Regularization Regression Recall Random Variable Random Forest R-Squared

Probability Density Function (PDF)

RIDGE AND LASSO REGRESSION

•

INTRODUCTION

In this post we will try to understand about regularization and hyperparameter-tuning using **Ridge** and **Lasso Regression.** Before that we need to understand few concepts of **Linear Regression.**

I will provide a brief explanation here which would suffice our motive of this topic, however if you want to get a more in-depth understanding of **Linear Regression** and the math behind it, please click here.

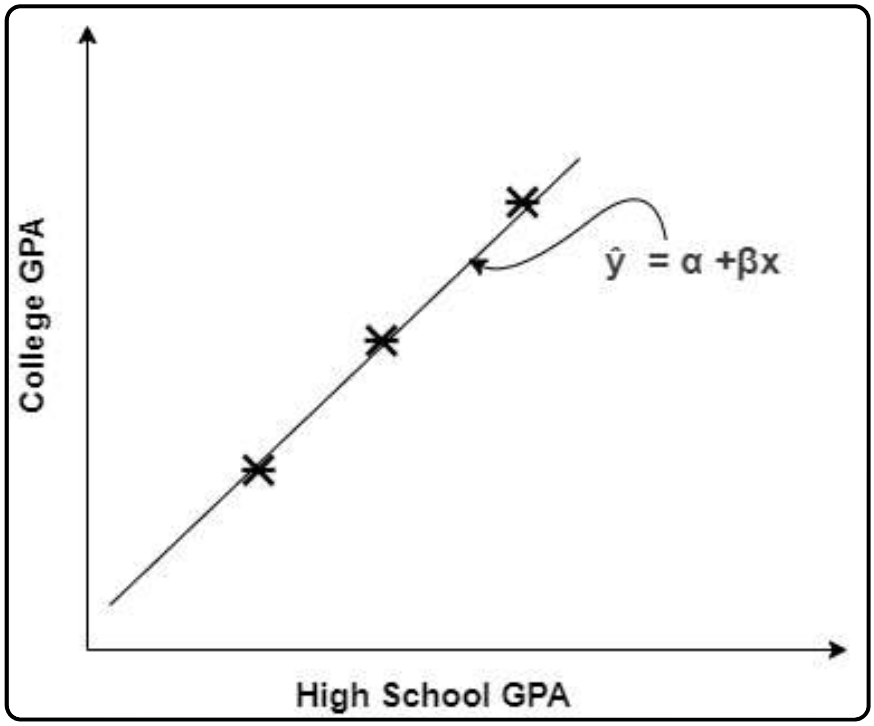
LINEAR REGRESSION

Let's go ahead and take an example to understand the idea behind **Linear Regression.**

We have the High School GPA for students of a batch. We want to predict the College GPA for students based on High School GPA**.**

For understanding purposes, we are considering the training set with only 3 data points. The best fit line is passing through all the 3 data points. This means that our data is showing very good results while **training** as shown below in the figure.

In the figure, $\hat{y}$ is the best fit line.



The **cost function** for the best fit like will be given by the below formula:

Cost Function (J)/Sum of Residuals : $\sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)} \right)^2$

