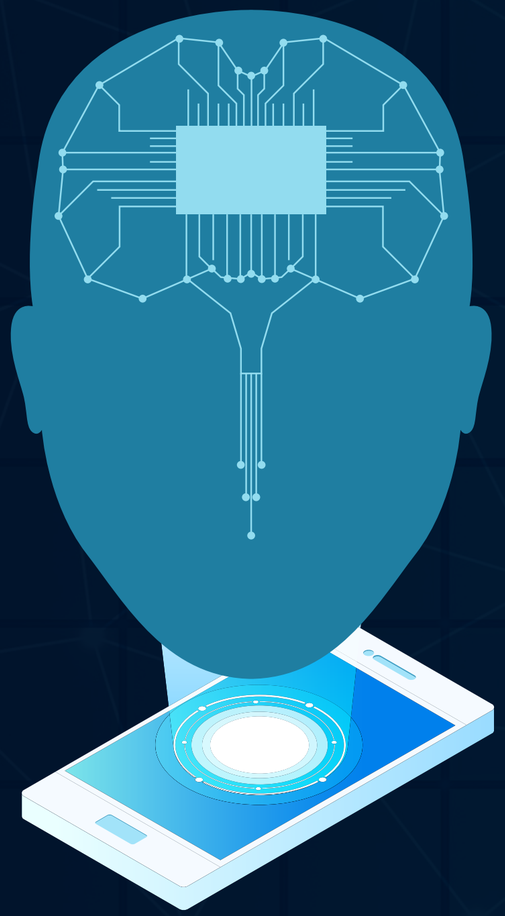
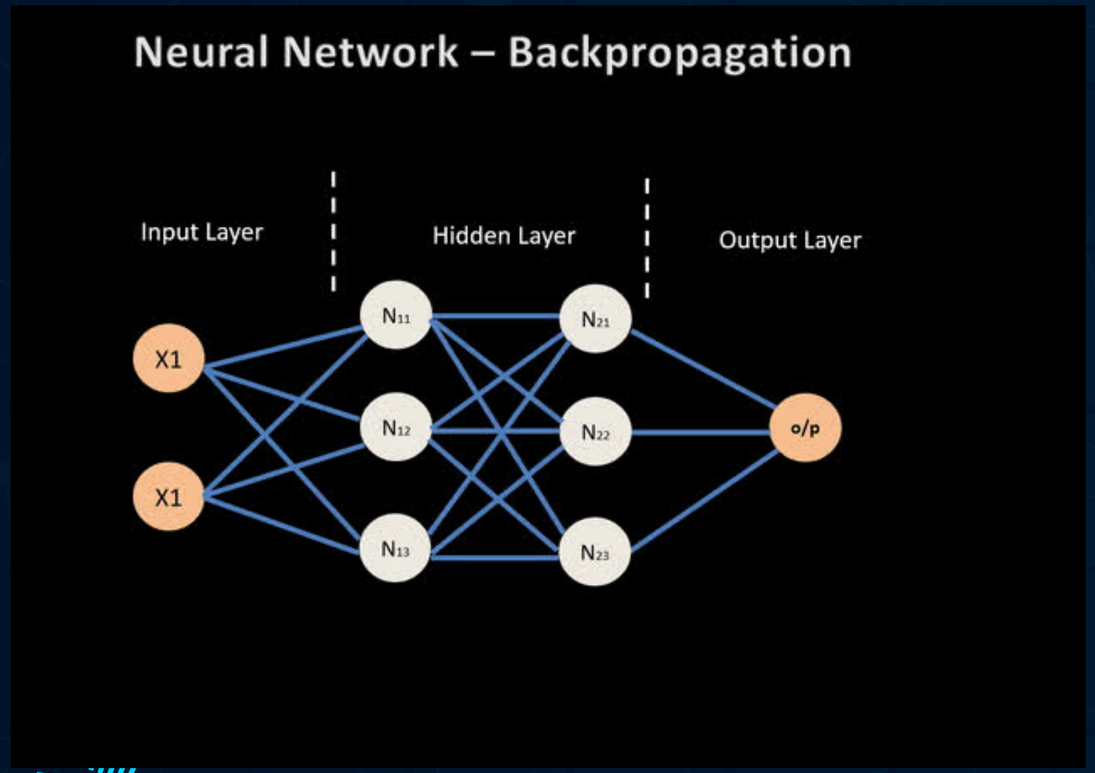
STRATEGIES FOR TRAINING NEURAL NETWORK

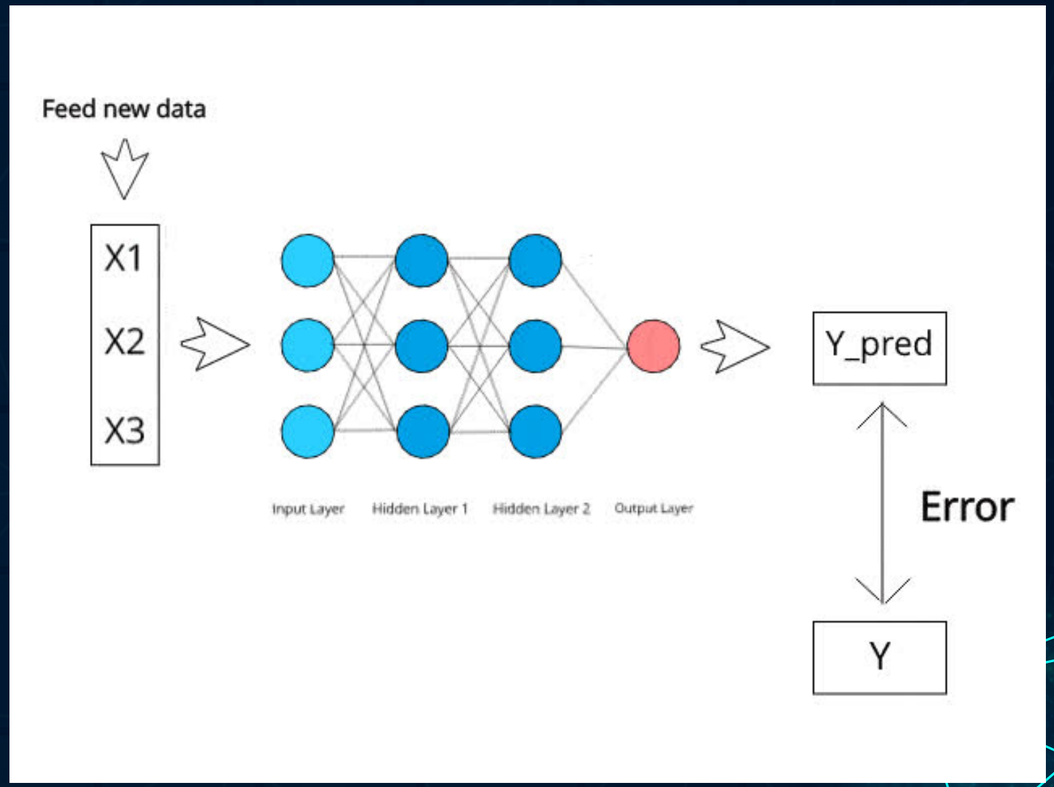


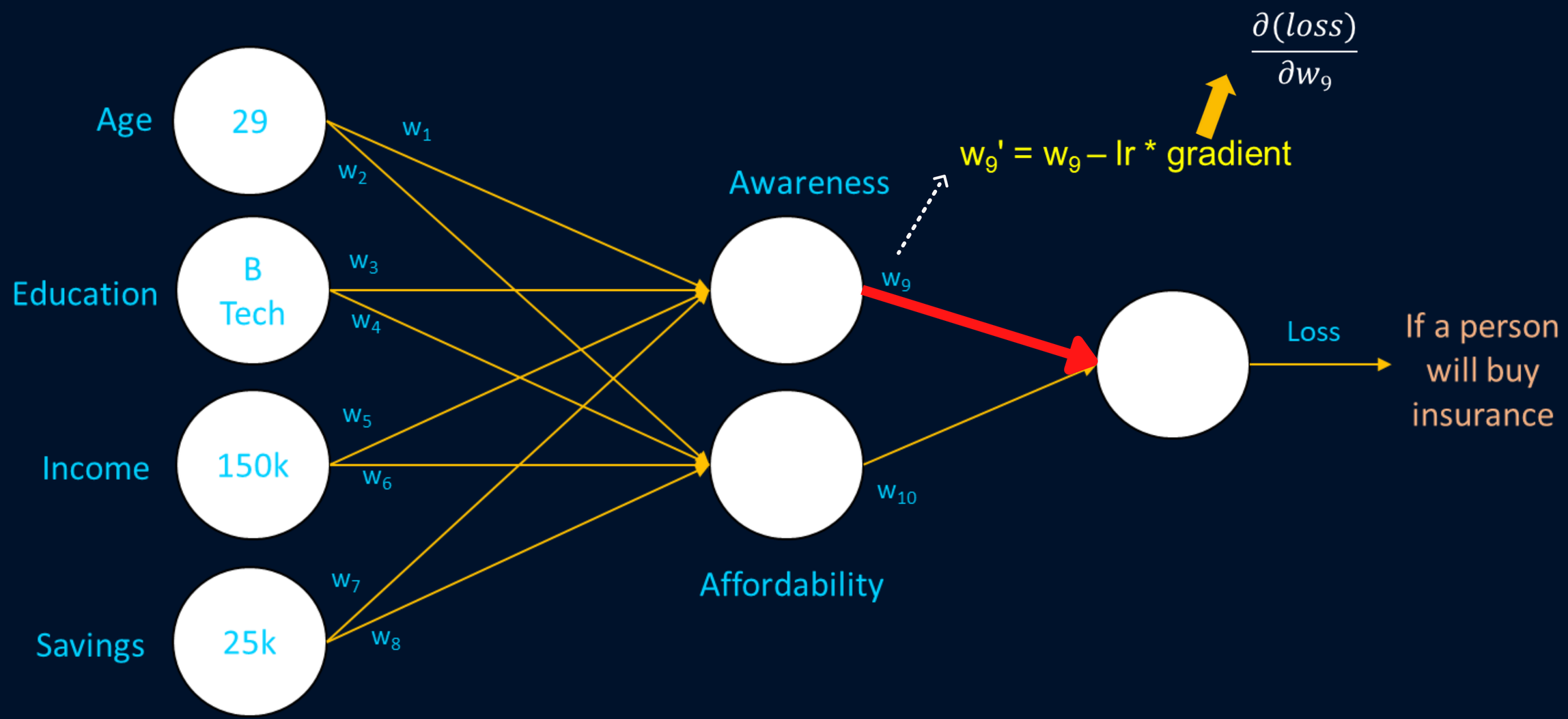
SCOPE

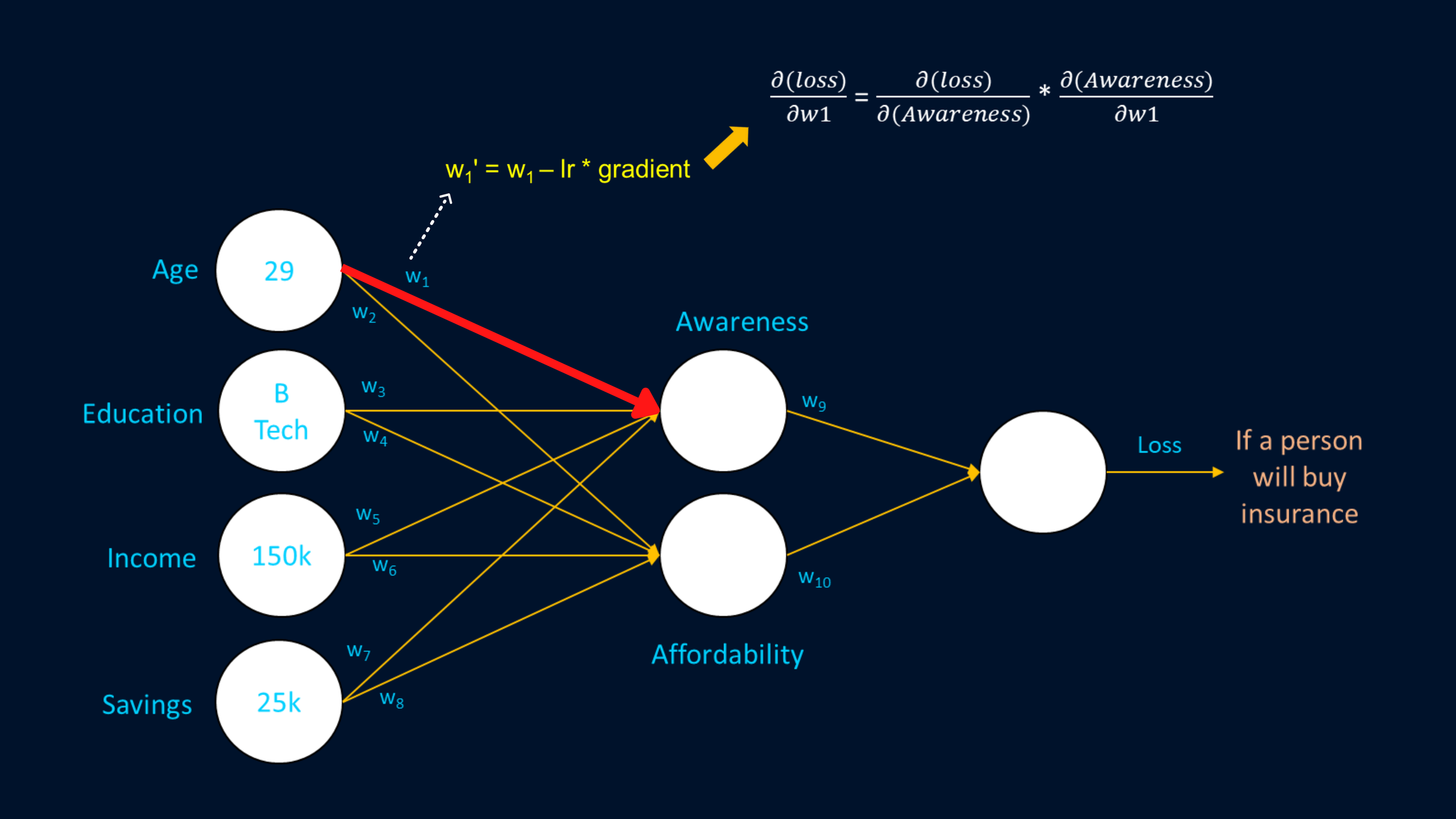
- No 1-size fits all.
- The need to deal with certain issues with the Neural Networks.
- The need to choose the right training parameters and metrics.

