
# FLIP ROBO TECHNOLOGIES

**Worksheet Set – 5**

**Batch DS2301**

**Intern: Vishnukanth**

**Mail ID: vishnukh25@gmail.com**

- ✓ **Machine Learning**
- ✓ **Statistics**

# Machine Learning

## Worksheet – 5

**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?**

R-squared and Residual Sum of Squares (RSS) are two different metrics used to evaluate the goodness of fit of a regression model.

R-squared is a measure of the proportion of variance in the dependent variable that is explained by the independent variables in the model. It ranges from 0 to 1, where a higher value indicates a better fit of the model. R-squared is useful for comparing the goodness of fit of different regression models, but it doesn't provide information about the magnitude of the errors in the model.

Residual Sum of Squares (RSS) is the sum of the squared differences between the actual and predicted values in the dependent variable. Lower values of RSS indicate a better fit of the model. Unlike R-squared, RSS provides information about the magnitude of the errors in the model. However, it doesn't provide a way to compare the goodness of fit between different regression models.

In conclusion, both R-squared and RSS have their own advantages and limitations, and the choice between them depends on the specific use case and the information needed.

**2. 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 (Total Sum of Squares)** measures the total variability in a dependent variable (also known as response or outcome variable) and is calculated as the sum of squared differences between each response value and the mean of all response values.

**ESS (Explained Sum of Squares)** measures the variability in the response variable that is explained by the predictor variables in a regression model. It is calculated as the sum of squared differences between each predicted response value (from the regression model) and the mean of all response values.

**RSS (Residual Sum of Squares)** measures the variability in the response variable that is not explained by the predictor variables. It is calculated as the sum of squared differences between each observed response value and the corresponding predicted response value (from the regression model).

The three metrics are related as follows:

**TSS = ESS + RSS**

This equation states that the total variability in the response variable is equal to the sum of the explained variability and the residual variability.

### 3. What is the need of regularization in machine learning?

Regularization is a technique used in machine learning to prevent overfitting, which occurs when a model is too complex and fits the training data too well, but does not generalize well to new, unseen data. Overfitting can lead to poor performance on the test data or when making predictions on new data.

Regularization adds a penalty term to the loss function that the model optimizes during training. The penalty term discourages the model from having too many parameters or having parameters with high magnitude. This effectively reduces the complexity of the model and helps it avoid overfitting.

There are two common types of regularization in machine learning: L1 regularization (also known as Lasso) and L2 regularization (also known as Ridge). The choice of which type to use depends on the specifics of the problem and the model being used.

Regularization is a critical component of many machine learning models and is used to improve their generalization ability and prevent overfitting, leading to better performance on new, unseen data.

### 4. What is Gini–impurity index?

Gini impurity index is a measure of how often a randomly chosen element from a set would be incorrectly labeled if it was randomly labeled according to the class distribution in the set. It is a commonly used metric in decision tree-based algorithms for binary classification, such as CART (Classification and Regression Trees) and Random Forest.

Gini impurity is calculated as the probability of misclassifying a randomly chosen element in the set. If a set contains elements from two classes, say class A and class B, with proportions p and (1-p) respectively, the Gini impurity index is calculated as:

**Gini impurity = 1 - (p^2 + (1-p)^2) = 2p(1-p)**

A set with a high Gini impurity index is considered to be "pure" if it contains elements from only one class, or if the proportion of elements from the two classes is close to 50-50. Conversely, a set with a low Gini impurity index is considered to be "impure" if it contains elements from both classes and the proportion of elements from the two classes is heavily skewed towards one class.

In decision tree algorithms, the goal is to minimize the Gini impurity index by splitting the set into two subsets in such a way that each subset is as pure as possible. The splits

are made based on the values of the predictor variables and the Gini impurity is used as a criterion to determine the best split.

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

Yes, unregularized decision trees are prone to overfitting. Decision trees are a type of non-parametric model that are built by recursively partitioning the data into smaller subsets based on the values of the predictor variables. This recursive partitioning can lead to overfitting if the tree is too deep or has too many leaves.

In an unregularized decision tree, there is no mechanism to prevent the tree from growing too complex or from fitting the training data too closely. This can result in a tree that has many branches and leaves that are specific to the training data and do not generalize well to new, unseen data.

Overfitting occurs because the tree may learn patterns in the noise or random fluctuations in the training data rather than the underlying relationship between the predictor and response variables. This leads to poor performance on the test data or when making predictions on new data.

