Module - II

Regularization for Deep Learning

Data Set Augmentation (or) Data Augmentation -

- Augmentation means adds more stuff or information to given data.

Data Augmentation is a technique used in machine learning to artificially expand the size and diversity of a data set by applying transformations or modifications to the existing data.

Ex - An image can be transformed into to number of images with few transformations.

- Data augmentation mainly used in Computer vision, Speech Recognition and NLP.

Types of Data Augmentation -

1. Image Data Augmentation -

- It is used in in Computer vision to increase the diversity.

Basic Transformations -

i) flipping - Horizontal or vertical flip of images.

ii) Rotation - Rotating the images by certain degree.

iii) Cropping - Cropping the image by Randomly.

iv) scaling - Resizing the image to different sizes.

v) Brightness Adjustment - Increase or decrease brightness.

vi) Saturation - changing the colour intensity

Scanned with OKEN Scanner

2. Text data Augmentation –

– It is used in NLP to generate variety of text data.

Basic Techniques –

i) Synonym replacement – Replace the words with synonyms.

ii) Random insertion – Adding random words from the vocabulary

iii) Random deletion – Removing the Randomly words from sentence.

iv) Random Swap – Swapping the positions of the word.

v) Back Translation – Translating a sentence to another language and back.

vi) Sentence paraphrasing – Using AI models to rewrite sentences.

3. Audio Augmentation –

– It is used in speech recognition.

Basic Transformations –

i) Pitch shifting – changing the pitch of an audio sample

ii) Speed variation – slowing down or speed up speech.

iii) volume scaling – Increase or decrease the loudness.

iv) Time stretching – changing speed without effecting pitch.

v) Background noise Addition – Adding real world noise like traffic, crowd.

Adv –

1. prevent overfitting
2. Improve model performance
3. Balances Imbalanced data set

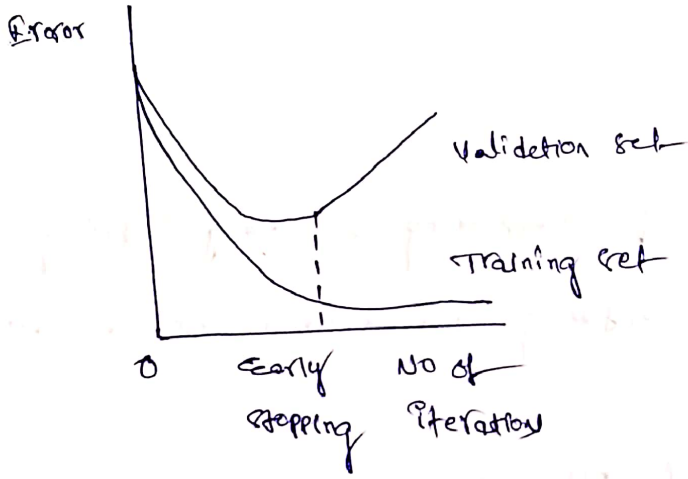
dis adv –

1. Computationally risk
2. Not always effective
3. Domain dependency

Early Stopping —

- Early stopping is a technique used in machine learning models to prevent the overfitting.
- When the model learns the training data too well and not well on new data or unseen data.
- Then it will cause best performance on training data and poor performance on validation data or testing data.

How early stopping works —



Step 1 — Training and validation sets —

- During the training the data can be divided into two sets,

i) Training set — It is used to train the model,

ii) Validation set — It is used to evaluate the working performance of the model during training.

2. Monitoring the model performance — When the model is training monitoring the model performance on both training set and validation set.

Scanned with OKEN Scanner

3. Identifying overfitting —

- Initially the model performance was good in both sets. At a certain point the model may start to perform worse on the validation set and continue the performance well on training set. This difference indicates the overfitting.

4. Stopping Training —

- With the early stopping, the training is halted when the model performance reduced on the validation set. it means the model parameters captured where the model best performs on new data.

Advantages —

1. Prevention of overfitting — early stopping helps to prevent the overfitting and enhances its generalization to unseen data.

2. Computational Efficiency — Early stopping reduces the computational resources like time and resources.

3. Automated Model Tuning — It provides an automated method to determine the optimal number of iterations and reducing the manual Tuning.

4. Improved Generalization — early stopping enhances the model ability to generalize the unseen data or new data.

5. Simple and easy implimentation method.

Scanned with OKEN Scanner

Dropout —

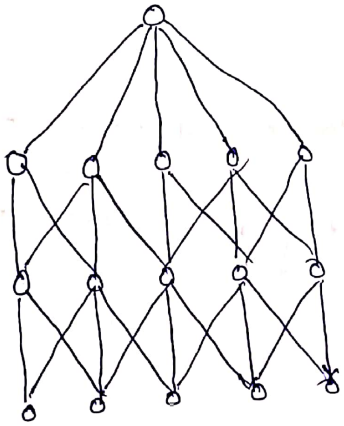
— Dropout is a technique used in neural networks used to prevent the overfitting.

