GROUP-1 Assignment

Algorithm Name	**Linear Regression**	**Logistic Regression**	**Artificial Neural networks**
What does it do	Regression Model	Classification Model	Both Regression and Classification
Loss Function	$(y-y')^2$	$-y\log(y')-(1-y)\log(1-y')$	
Use cases	Sales of product, House Price prediction..etc	Tumour Prediction, Spam detection..etc	Handwriting Recognition, Stock price prediction..etc
Time complexity	If n is the number of training data, W is the number of weights and each resolution of the weight space is set to m then $O(m^{W} * n)$	$O((f+1)csE)$ Where f = number of features(+1 because of bias) c = number of classes s = number of samples in your dataset e = number of epochs	The time complexity for training a neural network that has 4 layers with respectively i, j,k and l nodes, with t t training examples and n epochs. The result was **$O(nt*(ij+jk+kl))$** **$O(nt*(ij+jk+kl))$**.

Algorithm Name	**Decision Trees**	**SVM (Support Vector Machine)**	**K - Means**	**PCA (Principal Component Analysis)**
What does it do	Classification Model	Classification and Regression both	Classification Model	Regression Model
Loss Function	Gini Impurity : $G=i=1\sum C$ $p(i)*(1-p(i))$	**l(y) = max(0, 1 - t.y)** Where t is the actual output(either 1 or -1) and y is the output of the classifier		
Use Cases	Data Analytics, Civil Planning,	Face Detection, Classification of Images,	Image Segmentation, Image Compression,	Neuroscience, Quantitative

	Engineering	Handwriting Recognition	Document Clustering	Finance, Facial Recognition
Time Complexity	O(m - n^2), where m is the size of training data and n is the number of attributes.	O(n^3)	O(n^2)	O(n^3)

Gaussian distribution, often called Normal distribution, can be used to calculate the probability for any individual observation from the sample space which peaks out in the middle and gradually decreases towards both ends of the axis.

Uniform distribution is a probability distribution where probability is constant i.e it has all n number of possible outcomes equally likely.

Bernoulli distribution is a discrete probability distribution that has binary outcome as either 1 (success) or 0 (failure) for a single trial where probability of success and failure need not be equally likely.

Back propagation is a way of propagating the total loss back into the neural network to know how much loss every node is responsible for, and subsequently update the weights minimizing the loss i.e fine-tune the weights of neural network based on error rate obtained in the previous iteration by giving nodes with higher error rates lower weights and vice versa.