UNIT III

CHAPTER 3

Supervised Learning : Regression

Syllabus

Bias, Variance, Generalization, Underfitting, Overfitting, Linear regression,
Regression: Lasso regression, Ridge regression, Gradient descent algorithm.
Evaluation Metrics : MAE, RMSE, R2

3.1 TRAINING ERROR AND GENERALIZATION ERROR

GQ. Explain and compare training error and generalization error.

3.1.1 Training Error

In **machine learning, training** a predictive model means finding a function which maps a set of values x to a value y. If we apply the model to the data it was trained on, we are calculating the **training error**. If we calculate the **error** on data which was unknown in the **training** phase, we are calculating the test **error**. Training error is calculated as follows:

$$E_{train} = \frac{1}{n} \sum_{i=1}^{n} \text{error } (f_D (X_i), Y_i)$$

In the above equation n represents the number of training examples. $f_D (X_i)$ represents the predicted value and Y_i represents the true or actual values, error $(f_D (X_i), Y_i)$ is used to represent that these two values are same or not and if not then these values differs by how much.

3.1.2 Generalization Error

For supervised learning applications in machine learning and statistical learning theory, **generalization error** is a measure of how accurately an algorithm is able to predict outcome values for previously unseen data. Generalization error is calculated as follows:

$$E_{gen} = \int \text{error } (f_D (X_i), Y_i) \, P (Y, X) \, dX$$

In the above equation error is calculated over all possible values of X and Y. error (fD (X_i), Y_i)) is used to represent that these two values are same or not and if not then these values differes by how much. P(X, Y) represents how often we expect to see such X and Y.

3.1.3 Training Error versus Generalization Error

- The training error is the error of our model as calculated on the training dataset, while generalization error is the expectation of our model's error were we need to apply it to an infinite stream of additional data examples drawn from the same underlying data distribution as our original sample.
- Problematically, we can never calculate the generalization error exactly. That is because the stream of infinite data is an imaginary object. In practice, we must estimate the generalization error by applying our model to an independent test set constituted of a random selection of data examples that were withheld from our training set.
- Let's see an example. Consider a college student trying to prepare for his final exam. A diligent student will strive to practice well and test his abilities using exams from previous years. Nonetheless, doing well on past exams is no guarantee that he will excel when it matters. For instance, the student might try to prepare by rote learning the answers to the exam questions.
- This requires the student to memorize many things. She might even remember the answers for past exams perfectly. Another student might prepare by trying to understand the reasons for giving certain answers. In most cases, the latter student will do much better.

- Let's see one more example, consider the problem of trying to classify the outcomes of coin tosses (class 0: heads, class 1: tails) based on some contextual features that might be available. Suppose that the coin is fair.
- No matter what algorithm we come up with, the generalization error will always be 1/2. However, for most algorithms, we should expect our training error to be considerably lower, depending on the luck of the draw, even if we did not have any features! Consider the dataset {0, 1, 1, 1, 0, 1}. Our feature-less algorithm would have to fall back on always predicting the majority class, which appears from our limited sample to be 1.
- In this case, the model that always predicts class 1 will incur an error of 1/3, considerably better than our generalization error. As we increase the amount of data, the probability that the fraction of heads will deviate significantly from 1/2 diminishes, and our training error would come to match the generalization error.
- When we train our models, we attempt to search for a function that fits the training data as well as possible. If the function is so flexible that it can catch on to spurious patterns just as easily as to true associations, then it might perform *too well* without producing a model that generalizes well to unseen data.
- This is precisely what we want to avoid or at least control. Many of the techniques in deep learning are heuristics and tricks aimed at guarding against overfitting.
- When we have simple models and abundant data, we expect the generalization error to resemble the training error. When we work with more complex models and fewer examples, we expect the training error to go down but the generalization gap to grow.

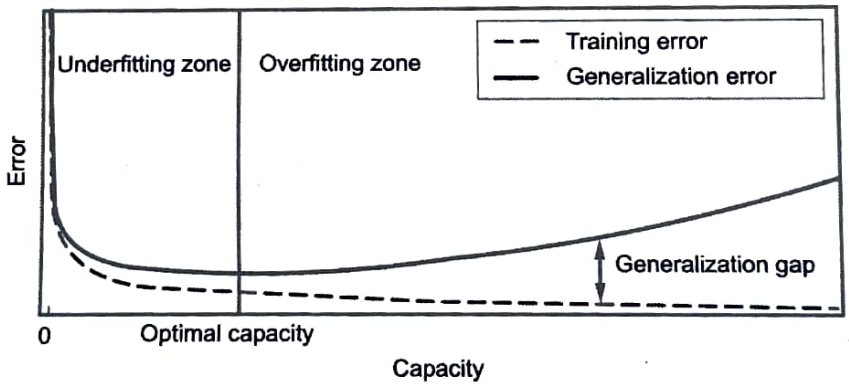


Fig. 3.1.1 : Training Error and Generalization Error

3.2 UNDERFITTING, OVERFITTING, BIAS AND VARIANCE TRADE OFF

GQ. Mention uses of Underfitting, Overfitting, Bias and Variance Trade Off.

- Let us consider that we are designing a machine learning model. A model is said to be a good machine learning model if it generalizes any new input data from the problem domain in a proper way. This helps us to make predictions in the future data, that data model has never seen.

- Now, suppose we want to check how well our machine learning model learns and generalizes to the new data. For that we have overfitting and underfitting, which are majorly responsible for the poor performances of the machine learning algorithms.

Before diving further let's understands two important terms:

- Bias – Assumptions made by a model to make a function easier to learn. (The algorithms error rate on the training set is algorithms bias.)
- Variance - If you train your data on training data and obtain a very low error, upon changing the data and then training the same previous model you experience high error, this is variance.
- (How much worse the algorithm does on the test set than the training set is known as the algorithms variance.)

3.2.1 Underfitting

- A statistical model or a machine learning algorithm is said to have underfitting when it cannot capture the underlying trend of the data.
- Underfitting destroys the accuracy of our machine learning model. Its occurrence simply means that our model or the algorithm does not fit the data well enough.
- It usually happens when we have less data to build an accurate model and also when we try to build a linear model with a non-linear data.
- In such cases the rules of the machine learning model are too easy and flexible to be applied on such minimal data and therefore the model will probably make a lot of wrong predictions.
- Underfitting can be avoided by using more data and also reducing the features by feature selection.
- In a nutshell, Underfitting – High bias and low variance

3.2.2 Techniques to Reduce Underfitting

1. Increase model complexity
2. Increase number of features, performing feature engineering
3. Remove noise from the data.
4. Increase the number of epochs or increase the duration of training to get better results.