When overfit occur —

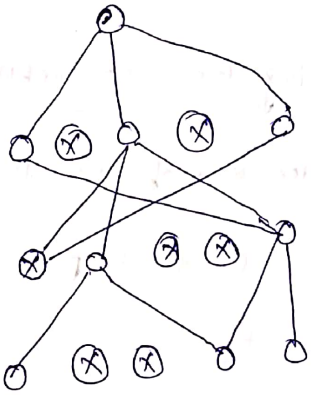
— When the model learns training data too well and not well on unseen data or new data.

— Then it will cause bad performance on training data and poor performance on validation data or testing data.

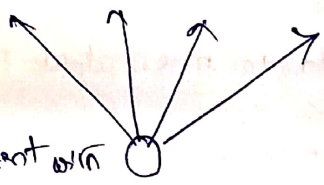
How it works —



(a) fully connected neural network

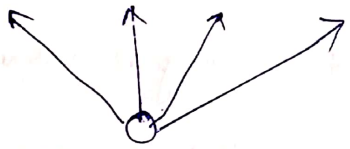


(b) After applying the dropout method



present with probability p

(c) At training time



(d) At test time

Scanned with OKEN Scanner

1. Randomly Dropping the neurons —

During training, dropout method selects the fraction of neurons randomly in the network and temporarily drops them i.e. These neurons are ignored in training pass. This is done for each iteration.

2. Combining the dropped neurons —

— When a neuron is dropped then its all incoming and outgoing connections are also removed for that iteration,

— This will creates new and smaller network from the original one.

3. Combining the sub-networks —

— At the end of the training, the model combine all these sub networks which will help to reduce the overfitting and improves the model ability.

Advantages —

1. preventing the overfitting
2. Improves the model performance
3. Efficient model averaging — At the end of the training model combining all sub networks, it will lead to more robust model.
4. Applicability to various network architecture —

— It is versatile architecture it can be applied in to various networks like CNN, RNN etc.

5. Reduced the inter-neuron dependency.

Scanned with OKEN Scanner

Parameter sharing and parameter Typing –

- Generally parameters plays an important role in machine learning models.
- When training the deep learning models managing the parameters is crucial step.
- Basically parameters are 'weight' and "bias". The model learn these parameters from the data. These parameters are helped to make the predictions.

Parameter sharing – and parameter Typing are the two technique that helps to reduce the no. of the parameters and making the models more efficient and faster.

$$y = wx + b$$

$y \rightarrow$ output
$x \rightarrow$ input
$b \rightarrow$ bias
$w \rightarrow$ weight

Advantages –

1. Reduce the memory size
2. Improves the performance of a model
3. Speed up the training process
4. Maintain the consistency in model behaviour.
5. Reducing the overfitting problems.

Scanned with OKEN Scanner

1. Parameter Sharing –

– It uses same set of parameters are used in multiple parts of the model.

– Parameter sharing mostly used in CNN.

Ex – In a CNN, an image is broken into small no. of parts.

– A single filter i.e kernal is scans the all small no. of parts of the image like (top-left, Centre, bottom-right etc)

– Instead of using different filters for each part, the same filter is used for all regions or parts of the image.

2. Parameter Typing –

– Here multiple parts containing equal weight i.e, each part contains equal weight.

– Parameter Typing is mostly used in RNN.

– In RNN, same set of weights are used for each step.

Importance of RNN –

– RNN provides time series data at each step, at time.

– At each time step, the same weight matrix is used for transforming the inputs.

– maintain consistency across different time steps.

Scanned with OKEN Scanner

Ex -

```
import torch
import torch.nn as nn
class SimpleCNN(nn.Module)
  def __init__(self)
  super(SimpleCNN, self).__init__()
self.conv = nn.Conv2d(in_channels=1, out_channels=1,
                      kernel_size=3)
  def forward(self, x)
    return self.conv(x)
image = torch.rand(1,1,5,5)
model = SimpleCNN()
output = model(image)
print("Input shape:", image.shape)
print("output shape:", output.shape)
```

output -

```
Input shape : torch.Size([1,1,5,5])
output shape : torch.Size([1,1,3,3])
```

Adversarial Training -

- Adversarial Training is a technique used in machine learning models to make stronger against the tricky attacks.
- These attacks use invisible changes in the initial input data to fool the model into making the wrong predictions.