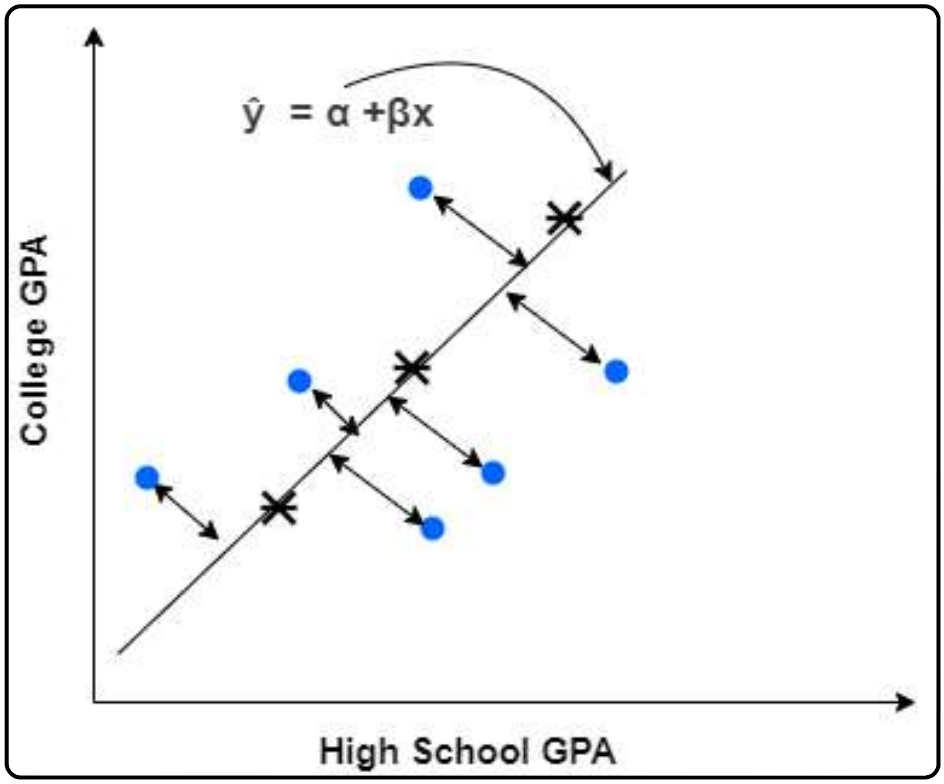
By just seeing the graph, we could clearly say that **ŷ** is equal to **y** as the line passes through each of the points. Substituting the values of **y** and **ŷ** to the equation for **cost function/sum of residuals** we will get the value as 0.

Wow!! That looks perfect.

Getting the value of **Cost Function** as 0 means our model is performing great on the **Training data**.

Well, don't get excited too soon!!

We have just trained our model. Running it on the test set will tell us how accurate or inaccurate our model is.



As you can see from the above graph, we ran the model on the test data and the model performance was not great.

The blue dots in the graph are the testing data points and as you could see, distance between the best fit line and the blue points are consistently high.

OVERFITTING – HIGH VARIANCE

This is a condition of **Overfitting** where the model performance is accurate for **Training** set whereas for **Testing** set, it performs poorly.

This **overfitting** model that we saw here has a very **high variance**. Our goal should always be to select such model which has **low bias** and **low variance**. Such models are generalized models as they behave pretty much the same for both **testing** and **training** sets.

Now, that we have discussed about **Linear Regression** and the **overfitting** condition, let's discuss how we could avoid this **overfitting** situation so that our model performs better.

1. RIDGE REGRESSION

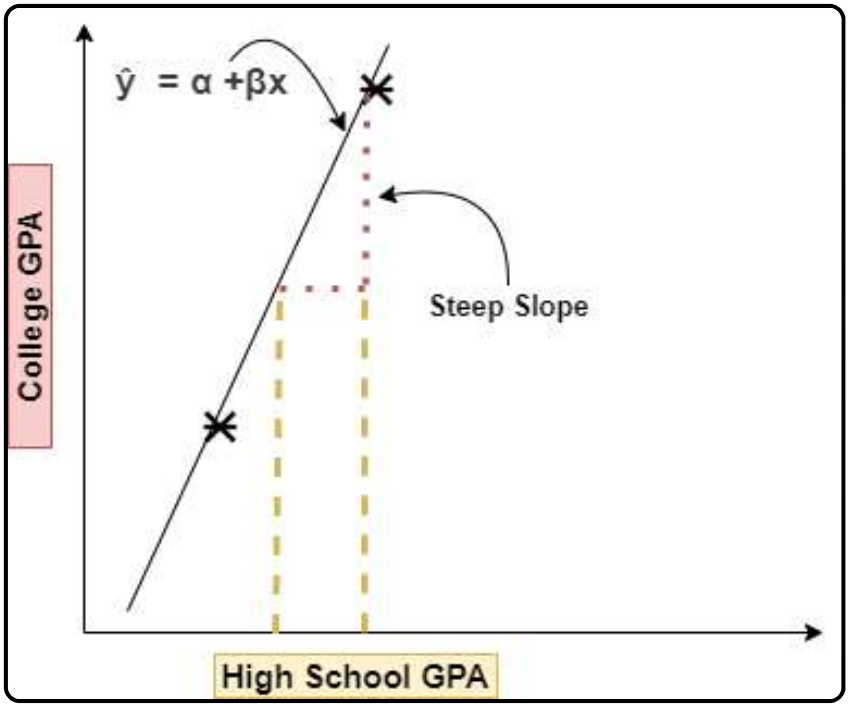
The basic difference between **Linear regression** and **Ridge regression** is, in case of linear regression we try to reduce the **cost function**. However, in **Ridge Regression** we add 2 more parameters to the cost function –> $\lambda * (Slope)^2$. Let's understand what these are.