3.2.3 Overfitting

- A statistical model is said to be overfitted, when we train it with a lot of data . When a model gets trained with so much of data, it starts learning from the noise and inaccurate data entries in our data set.
- Then the model does not categorize the data correctly, because of too many details and noise.
- The causes of overfitting are the non-parametric and non-linear methods because these types of machine learning algorithms have more freedom in building the model based on the dataset and therefore they can really build unrealistic models.

- A solution to avoid overfitting is using a linear algorithm if we have linear data or using the parameters like the maximal depth if we are using decision trees.
- In a nutshell, Overfitting – High variance and low bias

3.2.4 Techniques to Reduce Overfitting

GQ. What are Techniques to Reduce Overfitting.

1. Increase training data.
2. Reduce model complexity.
3. Early stopping during the training phase (have an eye over the loss over the training period as soon as loss begins to increase stop training).
4. Ridge Regularization and Lasso Regularization
5. Use dropout for neural networks to tackle overfitting.

- Ideally, the case when the model makes the predictions with 0 error, is said to have a *good fit* on the data. This situation is achievable at a spot between overfitting and underfitting. In order to understand it we will have to look at the performance of our model with the passage of time, while it is learning from training dataset.
- With the passage of time, our model will keep on learning and thus the error for the model on the training and testing data will keep on decreasing. If it will learn for too long, the model will become more prone to overfitting due to the presence of noise and less useful details. Hence the performance of our model will decrease. In order to get a good fit, we will stop at a point just before where the error starts increasing. At this point the model is said to have good skills on training datasets as well as our unseen testing dataset.

Bias-variance trade-off

So what is the right measure? Depending on the model at hand, a performance that lies between overfitting and underfitting is more desirable. This trade-off is the most integral aspect of Machine Learning model training. As we discussed, Machine Learning models fulfil their purpose when they generalize well. Generalization is bound by the two undesirable outcomes - high bias and high variance. Detecting whether the model suffers from either one is the sole responsibility of the model developer.

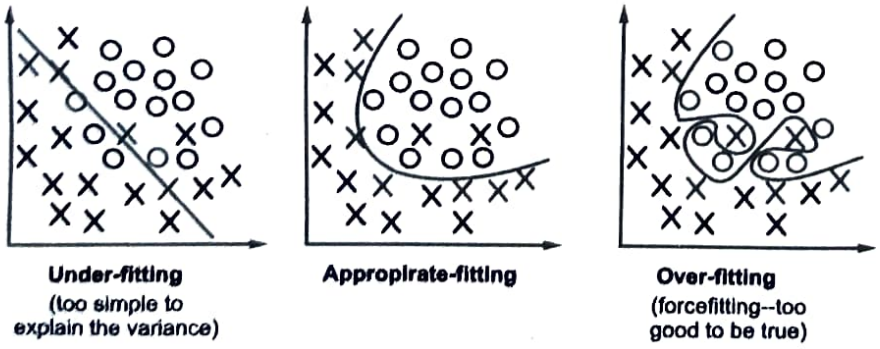


Fig. 3.2.1 : Underfitting and Overfitting

Bias-variance trade-off

So what is the right measure? Depending on the model at hand, a performance that lies between overfitting and underfitting is more desirable. This trade-off is the most integral aspect of Machine Learning model training. As we discussed, Machine Learning models fulfil their purpose when they generalize well. Generalization is bound by the two undesirable outcomes — high bias and high variance. Detecting whether the model suffers from either one is the sole responsibility of the model developer.

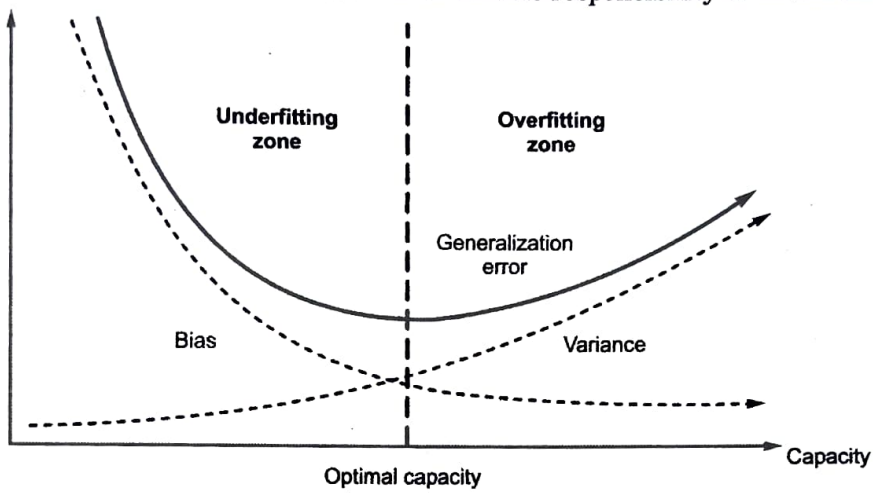


Fig. 3.2.2 : Bias variance Tradeoff as a function of model capacity

3.3 LINEAR REGRESSION

UQ. Explain Regression line, Scatter plot, Error in prediction and Best fitting line. (Ref. – May 15, 4 Marks)

UQ. Explain in brief Linear Regression Technique. (Ref – May 16, 5 Marks)

- One of the most important supervised learning tasks is regression. In regression set of records are present with X and Y values and this values are used to learn a function, so that if you want to predict Y from an unknown X this learn function can be used. In regression we have to find value of Y, So, a function is required which predicts Y given X. Y is continuous in case of regression.
- Here Y is called as criterion variable and X is called as predictor variable. There are many types of functions or modules which can be used for regression. Linear function is the simplest type of function. Here, X may be a single feature or multiple features representing the problem.

