NAME: Dipesh Ramesh Limaje

INTERNSHIP BATCH : 33

TOPIC: MACHINE LEARNING

SME : Mr. Shwetank Mishra

WORKSHEET NO : 6

Q1 Ans C High R-squared value for train-set and Low R-squared value for test-set.

Q2 Ans B Decision trees are highly prone to overfitting.

Q3 Ans C Random Forest

Q4 Ans B Sensitivity

Q5 Ans B Model B

Q6 Ans A & D Ridge & Lasso

Q7 Ans C & B Decision Tree & Random Forest

Q8 Ans D All of the above

Q9 Ans B A tree in the ensemble focuses more on the data points on which the previous tree was not performing well

Q10 Explain how does the adjusted R-squared penalize the presence of unnecessary predictors in the model?

ANS The adjusted R-squared is a modified version of the R-squared metric that takes into account the number of predictors in the model. The adjusted R-squared penalizes the presence of unnecessary predictors in the model by reducing the value of the R-squared as the number of predictors increases.

The formula for the adjusted R-squared is:

$$\text{Adjusted R-squared} = 1 - (1 - \text{R-squared}) * (n - 1) / (n - p - 1)$$

where n is the number of observations in the dataset and p is the number of predictors in the model.

The denominator (n - p - 1) represents the degrees of freedom in the residuals, and the numerator (1 - R-squared) represents the residual sum of squares.

As the number of predictors increases, the denominator decreases, which makes the adjusted R-squared decrease. This reduction in adjusted R-squared represents the increased complexity of the model and the potential for overfitting. In other words, the adjusted R-squared penalizes models with a large number of predictors relative to the number of observations.

Therefore, adjusted R-squared can be used as a tool for selecting the most parsimonious models that provide the best balance between the goodness of fit and model complexity.

Q11 Differentiate between Ridge and Lasso Regression.
ANS

Ridge Regression	Lasso Regression
1] Ridge regression is also called L1 Form	1] Lasso regression is also called L2 Form
2] Ridge penalizes the sum of the absolute value of the weights.	2] Lasso penalizes the sum of squares of the weights.
3] Ridge has a sparse solution	3] Lasso has the non-sparse solution
4] Ridge has multiple solutions	4] Lasso has only one solution
5] Ridge has built-in feature selection	5] Lasso has no feature selection
6] Ridge is robust to outliers	6] Lasso is not robust to outliers
7] Ridge generates models that are simple and interpretable but cannot learn complex patterns	7] Lasso regularization is able to learn complex data pattern

Q12 What is VIF? What is the suitable value of a VIF for a feature to be included in a regression modelling?

ANS A variance inflation factor(VIF) detects multicollinearity in regression analysis. Multicollinearity is when there's correlation between predictors (i.e. independent variables) in a model; it's presence can adversely affect your regression results. The VIF estimates how much the variance of a regression coefficient is inflated due to multicollinearity in the model.VIFs are usually calculated by software, as part of regression analysis. You'll see a VIF column as part of the output. VIFs are calculated by taking a predictor, and regressing it against every other predictor in the model.

$$VIF = 1 / 1-R^2$$

Interpreting the Variance Inflation Factor

Variance inflation factors range from 1 upwards. The numerical value for VIF tells you (in decimal form) what percentage the variance (i.e. the standard error squared) is inflated for each coefficient. For example, a VIF of 1.9 tells you that the variance of a particular coefficient is 90% bigger than what you would expect if there was no multicollinearity — if there was no correlation with other predictors.

A rule of thumb for interpreting the variance inflation factor:

1 = not correlated.

Between 1 and 5 = moderately correlated.

Greater than 5 = highly correlated.

Exactly how large a VIF has to be before it causes issues is a subject of debate. What is known is that the more your VIF increases, the less reliable your regression results are going to be. In

general, a VIF above 10 indicates a high correlation and is cause for concern. Some authors suggest a more conservative level of 2.5 or above. Sometimes a high VIF is no cause for concern at all. For example, you can get a high VIF by including products or powers from other variables in your regression, like x and x2. If you have high VIFs for dummy variables representing nominal variables with three or more categories, those are usually not a problem.

Q13 Why do we need to scale the data before feeding it to train the model?

ANS After building an understanding of how to do data scaling and which data scaling techniques to use, we can now talk about where to use these data scaling techniques.

If an algorithm uses gradient descent, then the difference in ranges of features will cause different step sizes for each feature. To ensure that the gradient descent moves smoothly towards the minima and that the steps for gradient descent are updated at the same rate for all the features, we scale the data before feeding it to the model. Having featured on a similar scale will help the gradient descent converge more quickly toward the minima.

Specifically, in the case of Neural Networks Algorithms, feature scaling benefits optimization by:

- It makes the training faster
- It prevents the optimization from getting stuck in local optima
- It gives a better error surface shape
- Weight decay and Bayes optimization can be done more conveniently

Methods for Scaling

Now, since you have an idea of what is feature scaling. Let us explore what methods are available for doing feature scaling. Of all the methods available, the most common ones are: Normalization and Standardization

Normalization is good to use when the distribution of data does not follow a Gaussian distribution. It can be useful in algorithms that do not assume any distribution of the data like K-Nearest Neighbors.

