**Assignment: 05**

**Q1. Explain what is Model Selection? Why we need Model Selection?**

* Model selection is the process of selecting one final machine learning model from among a collection of candidate machine learning models for a training dataset.
* Model selection is a process that can be applied both across different types of models (e.g. logistic regression, SVM, KNN, etc.) and across models of the same type configured with different model Hyperparameters.
* **Model selection is different from model assessment:** The process of evaluating a model’s performance is known as model assessment, whereas the process of selecting the proper level of flexibility for a model is known as model selection.
* **Need Model Selection:** The goal of model selection is to choose a sparse statistical model that adequately explains the data. A good model has three main characteristics: parsimony (model simplicity), Goodness-of-fit test (model fits the data well), and generalizability (model can be used to describe or predict new data). Need of a good model includes factors and covariates to

1. Avoid being underfit (too simple),
2. Avoid being overfit (unnecessarily complex)
3. Account for potential confounding (confusing)

**Q2. Write a short note on:**

**a) Training, Testing and Validation Dataset.**

|  |
| --- |
| https://miro.medium.com/v2/resize:fit:700/1*Nv2NNALuokZEcV6hYEHdGA.png |

* Training, testing, and validation datasets are essential components of machine learning and statistical modeling workflows. They play a crucial role in developing and evaluating models to ensure their accuracy, generalization, and performance. In this note, we will explore the concepts of training, testing, and validation datasets, their purposes, and their significance in the model development process.

1. **Training Dataset:** The training dataset is the foundational component of model development. It is a collection of labeled examples that are used to teach the model the underlying patterns and relationships in the data. The more diverse and representative the training dataset, the better the model can learn and generalize. The training process involves iteratively adjusting the model's parameters to minimize the difference between its predictions and the ground truth labels in the training dataset.

* **Key aspects of the training dataset:**
* **Size:** The training dataset should be large enough to capture the underlying complexity of the problem but manageable in terms of computational resources and time.
* **Diversity:** It should contain a broad range of representative examples, covering various scenarios and variations present in the target problem.
* **Labeling:** Each example in the training dataset must have associated ground truth labels that the model can learn from.

1. **Testing Dataset:** The testing dataset is used to assess the performance of the trained model and estimate its generalization ability. It serves as an unbiased evaluation set, allowing us to gauge how well the model performs on unseen data. The testing dataset should be separate from the training dataset to ensure that the model is not influenced by the testing examples during the training process. The testing dataset should resemble the real-world data that the model will encounter after deployment.

* **Key aspects of the testing dataset:**
* **Independence:** It should be independent of the training dataset to provide an unbiased evaluation of the model's performance.
* **Unseen data:** The testing dataset should consist of examples that the model has not seen during training to assess its generalization capabilities accurately.
* **Real-world similarity:** The testing dataset should reflect the distribution of real-world data to evaluate the model's effectiveness in practical scenarios.

1. **Validation Dataset:** The validation dataset plays a critical role in the model development process, particularly during hyperparameter tuning and model selection. It helps in assessing different models or configurations and enables the selection of the best-performing model. The validation dataset provides an intermediate checkpoint between the training and testing stages, allowing for fine-tuning and optimization.

* **Key aspects of the validation dataset:**
* **Unbiased evaluation:** Like the testing dataset, the validation dataset should be independent of the training dataset to avoid any bias during model evaluation.
* **Performance estimation:** It helps in estimating the model's performance on unseen data, providing insights into its generalization abilities and identifying potential overfitting or underfitting issues.
* **Hyperparameter tuning:** The validation dataset is crucial for tuning hyperparameters, such as learning rates, regularization parameters, or model architectures, to optimize the model's performance.

**b) Train Test Split:**

* Train-test split is a fundamental technique used in machine learning and statistical modeling to assess the performance and generalization capabilities of a model. It involves dividing a dataset into two subsets: a training set and a testing set. In this note, we will delve into the concept of train-test split, its purpose, and best practices for implementing it effectively.

