 ASSIGNMENT - 5

MACHINE LEARNING

**Q1 to Q15 are subjective answer type questions, Answer them briefly.**

1. R-squared or Residual Sum of Squares (RSS) which one of these two is a better measure of goodness of fit model in regression and why?

**RSS(Residual Sum of Squares) is useful when you want to evaluate the absolute goodness of fit, focusing on the magnitude of the prediction errors. If minimizing prediction errors is a primary concern, RSS is a more appropriate choice.**

R-squared is often used when you want to understand the proportion of variance explained by the model, especially in the context of comparing different models. It can provide insights into how well the predictors collectively contribute to explaining the variation in the dependent variable.

**RSS is useful when you want to evaluate the absolute goodness of fit, focusing on the magnitude of the prediction errors. If minimizing prediction errors is a primary concern, RSS is a more appropriate choice.**

R-squared (R²) and Residual Sum of Squares (RSS) are both commonly used measures to assess the goodness of fit of a regression model, but they capture different aspects of model performance, and the choice between them depends on the context and what we want to evaluate.

**R-squared** provides a high-level summary of the explanatory power of your model. R-squared can be useful for comparing different models or assessing how much of the variation in the dependent variable can be attributed to the predictors. **However, it has limitations. For example, it can be artificially inflated by adding more predictors to a model, even if those predictors do not have a meaningful relationship with the dependent variable.** - R-squared is a relative measure, and it does not provide information about the absolute goodness of fit or the quality of the model's predictions

**RSS** gives you a detailed view of the model's prediction accuracy. (The choice between them should align with your specific goals and the questions we want to answer about our regression model.)

RSS measures the total squared difference between the observed values of the dependent variable and the predicted values from the regression model. It quantifies the overall error or "residuals" in the model's predictions.

- A smaller RSS indicates a better fit because it means that the model's predictions are closer to the actual observed values.

**RSS is an absolute measure of the goodness of fit. It tells you how well the model fits the data in terms of minimizing prediction errors. Unlike R-squared, RSS does not provide information about the proportion of variance explained but gives you a direct measure of the model's prediction accuracy.**

1. What are TSS (Total Sum of Squares), ESS (Explained Sum of Squares) and RSS (Residual Sum of Squares) in regression. Also mention the equation relating these three metrics with each other.

**TSS = ESS + RSS,**

**( where TSS is Total Sum of Squares, ESS is Explained Sum of Squares and RSS is Residual Sum of Suqares). The aim of Regression Analysis is explain the variation of dependent variable Y**

**Total Sum of Squares (TSS)** :

is defined as the sum of errors of the data points from the mean of the response variable. Mathematically TSS is, Total Sum of Squares

The Total SS (TSS or SST) tells you how much variation there is in the [dependent variable](https://www.statisticshowto.com/dependent-variable-definition/).  
Total SS = Σ(Yi – mean of Y)2.  
**Note**: Sigma (Σ) is a mathematical term for [summation](http://www.columbia.edu/itc/sipa/math/summation.html) or “adding up.” It’s telling you to add up all the possible results from the rest of the equation.

Sum of squares is a measure of how a data set varies around a central number (like the [mean](https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/mean-median-mode/#mean)). You might realize by the phrase that you’re summing (*adding up*) squares—but squares of what? You’ll sometimes see this formula:  
[ss2](https://www.statisticshowto.com/wp-content/uploads/2015/04/ss2.jpg)

where y is the mean of the sample data points.

**Explained Sum of Squares (ESS) :**

The Explained SS tells you how much of the variation in the dependent variable your model explained.

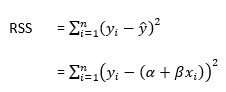
Explained SS = Σ(Y-Hat – mean of Y)2.

