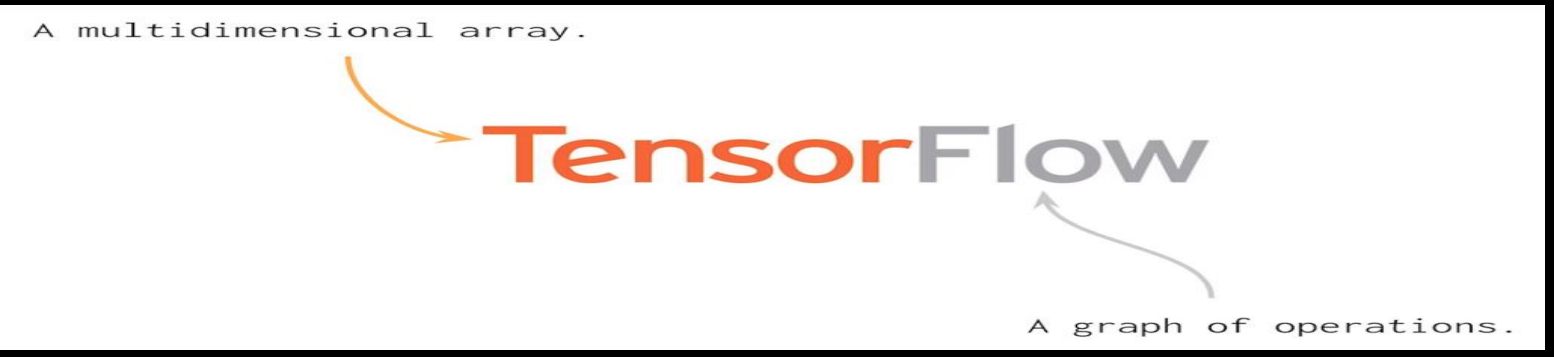


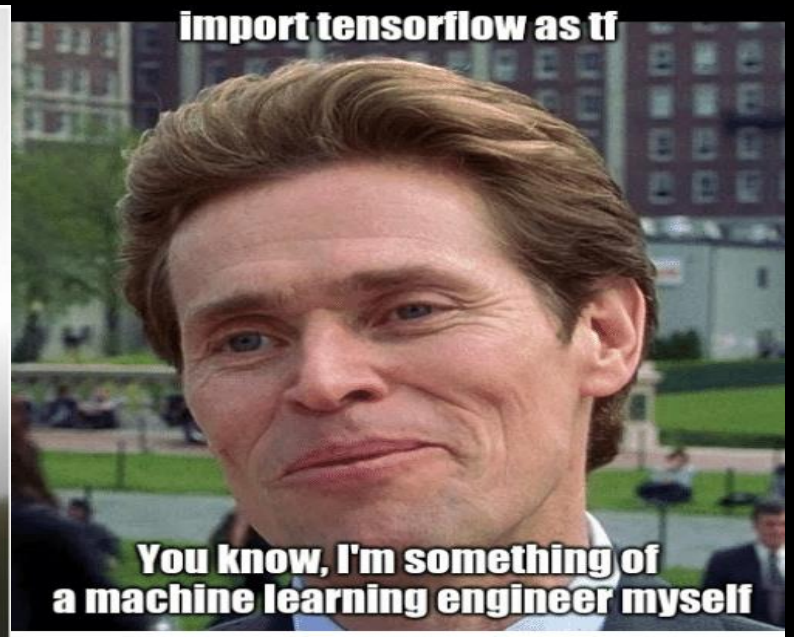
Ref: https://arxiv.org/pdf/1412.6980.pdf
https://github.com/prateek27/machine-learning-aug-18/blob/master/class_14/optimizers.pdf

Adam realizes the benefits of both AdaGrad and RMSProp.
Instead of adapting the parameter learning rates based on the average first moment (the mean) as in RMSProp, Adam also makes use of the average of the second moments of the gradients (the uncentered variance).



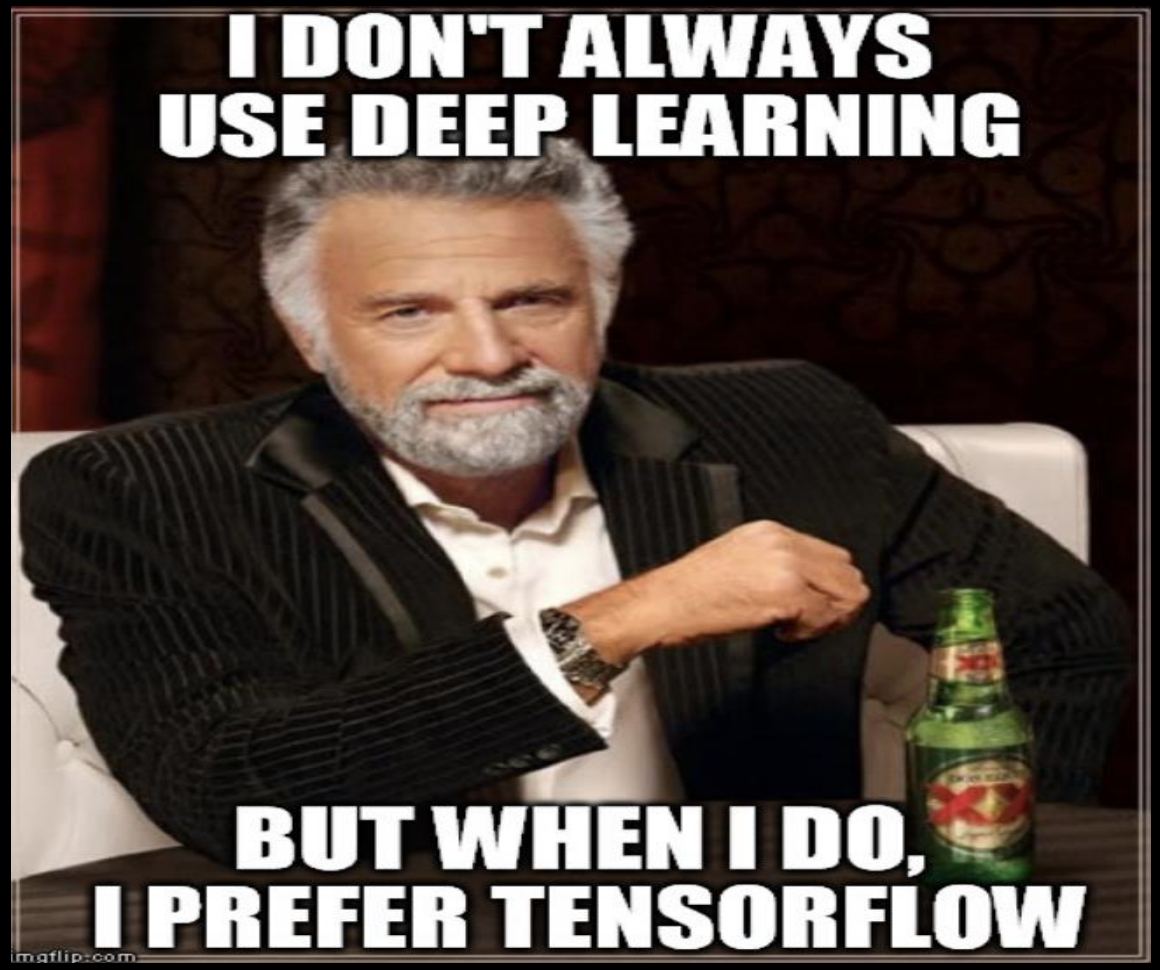
TENSORFLOW 2.0 AND KERAS BACKEND API





- TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.
- **TensorFlow** 2.0 removes redundant APIs, makes APIs more consistent (Unified RNNs, Unified Optimizers), and better integrates with the Python runtime with Eager execution. Many documentations have explained the changes that have gone into making **TensorFlow** 2.0. ... It's assumed you have some familiarity with **TensorFlow** 1.x {NOT NEEDED ACTUALLY}
- https://www.tensorflow.org

