# MODULE-1

## What is Machine Learning?

Machine Learning is a branch of Artificial Intelligence that deals with algorithms that learn and improve through experience. It is the science of getting computers to act and learn like humans through real world observed data by generalizing from the data given to it. These algorithms include patterns that are learnt and relationships between huge amounts of data given to the algorithm. They build Mathematical Models on training data in order to make predictions on unseen data without explicitly being programmed to do so. The key features of Machine Learning Algorithms is their ability to independently adapt to new data. Machine Learning algorithms are used in computer vision applications such as face recognition, object detection , video action Recognition etc. It also has a wide variety of applications ranging from simple classification to self driving cars.It is used in NLP to perform speech recognition, Image captioning etc. Machine Learning is used for predictive analysis and forecasting of weather data, fraud detection, in medical applications to analyse data and identify trends which can lead to faster diagnosis and treatment. Machine Learning has great significance in the Transportation Sector where it is used in finding the most efficient routes or routes with lesser traffic. [1]

## SUPERVISED AND UNSUPERVISED LEARNING

SUPERVISED LEARNING

Supervised Learning involves the mapping between a set of input variables (features) X and a set of output variables (labels) y which is applied to unseen data. Supervised Machine Learning algorithms are applied when the training labels or the outputs are available in the dataset provided. Y=f(X). All the data is labelled. Learning stops when the algorithm achieves an acceptable level of performance.
Supervised Learning is grouped into Regressors and Classifiers.[2]
We use classifiers when the output variable is a category(0 or 1). We use regressors when the output is real data(For eg: Weight).

Regression

[3]Regression is a branch of supervised learning that models the input variables with the continuous output variables. This technique is used for forecasting and time series analysis. Simple Linear Regression sets an arbitrary line through the data points and calculates the distance between the data points and the line. This gives us the prediction error. Through each

iteration, in order to minimize the error, the algorithm moves the line in hopes of figuring out the best fitting line.(Minimum error). The distance by which we move the line through each and every iteration is governed by the learning rate which is multiplied to all the other parameters. The method used to find the best fitted line is called the Gradient Descent Method.

Gradient Descent

[4]Our main aim is to minimize the distance between the predicted values and the actual values i.e We need to minimize the following cost(squared error) function:

$$J(\theta_0,\theta_1)=(1/2m)\Sigma^{m}_{i=1}(h_\theta(x^{(i)})-y^{(i)})^2$$

The learning rate determines how big the steps are to decrease J .If the learning rate is too small, the algorithm takes infinite time to reach a global minimum. On the other hand, if the learning rate is too large, the algorithm may fail to converge at the global minimum. When the algorithm reaches the minima, the derivative of the cost function becomes 0 at a fixed learning rate. The derivatives calculated decrease as the algorithm approaches the minima. Using this method, we may reach the local minima rather than the global method which is not the most optimized solution and there aren't any ways to get out of it.
To address this we use a method called Stochastic Gradient Descent. [5] In stochastic gradient descent, one training sample at a time is passed to the network or the algorithm at a time and the weights of each layer are updated correspondingly with the gradient(Neural Networks). The weights are updated frequently and hence it can converge faster to the minimum(For larger datasets). [6] In Linear Regression, the training examples are fed into the model one at a time, the model predicts the output and then the model is updated in order to reduce the error.

$$W = W - \alpha dW$$

As the weights are frequently updated, the steps taken have oscillations which helps the algorithm to exit the local minima. This feature however can prove to be problematic as frequent updates may lead to gradient descent in other directions.
[5]Batch Gradient Descent:
Here we pass the training data at once rather than one sample at a time. As a result of this, we get lesser oscillations and less noisy gradient descent with a stable convergence to the global minimum. It is computationally more efficient when compared to stochastic gradient descent. As the training datasets are usually large, additional memory might be required and the problem of the local minima still persists.
[7] Mini Batch Gradient Descent:
The number of samples per batch is taken between 1 and m(No of training examples). It is a trade off between stochastic and batch gradient descent. The cost is averaged over a smaller

number of samples. The noise or oscillations required to come out of the local minima , along with a stable convergence ,is retained.

Multiple Linear Regression:

[8]Multiple Linear Regression is a technique used when we have 1 dependent variable (y) and n independent variables(X). Here, we assume that the independent variables are not correlated with each other i.e they are not dependent on each other. The targeted values are selected independently and randomly. The reliability of the model is explained through the $R^2$ metric or the coefficient of determination. It measures how close the data fits to the regression line but this metric cannot indicate the bias of the predictions.

UNSUPERVISED LEARNING

[9] In some applications, the training data consists of input variables X without any labels and is required to find it's own structure in the input. In some places we may have to figure out the similarity in the data points and group them into clusters. These clusters can be later used for classification. An example of this approach would be extracting colors from an image. The pixel values extracted for every shade of each and every color are grouped together based on the distance between them. These groups can represent a particular color. We can choose how many clusters we require.

K means Clustering

Each data point's similarity is measured using euclidean distance metric or the correlation distance metric such that distance is minimum.The idea of this algorithm is to define k centers as far away from each other as possible in a 2D space. The algorithm is very sensitive to these randomly initialized centers. Next, we take each point and associate it with the nearest center. KMeans first starts with randomly selected centroids and performs repeated iterations to optimize the position of the centroids. Once the positions of these centers are stabilized, the iterations come to a halt.
Expectation-Maximization Approach:
The Expectation-Step is assigning the data point to the closest cluster and the Maximization-Step is computing the centroid for each cluster. This uses the Maximum Likelihood Estimation approach where the variables are first estimated and then optimized.

$$J=\Sigma_{i=1}^{m} \Sigma_{k=1}^{K} w_{ik}||x^i-\boldsymbol{\mu}_k||^2$$

If the data point($x^i$) belongs to the cluster, $w_{ik}$ is 1 or else $w_{ik}$ is 0. $\boldsymbol{\mu}_k$ is the centroid of the respective cluster. Then we minimize J by finding the partial derivatives of J with respect to the

$w_{ik}$ and $\boldsymbol{\mu}_k$. On differentiating with respect to $w_{ik}$ (Extractive Step), we update the cluster assignments. Next we differentiate with respect to the $\boldsymbol{\mu}_k$ and recompute the centroids after the cluster assignments(Maximization step).