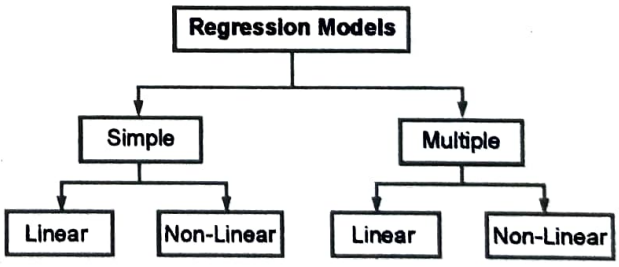
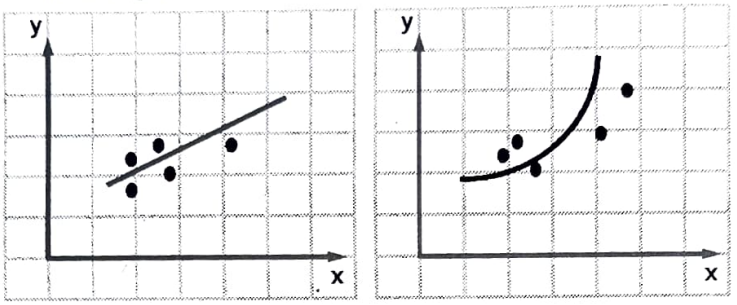


Fig. 3.3.1 : Types of Regression Models

3.3.1 Simple Linear Regression

- Let's see simple regression first, in this X contains a single feature. In multiple regressions, X contains more than one feature. In simple regression training records are plotted as value of X vs. value of Y. Next task is to find a function, so that if a random unknown X value is given we can predict Y.

- There are different types of functions that can be used. In linear regression, we assume that the function is linear as shown in Fig. 3.3.2(a).



(a) Linear Function (b) Non-Linear Function

Fig. 3.3.2 : Types of regression functions

- In Linear regression a best fitted straight line is drawn which passes through the points called as the regression line. The line shown in Fig. 3.3.2 is the regression line.
- Regression line shows the calculated values of Y for each possible value of X. Errors in the prediction is shown by the vertical lines drawn from the points to the regression line.
- Error in prediction is less when the point is very near to the regression line as shown by first and second point in Fig. 3.3.3. When the point is far from the regression line then the error in prediction is high, as shown by the third and fourth point in Fig. 3.3.3.

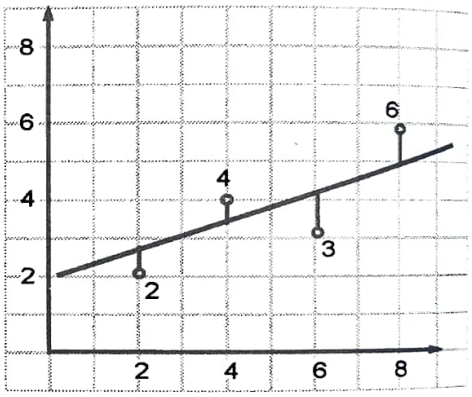


Fig. 3.3.3 : Prediction and the Error in the Prediction

- The difference between the value of point and the predicted value is called as error of prediction. Predicted value is the value of point on the line.
- Let's take an example, the predicted values (Y′) and the errors of prediction (Y-Y′) are shown in Table 3.3.1. From the table we can say that, the second point has Y value as 4 and a predicted Y value as 3.2. The error of prediction is 0.8.

Table 3.3.1 : Regression data

Sr. No.	X	Y	Y′	Y-Y′	(Y-Y′)²
1	2	2	2.2	– 0.2	0.04
2	4	4	3.2	0.8	0.64
3	6	3	4.3	– 1.3	1.69
4	8	6	5.4	0.6	0.36

- The best-fit line is called as the line that will minimize the sum of the squared errors of prediction. This criterion is used to draw the line in Fig. 3.3.3. The squared errors of prediction are shown in the last column of Table 3.3.1.
- The regression line is represented using the following equation,

$$Y' = aX + b + e$$

- In the above equation Y' represents the predicted value, a represents the slope of the line, b shows the Y intercept and e is the random error. Here we assume that mean value of random error is 0, so the equation becomes,

$$Y' = aX + b$$

Table 3.3.2 : Calculation of Regression Parameters

Sr. No.	X	Y	XY	X^2
1	2	2	4	4
2	4	4	16	16
3	6	3	18	36
4	8	6	48	64
Total	**20**	**15**	**86**	**120**

- The regression line equation is,

$$Y' = aX + b$$

$$a = \frac{n\sum XY - \sum X \sum Y}{n\sum X^2 - (\sum X)^2} = \frac{4 \times 86 - 20 \times 15}{4 \times 120 - 400} = 0.55$$

$$b = \frac{1}{n}(\sum Y - a \times \sum X) = \frac{1}{4}(15 - 0.55 \times 20) = 1$$

- Now the equation for the line becomes,

$$Y' = 0.55X + 2.2$$

For X = 2, $Y' = (0.55)(2) + 1 = 2.2$

For X = 4, $Y' = (0.55)(4) + 1 = 3.2$

For X = 6, $Y' = (0.55)(6) + 1 = 4.3$

For X = 8, $Y' = (0.55)(8) + 1 = 5.4$

3.3.2 Examples of Linear Regression

Ex. 3.3.1 : The expenditure of an organization (in thousand) for every month is shown in table below :

X (Month)	1	2	3	4	5
Y (Expenditure)	12	19	29	37	45

Find regression line, Y = a X + b using least square method.

Estimate the expenditure of company in 6th month using line as a model.

☑ **Soln. :**

Sr. No.	X	Y	XY	X^2
1	1	12	12	1
2	2	19	38	4
3	3	29	87	9
4	4	37	148	16
5	5	45	225	25
Total	**15**	**142**	**510**	**55**

(a) The equation for the regression line is,

$$Y' = aX + b$$

$$a = \frac{n\sum XY - \sum X \sum Y}{n\sum X^2 - (\sum X)^2} = \frac{5 \times 510 - 15 \times 142}{5 \times 55 - 225} = 8.4$$

$$b = \frac{1}{n}(\sum Y - a \times \sum X) = \frac{1}{5}(142 - 8.4 \times 15) = 3.2$$

Now the equation for the line becomes,

$$Y' = 8.4X + 3.2$$

(b) In 6^{th} month

$$Y' = 8.4 \times 6 + 3.2$$

$$Y' = 53.6 \text{ Thousand}$$

Ex. 3.3.2 : Consider the set of data as {(– 1, – 1), (2, 2), (3, 2)} (a) Find the equation of regression line. (b) Draw the scatter plot of data and regression line.