Programmers Nowadays



PROJECT MNIST CASSIFICATION

- 10 {number 0-9} classes more than 1000 images per class in .csv format
- Visualisation and Normalisation required as preprocessing
- Aim: To extract features and classify Images as numbers
- Bonus project: Fashion MNIST
- Result: Obtained more than 90% acc. with 3 layered NN using Tf2.0 Framework
- Other use cases of DL:
  1. Classification and regression even when features are absent
  2. Extraction of features and then using SVM/ Regression-Classification techniques
  3. Project like Animal Classifier, Face Detector and Facial Recogniser, GAN's, Music Generator, Word, Sentence-Rap generator, and much more

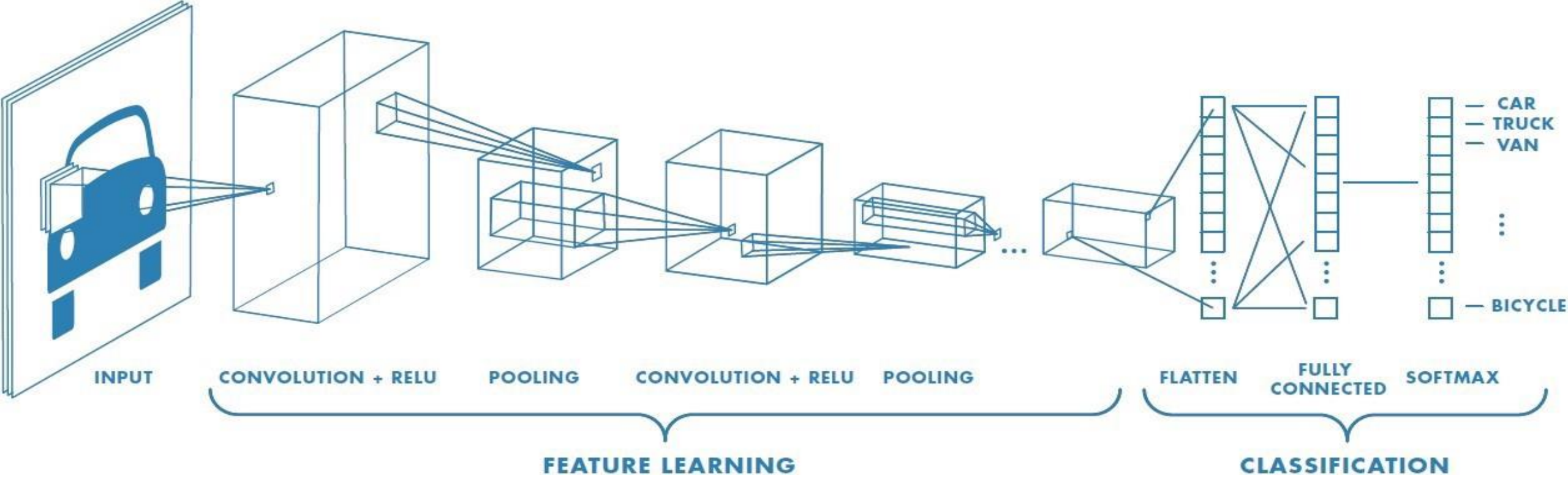

Me after training a neural network
WHEN DID YOU BECOME AN EXPERT IN DEEP LEARNING?
I DIDN'T

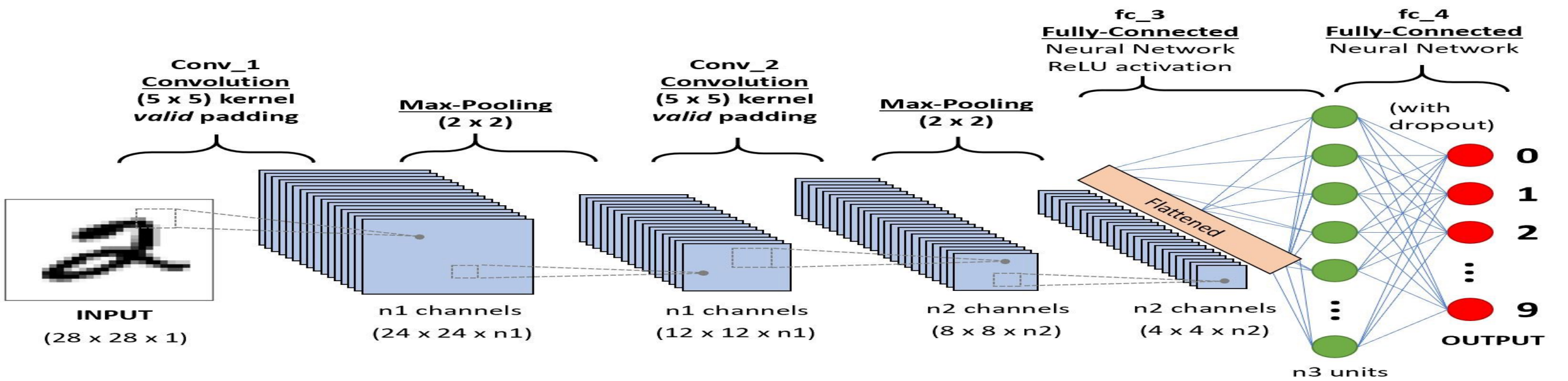

Me
Deep learning
Simple problem

DEEPER NETWORKS

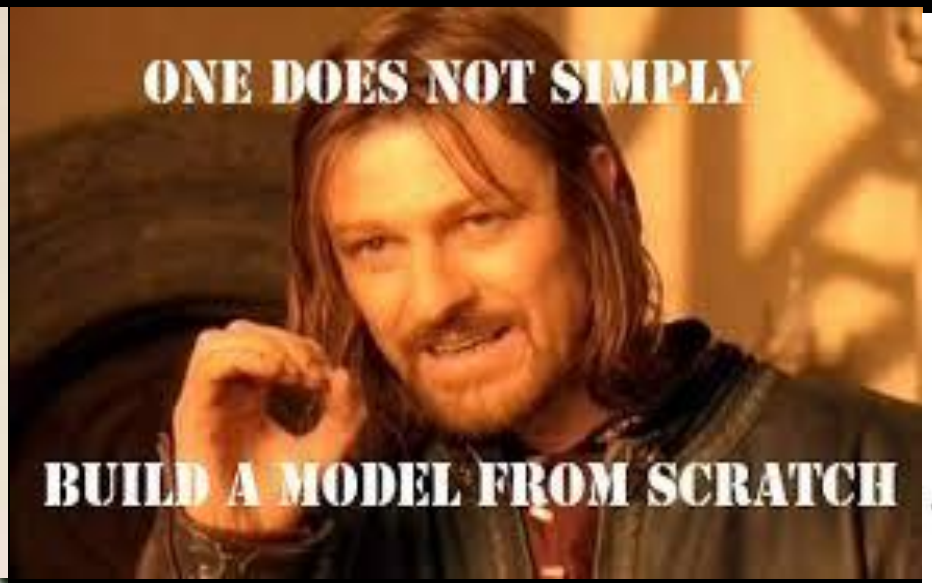
- CNN: Convolutional Neural Networks :Introduction to convents
- convolution-neural-network-tutorial/CNN_Concepts.ipynb
- Other Deeper Networks Via "TRANSFER LEARNING":
  1. VGG
  2. Resnet
  3. InceptionNet/Google Net
  4. InceptionResnet
  5. DenseNet
  6. And many more…

- Nets based on Scratch Construction from Architecture:
  1. Alexnet
  2. Light CNN
  3. VGGface
  4. And many more…

