Session 24 ML

18 October 2024 08:16 PM

Regularization

* Regularization is a technique used in machine learning & deeplearning to prevent overfitting & improve the generalization performance of a model. It involves adding penalty term to the loss function during training.

* The Penalty discourage the model for becoming too complex.

① L1 → Regularization (Lasso)

→ Least shrinkage and selection operation

→ L1 regularization tries to estimate the media of data

→ A regression model that uses L1 regularization technique is called Lasso Regression.

② L2 Regularization (Ridge regression)

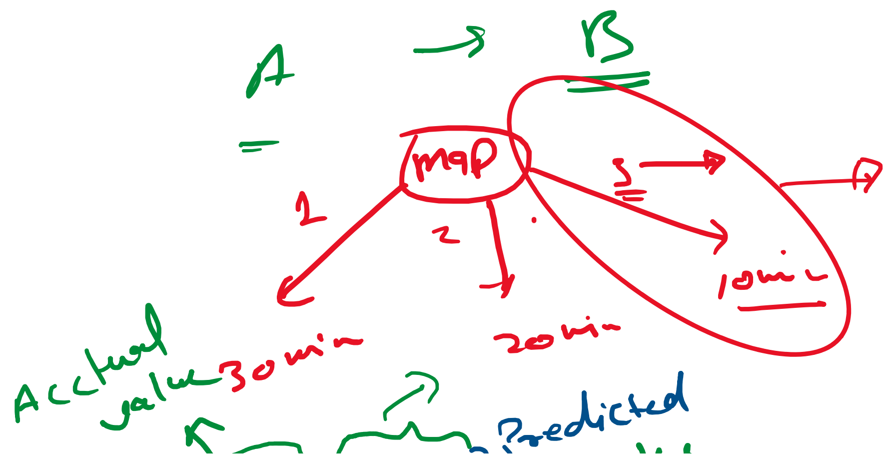
→ L2 regularization tries to estimate **mean** of the data to avoid **overfitting**

Lasso

L1	→	Median
L2	→	Mean

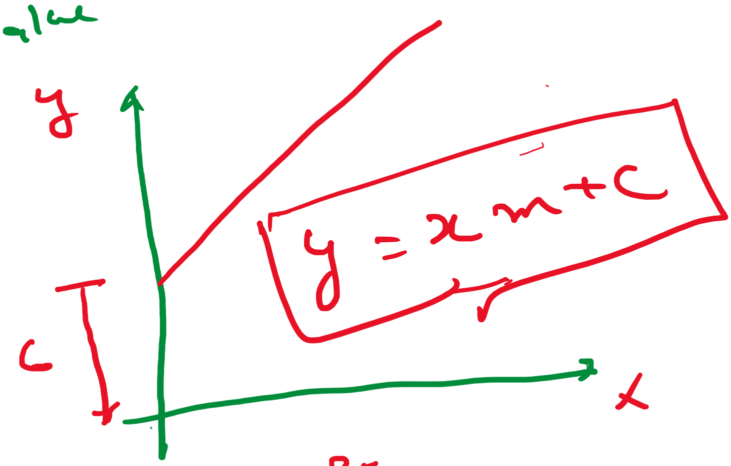
Ridge

Cost function & Loss function



Actual value
Predicted value

	wt	Ht	$\hat{y}$
→	78	180	185
→	68	175	180
→	70	165	175
→	65	150	145



$$\boxed{y = xm + c}$$

$$\boxed{y = xm + c}$$

m → slope
c → intercept

$$\frac{dy}{dx} = m$$ → slope

$$Ht = \beta_0 + \beta \times wt$$

$$\boxed{loss = Predicted - Actual}$$

→ The loss function computes the error for single training example

→ Cost function is the average of the loss fuction for all training examples.