Scanned with OKEN Scanner

Ex- Imagine a model that recognizes animals. If we change a few pixels in a cat image (so small that human can't notice) the model might mistakenly think it's a dog.

How Adversarial work —

1. We create these tricky examples like Adversarial example.
2. We train the model using both normal and tricky examples.
3. The model learns to ignore the tiny changes and make correct predictions.

Type of Adversarial Attacks —

i) FGSM (Fast gradient sign method) —

It is a quick way to add tiny changes to fool the model.

ii) PGD (projected Gradient decent) —

It is very strong multi attack to add small or minor changes.

iii) CEW (Carlini & Wagner) —

- It is very advanced trick to fool the machine learning models.

Adversarial Training Techniques —

To protect models we use some defensive techniques.

i) Basic Adversarial training —

This technique train the model on both normal and tricky examples.

ii) PGD Based training —

- It is advanced method to train the model using stronger attacks.

Scanned with OKEN Scanner

iii) TRADES - It is a method that balances accuracy and robustness

Challenges in Adversarial Training -

1. Takes more time
2. Accuracy drops
3. Attackers keep improving

Importance of Adversarial Training -

1. Cyber Security - It protects AI systems from hackers.
2. Self-driving Cars - Stop small changes in traffic signs from fooling the car.
3. Facial Recognition - It prevents the attackers from tricking security systems.

Parameter Initialization Strategies -

- Parameter initialization means weight initialization i.e, adding some weight to the each neuron in deep learning.
- Proper weight intialization will helps to

i) Avoiding vanishing gradients i.e, when gradients become too small then stop the learning process.

ii) Avoiding exploding gradients i.e, when gradients grow too large training process make unstable.

iii) Breaking Symmetry i.e, it ensures that different neurons learns different features.

iv) Speed up Convergance i.e, The model learn faster.

Scanned with OKEN Scanner

Types of Initialization —

1. Zero Initialization — Formula $W = 0$

problem — if all weights are initialized as zero, then every neuron in a layer will make same output and gradient during back propagation. i.e all neurons learn the same thing, making the network useless.

When it will use —

- Never used for weight initialization, but in some cases used for bias initialization because bias start with "0".

2. Random Initialization — W = random small values

- it avoids the problem of symetry.
- if the values are too large then the activations explode.
- if the value are too small then learning slow down.

When it use —

- Random values can efficiently work in simple networks only, but not work properly in large networks.

3. Xavier Initialization — $W = N\left(0, \frac{1}{n^{in}}\right)$

where n^{in} is the no. of input neurons

- it keeps the variance of activations consistent in layers.
- it prevents the activations from becoming too small or too large.

Scanned with OKEN Scanner

When it use –

- it is used for Sigmoid functions not for ReLU functions.

4. He Initialization – $W \sim N\left(0, \frac{2}{n^{in}}\right)$

- it can be helps to remove the negitive values.
- it prevent the ReLU problem i.e the neurons are stops learning.

When it use –

- it is used for ReLU and leacy ReLU functions.
- it is suitable for all deep learning networks.

5. LeCun Initialization – $W \sim N\left(0, 1n^{in}\right)$

- it is designed for specifally for SELU functions.
- it ensures the activations remain self-normalizing.

When it use –

- it is used for SELU functions.
- it support all deep neural networks very well.

Noise Robustness –

- Noisy data means it contains irrelevent information, missing features, Errors, outliers etc. These information will leads to noisey problem.
- Noise Robustness is machine learning model ability that handle the noisy data without reducing the performace of an model.

Scanned with OKEN Scanner

Types of Noises —

1. Feature Noise (Input Noise) —

- It will occurs when input variables are incorrect or missing.

Ex- In a house prediction problem, if the no. of rooms recorded incorrectly the model learns wrong patterns.

Causes — 1. Measurement errors.
2. Data Corruption
3. Human mistakes.

2. Label Noise (output Noise) —

- It will occurs when output variables are incorrect or missing.

Ex- In an image classification task, an image of cat is mislabled as dog.

Causes — 1. Human Errors.
2. Sensor malfunction
3. mis classification in data.

3. Adversiral Noise —

- It will occurs when intentionally added some noise to the original data it leads to mislead the model.

Ex- changing a few pixels in image it will lead to misleads.

Causes — 1. Malicious Attacks
2. Security Threat Attacks.

4. Environmental Noise — External factors that effect the data collection.

Ex- Background Noise in Speech Recognition.