IN BACKPROPAGATION



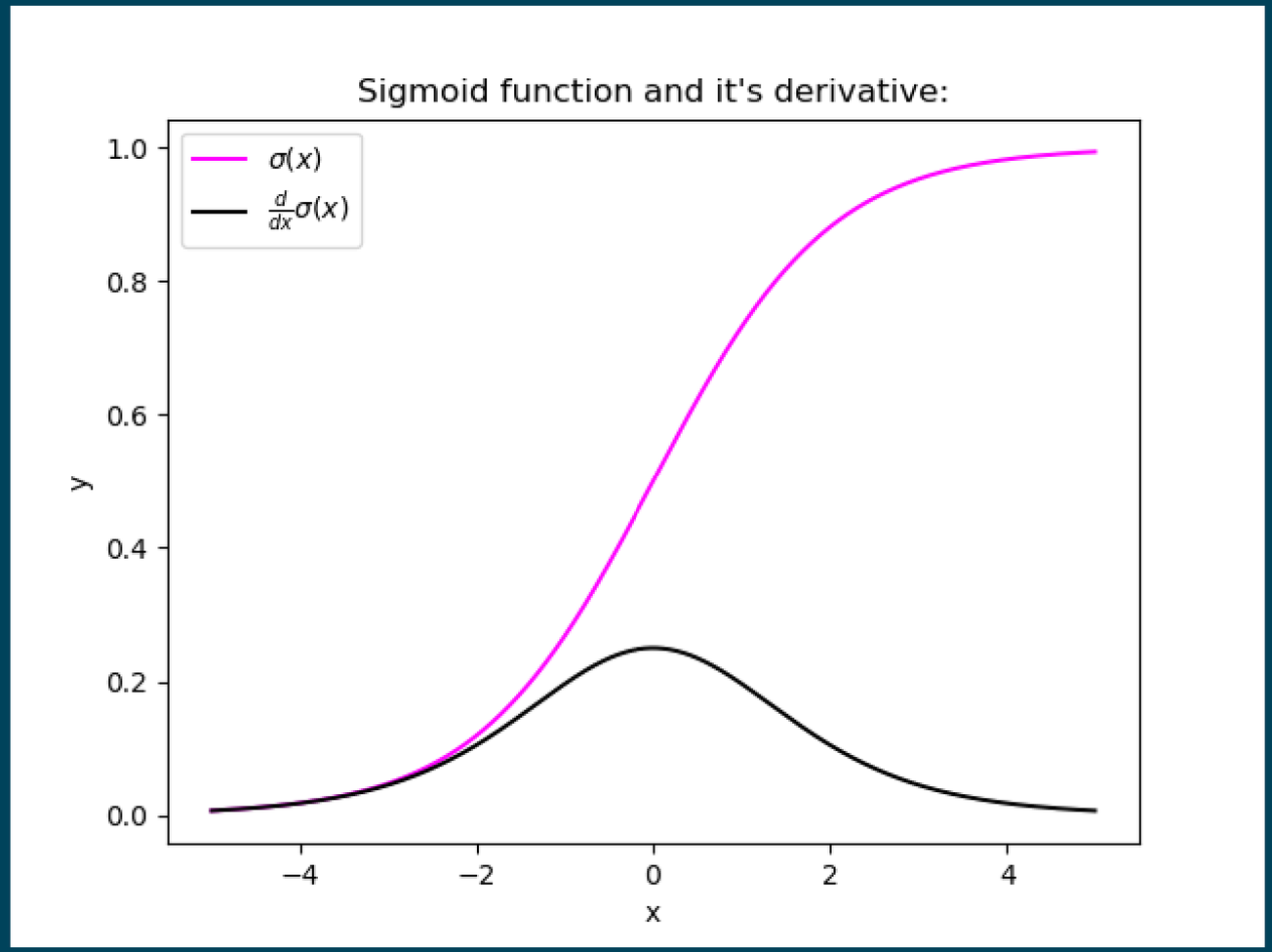



$\frac{\partial(loss)}{\partial w_9}$
$w_9' = w_9 - lr * gradient$
Age
29
Education
B Tech
Income
150k
Savings
25k
w_1
w_2
w_3
w_4
w_5
w_6
w_7
w_8
Awareness
Affordability
w_9
w_{10}
Loss
If a person will buy insurance


$\frac{\partial(loss)}{\partial w1} = \frac{\partial(loss)}{\partial(Awareness)} * \frac{\partial(Awareness)}{\partial w1}$
$w_1' = w_1 - lr * gradient$
Age
29
Education
B Tech
Income
150k
Savings
25k
w_1
w_2
w_3
w_4
w_5
w_6
w_7
w_8
Awareness
Affordability
w_9
w_{10}
Loss
If a person will buy insurance

VANISHING GRADIENT. WHY?

> The gradient of activation function is too small.
> The small gradient diminishes with each passing layer



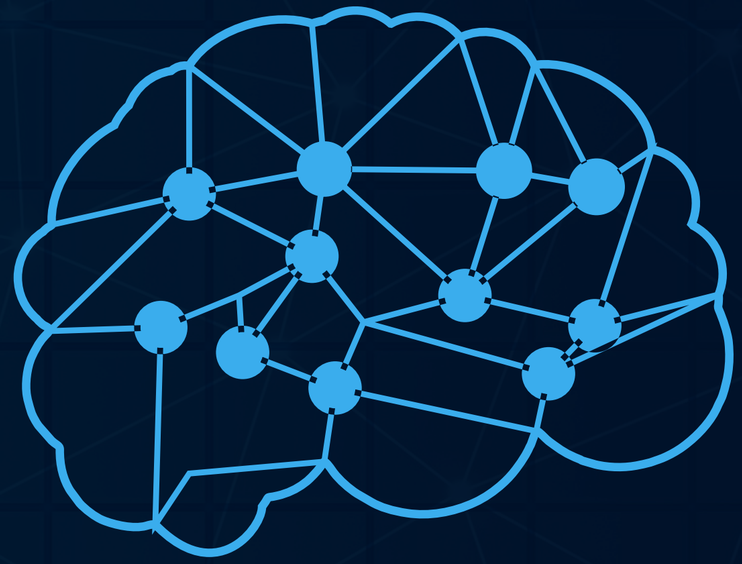
Exploding Gradient	Vanishing Gradient
Model weights grow exponentially.	Model weights shrink exponentially.
The model weights become NaN during training.	The model weights become 0 in the training phase.
The model experiences avalanche learning.	The model learns very slowly and may stagnate.

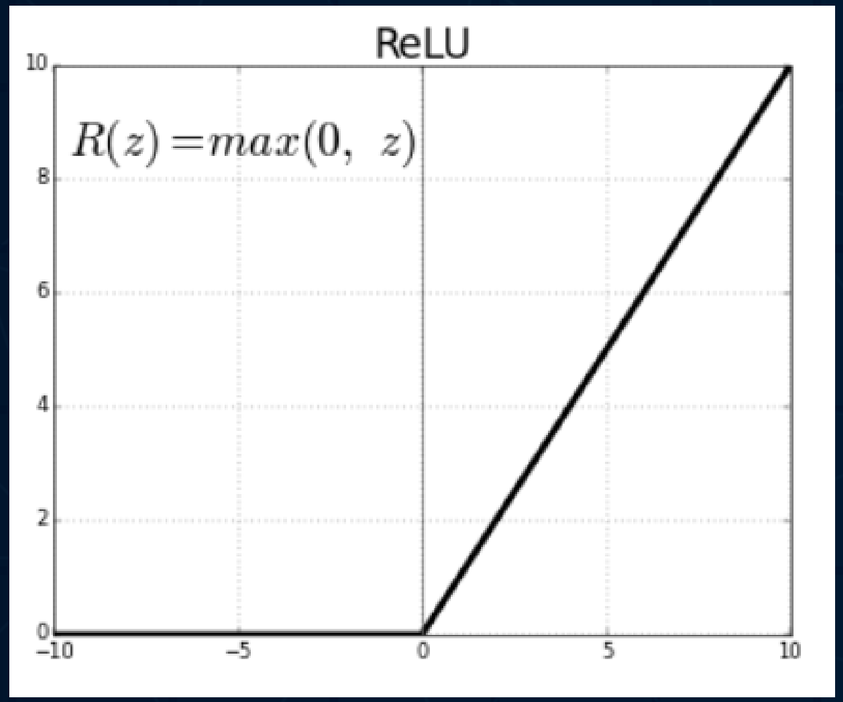
SOLUTIONS

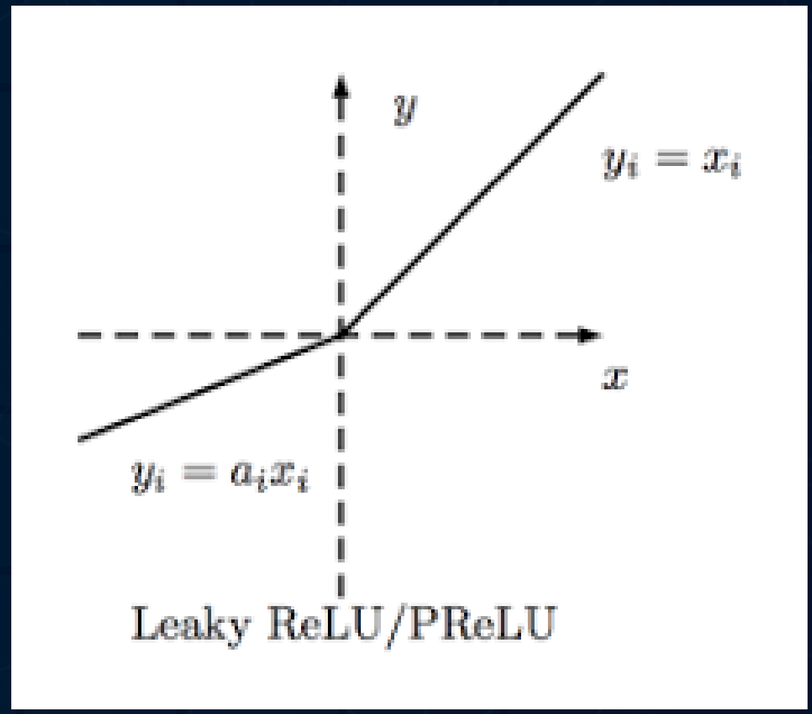
Reducing the number of layers?