Fuzzy K Means Clustering :

Fuzzy K Mean Clustering discovers soft clusters where a data point can belong to more than one cluster with a certain probability. The algorithm is similar to k means.

## CLASSIFICATION STUDY

[10]Classification Algorithms come under the Supervised Machine Learning Technique where the target variables are discrete,unordered values. [11]Natural Language processing uses classification algorithms to solve a wide range of problems.

Sentiment Analysis:

Sentiment classification is the most common application of NLP where the sentiment of the text or article is classified into a given set of categories. This is challenging even for humans. Sentiment analysis takes in the text as the input and classifies whether the comment is positive or negative. This is used to give movie reviews, to poll opinions on social media etc.

[12] Neural Networks have succeeded in a lot of NLP tasks such as machine translation, text summarization etc. However , it fails to address the aspects level sentiment analysis, where one part of the sentence may be positive and while the other may give a negative review. In order to capture relevant information in response to a given aspect,LSTM based Attention mechanism is used.Therefore Attention Mechanisms have proved to give superior results when compared to the others. Attention Mechanism , as the name suggests, figures out how much 'attention' it should pay to a given part of the sentence. This is handled by the parameter $\boldsymbol{\alpha}$. As it is possible to get opposite polarities from a sentence, the paper proposes to learn an embedding vector for each aspect . The embedding vector is calculated by multiplying the embedded matrix(similarity between the words of the sentence) with the one hot encoding of the word. This is the aspect embedding e. The attention weight($\boldsymbol{\alpha}$) is calculated from the embedding vectors with the formula:

$$\boldsymbol{\alpha}^{<t,t'>} = \exp(e^{<t,t'>})/(\Sigma_{t'=1}^{Tx}\exp(e^{<t,t'>}))$$

W is the word representation of the sentence with length N . This vector is passed through an LSTM to generate the hidden units H $\{h_1,h_2\ldots..h_n\}$ These hidden units are multiplied with $\boldsymbol{\alpha}$ to give the weighted representation of the sentence of a given aspect.

Activation functions:

[14]Softmax Activation function - This activation function outputs a vector that represents the probability of all the outcomes. Hardmax classifier gives the outputs as 1 or 0 where 1 represents the predicted category.The value ranges from -1 to 1.

$$S(y_i) = e^{yi}/\boldsymbol{\Sigma}_j e^{yj}$$

Tanh activation function:

[15]Tanh activation function comes with the vanishing gradient problem where the gradient tends to get smaller and smaller in the backpropagation optimization. The earlier neurons tend to learn slower than the later neurons. This results in a decrease in the prediction accuracy and the model takes a long time to train.

$$\tanh(x) = 2/(1+e^{-2x})-1$$

Sigmoid Activation function:

The sigmoid activation function translates the input range from -inf t-+ inf to (0,1]. Softmax is a more generalized version of this activation function. The problem of vanishing gradients also persists in this case.

$$\boldsymbol{\sigma}(x) = 1/(1+e^{-x})$$

[16]SVMs for classification:

The objective of Support Vector machines is to find the hyperplane in N dimensional space that distinguishes all the data points from different classes. Hyperplanes are chosen in such a way that the distance from the hyperplane to the data points from each class should be maximum. This is called the maximum margin. This makes sure that future points are classified with more confidence. The dimension of the hyperplanes depend on the number of features present. The margin of the classifier is maximized by taking into consideration the data points which are closest to the hyperplane(Support Vectors). These vectors highly determine the position and the orientation of the hyperplanes. The threshold values in SVM are changed to -1 and 1, the range of values ([-1,1]) acts as the margin.

For the given training data D, the set of n points is in the form

$$D = \{(x_i,c_i) \mid x_i \in R^p, c_i \in \{-1,1\}\}_{i=1}^{n}$$

*Where each $x_i$ is a p dimensional vector and $c_i$ is the class it belongs to {-1,1}.*

Any hyperplane can be written as the set of points x satisfying w.x - b =0 where w is the normal vector perpendicular to the hyperplane. To maximize the margin, w must be minimized. To prevent the data from falling into the margins ,

$$c_i(w.x_i - b) >= 1 \text{ for all } i = 1,2,....,n$$

SVMs are greatly effective when we take into consideration the extreme cases. These data points are the support vectors. Sometimes the data points are not linearly separable. In this case, we use the non linear SVMs that transfer the data into high dimensional space, but this is computationally expensive. Hence, we use the kernel trick to transform non linear SVMs into linear SVMs. The cost function for the SVMs is called the hinge loss. If the predicted value and the actual value are of the same sign, this shows that it has been classified correctly. Therefore the cost is 0.

For the sentiment analysis problem statement, the data is linearly separable with two classes i.e ‘positive’ or ‘negative’. Hence the kernel used here is the Linear SVM. Regularization parameter ‘c’ is added to decrease the overfitting and balance the margin maximization and loss.

The f1 score for the sentiment analysis on amazon review data with SVM was 91%. The f1 score is the harmonic mean of the precision and the recall values.

[17]Precision- The ratio of the correctly predicted positive values to the total number of positive values. High precision relates to low false positive rate.

$$\text{Precision} = \text{True Positive}/(\text{True Positive}+\text{False Positive})$$

Recall- Recall is the ratio of correctly predicted positive labels to all the observations.

$$\text{Recall} = \text{True positive}/(\text{True positive} + \text{False Negative})$$

F1 score works better than accuracy if we have uneven distribution of classes.

## CLASSIFICATION METHODS:

[18]Logistic Regression:

Logistic Regression is a common method for solving binary classification problems. Multinomial Classification is used where there are multiple classes present. For ex: IRIS dataset. Logistic Regression can be used for diabetes prediction, spam detection and the type of cancer (Malignant or Benign). Logistic Regression uses the logit function to predict the probability of occurrence of a binary event. The dependent variable in logistic regression follows the Bernoulli Distribution. It is a discrete distribution having two possible outcomes given by the equation:

$$P(n) = p^n ( 1 - p^n )$$

where p is the probability and n is the label 0 or 1.