**Residual sum of Squares (RSS)** :

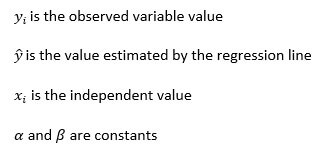
is defined as the sum of squares of the residual for each data point in the plot/data. It is the measure of the difference between the expected and the actual observed output.

RSS is a statistical method used to detect the level of discrepancy in a dataset not revealed by regression. If the residual sum of squares results in a lower figure, it signifies that the regression model explains the data better than when the result is higher. In fact, if its value is zero, it’s regarded as the best fit with no error at all.

Residual Sum of Squares (RSS) :



Where,



1. What is the need of regularization in machine learning?

**Regularization in Machine Learning:**

Regularization refers to techniques used to calibrate machine learning models to minimize the adjusted loss function and avoid overfitting or underfitting.

Regularization tries to reduce the variance of the model, without a substantial increase in the bias. It is one of the most important concepts of machine learning. This technique prevents the model from overfitting by adding extra information to it.

It is a form of regression that shrinks the coefficient estimates towards zero. In other words, this technique forces us not to learn a more complex or flexible model, to avoid the problem of overfitting. “In the Regularization technique, we reduce the magnitude of the independent variables by keeping the same  number of variables”. It maintains accuracy as well as a generalization of the model.

**Regularization works by adding a penalty or complexity term or shrinkage term with Residual Sum of Squares (RSS) to the complex model.**

Techniques of Regularization : Mainly, there are two types of regularization techniques, which are given below:

* Ridge Regression
* Lasso Regression

1. What is Gini–impurity index?

**Gini Index is a way of splitting a decision tree. Gini Index to originate binary splits. Both Gini Index and Gini Impurity are used interchangeably**.

The Entropy and Information Gain method focuses on purity and impurity in a node**. The Gini Index or Impurity measures the probability for a random instance being misclassified when chosen randomly. The lower the Gini Index, the better the lower the likelihood of misclassification.**

**The Gini Impurity formula is: 1 – (p₁)² – (p₂)²,**

where p₁ and p₂ represent the probabilities of the two classes in a binary classification problem.

1. Are unregularized decision-trees prone to overfitting? If yes, why?

**Yes, unregularized decision-trees are prone to overfitting**. Overfitting is a common problem, a data scientist needs to handle while training decision tree models. Comparing to other machine learning algorithms, decision trees can easily overfit. Is your Decision Tree Overfitting?

Overfitting refers to the condition when the model completely fits the training data but fails to generalize the testing unseen data. Overfit condition arises when the model memorizes the noise of the training data and fails to capture important patterns. A perfectly fit decision tree performs well for training data but performs poorly for unseen test data.

If the decision tree is allowed to train to its full strength, the model will overfit the training data. There are **various techniques** to prevent the decision tree model from overfitting. Such as :

* Pruning
* Ensemble

1. What is an ensemble technique in machine learning?

**Ensemble methods are techniques that aim at improving the accuracy of results in models by combining multiple models instead of using a single model. The combined models increase the accuracy of the results significantly.**

Ensemble learning is a machine learning technique that enhances accuracy and resilience in forecasting by merging predictions from multiple models. It aims to mitigate errors or biases that may exist in individual models by leveraging the collective intelligence of the ensemble.

The underlying concept behind ensemble learning is to combine the outputs of diverse models to create a more precise prediction. By considering multiple perspectives and utilizing the strengths of different models, ensemble learning improves the overall performance of the learning system. This approach not only enhances accuracy but also provides resilience against uncertainties in the data. By effectively merging predictions from multiple models, ensemble learning has proven to be a powerful tool in various domains, offering more robust and reliable forecasts.

1. What is the difference between Bagging and Boosting techniques?

**Bagging** : Bootstrap Aggregating, also known as bagging, is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It decreases the variance and helps to avoid overfitting. It is usually applied to decision tree methods. Bagging is a special case of the model averaging approach.