SOLUTIONS

USING NON-SATURATING ACTIVATION FUNCTIONS







SOLUTIONS

GRADIENT CLIPPING

ANOTHER TECHNIQUE TO MITIGATE THE EXPLODING GRADIENTS PROBLEM IS TO CLIP THE GRADIENTS

```
optimizer = keras.optimizers.SGD(clipvalue = 1.0)
```

[0.9, 100.0] → **[0.9, 1.0]**

SOLUTIONS

GRADIENT CLIPPING

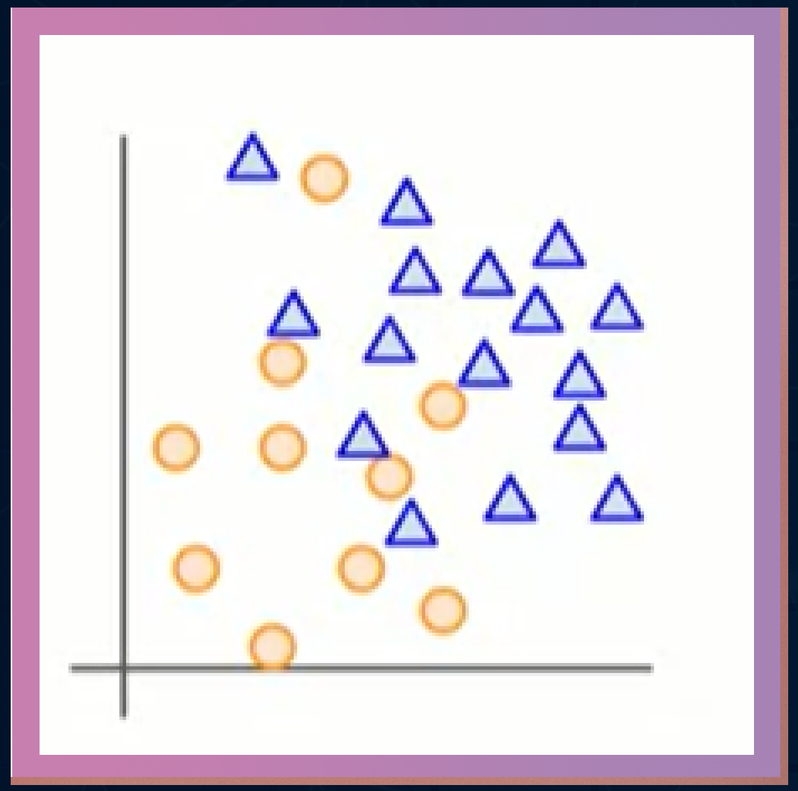
ANOTHER TECHNIQUE TO MITIGATE THE EXPLODING GRADIENTS PROBLEM IS TO CLIP THE GRADIENTS

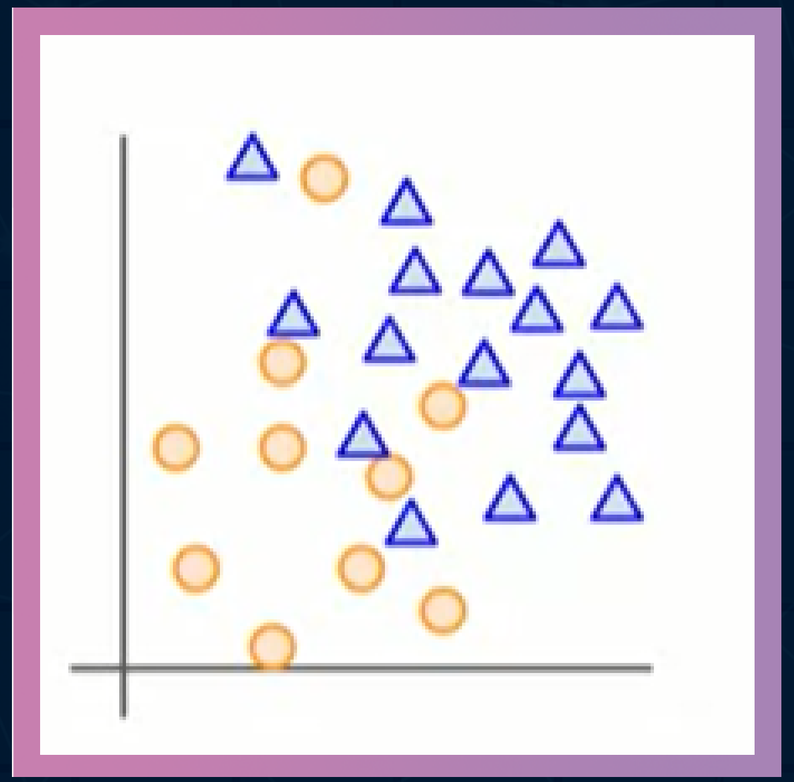
```
optimizer = keras.optimizers.SGD(clipnorm = 1.0)
```