The formula for **Ridge Regression** is $\sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2 + [\lambda * (Slope)^2]$

Unlike **Linear Regression** where we used to reduce only the **cost function**, in this case we will try to reduce the whole function which includes the 2 parameters also $\sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2 + [\lambda * (Slope)^2]$

ITERATION -1

To understand the significance of these parameters let's go back to our example of **overfitting condition.**



If you notice the above graph, the slope that we have is very steep. This means with the unit increase in the X- axis, there will be a comparatively larger increase in Y-axis.

Also, we have the best fit line aligning perfectly with the training data points causing an overfitting condition and making value for the cost function as 0.

To counter all these problems, we use the **Ridge Regression** where the parameters $\lambda * (Slope)^2$ are used to penalize the steep **slope**.

Let's understand this mathematically below:

$\sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2 = 0$, due to the **overfitting condition.**

$\lambda > 0$(let's take 1 for our understanding purpose)

Slope = High steep slope(Let's take 1.5 for our understanding purpose)

Substituting, the values to the whole equation **= 0 + (1) * (1.5)^2 = 2.25**

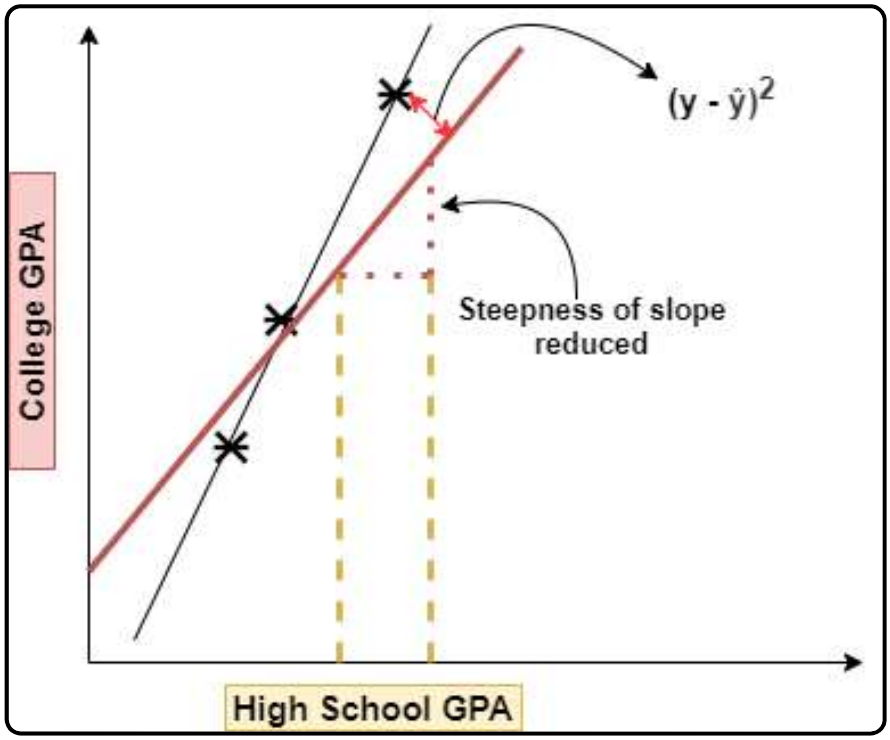
Now, the **cost function** that we got is 2.25.

