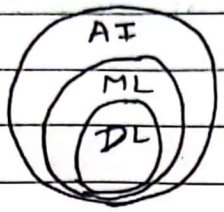
Deep Learning.

Date:
Page No.:

Day 56 Introduction to ANN.

1. Artificial Neural Network. (ANN)

- Deep learning is subset of AI.



Q What is AI ?

→ AI is anything where you are using the machines to solve the complex problem

ANN :-

There networks are the subsets of machine learning and they are designed the way human brain.

How human learn day to day life.

- At every step or iteration we try to reduce the error which means we are learning from the previous step and model keeps on getting better and better.

- ANN we will iterate the process to get the best Model.

Date:
Page No.:

Q Can human brain solve the Complex prm?
→ YES.

- In ANN we will use the Raw data

* Usages / Advantages of Neural Networks

- Used in NLP, Image recognition, Computer Vision
- very general & adaptive
- can work directly on raw data

* Drawbacks of Neural Networks

- High Memory intencive operation Computational and time demanding

- Tuning of hyper parameters is a very tricky task.

- Its a kind of grey box, difficult to understand how the decision was arrived at.

* ANN define.

ANN is a computing system made up of numb of simple highly interconnected processing elemen process information by their dynamic state response to external inputs

- In

Date:
Page No.:

- In NN you cannot interpret the model, you cannot tell why a particular decision/pred was done. That's why this part cannot be answered by the neural network.

Day 57 Gradient Descent.

- An artificial neuron is a mathematical function based on the model of biological neurons where each neurons takes the inputs, weight them seperately, does the summation and this summation is passed through a non linear function (activation function) to produce output. Another name of neuron is perceptron.

* key Features and assumptions

- Deep learning is a branch of machine learning based on the set of algorithms that attempts to model high level and hierarchical representation using multiple processing layers.

Main assumption in DL

1. Everything needed to solve the problem can be learned From the data.

2. Employing a large number of very simple computational units, we can solve even the complex problem.

Date:
Page No.:

* Common Terms used in Neural Network.

1. Convergence
2. Cost Function
3. Gradient Function
4. Stochastic gradient descent
5. Learning Rate
6. Learning Rate decay
7. OverFitting (Dropout), & Cross Validation.

1) Convergence :-

Q My model gets converged ??
→ It means you got the best model

Q What would be the convergence criteria ??
→ 1. Desired level of accuracy - Suppose 85%
2 Iterations

2) Gradient Descent :-

1. GD is an optimization algorithm.
2.

imp * Cost Function is used to capture the amount of error in your model in any given time.

Equilibrium Descent is an optimization algorithm not a cost Function.

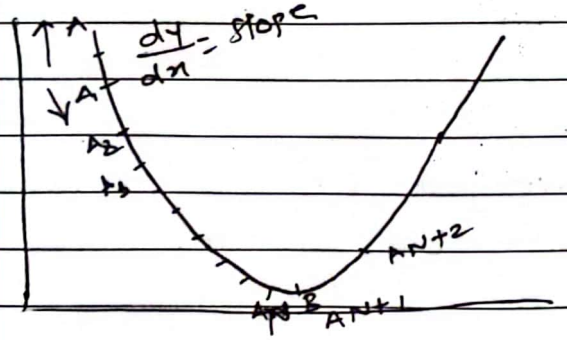
Date:
Page No.:

Optimisation algo will help us to reach the point or scenario where cost is least cost, loss or error are same only.

Cost Function is MSE / it is square function or square Fuction would always be 1

When $x = -1$

$x = 1$

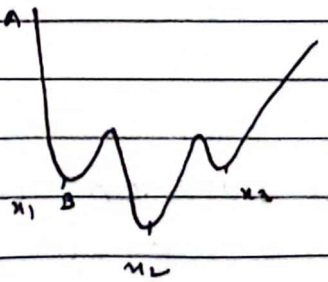


$N = 50$.