Logistic Regression is estimated using the Maximum Likelihood Estimation. This estimation determines the parameters that are most likely to produce the observed data. The mean and variance are the parameters used for predicting the data in the normal distribution.[19] Different values of variance and mean result in different curves. This estimation figures out those parameter values which results in a curve that best fits the data. The probability density of observing a single data point x is :

$$P( x ; \boldsymbol{\mu},\boldsymbol{\sigma}) = 1/(\boldsymbol{\sigma} \sqrt{2} ) \exp(-(x - \boldsymbol{\mu})^2/2\boldsymbol{\sigma}^2)$$

The mean and variance are found out by maximizing the above function. We assume that the data points are independent of each other to avoid conditional probability. The natural log of the expression is taken to differentiate it. As natural logarithm is a monotonically increasing function, it ensures that the maximum value of the log of the expression occurs at the same point as the original expression. The sigmoid function is called the logistic function. If the output of sigmoid is greater than 0.5, we classify it as 1 or else 0.

[20]Tree Based Classification:

Tree based algorithms are considered as one of the most common supervised Machine learning techniques. They can handle linear relationships as well as non linear relationships with great accuracy. Tree based algorithms can be used for both regression and classification. This approach requires less cleaning when compared to the other modelling techniques and is not affected by missing values or outliers.

Decision Tree Classifiers- In this classifier, the population is split into two or more homogenous sets based on the significant splitter in the input variables, i.e the input feature which affects the target value the most. The root node represents the entire population or the sample. This further gets divided into sub nodes. When a sub node divides further, this is called the decision node. The terminal node or the leaf node does not divide into further nodes. The decision tree follows the 'top down greedy approach' as it cares only about the current split and doesn't care about future splits which might lead to a better position. The value obtained by the leaf node is the mode of the observations in that region while in regression, the mean of the observations is taken. Decision Trees uses multiple algorithms on how it decides the split. The common algorithm is the gini test.

[21]Gini algorithm:

Gini impurity metric evaluates how good a split is. A datapoint is randomly picked and classified according to the class distribution. The probability that the data point is classified incorrectly is called the gini impurity. The formula for the gini impurity is given by:

$$G = \mathbf{\Sigma}_{i=1}^{C} p(i) * (1 - p(i))$$

*Where C is the total classes and p(i) is the probability of picking a datapoint with class i.*

A gini impurity of zero indicates the best possibility which can be achieved when everything is the same class. The gini impurity of the split will be low if the most effective input feature is taken as the root node for the split.

Random Forests - Decision Trees are more prone to overfitting as they can be too specific when we deal with smaller samples. Therefore, we use Random Forests. Random Forest consists of many decision Trees that have very low correlation with each other. If some trees provide wrong predictions, there are many more that provide the correct ones. This classifier decreases the chances of overfitting as different samples of the data are trained each time or random subsets of the feature are trained on. Decision trees are very sensitive to the data they are trained on. Small changes can result in a different tree. Random forest allows each tree to randomly sample from the dataset. This is called bagging ( bootstrap aggregation). For data with different levels, Random Forests may be biased towards those attributes with different levels. They take up large amounts of memory and are slow to evaluate.

[22]XGBoost Algorithm

AdaBoost Algorithm - Adaboost Algorithm begins by assigning each observation with an equal weight while training the decision tree. After training, the observations which are difficult to

classify are given greater weights while the others have lower weights. The second tree is trained on these weights. Therefore the new model is Tree1 + Tree2. The classifications are predicted from this tree and a new third tree is grown from the residuals. This process helps us classify observations that are very difficult and are not classified well by the previous ones. Prediction of the final model is the weighted sum of the previous models. The major difference between the AdaBoost and Gradient Boosting is how they identify the misclassifications. Adaboost uses high weights while Gradient Boost uses gradient descent in the loss function which is the measure of how well the coefficients fit the data. Gradient Boost allows the optimization of the cost function to achieve better results. XGBoost or Extreme Gradient Boost is a library for gradient Boost algorithms used widely for classification problems.

Naive Bayes Algorithm:
Naive Bayes assumes that each and every feature in a class is completely unrelated to the other features. This is a huge disadvantage as it barely occurs in real life. Bayes Theorem is as follows:

$$P(A|B) =( P(B|A) P(A))/P(B)$$

*Where P(A) is the priori probability and P(A/B) is the posterior probability*

Naive Bayes Classifier is used in text classification problems where the probability of each and every word of the sentence is calculated given that it is present in that particular category. For example, if we want to predict the class of the sentence 'A very close game' in the category of sports/ non sports, The probability of each and every word in that sentence given that it belongs to sports and the same for non sports must be calculated.

$$P(\text{A very close game}|\text{sports}) = P(\text{A}|\text{sports})P(\text{very}|\text{sports})P(\text{close}|\text{sports})P(\text{game}|\text{sports})$$

$$P(\text{A very close game}|\text{sports}) = P(\text{A}|\text{non sports})P(\text{very}|\text{non sports})P(\text{close}|\text{non sports})P(\text{game}|\text{non sports})$$

All of these probabilities are calculated by the Laplace Smoothing method i.e

$$P(\text{word}) = ( \text{word Count} + \boldsymbol{\alpha}) / (\text{Total number of words} + \text{Number of unique words})$$

Where $\boldsymbol{\alpha} = 1$ (Laplace Smoothing)

This makes sure that we do not get zero probabilities.
The probability with the highest score is taken . It was found that P(A very close game|sports) > P(A very close game|non sports). Therefore 'A very close game' most probably belongs to the category sports. Gaussian Naive Bayes Classifier assumes that the data follows a normal distribution. The above problem implements the MultiNominalNB naive Bayes algorithm where it counts how often a word occurs in the document. If a variable present in the test dataset is not present in the training data, a zero probability is assigned and we do not get the correct prediction. This is called Zero Frequency. Laplace Smoothing addresses this problem.