Regularization techniques, such as pruning or using ensembles of trees, can help prevent overfitting in decision trees. These techniques aim to simplify the tree by reducing the number of branches and leaves or combining the predictions from multiple trees to produce a more robust model.

**6. What is an ensemble technique in machine learning?**

Ensemble techniques in machine learning are methods for combining multiple models to produce a more robust and accurate prediction. The idea behind ensemble techniques is that combining multiple models can lead to better performance than using a single model because the strengths of the individual models can complement each other and mitigate the weaknesses.

There are two main types of ensemble techniques:

**Bagging (Bootstrapped Aggregation):** This technique involves training multiple versions of the same model on different random samples (with replacement) of the training data. The final prediction is made by combining the predictions of all the individual models, typically by taking a mean or a majority vote. Bagging can help reduce overfitting by averaging out the noise in the training data and smoothing the predictions.

**Boosting:** This technique involves training multiple models in a sequence, where each model focuses on correcting the mistakes made by the previous models. The final prediction is made by combining the predictions of all the individual models, typically by taking a weighted sum. Boosting can help improve the accuracy of the model by focusing on the most difficult examples in the training data.

Ensemble techniques are widely used in machine learning and have been shown to be effective in a wide range of tasks, such as classification, regression, and anomaly detection. Some popular examples of ensemble techniques include Random Forest, Gradient Boosting, and AdaBoost.

**7. What is the difference between Bagging and Boosting techniques?**

Bagging and Boosting are two popular ensemble techniques in machine learning for combining multiple models to produce a more robust and accurate prediction. While both methods aim to improve the performance of a single model by combining the predictions of multiple models, they do so in different ways.

Bagging (Bootstrapped Aggregation) involves training multiple versions of the same model on different random samples (with replacement) of the training data. The final prediction is made by combining the predictions of all the individual models, typically by taking a mean or a majority vote. Bagging can help reduce overfitting by averaging out the noise in the training data and smoothing the predictions.

Boosting, on the other hand, involves training multiple models in a sequence, where each model focuses on correcting the mistakes made by the previous models. The final prediction is made by combining the predictions of all the individual models, typically by taking a weighted sum. Boosting can help improve the accuracy of the model by focusing on the most difficult examples in the training data. The weights assigned to the individual models in the weighted sum are determined by the misclassification rate or error of the individual models. The models with lower error have higher weights and contribute more to the final prediction.

In summary, Bagging aims to reduce overfitting by averaging out the noise in the training data, while Boosting aims to improve the accuracy of the model by focusing on the most difficult examples in the training data. Both techniques can lead to improved performance over a single model, but the choice of which to use depends on the specifics of the problem and the model being used.

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

In random forests, the out-of-bag (OOB) error is a measure of the accuracy of the model that can be estimated without using cross-validation. Random forests is an ensemble machine learning technique based on decision trees where multiple trees are trained on random subsets of the data.

When training a random forest, each tree is trained on a random subset of the data, which is different for each tree. On average, about one-third of the data points are left out or "out-of-bag" when each tree is trained. These out-of-bag data points can be used to estimate the performance of the model without the need for cross-validation.

The out-of-bag error is calculated by taking the mean prediction error of the individual trees on the out-of-bag data points. To calculate the prediction error, the individual

trees make predictions for the out-of-bag data points and these predictions are compared to the actual values of the response variable. The mean prediction error is then used as an estimate of the model's generalization error, which is a measure of how well the model is expected to perform on new, unseen data.

The OOB error can be a useful tool for evaluating the performance of a random forest model, especially when cross-validation is not possible or impractical due to the size or computational cost of the data. It can also provide insight into the bias and variance of the model, which can be useful for diagnosing overfitting or underfitting.

**9. What is K-fold cross-validation?**

K-fold cross-validation is a resampling technique used in machine learning to evaluate the performance of a model. The goal of cross-validation is to estimate the expected performance of a model on new, unseen data, and to prevent overfitting.

In K-fold cross-validation, the original data is divided into K equally-sized folds or subsets, where K is a positive integer chosen by the practitioner. For each fold, one of the K subsets is used as the validation set, while the remaining K-1 subsets are used as the training set. The model is trained on the training set and evaluated on the validation set. This process is repeated K times, with each fold used once as the validation set.

The performance of the model is then estimated by taking the average of the K performance estimates, each of which is based on the results of a different validation set. This average performance estimate is used as an estimate of the model's expected performance on new, unseen data.

K-fold cross-validation is a popular technique for model selection and hyperparameter tuning, as it provides a robust estimate of the model's performance and helps prevent overfitting by making use of all the data for both training and validation. The value of K can have an impact on the performance estimate, with larger values of K providing a more accurate estimate, but at the cost of increased computational cost. A common value for K is 10, but other values can be used depending on the specifics of the problem and the size of the data.