Day 58 LR and LR Decay

3) Stochastic Gradient Descent (SGD)

Local Gradient
Global Gradient.



Date:
Page No.:

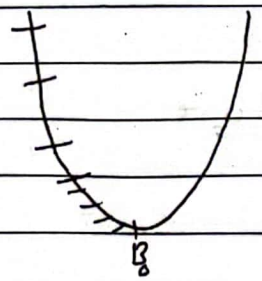
4) Learning Rate :-

- It's how fast or slow you can learn things

- If LR is high, model will coverageed in Less time and if LR is low then model will take more time to convergeed.

5) LR Decay -

Using LR decay we can overcome the disadvantages associated with High LR or Low LR.



LR Decay

LR Disadvantage
↑ Time consuming & cost eff

Day 59 Overfitting and Cross Validation.

6) Overfitting

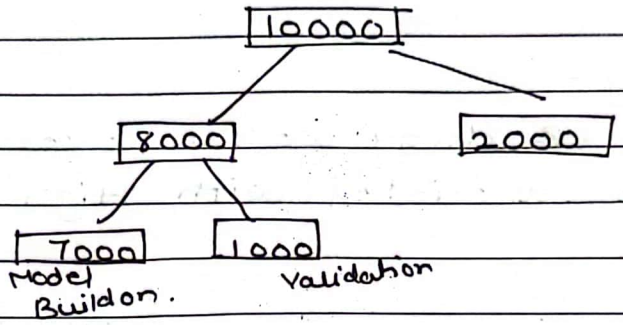
Two ways to overcome the problem of overfitting

1. Drop out
2. Cross validation.

Date:
Page No.:

2. Cross Validation :- It is a Methedology or Terminology to overcome the overFitting Technique

* Lets say I have 10000 Records, so here I will do the 80-20 sampling.

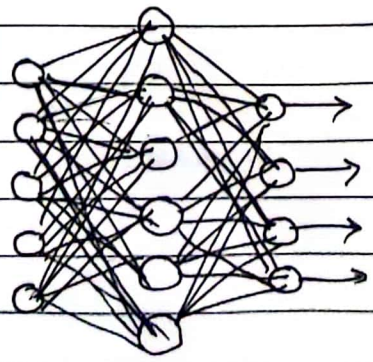


Day 60 Neural Network Structure

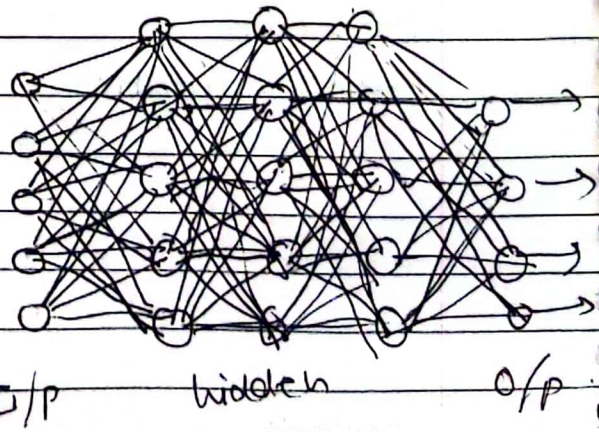
* Neural Network :-

In any NN or NN will have at least 3 layers
Those 3 layers will be 1. Input layer
2. Hidden layer
3. O/p layer

Simple Neural Networks



Deep Neural Networks



penkraft

$\Sigma - p \log_2(p)$

Date:
Page No.:

1. Number of hidden layer in NN is a hyperparameter.

2. Number of neuron in each hidden layer is also a hyperparameter.

3. Number of neuron in O/p & I/p layer is not hyperparameter.

Depending upon the data the no. of neuron in O/p and i/p layer are Fixed.
- No of neurons in I/p layer is equal to number of x-variables
- No of neurons in O/p layer is mapped to no. of classes in your target variable.

