Statistics MACHINE LEARNING ASSIGNMENT 5

Answer PDf 1

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

Answer. **Expected**

Chisquare is used to analyse

Answer. **Frequencies**

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

Answer. **6**

Which of these distributions is used for goodness of fit testing?

Answer. **Chi-square distribution**

5. Which of the following distributions is Continuous.?

Answer. **Distribution**

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

Answer **Hypothesis**

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

Answer. **Null Hypothesis**

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

Answer. **Two-tailed**

9. An Alternative Hypothesis is also called an?

Answer. **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-----------

Answer. **np**

Statistics MACHINE LEARNING ASSIGNMENT 5

Answer PDF 2

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

Answer. Both R-squared (R^2) and Residual Sum of Squares (RSS) are important measures of goodness of fit in regression analysis, but they serve different purposes and provide complementary information about the model's performance.

R-squared (R^2)

R-squared measures the proportion of variance in the dependent variable explained by the independent variables in the regression model. It ranges from 0 to 1, with higher values indicating a better fit.

- R-squared is a relative measure, comparing the model's fit to a baseline model that predicts the mean of the dependent variable.

- It provides insight into how well the independent variables collectively predict the variation in the dependent variable.

- However, R-squared tends to increase with the addition of more predictors, even if those predictors do not add much explanatory power to the model.

Residual Sum of Squares (RSS)

- RSS measures the total squared difference between the observed values of the dependent variable and the values predicted by the regression model.

- It represents the amount of unexplained variation or error in the data. Lower values of RSS indicate a better fit, as they suggest that the model's predictions are closer to the observed values.

- RSS is an absolute measure of model fit and directly reflects the magnitude of the residuals or prediction errors.

- Unlike R-squared, RSS does not inherently increase with the addition of more predictors, as it directly reflects the discrepancy between observed and predicted values.

Statistics MACHINE LEARNING ASSIGNMENT 5

Maily R-squared and RSS are valuable measures of model fit, but they capture different aspects of model performance. R-squared provide an overall assessment of the explanatory power of the model, while RSS measures the accuracy of individual predictions. It's often useful to consider both measures together when evaluating the performance of a regression model.

Question-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 regression analysis Total Sum of Squares (TSS), Explained Sum of Squares (ESS), and Residual Sum of Squares (RSS) are important metrics used to assess the goodness of fit of the regression model. These metrics are related to each other through the following equation:

\[TSS = ESS + RSS \]

Here's what each of these metrics represents:

Total Sum of Squares (TSS) -

- TSS represents the total variance in the dependent variable (Y).
- It measures the total variability in the observed values of the dependent variable without considering the regression model.
- Mathematically, TSS is calculated as the sum of squared differences between each observed value of Y and the overall mean of Y.

\[TSS = \sum_{i=1}^{n} (Y_i - \bar{Y})^2 \]

Explained Sum of Squares (ESS)

- ESS represents the variance in the dependent variable that is explained by the regression model.
- It measures the variability in the dependent variable that is accounted for by the independent variables in the model.
- Mathematically, ESS is calculated as the sum of squared differences between the predicted values of Y (based on the regression model) and the overall mean of Y.

\[ESS = \sum_{i=1}^{n} (\hat{Y}_i - \bar{Y})^2 \]

Residual Sum of Squares (RSS)

- RSS represents the unexplained variance or error in the dependent variable that is not accounted for by the regression model.

- It measures the variability in the dependent variable that is not explained by the independent variables in the model.

- Mathematically, RSS is calculated as the sum of squared differences between the observed values of Y and the predicted values of Y based on the regression model.

\[RSS = \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \]

The equation relating these three metrics (\(TSS = ESS + RSS \)) indicates that the total variance in the dependent variable (TSS) can be decomposed into the variance explained by the regression model (ESS) and the unexplained variance or error (RSS). This equation is a fundamental concept in regression analysis and is often used to assess the proportion of variance explained by the model (R-squared).

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

Answer - Regularization is a technique used in machine learning to prevent overfitting and improve the generalization performance of models. Overfitting occurs when a model learns the training data too well, capturing noise or irrelevant patterns that do not generalize well to unseen data. Regularization addresses this issue by adding a penalty term to the model's objective function, encouraging simpler models that are less prone to overfitting. Here are some key reasons for using regularization in machine learning:

Preventing Overfitting-

Regularization helps prevent overfitting by discouraging overly complex models that fit the training data too closely. By penalizing large coefficients or complex model structures, regularization encourages models to prioritize simpler explanations that are more likely to generalize well to new data.