☑ **Soln. :**

Sr. No.	X	Y	XY	X^2
1	– 1	– 1	1	1
2	2	2	4	4
3	3	2	6	9
Total	4	3	11	14

(a) The equation for the regression line is,

$$Y' = aX + b$$

$$a = \frac{n\sum XY - \sum X \sum Y}{n\sum X^2 - (\sum X)^2} = \frac{3 \times 11 - 4 \times 3}{3 \times 14 - 16} = 0.807$$

$$b = \frac{1}{n}(\sum Y - a \times \sum X) = \frac{1}{3}(3 - 0.807 \times 4) = -0.076$$

Now the equation for the line becomes,

$$Y' = 0.807X - 0.076$$

(b) Now we can plot the regression line given by equation $Y' = 0.807\ X - 0.076$ and the given data points.

Sr. No.	X	Y'
1	– 1	– 0.883
2	2	1.54
3	3	2.34

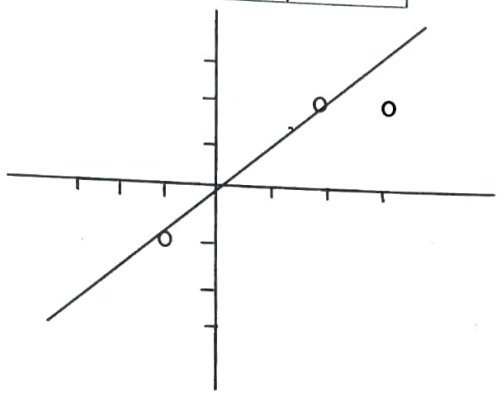


Fig. Ex. 3.3.2 : Scatter plot of data

UEx. 3.3.3 **Ref. - May 17, 10 Marks**

Following table shows the midterm and final exam grades obtained for students in a database course. Use the method of least squares using regression to predict the final exam grade of a student who received 86 in the mid term exam.

Midterm exam(X)	72	50	81	74	94	86	59	83	86	33	88	81
Final exam(Y)	84	53	77	78	90	75	49	79	77	52	74	90

☑ **Soln. :**

Sr. No.	X	Y	XY	X^2
1	72	84	6048	5184
2	50	53	2650	2500
3	81	77	6237	6561
4	74	78	5772	5476
5	94	90	8460	8836
6	86	75	6450	7396
7	59	49	2891	3481
8	83	79	6557	6889
9	86	77	6622	7396
10	33	52	1716	1089
11	88	74	6512	7744
12	81	90	7290	6561
Total	**887**	**878**	**67205**	**69113**

The equation for the regression line is,

$$Y' = aX + b$$

$$a = \frac{n\sum XY - \sum X \sum Y}{n\sum X^2 - (\sum X)^2} = 0.65$$

$$b = \frac{1}{n}(\sum Y - a \times \sum X) = 25.12$$

Now the equation for the line becomes,

$$Y' = 0.65X + 25.12$$

The final exam grade of a student who received 86 in the mid term exam,

$$Y' = 0.65 \times 86 + 25.12$$

$$Y' = 81.02$$

UEx. 3.3.4 Ref. - Dec. 19, 10 Marks

Given the following data for the sales of car of an automobile company for six consecutive years. Predict the sales for next two consecutive years.

Years (x)	2013	2014	2015	2016	2017	2018
Sales (y)	110	100	250	275	230	300

Soln :

We will take t = x – 2013

Sr. No.	t	Y	tY	t^2
0	0	110	0	0
1	1	100	100	1
2	2	250	500	4
3	3	275	825	9
4	4	230	920	16
5	5	300	1500	25
Total	**15**	**1265**	**3845**	**55**

The equation for the regression line is,

$$Y' = at + b;$$

$$a = 39; \qquad b = 113.33$$

Now the equation for the line becomes,

$$Y' = 39t + 113.33$$

The sale of company for next two years,

$$X = 2019, t = 6; \qquad Y' = 39 \times 6 + 113.33 = 347.33$$

$$X = 2020, t = 7; \qquad Y' = 39 \times 7 + 113.33 = 386.33$$

3.4 LASSO RE

GQ. Explain lasso regr

- In machine learni regression analys
- It performs both interpretability of
- Here we shall s Regularization is

3.4.1 What is L

- Lasso regression
- Shrinkage is whe
- The lasso procedu
- This type of regr wants to automat
- Lasso Regression automatically per
- Lasso is a statisti

3.4.2 Regulariz

- Regularization is and test data are
- Regularization is It achieves a le variables over th
- In regularization The magnitude discuss the techn

3.4.3 Lasso Reg

- If a regression m
- L1 regularizatior
- This regularizati
- Larger penalties

3.4.4 Mathema

- The mathematic

3.4 LASSO REGRESSION

GQ. Explain lasso regression in detail.

- In machine learning, lasso (least absolute shrinkage and selection operator, also Lasso or LASSO) is a regression analysis method.
- It performs both variable selection and regularization to enhance the prediction accuracy and interpretability of the resulting statistical model.
- Here we shall see the techniques used to overcome **overfitting** for a lasso regression model. Regularization is one of the methods, used to make the given model more generalized.

3.4.1 What is Lasso Regression ?

- Lasso regression is a regularization technique. It uses shrinkage.
- Shrinkage is where data values are shrunk towards a central point as the mean.
- The lasso procedure encourages simple, sparse model, i.e.; models with fewer parameters.
- This type of regression is suitable for models showing high levels of multicollinearity or when one wants to automate certain parts of model selection, like variable selection/parameter elimination.
- Lasso Regression uses L1 regularization technique, when there are more features. It is because it automatically performs feature selection.
- Lasso is a statistical formula for the regularization of data models and feature selection.

3.4.2 Regularization

- Regularization is an important concept. It is used to avoid over fitting of the data, when the trained and test data are much varying.
- Regularization is implemented by adding a '**penalty**' term to the best fit derived from the trained data. It achieves a **lesser variance** with the tested data and also restricts the influence of predictor variables over the output variable and this is achieved by compressing their coefficients.
- In regularization, we keep the same number of features but reduce the magnitude of the coefficients. The magnitude of the coefficients can be reduced by using different types of techniques. First we discuss the techniques.

3.4.3 Lasso Regularization Techniques

- If a regression model uses the L1 regularization technique, it is called Lasso Regression.
- L1 regularization adds a penalty that is equal to the absolute value of the magnitude of the coefficient.
- This regularization type develops sparse models with few coefficients.
- Larger penalties result in coefficient values that are closer to zero.