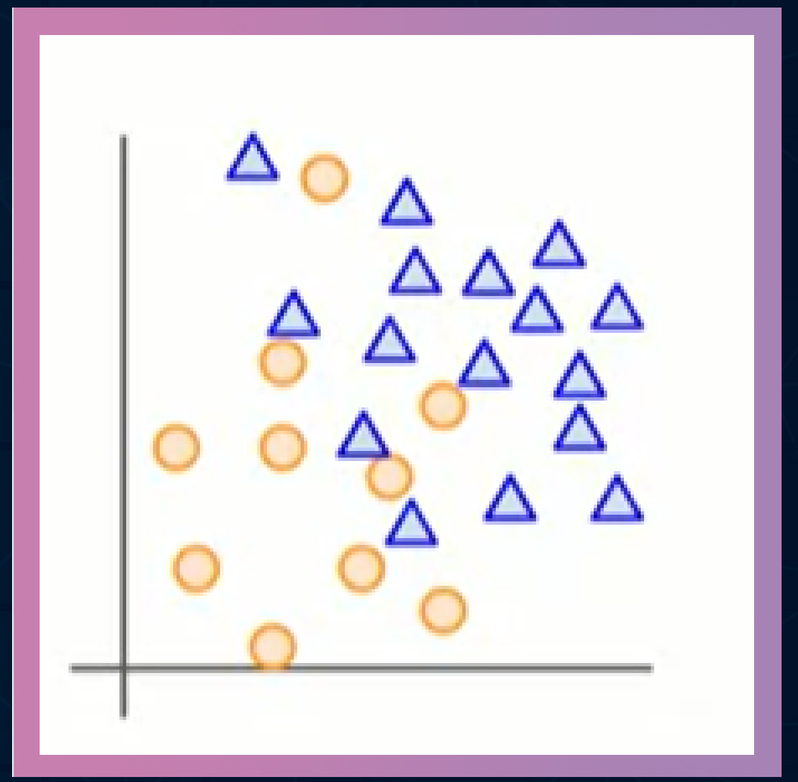
[0.9, 100.0] → **[0.0089, 0.9995]**

ML PROBLEM

UNDERFITTING VS OVERFITTING



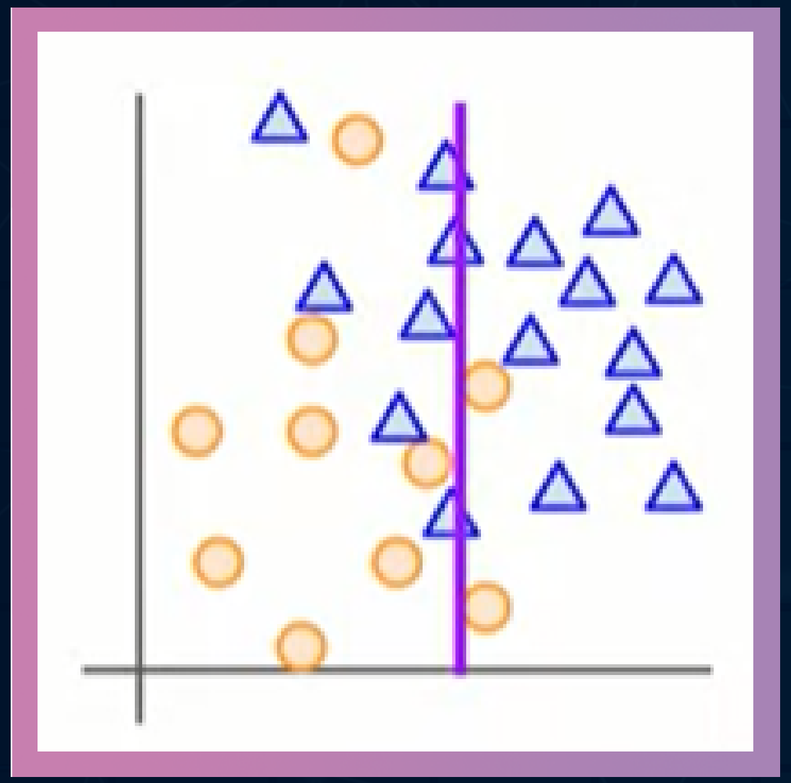


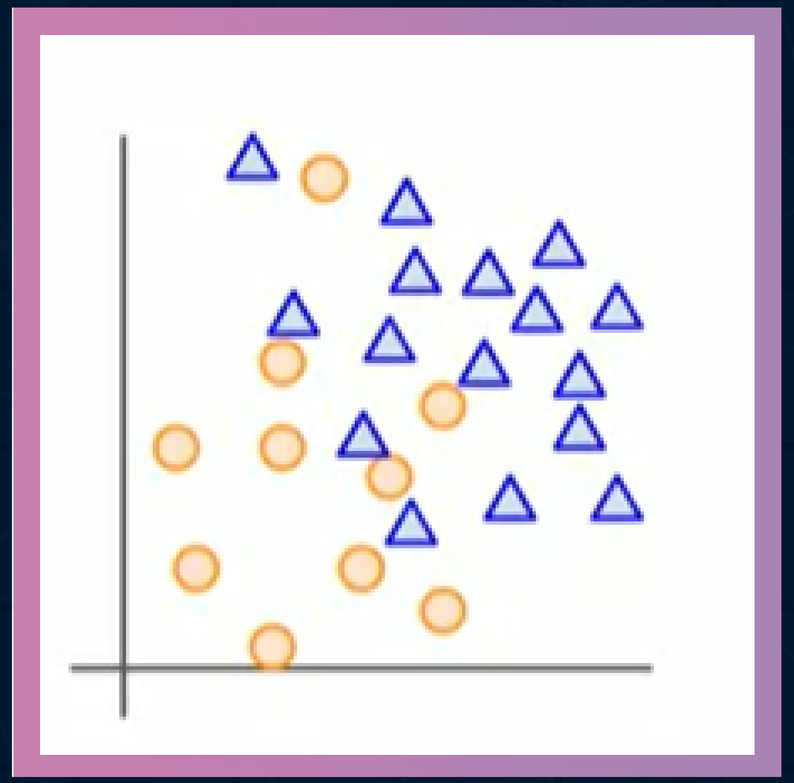


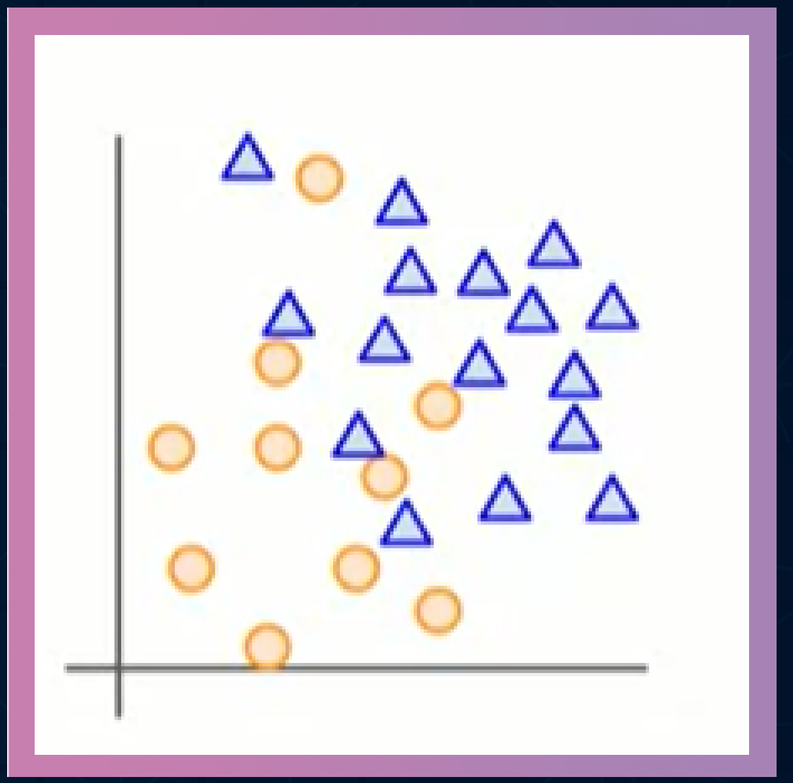
ML PROBLEM

UNDERFITTING VS OVERFITTING

Model fails to capture trends in the data



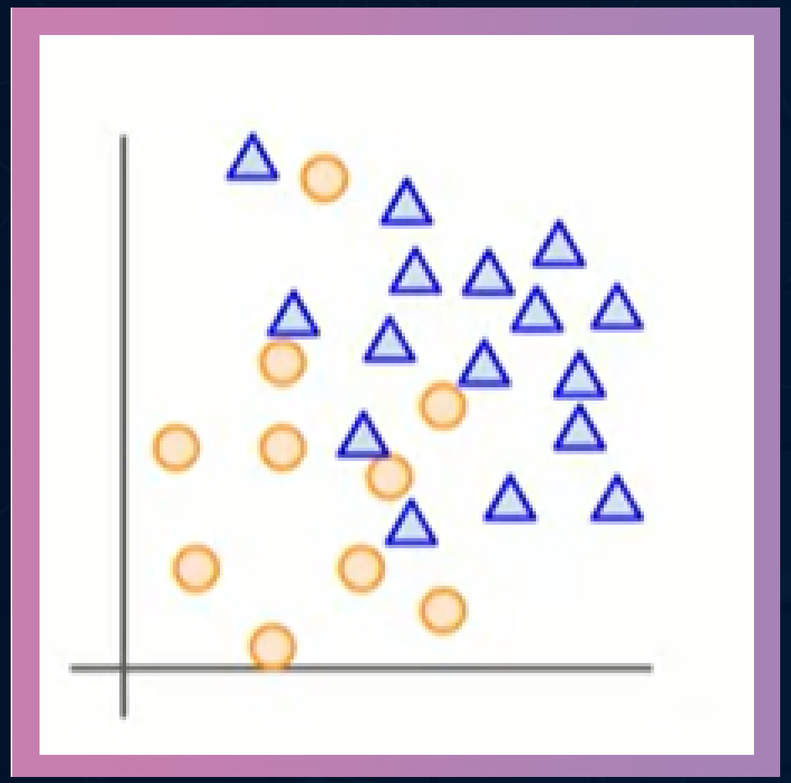


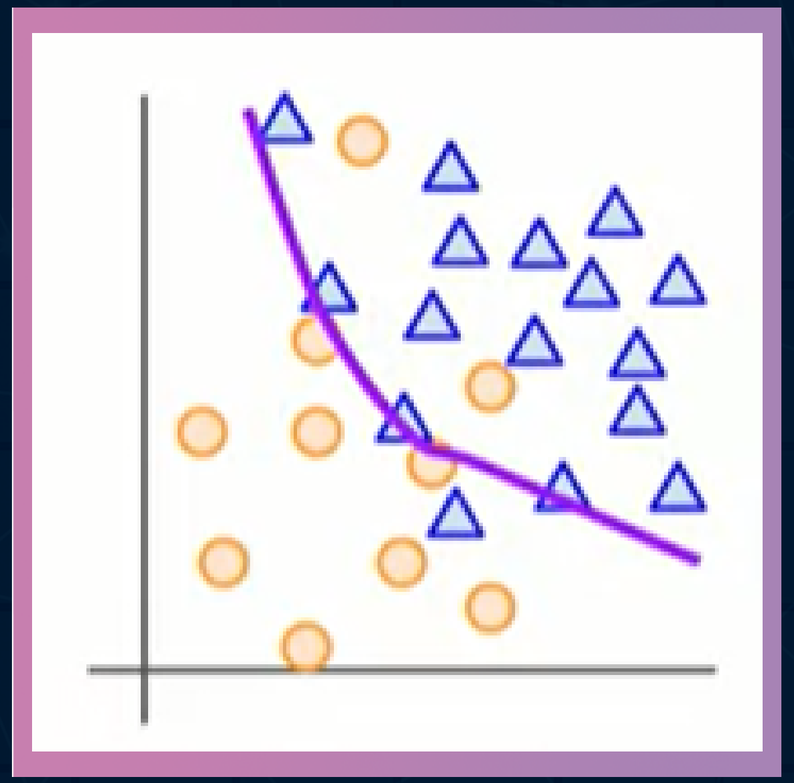


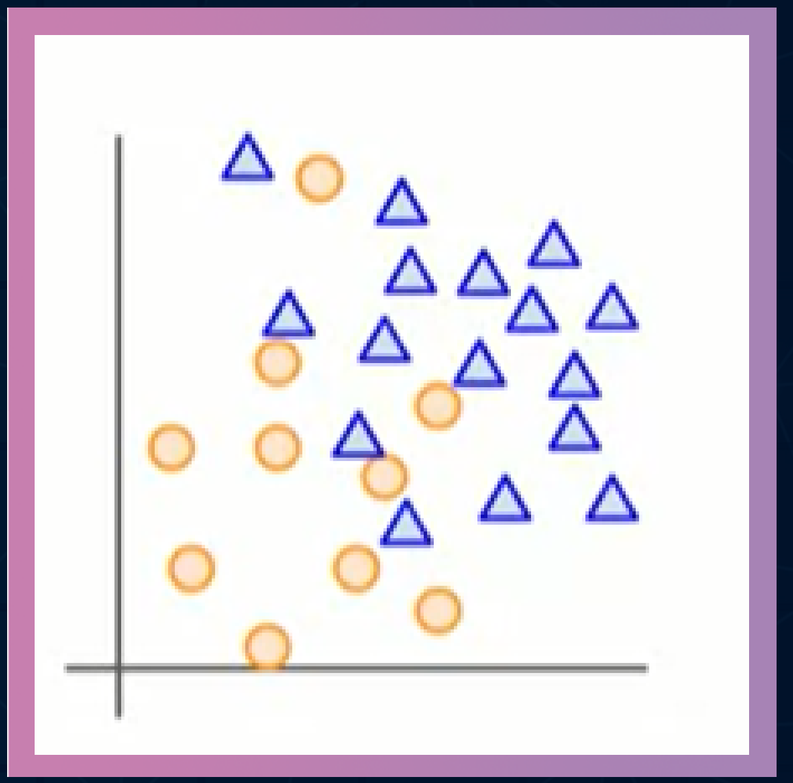
ML PROBLEM

UNDERFITTING VS OVERFITTING

Model captures the trends and
can generalize to unseen data



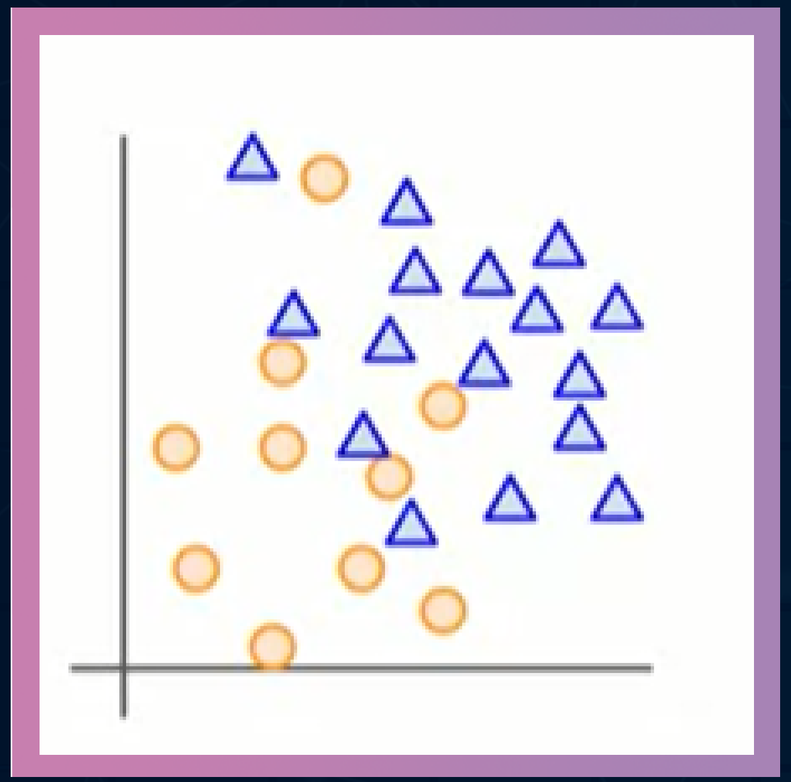


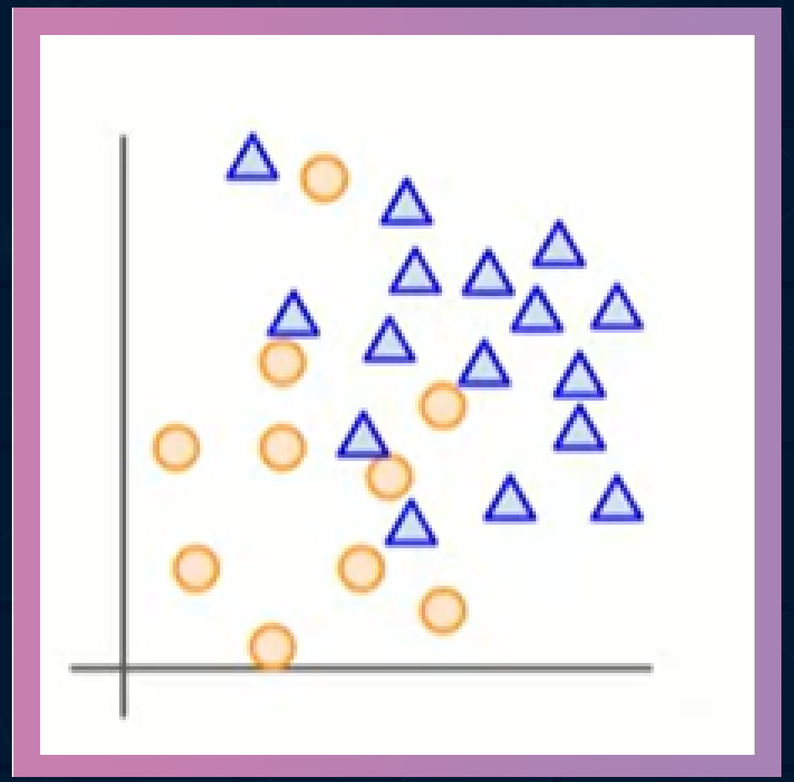


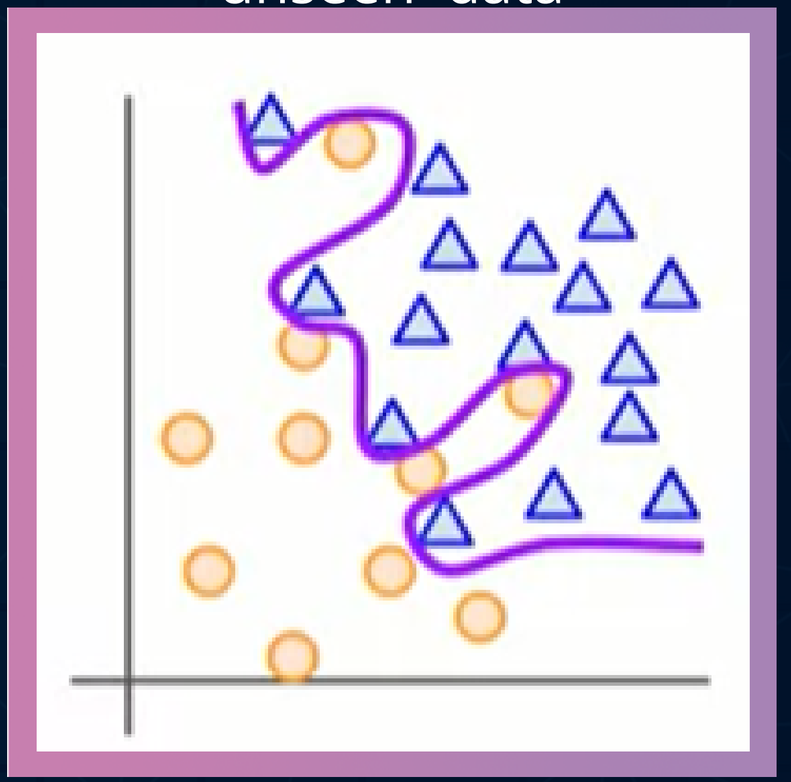
ML PROBLEM

UNDERFITTING VS OVERFITTING

Model captures training data trends but cannot generalize to unseen data



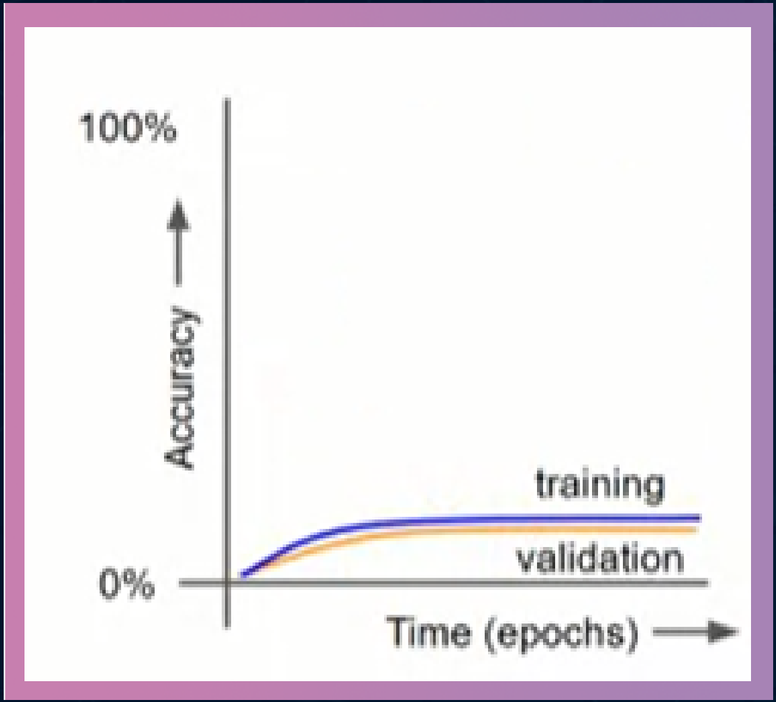




ML PROBLEM

Spotting the problem