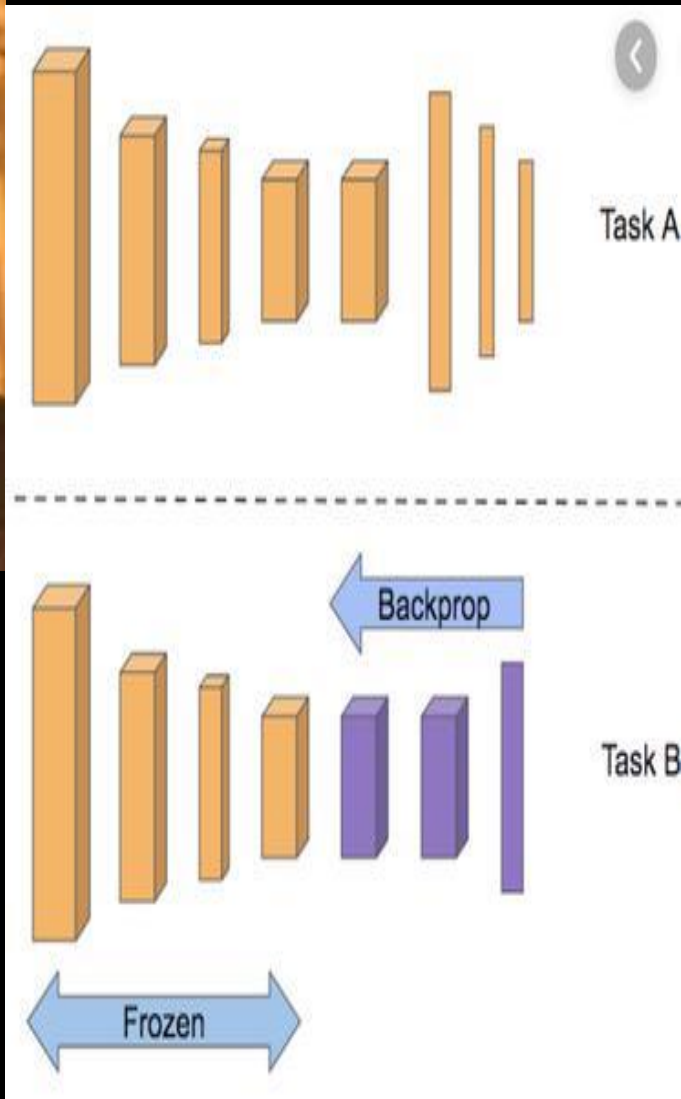

CAR
TRUCK
VAN
BICYCLE
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
FEATURE LEARNING
CLASSIFICATION


Conv_1
Convolution
(5 x 5) kernel
valid padding
Max-Pooling
(2 x 2)
Conv_2
Convolution
(5 x 5) kernel
valid padding
Max-Pooling
(2 x 2)
fc_3
Fully-Connected
Neural Network
ReLU activation
fc_4
Fully-Connected
Neural Network
(with dropout)
Flattened
INPUT
(28 x 28 x 1)
n1 channels
(24 x 24 x n1)
n1 channels
(12 x 12 x n1)
n2 channels
(8 x 8 x n2)
n2 channels
(4 x 4 x n2)
n3 units
0
1
2
9
OUTPUT

INTRODUCTION TO TRANSFER LEARNING

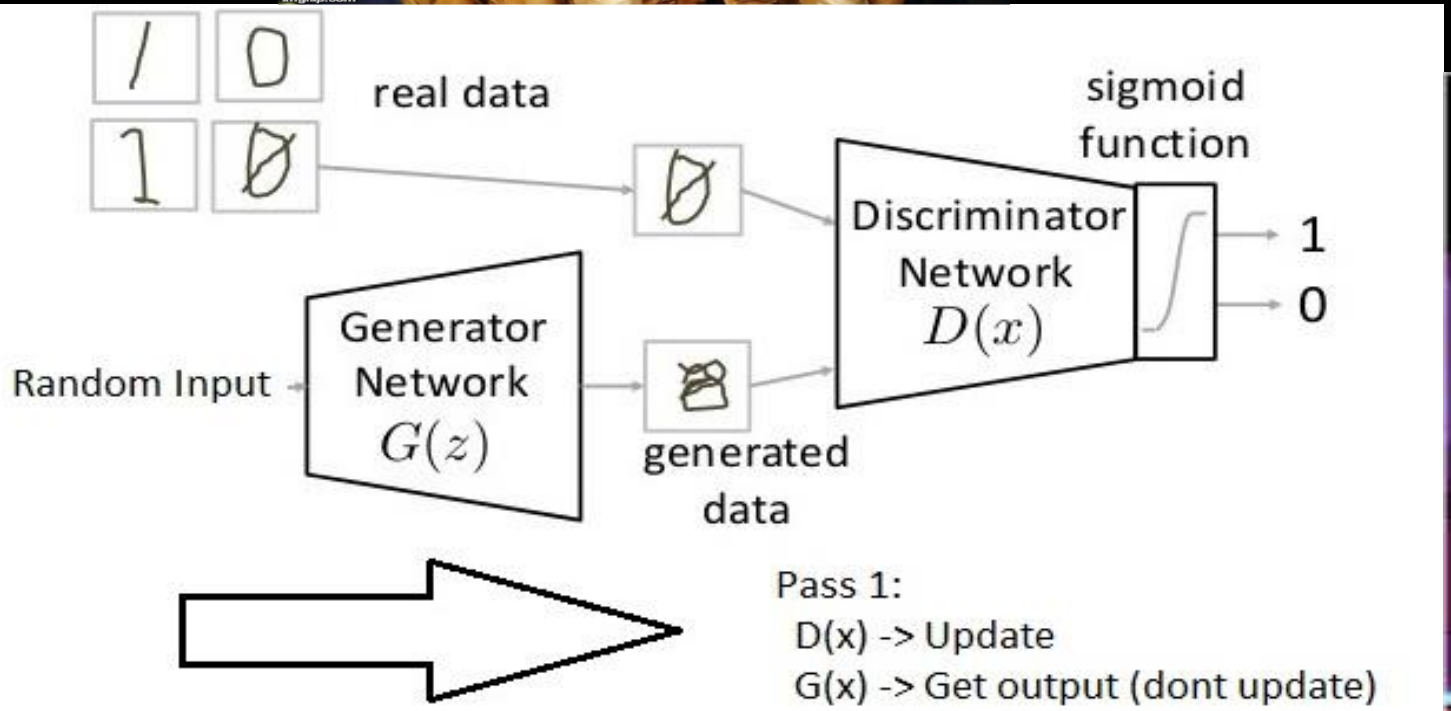


- Transfer Learning helps to collect weights of a pretrained model on a particular dataset eg. RESNET50 on Imagenet100
- It helps to reduce computation and ETA of training, as only last few layers of model have to be trained



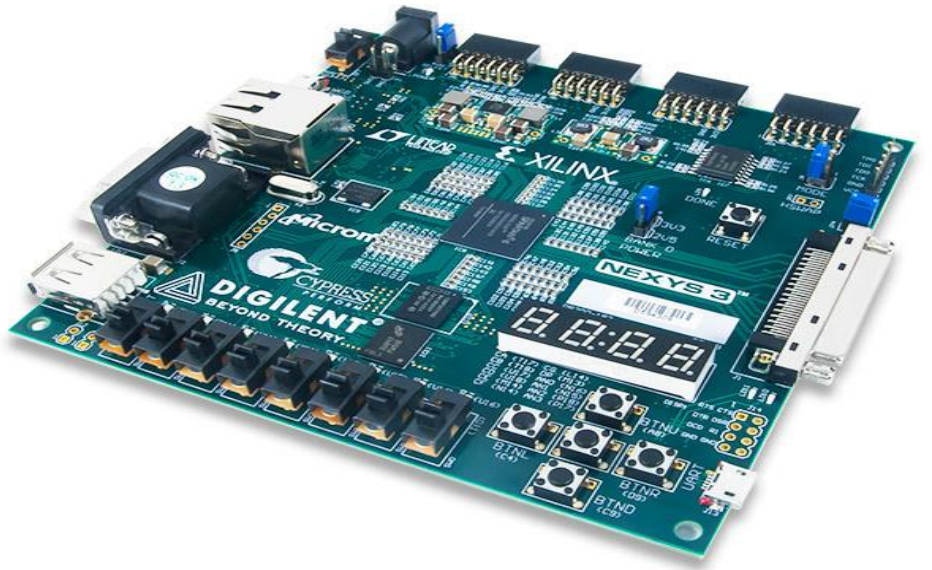
GENERATIVE ADVERSAL NETWORKS (GAN'S)