$$\frac{1}{\;} \sum^{n} (\hat{y} - y_i)^2$$

$$\boxed{\text{Cost/loss} = \frac{1}{N}\sum_{i=1}(\hat{y} - y_i)}$$

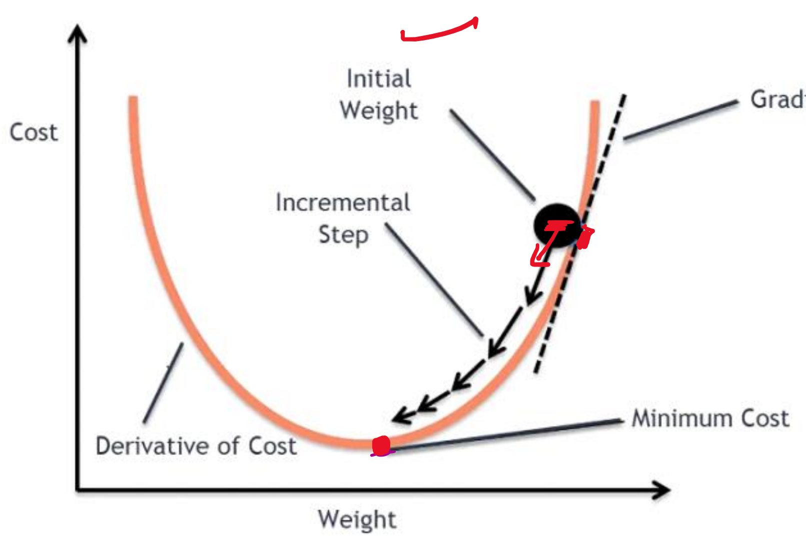
Gradient Descent

$\boxed{\frac{dy}{dx}}$

Gradient descent is an optimization algorithm used in machine learning to minimize the cost function by iteratively adjusting parameters in the direction of the negative gradient aiming to find the optimal set of parameter.

$\Delta \frac{y}{x}$

$\tan\theta = \frac{y}{x}$



$\boxed{\frac{dy}{dx}}$

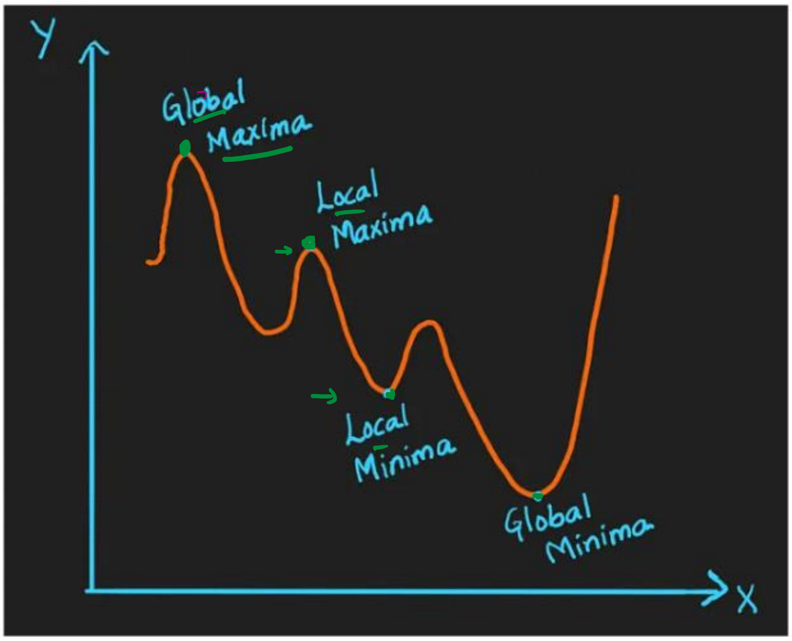
Gradient descent algorithm

$$\text{repeat until convergence } \{$$
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$
$$(\text{for } j = 1 \text{ and } j = 0)$$
$$\}$$

learning rate

0.01, 0.02

$\boxed{0 - 1}$



Evaluation metrics in ML

① Regression metrics

ⓐ Mean Absolute Error

MAE → It is a average distance b/w predicted & original values.

$$\boxed{MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|}$$

② Mean Squared Error (MSE)

It is similar to mean absolute error but the difference is it takes square of the average of b/w predicted & original values.

Note → It is not robust to outlier

$$\sum^{n} (y_i - \hat{y}_i)^2$$

$$\boxed{MSE = \frac{1}{M} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2}$$

(c) Root mean Square Error (RMSE)

It is square root of the mse value

$$\boxed{RMSE = \sqrt{\frac{\sum_{i=1}^{M} (y_i - \hat{y}_i)^2}{M}}}$$

(d) R2 - Score

→ The cofficient of determination also called the R2 Score is used to evaluate the performance of a linear regression model.

② Classification Accuracy

o/p

o/p

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

1
0
1
0
0

X

y

I/O

Independent feature

O/P

Dependent feature

Evaluation

Introduction

Till now we have learnt about the 4 stages of AI project cycle, viz. Problem scoping, Data acquisition, Data exploration and modelling. While in modelling we can make different types of models, how do we check if one's better than the other? That's where Evaluation comes into play. In the Evaluation stage, we will explore different methods of evaluating an AI model. Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future

What is evaluation?

Evaluation is the process of understanding the reliability of any AI model, based on outputs by feeding test dataset into the model and comparing with actual answers. There can be different Evaluation techniques, depending of the type and purpose of the model. Remember that It's not recommended to use the data we used to build the model to evaluate it. This is because our model will simply remember the whole training set, and will therefore always predict the correct label for any point in the training set. This is known as overfitting.

Firstly, let us go through various terms which are very important to the evaluation process.

Model Evaluation Terminologies

There are various new terminologies which come into the picture when we work on evaluating our model. Let's explore them with an example of the Forest fire scenario.