One thing we need to notice here is, in case of **linear regression**, we stopped when the cost function was zero because our best fit line perfectly aligned with the data points.

However, in **Ridge Regression** we will not stop until we reach a particular value which would yield us a **generalized model.**

ITERATION-2

Now let's take another best fit line and calculate the cost function:



In the above graph, the value for **(y – ŷ)** which was zero in the overfit condition will now be a smaller value as our best fit line is not passing exactly through the data points.

Also, if you notice, the steepness of the curve has reduced a bit, which means the slope has also reduced by a smaller value.

Substituting the individual values to the cost function formula for **Ridge regression:** $\sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2 + [\lambda * (Slope)^2]$

Where,

$$\sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2 = small\ value$$

$\lambda > 0$ (let's take 1 for our understanding purpose)

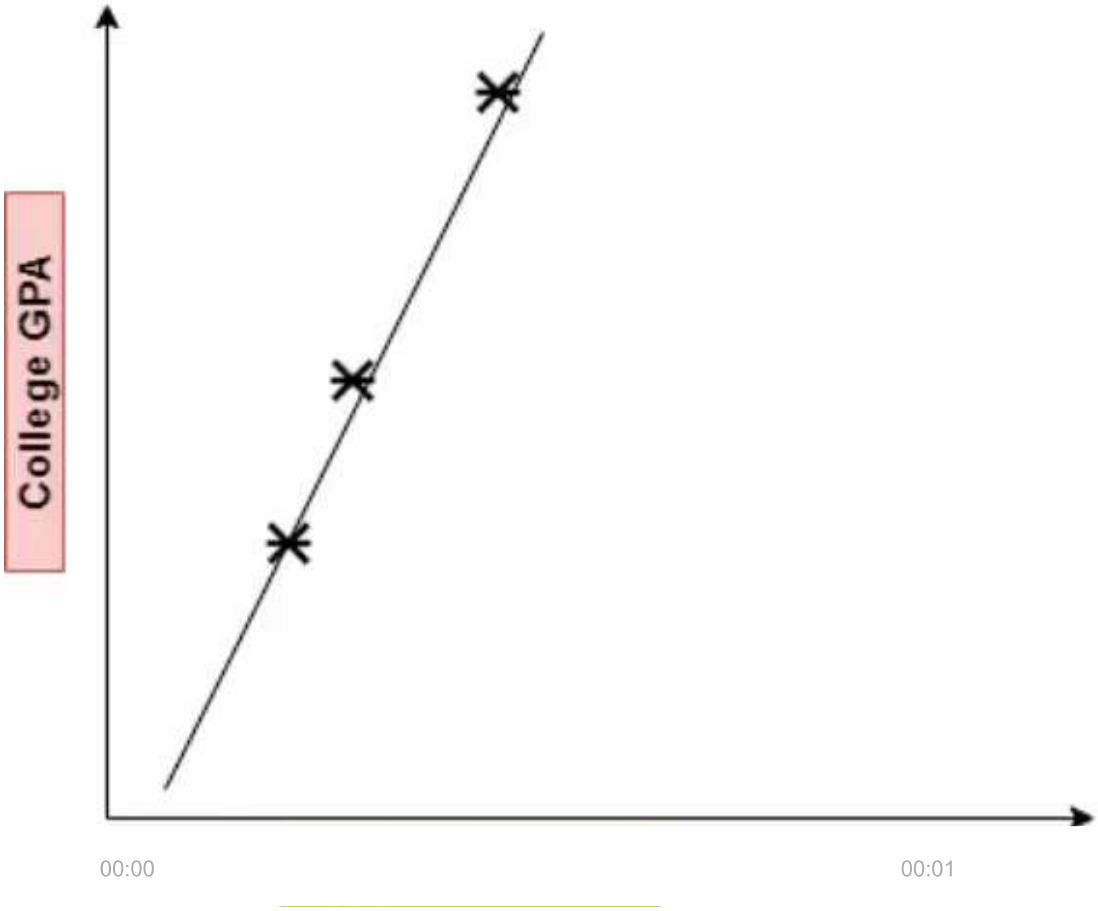
Slope = Less steep slope (Let's take 1.3 for our understanding purpose)