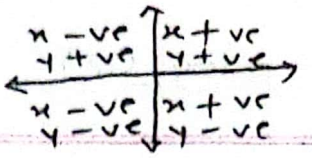
Day 61 Activation Function.

- Each neuron in hidden layer has its own activation Function.

- We are continuosly getting the info but the differentiate between significant and insignificant info we need to use the Activation Function.

1. ReLu (Rectified linear unit)

- Relu is the most important and highly used Activation Fuction

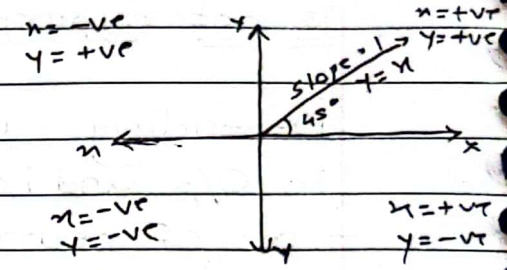


Date:
Page No.:

- It is a non linear function.

- Info continuously getting can be X
- Whether it is significant or insignificant can be taken as Y.

normally 'Y' is depend on 'X'



eg:- Eq of straight line is,

$$Y = mx + C$$

For green line slope = 1

$$Y = x + C$$

Since line is passing through the origin

$$C = 0$$

then your eqn becomes,

$$Y = X$$

Equation of green line is $Y = X$.

In the diagram,
If x is less than 0 then also Y=0
Y is nothing but output,
If y is less than or equal to 0 the no o/p is generated that means this info is insignificant

for green line,
x>0 then o/p is also equal to x
$Y = X$
here output is equal to input

Date:
Page No.:

If you are getting the o/p that mean info is significant.

- If we combine both green & red line then it is nothing but ReLu AF
- It is used to differentiate whether the info is significant or insignificant.

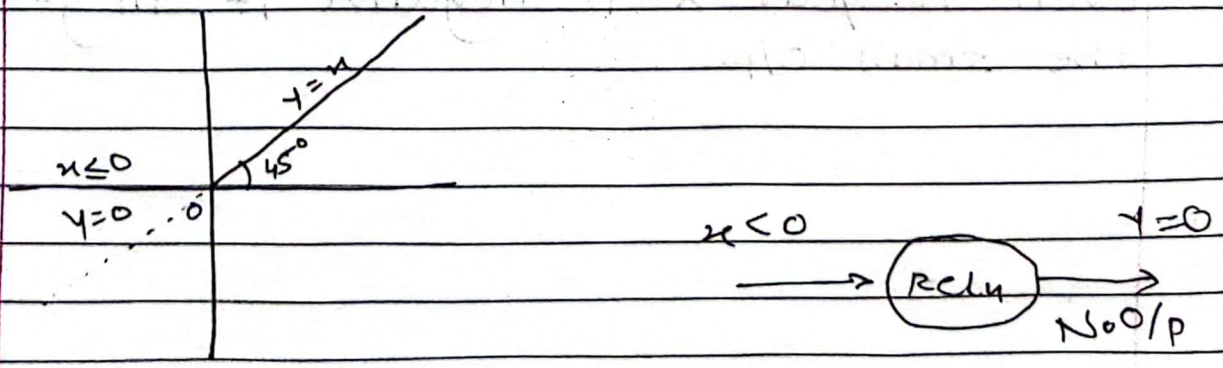
Day 62 Activation Function Part - 2

* Leaky Relu.

- It is also a activation Function,
- In Leaky ReLu 'x' should be anything but 'Y' is always '0'

- ~~Advantage~~

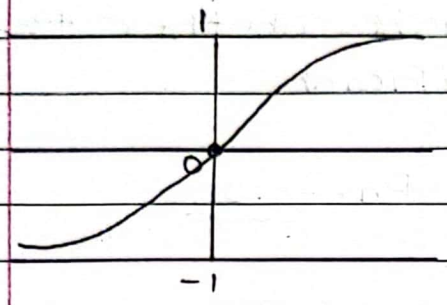
- Leaky relu are one attempt to Fix the 'Dying ReLu' problem by having the small negative slope.