Statistics MACHINE LEARNING ASSIGNMENT 5

Improving Generalization-

Regularization techniques improve the generalization performance of models by reducing variance and increasing bias. By finding the right balance between bias and variance, regularization helps create models that perform well on both training and unseen data, leading to better overall performance.

Handling Multicollinearity-

In regression analysis, multicollinearity occurs when predictor variables are highly correlated with each other. Regularization techniques like Ridge regression and ElasticNet can help mitigate multicollinearity by shrinking the coefficients of correlated predictors, making the model more stable and interpretable.

Feature Selection-

Some regularization techniques automatically perform feature selection by shrinking the coefficients of less important predictors to zero. This can help simplify the model and improve its interpretability by focusing on the most relevant features.

Dealing with Noisy Data-

Regularization can help models generalize better to noisy data by reducing the impact of noise on the model's parameters. By penalizing overly complex models that fit the noise in the training data, regularization encourages models to focus on the underlying patterns that are more likely to be true across different datasets.

regularization is a critical tool in machine learning for building models that generalize well to new data, improve interpretability, and handle various challenges such as overfitting, multicollinearity, and noisy data.

Question .4 What is Gini–impurity index?

Answer - The Gini impurity index is a measure of impurity or disorder used in decision tree algorithms for classification tasks. It quantifies the probability of incorrectly classifying a randomly chosen element if it were labeled randomly according to the distribution of class labels in a subset.

In a given dataset, suppose we have a set of classes \(C\) and \(p_i\) represents the probability of randomly selecting an element of class \(i\). The Gini impurity \(Gini(C)\) is calculated as follows:

\[Gini(C) = 1 - \sum_{i=1}^{n} (p_i)^2 \]

where \(n\) is the number of classes.

The Gini impurity index ranges from 0 to 1, where:

- A Gini impurity of 0 indicates that the set contains only one class (pure node).
- A Gini impurity of 1 indicates that the set contains an equal proportion of elements from different classes (maximum impurity or maximum disorder).

In the context of decision trees, the Gini impurity is used to determine the best split at each node. When building a decision tree, the algorithm aims to minimize the Gini impurity of the resulting child nodes after a split. A split that results in child nodes with lower Gini impurity is considered better as it leads to more homogeneous subsets.

Another word , the Gini impurity index is a measure of impurity or disorder commonly used in decision tree algorithms to evaluate the quality of splits and make decisions about node splitting during the tree-building process.

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

Answer – The unregularized decision trees are prone to overfitting. Overfitting occurs when a model learns the training data too well, capturing noise or random fluctuations that are specific to the training set and do not generalize well to unseen data. Decision trees have a tendency to create complex, deep trees that can perfectly fit the training data, leading to overfitting. Here's why unregularized decision trees are prone to overfitting:

High Variance-
Decision trees have high variance, meaning they are sensitive to small fluctuations in the training data. Without any constraints on the tree's structure or size, the algorithm may grow a large, deep tree that captures noise or outliers in the training data, resulting in poor generalization to new data.

Memorization of Training Data:-

Decision trees can memorize the training data by creating branches and nodes that perfectly classify each data point in the training set. This memorization can lead to a lack of generalization ability, as the model fails to capture the underlying patterns in the data and instead learns to reproduce the training set exactly.

Lack of Smoothing:-

Unregularized decision trees do not have mechanisms for smoothing or generalizing the learned patterns. As a result, they may capture too much detail from the training data, including noise and irrelevant features, leading to poor performance on unseen data.

Limited Pruning:-

While some decision tree algorithms support pruning to reduce the size of the tree and improve generalization, unregularized decision trees may not perform sufficient pruning. This can result in overly complex trees with many branches and nodes, which are more likely to overfit the training data.

in decision trees, regularization techniques such as pruning, limiting the maximum depth of the tree, or restricting the number of features considered at each split can be applied. These techniques help prevent the tree from becoming too complex and capturing noise in the training data, leading to better generalization performance on unseen data.

Question - What is an ensemble technique in machine learning?

Answer - Ensemble techniques in machine learning involve combining multiple models to produce a more robust and accurate predictive model than any individual model in the ensemble. The idea is to leverage the diversity of different models and combine their predictions in a way that reduces errors and improves overall performance.

Statistics MACHINE LEARNING ASSIGNMENT 5

Ensemble techniques can be broadly categorized into two main types:

Sequential Ensemble Methods (Boosting)

In boosting, base learners are trained sequentially, with each subsequent model focusing more on the instances that were misclassified by earlier models. This sequential training process aims to correct the errors made by previous models and improve overall performance. Common boosting algorithms include AdaBoost, Gradient Boosting Machines (GBM), and XGBoost.