The Scenario

Imagine that you have come up with an AI based prediction model which has been deployed in a forest which is prone to forest fires. Now, the objective of the model is to predict whether a forest fire has broken out in the forest or not. Now, to understand the efficiency of this model, we need to check if the predictions which it makes are correct or not. Thus, there exist two conditions which we need to ponder upon: Prediction and Reality. The prediction is the output which is given by the machine and the reality is the real scenario in the forest when the prediction has been made. Now let us look at various combinations that we can have with these two conditions.

0/1 → $\boxed{\text{model, ml}}$ → o/P

accuracy

The Confusion Matrix		Reality	
		Yes	No
Prediction	Yes	True Positive (TP)	False Positive (FP)
	No	False Negative (FN)	True Negative (TN)

FN3

yes or No

$\boxed{M}$

T H WS → yes

① T | | |

$\boxed{TP}$

② T | H | WS | → P

Model_P → yes
Reality → No

Confusion matrix

The result of comparison between the prediction and reality can be recorded in what we call the confusion matrix. The confusion matrix allows us to understand the prediction results. Note that it is not an evaluation metric but a record which can help in evaluation. Let us once again take a look at the four conditions that we went through in the Forest Fire example:

TP
- True Positive
- Prediction & Reality matches (True)
- Prediction is True (Positive)

- True Negative
- Prediction & Reality matches (True)

Reality → No

- **TP**
  - •Prediction is True (Positive)
- **TN**
  - •True Negative
  - •Prediction & Reality matches (True)
  - •Prediction is False (Negative)
- **FP**
  - •False Postive
  - •Prediction and Reality do not match (False)
  - •Prediction is True (Positive)
- **FN**
  - •False Negative
  - •Prediction and Reality do not match (False)
  - •Prediction is False (Negative)

③ T | H | WS → P, → No

④ T | H | W

①

Case 1: Is there a forest fire?

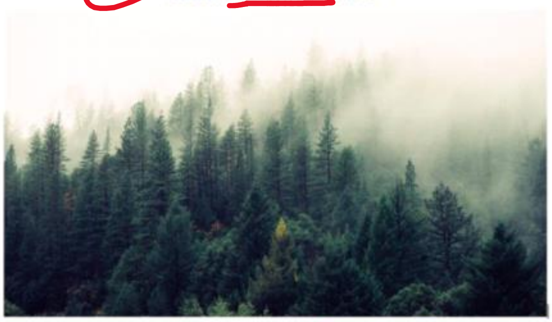
Prediction: Yes — Reality: Yes

True Positive — TP

Here, we can see in the picture that a forest fire has broken out in the forest. The model predicts a Yes which means there is a forest fire. The Prediction matches with the Reality. Hence, this condition is termed as **True Positive**.

TP & TN

Case 2: Is there a forest fire?



TN

Prediction: No — Reality: No

True Negative

Here there is no fire in the forest hence the reality is No. In this case, the machine too has predicted it correctly as a No. Therefore, this condition is termed as **True Negative**.

Case 3: Is there a forest fire?

Case 3: Is there a forest fire?

FP

Prediction: Yes

Reality: No

False Positive

Here the reality is that there is no forest fire. But the machine has incorrectly predicted that there is a forest fire. This case is termed as **False Positive**.

Case 4: Is there a forest fire?

FN

Prediction: No

Reality: Yes

False Negative

Here, a forest fire has broken out in the forest because of which the Reality is Yes but the machine has incorrectly predicted it as a No which means the machine predicts that there is no Forest Fire. Therefore, this case becomes **False Negative**.

Evaluation Methods

Now as we have gone through all the possible combinations of Prediction and Reality, let us see how we can use these conditions to evaluate the model.

1

Accuracy

Accuracy is defined as the percentage of correct predictions out of all the observations. A prediction can be said to be correct if it matches the reality. Here, we have two conditions in which the Prediction matches with the Reality: True Positive and True Negative. Hence, the formula for Accuracy becomes:

$$Accuracy = \frac{Correct\ prediction}{Total\ cases} * 100\%$$

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} * 100\%$$

Here, total observations cover all the possible cases of prediction that can be True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN).

Here, total observations cover all the possible cases of prediction that can be True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN).

Q

Precision

Precision is defined as the percentage of true positive cases versus all the cases where the prediction is true. That is, it takes into account the True Positives and False Positives.

$$Precision = \frac{True\ Positive}{All\ Predicted\ Positives} * 100\%$$

TP

$$Precision = \frac{TP}{TP + FP} * 100\%$$

FP

Recall

Another parameter for evaluating the model's performance is Recall. It can be defined as the fraction of positive cases that are correctly identified. It majorly takes into account the true reality cases where in Reality there was a fire but the machine either detected it correctly or it didn't. That is, it considers True Positives (There was a forest fire in reality and the model predicted a forest fire) and False Negatives (There was a forest fire and the model didn't predict it).

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

FN

$$Recall = \frac{TP}{TP + FN}$$

F1 Score

F1 score can be defined as the measure of balance between precision and recall.

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$