Date:
Page No.:

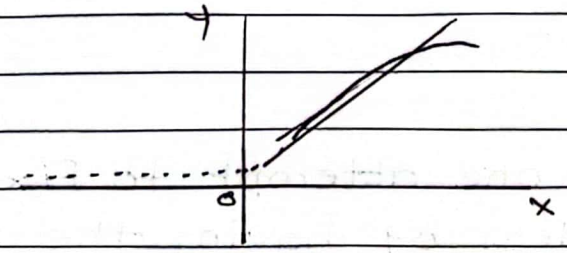
* — Tan.h

- also know as Sigmoid. Function.
- It is an Hyperbolic Function.



* SoftMax.

- It is an Activation Function which we will be using for neurons in O/p layer



- Even if your x is negative it will generate the small O/p.

Neural Nets playground.

Date:
Page No.:

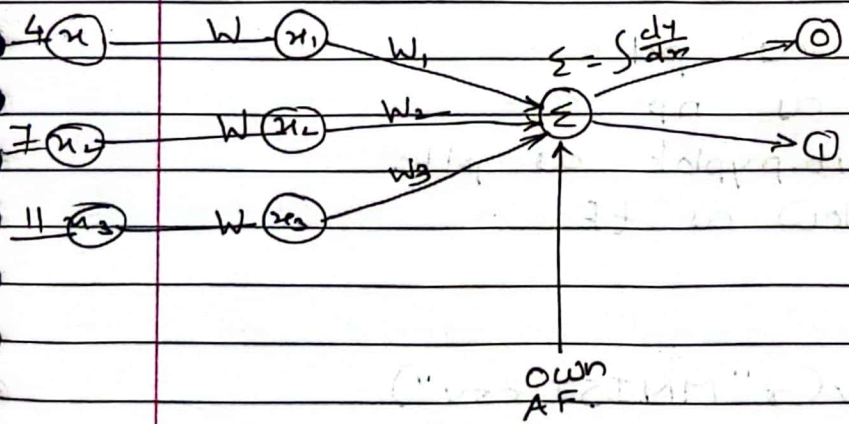
- For neural network we are using the Frame work.

1. Tensorflow
   ---> It is developed by the Google.
2. Pytorch
   ---> Developed by Facebook.
3. Mnet
4. H2O.

Day 63 How Neural Networks Actually works ! 1.

Left to Right.

1) I/p to O/p Forward propagation
2) Error are Backward propagation
   ↑
   ~~left~~ to ~~right~~
   Right left



$W_1 = 0.01$
$W_2 = 0.0002$
$W_3 = .012$

$x_1 W_1, x_2 W_2, x_3 W_3$

$\sum w_i x_i$

$y = mx + c$

$= w_i x_i + \beta_{..}(Bias)$

eqn +ve or -ve

- each input x weight (W are randomly assign)

  $W = 0.001$

- To find important parameter

- The eqn showed pass through AF (ReLu)

Date:
Page No.:

- errors are come because of (W).

$$\frac{dE}{dw} + \left\{ NW = OW + \frac{dE}{dw} \times LR \right\}$$

- $\frac{dE}{dw}$ are get by using Gradient descent is worked in background

Batch size depend on the total size

Day 64 ANN on Mnist data

eg:- pip install tensorflow
eg:- pip install keras.

```
eg:- import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt.
     import tensorflow as tf
     import keras.
```

```
eg:- mn = pd.read_csv(r"MNIST.csv")
eg:- mn.head()
```

```
eg:- mn.label.value_counts()
```

```
# Lets try to do coding in 2 parts
# plot to understand this is image data
```

```
# Build the neural net
```

penkraft

Date:
Page No.:

eg:- mnist1 = mn.iloc[:, 1::]
abcd = mnist1.iloc[0]
abcd = np.array(abcd)
abcd = abcd.reshape(28,28)
plt.imshow(abcd)

to see the image of the first 30 records.

