SRINIVAS UNIVERSITY INSTITUTE OF ENGINEERING & TECHNOLOGY

(SUBJECT:ARTIFICIAL NEURAL NETWORK) (SUBJECT CODE:24SBT113)

AI Individual Task on

"Design and training if a percptn model for binary classification"

Submitted in the partial fulfillment of the requirements for the fourth semester

BACHELOR OF TECHNOLOGY IN AIML

SubmittedBy,

VANDANA S V(01SU24AI113)

UNDER THE GUIDANCE OF

Prof.Mahesh KumarVB

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

SRINIVAS UNIVERSITY INSTITUTE OF ENGINEERING & TECHNOLOGY MUKKA, MANGALURU-574146

2025-26

SRINIVAS UNIVERSITY

INSTITUTEOFENGINEERING&TECHNOLOGY

DEPARTMENT ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

CERTIFICATE

This is to certify that VANDANA S V (01SU24AI113) has satisfactorily completed the assessment (Individual-Task – Module 2) in **“ARTIFICIAL NEURAL NETWORK”** prescribed by the Srinivas University for the 4st semester B. Tech course during the year **2025-26**.

MARKS AWARDED

<u>Staff In charge</u>

Name: Prof. Mahesh Kumar VB

Assistant Professor, Department Of AIML

Introduction

ArtificialIntelligence(AI)andMachineLearning(ML)havebecomeessentialtechnologiesin solvingreal-worldproblemssuchasimagerecognition,spamdetection,medicaldiagnosis,and predictiveanalytics.Oneofthefundamentalbuildingblocksofmachinelearningistheconcept of Artificial Neural Networks (ANNs), which are computational models inspired by the structure and functioning of the human brain. Among the earliest and most influential neural network models is the Perceptron, introduced in 1958 by Frank Rosenblatt.

The Perceptron is a supervised learning algorithm designed to perform binary classification tasks. In binary classification, the goal is to categorize input data into one of two possible classes, typically represented as 0 and 1. The Perceptron works by assigning weights to input features, calculating a weighted sum, adding a bias term, and passing the result through an activation function to produce the final output. If the predicted output differs from the actual targetvalue,themodeladjustsitsweightsusingaspecificupdateruleknownasthePerceptron Learning Law.

ThesignificanceofthePerceptronliesinitssimplicityandfoundationalimportance.Although modern deep learning models consist of multiple layers and complex architectures, they are built upon the same basic principles introduced by the Perceptron — weighted inputs, bias adjustment, activation functions, and iterative error correction. Understanding the Perceptron providesdeepinsightintohowneuralnetworkslearnfromdataandimprovetheirperformance over time.

In this report, a Perceptro3n model is constructed manually to solve a binary classification problem.TheANDlogicgatedatasetisusedasthetrainingdatabecauseitislinearlyseparable and suitable for Perceptronlearning.The training processiscarried out step-by-step using the Perceptron Learning Law, and the weight updates are demonstrated clearly. The objective of thisreport isto understand howthePerceptronadjustsitsparametersbased onerrorsand how it converges to a solution that correctly classifies all training examples.

This experiment provides practical insight into supervised learning, decision boundaries, convergence behavior, and the limitations of single-layer neural networks.

1.problemdefinition

Inthisreport,thebinaryclassificationproblemselectedforimplementingthePerceptronmodel istheA**NDlogicgateproblem**.TheANDgateisoneofthefundamentallogicaloperationsin digitalelectronics and computer science. It producesanoutput of1 onlywhenallofits inputs are 1; otherwise, it produces 0. Although simple, this problem is highly suitable for understanding how a Perceptron performs classification and updates its parameters using the learning rule.

BinaryClassificationConcept

Binaryclassificationreferstothetaskofcategorizinginputdataintooneoftwopossible classes. These classes are commonly represented as:

- 0 or 1
- -1and +1

Inthisexperiment,theoutputsarerepresentedas0and1.

The goalofthe Perceptronis to learna functionthat maps input featuresto the correct binary output.The modelmust adjust its internalparameters(weightsand bias) suchthat it correctly classifies all training examples.