[23]CNN for classification:
Convolutional Neural Networks are one of the most common and most powerful Neural Network architectures used for various Computer Vision problems ranging from a simple dog- cat classifier to self driving cars. However Convolutional Neural Networks can also be used to address many problems that come under NLP also. Convolutions are nothing but 'sliding windows' applied to the input matrix be it image or text. We perform an element wise multiplication of a 3x3 filter over the input matrix and then sum it up. CNNs are several layers of convolutions and non linear activations on relu, tanh applied to results. During the training phase, these CNNs learn the values of filters based on the task we want to perform. Probably in the first layer, it might detect the vertical and the horizontal edges, the second layer, it might detect simple shapes and as the Network progresses, it might be able to detect high level features in the last layers. Therefore each filter composes a local path from low dimensionality to high level dimensionality which is why CNNs are so powerful.
Zero padding- As the filters cannot be applied to the edges of the matrix, zeros are appended to all elements outside the matrix. This is known as 'wide convolution'.
In NLP tasks, the inputs are usually sentences or documents represented as matrices. Each row is a vector that corresponds to a word. These are usually word embeddings(word2vec or Glove) of n dimension. If there are m words in a sentence, the resultant matrix is of the size mxn.
Stride size - This is defined by how much you want to shift the filter over the matrix. Larger the stride size, smaller the output size.
Pooling - The pooling layer usually comes after the convolution layer. The most common pooling used is the 'max pooling'. Pooling can be applied to the whole matrix or to just the window. The maximum value of each window is taken to construct the output matrix. For nlp purposes, usually the whole matrix is considered and then we get only 1 output after passing it to the pooling layer. One useful property of the pooling layers is that the output size is always fixed. If we have n filters, we get an n dimensional output regardless of the size of the filters or the size of the input. It keeps the salient features, applying max pooling gives information on whether or not the feature appeared but losing information about where it appeared. You are losing the global information about the locations but keeping the local information captured by the filters.
Given this RNNs are considered to be a better fit for NLP applications rather than CNNs.
Channels - In image recognition, you have RGB channels i.e how the data is viewed and in NLP you might have channels for different word embeddings.
Applications of CNN in NLP:
CNNs can be applied to spam detection , Sentiment Analysis etc. For sentence classification, the input layer is a sentence composed of word2vec embeddings followed by convolutional layers with multiple filters, then max pooling layers and then finally a softmax layer for classification. Some CNN models have been trained from scratch that is passing the one hot encoded vector of each word as the input. Research has also been done on applying CNNs to characters without

any pre trained word embeddings. Direct character level inputs work on large datasets but fail on simpler models.

[24]RNN for classification:

Recurrent Neural Networks are networks with loops in them that allow information to persist. They can be considered as a chain of networks where each of them pass a message to its successor. RNNs are applied to a variety of problems such as Speech Recognition, Image Captioning, translation etc. These networks have the capability of retaining past information and predict the next word based on this information. In cases where the gap between relevant information and places that it's needed is small, RNNs can learn from past information. Sometimes we need more context. If the gap between relevant information and place of the word increases, RNNs face difficulty in learning. For example: 'I grew up in France.. I speak fluent French'. If the word 'French' is to be predicted, the network requires the context 'France'. In this case, the gap is considerably large.

LSTMs( Long Short Term Memory):

LSTMs are used to address the above issue. The key parameter of the LSTM is the cell state. The cell state consists of information that may be removed or added by the LSTM. The information stored in the cell state is controlled by the gate which contains a sigmoid layer and a pointwise multiplication operator. It decides as to how much information must be passed through.

For example: we might want to store the gender of a subject. When we see a new subject, we might want to forget the gender of the old subject.

$$f_t = \sigma(W_f . [h_{t-1} , x_t] + b_f)$$

If the output of this layer is 0, the cell state forgets the previous information but if the output is 1 it retains it.

Next, we decide on what to update to the cell state. First we have the sigmoid layer that decides which values we will update to the cell state. Then a tanh layer that gives a vector of new values that could be updated to the cell state. These two are combined to give the update state.

$$i_t = \sigma(W_i . [h_{t-1} , x_t] + b_i)$$
$$\tilde{C}_t = \tanh(W_C . [h_{t-1} , x_t] + b_C)$$

We then multiply the old cell state with the $f_t$ and then add it to the product of the above equations.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Next, we decide on what we are going to output. We run a sigmoid layer to get what parts of the cell state we are going to output, then we pass the cell state through the tanh layer( to get values between -1 and 1). We multiply this to the output of the sigmoid layer.

$$o_t = \sigma(W_o . [h_{t-1} , x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$

In language models , LSTMs might want to output the relevant verb based on whether the noun is singular or plural. For example $C_t = 1$ if singular and 0 if plural. An extension for LSTM is a Bidirectional LSTM which takes in information from both earlier and later parts of the sentence. [13]They have many applications in handwriting recognition, emotion recognition, translation, phoneme classification , human behaviour analysis, audio video data etc.

**REFERENCES:**


1.https://www.sas.com/en_us/insights/analytics/machine-learning.html

2.https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

3.https://towardsdatascience.com/supervised-learning-basics-of-linear-regression-1cbab48d0eba#:~:text=Regression%20analysis%20is%20a%20subfield,and%20a%20continuous%20target%20variable.

4.https://www.hackerearth.com/blog/developers/gradient-descent-algorithm-linear-regression/

5.https://medium.com/@divakar_239/stochastic-vs-batch-gradient-descent-8820568eada1

6.https://machinelearningmastery.com/linear-regression-tutorial-using-gradient-descent-for-machine-learning/#:~:text=called%20multiple%20regression.-,Stochastic%20Gradient%20Descent,gradients%20of%20the%20cost%20function.&text=In%20Machine%20learning%20we%20can%20use%20a%20similar%20technique%20called,model%20on%20our%20training%20data.