Parallel Ensemble Methods (Bagging and Random Forests):-

In bagging (Bootstrap Aggregating), multiple instances of a base learner are trained independently on different subsets of the training data (usually sampled with replacement). The final prediction is typically made by averaging (for regression) or taking a majority vote (for classification) of the predictions from all the models. Random Forest is a popular ensemble method based on bagging, where decision trees are the base learners.

Ensemble techniques offer several advantages, including:-

- Improved generalization: Ensembles can reduce overfitting and improve generalization by combining the strengths of different models and reducing the impact of individual model weaknesses.

- Increased robustness: Ensembles are less sensitive to noise and outliers in the data compared to individual models.

- Enhanced predictive accuracy: Ensembles often achieve higher predictive accuracy than single models, especially when the individual models in the ensemble are diverse.

Overall, ensemble techniques are powerful tools in machine learning for improving model performance and building more accurate and reliable predictive models.

Question – What is the difference between Bagging and Boosting techniques?

Statistics MACHINE LEARNING ASSIGNMENT 5

Answer … Bagging and boosting are both ensemble techniques used in machine learning to improve the performance of predictive models, but they differ in their approach to combining multiple models and training them.

Bagging (Bootstrap Aggregating) :-

- Bagging involves training multiple instances of the same base learner on different subsets of the training data, sampled with replacement (bootstrap samples).

- Each model in the ensemble is trained independently and has an equal weight in the final prediction.

- The final prediction is typically made by averaging (for regression) or taking a majority vote (for classification) of the predictions from all the models.

- Bagging helps reduce variance and overfitting by creating diverse models that learn different aspects of the data.

Boosting:

- Boosting is an iterative ensemble technique that sequentially trains a series of weak learners (models that perform slightly better than random guessing).

- Each subsequent model in the sequence focuses more on the instances that were misclassified by earlier models.

- The final prediction is typically made by combining the predictions of all the models, often weighted by their individual performance.

- Boosting aims to correct the errors made by previous models and improve overall performance by iteratively focusing on difficult-to-classify examples.

main differences between bagging and boosting are:

- Bagging trains multiple models independently in parallel, while boosting trains a sequence of models sequentially.

- In bagging, each model in the ensemble has equal weight in the final prediction, while boosting assigns different weights to each model based on its performance.

- Bagging aims to reduce variance and overfitting by creating diverse models, while boosting focuses on reducing bias and improving accuracy by iteratively correcting errors made by earlier models.

Question – What is out-of-bag error in random forests?

Answer .- In Random Forests, the out-of-bag (OOB) error is an estimate of the model's prediction error calculated using the samples that were not included in the bootstrap sample used to train each individual decision tree.

Bootstrap Sampling:-

1. When building a Random Forest, multiple decision trees are trained. Each tree is trained on a bootstrap sample of the original dataset, which is created by sampling with replacement. This means that some data points may be selected multiple times, while others may not be selected at all.

Out-of-Bag Samples:-

For each tree in the Random Forest, there are data points that were not included in its bootstrap sample. These data points are referred to as out-of-bag samples.

Estimating Error:-

2. The OOB error is calculated by making predictions on the out-of-bag samples using the corresponding tree in the ensemble. Since these samples were not used in training the tree, they serve as a validation set to estimate the model's prediction error.

Aggregating Results:-

The OOB error for the entire Random Forest model is computed by aggregating the errors from all individual trees in the ensemble.

The OOB error provides an estimate of how well the Random Forest model will generalize to unseen data without the need for a separate validation set. It is a useful metric for evaluating the model's performance and can help in model selection and parameter tuning.

Statistics MACHINE LEARNING ASSIGNMENT 5

Question - What is K-fold cross-validation?

Answer .- K-fold cross-validation is a technique used to assess the performance and robustness of a machine learning model. It involves splitting the dataset into K subsets, or folds, of approximately equal size. The model is then trained and evaluated K times, each time using a different fold as the validation set and the remaining K-1 folds as the training set.

Dataset Splitting-:

The original dataset is randomly partitioned into K equally sized folds.

Model Training and Evaluation:

The model is trained K times, with each fold used once as the validation set while the remaining K-1 folds are used as the training set. For each iteration, the model is trained on the training set and evaluated on the validation set.

Performance Metric Calculation:-

After each iteration, a performance metric (such as accuracy, precision, recall, or F1-score) is computed based on the model's predictions on the validation set.

Aggregating Results:-

The performance metrics from all K iterations are averaged to obtain an overall performance estimate of the model. This average performance metric is typically used to assess the model's performance and compare it with other models or parameter settings.