eg:- mnist1 = mn.iloc[:, 1:786]
mnist1 = np.array(mnist1)
for i in range(30):
    plt.subplot(6,5,i+1)
    plt.imshow(mnist1[i,:].reshape(28,28))
    plt.axis('OFF')

Now everyone is agreed this is image data.

eg:- mn.isnull().sum()

Sampling

eg:- from sklearn.model_selection import train_test_spli

eg:- mn_train, mn_test = train_test_split(mn, test_size=0.2)

eg:- mn_train_x = mn_train.iloc[:, 1::]
mn_train_y = mn_train.iloc[:, 0]

mn_test_x = mn_test.iloc[:, 1::]
mn_test_y = mn_test.iloc[:, 0]

Date:
Page No.:

eg:- mn_train_x = tf.keras.utils.normalize(mn_train_x)
mn_test_x = tf.keras.utils.normalize(mn_test_x)

this step is optional but recommended

eg:- model = tf.keras.Sequential()
model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
model.add(tf.keras.layers.Dense(128, activation = tf.nn.relu))
model.add(tf.keras.layers.Dense(10, activation = tf.nn.softmax))

model.compile(optimizer = 'adam', loss = 'sparse_categorical_crossentropy', metrics = ['accuracy'])

eg:- mn_train_x = np.array(mn_train_x)
mn_train_y = np.array(mn_train_y)

eg:- model.fit(mn_train_x, mn_train_y, epochs = 12, batch_size = 128)

eg:- pred_values = model.predict(mn_test_x)
pred_values

probality value are show because of using the softmax.

Date:
Page No.:

eg:- np.sum(pred_values[0])

eg:- pred_classes = np.argmax(pred_values, axis=1)
pred_classes

eg:- From sklearn.metrics import confusion_matrix
eg:- accuracy_score(mn_test_y, pred_classes) * 100

Day 65 Cross Validation on MNIST data.

only check the accuracy.

Date:
Page No.:

Day 66 Introduction to CNN.

- CNN is nothing but Convolutional Neural Network.

- CNN's can only be used while Image classification.

- CNN has to be used only For Image data and Nothing else.

- Data Compression.

- CNN are used For Image Classification.

- CNN's we are doing the data compression

CNN:-

- CNN are consists of 3 layers
  1. Convolutional layer
  2. Pooling layer
  3. Fully connected layer.

- Main purpose of this layer is to extract the Features from the I/P Image
- It preserves the spatial relationship b/w pixels by learning image Features using small squares of I/P data.
- Here a matrix (filter) is slided over the image and computing the dot product and this is called (Convolutional Feature or the activation map or the the feature map.)

penkraft

Date:
Page No.:

Day 67 CNN Part:-2.

1. Convdiution only apply on original data (image)
2. Pooling layer.

After convolutional layer only then we will apply the pooling layers.

- Pooling layer cannot be performed or apply directly on the original image.

There are two type of pooling layer

1. Max pooling
2. Average pooling

I will consider the max pooling with size of 2×2.

3. Fully Connected layer

- it is nothing but the each neuron of previous layer is connected to the all the neuron in the next layer

- No of in this layer is hyperparameter.

- The size of pooling layer and whether to do the max or average pooling is a hyperparameter

- The size of filter or slider is a hyperparameter

penkraft

Date:
Page No.:

Day 68 CNN model on Catalog Data.