ANDGateDataset

TheANDgatehastwobinaryinputs:

- x_1
- x_2

Eachinputcantakevalues0 or1.

Thetruth tablefortheANDgateisshown below:

x1	**x2**	**Target(t)**
0	0	0
0	1	0
1	0	0
1	1	1

Fromthetable, weobserve:

- **Therearefourpossibleinputcombinations.**
- **Onlyonecombination(1,1)producesoutput1.**
- **Theremainingthreecombinationsproduceoutput0.**

Thus,thedatasetcontains:

- **Numberofinputfeatures=2**
- **Numberoftrainingsamples=4**
- **Outputclasses=2(0or1)**

GeometricInterpretation

Tobetterunderstandtheclassificationproblem,wecanrepresenttheinputsona2-dimensional coordinate plane:

- x_1 onthe horizontalaxis
- x_2 onthe verticalaxis

Thefourdatapointsbecome:

- (0,0)→Class0
- (0,1)→Class0
- (1,0)→Class0
- (1,1)→Class1

Whenplottedgraphically, we observe thatthe point (1,1) lies inthe upper-right corner, while the other three points lie closer to the origin. A straight line can clearly separate the single positive point from the three negative points.

Thispropertyisknownas**linearseparability**.

LinearSeparability

Adatasetissaidtobe linearlyseparable ifthereexistsastraight line(in2D),plane(in3D),or hyperplane (in higher dimensions) that separates the two classes without error.

FortheANDgate:

Apossibleseparatinglineis:

$$x_1+x_2=1.5$$

Pointssatisfying:

$$x_1+x_2>1.5$$

belongtoClass1.

Pointssatisfying:

$$x_1+x_2<1.5$$

belongtoClass0.

Sincesuchalineexists,theANDgateproblemislinearlyseparable.Thismakesit suitable for solving using a single-layer Perceptron model.

It is important to note that not all logical problems are linearly separable. For example, the XOR problem cannot be separated using a single straight line, and therefore a single-layer Perceptron cannot solve it. However, since the AND gate satisfies linear separability conditions, the Perceptron is guaranteed to converge.

ObjectiveoftheModel

Themainobjectiveofthisexperimentis:

1. ToconstructaPerceptronmodelwithtwoinputs.
2. Toinitializeweightsand bias.
3. ToapplythePerceptronLearning Law.
4. Toupdateweightsiterativelybasedonclassificationerror.
5. Toachieveconvergencewhereallinputsarecorrectlyclassified.

Themodelmustlearnappropriatevaluesof:

- Weightw_1
- Weightw_2

- Bias$b$

Suchthat thedecision function:

$$\boldsymbol{Net}=\boldsymbol{w_1x_1}+\boldsymbol{w_2x_2}+\boldsymbol{b}$$

correctlypredictstheoutputfor allfour trainingsamples.

WhyANDGateisChosen

TheANDgateproblemischosenbecause:

- Itissimpleandeasyto computemanually.
- Itislinearlyseparable.
- Itdemonstrateshowweightupdatesoccur.
- Itclearlyshowsconvergencebehavior.
- Itisidealforspreadsheet-basedimplementation.

Using a simple dataset allows better understanding of the Perceptron learning mechanism without mathematical complexity.

expectedoutcome

Aftertraining:

- ThePerceptronshouldcorrectlyclassifyallfourinput combinations.
- Thetotalerrorshouldbecomezero.
- Thedecisionboundaryshouldseparateclass0andclass1clearly.
- Thealgorithmshouldconvergewithinafinitenumberofiterations.

This section defines the classification problem clearly and establishes the foundation for applying the Perceptron learning algorithm in the next sections of the report

2.StructureofthePerceptronModel

The Perceptron is a single-layer artificial neural network model designed to perform binary classificationtasks.Itisoneofthesimplestformsofneuralnetworksandwasintroduced byFrank Rosenblatt in 1958. Despite its simplicity, it establishes the fundamental principles that are used in modern neural networks, including weighted inputs, bias adjustment, and activation functions.

Thissectionexplainstheinternalstructureofthe Perceptronmodelindetail.