3.4.4 Mathematical equation of Lasso Regression

- The mathematical equal of L.R. is given by :

- Residual sum of squares + λ. [Sum of the absolute value of the magnitude of coefficients]

i.e., $$\sum_{i=1}^{n} \left[y_i - \sum_{j} x_{ij} B_j \right]^2 + \lambda \sum_{j=1}^{P} | \beta_j |$$

where

(i) λ denotes the amount of shrinkage,

(ii) λ = 0 implies all features are considered and it is equivalent to the linear regression where only the residual sum of squares is considered. That builds a predictive model.

(iii) λ = ∞ implies no feature is considered i .e., as λ is very large, it eliminates more and more features.

(iv) The bias increases with increase in λ.

(v) Variance increases with decrease in λ.

☞ **Lasso regression example**

```
import numpy as np
```

Creating a New Train and Validation Datasets

```
from sklearn.model_selection import train_test_split
data_train, data_val = train_test_split(new_data_train, test_size = 0.2, random_state = 2)
```

Classifying Predictors and Target

```
#Classifying Independent and Dependent Features
#_____
#Dependent Variable
Y_train = data_train.iloc[:, -1].values
#Independent Variables
X_train = data_train.iloc[:,0 : -1].values
#Independent Variables for Test Set
X_test = data_val.iloc[:,0 : -1].values
Evaluating The Model With RMLSE
def score(y_pred, y_true):
error = np.square(np.log10(y_pred +1) - np.log10(y_true +1)).mean() ** 0.5
score = 1 - error
return score
actual_cost = list(data_val['COST'])
actual_cost = np.asarray(actual_cost)
```

☞ **Building the Lasso Regressor**

```
#Lasso Regression
from sklearn.linear_model import Lasso
```

```
#Initializing the Lasso Regressor with Normalization Factor as True
lasso_reg = Lasso(normalize=True)
#Fitting the Training data to the Lasso regressor
lasso_reg.fit(X_train,Y_train)
#Predicting for X_test
y_pred_lass =lasso_reg.predict(X_test)
#Printing the Score with RMLSE
print("\n\nLasso SCORE : ", score(y_pred_lass, actual_cost))
```

Output

```
0.7335508027883148
The Lasso Regression attained an accuracy of 73% with the given Dataset.
```

3.4.5 Bayesian Interpretation

- Lasso can be interpreted as linear regression for which the coefficients have Laplace prior distributions.
- The Laplace distribution is sharply peaked at zero (its first derivative is discontinuous at zero) and it concentrates its probability mass closer to zero than the normal distribution.
- This gives an alternative explanation of why lasso tends to set some coefficients to zero.

3.4.6 Choice of Regularisation Parameter

- Choosing the regularization parameter (λ) is a fundamental part of lasso. A good value is essential to the performance of lasso, because it controls the strength of shrinkage and variable selection. It improves both prediction accuracy and interpretability.
- But, if the regularization is too strong, then important variables may be omitted and coefficients get shrunk excessively. This harms both predictive capacity and inferencing.
- Cross-validation is often used to find the regularization parameter.
- Information criterion such as the Bayesian information criterion (BIC) is preferable to cross-validation, because it is faster to compute and its performance is less volatile in small samples.

3.4.7 Selected Applications

LASSO has been applied in economics and finance, and it is found to improve prediction and to select sometimes neglected variables, for example, in corporate bankruptcy prediction literature, or high growth firms prediction.

3.5 RIDGE REGRESSION

GQ. What is Ridge regression models ?

- Ridge **regression** is a model tuning method. It is used to analyse any data that suffers from multicollinearity.

- This method performs L2 regularization. When we get multicollinearity problem, least-squares method are unbiased, and variances are large. This makes predicted values far away from the actual values.
- The cost function for ridge regression :

$$\text{Min}\left(|| Y - X(\text{theta}) ||^2 + \lambda || \text{theta} ||^2 \right)$$

- Lambda is the penalty term, λ given here is denoted by an alpha-parameter in the ridge function. So, by changing the values of alpha, we can control penalty term.
- The higher the values of alpha, the bigger is the penalty and therefore the magnitude of the coefficient is reduced.
  (i) It shrinks the parameters. Hence, it is used to prevent multicollinearty.
  (ii) It reduces the model complexity by coefficient shrinkage.

3.5.1 Ridge Regression Models

- The usual regression equation forms the base for any type of regression machine learning model. It is given by :

$$Y = XB + e$$

- Where Y is the dependent variable, X represent the independent variables, B is the regression coefficient to be estimated, and e represents the errors due to residuals.
- We add the lambda function to this equation, the variance that is not evaluated by the general method is considered.
- When the data is ready and is part of L2 regularization, we can under take the steps :

3.5.2 Standardisation

- In ridge regression, the first step is to standardize the variables (both dependent and independent) by subtracting their means and dividing by their standard deviations.
- As far as standardization is concerned, all ridge regression calculations are based on standardized variables.
- When the final regression coefficients are displayed, they are adjusted back into their original scale.

3.5.3 Bias and Variance trade-off

- Bias and variance trade-off is generally complicated when it comes to building ridge regression models on an actual dataset.
- The general trends are :
  1. The bias increases as λ increases.
  2. The variance decreases as λ increases.

3.5.4 Assumptions of Ridge Regressions

- The assumptions of ridge regression are the same as that of linear regression : linearity, constant variance and independence.

- Since ridge regression does not give confidence limits, the distribution of errors need not be normal.

☞ **Example**

Upload Required Libraries

```
import numpy as np
import pandas as pd
import os

import seaborn as sns
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
import matplotlib.style
plt.style.use('classic')

import warnings
warnings.filterwarnings("ignore")

df = pd.read_excel("food.xlsx")
```

	Week	Final_price	Unit_price	Website_homepage_mention	Food_category	Cuisine	Center_type	Home_delivery	Nigh_service	Area_range	order
0	122.0	150.35	152.35	0.0	Beverages	Italian	Delhi	1.0	1	7.0	972
1	95.0	484.03	485.03	0.0	Desert	Italian	Noida	0.0	1	5.6	150
2	52.0	281.33	281.33	0.0	Starters	Italian	Gurgaon	0.0	1	3.0	55
3	10.0	167.81	196.94	0.0	Extras	Italian	Delhi	1.0	1	4.0	256
4	122.0	212.46	256.14	0.0	Salad	Italian	Dehil	0.0	1	3.4	82

After conducting all the EDA on the data, treatment of missing values, we shall now go ahead with creating dummy variables, as we cannot have categorical variables in the dataset.

```
df =pd.get_dummies(df, columns=cat,drop_first=True)
```

Where columns=cat is all the categorical variables in the data set.

