FLIP ROBO

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?**

   Answer - The residual sum of squares (RSS) is the absolute amount of variation, whereas R-squared is the absolute amount of variation as a proportion of total variation.
   The residual standard error (RSE) is another statistical term used to describe the difference in standard deviations of observed values versus predicted values as shown by points in a regression analysis. It is a goodness-of-fit measure that can be used to analyze how well a set of data points fit with the actual model.

   RSE is computed by dividing the RSS by the number of observations in the sample less 2, and then taking the square root: RSE = $[RSS/(n-2)]^{1/2}$

   The residual sum of squares can be zero. The smaller the residual sum of squares, the better your model fits your data; the greater the residual sum of squares, the poorer your model fits your data. A value of zero means your model is a perfect fit.

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.**

   Answer – The **explained sum of squares (ESS),** alternatively known as the model sum of squares or sum of squares due to regression (SSR – not to be confused with the residual sum of squares (RSS) or sum of squares of errors), is a quantity used in describing how well a model, often a regression model, represents the data being modelled. In particular, the explained sum of squares measures how much variation there is in the modelled values and this is compared to the **total sum of squares (TSS)**, which measures how much variation there is in the observed data, and to the **residual sum of squares (RSS)**, which measures the variation in the error between the observed data and modelled values.

   total sum of squares (TSS) = explained sum of squares (ESS) + residual sum of squares (RSS).

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

   Answer – Regularization consists of different techniques and methods used to address the issue of over-fitting by reducing the generalization error without affecting the training error much. Choosing overly complex models for the training data points can often lead to overfitting. On the other hand, a simpler model leads to underfitting the data. Hence choosing just the right amount of complexity in the model is critical. Since the complexity of the model cannot be directly inferred from the available training data, it is often impossible to stumble upon the right model complexity for training. This is where regularization comes into play making the complex model prone to overfitting.
   Common causes for overfitting are
   - When the model is complex enough that it starts modeling the noise in the training data.
   - When the training data is relatively small and is an insufficient representation of the underlying distribution that it is sampled from, the model fails to learn a generalizable mapping.

4. **What is Gini–impurity index?**

   Answer - Gini Impurity is a measurement used to build Decision Trees to determine how the features of a dataset should split nodes to form the tree. More precisely, the Gini Impurity of a dataset is a number between 0-0.5, which indicates the likelihood of new, random data being misclassified if it were given a random class label according to the class distribution in the dataset.

5. **Are unregularized decision-trees prone to overfitting? If yes, why?**
   Answer - Decision trees are prone to overfitting, especially when a tree is particularly deep. This is due to the amount of specificity we look at leading to smaller sample of events that meet the previous assumptions. This small sample could lead to unsound conclusions

6. **What is an ensemble technique in machine learning?**
   Answer - Ensembling is nothing but the technique to combine several individual predictive models to come up with the final predictive model. And in this article, we're going to look at some of the ensembling techniques for both Classification and Regression problems such as Maximum voting, Averaging, Weighted Averaging, and Rank Averaging.

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

8. **What is out-of-bag error in random forests?**
   Answer - The out-of-bag (OOB) error is the average error for each calculated using predictions from the trees that do not contain in their respective bootstrap sample. This allows the RandomForestClassifier to be fit and validated whilst being trained.

9. **What is K-fold cross-validation?**
   Answer - K-fold Cross-Validation is when the dataset is split into a K number of folds and is used to evaluate the model's ability when given new data. K refers to the number of groups the data sample is split into. For example, if you see that the k-value is 5, we can call this a 5-fold cross-validation

10. **What is hyper parameter tuning in machine learning and why it is done?**
    Answer - Hyperparameter tuning consists of finding a set of optimal hyperparameter values for a learning algorithm while applying this optimized algorithm to any data set. That combination of hyperparameters maximizes the model's performance, minimizing a predefined loss function to produce better results with fewer errors.

11. **What issues can occur if we have a large learning rate in Gradient Descent?**
    Answer - If learning rate is too large, gradient descent can overshoot the minimum. It may fail to converge and even diverge.

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

Answer - Non-linear 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.