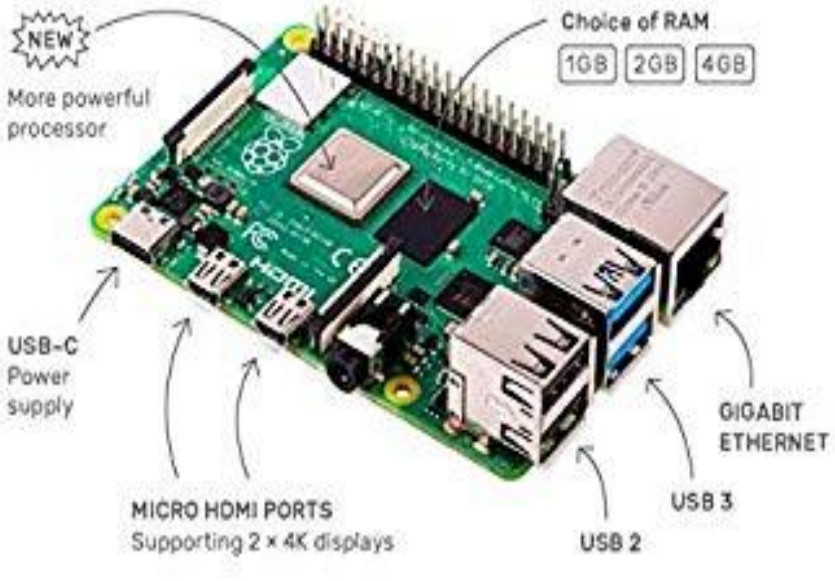
- Thispersondoesnotexist example
- Processing on faces and MNIST data
- Usecases in Data Generation, Data Normalisation using fake data removal etc.
- Dependency on random intialisation



ML/DL ON HARDWARE

- This is where Robotics/IoT and ML-DL connects
- Tensorflow lite is a good option, tflite for Arduino
- Discussion of FPGA/ASIC's/Rpi/ programmable_GPU
- Need of Deployment on Hardware





TfLite Written in C++11 for micro-controllers: Arduino33 BLE

ANKUR {STUDENT MENTOR AT IOSD UIET KUK}

https://www.linkedin.com/in/ankur-verma-0bb747154

https://github.com/Ankuraxz

ankurvermaaxz@gmail.com

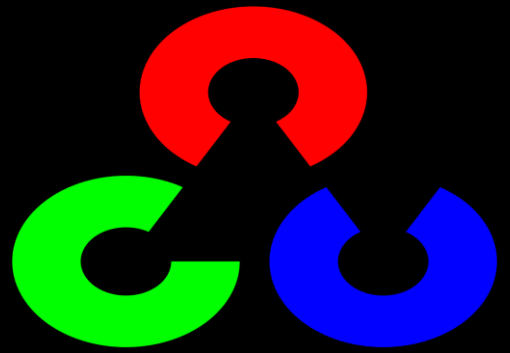

Computer vision

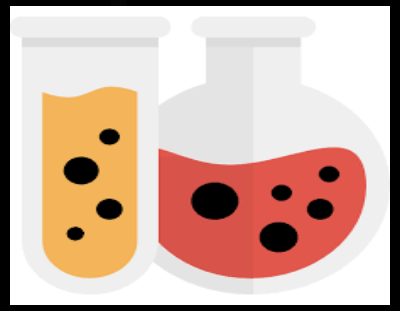
DAY3- IOSD DEVELOPMENT WEEK

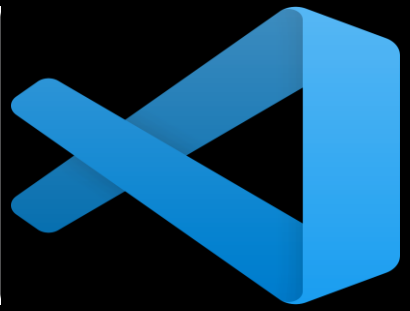
AGENDA:

1. Introduction to Computer Vision and Image Processing
2. Introduction to OpenCV-Python
3. Introduction to Haar-Cascade
4. Introduction to KNN Classification and SVM classification
5. Project on Facial Detection and Recognition
6. Project on Facial Features Detection and SnapChat Filter Generator
7. Introduction to Kaggle and Google Datasets
8. What's Next in AI? Completion of DAY 2 optional topics(OPTIONAL)
9. Reading a DL Paper and its necessity(OPTIONAL)

TOOLS FOR CV: (DAY 3)





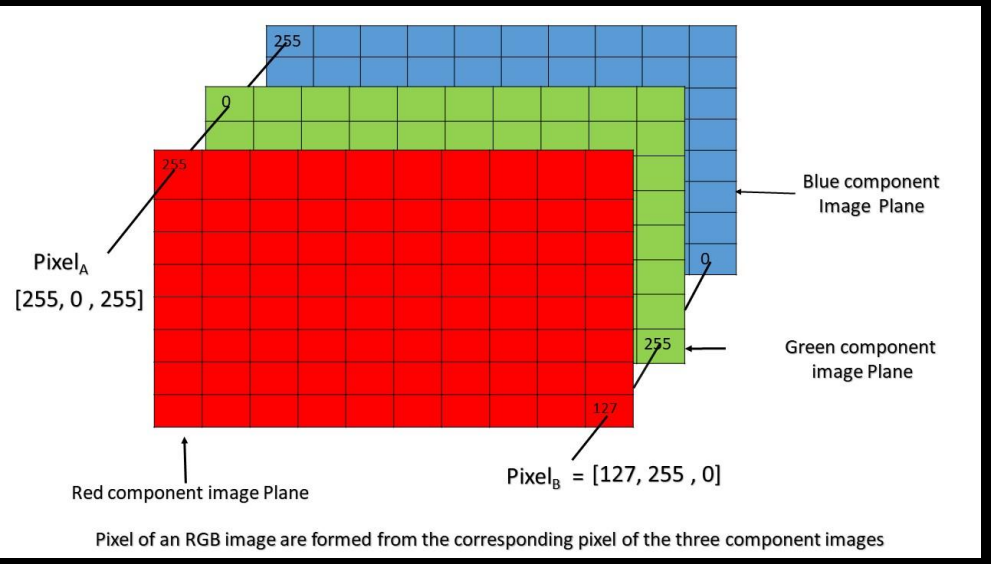


NumPy

COMPUTER VISION AND IMAGE PROCESSING

- Computer vision is an interdisciplinary scientific field that deals with how computers can be made to gain high-level understanding from digital images or videos. From the perspective of engineering, it seeks to automate tasks that the human visual system can do.
- **Image processing** is a method to perform some operations on an **image**, in order to get an enhanced **image** or to extract some useful information from it. It is a type of signal **processing** in which input is an **image** and output may be **image** or characteristics/features associated with that **image**. {Part of DSP}

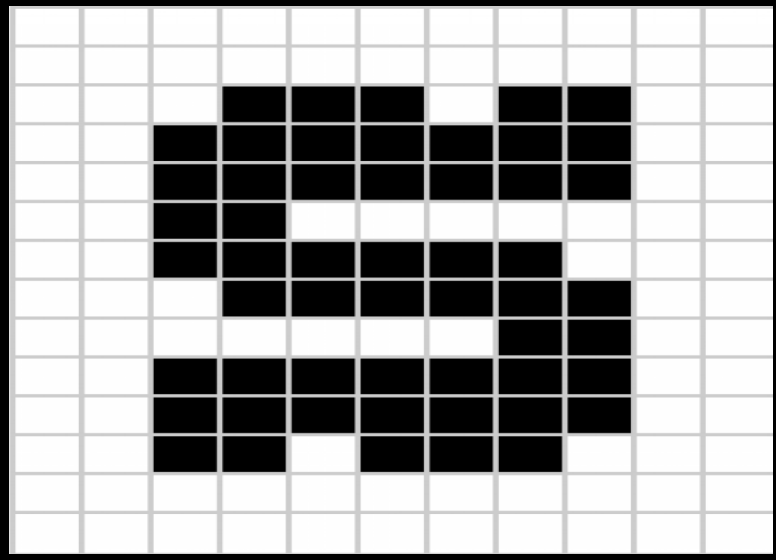
IMAGES AS AN ARRAY OF NUMBERS



RGB IMAGE

94	178	124	90	131	0
23	94	135	147	94	138
153	120	140	73	162	6
72	64	10	124	56	64
3	60	75	82	86	129
116	92	165	106	170	89