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

Hyperparameter tuning is the process of optimizing the hyperparameters of a machine learning model to achieve better performance on a given task. Hyperparameters are parameters that are set before training a model and control its behavior or structure. Examples of hyperparameters include the learning rate for gradient descent, the number of trees in a random forest, or the regularization parameter in a linear regression model.

The choice of hyperparameters can have a significant impact on the performance of a machine learning model, as different hyperparameter values can lead to different model

structures and behaviors. Hence, the goal of hyperparameter tuning is to find the best set of hyperparameters for a given task that lead to the highest performance.

Hyperparameter tuning is done by training multiple versions of the same model with different hyperparameter values and evaluating their performance using a validation set. The performance is then used to choose the best set of hyperparameters. Common techniques for hyperparameter tuning include grid search, random search, and Bayesian optimization.

Hyperparameter tuning is important because it helps to prevent overfitting and improve the generalization performance of the model. By training multiple versions of the model with different hyperparameters and evaluating their performance, the best set of hyperparameters that lead to good generalization performance can be found. This can result in a more accurate and robust model for the given task.

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

A large learning rate in gradient descent can result in several issues that can negatively impact the performance of the model. These issues include:

**Oscillations:** With a large learning rate, the model's parameters can oscillate and never converge to a minimum. This can lead to the model overshooting the minimum and oscillating back and forth, resulting in slow convergence or never converging at all.

**Overshooting:** A large learning rate can cause the model to overshoot the minimum, resulting in the model continuing to update its parameters and move away from the optimal solution.

**Divergence:** A large learning rate can cause the model to converge very slowly or even diverge, resulting in a poor solution or no solution at all.

**Poor Local Minima:** A large learning rate can cause the model to converge to a poor local minimum, instead of the global minimum. This can result in a suboptimal solution.

**Noisy Gradients:** If the gradient of the cost function is noisy, a large learning rate can amplify the noise, making it difficult for the model to converge to an optimal solution.

To avoid these issues, it is generally recommended to use a small learning rate in gradient descent. A learning rate that is too large can cause the model to converge slowly or not converge at all, while a learning rate that is too small can cause the model to converge slowly. The optimal learning rate is usually found by trying different values and selecting the one that results in the fastest convergence to a good solution.

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

Logistic regression is a linear model, and as such, it is not well-suited for classifying non-linear data. Logistic regression models the relationship between the input variables and the log-odds of the output class as a linear combination of the input variables. This means that logistic regression models are limited to linear decision boundaries

If the relationship between the input variables and the output class is non-linear, logistic regression may not be able to capture the complex relationships between the input variables and the output class. In such cases, a non-linear model, such as a decision tree, random forest, or support vector machine, may be more appropriate.

To handle non-linear relationships, it is common to transform the input variables into a higher-dimensional space using basis functions, such as polynomials, radial basis functions, or Fourier transforms. These transformed input variables can then be used as inputs to a linear model, such as logistic regression. This approach is known as kernel methods, and it allows the model to capture non-linear relationships in the data by projecting the data into a higher-dimensional space.

In summary, while logistic regression can be used for classification, it is limited to linear decision boundaries and may not be well-suited for classifying non-linear data. For non-linear data, a non-linear model or a kernel method may be a better choice.

**13. Differentiate between Adaboost and Gradient Boosting.**

AdaBoost and Gradient Boosting are both ensemble learning techniques used for classification and regression problems. However, they differ in several key ways:

**Base Learner:** Adaboost uses simple base learners, such as decision stumps (decision trees with only one split), whereas Gradient Boosting uses any base learner, including decision trees, regression trees, or shallow neural networks.

**Training Process:** Adaboost trains the base learners sequentially, with each base learner being trained on the residuals (errors) of the previous base learner. Gradient Boosting, on the other hand, trains the base learners in parallel, updating the weights of the samples after each iteration based on the gradient of the loss function.

**Loss Function:** Adaboost uses a binary cross-entropy loss function for classification problems and a mean squared error loss function for regression problems. Gradient Boosting, on the other hand, can use any differentiable loss function, including mean squared error, mean absolute error, and cross-entropy loss.

**Regularization:** Adaboost does not have a built-in regularization mechanism, but it is relatively insensitive to overfitting due to its sequential training process. Gradient Boosting, on the other hand, can use regularization techniques such as early stopping, shrinkage, or feature selection to prevent overfitting.