Scanned with OKEN Scanner

Drawbacks of Noise —

1. Increasing the training time.
2. Reduces Accuracy
3. Leads to overfitting problems.
4. Bias - variance trade off

Methods to Improve Noise Robustness —

1. Data pre Processing Techniques -

i) outlier detection & Removal — it detect and remove the extreme values like Z-Score method and IQR method.

ii) Data Augmentation — it will add some variations to training data like rotation, flipping, Colour changes etc).

iii) missing data values — it will replace the missing values by adding mean or mode.

2. Robust model Architecture —

i) Regularization Technique — it will helps to preventing the overfitting problems by L_1 and L_2 Regularization techniques.

ii) Dropout method - it randomly drops the neurons during the training period to prevent overfitting problems.

iii) ensemble learning — it will Combine all multiple weak models to make stronger and improves the stability.

Ex - Random forest, Bagging

Scanned with OKEN Scanner

3. Noise robust Training —

i) Early stopping — It will stop the learning process when the loss of functions becomes higher.

ii) Adversiral training — Here train the model with intentionally added noise, to improve the robustness.

Ex - Add small changes in an image for better appriance,

iii) Data weighting method — By Assinging weights to each sample based on the importance of the sample,

Applications —

1. Speech Recognition
2. medical daignosis
3. finance
4. Autonomous vechicles,

Multitasking learning —

- multitasking learning is a machine learning technique where a model is trained to perform multiple relative tasks simultaneously.
- Generally for each task different model will be trained but in multitask learning for ~~each~~ tasks we use one better multiple trained model,

Advantages —

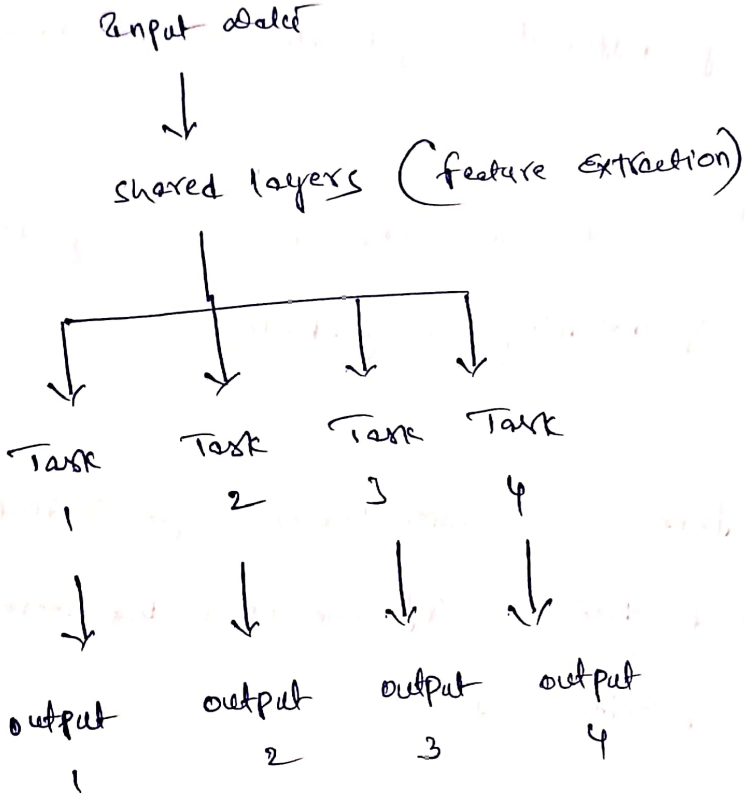
1. Improves generalisation
2. Reduce overfitting
3. Efficient learning
4. Handles data scarcity

Scanned with OKEN Scanner

How multitasking work — Basically it works based on 2 types.

1. Hard parameter sharing — Here we are using a shared network with seperate output layers for each task.

— It will helps to reduce the overfitting.



2. Soft parameter sharing —

— Here for each task it has it own model but parameters are regularized to be similar.

Applications of Multitasking learning —

1. Computer vision — for detecting the objects
2. NLP — for sentimental analysis, entity Recognization
3. Health Care — diagnosing the multiple diseases
4. Autonomous Vechicles — for object detection, predictions.

Scanned with OKEN Scanner

Multitasking Algorithms —

1. multitask - Neural Network (MTNN) —
   - it uses shared layers for feature extraction and seperate layer for each task.
2. Uncertainly based Weighting —
   - Assinging different weight to the each task based on the importance.
3. Grad Norm — it will balances the training procss across all tasks by adjutting the gradients.

challenges —

1. Task Interence — Some Tasks might Conflict.
2. Imbalanced data — If one task has more data, the model can prioritize it.