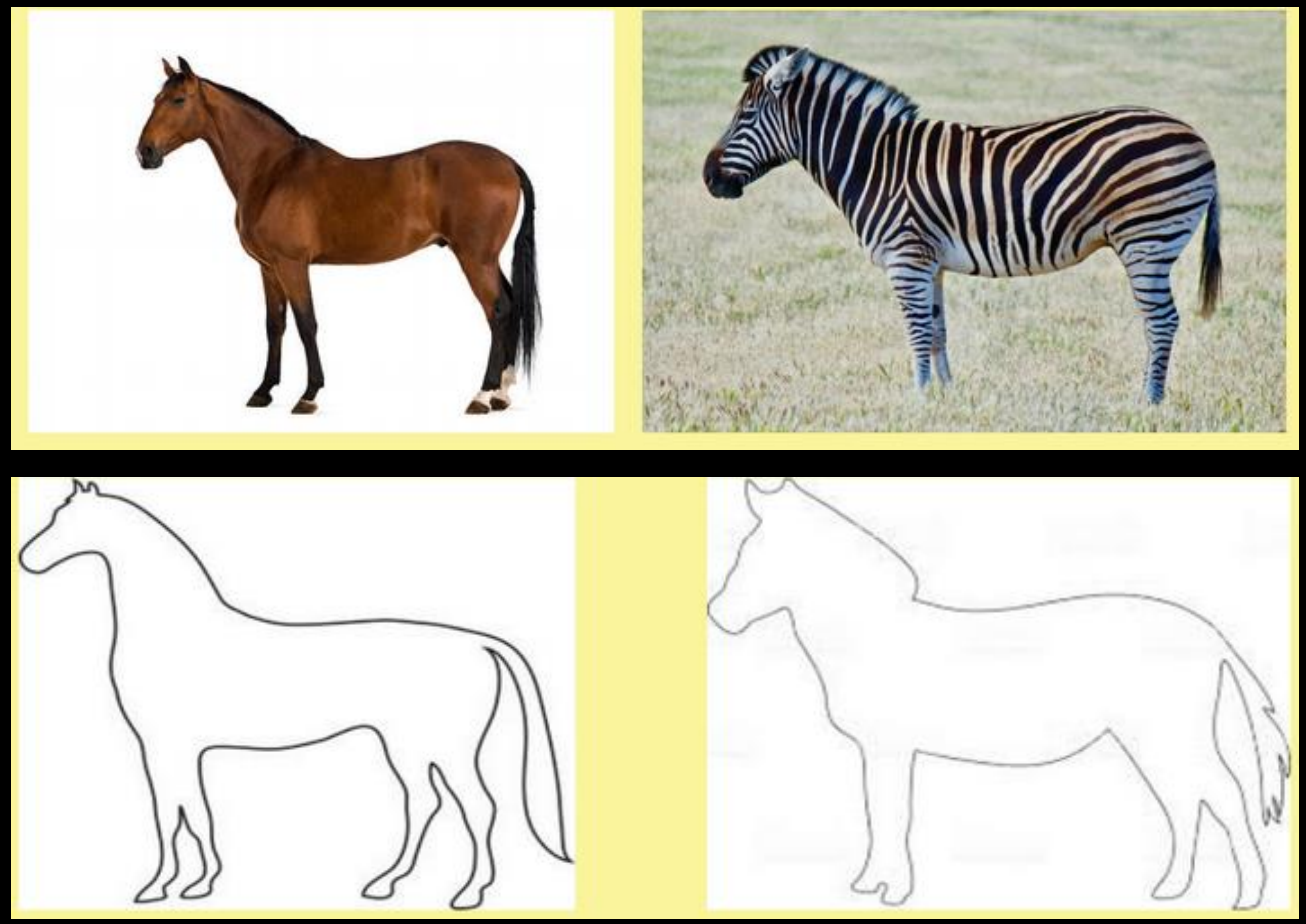
GrayScale IMAGE



B/W IMAGE

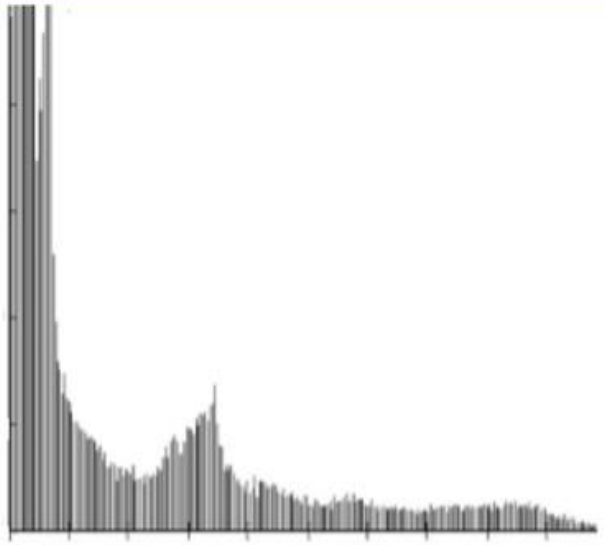
IMAGE DESCRIPTORS(REGION AND BOUNDARIES)

- Image Descriptors are features extracted using Haar-Cascade, HOG or " DEEP-Learning" that can be used in Image Detection or Segmentation.
- Calculating Distance such as Cosine Distance or Euclidian Distance between these features of various images all of different classes helps in Image Classification done using KNN, SVM, Logistic Regression, or " Deep Learning"