After this, we need to standardize the data set for the Linear Regression method.

Ridge expression can be used for the analysis of prostrate-specific antigen and clinical measures among people who were about to have prostrates removed.

The performance of ridge regression is good when there is a subset of true coefficients which are small or even zero.

3.6 MULTICOLLINEARITY

- The term multicollinearity refers to collinearity concepts in statistics.
- In this model, one predicted value in multiple regression models is linearly predicted with others to attain a certain level of accuracy.
- The concept multicollinearity occurs when there are high correlation between more than two predicted values.

☞ **The Regularisation techniques are as follows**

(i) Penalise the magnitude of coefficients of features,

(ii) Minimise the error between the actual and predicted observations.

3.6.1 Working of Ridge-Regression model

- Ridge Regression performs L2 regularization.
- Here the penalty equivalent is added to the square of the magnitude of coefficients.
- The minimization objective is as follows :
- Consider a response vector $y \in R_n$ and a predictor matrix $X \in R_{n \times p}$, the ridge regression coefficients are defined as :

(i) Here λ is the turning factor that controls the strength of the penalty term.

(ii) If $\lambda = 0$, the objective becomes similar to simple linear regression. Hence we get the some coefficients as simple linear regression.

(iii) If $\lambda = \infty$, the coefficient will be zero, because of infinite weightage on the square of coefficients as anything less than zero makes the objective infinite.

(iv) $0 < \lambda < \infty$, the magnitude of λ decides the weightage given to the different parts of the objective.

(v) The minimization objective

$$= \text{LS obj} + \lambda \text{ (sum of squares of coefficients)}$$

(vi) Where LS Obj is least square objective that is the linear regression objective without regularization.

3.6.2 Benefit of ridge-regression

- Ridge Regression solves the problem of overfitting, as just regular squared error regression fails to recognize the less important and uses all the them, leading to overfitting.
- Ridge regression adds a slight bias to fit the model according to the true values of the data.

3.6.3 Difference between Ridge Regression and Lasso Regression

(1) (i) Ridge regression is mostly used to reduce the overfitting in the model, and it includes all the features present in the model.

(ii) It reduces the complexity of the model by shrinking the coefficients

(2) Lasso Regression helps to reduce the overfitting in the model as well as feature selection.

3.7 THE GRADIENT DESCENT ALGORITHM

GQ. Explain the gradient descent algorithm.

- The Gradient Descent method lays the foundation for machine learning and deep learning techniques. Let's explore how does it work, when to use it and how does it behave for various functions.

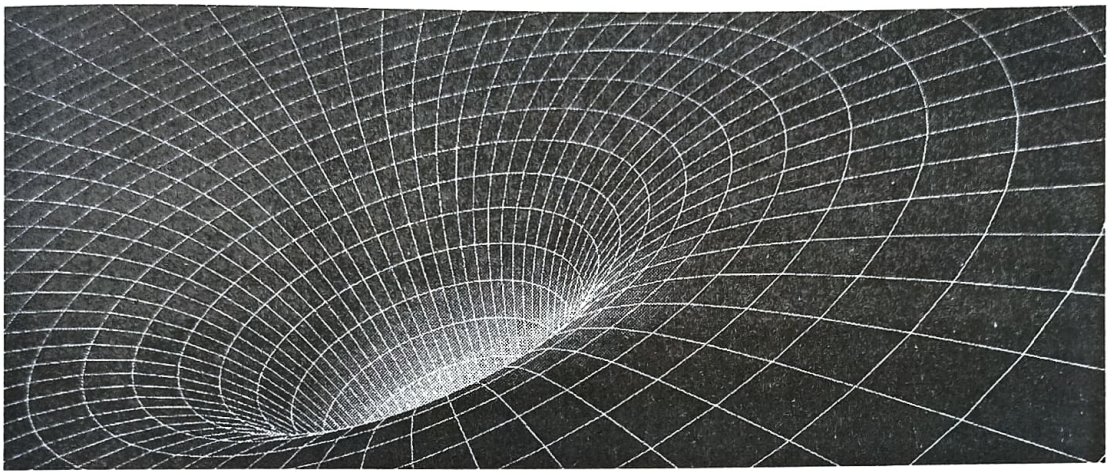


Fig. 3.7.1

3.8 INTRODUCTION

- Gradient descent (GD) is an iterative first-order optimisation algorithm used to find a local minimum/maximum of a given function. This method is commonly used in *machine learning* (ML) and *deep learning*(DL) to minimise a cost/loss function (e.g. in a linear regression). Due to its importance and ease of implementation, this algorithm is usually taught at the beginning of almost all machine learning courses.
- However, its use is not limited to ML/DL only, it's being widely used also in areas like:
  - Control engineering (robotics, chemical, etc.)
  - Computer games
  - Mechanical engineering
- That's why today we will get a deep dive into the math, implementation and behaviour of first-order gradient descent algorithm. We will navigate the custom (cost) function directly to find its minimum, so there will be no underlying data like in typical ML tutorials — we will be more flexible in terms of a function's shape.
- This method was proposed before the era of modern computers and there was an intensive development meantime which led to numerous improved versions of it but in this article, we're going to use a basic/vanilla gradient descent implemented in Python.