**Boosting** : is an ensemble modeling technique that attempts to build a strong classifier from the number of weak classifiers. It is done by building a model by using weak models in series. Firstly, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models is added.

**Bagging and Boosting: Differences**:

Bagging is a method of merging the same type of predictions. Boosting is a method of merging different types of predictions.

Bagging decreases variance, not bias, and solves over-fitting issues in a model. Boosting decreases bias, not variance.

In Bagging, each model receives an equal weight. In Boosting, models are weighed based on their performance.

Models are built independently in Bagging. New models are affected by a previously built model’s performance in Boosting.

In Bagging, training data subsets are drawn randomly with a replacement for the training dataset. In Boosting, every new subset comprises the elements that were misclassified by previous models.

Bagging is usually applied where the classifier is unstable and has a high variance. Boosting is usually applied where the classifier is stable and simple and has high bias.

1. What is out-of-bag error in random forests?

Out-of-bag (OOB) error, also called out-of-bag estimate, is a method of measuring the prediction error of random forests, boosted decision trees, and other machine learning models utilizing bootstrap aggregating (bagging). Bagging uses subsampling with replacement to create training samples for the model to learn from.

1. What is K-fold cross-validation?

K-fold cross-validation is a technique for evaluating predictive models. The dataset is divided into k subsets or folds. The model is trained and evaluated k times, using a different fold as the validation set each time. Performance metrics from each fold are averaged to estimate the model's generalization performance.

1. What is hyper parameter tuning in machine learning and why it is done?

Hyperparameters are external configuration variables that data scientists use to manage machine learning model training. Sometimes called model hyperparameters, the hyperparameters are manually set before training a model. Hyperparameters are parameters whose values control the learning process and determine the values of model parameters that a learning algorithm ends up learning.

Machine learning algorithms are incredibly powerful, but their performance can vary based on how their hyperparameters are set. **Hyperparameter tuning, also known as hyperparameter optimization, is the process of finding the best hyperparameters for a machine learning model to achieve optimal performance.**

In machine learning, hyperparameters are parameters that are not learned from the data but are set prior to training. They control various aspects of the learning process and can significantly impact the model's performance. While model parameters are learned during training (e.g., weights in a neural network), hyperparameters must be chosen beforehand.

Examples of hyperparameters include: 1. **Learning Rate**: Affects the step size in updating model parameters during training. It can impact both convergence speed and final accuracy.

2. **Number of Trees (for ensemble methods)**: In algorithms like Random Forest and Gradient Boosting, the number of trees in the ensemble is a critical hyperparameter.

3. **Kernel Choice (for Support Vector Machines)**: In SVMs, the choice of kernel (e.g., linear, polynomial, or radial basis function) is a hyperparameter that influences the decision boundary.

4. **Depth of Trees (for Decision Trees)**: The maximum depth of decision trees controls their complexity and ability to fit the training data.

**Why Hyperparameter Tuning Matters** : Hyperparameter tuning is essential for several reasons:

1. Model **Performance**: The right hyperparameters can significantly improve a model's performance, making it more accurate and effective.

2. **Overfitting/Underfitting Prevention**: Properly tuned hyperparameters can help prevent overfitting (fitting the training data too closely) or underfitting (failing to capture the data's underlying patterns).

3. **Generalization**: Tuning ensures that your model generalizes well to new, unseen data, which is the ultimate goal of machine learning.

4. **Resource Efficiency**: It can save computational resources by finding optimal hyperparameters without excessive trial and error.

1. What issues can occur if we have a large learning rate in Gradient Descent?

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 parameters. The cost function represents the discrepancy between the predicted output of the model and the actual output.

The goal of gradient descent is to find the set of parameters that minimizes this discrepancy and improves the model’s performance.

**An excessively large learning rate can cause the optimization algorithm to overshoot the optimal parameter values, leading to divergence or oscillations that hinder convergence.**

1. Can we use Logistic Regression for classification of Non-Linear Data? If not, why?

**No, it is not recommended**.

Non-linear (data) problems can’t be solved with logistic regression because it has a linear decision surface. Linearly separable data is rarely found in real-world scenarios.