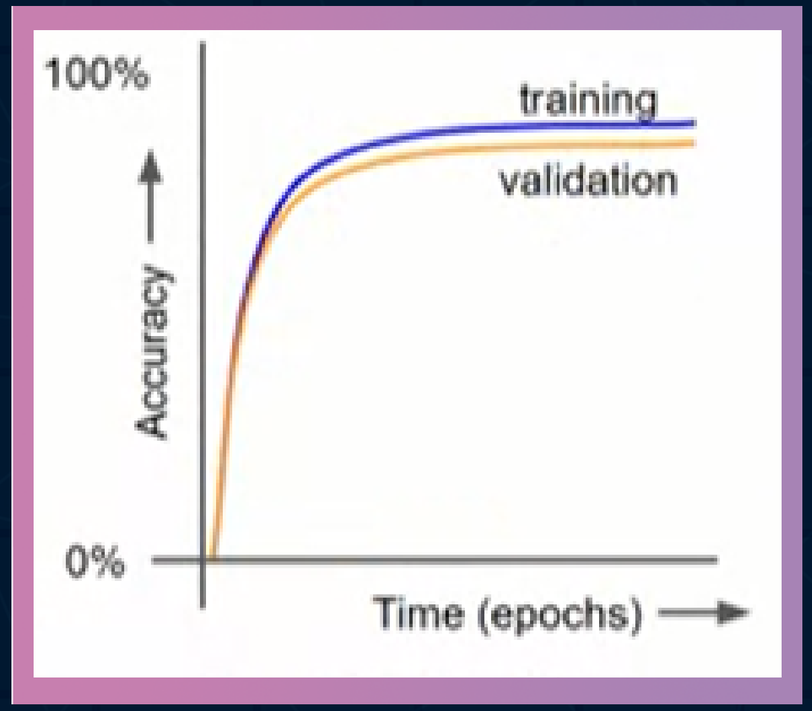
Model performs poorly on training and validation data



ML PROBLEM

Spotting the problem

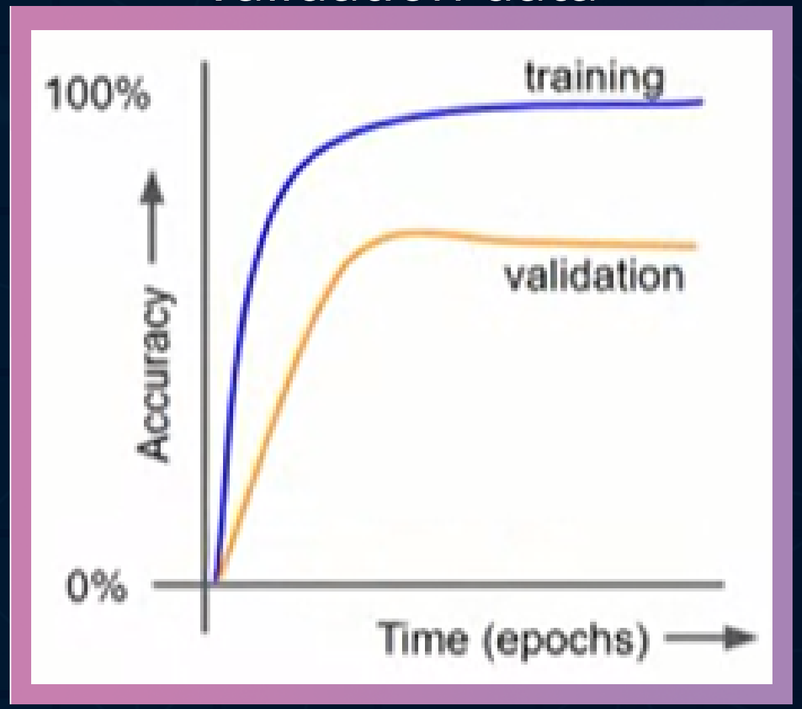
Model generalizes well from training to validation data



ML PROBLEM

Spotting the problem

Model predicts training data well but fails to generalize to validation data



UNDERFITTING

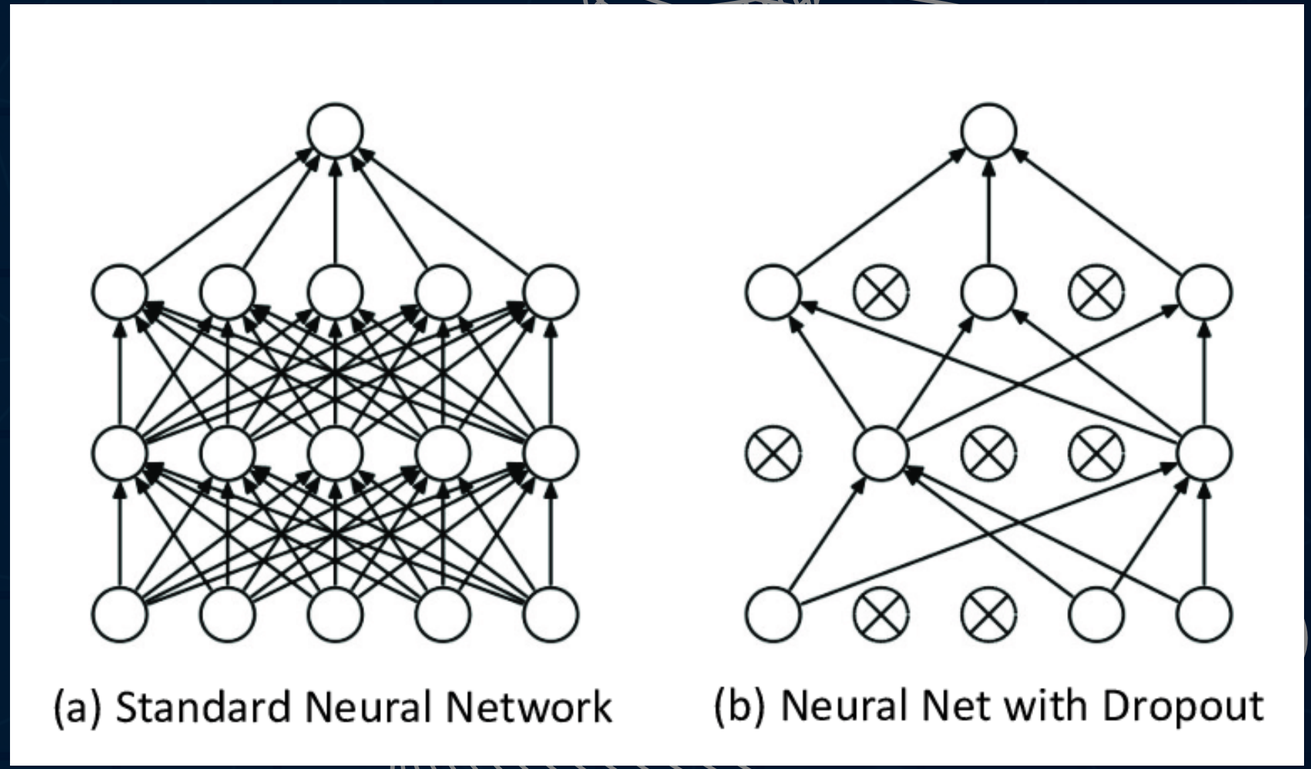
SOLUTIONS

- Train more data
- Try different features or more features
- Train for longer time
- Try a more complex model(more layers, more nodes,etc)

OVERFITTING

SOLUTIONS

- Get more data
- Early stopping
- Reduce model complexity
- Add regularization terms
- Cross Validation
- Ensembling
- Dropout layers(for neural networks)



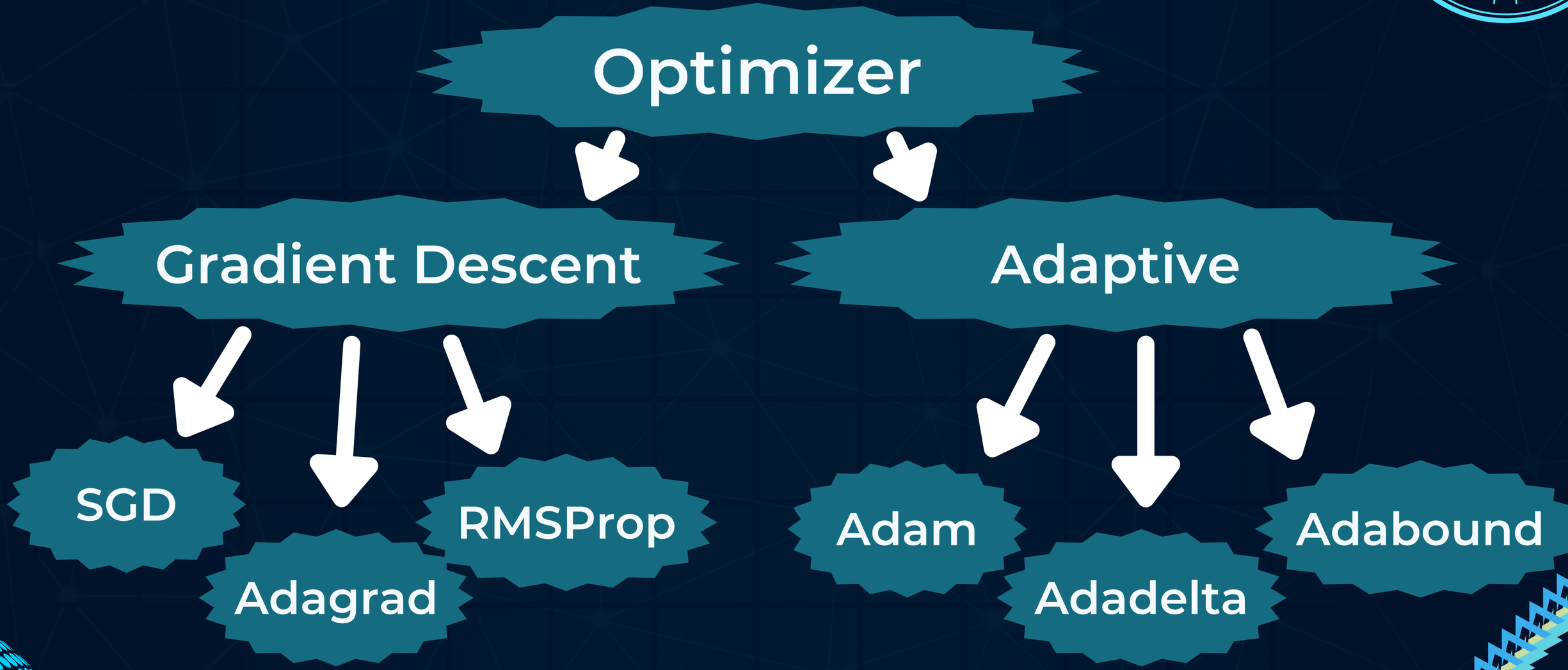
HYPERPARAMETERS

Learning Rate

- The amount by which the weights are updated.
- Neither too small nor too big
- Using Adaptive Learning rates