1. **Purpose of Train-Test Split**: The primary purpose of the train-test split is to evaluate how well a model generalizes to unseen data. By training a model on a subset of the data and evaluating its performance on a separate, unseen portion, we can estimate how well the model will perform on new, real-world data. The train-test split allows us to detect potential issues such as overfitting, where the model performs well on the training data but fails to generalize to new examples.
2. **Implementation of Train-Test Split:** The train-test split is typically performed by randomly partitioning the dataset into two subsets: a training set and a testing set. The training set, which is usually larger, is used to train the model, while the testing set is used for evaluation. The partitioning ratio may vary depending on the dataset size and the specific problem, but a common split is 70-30 or 80-20, where 70% or 80% of the data is allocated to the training set, and the remaining 30% or 20% is assigned to the testing set.
3. **Best Practices for Train-Test Split:** To ensure the train-test split is performed effectively, the following best practices should be considered:
   1. **Randomness:** The dataset should be randomized before the split to prevent any potential biases caused by the order or arrangement of the data. Randomization helps ensure that the training and testing sets are representative of the overall data distribution.
   2. **Data Distribution:** The train-test split should preserve the original data distribution, particularly if the dataset has imbalanced classes or specific patterns. Stratified sampling can be employed to maintain the proportion of each class or pattern in both the training and testing sets.
   3. **Single Split:** It is crucial to perform the train-test split only once and refrain from iteratively modifying the split based on model performance. Multiple iterations of the split may lead to overfitting to the testing set and overestimation of the model's performance.
   4. **Size of the Testing Set:** The size of the testing set should be sufficient to provide a reliable evaluation of the model's performance. A larger testing set can yield more robust performance estimates, but it may reduce the amount of data available for training the model. Striking a balance between the training and testing set sizes is important.
   5. **Temporal Considerations:** If the data has a temporal or sequential nature, it is essential to preserve the temporal order during the train-test split. This means that the testing set should consist of examples that come after the training set chronologically. This ensures that the model is evaluated on data that simulates real-world scenarios where future predictions are made based on past observations.
   6. **Cross-Validation:** In situations where the dataset is limited, and obtaining reliable performance estimates is critical, cross-validation techniques such as k-fold cross-validation can be employed. Cross-validation involves dividing the dataset into multiple folds and performing multiple train-test splits, rotating the role of the training and testing sets. This enables a more robust evaluation of the model's performance.
4. **Train-Validation-Test Split:** In some cases, a three-way split, known as the train-validation-test split, is employed. The validation set is introduced between the training and testing sets. The training set is used to train the model, the validation set is used for hyperparameter tuning and model selection, and the testing set remains independent for final model evaluation. This allows for a more detailed evaluation of the model's performance and helps prevent overfitting to the testing set. The train-validation-test split follows these principles:

* **Training Set:** The largest portion of the dataset is allocated to the training set. It is used to train the model's parameters and learn the underlying patterns in the data.
* **Validation Set:** The validation set is used to fine-tune the model's hyperparameters, such as learning rate, regularization parameters, or model architectures. By evaluating different configurations on the validation set, the model with the best performance can be selected.
* **Testing Set:** The testing set remains independent and serves as the final evaluation set. It is used to assess the performance of the selected model, providing an unbiased estimate of how well the model will generalize to new, unseen data.

It is important to note that the validation set is not used for model training. Its purpose is solely to assist in hyperparameter tuning and model selection. Once the model has been finalized based on the validation set's performance, it is evaluated on the testing set to obtain the final performance metrics. By employing a train-validation-test split, the model development process becomes more robust and reliable. It helps to ensure that the model's performance estimates are realistic and unbiased, as the testing set remains unseen until the final evaluation. Additionally, the validation set provides an opportunity to detect and mitigate overfitting issues, as the model is evaluated on data that it has not been directly trained on.