**Computational Complexity:** Adaboost is computationally simple, as it trains the base learners one by one. Gradient Boosting is more computationally expensive, as it trains the base learners in parallel and updates the weights of the samples after each iteration.

In conclusion, AdaBoost and Gradient Boosting are both powerful ensemble learning techniques, but they differ in their base learners, training process, loss function, regularization mechanism, and computational complexity. The choice of technique will depend on the specific requirements of the problem being solved.

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

The bias-variance trade-off is a fundamental concept in machine learning that refers to the balance between two sources of error in a model: bias and variance.

Bias refers to the error introduced by approximating the underlying relationship between the input variables and the output variable. A model with high bias oversimplifies the relationship between the inputs and outputs and may not fit the data well. This results in underfitting, where the model has poor performance on both the training and validation sets.

Variance, on the other hand, refers to the error introduced by the model's sensitivity to small fluctuations in the training data. A model with high variance overfits the training data, capturing the noise in the data instead of the underlying relationship. This results in poor generalization performance, where the model performs well on the training set but poorly on the validation or test set.

The bias-variance trade-off states that a model cannot have both low bias and low variance. Increasing the complexity of a model, such as by adding more features or increasing the depth of a decision tree, will reduce bias and increase variance. Decreasing the complexity of a model will reduce variance and increase bias.

Finding the optimal balance between bias and variance is a key goal in machine learning, and techniques such as regularization, ensemble learning, and cross-validation can be used to mitigate the impact of both bias and variance and improve the overall performance of a model.

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

Support Vector Machines (SVM) is a popular machine learning algorithm used for classification and regression problems. The choice of kernel function is an important factor in the performance of an SVM model.

**Linear Kernel:** The linear kernel is the simplest kernel function and represents the dot product between two input vectors. This kernel is used when the data is linearly separable.

**RBF (Radial Basis Function) Kernel:** The RBF kernel is a non-linear kernel that maps the input data into a higher-dimensional feature space through a radial basis function. This allows for non-linear decision boundaries to be learned.

**Polynomial Kernel:** The polynomial kernel is another non-linear kernel that maps the input data into a higher-dimensional feature space through a polynomial transformation. This kernel can be used to model polynomial relationships between the inputs and the output.

Each kernel has its own advantages and disadvantages, and the choice of kernel will depend on the specific requirements of the problem being solved. The linear kernel is computationally efficient and works well when the data is linearly separable. The RBF kernel is a flexible and powerful kernel, but it may overfit the data if the parameters are not set properly. The polynomial kernel is useful when modeling polynomial relationships between inputs and outputs, but it may also overfit the data if the polynomial degree is too high.

# STATISTICS

## Worksheet – 5

**Q1 to Q10 are MCQs with only one correct answer. Choose the correct option.**

1. Using a goodness of fit,we can assess whether a set of obtained frequencies differ from a set of frequencies.

a) Mean

b) Actual

c) Predicted

d) Expected

**Answer : d) Expected**

2. Chi - square is used to analyse

a) Score

b) Rank

c) Frequencies

d) All of these

**Answer : c) Frequencies**

3. What is the mean of a Chi Square distribution with 6 degrees of freedom?

a) 4 b) 12 c) 6 d) 8

**Answer : c) 6**

4. Which of these distributions is used for a goodness of fit testing?

a) Normal distribution

b) Chi - squared distribution

c) Gamma distribution

d) Poission distribution

**Answer : b) Chi - squared distribution**

5. Which of the following distributions is Continuous

a) Binomial Distribution

b) Hypergeometric Distribution

c) F Distribution

d) Poisson Distribution

**Answer : c) F Distribution**

6. A statement made about a population for testing purpose is called?

a) Statistic b) Hypothesis

c) Level of Significance d) Test Statistic

**Answer : b) Hypothesis**

7. If the assumed hypothesis is tested for rejection considering it to be true is called?

a) Null Hypothesis

b) Statistical Hypothesis

c) Simple Hypothesis

d) Composite Hypothesis

**Answer : a) Null Hypothesis**

8. If the Critical region is evenly distributed then the test is referred as?

a) Two tailed b) One tailed c) Three tailed d) Zero tailed

**Answer : a) Two tailed**

9. Alternative Hypothesis is also called as?

a) Composite hypothesis

b) Research Hypothesis

c) Simple Hypothesis

d) Null Hypothesis

**Answer : b) Research Hypothesis**

10. In a Binomial Distribution, if ‘n’ is the number of trials and ‘p’ is the probability of success, then the mean value is given by

a) np

b) n

**Answer : a) np**