eg:-
```
import os
import cv2
import matplotlib.pyplot as plt.
import numpy as np
import random.
```

eg:-
```
path = r"C:\Users\om\OneDrive\Desktop\Dog and Cat
cate = ["Cat", "Dog"]
```

eg:-
```
For i in cate:
    Folders = os.path.join(path, i)
    print(Folders)
```

```
# Here we are joining the two path so that
it will be easy for us to read the images
# We have just entered into the folders bt
we are not reading images
```

eg:-
```
For i in cate:
    Folders = os.path.join(path, i)
    for image in os.listdir(Folders):
        image_path = os.path.join(Folders, image)
        print(image_path)
```

eg:-
```
For i in cate:
    Folders = os.path.join(path, i)
    for image in os.listdir(Folders):
        image_path = os.path.join(Folders, image)
        image_array = cv2.imread(image_path)
        plt.imshow(image_array)
        break.
```

penkraft

Date:
Page No.:

eg:- image_size = []

eg:- input_image = []

```
For i in cate:
    Folders = os.path.join(path, i)
    label = cate.index(i)
    print("Value of label is ", label)
    for image in os.listdir(Folders):
        image_path = os.path.join(Folders, image)
        image_array = cv2.imread(image_path)
        image_array = cv2.resize(image_array, (image_size
                                               , image_size))
        input_image.append([image_array, label])
```

eg:- len(input_image)

eg:- np.random.shuffle(input_image)

Till this point x & y values are together now lets seperate them.

```
eg: X = []
    Y = []
    for X_values, label in input_image:
        X.append(X_values)
        Y.append(label)
```

```
eg:- X = np.array(X)
     Y = np.array(Y)
```

eg:- X.shape

penkraft

epochs is nothing but iterations.

Date:
Page No.:

eg:- y.

eg:- x.

eg:- plt.imshow(x[700])

Day 69 CNN model on Catdog Classification

~~#eg:~~ Designing the Convolutional Neural Network.

eg:- From keras.models import Sequential
From keras.layers import Conv2D, MaxPool2D, Flatten, Dense, Dropout.

eg:- model = Sequential()

model.add(Conv2D(filters=10, kernel_size=(5,5), activation='relu', padding='same', input_size=(200,200,3)))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Flatten())

model.add(Dense(128, activation='relu'))
model.add(Dense(2, activation='softmax'))

eg:- model.compile(optimizers='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

eg:- model.fit(X, Y, epochs=20, batch_size=250)

eg:- pred_values = model.predict(X)
pred_values

Date:
Page No.:

eg:- pred_classes = np.argmax(pred_values, axis=1)
pred_classes

eg:- From sklearn.metrics import confusion_matrix

eg:- catdog_tab = confusion_matrix(Y, pred_classes)
catdog_tab

eg:- From sklearn.metrics import accuracy_score
accuracy_score(Y, pred_classes) * 100.

Day 10 Final session on CNN.

eg:- idx2 = random.randint(0, len(x))
plt.imshow(x[idx2, :])
plt.show

y_pred = model.predict(x[idx2, :].reshape(1, 200, 200,
y_pred = y_pred > 0.5
if y_pred[0][0] == True are y_pred[0][0] == False:
    print("Our Model says it is a : Cat")
else:
    print("Our Model says it is a : Dog")

Date:
Page No.:

Day 71 Introduction to RNN.

- Recurrent Neural Network
- RNN's are used where there is some kind of time component or we can say ordering matters
- RNN's can also be used for the text-analysis so whenever we do NLP there also we can use RNN's

eg:- Today is Tuesday

- RNN's has lot of drawbacks there is another model which we are going to use is LSTM algorithm.

- LSTM is Long Short Term Memory.

Theory will be discussed in 4 parts:-

1. RNN and how it works
2. Draw back of RNN.
3. LSTM and how it works
4. How LSTM handles the drawbacks of RNN

- RNN are class of neural network that creates cycles in the network graph in order to exhibit dynamic temporal behaviour. In other words RNNs involve recurrent or circular loop between the neuron where the output of network is feedback as a additional input to the network (send output back to itself) for subsequent processing

penkraft

Date:
Page No.:

- RNN are very useful when you have some time like component... what would be the next sequence.

eg:-

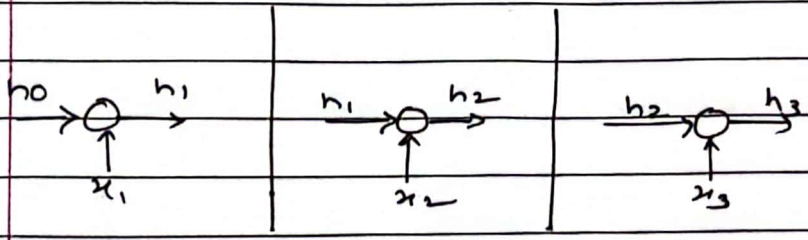
Apple is a delicious . This is is good for health also.

