Project Task 2

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?

- **ANS : R-squared**: This statistic represents the proportion of the variance in the dependent variable that is predictable from the independent variables. It provides a normalized measure (between 0 and 1) of how well the regression model explains the variability of the data. A higher R-squared value indicates a better fit.

- **Residual Sum of Squares (RSS):** This is the sum of the squared differences between the observed and predicted values. While it indicates the total deviation of the response values from the fit to the response values, it is not normalized and can be difficult to interpret on its own, especially when comparing models with different scales or units.

  In summary, R-squared is more intuitive and easier to interpret, making it a preferred measure for assessing the goodness of fit in regression models.

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.

ANS :

n regression analysis, TSS, ESS, and RSS are key metrics used to evaluate the fit of a model:

1. **Total Sum of Squares (TSS)**: This measures the total variance in the observed data. It is the sum of the squared differences between each observed value and the mean of the observed values. Mathematically, it is given by:

   TSS=i=1∑n(yi−y¯)2

   where (y_i) is the observed value and (\bar{y}) is the mean of the observed values.

2. **Explained Sum of Squares (ESS)**: Also known as the Regression Sum of Squares (SSR), this measures the variance explained by the regression model. It is the sum of the squared differences between the predicted values and the mean of the observed values:

   ESS=i=1∑n(y^i−y¯)2

   where (\hat{y}_i) is the predicted value.

3. **Residual Sum of Squares (RSS)**: This measures the variance that is not explained by the model, i.e., the sum of the squared differences between the observed values and the predicted values:

   RSS=i=1∑n(yi−y^i)2

The relationship between these three metrics is given by the equation:

TSS=ESS+RSS

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

ANS : Regularization is crucial in machine learning for several reasons:

1. **Preventing Overfitting:** Regularization techniques add a penalty to the model's complexity, discouraging it from fitting the noise in the training data. This helps the model generalize better to unseen data.
2. **Improving Model Performance:** By controlling the complexity of the model, regularization can lead to better performance on test data, as it reduces the variance without significantly increasing the bias.

3. **Feature Selection:** Regularization methods like Lasso (L1 regularization) can shrink some coefficients to zero, effectively performing feature selection and simplifying the model.
4. **Handling Multicollinearity:** In cases where features are highly correlated, regularization can help stabilize the estimates and improve the model's robustness.

Common regularization techniques include:

- **L1 Regularization (Lasso):** Adds the absolute value of the coefficients as a penalty term to the loss function.
- **L2 Regularization (Ridge):** Adds the squared value of the coefficients as a penalty term to the loss function.
- **Elastic Net:** Combines both L1 and L2 regularization.

By incorporating these penalties, regularization ensures that the model remains simple and interpretable, while still capturing the essential patterns in the data.

4. What is Gini–impurity index?

ANS : The **Gini impurity index** is a measure used in decision trees to evaluate the quality of a split. It quantifies the probability of a randomly chosen element being incorrectly classified if it was randomly labeled according to the distribution of labels in the dataset.

Mathematically, the Gini impurity for a dataset (D) with (k) classes is defined as:

$$Gini(D) = 1 - \sum_{i=1}^{k} p_i^2$$

where (p_i) is the probability of an element belonging to class (i).

Key points about Gini impurity:

- **Range**: The Gini impurity ranges from 0 (perfect purity, where all elements belong to a single class) to 0.5 (maximum impurity, where elements are equally distributed among all classes).
- **Usage:** It is used to decide the best feature to split the data at each node in a decision tree. The feature that results in the lowest Gini impurity is chosen for the split12.

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

ANS : Yes, unregularized decision trees are prone to overfitting. Here's why:

1. **High Variance:** Decision trees can create very complex models that perfectly fit the training data, capturing noise and outliers. This leads to high variance, where the model performs well on training data but poorly on unseen data.
2. **Deep Trees:** Without regularization, decision trees can grow very deep, with many branches. Each branch can represent a very specific condition, which may not generalize well to new data.
3. **Lack of Pruning:** Regularization techniques like pruning help to remove branches that have little importance and do not contribute significantly to the model's predictive power. Without pruning, the tree can become overly complex.
4. **Sensitive to Small Variations:** Unregularized trees can be very sensitive to small variations in the data, leading to different splits and a model that does not generalize well.

Regularization techniques such as pruning, setting a maximum depth, or requiring a minimum number of samples per leaf can help mitigate overfitting by simplifying the model and improving its generalization to new data.

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

ANS : An ensemble technique in machine learning involves combining multiple models to improve the overall performance and robustness of predictions. The idea is that by aggregating the predictions of several models, the ensemble can achieve better accuracy and generalization than any single model alone. Here are some common ensemble techniques:

1. **Bagging (Bootstrap Aggregating):** This technique involves training multiple instances of the same model on different subsets of the training data (created by random sampling with replacement). The final prediction is typically made by averaging the predictions (for regression) or taking a

majority vote (for classification). Random Forest is a popular example of a bagging method.