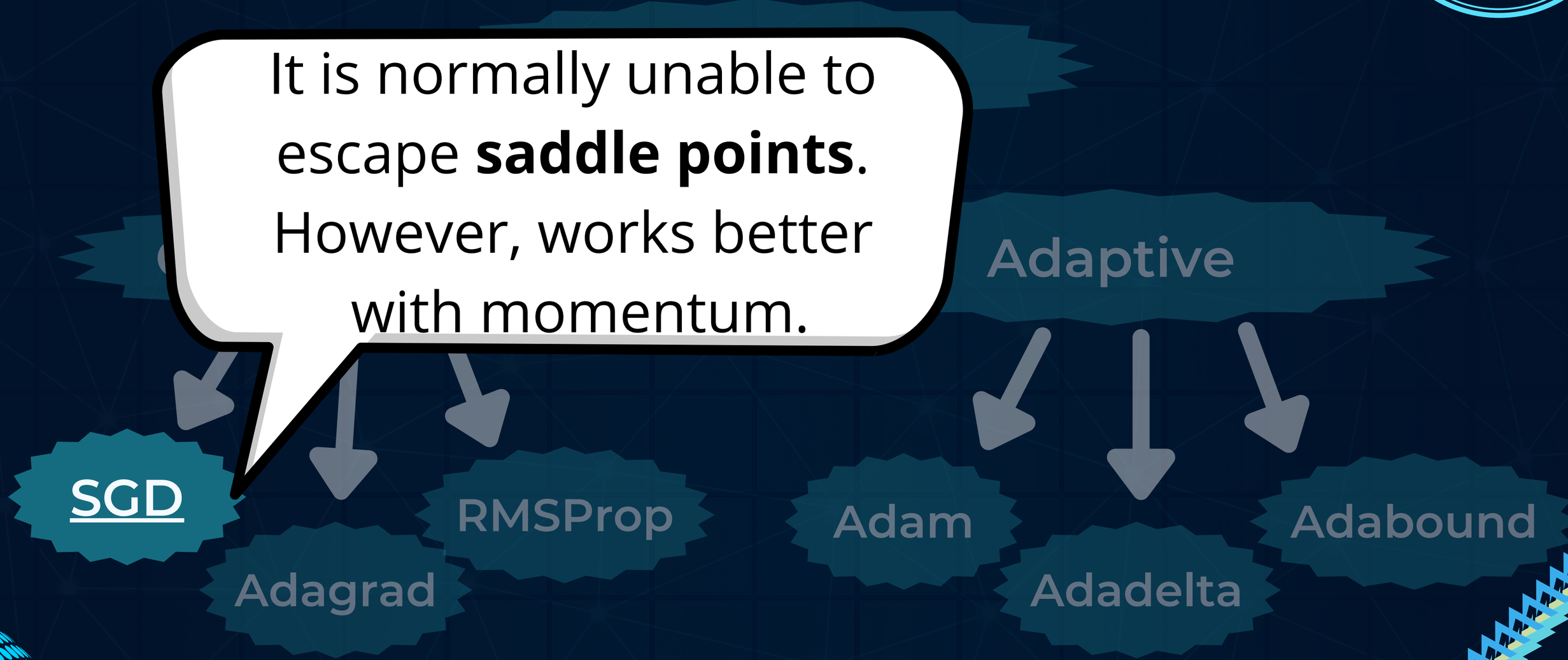
When you set the learning rate to zero

HYPERPARAMETERS

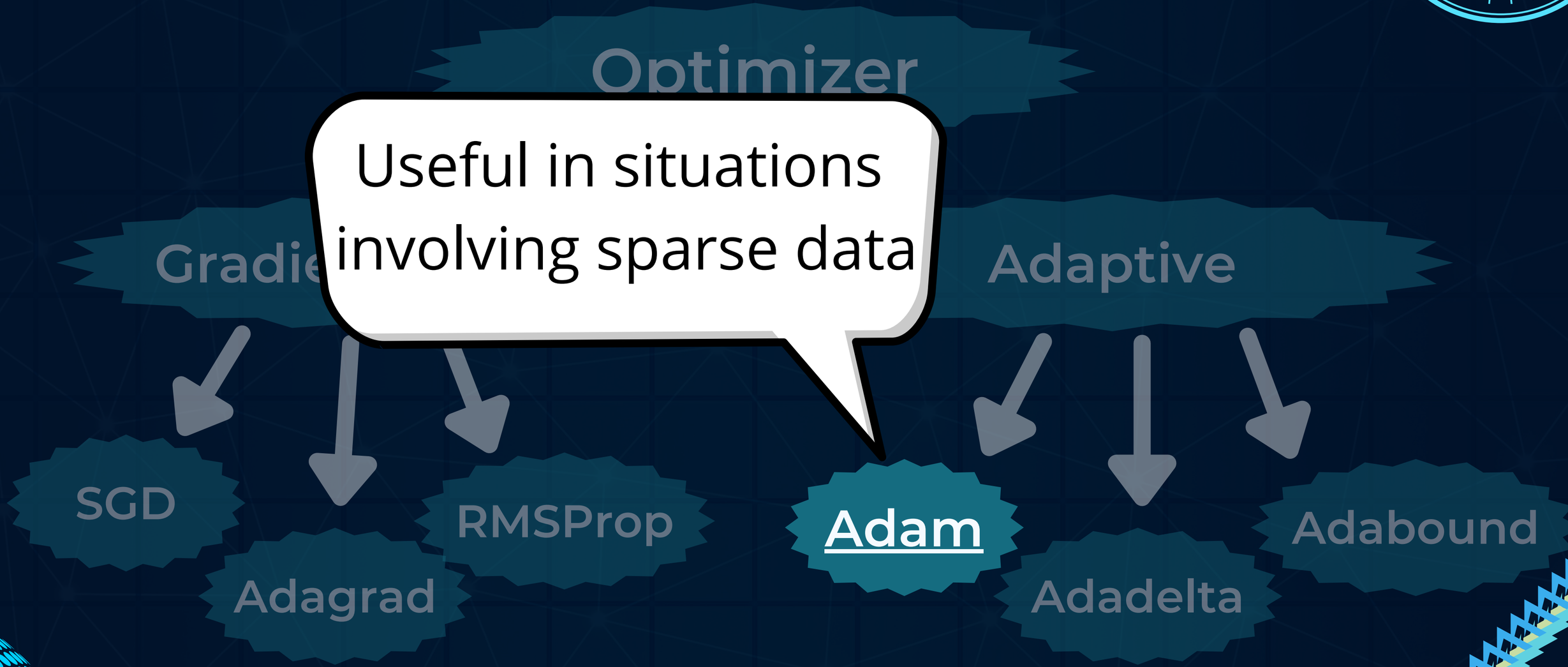


HYPERPARAMETERS

It is normally unable to escape **saddle points**. However, works better with momentum.



HYPERPARAMETERS



HYPERPARAMETERS

Batch Size

Batch_Size
||
100

1050

10 epoch each of
batch size 100

+

The last epoch containing
50 samples

HYPERPARAMETERS

Batch Size

Advantage of using batch_size < No of samples

- Less Memory needed
- Trains Network faster

HYPERPARAMETERS

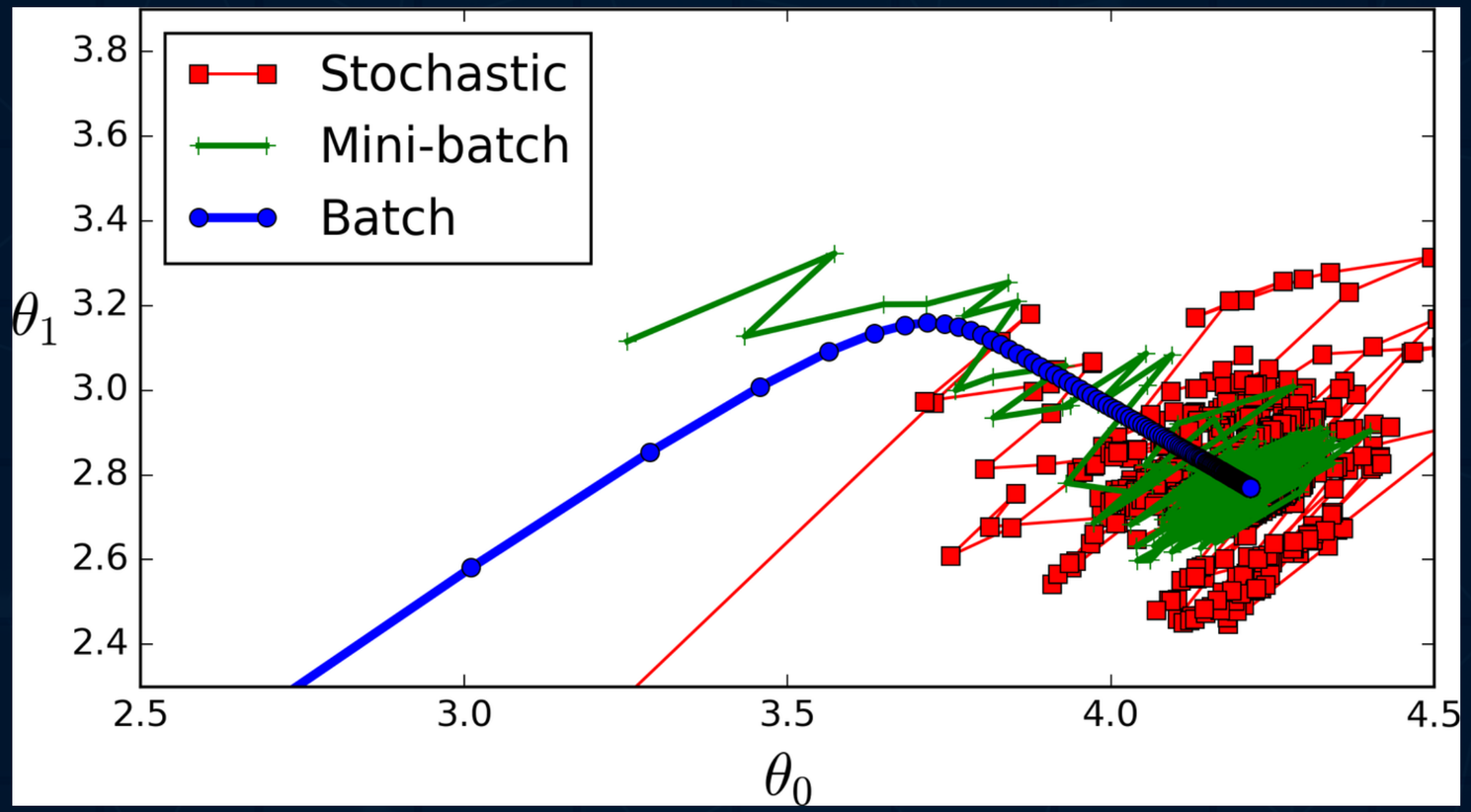
Batch Size

Disadvantage of using batch_size < No of samples

- Accuracy decreases as well.

HYPERPARAMETERS

Batch Size

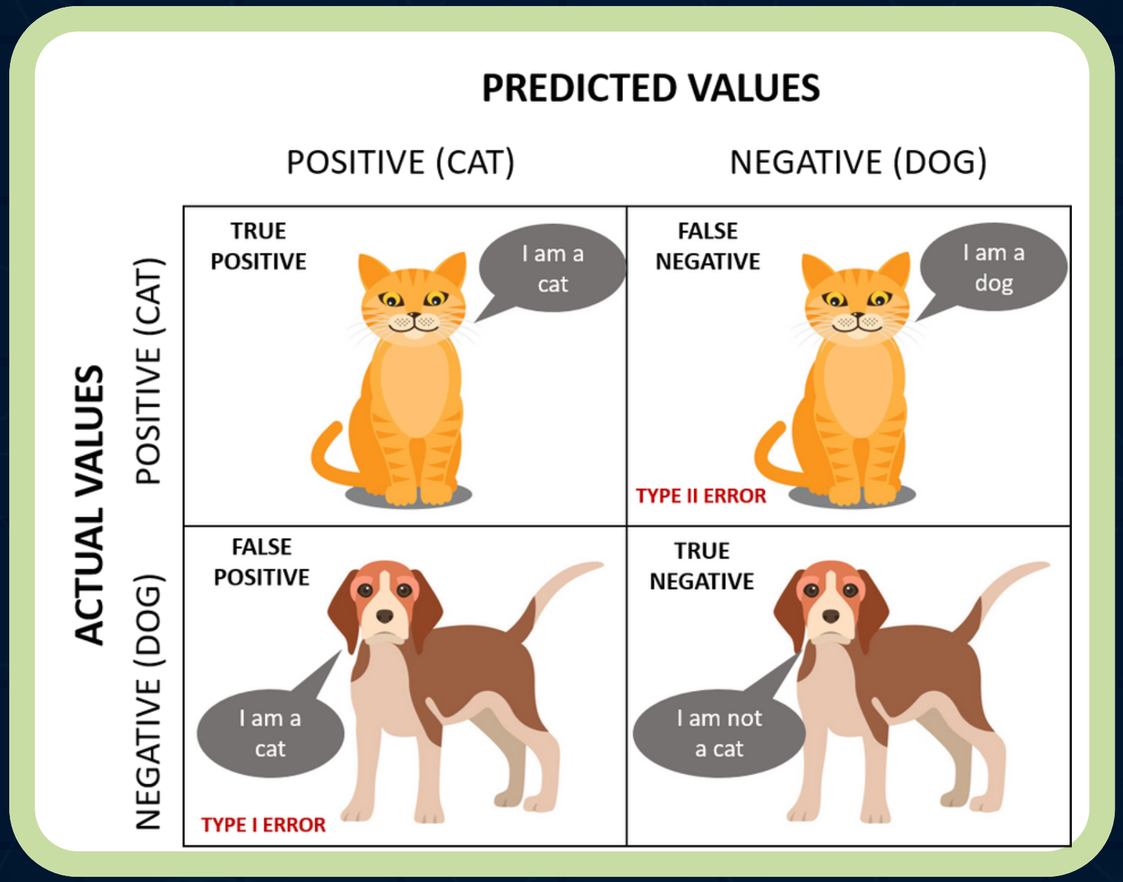


EVALUATION METRICS

- The idea of building machine learning models works on a constructive feedback principle.
- The choice of metric completely depends on the type of model and the implementation plan of the model.