Day 12 Drawbacks OF RNN.

- In RNN sequencing matters
- RNN are Fully connected layers.

imp - CNN is not an Fully connected model

- RNN are Fully connected layer but with the little difference.



- In RNN at a given time there are two input
1. New input
2. O/p from the previous time cycle acts as a additional input

Date:
Page No.:

- RNN we can do the sentiment analysis.

2. Drawbacks of RNN. :-

- Mainly two issues with RNN
  1. Vanishing Gradient descent
  2. RNN do not perform good when we have the long term dependency.

1. Lets try to understand the vanishing gradient descent:-

2. long term dependancy :-

We need to check for things (Dicision) past thing.

Date:
Page No.:

Day 73 Introduction to LSTM's.

- LSTM.

Ex:

Japan as a country is aging very fast and they need a huge young workforce and there are many employment opportunity present over there. But to work there you need to know their langauge I am planning to join japanese classes.
Despite of less labor force this country is still a very large economy and highly industrialized nation and it has pioneered in automobiles & manufacturing but it is facing huge competition from south Korea Taiwan and Indonesia coz labour cost is less It is expected to grow by 2% in GDP GDP is your gross domestic product GDP is measured in USD Trillion and expected GDP of Japan is 5.5 —— Dollars

- To learn something new some past info has to be flushed out.

Bas exactly same concept pe aapka

Day 74 LSTM on Spam data.

eg:-
```
import pandas as pd.
import numpy as np
import random
import tensorFlow as tf
import Matplotlib.pyplot as plt

From sklearn.metrics import accuracy_score

From tensorflow.keras.models import Sequential
From tensorFlow.keras.layers import Flatten, Conv2D
Dense, MaxPool2D.
From tensorflow.keras.utils import to_categorical
From keras.layers import Dense, LSTM, Embedding
From tensorFlow.keras.optimizers import Adam
```

eg:-
```
SP = pd.read_csv(r"spam.csv",encoding='latin-1')
SP.
```

eg:-
```
SP.isnull().sum()
```

eg:-
```
SP.rename(columns={'v1':'label', 'v2':'Message'}
,inplace=True)
```

eg:-
```
SP.Label.value_counts()
```

eg:-
```
SP.Label.replace({'spam':1,'ham':0}, inplace=True)
```

eg:-
```
SP.Label.value_counts()
```

penkraft

Date:
Page No.:

eg:- SP.Message = SP.Message.str.lower()

eg:- SP.

Day 75 NLP imp terms

- In NLP, there will be unstructured data
- Word Boundary :- Where one word ends and another word starts (space bet^n word)
- Tokenization :- Assigning a token to each unique word
- stemming -
Mapping to the root word

eg:- Singing - Sing
Mens - Men
Running - Run

* - Term Frequency :- Inverse document Freq (TF-idF)
- Each unique word is known as Term
- Document is nothing but each sentence or no. of terms.
- Collection of document is known as "corpus"
- Frequency is how many times (count)
- TF is how many times each word is appearing.

Higher the TF more significant is the word.

Date:
Page No.:

- Disambiguation
  Context vs Content.