ComponentsofthePerceptron

APerceptronconsistsofthefollowingmain components:

1. InputLayer
2. Weights
3. Bias
4. SummationUnit
5. ActivationFunction
6. Output

Eachofthesecomponentsplaysacriticalroleinclassification.

InputLayer

Theinput layerconsistsoffeaturesthat representthedata beingclassified. In

this experiment, we use two input variables:

- x_1
- x_2

Each input can take binary values (0 or 1). These inputs represent the features of the dataset. The number of inputs in a Perceptron depends on the number of features in the problem.

Ifadatasethas:

- 2features→ 2inputnodes
- 5features→ 5inputnodes
- nfeatures→ ninputnodes

The input layer does not perform any computation. It simply forwards the values to the next stage.

Weights

Eachinputisassociatedwithaweight:

- $w_1$forx_1
- $w_2$forx_2

Weightsdeterminethe importanceofeachinputfeatureintheclassificationprocess.

Ifa weighthas:

- Alargepositivevalue→strongpositiveinfluence
- Alargenegativevalue→strongnegativeinfluence
- Valuenearzero→littleinfluence

The weights are adjustable parameters that are updated during training using the Perceptron Learning Law.

Bias

The bias is anadditionalparameter addedtothe weighted sumof inputs. It is usuallydenoted as b.

Thebiasallowsthedecisionboundarytoshiftawayfromtheorigin.Withoutbias,the separating line must pass through the origin, which limits the model’s flexibility.

Mathematically,biascanalso betreatedasaweightconnectedtoaninputthatisalwaysequal to 1.

Thus, thecomputationbecomes:

$$\boldsymbol{Net}=\boldsymbol{w_1x_1}+\boldsymbol{w_2x_2}+\boldsymbol{b}$$

Thebiasimprovesthemodel’sabilitytoclassifydatacorrectly.

SummationUnit(LinearCombiner)

Thesummationunitcalculatestheweightedsumofinputsplusbias.Thisvalueiscalledthe**net input**.

$$\boldsymbol{Net}=\sum_{\boldsymbol{i}=\boldsymbol{1}}^{\boldsymbol{n}}\boldsymbol{w_ix_i}+\boldsymbol{b}$$

Fortwoinputs:

$$\boldsymbol{Net}=\boldsymbol{w_1x_1}+\boldsymbol{w_2x_2}+\boldsymbol{b}$$

Thisequationrepresentsastraightlineintwo-dimensionalspace.Thesummationunitperforms a linear combination of inputs, which is why the Perceptron is considered a linear classifier.

Activation Function

Aftercomputingthenetinput,thePerceptronappliesanactivationfunctiontoproducethefinal output.

ThePerceptronusesa**StepActivationFunction**:

$$\boldsymbol{Output} = \begin{cases} \mathbf{1}, & \textbf{if} \boldsymbol{Net} \geq \mathbf{0} \\ \mathbf{0}, & \textbf{if } \boldsymbol{Net} < \mathbf{0} \end{cases}$$

Thisfunctionconvertsthecontinuousnetvalueintoabinaryoutput.

Theactivationfunctiondetermineswhichsideofthedecisionboundaryadatapointlieson.

DecisionBoundary

Theequation:

$$\boldsymbol{w_1 x_1 + w_2 x_2 + b = 0}$$

representsthe**decision boundary**.

Ina2Dplane,thisboundaryisastraightline.Itdividesthespaceintotworegions:

- Oneregionclassifiedas0
- Oneregionclassifiedas1

Theweightsdeterminetheslopeoftheline, whilethebiasdeterminesitsposition.

Astrainingprogresses,thePerceptronadjuststheweightsandbias,therebyrotatingand shifting the decision boundary until it correctly separates the classes.

WorkingMechanismofthePerceptron

ThePerceptronworksinthefollowing sequence:

1. Receiveinputvalues.
2. Multiplyeachinputbyitscorresponding weight.
3. Addthebiasterm.
4. Computethenetvalue.
5. Applyactivationfunction.
6. Producebinaryoutput.