7.https://adventuresinmachinelearning.com/stochastic-gradient-descent/

8.https://www.investopedia.com/terms/m/mlr.asp

9.https://towardsdatascience.com/unsupervised-learning-and-data-clustering-eeecb78b422a

10.https://books.google.co.in/books?hl=en&lr=&id=vLiTXDHr_sYC&oi=fnd&pg=PA3&dq=classification+algorithms+in+machine+learning&ots=CYtvAz3Cjm&sig=aEmfpTNUivmwTwSxzyQ3CWxWQPI#v=onepage&q=classification%20algorithms%20in%20machine%20learning&f=false

11.https://chatbotslife.com/top-5-applications-of-natural-language-processing-d45409c711e3

12.https://www.aclweb.org/anthology/D16-1058.pdf

13.https://link.springer.com/article/10.1007/s00521-017-3210-6

14.https://medium.com/data-science-bootcamp/understand-the-softmax-function-in-minutes-f3a59641e86d

15.https://towardsdatascience.com/complete-guide-of-activation-functions-34076e95d044

16.https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

17.https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/#:~:text=Recall%20(Sensitivity)%20%2D%20Recall%20is,observations%20in%20actual%20class%20%2D%20yes.&text=F1%20score%20%2D%20F1%20Score%20is,and%20false%20negatives%20into%20account.

18.https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python

19.https://towardsdatascience.com/probability-concepts-explained-maximum-likelihood-estimation-c7b4342fdbb1#:~:text=Maximum%20likelihood%20estimation%20is%20a,data%20that%20were%20actually%20observed.

20.https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/

21.https://victorzhou.com/blog/gini-impurity/

22.https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab

23.http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

24.http://colah.github.io/posts/2015-08-Understanding-LSTMs/

# MODULE - 2 - Data Preprocessing

## Why is data preprocessing required in general?

The desired outcome of any problem statement is highly correlated with the data we pass to the algorithm. Machine Learning Algorithms learn by creating patterns in the huge amounts of data passed to it and they get better by gaining more and more experience while training. Therefore it is required to pass relevant data of a particular data type if we expect the model to give us good results. [1] Real world data comes in different forms broadly categorised into unstructured(images,text,audio) and structured data(tables). This data needs to be processed into information readable to the machine. Raw Data collected is almost always unclean which consists of missing attributes, noise,outliers,duplicate or wrong data. Data Preprocessing techniques are used to handle these difficulties posed by raw data. Data passed to the algorithm is known as Feature which represents the characteristics of an object or a phenomena in measurable scales. The types of features in Structured Data include :
Categorical Data - The Data whose values are taken from a predefined set or whose values always fall under a category in a set. If the Data has a particular order, it is called ordinal data For example: movie ratings out of 5. If there is no particular order, it is called Nominal Data.
Numerical Data - The features whose values are continuous or integer.
Data PreProcessing Steps:

→ Dealing with Missing Values - Sometimes eliminating the feature which consists of a large percentage of missing values can prove to be effective. However we can't apply this if many objects have missing values. If the percentage of missing values is small, we use interpolation methods to fill in the values.
[2] Interpolation: Interpolation is a model that adjusts the function to the data and extrapolates it to calculate the missing value. The most common type of interpolation is Linear Interpolation. It takes the mean of the values before the missing data and after. Missing values are easily approximated if the temporal data has a clear cut pattern. If this is not the case, the mean of the entire series is taken. Missing data can also be filled with the data around it . The mean of the previous data point and next data point can fill the missing value.
If the data is categorical, we fill the missing values with the mode of the data.

→ [1]Feature Sampling - Feature Sampling involves the selection of a subset from the dataset That represents the properties of the original dataset. This reduces the memory consumptions and the time complexity of the algorithm. *Simple Random Sampling* is commonly performed which sets the condition that there must be equal probability in choosing a sample of the dataset.Sampling without Replacement creates a bias in the dataset where the probability of selecting a sample is not equal to the others. If the dataset contains objects that vary drastically

with the others, we use *Stratified Random Sampling.* A dataset is said to be imbalanced if the number of objects of one class is significantly higher than the objects of the other classes. [3]A stratified sample divides the entire population into subgroups or strata based on the shared characteristics. These subgroups adequately represent the whole sample population in a study.A simple version of stratified sampling is to pick an equal number of objects from each class irrespective of the size of the class in the dataset. In proportionate Stratified Sampling, the sample size is proportionate to the population size. One of the disadvantages of this is method is that sometimes one entity can fall into more than 1 stratum. This introduces a bias where those present in multiple groups are likely to be chosen while performing simple random sampling.

→ Dimensionality Reduction - Most Real World problems have a large number of features that describe an entity especially in Image Processing problems. When the number of dimensions increases, the number planes occupied by the data increases which increases the sparsity to the data (More empty space). This becomes harder to visualize and model. This is called the *Curse Of Dimensionality.* [4] As there is an increase in the number of dimensions, the number of entries in the feature vector increases that represents each observation in a Euclidean Space. We measure distance in a vector space using the Euclidean distance.

$$d(p,q) = \sqrt{\Sigma^{n}_{i=1}(p_i - q_i)^2}$$

*Where p and q are 2 feature vectors.*

If new features are added, the distance increases which means that the feature space becomes sparse or emptier. Curse of Dimensionality is the main cause for overfitting while performing K Nearest Neighbours. Dimensionality Reduction is used to address this problem.