In Neural Networks algorithm that require data on a 0–1 scale, normalization is an essential pre-processing step. Another popular example of data normalization is image processing, where pixel intensities have to be normalized to fit within a certain range (i.e., 0 to 255 for the RGB color range).

Standardization can be helpful in cases where the data follows a Gaussian distribution. Though this does not have to be necessarily true. Since standardization does not have a bounding range, so, even if there are outliers in the data, they will not be affected by standardization.

In clustering analyses, standardization comes in handy to compare similarities between features based on certain distance measures. Another prominent example is the Principal Component Analysis, where we usually prefer standardization over Min-Max scaling since we are interested in the components that maximize the variance.

Q14 What are the different metrics which are used to check the goodness of fit in linear regression?

ANS There are three error metrics that are commonly used for evaluating and reporting the performance of a regression model; they are:

1. Mean Squared Error (MSE).
2. Root Mean Squared Error (RMSE).
3. Mean Absolute Error (MAE)

There are many other metrics for regression, although these are the most commonly used. You can see the full list of regression metrics supported by the scikit-learn Python machine learning library

Mean Squared Error

Mean Squared Error, or MSE for short, is a popular error metric for regression problems.

It is also an important loss function for algorithms fit or optimized using the least squares framing of a regression problem. Here "least squares" refers to minimizing the mean squared error between predictions and expected values.

The MSE is calculated as the mean or average of the squared differences between predicted and expected target values in a dataset.

$$MSE = 1 / N * \text{sum for i to N } (y\_i - yhat\_i)^2$$

Where y_i is the i'th expected value in the dataset and yhat_i is the i'th predicted value. The difference between these two values is squared, which has the effect of removing the sign, resulting in a positive error value.

Root Mean Squared Error

The Root Mean Squared Error, or RMSE, is an extension of the mean squared error.

Importantly, the square root of the error is calculated, which means that the units of the RMSE are the same as the original units of the target value that is being predicted.

For example, if your target variable has the units "dollars," then the RMSE error score will also have the unit "dollars" and not "squared dollars" like the MSE.

As such, it may be common to use MSE loss to train a regression predictive model, and to use RMSE to evaluate and report its performance.

The RMSE can be calculated as follows:

$$RMSE = sqrt(1 / N * \text{sum for i to N } (y\_i - yhat\_i)^2)$$

Where y_i is the i'th expected value in the dataset, yhat_i is the i'th predicted value, and sqrt() is the square root function.

We can restate the RMSE in terms of the MSE as:

$$RMSE = sqrt(MSE)$$

Mean Absolute Error

Mean Absolute Error, or MAE, is a popular metric because, like RMSE, the units of the error score match the units of the target value that is being predicted.

That is, MSE and RMSE punish larger errors more than smaller errors, inflating or magnifying the mean error score. This is due to the square of the error value. The MAE does not give more or less weight to different types of errors and instead the scores increase linearly with increases in error.

As its name suggests, the MAE score is calculated as the average of the absolute error values. Absolute or abs() is a mathematical function that simply makes a number positive. Therefore, the difference between an expected and predicted value may be positive or negative and is forced to be positive when calculating the MAE.

The MAE can be calculated as follows:

$$MAE = 1 / N * \text{sum for i to N } abs(y\_i - yhat\_i)$$

Where y_i is the i'th expected value in the dataset, yhat_i is the i'th predicted value and abs() is the absolute function.

Q15 From the following confusion matrix calculate sensitivity, specificity, precision, recall and accuracy.

ANS

Actual / Predicted	True	False
True	1000 (TP)	50 (FP)
False	250 (FN)	1200 (TN)

1] **Accuracy :** It is defined as the total Number of Correct classification Divided By the total Number of classifications.

Accuracy = TP + TN / (TP + TN + FP + FN)

= 1000 + 1200 / (1000 + 50 + 250 + 1200)

= 2200 / 2500

= 0.88

Therefore The accuracy is 88%

2] **Sensitivity**: It is the measure of the total number of positive results how many positives were corrected by the model.

Sensitivity = TP / (TP + FN)

= 1000 / (1000 + 250)

= 1000/1250

= 0.8

Therefore The Sensitivity is 80%

3] **Precision**: Precision is a measure of among all positive predictions, how many among them were actually positive.

Precision = TP / (TP + FP)

= 1000 / (1000 + 50)

= 1000 / 1050

= 0.9523

Therefore Precision is 95.23%

4] **F1_Score** : We have Recall and precisiom metrics but when to use recall and when to use precision so we do trade off technique that is F1_Score.

F1 Score = 2 * (Precision * Recall) / (Precision + Recall)

= 2 * (95.23 * 80) / (95.23 + 80)

= 2 * 7618.4 / 175.23

= 15236.8 / 175.23

= 86.95

Therefore the F1_Score is 86.95%

5] **Specificity :** The total number of negative predicted by the model with respect to the total number of actual Negative.

Specificity = TN / (TN + FP)

= 1200 / (1200 + 50)

= 1200 / 1250

= 0.96

Therefore the Specificity is 96%.