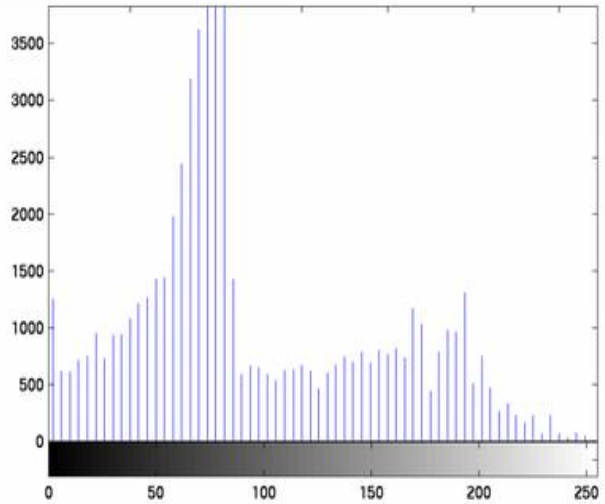


Boundary/Edge Descriptor

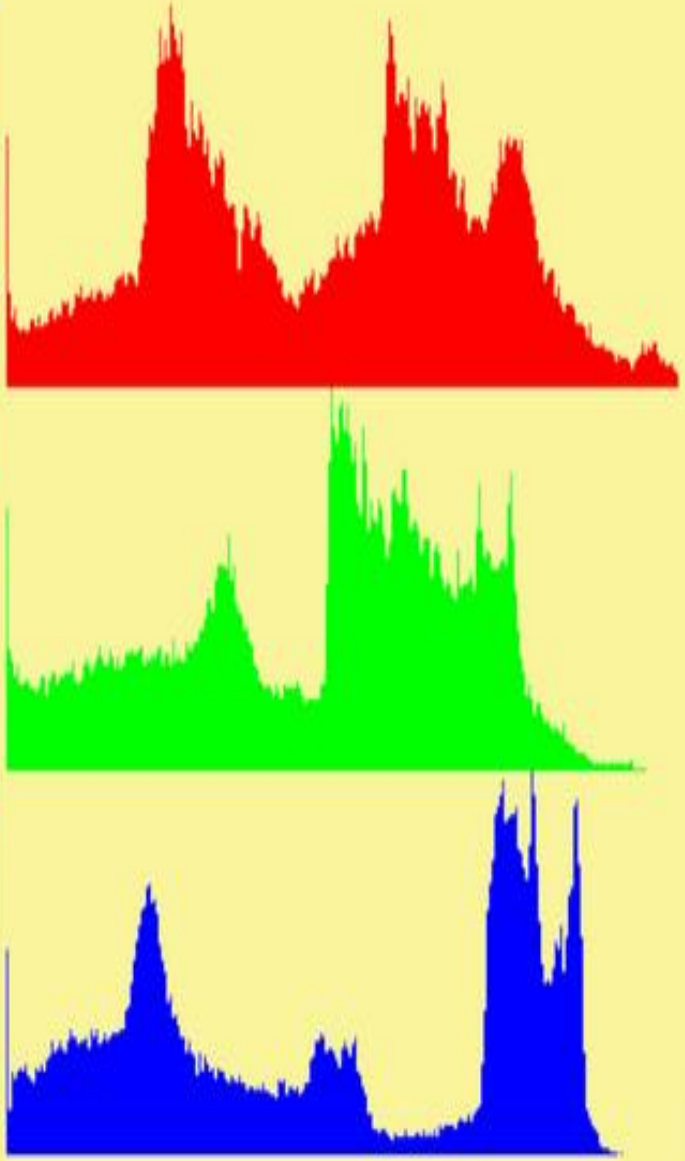
Intensity Descriptor



Intensity Descriptor



Colour Feature



OPEN-CV

Pip install opencv-python

- Library for Image Processing with a lot of features
- Imread() coverts image to array
- **OpenCV** (*Open source computer vision*) is a library of programming functions mainly aimed at real-time computer vision.Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source BSD license.
- OpenCV supports the deep learning frameworks TensorFlow, Torch/ PyTorch and Caffe

h

Open Source

OpenCV is open source and released under the BSD 3-Clause License. It is free for commercial use.

B

Optimized

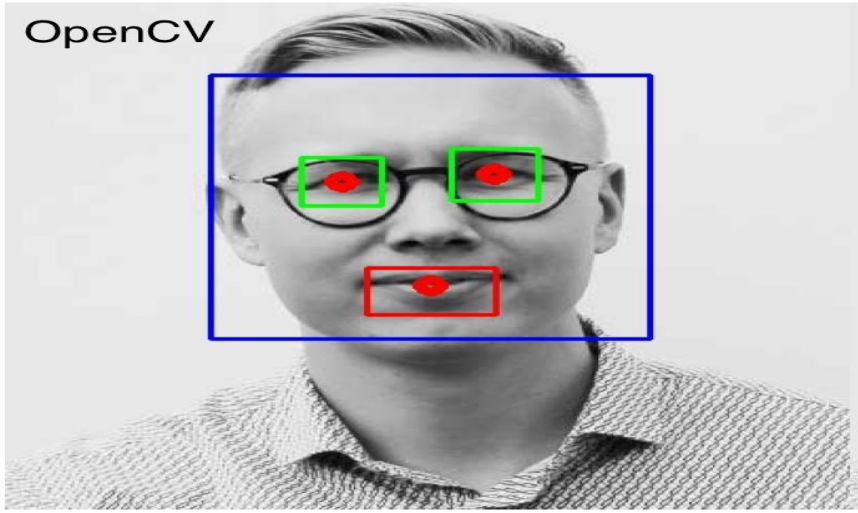
OpenCV is a highly optimized library with focus on real-time applications.

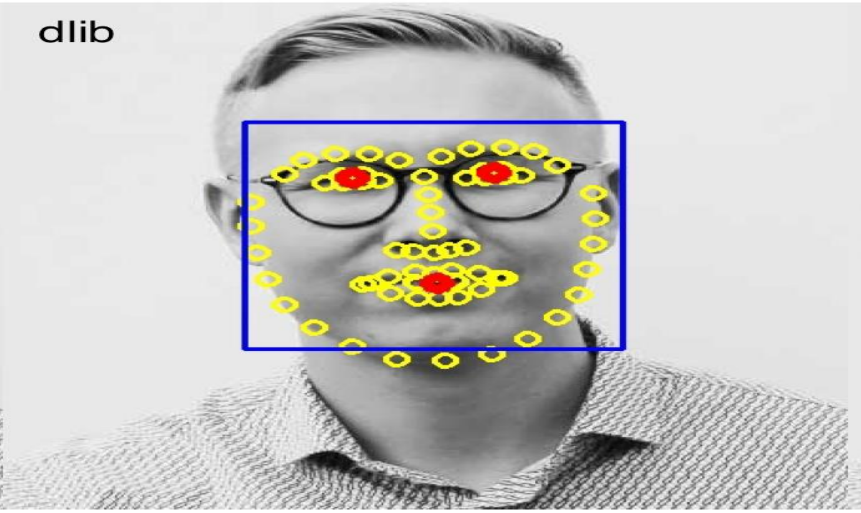
U

Cross-Platform

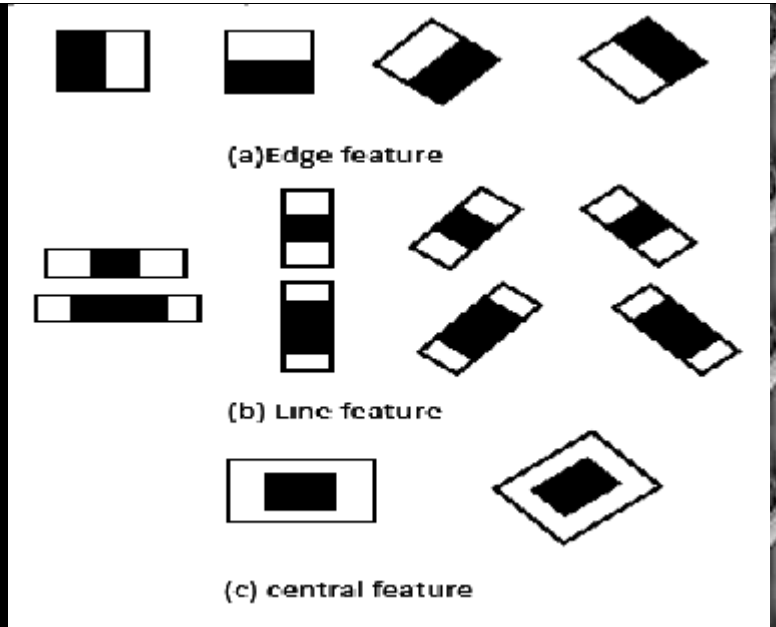
C++, Python and Java interfaces support Linux, MacOS, Windows, iOS, and Android.