13. Differentiate between Adaboost and Gradient Boosting.

Answer-

Loss Function:

The technique of Boosting uses various loss functions. In case of Adaptive Boosting or AdaBoost, it minimises the exponential loss function that can make the algorithm sensitive to the outliers. With Gradient Boosting, any differentiable loss function can be utilised. Gradient Boosting algorithm is more robust to outliers than AdaBoost.

Flexibility

AdaBoost is the first designed boosting algorithm with a particular loss function. On the other hand, Gradient Boosting is a generic algorithm that assists in searching the approximate solutions to the additive modelling problem. This makes Gradient Boosting more flexible than AdaBoost.

Benefits

AdaBoost minimises loss function related to any classification error and is best used with weak learners. The method was mainly designed for binary classification problems and can be utilised to boost the performance of decision trees. Gradient Boosting is used to solve the differentiable loss function problem. The technique can be used for both classification and regression problems.

Shortcomings

In the case of Gradient Boosting, the shortcomings of the existing weak learners can be identified by gradients and with AdaBoost, it can be identified by high-weight data points.

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

Answer - The **bias** is known as the difference between the prediction of the values by the ML model and the correct value. Being high in biasing gives a large error in training as well as testing data. Its 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 **Underfitting of Data**. This happens when the hypothesis is too simple or linear in nature.

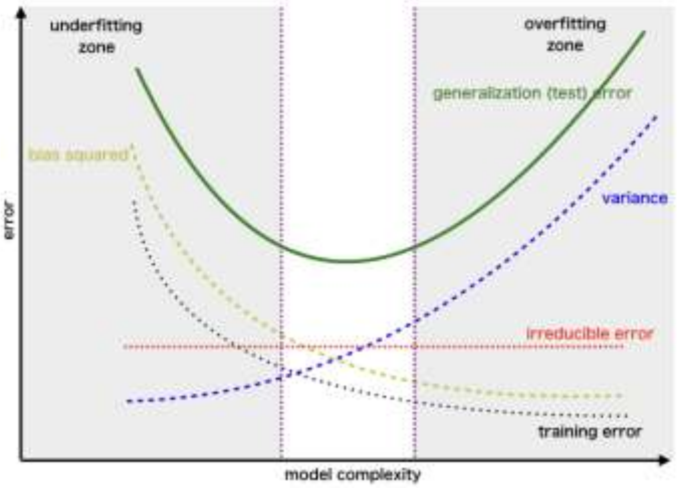
The variability of model prediction for a given data point which tells us spread of our data is called the **variance** of the model. The model with high variance has a very complex fit to the training data and thus is not able to fit accurately on the data which it hasn't seen before. As a result, such models perform very well on training data but has 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 eq.) 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 eq.) 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 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. The best fit will be given by hypothesis on the tradeoff point.

The error to complexity graph to show trade-off is given as –



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

Answer –

- **Linear Kernel -** It is the most basic type of kernel, usually one dimensional in nature. It proves to be the best function when there are lots of features. The linear kernel is mostly preferred for **text-classification problems** as most of these kinds of classification problems can be linearly separated. Linear kernel functions are **faster** than other functions.

  Linear Kernel Formula

  F(x, xj) = sum(x.xj)

  Here, **x, xj** represents the data you're trying to classify.

- **Polynomial Kernel -** It is a more generalized representation of the linear kernel. It **is not** as preferred as other kernel functions as it is **less efficient** and accurate.

  Polynomial Kernel Formula

  F(x, xj) = (x.xj+1)^d

  Here '.' shows the **dot product** of both the values, and **d** denotes the degree.

  F(x, xj) representing the **decision boundary** to separate the given classes.

- **RBF Kernel -** It is one of the most preferred and used kernel functions in svm. It is usually chosen for non-linear data. It helps to make proper separation when there is no prior knowledge of data.

  Gaussian Radial Basis Formula

  F(x, xj) = exp(-gamma * ||x - xj||^2)

The value of gamma varies from **0 to 1**. You have to manually provide the value of gamma in the code. The most preferred value for **gamma is 0.1**.