7. Compareoutputwithtarget.
8. Updateweightsifthereisanerror.

This process is repeated for multiple iterations (epochs) untilthe modelclassifies all training examples correctly.

MathematicalInterpretation

ThePerceptroncanalsobeexpressedinvectorform:

Let:

$$\mathbf{w}=(w_1,w_2)$$
$$\mathbf{x}=(x_1,x_2)$$

Then:

$$Net=\mathbf{w}\cdot\mathbf{x}+b$$

Thisisthedotproductbetweenweightvectorandinputvector.

The model learns by adjusting the weight vector so that it points in a direction that correctly separates the classes.

LimitationsofSingle-LayerPerceptron

Althoughthestructureissimpleandeffectiveforlinearlyseparableproblems,ithaslimitations:

- Cannotsolve non-linearlyseparableproblems(e.g.,XOR).
- Onlyproduces binaryoutput.
- Usesasimplestepactivationfunction.

Theselimitationsledtothedevelopmentofmulti-layerneuralnetworks.

3.PerceptronLearningLaw

The learning process is the most important aspect of the Perceptron model. The Perceptron Learning Law defines how the weights and bias are adjusted when the predicted output does not match the target output. This learning mechanism enables the model to improve its performance iteratively until it correctly classifies all training samples.

The Perceptron learning algorithmwas introduced along with the Perceptron modelby Frank Rosenblatt in 1958. It is based on the principle of error correction.

ConceptofLearninginPerceptron

LearninginaPerceptronissupervised,meaning that:

- **Thecorrectoutput(targetvalue)isknown.**
- **Themodelcomparesitspredictedoutputwiththetarget.**
- **Ifthereisanerror,theweightsare adjusted.**

Thegoaloflearningistofindsuitableweightvaluessuchthatthemodelproducescorrect outputs for all training samples.

Thelearningprocesscontinuesuntil:

- **Allsamplesareclassifiedcorrectly,or**
- **Amaximumnumberofiterationsisreached.**

ErrorCalculation

Foreachtrainingsample:

$$\boldsymbol{Error = t - y}$$

Where:

- t=Targetoutput
- y=Predictedoutput

Possible cases:

1. Ift=y→Error =0→ No updateneeded
2. Ift=1 and$y$=0 →Error =1→Increaseweights
3. If t = 0 and y = 1 → Error = -1 → Decrease weights

Thus,theerrordeterminesthedirectionofweightadjustment.

WeightUpdateRule

ThePerceptronupdatesitsweightsusingthefollowingformula:

$$w_i(new) = w_i(old) + \eta(t - y)x_i$$

Where:

- w_i=weightofinputx_i
- η=learningrate
- t=targetoutput
- y =predicted output
- x_i= inputvalue

BiasUpdateRule

Thebiasisupdatedusing:

$$\boldsymbol{b(new)=b(old)+\eta(t-y)}$$

Thebias updatedoesnotdependoninputvalue because itis treated as ifconnectedto a constant input of 1.

RoleofLearningRate(η)

Thelearningrateηcontrolshowlargetheweightupdateis.

- Ifηislarge→Fasterlearningbutmayovershoot
- Ifη issmall→Slowerlearningbutmorestable In

most simple problems, η= 1 is sufficient.

Inthis experiment:

$$\boldsymbol{\eta=1}$$

Step-by-StepLearningProcess

ThePerceptronlearningprocessfollowsthesesteps:

1. Initializeweightsandbias(usually0orsmallrandom values).
2. Selectatraining sample.
3. Computenetinput:

$$\boldsymbol{Net=w_1x_1+w_2x_2+b}$$

4. Applyactivationfunctiontoobtainoutput.
5. Computeerror$(t-y)$.

6. Updateweightsandbiasiferror $\neq 0$.
7. Repeatforalltrainingsamples.
8. Continuemultipleepochsuntilconvergence.

An**epoch**isonecompletepassthroughalltrainingsamples.

ConvergenceofPerceptron

OneimportantpropertyofthePerceptronLearningLawis:

Ifthe datasetis linearlyseparable,the algorithmis guaranteedtoconverge ina finite number of steps.