2. **Boosting:** Boosting trains models sequentially, with each new model focusing on correcting the errors made by the previous ones. The models are combined to form a strong learner. AdaBoost and Gradient Boosting are well-known boosting algorithms.
3. **Stacking (Stacked Generalization):** In stacking, multiple models (often of different types) are trained, and their predictions are used as input features for a meta-model, which makes the final prediction. This technique leverages the strengths of various models.
4. **Voting:** This method involves training multiple models and combining their predictions by voting. For classification, the final prediction is the class that gets the most votes. For regression, the final prediction is the average of all model predictions.

Ensemble techniques are powerful because they reduce the risk of overfitting and improve the model's ability to generalize to new data by leveraging the strengths of multiple models.

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

ANS : Bagging and Boosting are both ensemble techniques in machine learning, but they have distinct differences in how they build and combine models:

Bagging (Bootstrap Aggregating)

1. **Model Independence:** Each model in the ensemble is trained independently on different subsets of the training data.
2. **Data Sampling:** Uses random sampling with replacement to create different subsets of the training data.
3. **Parallel Training:** Models can be trained in parallel since they are independent of each other.
4. **Error Reduction:** Aims to reduce variance by averaging the predictions of multiple models, which helps in reducing overfitting.

5. **Example:** Random Forest, where multiple decision trees are trained on different subsets of the data.

Boosting

1. **Sequential Training:** Models are trained sequentially, with each new model focusing on correcting the errors made by the previous models.
2. **Data Weighting:** Adjusts the weights of the training data based on the errors of previous models, giving more importance to misclassified instances.
3. **Iterative Improvement:** Each model tries to improve the overall performance by learning from the mistakes of the previous models.
4. **Error Reduction:** Aims to reduce bias and variance by combining weak learners to form a strong learner.
5. **Example:** AdaBoost, Gradient Boosting, where each model is trained to correct the errors of the previous ones.

Key Differences

- **Training Process:** Bagging trains models independently and in parallel, while Boosting trains models sequentially, with each model learning from the errors of the previous ones.
- **Focus:** Bagging focuses on reducing variance and preventing overfitting, whereas Boosting focuses on reducing bias and improving the model's accuracy by correcting errors iteratively.

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

ANS : Out-of-bag (OOB) error is a method used in Random Forests to estimate the prediction error of the model without the need for a separate validation set. Here's how it works:

1. **Bootstrap Sampling:** In Random Forests, each tree is trained on a bootstrap sample, which is a random sample with replacement from the original dataset. This means some data points are included multiple times, while others are left out.

2. **Out-of-Bag Data:** The data points that are not included in the bootstrap sample for a particular tree are called out-of-bag data. On average, about one-third of the data points are left out of each bootstrap sample.
3. **Error Estimation:** After training, each tree can be used to predict the out-of-bag data points. The OOB error is calculated by aggregating the predictions for these out-of-bag data points across all trees and comparing them to the actual values.
4. **Advantages:** The OOB error provides an unbiased estimate of the model's performance because it uses data that was not seen during the training of each tree. This makes it a useful metric for evaluating the model without needing a separate validation set.

In summary, the OOB error is a built-in cross-validation method in Random Forests that helps assess the model's accuracy and generalization ability.

9. What is K-fold cross-validation?

ANS : K-fold cross-validation is a robust technique used to evaluate the performance of machine learning models. It helps ensure that the model generalizes well to unseen data by using different portions of the dataset for training and testing in multiple iterations. Here's how it works:

1. **Splitting the Data:** The dataset is randomly divided into (k) equal-sized subsets, or "folds."
2. **Training and Validation:** The model is trained (k) times. Each time, one of the (k) folds is used as the validation set, and the remaining (k-1) folds are used as the training set.
3. **Performance Measurement:** The performance metric (e.g., accuracy, mean squared error) is calculated for each of the (k) iterations.
4. **Averaging Results:** The overall performance of the model is obtained by averaging the performance metrics from all (k) iterations.

Steps in K-fold Cross-Validation:

1. **Randomly divide the dataset into (k) folds.**
2. **For each fold:**
    - Use the fold as the validation set.

- Use the remaining (k-1) folds as the training set.
- Train the model and evaluate its performance on the validation set.

3. Average the performance metrics across all (k) folds to get the final performance estimate.

Advantages:

- Reduces Overfitting: By using multiple training and validation sets, it provides a more reliable estimate of model performance.
- Efficient Use of Data: All data points are used for both training and validation, maximizing the use of the dataset.