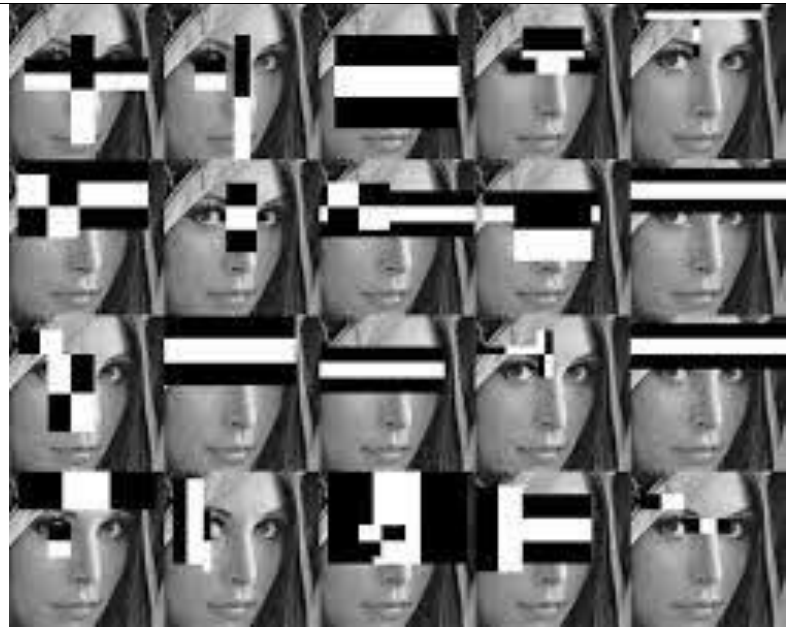
HAAR-CASCADE



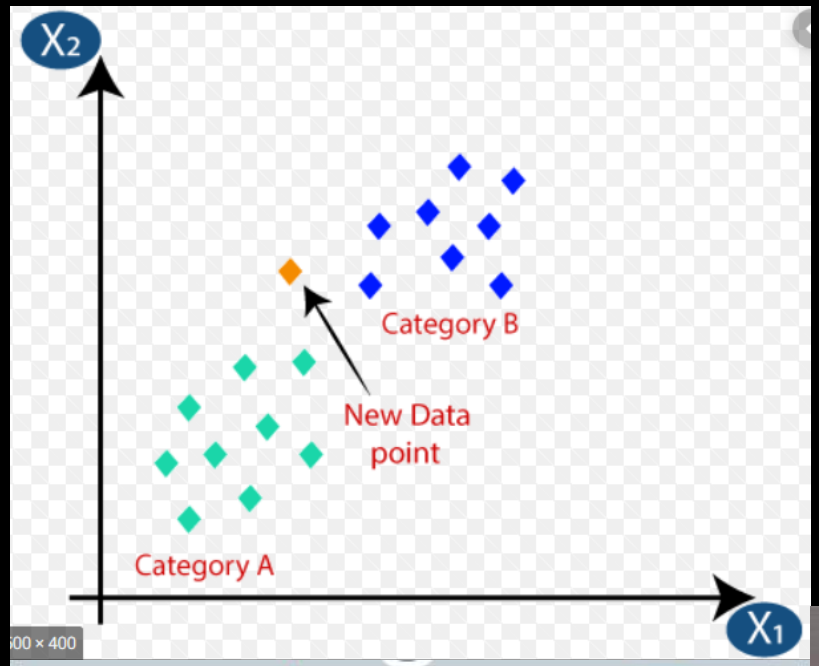


- Used in feature extraction and Object Detection
- **Haar-like features** are digital image features used in object recognition.
- DL is more advance and better for feature extraction





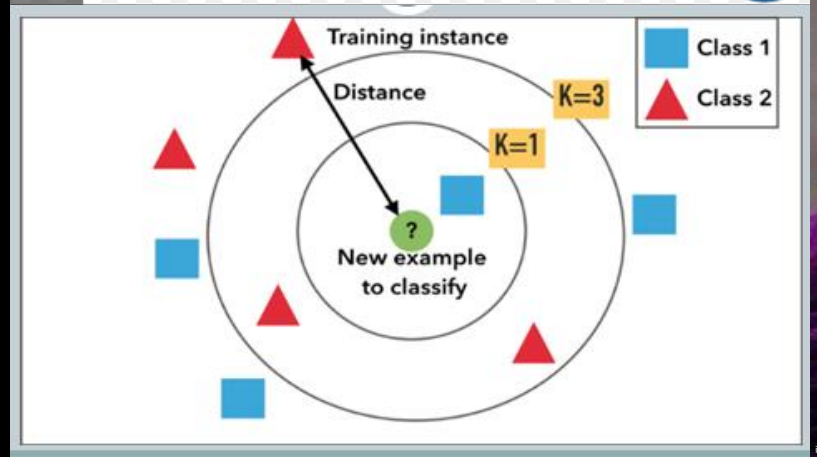
KNN-ALGORITHM FOR PROJECT



- **Machine Learning** for regression and classification problem. **KNN** algorithms use data and classify new data points based on similarity measures (e.g. distance function). Classification is done by a majority vote to its neighbors.
- MinKowski Distance : $D(X,Y) = (\sum_{i=1}^{n} |x_i - y_i|^{(p)})^{(1/p)}$
- Cosine Distance: Cosine Similarity (dot product of vectors)

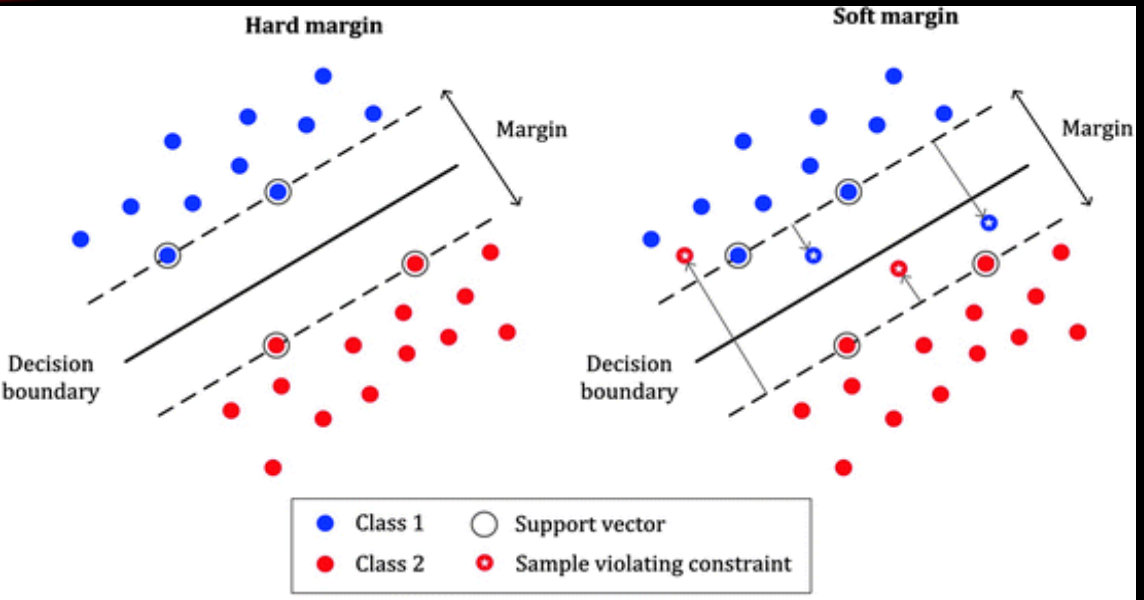
$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$$

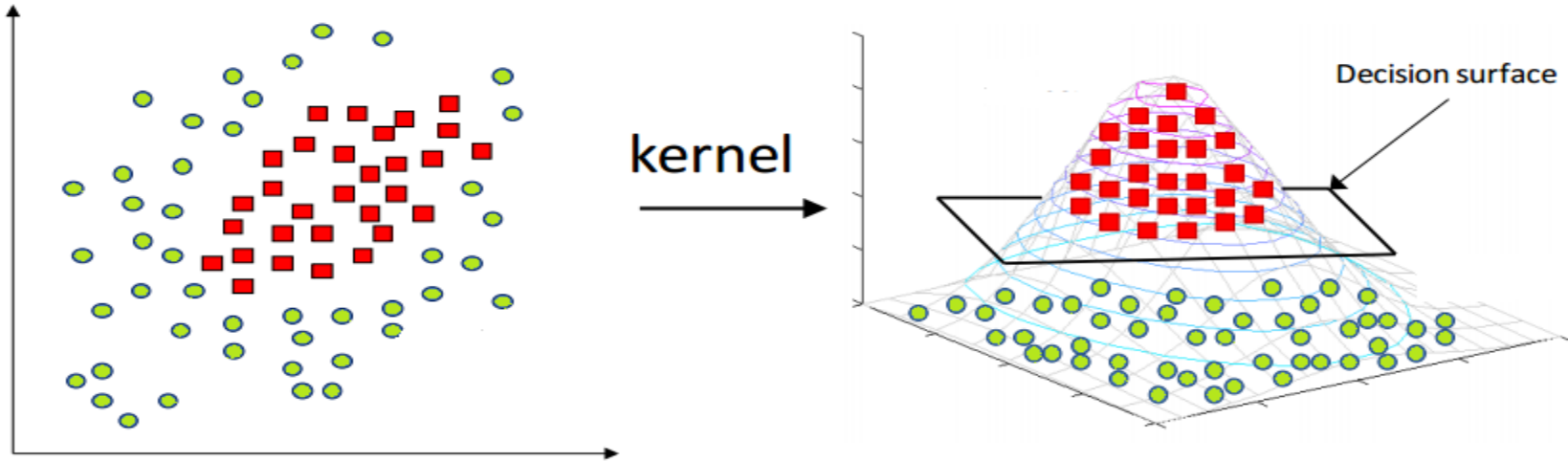


- P=1: Manhattan Distance
- P=2: Euclidian distance
- P = infinity: Chebyshev distance

SVM-ALGORITHM



- A support vector machine (SVM) is a supervised machine learning model that uses classification algorithms for two-group classification problems. After giving an SVM model sets of labeled training data for either of two categories, they're able to categorize new examples.
- The function of **SVM kernel** is to take data as input and transform it into the required form. ... These functions can be different types. For example linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid.