Sparse Representations —

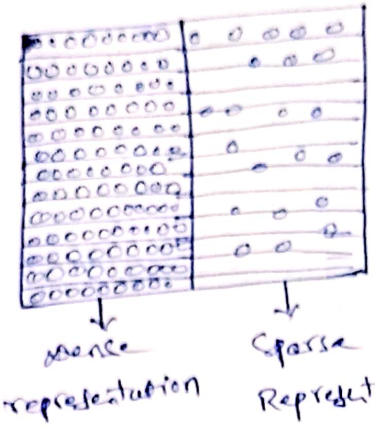
- Sparse Representation is the way of encoding the data where the most elements are zero or not active. This is the opposite of dense representation.
- Instead of representing a signal with all basis vectors, a sparse representation is uses only a few significant ones.

Why Sparsity is Need —

1. To improve the efficiency
2. To improve Robustness
3. To maintain interpretability

Scanned with OKEN Scanner

Dense Vs Sparse representation —



Dense Representation — 1. The grid is mostly filled with blank cells
2. Most of the elements are non-zero values.
3. It uses all features.

Sparse Representation — 1. The grid is filled with few colour cells.
2. most of the elements are zero values.
3. It uses a few features.

How to define Sparse Representations —

1. **L_1 Regularization** — It Encourages the sparsity in models.
2. **Dictionary learning** — It learns a set of functions where data can be represented sparsely.
3. **Auto Encoders** — It encourages the hidden layers to have sparse activations.
4. **Thresholding** — set small values to zero.

Applications — 1. Computer Vision
2. NLP
3. Neuro Science

Scanned with OKEN Scanner

Tangent distance, Tangent prop and manifold Tangent classifier —

1. **Tangent distance** — it is a method used in image recognition to compare two images in a more flexible way than just pixel-to-pixel comparision.

- Instead of directly comparing two images we check how much we need to transform one image to match another image.

Why it is useful —

- Small changes in position, rotation can make two images look different in pixels but they still represent same object.
- Tangent distance allows us to compare both images based on their shape not in pixel value.

How it works —

1. **find Tangent vectors** — Identify small transformations like shifting, rotation and scaling that don't change the image meaning.
2. **Measure distance** — Compute the shortest distance between the transformed versions of the two images.
3. **Better matching** — If the two images belong to the same object, their transformed versions should be close.

2. **Tangent propagation** —

- it is a technique used in neural networks to make models to learn invariant features i.e, the model recognize if few transformations are applied in an image.

Scanned with OKEN Scanner

- Instead of just training the neural network on normal data, we also train it to recognize slightly transformed versions of the data.

Why it is useful -

- It helps the model ignore small changes like rotation or brightness.
- It improves the generalization i.e the model works well on new unseen data.

How it works -

1. Compute Tangent vectors - find small change like rotation, shifting
2. Modify Training - Instead of just learning from raw inputs, the model also learn from these transformed versions.
3. Better prediction - the network become more robust and make better predictions.

3. Tangent manifold classifier -

- It is a method used in deep learning to make neural networks learn better representations of data by considering the underlying structre of the data.
- These data points can be lie on low-dimentional surface in high dimentional space. We want our model to learn this structre so it can generalize better.

Why it is usefull -

- data exists in high dimentions, but
- It helps the model focus on meaningfull variations and ignore unnecessary noise.

Scanned with OKEN Scanner

How it still works -

1. Find the manifold - Identify small changes that do not change the data meaning.
2. Train with Tangent Constraints - make the neural network understand and repeat these small changes.
3. classify based on the manifold - Instead of treating each point seperately the model classifies data based on its location on the manifold.

Learning vs Pure optimization -

learning is a when a model improve its performance by understanding its patterns from the data. it generalizes to new situation instead of just memorizing.

Ex. - Training data → Learning Algorithm → model generalized
(Cats) (Neural Network) (recognize new cats)

features -

1. Uses past experience like training data.
2. Generalizes to unseen data
3. Tries to minimize the errors while still understanding patterns.

When to use learning -

1. You need to generalize
2. The data is complex and has patterns
3. You want to predict future outcomes.

Scanned with OKEN Scanner

Pure optimization — Pure optimization is a method to finding the best possible solution to the given problem without worrying about the learning patterns.