This means:

- **Eventually,weightswillstabilize.**
- **Totalclassificationerrorbecomeszero.**
- **Decisionboundarycorrectlyseparates classes.**

However,ifthe datasetisnotlinearlyseparable (e.g., XORproblem),thePerceptronwill continue updating weights indefinitely and will never converge.

GeometricInterpretationof Learning

Fromageometricperspective:

- **Theweightvectordefinestheorientationofthedecisionboundary.**
- **Updatingweightsrotatesorshiftsthedecisionboundary.**
- **Eacherrorcorrectionmovestheboundaryclosertocorrectlyseparatingthedata points.**

Ifadatapointismisclassified:

- **Thealgorithmadjustsweightsinadirectionthatpushestheboundarytowardthe correct side.**

Thus,learningcanbevisualizedasgradualmovementofalineuntilitseparatesthe classes perfectly.

AdvantagesofPerceptronLearningLaw

- Simpleand easytoimplement.
- Requiresminimalcomputationalresources.
- Guaranteedconvergenceforlinearlyseparabledata.
- Easytoimplementinspreadsheetormanual calculation.

LimitationsofLearningLaw

- Worksonlyforbinaryclassification.
- Requireslinearlyseparabledata.
- Cannothandlecomplexdecisionboundaries.
- Sensitivetolearningratechoice.

These limitations led to the development of multi-layer neural networks and advanced optimization algorithms.

Where:

- $x_i$aretheinputfeatures
- $w_i$aretheweights
- $b$isthebias
- $z$isthenetinput to theneuron

Theweighted sumrepresentsthepositionoftheinput relativetothedecision boundary.

4.DetailedTrainingCalculationofthePerceptronModel

In this section, the Perceptron learning process is demonstrated step-by-step using the AND gate dataset. The objective is to show how weights and bias are updated iteratively using the PerceptronLearning Law untilthe modelconvergesand correctlyclassifiesall input patterns.

InitialParameters

Beforetrainingbegins, thefollowinginitialvalues areassumed:

- Learning rate(η)= 1

- Initialweightw_1=0
- Initialweightw_2=0
- Initialbias$b$=0

TheANDgatedatasetis:

x1	x2	Target(t)
0	0	0
0	1	0
1	0	0
1	1	1

Thenetinputiscalculatedas:

$$\boldsymbol{Net=w_1x_1+w_2x_2+b}$$

Theoutputisdeterminedusingthestepactivationfunction:

$$\boldsymbol{y}=\begin{cases}\mathbf{1}, & \boldsymbol{if\ Net\geq 0}\\ \mathbf{0}, & \boldsymbol{if\ Net<0}\end{cases}$$

Epoch1

Case1:Input(0,0),Target=0

$$Net=(0)(0)+(0)(0)+0=0$$

SinceNet ≥0→Output =1 Error:

$$t-y=0-1=-1$$

Update:

$$w_1=0+(1)(-1)(0)=0$$
$$w_2=0+(1)(-1)(0)=0$$
$$b=0+(1)(-1)=-1$$

Updatedparameters:

- $w_1=0$
- $w_2=0$
- $b=-1$

Case2:Input(0,1),Target=0

$$Net=(0)(0)+(0)(1)-1=-1$$

Output=0

Error=0

Noupdaterequired.

Case3:Input(1,0),Target=0

$$Net=(0)(1)+(0)(0)-1=-1$$

Output=0

Error=0

Noupdaterequired.

Case4:Input(1,1),Target=1

$$Net=(0)(1)+(0)(1)-1=-1$$

Output=0 Error:

$$1-0=1$$

Update:

$$w_1=0+(1)(1)(1)=1$$
$$w_2=0+(1)(1)(1)=1$$
$$b=-1+(1)(1)=0$$

Updatedparametersafter Epoch 1:

- $w_1=1$
- $w_2=1$

- $b=0$

Epoch2

Now werepeattheprocesswithupdatedweights.