Substituting, the values to the whole equation **= small value + (1) * $(1.3)^2$ => small value +1.69**

The value that we will get will be less that the previous **cost function** output of **2.25** because we have a reducing slope in this case.

So, as we got a smaller **cost function** now compared to the previous iteration, we will select the current one as our best fit line.

So, if you notice, we are basically **penalizing** those features which are having **steeper slopes**. This is the basic motive of **Ridge Regression.**



If you see the video above, its basically iterating through different values of slope to find the best fit line. This will yield us our **generalized model.**

One thing to note here is, we will usually choose the λ to be a small and as it increases, the slope of our best fit line also approaches towards 0, but never becomes 0.

Now that we have discussed at length about **Ridge Regression** and how it can be used to overcome **overfitting condition**, let's understand another technique which is called **Lasso Regression.**

2. LASSO REGRESSION

The difference between the formula for Ridge and Lasso regression is instead of using $(Slope)^2$ as we did in case of Ridge, we will be using the magnitude of slope: $|Slope|$

Below is the formula for **Ridge Regression:**

$$\sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2 + [\lambda * |Slope|]$$

The advantage of **Lasso Regression** is it helps to overcome the **overfitting condition**, and also helps in **feature selection**.

To understand how it helps in **Feature selection**, let's take the below equation with multiple features.

$$\mathbf{y = m_1x_1 + m_2x_2 + m_3x_3 + ... C}$$

Now if we take the magnitude of the slopes: $|m1 + m2 + m3 + ...|$ the ones with very low values of slope will be removed. This basically means that the overall slope of the best fit line would move towards zero over the period of multiple iterations. It might also reach zero.

The difference between **Ridge** and **Lasso** is, in case of **Ridge**, the slope of the overall curve will always approach towards zero but will never reach zero.

Whereas in **Lasso** it could reach zero in cases where we have very small values of slopes for different features. The reason for this is, the features

that won't have any impact on the model's overall performance will be getting removed as we are considering the magnitude of slope in **Lasso.** This will make the overall value of slope smaller or may be zero.

Now let's implement the **Ridge** and **Lasso regression** in code and see how it works !!

About the Dataset

The data was collected and made available by "National Institute of Diabetes and Digestive and Kidney Diseases" as part of the Pima Indians Diabetes Database. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here belong to the Pima Indian heritage (subgroup of Native Americans), and are females of ages 21 and above.

The columns of the dataset are as follows:

1. Pregnencies
2. Glucose
3. BloodPressure
4. Skin Thickness
5. Insulin
6. BMI
7. DiabetesPedigreeFunction
8. Age
9. Outcome

You can download the dataset from here: https://www.kaggle.com/kandij/diabetes-dataset (https://www.kaggle.com/kandij/diabetes-dataset)

In this exercise, we will do hyper parameter tuning using Ridge and Lasso Regression. Before

REFERENCES :

1. https://www.analyticsvidhya.com/blog/2017/06/a-comprehensive-guide-for-linear-ridge-and-lasso-regression/
2. https://en.wikipedia.org/wiki/Lasso_(statistics)
3. https://en.wikipedia.org/wiki/Tikhonov_regularization
4. https://www.youtube.com/watch?v=Xm2C_gTAl8c