Principal Component Analysis(PCA) -

[5]After preprocessing the data , we perform Principal Component Analysis on the features values to reduce the dimensions. The idea of PCA is to reduce the number of dimensions of the dataset correlated to each other while preserving the variations in the dataset to a maximum extent.In short PCA is nothing but an eigenvalue method to reduce while preserving important information. The principal components are nothing but the measures of variations in the data.Basically,Principal Components are given by an orthogonal linear transformation of a set of variables optimizing a certain algebraic criterion.The two main steps required in finding out the principal components are to subtract the mean from the data point and to find the covariance matrix i.e the relation between the dimensions.The Variance of the data is defined as the deviation of each and every term from it's arithmetic mean.Co-variance is nothing but the variance that is taken with respect to multiple dimensions.

$$Cov(X,Y) = (\Sigma_{i=1}^{n}(X_i - X')(Y_i - Y'))/(n-1)$$

*Where X' is the arithmetic mean of data X Y' is the arithmetic mean of data Y and n is the number of observations*

On multiplication of any random vector to this matrix, we get a vector of greater magnitude with direction turned towards the vector with highest variation that is principal component 1 or PC1.The Principal Components are highly sensitive to the units of measurements which is a major drawback in PCA on covariance matrices.Further multiplication with random vectors reaches a point where the direction of the vector does not change.This vector is known as an eigenvector. An Eigenvector of a matrix A is a vector when multiplied by A returns a vector which is a scalar multiple of itself.

$$Av = \lambda v$$

Covariance matrices are symmetric and symmetric matrices have orthogonal vectors. Therefore PCA always leads to orthogonal components.Next, we project these eigenvectors onto our new dimensions, that is the reduced dimensions(n). The first step involved in this is to center our data around the mean i.e to subtract the mean from each datapoint. The first n eigenvectors are chosen and the dot product of these vectors with the transposed center data gives the projection and results in an n dimensional vector which is the PCA'd vector of the original dataset.

[6]Singular Value Decomposition - It is a matrix factorization technique that decomposes a matrix into 3 matrices.

$$A = USV^T$$

*Where S is a diagonal matrix of singular values. This represents the rank of the matrix A.*

*A is a matrix of size m x n . U is a matrix of size m x r and V is a matrix of size n x r.*

We can think of U as a matrix representing the similarity of the rows of the data with the concept the data is trying to represent. V can be thought of the similarity of columns and the concept. For example, if we consider the rows of the matrix A as users ratings and the columns of A as the movies. The concept would be either romantic or scifi. Based on the user ratings, we can figure out the concepts that are mostly represented. SVD represents the best axis to project the data on. By 'best', we mean the projection that results in the minimum sum of squares of the errors of all the movie ratings. S represents the variance('spread') on the projection axis. As we saw in PCA, the value with the most variance represents the dataset the most. Therefore, we replace the element of the least variance from matrix S and it's corresponding rows in U and V with zero. Next we multiply, all the resultant matrices. Let's say this matrix is B. The Frobenius distance between the two results in a very small value.

$$||A-B||_F = \sqrt{\Sigma_{ij}} (A_{ij} - B_{ij})^2$$

## What are the difficulties involved in handling text data in particular?

[7] Text analysis allows us to extract information such as sentiment, intentions, reviews and responses from documents,tweets, reddit etc. The most common application is the sentiment analysis of tweets, language translation, etc. To perform tasks related to text data, the machine is required to understand the emotion and the hidden meaning behind the text. Each word from the given text is tagged, this helps the algorithm to identify the relations and associations between

the words. A popular application of text analysis is priority detection . The machine must be able to identify which text must be prioritised over another. A common problem while handling similar applications may be the ambiguity in detecting sarcasm, idioms or slang language in the text. This may result in the algorithm in giving a completely different output. Speech to text translation can face difficulty in translating homophones. It also needs to take into account various different accents of the user. While performing sentiment analysis there might come a situation where the text has mixed emotions. This is where we use Attention Mechanism with LSTMs to consider only the important features.

[8] Difficulties in Sentiment Analysis - Sentiment Analysis looks just like a simple text classification problem from the outside but there are many difficulties faced while addressing this problem statement. Sentiment analysis has 3 broad categories namely, positive,negative and neutral sentiments. Sarcastic comments are usually positive comments but with negative sentiment.These comments make it very difficult for the algorithm to classify the text without understanding the context of the situation or the environment. Sarcasm is also sometimes hard for humans to understand it . The most common type of sarcasm was found to be numeric sarcasm found in a lot of social media platforms. These are the types of conversations where numerical values can affect text polarity. Example :

*We drove slowly ,only 20kmph* (non sarcastic)

*We drove slowly , only 160 kmph* (sarcastic)

[9] Neural Network architectures, such as CNN , DNN and RNN have shown excellent capabilities. A sarcastic text can be considered elementally as a sequence of text signals or word combinations. RNN is a perfect fit for modelling temporal text signals as it includes a memory component that stores contextual information directly into the model. CNNs have the ability to reduce frequency variations in the input and map the input features into robust features which are fed into the LSTM. The main disadvantage of CNN is that it has only fixed filter widths and is not suitable to handle text with different lengths. CNNs can catch patterns in temporal text with shorter lengths only. Element wise multiplication is performed between the input matrix and filter to produce composite features to be passed into the LSTM model. The sigmoid activation function is used to ensure non linearity. RNN includes a memory component which allows the model to store temporal contextual information. If the gap between two time steps becomes large, we use LSTMs that define a memory cell and a set of gates. LSTMs do not suffer from vanishing or exploding gradients while performing backpropagation. The output of the LSTM layer is passed to the Deep Neural Network layer which produces a high order feature set. This feature set can be easily categorized into the required classes.

[8] Negation Detection - Negation is a way of reversing the meaning of a sentence or a phrase. Negation Detection can prove to be difficult as it is necessary to figure out the range of words that are affected by this negation word which isn’t fixed. Negation words include ‘not’ , ‘diss’, ‘less’. Most sentiment analysis techniques negate all the words that occur after the negation

word. This method will not work if the sentence does not have any negation word but conveys negative meaning . Example :

'*If this kind of behaviour continues, this would be the last time you go.'*

Various LSTM models outperform other models in detecting negative emotions.