Logistic Regression constructs linear boundaries, The major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables. It best works with Linear data

Therefore the model performance and target output may not be accurate.

1. Differentiate between Adaboost and Gradient Boosting.

AdaBoost and [Gradient Boosting (GBM)](https://aiml.com/what-is-gradient-boosting-gbm/) are both ensemble learning techniques that combine multiple weak learners to create a stronger model, but they differ in their approach to building the ensemble and updating the weights of the instances in the dataset.

The key differences between Adaboost and [Gradient Boosting](https://aiml.com/what-is-gradient-boosting-gbm/) are shown in the table below:

|  |  |
| --- | --- |
| **Gradient boosting** | **Adaptive Boosting** |
| This approach trains learners based upon minimising the loss function of a learner (i.e., training on the residuals of the model) | This method focuses on training upon misclassified observations. Alters the distribution of the training dataset to increase weights on sample observations that are difficult to classify. |
| Weak learners are decision trees constructed in a greedy manner with split points based on purity scores (i.e., Gini, minimise loss). Thus, larger trees can be used with around 4 to 8 levels. Learners should still remain weak and so they should be constrained (i.e., the maximum number of layers, nodes, splits, leaf nodes) | The weak learners incase of adaptive boosting are a very basic form of decision tree known as stumps. |
| All the learners have equal weights in the case of gradient boosting. The weight is usually set as the learning rate which is small in magnitude. | The final prediction is based on a majority vote of the weak learners’ predictions weighted by their individual accuracy. |

14.What is bias-variance trade off in machine learning?

In statistics and machine learning, the bias–variance tradeoff describes the relationship between a model's complexity, the accuracy of its predictions, and how well it can make predictions on previously unseen data that were not used to train the model.

**The bias-variance method is an approach in machine learning that analyzes the tradeoff between bias and variance to optimize model performance. By adjusting a model’s complexity, it aims to strike a balance between underfitting (high bias) and overfitting (high variance). This method guides the selection of appropriate models, helping to create accurate and robust predictions on new data**

It is important to understand prediction errors (bias and variance) when it comes to accuracy in any machine-learning algorithm. There is a tradeoff between a model’s ability to minimize bias and variance which is referred to as the best solution for selecting a value of [Regularization](https://www.geeksforgeeks.org/regularization-in-machine-learning/) constant. A proper understanding of these errors would help to avoid the overfitting and underfitting of a data set while training the algorithm.

## What is Bias?

The bias is known as the difference between the prediction of the values by the [Machine Learning](https://www.geeksforgeeks.org/machine-learning/) model and the correct value. Being high in biasing gives a large error in training as well as testing data. It recommended that an algorithm should always be low-biased to avoid the problem of underfitting. By high bias, the data predicted is in a straight line format, thus not fitting accurately in the data in the data set. Such fitting is known as the [Underfitting](https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/) of Data. This happens when the [hypothesis](https://www.geeksforgeeks.org/understanding-hypothesis-testing/) is too simple or linear in nature.

**What is Variance?**

The variability of model prediction for a given data point which tells us the spread of our data is called the variance of the model. The model with high variance has a very complex fit to the training data and thus is not able to fit accurately on the data which it hasn’t seen before. As a result, such models perform very well on training data but have high error rates on test data. When a model is high on variance, it is then said to as Overfitting of Data. Overfitting is fitting the training set accurately via complex curve and high order hypothesis but is not the solution as the error with unseen data is high. While training a data model variance should be kept low

**Bias Variance Tradeoff**

If the algorithm is too simple (hypothesis with linear equation) then it may be on high bias and low variance condition and thus is error-prone. If algorithms fit too complex (hypothesis with high degree equation) then it may be on high variance and low bias. In the latter condition, the new entries will not perform well. Well, there is something between both of these conditions, known as a Trade-off or Bias Variance Trade-off. This tradeoff in complexity is why there is a tradeoff between bias and variance. An algorithm can’t be more complex and less complex at the same time.