- Stop words
  These are words which are very Frequent in the document but they do not add any value or any meaning to the document.

eg:- is, of, and, or, this, a, an

- Speech tagging
  In this it will tell you whether it is Noun / Pronoun / Adj / Verb

- NER (Name Entity Recognition)
  Using NER we can tell if object or word is related to money / time / location / organization

eg:- He earn 1 lac. ....> Money

- Topic Model -
Discovering the hidden or abstract pattern

* TF - IDF :.
- All the stop words should be removed before calculating the term Freq
- Some times the Frequency may be very high but word is not that significant so that why we need IDF
- IDF is nothing but less Frequent but more impactful or significant

Date:
Page No.:

IDF = ln (No of total document / No of document in which that word is pres

ex:- Consider we have 1000 documents in claim document
Word Policy is appearing in all the documents
TF = 1000

Then, IDF = ln (1000/1000)
= 0

Now lets say,
Word "Cancer" is present in only 10 document
IDF = ln (1000/10)
= ln (100)
= 4.6

* TDM (Term Document Matrix)

- Each unique word becomes a column
- Value 1 will indicates the presence of that particular word while 0 indicate the absence of that word
- It is sparse matrix (more 0 values than the other values)

Day 76 NLP on Spam data.

Day 77 NLP model on Spam data Part 2

Date:
Page No.:

eg:-
```
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.max_row', None)
import numpy as np
import nltk.
```
→ `msg = pd.read_csv(r"spam1.csv", encoding = "cp1252")`
→ `msg.head()`
→ `msg = msg.iloc[:,[0,1]]`
→ `msg.head()`
→ `msg.rename(columns = {'v1':'label','v2':'Message'},inplace = True)`
→ `msg.label.replace({'spam':1,'ham':0},inplace = True)`
→ `msg.label.value_counts()`
→ `msg.head()`
Whenever we are doing NLP either we will convert data into upper case to lower case because python is case sensitive.
→ `msg.Message = msg.Message.str.lower()`
→ `msg.head()`
→ `nltk.download('stopwords')`
→ `From nltk.corpus import stopwords`
→ `stopwords.words('english')`
→ `import string`
→ `string.punctuation`
We will remove the stopwords, punctuation and then tokenization and create TDM
After this do the sampling, build the model and then do the predictions

Date:
Page No.:

```
→ def text_process(mess):
    """
    1. Remove the punctuation
    2. Remove the stopwords.
    3. Return the list of clean textwords.
    """
    nopunc = [char for char in mess if char not in
              string.punctuation]
    nopunc = ''.join(nopunc)

    return [word for word in nopunc.split() if
word not in stopwords.words('english')]
→ from sklearn.feature_extraction.text import
  CountVectorizer
# it is used to give the count for each unique word.
→ after_trans = CountVectorizer(analyzer = text_process).fit
                (msg['Message'])
→ len(after_trans.vocabulary_)
# Building the TDM.
→ tdmspamdata = after_trans.transform(msg['Message'])
→ tdmspamdata.shape
→ type(tdmspamdata)
→ tdmspamdata.toarray()[0::]
→ abc = tdmspamdata.toarray()
→ abc = pd.DataFrame(abc)
→ abc.shape
→ abc.head()
# Sampling.
→ from sklearn.model_selection import train_test_split
→ train_x, test_x, train_y, test_y = train_test_split
(tdmspamdata, msg.Label, test_size = 0.2)
```

penkraft

Date:
Page No.:

After sampling we can use any of our classification model.

```
→ from sklearn.naive_bayes import MultinomialNB
→ nb_spam = MultinomialNB()
→ nb_spam.fit(train_x, train_y)
→ pred = nb_spam.predict(test_x)
  pred.
→ from sklearn.metrics import confusion_matrix.
→ confusion_matrix(test_y, pred)
→ from sklearn.metrics import accuracy_score
→ accuracy_score(test_y, pred)
→ from wordcloud import WordCloud
→ WordCloud
→ wd_word = WordCloud(stopwords = stopwords.words
  ('english'), max_words = 15, random_state = 123).
  generate(str(msg['Message']))
→ wd_word
→ import matplotlib.pyplot as plt
→ plt.imshow(wd_word)
```

Day 78 Sentiment analysis on Trip advisor