[10] The method described in this paper aims at extracting and analysing data about a specific topic from social media networks. It performs sentiment analysis by combining the unigram model with the Word Sense Disambiguation(WSD) Technique on bigram. WSD is a technique in finding out which meaning of the word is to be considered given the context of the sentence. It is usually applied in Machine Translation, Information Retrieval and Text Mining. The text analysis process includes text tokenization, parts of speech tagging (POS),lemmatization and finally the sentiment scores of the words that are -1,0 or 1 for negative, neutral or positive sentiment. The algorithm worked poorly while handling negative data reaching a 64.4 % . Therefore the Negation Handling technique was introduced. Bag Of Words is an algorithm that takes into account only the multiplicity of the words. Due to this simplicity, it fails to handle the negation words in a sentence. A possible approach may be to use dependencies or grammatical relations among words. This kind of parser produces a list of relations between word pairs. However sometimes a negative word in a sentence may not result in a negative sentiment . For example:

*I do not hate my enemies*

This will be classified as a negative sentiment even if it is a positive one. This paper proposes to build a dependency parse tree with both grammatical relations and the order of appearance of terms. The algorithm builds this tree and explores it to find the negation words using the DFS approach. It is assumed that the negative words only affect the nearby words of the same clause. If the negation word is the tree node, the algorithm inverts the polarity of the sibling nodes as they belong to the same clause. The sentiment score of the negation word is 0 itself. The WDS along with the Negation Handling approach gave an improved accuracy of 67%.

## Basic steps involved in preprocessing/ cleaning of text data.

[11]Before we pass our data into any model, it is necessary we first preprocess the data. Data preprocessing when it comes to text has many stages. The 3 main important tasks to be performed for data preprocessing are -

Tokenization - Tokenization is the process of splitting sentences or longer strings precisely into words. We use the inbuilt tokenize( ) from the nltk package. It may seem like we could accomplish this task manually by splitting the string based on punctuations but there are many examples where this approach will not work. This approach would give bad results when we have to figure out all the sentences from a document. For example :

*Dr.Ford asked the name of Mr.Smith's dog.*

Note that *Dr.Ford* and *Mr.Smith* are two entities where Dr and Ford would have been separated by using traditional methods.

Normalization - Normalization of text refers to tasks where the text is brought down to the same level. Steps include converting all the text to lower or upper case, removing punctuations, removing stop words (and,or..) according to the problem we are required to solve,converting the numbers into their word equivalents etc. Stop words do not contribute to the inner meaning of the context. For example in classification of positive or negative sentiments, these stopwords have an equal probability of being in the positive sentiment column or the negative sentiment column. Further, normalization is divided into stemming and lemmatization. [12]Stemming is the process of removing the prefixes and suffixes of words and mapping a group of words to a single stem even if it is not valid. For example :

*Playing - play*

*Plays - play*

*Played - play*

PortStemmer uses *SuffixStripping* for stemming. The reason why PortStemmer does not often output English words is because it does not keep a lookup table for actual stems of words but uses algorithmic rules to generate these stems . It decides whether or not to keep the suffixes through these rules. PortStemmer is known for its simplicity and speed. LancasterStemmer is an iterative algorithm with rules specified externally. A table contains about 120 rules indexed by the last letter of the suffix of a word. Based on this last letter, it decides the rule i.e whether to delete the letter or to replace it. Heavy stemming during the iterations may cause the algorithm to over - stem i.e the words might seize to have any meaning.

[11] Lemmatization is similar to Stemming but this algorithm is able to catch the canonical form of the word. For example *Better* would result in *good* on lemmatization. Lemmatization ensures that the root word belongs to the English vocabulary. You need to specify the context in which you want to lemmatize.

Another method that is necessary for pre processing is the removal of contractions. For example:

*What's* is replaced with *What is*

*You're* is replaced with *You are*

Noise removal - Usually , we obtain our textual dataset by web scraping from websites using tools such as BeautifulSoup. To extract the exact data that we need, we must remove the text file header, footers, HTML.XML, metadata etc. We might need to extract valuable data that is in the JSON format .

[13]Parts Of Speech Tagging - POS tagging is a mechanism used for labelling each word in a sentence with it's appropriate part- of - speech. Rule- based POS tagging uses dictionaries to get the possible tags for a word. If the word contains more than one tag, the algorithm considers the features of the preceding tag as well as the following one to assign the appropriate tag for the

word. For example : If the preceding word is an article, the following word must be a noun.The rules in Rule Based tagging are manually written which are limited to around 1000 rules.
Stochastic POS tagging performs tagging by calculating the probability of a tag occurring. It might consider the most frequent tag of the corresponding word in the train corpus for unseen or ambiguous situations in the test corpus.This is called Word Frequency Approach. Another approach would be to assign the best tag to a word by calculating the probability of the tag with *n* previous tags.
Transformation - Based Tagging shows similarity between Rule Based as well as Stochastic tagging. It is a rule based algorithm for automatic assignment of POS tags to a word.It also uses rules that decides what tags need to be assigned to a word and these rules are derived by ML models from the data.
Hidden Markov Model for POS tagging - The idea of this algorithm is to find the sequence of tags which most likely generated the sequence of words. We can model this POS process by using HMM where we specify the tags as hidden states that produced the words. We need to maximize the probability of a tag sequence given a word sequence.

## Advanced text processing steps.

[14]Ngram Model - Ngram Model is a simple model that assigns probabilities to a given sequence of words or sentences. Ngram is nothing but a sequence of N words. Bi-Gram is a sequence of two words while tri- gram is a sequence of 3 words.
P(w|h) is the probability of a word w given it's history h . For example :

$$P(the|It\ rained\ so\ hard\ that)$$

To find this, we will need to count the number of times *'It rained so hard that'* and the number of times '*the'* followed it. This becomes practically impossible and infeasible to compute in a large training corpus. Therefore we use the Bi-Gram approach where we calculate the probability of a given word with respect to its history by just considering the previous word. This assumption is known as the Markov Assumption i.e :

$$P(W_n \mid W_1^{n-1}) = P(W_n \mid W_{n-1})$$

To calculate this probability, we use Maximum Likelihood Estimation. We get the n grams count from the corpus and normalize the values so that it lies between 0 and 1. For example, if we need to find the probability of a word *y* given the previous word *x*, we need to count all the bigrams *C(xy)* and normalize it by the sum of all the bigrams that share the same first word.
Markov models make an assumption that we can predict into the future without looking far into the past. Any training corpus is limited therefore acceptable English sequences might be missing. This might result in the N gram model giving a zero probability. The N gram model is highly dependent on the training corpus. The accuracy can change if we convert bi gram into tri gram.