**Reference**: <https://algotrading101.com/learn/train-test-split/>

**Q3. Describe what is cross validation, draw suitable diagram and explain k-Fold Cross Validation.**

* **Describe what is cross validation:** Cross validation is a technique used in machine learning to evaluate the performance of a model on unseen data. It involves dividing the available data into multiple folds or subsets, using one of these folds as a validation set, and training the model on the remaining folds. This process is repeated multiple times, each time using a different fold as the validation set. Finally, the results from each validation step are averaged to produce a more robust estimate of the model’s performance. The main purpose of cross validation is to prevent overfitting, which occurs when a model is trained too well on the training data and performs poorly on new, unseen data. By evaluating the model on multiple validation sets, cross validation provides a more realistic estimate of the model’s generalization performance, i.e., its ability to perform well on new, unseen data. There are several types of cross validation techniques, including k-fold cross validation, leave-one-out cross validation, and stratified cross validation.
* **k-Fold Cross Validation:** k-Fold Cross-Validation is a resampling technique commonly used in machine learning and statistical modeling to evaluate the performance of a model and estimate its generalization capability. It involves dividing the available dataset into k subsets or folds, performing model training and evaluation k times, and then aggregating the results to obtain an overall performance estimate.

|  |
| --- |
| K-Fold Cross Validation |

* **Here's a step-by-step explanation of the k-Fold Cross-Validation process:**
* **Dataset Partitioning:** The original dataset is divided into k non-overlapping folds of approximately equal size. Each fold represents a subset of the data.
* **Model Training and Evaluation:** The process then iterates k times. In each iteration, one fold is designated as the testing set, while the remaining k-1 folds are used as the training set. The model is trained on the training set and evaluated on the testing set.
* **Performance Metric Calculation:** The performance of the model is assessed using a chosen evaluation metric, such as accuracy, precision, recall, mean squared error, or others, depending on the specific task. The evaluation metric is calculated based on the model's predictions on the testing set and the corresponding true values.
* **Iteration:** Steps 2 and 3 are repeated k times, with each fold serving as the testing set once. This ensures that every data point is used for both training and testing at some point during the cross-validation process.
* **Aggregation of Results:** The performance metrics obtained from each iteration (e.g., accuracy scores) are aggregated to calculate an overall performance estimate. Common approaches include taking the average, weighted average, or other statistical measures of the metrics across the iterations.

**Q4. Define what is Boosting? Why we need Boosting?**

* Boosting is a machine learning ensemble technique that combines multiple weak or base models to create a strong predictive model. The basic idea behind boosting is to iteratively train weak models, where each subsequent model focuses on correcting the mistakes made by the previous models. The final prediction is obtained by aggregating the predictions of all the weak models, giving more weight to the models that perform better on the training data.
* **The main objectives of boosting are:**
* **Improved Predictive Performance:** Boosting aims to improve the predictive accuracy of a model by combining multiple weak models into a stronger and more accurate ensemble. Weak models, such as decision trees with limited depth or small number of features, are often used as base models. The ensemble approach of boosting helps to reduce bias and variance, leading to improved generalization and prediction capabilities.
* **Handling Complex and Nonlinear Relationships:** Boosting can effectively handle complex and nonlinear relationships in the data. By combining multiple weak models, boosting can capture intricate patterns, interactions, and dependencies that might be missed by a single model. This allows boosting to handle a wide range of machine learning tasks, such as classification, regression, and ranking problems.
* **Robustness to Overfitting:** Boosting incorporates a sequential learning process where each subsequent weak model is trained to focus on the examples that the previous models struggled to classify correctly. This adaptive learning helps to reduce overfitting and makes the ensemble more robust by minimizing the errors and biases introduced by the individual models. The boosting algorithm learns to assign higher weights to misclassified examples, forcing subsequent models to prioritize those examples and improve their predictions.
* **Feature Importance and Selection:** Boosting algorithms provide a measure of feature importance, indicating the relative importance of each feature in making predictions. By examining feature importance scores, one can gain insights into the relevance and contribution of different features in the predictive task. This information can be valuable for feature selection, dimensionality reduction, and understanding the underlying factors driving the predictions.
* Some popular boosting algorithms include AdaBoost (Adaptive Boosting), Gradient Boosting, and XGBoost (Extreme Gradient Boosting). These algorithms differ in their specific methodologies and optimizations, but they share the common principle of iteratively improving the ensemble's performance.