EVALUATION METRICS

Confusion Matrix



EVALUATION METRICS

Confusion Matrix

Accuracy **Precision** **NPV** **Recall** **Specificity**

Confusion Matrix		Target			
		Positive	Negative		
Model	Positive	a	b	*Positive Predictive Value*	a/(a+b)
	Negative	c	d	*Negative Predictive Value*	d/(c+d)
		Sensitivity	*Specificity*	**Accuracy** = (a+d)/(a+b+c+d)	
		a/(a+c)	d/(b+d)		

EVALUATION METRICS

F 1 Score

Here we are trying to get the best precision and recall at the same time

$$F_1 = \left(\frac{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}.$$

EVALUATION METRICS

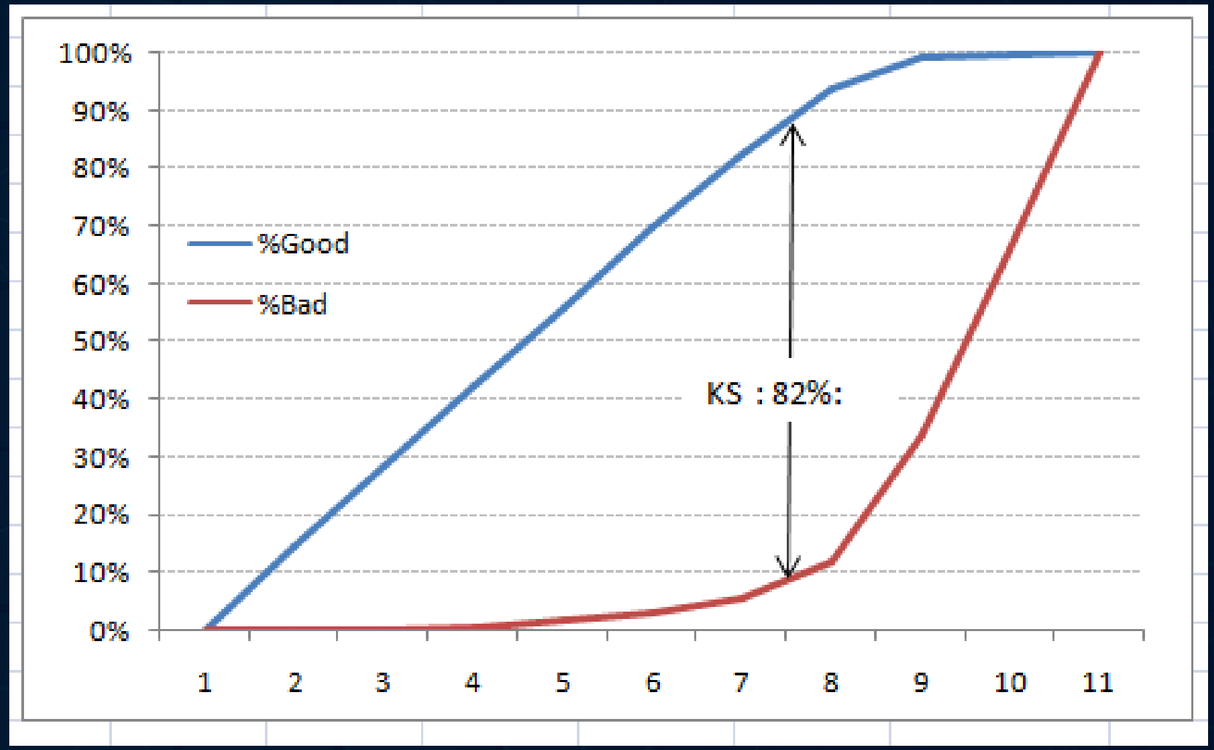
Kolmogorov-Smirnov Test

- This measures the performance of a classification model.
- K-S value ranges from 0 to 100.

EVALUATION METRICS

Kolmogorov-Smirnov Test

We can also plot the %Cumulative Good and Bad to see the maximum separation.



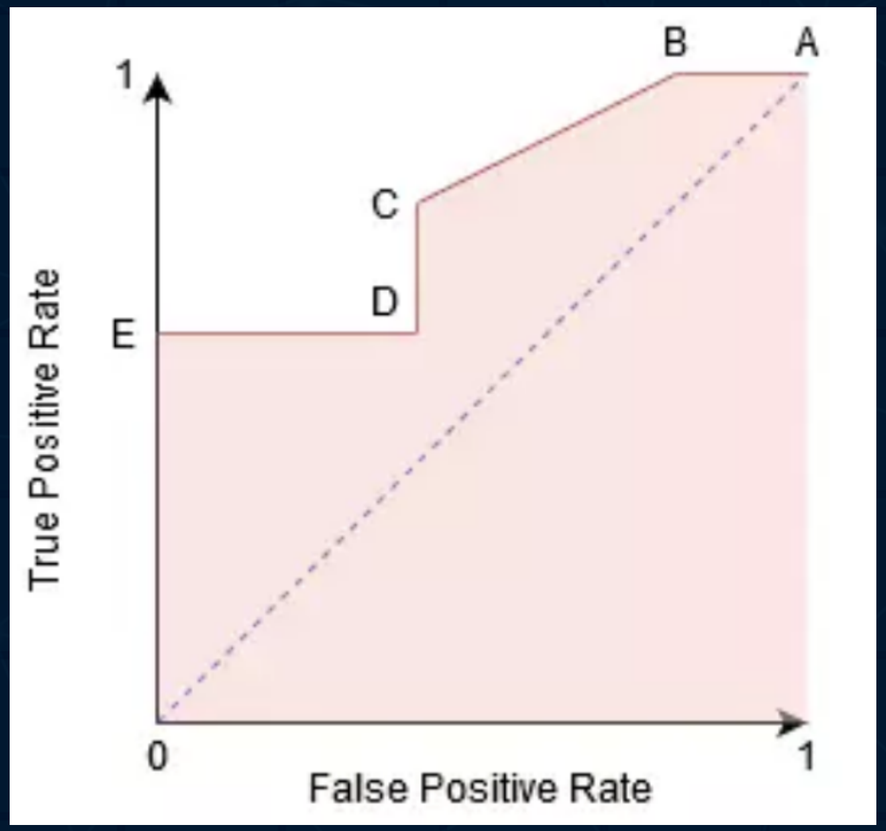
EVALUATION METRICS

AUC-ROC

- It is a probability curve of TPR against FPR.
- The AUC measures the ability of a classifier to differentiate between target classes.

EVALUATION METRICS

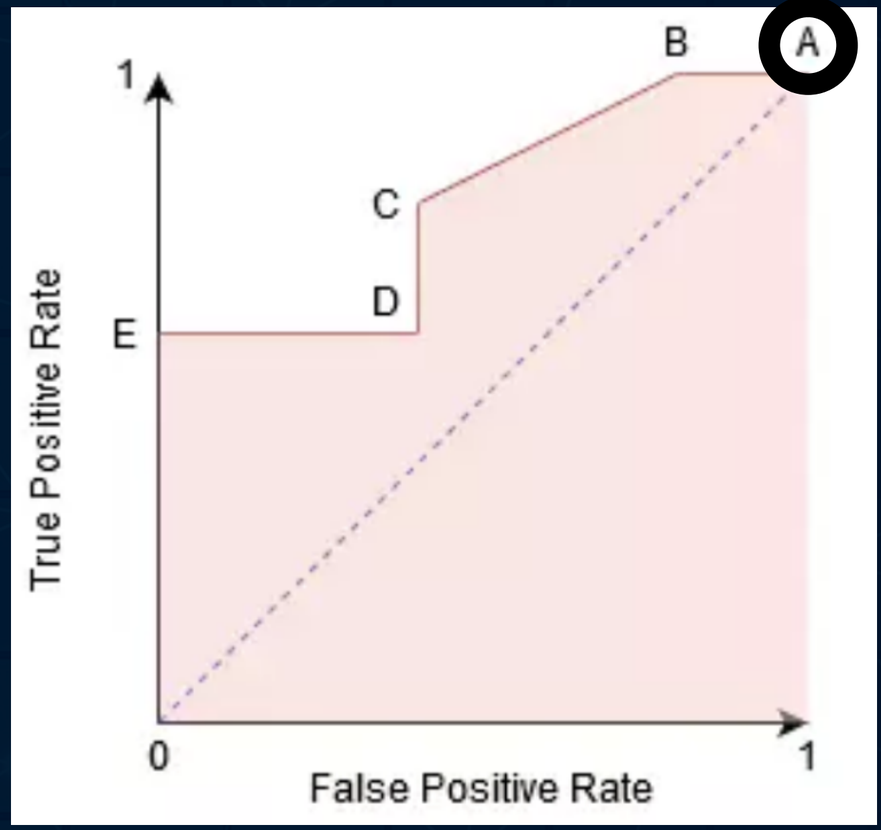
AUC-ROC



EVALUATION METRICS

AUC-ROC

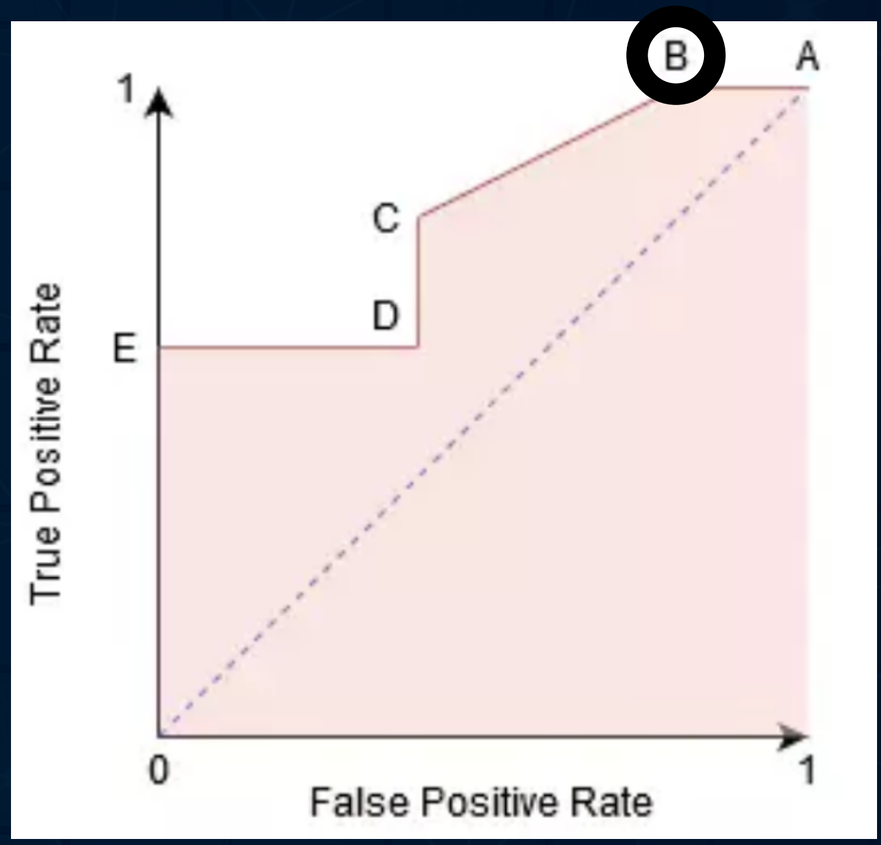
- Point A is where the sensitivity is highest and specificity is lowest.



EVALUATION METRICS

AUC-ROC

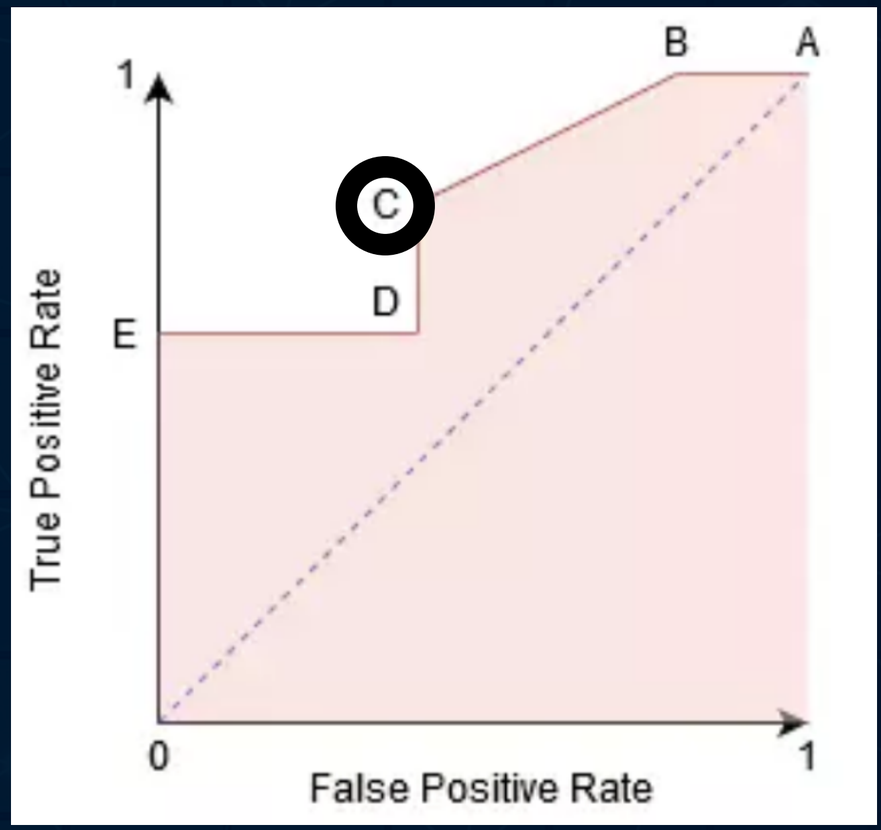
- Point B has the same sensitivity as point A, but higher specificity.