3.9 FUNCTION REQUIREMENTS

- Gradient descent algorithm does not work for all functions. There are two specific requirements. function has to be:
  - Differentiable
  - Convex
- First, what does it mean it has to be **differentiable**? If a function is differentiable it has a derivati for each point in its domain - not all functions meet these criteria. First, let's see some examples functions meeting this criterion:

let's see some examples of functions meeting this criterion:

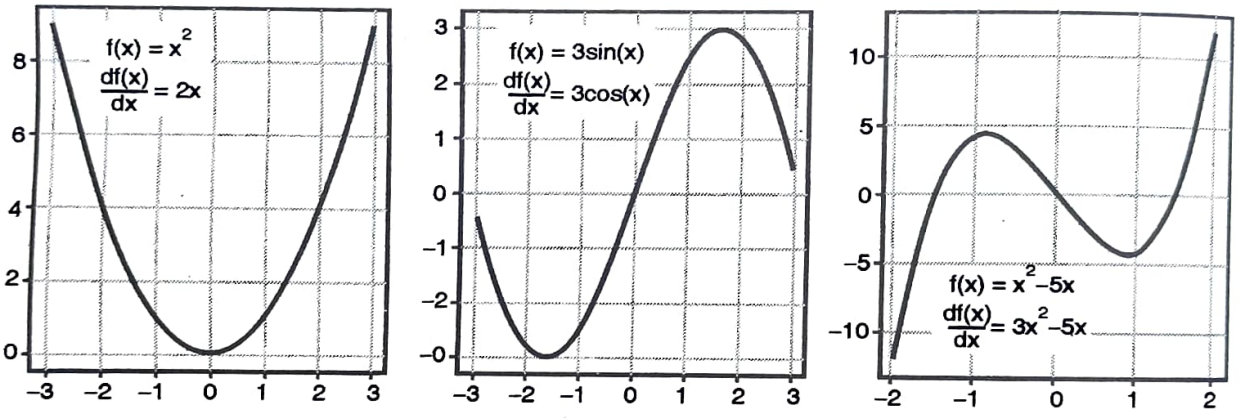


Fig. 3.9.1

- Typical non-differentiable functions have a step a cusp or a discontinuity:

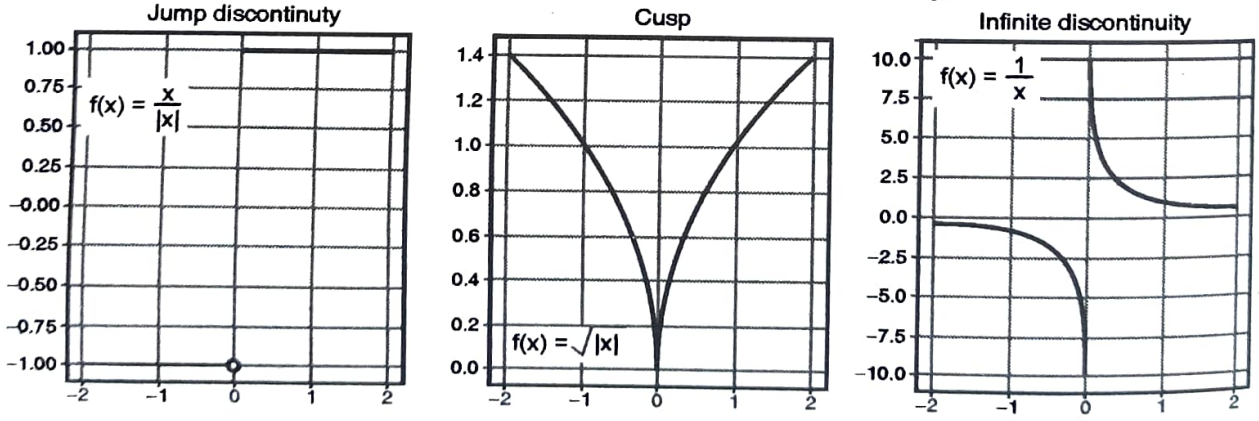


Fig. 3.9.2

- Next requirement - **function has to be convex**. For a univariate function, this means that the line segment connecting two function's points lays on or above its curve (it does not cross it). If it does it means that it has a local minimum which is not a global one.
- Mathematically, for two points x_1, x_2 laying on the function's curve this condition is expressed as:

$$f[\lambda x_1 + (1-\lambda) x_2] \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$

- where λ denotes a point's location on a section line and its value has to be between 0 (left point) and 1 (right point), e.g. λ = 0.5 means a location in the middle.
- Below there are two functions with exemplary section lines.

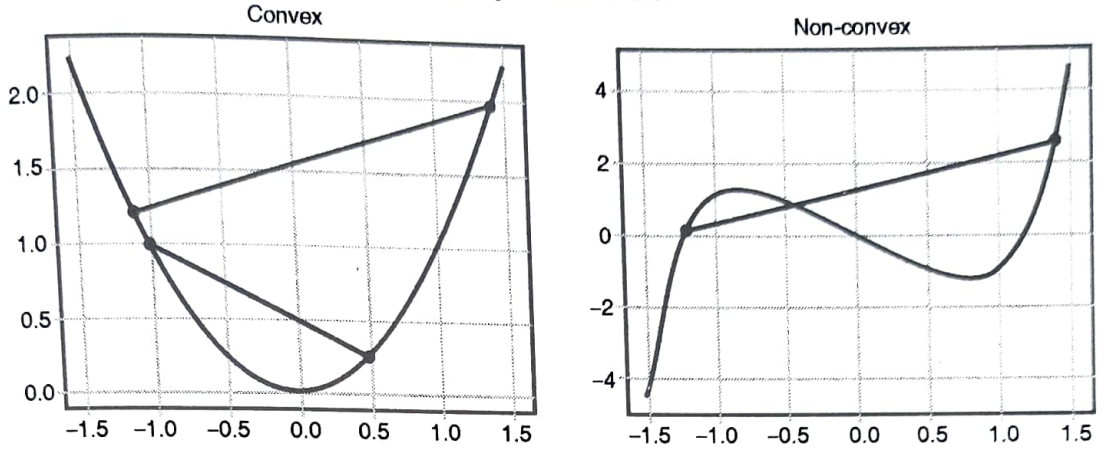


Fig. 3.9.3

- Exemplary convex and non-convex functions;

  i.e. $\frac{d^2 f(x)}{dx^2} > 0$

- Another way to check mathematically if a univariate function is convex is to calculate the second derivative and check if its value is always bigger than 0.

☞ **Gradient Descent Algorithm**

- Gradient Descent Algorithm iteratively calculates the next point using gradient at the current position, then scales it (by a learning rate) and subtracts obtained value from the current position (makes a step). It subtracts the value because we want to minimise the function (to maximise it would be adding). This process can be written as:

$$P_{n-1} = P_{n-\eta} \nabla f(P_n)$$

  - There's an important parameter η which scales the gradient and thus controls the step size. In machine learning, it is called learning rate and have a strong influence on performance.
  - The smaller learning rate the longer GD converges, or may reach maximum iteration before reaching the optimum point
  - If learning rate is too big the algorithm may not converge to the optimal point (jump around) or even to diverge completely.