[15]Term Frequency - The main purpose of Term Frequency and Inverse Document Frequency was for document extraction and information retrieval. It's proportionate to the number of times a key word occurs in the document as well as the number of documents containing this word. This ensures that stop words are not considered as they do not add meaning to the document. It helps us associate the words of a document with its relevance to that document. Similar words will have similar relevance to a document which is what our algorithm needs to find out. Tf-IDf approach can help in extracting relevant keywords from a document. The higher the tf-idf score,the more weight it carries. Term Frequency is the raw count of the instances that occur in a document. The frequency of this can be adjusted. Inverse Document Frequency is how common or rare a word is in a document.The closer the value is to 0,more common the word is. The metric can be calculated by taking the total number of documents and dividing it by the number of documents that contain that word and then applying logarithm to that. Multiplying these two frequencies, we get the TF-IDF score of the word in the document.

$$tf(t,d,D) = tf(t,d).idf(t,D)$$

[16]Bag Of Words - Textual data is usually messy and machine learning algorithms usually prefer fixed length inputs. We cannot send in raw text directly to the model, therefore we will need to convert the text into a vector of numbers. This is called feature encoding. Bag-Of-Words is a technique for extracting features from a text. It's a representation that gives the occurrence of words in a document. It involves the vocabulary of the given words and how often they occur in the document. The algorithm does not take into consideration the location of the word in a document but only the histogram of words within a text i.e the word count in the text. The Bernoulli Document model represents documents as Bag Of Words using Naive Bayes assumption. A document is represented by a feature vector with binary elements taking the value 1 if the corresponding word is present in the document or 0 if it is not. Example of Bag Of Words-

*Sentence 1 : The cat sat on the hat*

*Sentence 2: The dog ate the hat*

Vocabulary :{ *The,cat,sat,on,hat,dog,ate*}

The encoding for sentence 1 is {2,1,1,1,1,0,0}

Sentence 2 is {2,0,0,0,1,1,1}

In computer vision, the bag of visual words is the vector of occurrence counts of a vocabulary of local image features.

[17]Continuous Bag Of Words - This algorithm tends to predict a probability of a word given a context which may be a single word or a group of words. If we have a corpus C , we first need to construct a matrix with the one hot encoding of each word. The matrix is sent into a shallow neural network with an input layer, hidden layer and an output layer. If we have 3 context

vectors for a single target word, we will have 3 initial hidden activations which are then averaged element-wise to obtain the final activation.

$$p(wo|wi) = \exp(v'^{T}_{wo} | v_{wi}) / (\Sigma_{w=1}^{W} \exp(v'^{T}_{w} v_{wi})$$

$$CBOW = -\log(p(wo/wi))$$

wo : output words

wi : context words

CBOW takes the average of the context of words. For ex: Apple can either be a fruit or a company. CBOW takes the average of these two contexts and places it in between a cluster for both fruits and companies.

[18]Skip Gram Model - The skip gram model achieves the reversal of the CBOW algorithm. It predicts the context vector i.e the surrounding text given a target variable( center text). For example this model aims to predict the context *[quick,fox]* given the word *[brown]* or *[the, brown]* given the target word *[quick]*. We feed the model with a pair of words (X,Y) where X is the input target word and Y is the context. We feed in ([target,context],1) to specify a positive label which means that the target and the context are related . The negative example passed is in the form ([target,context],0) which signifies that the pair of words are irrelevant. This makes sure we get the  similar word embeddings for semantically similar words.

Word2Vec is the most popular method of learning features from text and transforming them to their associated embeddings. It can use either the CBOW approach or the Skip Gram approach. It was found that the Skip Gram approach showed better results for small amounts of data and the CBOW approach works well for more frequent words.

## REFERENCES:


1. https://towardsdatascience.com/data-preprocessing-concepts-fa946d11c825

2.https://leportella.com/missing-data.html#:~:text=Interpolation%20is%20a%20mathematical%20method,data%20and%20the%20value%20after.

3.https://www.investopedia.com/terms/stratified_random_sampling.asp

4.https://builtin.com/data-science/curse-dimensionality

5.https://www.researchgate.net/publication/316652806_Principal_Component_Analysis

6.https://www.analyticsvidhya.com/blog/2019/08/5-applications-singular-value-decomposition-svd-data-science/

7.https://monkeylearn.com/text-analysis/

8.https://www.toptal.com/deep-learning/4-sentiment-analysis-accuracy-traps

9.https://www.aclweb.org/anthology/W16-0425.pdf

10.https://www.computer.org/csdl/pds/api/csdl/proceedings/download-article/12OmNrY3LAH/pdf

11.https://www.kdnuggets.com/2017/12/general-approach-preprocessing-text-data.html

12.https://www.datacamp.com/community/tutorials/stemming-lemmatization-python?utm_source=adwords_ppc&utm_campaignid=1455363063&utm_adgroupid=65083631748&utm_device=c&utm_keyword=&utm_matchtype=b&utm_network=g&utm_adpostion=&utm_creative=332602034358&utm_targetid=aud-299261629574:dsa-429603003980&utm_loc_interest_ms=&utm_loc_physical_ms=9062013&gclid=CjwKCAjwm_P5BRAhEiwAwRzSO-CBVwrN8ZfxDRF40zZCiPiriE1Ha0p-OfhRAhLwb_QbephP6g9iYxoCziQQAvD_BwE

13.https://www.tutorialspoint.com/natural_language_processing/natural_language_processing_part_of_speech_tagging.htm#:~:text=In%20simple%20words%2C%20we%20can,conjunction%20and%20their%20sub%2Dcategories.

14.https://towardsdatascience.com/introduction-to-language-models-n-gram-e323081503d9

15.https://monkeylearn.com/blog/what-is-tf-idf/

16.https://machinelearningmastery.com/gentle-introduction-bag-words-model/

17.https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/

18.https://www.kdnuggets.com/2018/04/implementing-deep-learning-methods-feature-engineering-text-data-skip-gram.html#:~:text=The%20Skip%2Dgram%20model%20architecture,jumps%20over%20the%20lazy%20dog%E2%80%9D.