EVALUATION METRICS

AUC-ROC

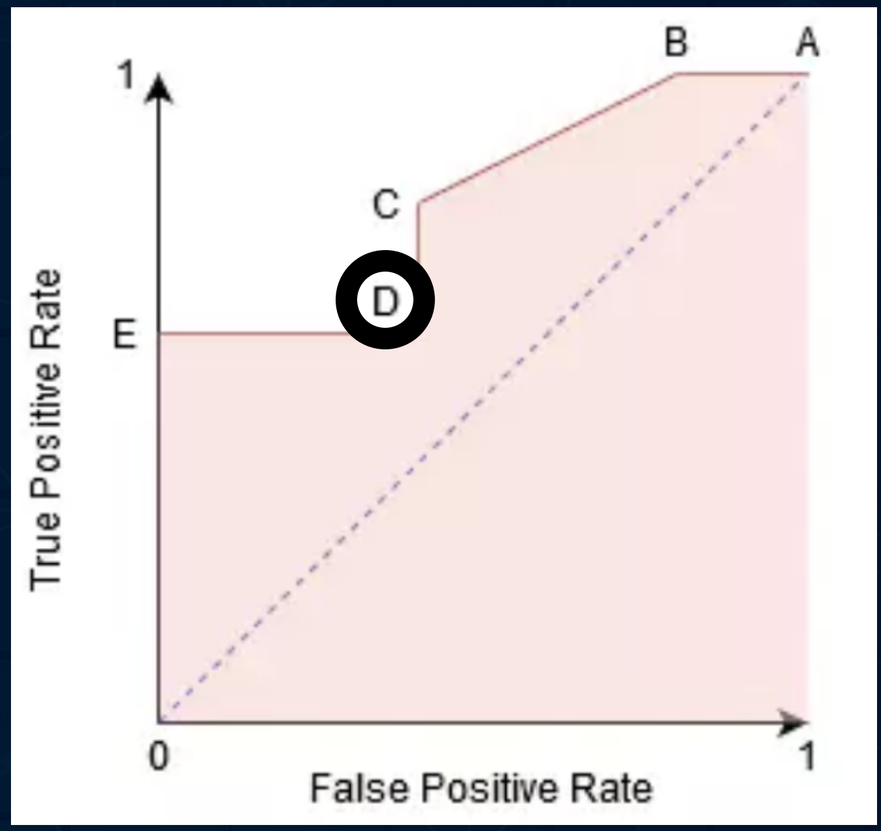
- Point C has lower sensitivity , but higher specificity than point B.



EVALUATION METRICS

AUC-ROC

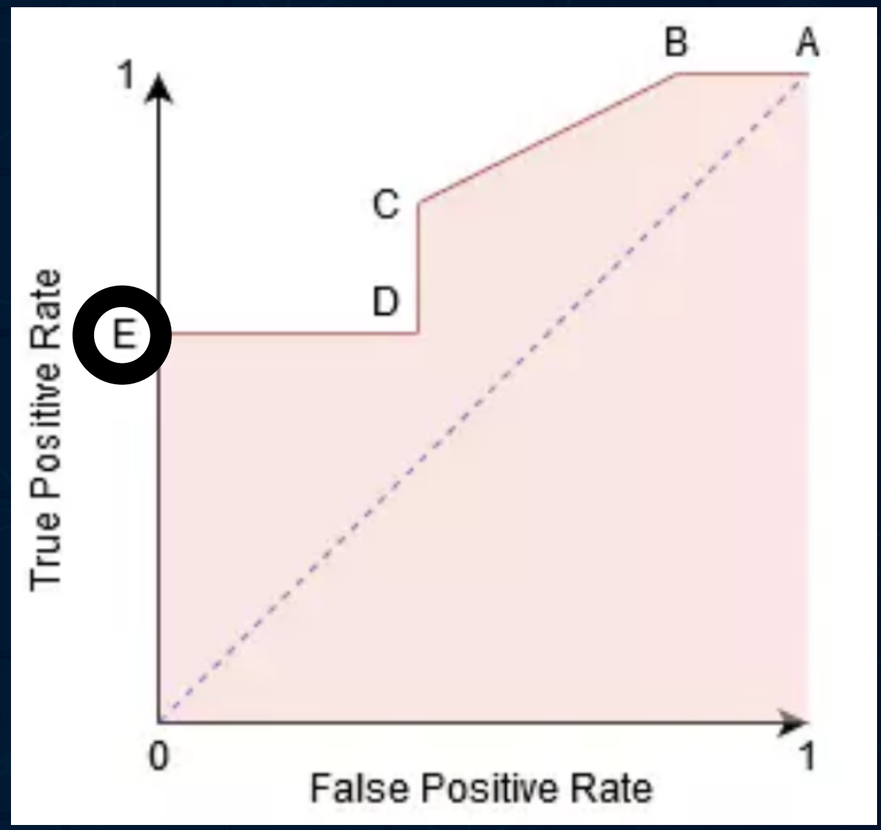
- Point D has the same specificity as point C, but lower sensitivity.



EVALUATION METRICS

AUC-ROC

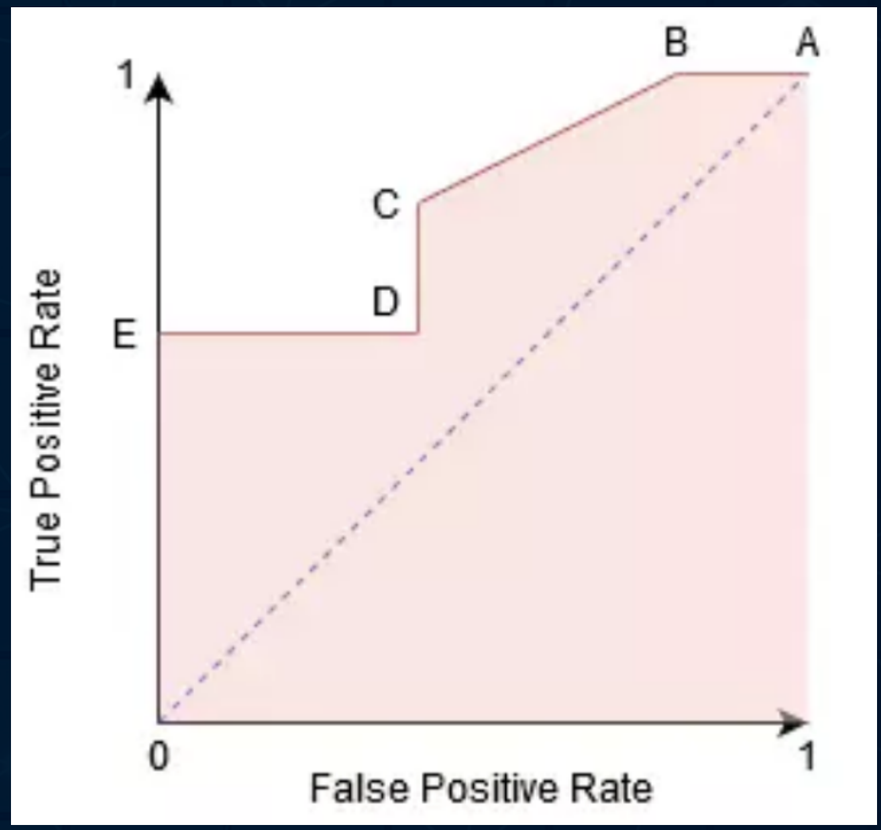
- Point E has the highest specificity.



EVALUATION METRICS

AUC-ROC

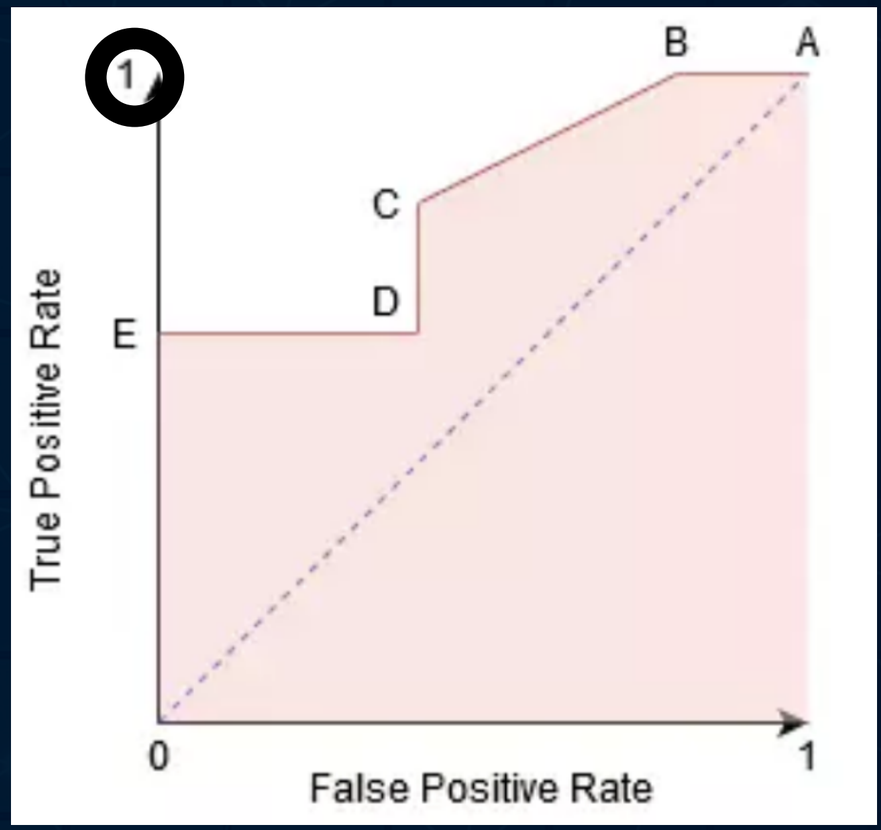
- Going by this logic, can you guess where the point corresponding to a perfect classifier would lie on the graph?



EVALUATION METRICS

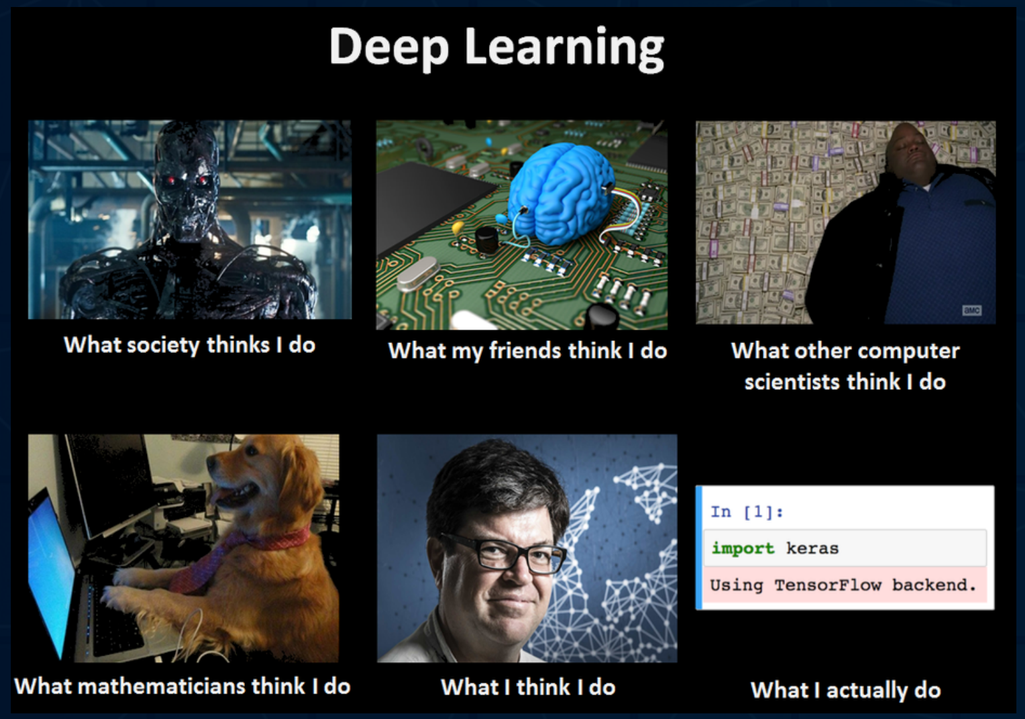
AUC-ROC

- Going by this logic, can you guess where the point corresponding to a perfect classifier would lie on the graph?



- Top left corner of the ROC curve

THANKS!


Deep Learning
What society thinks I do
What my friends think I do
What other computer scientists think I do
What mathematicians think I do
What I think I do
In [1]:
import keras
Using TensorFlow backend.
What I actually do