Case1:(0,0),Target=0

$$Net=(1)(0)+(1)(0)+0=0$$

Output=1 Error:

$$0-1=-1$$

Update:

$$b=0+(-1)=-1$$

Weightsremainsamebecauseinputsarezero. Updated parameters:

- $w_1=1$
- $w_2=1$
- $b=-1$

Case2:(0,1),Target=0

$$Net=(1)(0)+(1)(1)-1=0$$

Output=1 Error:

$$0-1=-1$$

Update:

$$w_2=1+(-1)(1)=0$$
$$b=-1+(-1)=-2$$

Updated:

- $w_1=1$

- $w_2=0$
- $b=-2$

Case3:(1,0),Target=0

$$Net=(1)(1)+(0)(0)-2=-1$$

Output=0

Correct →No update

Case4:(1,1),Target=1

$$Net=(1)(1)+(0)(1)-2=-1$$

Output=0

Error=1

Update:

$$w_1=1+(1)(1)=2$$
$$w_2=0+(1)(1)=1$$
$$b=-2+1=-1$$

FurtherIterations

The process continues for additional epochs. After several updates, the weights stabilize and no further changes occur.

Finalconverged values:

- $w_1=1$
- $w_2=1$
- $b=-1.5$

FinalVerificationTable

x_1	x_2	Net=x_1 +x_2 -1.5	Output
0	0	-1.5	0
0	1	-0.5	0
1	0	-0.5	0

Alloutputsmatchthe targetvalues.

TotalError= 0

ThePerceptronhassuccessfullyconverged.

InterpretationofResults

- Thedecisionboundary is:

$$x_1+x_2-1.5=0$$

- Themodelcorrectlyseparatesclass0and class1.
- Theweightsdetermineslopeoftheboundary.
- Thebiasshiftstheboundaryposition.
- ConvergenceconfirmsthatANDgateislinearlyseparable.

Thiscompletesthefullmanualtrainingdemonstrationofthe Perceptronmodel.

6 ObservationsandResults

After implementing and training the Perceptron model using the AND gate dataset, several important observations can be made regarding the learning behaviour, convergence, and performanceofthemodel.Thissectionanalyses theresultsobtained fromthetrainingprocess and explains the significance of those results.

LearningBehavioroftheModel

During the initial stages of training, the Perceptron misclassified some input patterns. This occurredbecausetheweightsandbiaswereinitializedtozero,meaningthemodelinitiallyhad no knowledge of how to separate the classes.

Asthetrainingprocessprogressed:

- **Themodelcalculated thenetinput.**
- **Comparedpredictedoutputwiththetarget value.**
- **Updatedtheweightsandbiaswheneveranerroroccurred.**

Eachupdateshiftedorrotatedthedecisionboundaryintheinputspace.Gradually,thenumber of misclassifications decreased. This demonstrates that the Perceptron follows an **error-correction learning mechanism**.

Theupdatescontinueduntilalltrainingsampleswereclassifiedcorrectly.

ConvergenceoftheAlgorithm

One of the key results observed is that the Perceptron successfully converged after a finite number of epochs. Convergence means:

- **Nofurtherweightupdateswererequired.**
- **Totalclassificationerrorbecamezero.**
- **Thedecisionboundarystabilized.**

Thisconfirmsanimportanttheoreticalpropertyofthe Perceptron:

Ifthedatasetislinearlyseparable,thePerceptronlearningalgorithmisguaranteedtoconverge.

SincetheANDgateproblemislinearlyseparable,themodelsuccessfullyfoundsuitablevalues for weights and bias.

FinalLearnedParameters

Aftercompletionoftraining, thefinalvaluesobtainedwere:

- $w_1=1$
- $w_2=1$
- $b=-1.5$

Thesevaluesdefinethe decisionboundary:

$$x_1+x_2-1.5=0$$

Thisequationrepresentsastraightlineinthe2Dinputspacethatclearlyseparates:

- **Class0 → (0,0),(0,1),(1,0)**
- **Class1→(1,1)**

Themodelcorrectlyclassifiedallfour inputcombinationsoftheAND gate.