Ex — Solving a math equation find x that makes $3x + 5 = 25$

objective function → optimization Algorithm → Best solution
(minimize cost) (Gradient decent) (Lowest error)

features —

1. No generalization it focus only the specific problem.
2. It will focus on minimize or maximize the function.
3. Does not improve over time it only finds best possible solution.

differences —

feature	Learning	Pure optimization
1. Goal	understand the patterns and generalize	find the best solution for specific problem.
2. Memory of data	Yes remember it improves over time	No, only solves once
3. Example	Recognize Cats in images	finding the shortest path in a maze
4. Generalization	Yes, works on new data	No, only for the given problem
5. Method	Using training data and adjust parameters	Uses formulas or Algorithms to min/max the function.

Scanned with OKEN Scanner

challenges in neural Network optimization –

– Neural Network optimization is a process of improving the performance of a neural network by adjusting the parameters like weight and biases.

– The main goal of optimization is to reduce the loss function and improves the accuracy.

– However optimizing neural network is difficult due to several challenges.

1. Vanishing & Exploding gradients –

What happens –

– When training deep neural networks the gradients (updates to weights) becomes too small or too large.

– This may learning slow or causes to model fail.

Solution –

– Use Relu function instead of sigmoid activation function.

– Use batch normalization to stabilizes the updates.

– Use gradient clipping to prevent gradient exploding.

2. Getting stuck in local minima –

What happens –

During training the algorithm tries to find best weights by minimize the loss function.

– Sometimes it will get stuck in local minimum i.e small dip instead of find global minimum (deepest point)

Scanned with OKEN Scanner

Solution -

- use optimizers like Adam, RMSdrop to escape local minima.
- use learning rate scheduling to make learning process faster.
- Try Random weight initialization to avoid bad starting points.

3. Overfitting (Poor Generalization) -

What happens -

- The model performance well on training data but fails on Unseen data. it memorizes the training Example instead of learning pattern.

Solution -

- use Regularization techniques L_1, L_2, Dropout to prevent overfitting.
- increase training data using data Augmentation.
- use early stopping to reduce overfitting.

4. choose Right learning Rate -

What happens -

- if the learning rate is too high the model jumps around and never converges.
- if the learning rate is too low learning is slow and may get stuck.

solution -

- use learning rate scheduling to reduce learning rate over time.
- use Adaptive learning rate Adam, RMSdrop to adjust the learning rate dynamically.

Scanned with OKEN Scanner

5. Hyper parameter Tuning difficult –

What happens –

- choosing the right batch size, learning rate, number of layers, number of neurons is very tricky.
- A wrong combination can make the model performance poor.

Solution –

- use Grid Search to test different values.
- Use Bayesian optimization for automated tuning.

Semi Supervised Learning –

- Semi Supervised learning is type of machine learning Algorithm where we use both labeled and unlabeled data to train a model.
- it is in middle of the both supervised and unsupervised learning Algorithms.

Why we need Semi Supervised learning –

- labeled data is expensive and time consuming to collect.
- unlabeled data is too large and harder to use.
- Semi Supervised learning helps by using small amount of labeled data and large amount of unlabeled data to improve accuracy.

How it will works –

- first, the Model learns from the labeled data.
- Then it use these patterns to make predictions on the unlabeled data.
- finally it learns from both to improve overall performance

Scanned with OKEN Scanner

Techniques used in semi supervised -

1. Self-Training - The model predicts labels for unlabeled data, Then reuse those predictions for training.

2. Consistency regularization -

The model makes small changes to the data like rotation, flipping and expects the prediction to stay the same.

3. Graph Based learning -

- data points can be treated as node in a graph
- labeled points help spread knowledge to unlabeled points that are similar.

4. Pseudo - labeling -

- The model assigns fake labels to unlabeled data and learns from them.

Advantages -

- Uses less labeled data and saves the money.
- Performs better than unsupervised learning method.
- It can work with large data sets.