Choosing (k):

Common choices for (k) are 5 or 10, as these values provide a good balance between bias and variance[12].

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

ANS : Hyperparameter tuning in machine learning is the process of selecting the optimal set of hyperparameters for a model. Hyperparameters are the configuration settings used to control the learning process and the structure of the model, such as learning rate, number of layers in a neural network, or the depth of a decision tree. Unlike model parameters, which are learned from the data, hyperparameters are set before the training process begins.

Why Hyperparameter Tuning is Important:

1. **Improves Model Performance:** Properly tuned hyperparameters can significantly enhance the accuracy and efficiency of a model.
2. **Prevents Overfitting/Underfitting:** Tuning helps in finding a balance between overfitting (model too complex) and underfitting (model too simple).
3. **Optimizes Training Time:** Efficient hyperparameter settings can reduce the time required to train the model.
4. **Enhances Generalization:** Well-tuned hyperparameters help the model generalize better to unseen data, improving its predictive performance.

Common Techniques for Hyperparameter Tuning:

1. **Grid Search:** Exhaustively searches through a specified subset of hyperparameters. It is simple but can be computationally expensive.
2. **Random Search:** Randomly samples hyperparameters from a specified distribution. It is more efficient than grid search for large hyperparameter spaces.
3. **Bayesian Optimization:** Uses probabilistic models to find the optimal hyperparameters by building a surrogate model of the objective function and iteratively improving it.
4. **Gradient-Based Optimization:** Uses gradient descent to optimize hyperparameters, typically used in neural networks.
5. **Automated Hyperparameter Tuning Tools:** Libraries like Scikit-Optimize, Hyperopt, and Keras Tuner provide advanced methods for hyperparameter optimization.

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

ANS : Using a large learning rate in Gradient Descent can lead to several issues:

1. **Overshooting :** The algorithm may take steps that are too large, causing it to overshoot the minimum of the loss function. This can result in the algorithm bouncing around the minimum rather than converging to it.
2. **Divergence:** In extreme cases, the steps may be so large that the algorithm diverges, meaning it moves away from the minimum and the loss function increases instead of decreasing.
3. **Instability:** Large learning rates can cause the optimization process to become unstable, with the loss function fluctuating wildly instead of steadily decreasing.

4. **Poor Convergence:** Even if the algorithm does not diverge, a large learning rate can lead to poor convergence, where the algorithm fails to find the true minimum and settles at a suboptimal point.

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

ANS : Logistic Regression is inherently a linear classifier, meaning it is designed to find a linear decision boundary between classes. However, it can be adapted to handle non-linear data through certain techniques:

Why Logistic Regression Struggles with Non-Linear Data:

- **Linear Decision Boundary:** Logistic Regression models a linear relationship between the input features and the log-odds of the outcome. This means it can only separate classes with a straight line (or hyperplane in higher dimensions), which is insufficient for non-linear data**.**

Adapting Logistic Regression for Non-Linear Data:

1. **Feature Engineering:** By creating new features that capture non-linear relationships (e.g., polynomial features, interaction terms), you can transform the data into a higher-dimensional space where a linear decision boundary can effectively separate the classes.
2. **Kernel Trick:** Similar to Support Vector Machines (SVMs), you can apply the kernel trick to Logistic Regression to implicitly map the input features into a higher-dimensional space, allowing for non-linear decision boundaries.
3. **Using Non-Linear Models:** Alternatively, you can use inherently non-linear models like Decision Trees, Random Forests, or Neural Networks, which are better suited for capturing complex patterns in the data.

In summary, while standard Logistic Regression is not suitable for non-linear data, you can extend it through feature engineering or use other models designed for non-linear classification.

13. Differentiate between Adaboost and Gradient Boosting.

ANS : AdaBoost and Gradient Boosting are both popular ensemble learning techniques, but they differ in their approaches and mechanisms:

AdaBoost (Adaptive Boosting)

1. **Focus:** AdaBoost focuses on improving the performance of weak learners by adjusting the weights of incorrectly classified instances. It gives more weight to misclassified instances so that subsequent learners focus more on these hard-to-classify cases.
2. **Sequential Learning:** Each weak learner is trained sequentially, with each new learner correcting the errors of the previous ones.
3. **Weight Adjustment:** After each iteration, the weights of misclassified instances are increased, and the weights of correctly classified instances are decreased.
4. **Algorithm:** Typically uses decision stumps (one-level decision trees) as weak learners.
5. **Sensitivity:** More sensitive to noisy data and outliers because it focuses on hard-to-classify instances.