- In summary, Gradient Descent method's steps are:
  (1) choose a starting point (initialisation)
  (2) calculate gradient at this point
  (3) make a scaled step in the opposite direction to the gradient (objective: minimise)
  (4) repeat points 2 and 3 until one of the criteria is met:

(i) maximum number of iterations reached

(ii) step size is smaller than the tolerance.

Below there's an exemplary implementation of the Gradient Descent algorithm (with steps tracking):

```
import numpy as np

def gradient_descent(start, gradient, learn_rate, max_iter, tol=0.01):
  steps = [start] # history tracking
  x = start

  for _ in range(max_iter):
    diff = learn_rate*gradient(x)
    if np.abs(diff)<tol:
      break
    x = x - diff
    steps.append(x) # history tracing

  return steps, x
```

☛ **This function takes 5 parameters**

1. starting point - in our case, we define it manually but in practice, it is often a random initialisation
2. gradient function - has to be specified before-hand
3. learning rate - scaling factor for step sizes
4. maximum number of iterations
5. tolerance to conditionally stop the algorithm (in this case a default value is 0.01)

3.10 EVALUATION METRICS

GQ. Explain Evaluation metrics.

☛ **Introduction**

- Machine Learning is a branch of Artificial Intelligence. It contains many algorithms to solve various real-world problems. Building a Machine learning model is not only the Goal of any data scientist but deploying a more generalized model is a target of every Machine learning engineer.
- Regression is also one type of supervised Machine learning and in this tutorial, we will discuss various metrics for evaluating regression Models and How to implement them using the sci-kit-learn library.

☛ **Regression**

- Regression is a type of Machine learning which helps in finding the relationship between independent and dependent variable.

- In simple words, Regressio
discrete values like price, R

☛ **Why We require Evaluation**

- Machine learning model ca
model. which further includ
- It is necessary to obtain th
approximate result on unse
- So to build and deploy a ge
helps us to better optimize
- If one metric is perfect, the
- Now, I hope you get the im
metrics used for regression

3.11 MEAN ABSOLUTE

GQ. Explain MAE, RMSE, R2

- MAE is a very simple met
values.
- To better understand, let's
data and output data and u
a best-fit line.
- Now you have to find th
difference between the actu
- so, sum all the errors and
observations And this is MA
MAE because this is a loss.

☛ **Advantages of MAE**

- The MAE you get is in the s
- It is most Robust to outliers.

☛ **Disadvantages of MAE**

- The graph of MAE is not di
which can be differentiable.

```
from sklearn.metrics import
print("MAE",mean_absolute_
```

- In simple words, Regression can be defined as a Machine learning problem where we have to predict discrete values like price, Rating, Fees, etc.

Why We require Evaluation Metrics?

- Machine learning model cannot have 100 per cent efficiency otherwise the model is known as a biased model. which further includes the concept of overfitting and underfitting.
- It is necessary to obtain the accuracy on training data, But it is also important to get a genuine and approximate result on unseen data.
- So to build and deploy a generalized model we require to Evaluate the model on different metrics which helps us to better optimize the performance, fine-tune it, and obtain a better result.
- If one metric is perfect, there is no need for multiple metrics.
- Now, I hope you get the importance of Evaluation metrics. let's start understanding various evaluation metrics used for regression tasks.

3.11 MEAN ABSOLUTE ERROR(MAE)

GQ. Explain MAE, RMSE, R2

- MAE is a very simple metric which calculates the absolute difference between actual and predicted values.
- To better understand, let's take an example you have input data and output data and use Linear Regression, which draws a best-fit line.
- Now you have to find the MAE of your model. Find the difference between the actual value and predicted value
- so, sum all the errors and divide them by a total number of observations And this is MAE. And we aim to get a minimum MAE because this is a loss.

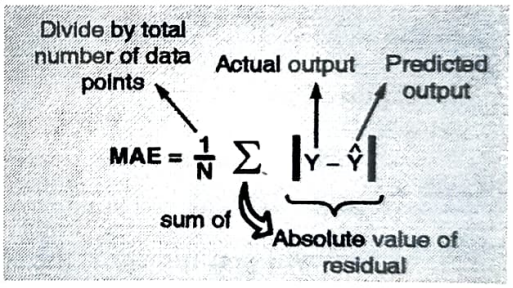


Fig. 3.11.1

Advantages of MAE

- The MAE you get is in the same unit as the output variable.
- It is most Robust to outliers.

Disadvantages of MAE

- The graph of MAE is not differentiable so we have to apply various optimizers like Gradient descent which can be differentiable.

```
from sklearn.metrics import mean_absolute_error
print("MAE",mean_absolute_error(y_test,y_pred))
```

3.12 ROOT MEAN SQUARED ERROR(RMSE)

As RMSE is clear by the name itself, that it is a simple square root of mean squared error.

$$RMSE = \sqrt{MSE}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(yj - \hat{y}_j)}$$

☞ **Advantages of RMSE**

The output value you get is in the same unit as the required output variable which makes interpretation of loss easy.

☞ **Disadvantages of RMSE**

- It is not that robust to outliers as compared to MAE.
- Most of the time people use RMSE as an evaluation metric and mostly when you are working with deep learning techniques the most preferred metric is RMSE.

3.13 R SQUARED (R2)

- R2 score is a metric that tells the performance of your model.
- In contrast, MAE and MSE depend on the context whereas the R2 score is independent of context.
- Hence, R2 squared is also known as Coefficient of Determination or sometimes also known as Goodness of fit.

$$R2\ Squared = 1 - \frac{SSr}{SSm}$$

SSr = Squared sum error of regression line

SSm = Squared sum error of mean line

- Now, how will you interpret the R2 score ? suppose If the R2 score is zero then the above regression line by mean line is equal means 1 so 1-1 is zero. So, in this case, both lines are overlapping means model performance is worst, It is not capable to take advantage of the output column.
- Now the second case is when the R2 score is 1, it means when the division term is zero and it will happen when the regression line does not make any mistake, it is perfect. In the real world and it is not possible. So we can conclude that as our regression line moves towards perfection, R2 score move towards one. And the model performance improves.
- The normal case is when the R2 score is between zero and one like 0.8 which means your model is capable to explain 80 per cent of the variance of data.

Chapter Ends ...

□□□