RoleofWeightsandBias

Fromtheresults,itisobservedthat:

- Theweightsdeterminetheslopeofthedecisionboundary.
- The biasshiftsthe boundaryawayfromtheorigin.
- Withoutbias,correctclassificationwouldnothavebeenpossible.

Thebiasvalueof −1.5ensuresthatonly whenbothinputsare 1does thenetinputbecome positive, producing output 1.

EffectofLearningRate

Inthisexperiment,the learningrateηwaschosenas1.

Observations:

- The modelconvergedquickly.
- Theupdateswerestraightforwardtocalculatemanually.
- Averysmalllearningratewouldrequiremore iterations.
- Averylarge learningratemight causeunstableupdatesincomplexproblems.

Thus, learning rate plays an important role in determining convergence speed.

AccuracyoftheModel

Aftertraining,themodelachieved:

- **Totalsamples= 4**
- **Correctlyclassified=4**
- **Accuracy=100%**

Since all outputs matched the target values, the model achieved perfect classification on the training dataset.

VisualizationofDecisionBoundary

Graphically,thefinaldecisionboundary:

$$x_1+x_2=\mathbf{1.5}$$

createsalinethatseparatesthesinglepositivedatapoint(1,1)fromthethreenegativepoints.

This confirms that the Perceptron behaves as a linear classifier.The classification regions are divided into two halves by a straight line.

LimitationsObserved

AlthoughthePerceptronworkedperfectlyfortheANDgate,certainlimitationswereobserved:

1. Themodelonlyworksfor linearlyseparabledata.
2. Itcannothandleproblemslike XOR.
3. Itproducesonlybinaryoutputs.
4. Itusesanon-differentiablestepactivationfunction.

Theselimitationsmotivated thedevelopmentof multi-layerneural networksandadvanced learning algorithms.

OverallResult

Theexperimentdemonstratesthat:

- ThePerceptronsuccessfullylearnsfromerrors.
- Weightupdatesmovethedecisionboundarytowardcorrectclassification.
- Thealgorithmconverges forlinearlyseparableproblems.
- Manualorspreadsheetimplementationclearlyshowshow learningoccurs.

ThefinalresultconfirmsthatthePerceptronmodeliseffectiveforsimplebinaryclassification tasks and provides foundational understanding for more advanced neural network models.

7.Conclusion

In this report, a Perceptron model was successfully constructed and trained to solve a binary classificationproblemusingtheANDlogicgatedataset.ThePerceptron,originallyintroduced by**Frank Rosenblatt**in1958, represents one ofthe earliest and most fundamentalmodels in the field of Artificial Neural Networks.

The objective of this experiment was to understand the working mechanism of a single-layer Perceptron and apply the Perceptron Learning Law step-by-step. The model was initialized withzero weights and bias, and the training process was carried out manuallyusing theAND gate truth table. Through iterative updates based on classification error, the model gradually adjusted its parameters until it correctly classified all input patterns.

During the training process, it was observed that the Perceptron follows an error-correction learningmechanism.Wheneverthepredictedoutputdifferedfromthetargetvalue,theweights and bias were updated using the learning rule:

$$\boldsymbol{w_i(new)=w_i(old)+\eta(t-y)x_i}$$
$$\boldsymbol{b(new)=b(old)+\eta(t-y)}$$

These updates shifted and rotated the decision boundary in the input space. After several epochs, the model converged, meaning that no further updates were required and the total classification error became zero.

Thefinallearneddecisionboundary:

$$\boldsymbol{x_1+x_2-1.5=0}$$

successfully separated the two classes of the AND gate dataset. The model achieved 100% accuracyonthetrainingdata,confirmingthattheproblemislinearlyseparableandsuitablefor a single-layer Perceptron.

Thisexperimenthighlightsseveralimportantconcepts:

- Theimportanceofweightsandbiasinclassification
- Theroleoftheactivation function
- Thesignificanceoflearningrate
- Theconceptofconvergence
- The limitationoflinear classifiers

Although the Perceptron is a simple model, it forms the conceptual foundation for more advanced neural network architectures such as multi-layer perceptron and deep learning systems. Understanding the Perceptron provides valuable insight into how modern machine learning models learn from data.