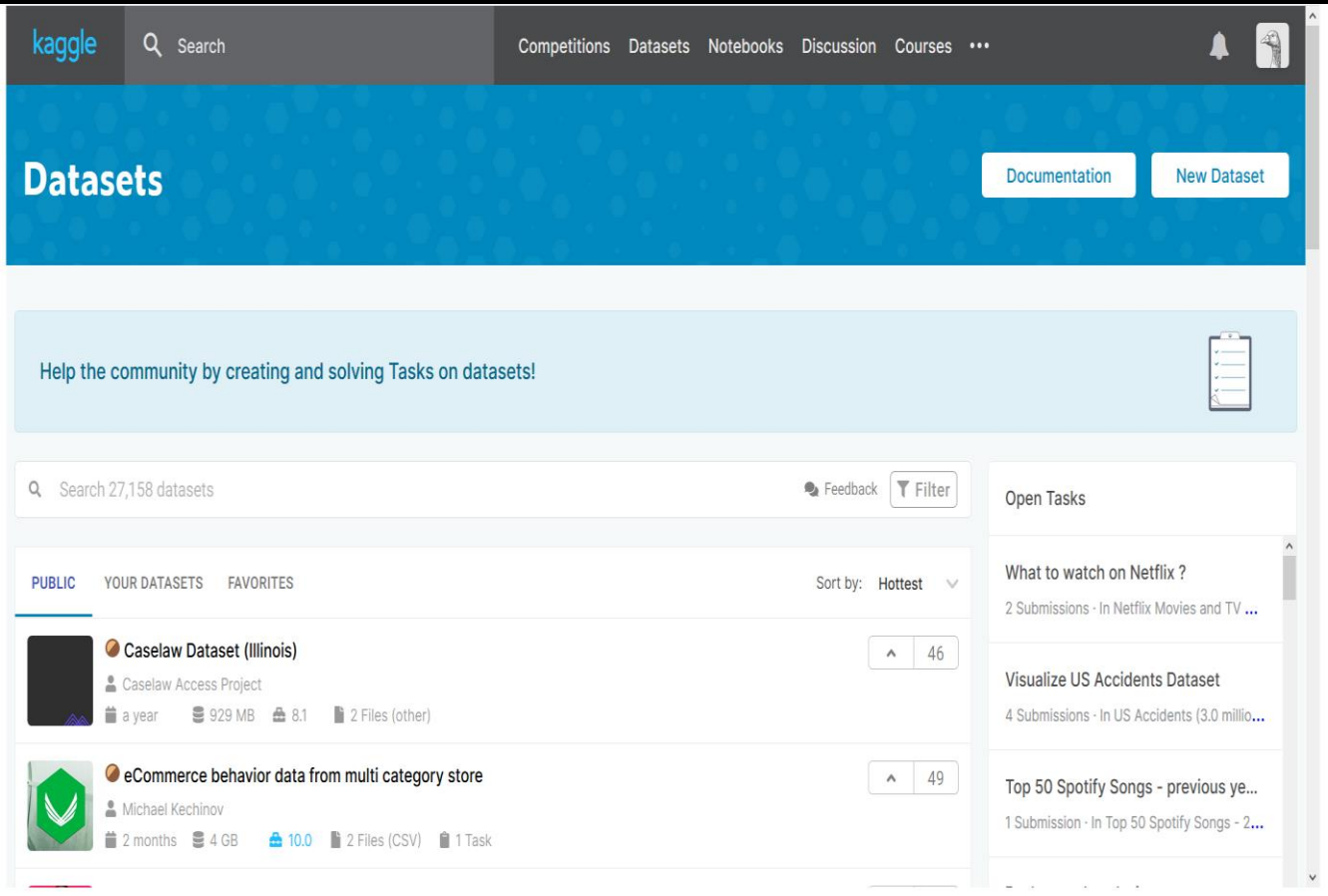


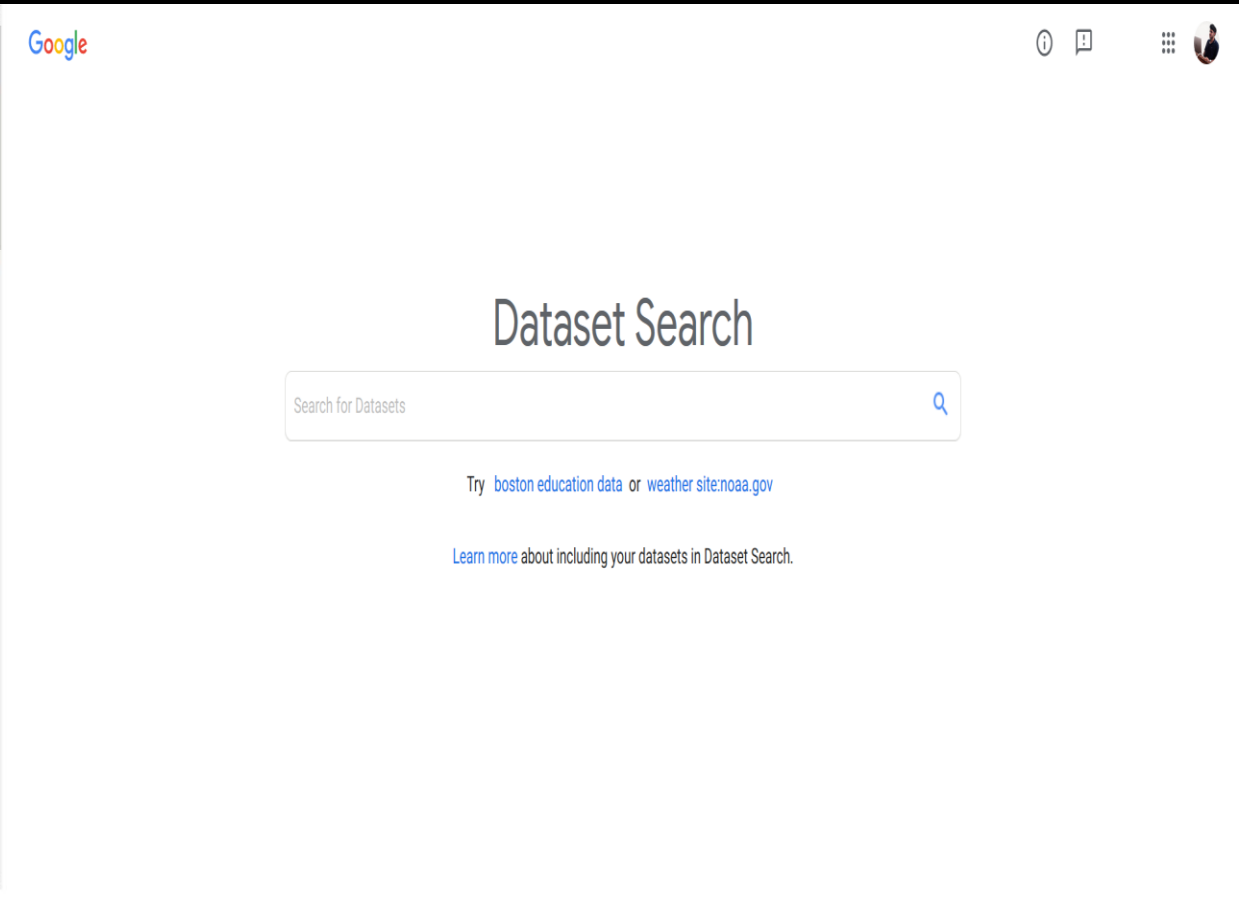
PROJECT WORK

- Detect face using Webcam and Haarcascade
- Save features as .npy
- Classify .npy features as names of persons using KNN/SVM
- Perform test in form of Face Recognition
- Detect feature like eyes, nose, lips
- Superimpose filters and DONE!!!

KAGGLE AND GOOGLE DATASETS

- https://www.kaggle.com/datasets
- https://datasetsearch.research.google.com/





WHAT'S NEXT IN AI