Dis advantages -

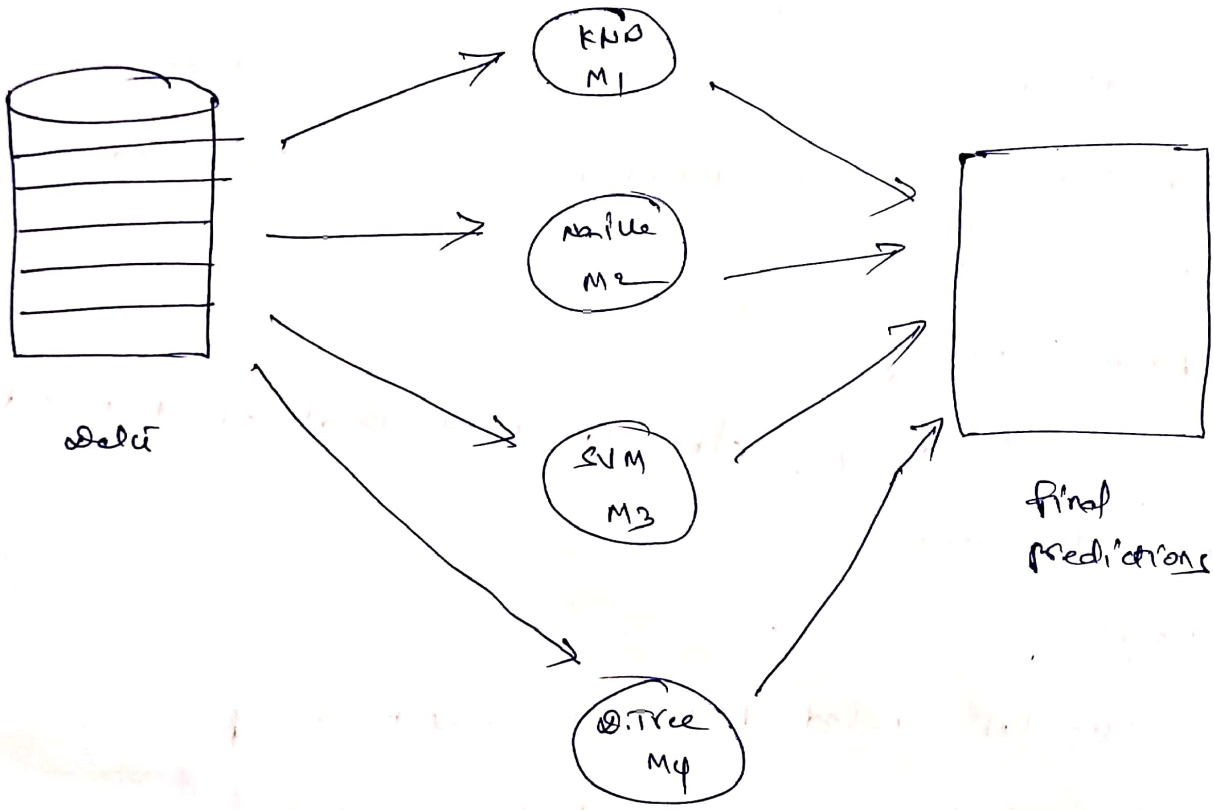
- Wrong labels can reduce the accuracy.
- more complex than supervised learning
- not always reliable, it depends on the how unlabeled data matches with labeled data.

Scanned with OKEN Scanner

Ensemble learning methods -

Bagging and Pasting -

- Ensemble method is a technique that combines all the predictions from the multiple learning algorithms to make more accurate predictions than individual model.
- A model is comprised by many model is called as ensemble model.