We try to optimize the value of the total error for the model by using the [Bias-Variance](https://www.geeksforgeeks.org/bias-vs-variance-in-machine-learning/) Tradeoff.



**The best fit will be given by the hypothesis on the tradeoff point.**

1. Give short description each of Linear, RBF, Polynomial kernels used in SVM.

**Kernel Method in SVMs**

Support Vector Machines (SVMs) use kernel methods to transform the input data into a higher-dimensional feature space, which makes it simpler to distinguish between classes or generate predictions.

The kernel function in SVMs is essential in determining the decision boundary that divides the various classes.

**Major Kernel Function in Support Vector Machine**

In Support Vector Machines (SVMs), there are several types of kernel functions that can be used to map the input data into a higher-dimensional feature space. The choice of kernel function depends on the specific problem and the characteristics of the data.

Here are some most commonly used kernel functions in SVMs:

**Linear Kernel**

A linear kernel is a type of kernel function used in machine learning, including in SVMs (Support Vector Machines). It is the simplest and most commonly used kernel function, and it defines the dot product between the input vectors in the original feature space

K(x, y) = x .y

Where x and y are the input feature vectors. The dot product of the input vectors is a measure of their similarity or distance in the original feature space.

When using a linear kernel in an SVM, the decision boundary is a linear hyperplane that separates the different classes in the feature space. This linear boundary can be useful when the data is already separable by a linear decision boundary or when dealing with high-dimensional data, where the use of more complex kernel functions may lead to overfitting.

**Polynomial Kernel**

A particular kind of kernel function utilised in machine learning, such as in SVMs, is a polynomial kernel (Support Vector Machines). It is a nonlinear kernel function that employs polynomial functions to transfer the input data into a higher-dimensional feature space.

**One definition of the polynomial kernel is:**

Where x and y are the input feature vectors, c is a constant term, and d is the degree of the polynomial, K(x, y) = (x. y + c)d. The constant term is added to, and the dot product of the input vectors elevated to the degree of the polynomial.

The decision boundary of an SVM with a polynomial kernel might capture more intricate correlations between the input characteristics because it is a nonlinear hyperplane.

The degree of nonlinearity in the decision boundary is determined by the degree of the polynomial.

The polynomial kernel has the benefit of being able to detect both linear and nonlinear correlations in the data. It can be difficult to select the proper degree of the polynomial, though, as a larger degree can result in overfitting while a lower degree cannot adequately represent the underlying relationships in the data.

In general, the polynomial kernel is an effective tool for converting the input data into a higher-dimensional feature space in order to capture nonlinear correlations between the input characteristics.

**Gaussian (RBF) Kernel**

The Gaussian kernel, also known as the radial basis function (RBF) kernel, is a popular kernel function used in machine learning, particularly in SVMs (Support Vector Machines). It is a nonlinear kernel function that maps the input data into a higher-dimensional feature space using a Gaussian function.

**The Gaussian kernel can be defined as:**

K(x, y) = exp(-gamma \* ||x - y||^2)

Where x and y are the input feature vectors, gamma is a parameter that controls the width of the Gaussian function, and ||x - y||^2 is the squared Euclidean distance between the input vectors.

When using a Gaussian kernel in an SVM, the decision boundary is a nonlinear hyper plane that can capture complex nonlinear relationships between the input features. The width of the Gaussian function, controlled by the gamma parameter, determines the degree of nonlinearity in the decision boundary.

One advantage of the Gaussian kernel is its ability to capture complex relationships in the data without the need for explicit feature engineering. However, the choice of the gamma parameter can be challenging, as a smaller value may result in under fitting, while a larger value may result in over fitting.