Gradient Boosting

1. **Focus:** Gradient Boosting builds an ensemble by optimizing a loss function. Each new learner is trained to minimize the residual errors (the difference between the actual and predicted values) of the previous learners.
2. **Sequential Learning:** Similar to AdaBoost, Gradient Boosting also trains learners sequentially, but it focuses on reducing the overall prediction error.
3. **Residuals:** Each new learner is trained on the residuals of the previous learners, effectively learning from the mistakes made by the ensemble so far.
4. **Algorithm:** Can use various types of weak learners, but decision trees are commonly used.

5. **Flexibility:** More flexible and can handle both regression and classification problems. It also includes regularization techniques to prevent overfitting.

Key Differences

- **Error Focus:** AdaBoost adjusts weights based on misclassifications, while Gradient Boosting minimizes the residual errors.
- **Sensitivity:** AdaBoost is more sensitive to noisy data and outliers, whereas Gradient Boosting is generally more robust due to its focus on residuals.
- **Regularization:** Gradient Boosting includes regularization techniques to prevent overfitting, which AdaBoost does not inherently have.

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

ANS : The bias-variance trade-off is a fundamental concept in machine learning that describes the balance between two types of errors that affect the performance of predictive models **:**

Bias

- **Definition:** Bias refers to the error introduced by approximating a real-world problem, which may be complex, by a simplified model.
- **High Bias:** Models with high bias make strong assumptions about the data, leading to systematic errors. This often results in underfitting, where the model is too simple to capture the underlying patterns in the data.
- **Example:** Linear regression applied to a non-linear problem.

Variance

- **Definition:** Variance refers to the error introduced by the model's sensitivity to small fluctuations in the training data.
- **High Variance:** Models with high variance are highly flexible and can fit the training data very closely, including the noise. This often results in overfitting,

where the model captures noise as if it were a true pattern, leading to poor generalization to new data.

- **Example:** A very deep decision tree.

Trade-Off

- **Balancing Act:** The goal is to find a balance between bias and variance to minimize the total error. A model with low bias and low variance is ideal, but in practice, reducing one often increases the other.
- Total Error: The total error in a model can be expressed as the sum of bias squared, variance, and irreducible error (noise inherent in the data):

Total Error=Bias2+Variance+Irreducible Error

Visualization

- **High Bias, Low Variance:** The model is too simple and does not capture the complexity of the data (underfitting).
- **Low Bias, High Variance:** The model is too complex and captures noise in the data (overfitting).
- **Optimal Model:** Strikes a balance between bias and variance, capturing the underlying patterns without fitting the noise.

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

ANS : Sure! Here are brief descriptions of the Linear, RBF, and Polynomial kernels used in Support Vector Machines (SVM):

Linear Kernel

- **Description:** The linear kernel is the simplest kernel function. It is used when the data is linearly separable, meaning a straight line (or hyperplane in higher dimensions) can separate the classes.

- **Equation: (K(x, y) = x \cdot y)**
- **Use Case:** Effective for high-dimensional data where the number of features is large compared to the number of samples.

Radial Basis Function (RBF) Kernel

- **Description:** The RBF kernel, also known as the Gaussian kernel, is used for non-linear data. It maps the input features into an infinite-dimensional space, allowing the SVM to find a non-linear decision boundary.
- **Equation**: (K(x, y) = \exp(-\gamma |x - y|^2)), where (\gamma) is a parameter that defines the influence of a single training example**.**
- **Use Case:** Suitable for complex datasets where the relationship between features and classes is non-linear.

Polynomial Kernel

- **Description:** The polynomial kernel represents the similarity of vectors in a feature space over polynomials of the original variables. It can model non-linear relationships by considering polynomial combinations of the input features.
- **Equation: (K(x, y) = (\alpha x \cdot y + c)^d), where (\alpha), (c), and (d) are parameters.**
- **Use Case:** Useful for problems where the relationship between features is polynomial.

STATISTICS WORKSHEET

1. **d) Expected** - Using a goodness of fit, we can assess whether a set of obtained frequencies differ from a set of expected frequencies.
2. **c) Frequencies** - Chi-square is used to analyze frequencies.
3. **c) 6** - The mean of a Chi-Square distribution with 6 degrees of freedom is 6.
4. **b) Chi-squared distribution** - The Chi-squared distribution is used for goodness of fit testing.
5. **c) F Distribution** - The F distribution is continuous.
6. **b) Hypothesis** - A statement made about a population for testing purposes is called a hypothesis.
7. **a) Null Hypothesis** - If the assumed hypothesis is tested for rejection considering it to be true, it is called the null hypothesis.
8. **a) Two tailed** - If the critical region is evenly distributed, the test is referred to as two-tailed.
9. **b) Research Hypothesis** - The alternative hypothesis is also called the research hypothesis.
10. **a) np** - 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 (np).