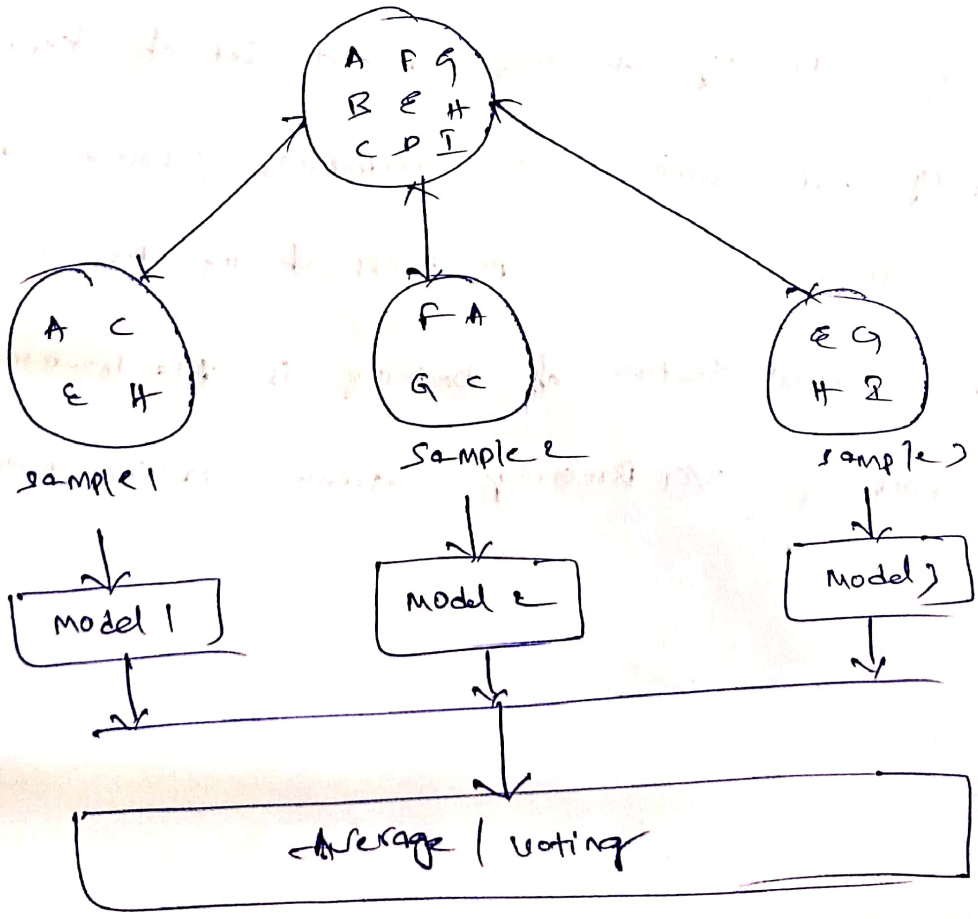


- Many ensemble methods contain same type of learning algorithms is called Homogenous ensemble and some methods contains different types of learning algorithms is called Heterogenous Ensemble learning methods.

Scanned with OKEN Scanner

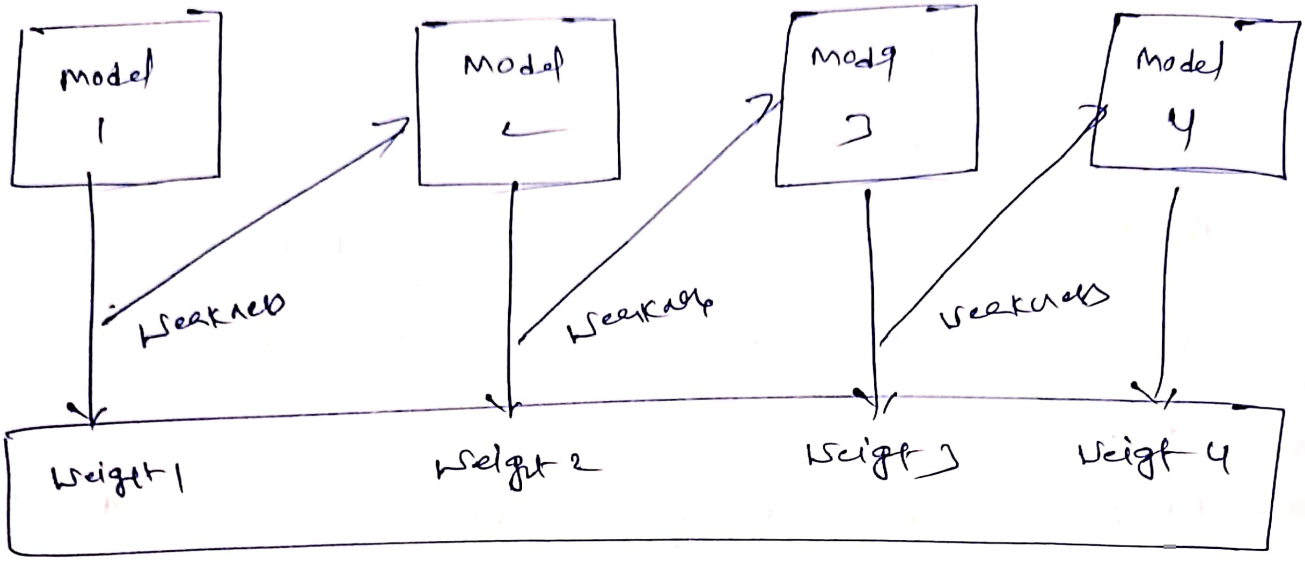
1. Bagging – (Bootstrap Aggregation)

- Bagging is a procedure it is used to reduce the high variance of an algorithm i.e. reduce the overfitting problems.
- Bagging makes each model to run independently and aggregates the output at the end without preference to any model.
- Bagging is performed by two factors i.e. Bootstrap and Aggregation.
- Bootstrap means suppose a data for certain 'n' sample we take 'm' sample and trained with an model.
- Aggregation means after training data combine the predictions and take majority of mode or median.



Scanned with OKEN Scanner

2. Boosting –



Ensemble with all predecessors.

– Boosting is an algorithm it will convert the weak learners to strong learners. Here base learners are trained sequentially then the next learner is trying to reduce the errors and updating the parameters and compared to previous learners.

– Boosting work by training different set of learners sequentially and combining the predictions, where later learners focus more on the mistakes of the previous learner.

– Most important feature of boosting is the Boosting Gradient boosting, XG Boosting Extreme Gradient Boosting.

Scanned with OKEN Scanner