**Q5. Describe Boosting Algorithm along with its types.**

* Boosting is an ensemble learning method that combines a set of weak learners into a strong learner to minimize training errors. In boosting, a random sample of data is selected, fitted with a model and then trained sequentially With each iteration, the weak rules from each individual classifier are combined to form one, strong prediction rule. The general idea of most boosting methods is to train predictors sequentially, each trying to correct its predecessor.
* **Three popular types of boosting algorithms:** Adaptive Boosting (AdaBoost), Gradient Boosting, and Extreme Gradient Boosting (XGBoost).

1. **Adaptive Boosting (AdaBoost):** AdaBoost is one of the earliest and most well-known boosting algorithms. It works by iteratively training a series of weak models on the data, where each subsequent model focuses on the examples that were misclassified by the previous models. The algorithm assigns higher weights to the misclassified examples, allowing subsequent models to prioritize them and improve their predictions. **The steps involved in the AdaBoost algorithm are as follows:**

* Initialize the weights of the training examples uniformly.
* Train a weak model on the training data, and compute its error rate.
* Adjust the weights of the training examples based on their misclassification rate. Increase the weights of the misclassified examples to emphasize their importance in subsequent iterations.
* Repeat steps b and c for a predefined number of iterations or until a stopping criterion is met.
* Combine the predictions of all the weak models using weighted majority voting to obtain the final prediction.

AdaBoost focuses on correcting the mistakes of the previous models by assigning more weight to the misclassified examples. This iterative learning process improves the overall performance of the ensemble.

1. **Gradient Boosting:** Gradient Boosting is a boosting algorithm that constructs an ensemble of weak models in a sequential manner. Unlike AdaBoost, Gradient Boosting builds subsequent models by minimizing the errors using the gradient descent optimization algorithm. Each weak model is trained to predict the negative gradient of the loss function associated with the ensemble's current predictions. The steps involved in the Gradient Boosting algorithm are as follows:

* Initialize the ensemble by fitting an initial model to the training data.
* Compute the residuals or negative gradients of the loss function with respect to the current ensemble's predictions.
* Train a weak model to predict the residuals, aiming to minimize the residuals using a base learner.
* Add the weak model to the ensemble by updating the predictions of the ensemble with the predictions of the newly trained model.
* Repeat steps b-d for a predefined number of iterations or until a stopping criterion is met.

Gradient Boosting focuses on reducing the residual errors at each iteration, gradually improving the ensemble's predictions.

1. **Extreme Gradient Boosting (XGBoost):** Extreme Gradient Boosting, or XGBoost, is an advanced and highly optimized implementation of gradient boosting. It improves upon traditional Gradient Boosting by incorporating additional regularization techniques, parallel processing, and advanced algorithms to enhance performance and accuracy. XGBoost includes several key features:

* **Regularization:** XGBoost supports regularization techniques, such as L1 and L2 regularization, to control model complexity and prevent overfitting.
* **Tree Pruning:** XGBoost uses a technique called tree pruning to remove or collapse nodes in the decision trees that do not contribute significantly to improving the performance.
* **Parallel Processing:** XGBoost utilizes parallel processing and distributed computing to speed up training, making it highly scalable and efficient.
* **Handling Missing Values:** XGBoost has built-in capabilities to handle missing values in the data during the training process.
* **Cross-Validation:** XGBoost supports cross-validation techniques to evaluate the model's performance and fine-tune hyperparameters.

XGBoost has gained popularity due to its exceptional performance in various machine learning competitions and real-world applications.

* **Reference**: <https://www.analyticsvidhya.com/blog/2022/02/k-fold-cross-validation-technique-and-its-essentials/>

**Q6. With suitable diagram and necessary equations explain Ada Boost Algorithm.**

|  |
| --- |
| Image result for boosting |

* AdaBoost is the first stepping stone in the world of Boosting. AdaBoost (Adaptive Boosting)-a statistical classification meta-algorithm formulated by Yoav Freund and Robert Schapire in 1995 Many other types of learning algorithms called weak learners are used to improve performance that yield strong learner. The output of the other learning algorithms (weak learners) is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost algorithms can be used for both classification and regression problem.

|  |
| --- |
| https://miro.medium.com/v2/resize:fit:700/1*131sBdM2wNqbj5IBp_bFCw.jpeg |

* **Example:** Let us assume we have to predict whether someone is going to buy a house or not using Machine Learning algorithms. Instead of creating one decision tree or logistic regression model to predict, what we do is create multiple weak learner i.e. Decision Stumps or poor logistic regression model or tree with depth 1 or 2, one after the other to predict the output. In AdaBoost we rig the system in such a way that features that are predicted poorly in previous learner are predicted with better results in the subsequent model.
* Let us give an example input to the above Decision stumps.https://miro.medium.com/v2/resize:fit:495/1*-jD-SGLBbGSAb17ne_TgaQ.gif

Input to the stumps

* The *Numbedroom* in the above input is ‘*2*’, this will go to the primary stump and yield prediction ‘*+1*’. The *sqft*will go to the second stump and will yield a prediction of ‘*+1*’, *Lawn*will get a prediction of ‘*-1*’ and similarly *Area* will get a prediction of ‘*+1*’ . So the final question that we need to ask is , how do we combine all these predictions from various weak learners and give one final prediction to the above data point?. This is done with the formula below

https://miro.medium.com/v2/resize:fit:386/1*24N7Pv8Goe7VVFx24Fy39w.gif

**weighted voting scheme for 1st data point**

https://miro.medium.com/v2/resize:fit:215/1*KzTQRnOLvrAGvHiQJF2EFA.gif

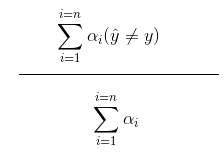
**Predictions from each scheme i.e. +1 or -1**

* We basically take the final sign after we multiply a weight associated with each stump is multiplied with the prediction and added with the rest of the values from each stump.

https://miro.medium.com/v2/resize:fit:155/1*e2oCyMEht7dVhGntT_Z_Ew.gif**Equation of final prediction. L number of stumps.**

* **The way AdaBoost works is as follows:**
  + Primarily each data point is initialized with a weight ‘*alpha*’ that is equal to (1/number of data points).
  + Then for each weak learner or model, we iterate to calculate the predicted value, then go onto compute the weight ‘*w*’(weighted error) and based on that re-adjust the ‘*alpha*’ term for next iteration until the very end.
  + After all the iterations are done we calculate the prediction based on the above equation.
* There are two fundamental questions with respect to AdaBoost that we need to ask here, How are we going to update the weights ‘*w*’ and how are we going to update the ‘*alpha*’ values.? In this algorithm what we do is add all the weights of data points that have been miss-classified and divide it by the weight of all data points i.e.

**https://miro.medium.com/v2/resize:fit:101/1*I-eJUBiTTOf0BcPHoiNOQA.gifSumming over weights of misclassified points**

**Weighted** error formula

* Following the creation of weighted error, we ensure that we update the weight associated with each stump , whose formula is https://miro.medium.com/v2/resize:fit:424/1*xies2dpxdSsAV6SMzQk2aA.gif

**(PPT Give you more understanding than PDF)** The material for the PDF has been compiled from various sources such as books, tutorials (offline and online), lecture notes, several resources available on Internet. The information contained in this PDF is for general information and education purpose only. While we endeavor to keep the information up to date and correct, **we make no representation of any kind about the completeness and accuracy of the material.** The information shared through this PDF material should be used for educational purpose only.

**Thank you…**