K-fold cross-validation provides several advantages:

- It provides a more reliable estimate of the model's performance compared to a single train-test split, as it uses multiple train-test splits and averages the results.

- It reduces the variance of the performance estimate by using multiple validation sets.

- It ensures that each data point is used for both training and validation, thus maximizing the use of available data.

Statistics MACHINE LEARNING ASSIGNMENT 5

Common choices for the value of K include 5-fold and 10-fold cross-validation, but other values can be used depending on the size of the dataset and computational constraints. Overall, K-fold cross-validation is a valuable technique for evaluating and comparing machine learning models in a robust and reliable manner.

Question - What is hyper parameter tuning in machine learning and why it is done?

ANSWER - Hyperparameter tuning in machine learning refers to the process of selecting the optimal values for the hyperparameters of a machine learning algorithm. Hyperparameters are configuration settings that are not learned from the data but are set before the learning process begins. These settings can significantly impact the performance and behavior of the model.

Examples of hyperparameters include:

- Learning rate in gradient descent algorithms
- Number of hidden layers and neurons in a neural network
- Depth and minimum samples per leaf in decision trees
- Regularization parameters in linear models
- Kernel type and gamma in support vector machines

Hyperparameter tuning is essential for several reasons:

Optimizing Model Performance-

The choice of hyperparameters can have a significant impact on the performance of the model. Tuning hyperparameters allows us to find the combination that results in the best performance metrics, such as accuracy, precision, recall, or F1-score.

Preventing Overfitting:-

Hyperparameter tuning can help prevent overfitting, where the model performs well on the training data but fails to generalize to unseen data. By optimizing hyperparameters, we can find the settings that lead to a model with better generalization performance.

Statistics MACHINE LEARNING ASSIGNMENT 5

Improving Robustness:-

Tuning hyperparameters can lead to a more robust and stable model that performs consistently across different datasets or subsets of the data.

Reducing Computational Resources:-

Hyperparameter tuning helps identify the optimal settings for the model, potentially reducing the computational resources required for training and inference. It allows us to avoid unnecessary computational costs associated with training models with suboptimal hyperparameter settings.

Hyperparameter tuning is typically performed using techniques such as grid search, random search, Bayesian optimization, or evolutionary algorithms. These methods systematically explore the hyperparameter space to find the combination that maximizes the performance of the model on a validation set or through cross-validation. Overall, hyperparameter tuning is a critical step in the machine learning pipeline for building models that achieve the best possible performance on the task at hand.

Question - What issues can occur if we have a large learning rate in Gradient Descent?

Answer - If we use a large learning rate in gradient descent, several issues can occur, which can hinder the optimization process and the convergence of the algorithm. Here are some common issues associated with using a large learning rate:

Overshooting the Minimum:-

A large learning rate can cause the algorithm to take excessively large steps in the direction of the gradient. This can lead to overshooting the minimum of the loss function and oscillating around the minimum rather than converging to it.

Instability and Divergence:-

With a large learning rate, the optimization process can become unstable, leading to divergent behavior. Instead of converging to a minimum, the algorithm may bounce around erratically or even diverge to infinity.

Statistics MACHINE LEARNING ASSIGNMENT 5

Skipping the Minimum :-

In extreme cases, a large learning rate can cause the algorithm to skip over the minimum altogether, especially if the loss function has steep and narrow valleys. This results in suboptimal or even invalid solutions.

Poor Generalization:-.

Using a large learning rate can result in poor generalization performance, as the model may converge to a solution that is specific to the training data and does not generalize well to unseen data.

Slow Convergence:-

Paradoxically, while a large learning rate can cause the optimization process to diverge, it can also slow down the convergence process if the steps are too large. This is because the algorithm may oscillate around the minimum or take inefficient steps back and forth.

To address these issues, it's important to choose an appropriate learning rate for gradient descent. This often involves tuning the learning rate hyperparameter through techniques like grid search, random search, or adaptive learning rate methods such as Adam or RMSprop. Additionally, techniques like learning rate schedules, where the learning rate decreases over time, can also help stabilize the optimization process and improve convergence.

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

Answer - Logistic Regression is a linear classification algorithm that models the relationship between the independent variables and the probability of belonging to a certain class. While Logistic Regression is effective for linearly separable data, it may not perform well for classification tasks involving non-linear data. Here's why:

Linear Decision Boundary:-

Logistic Regression assumes a linear decision boundary between classes. This means that it can only separate classes using a straight line (or hyperplane in higher dimensions). If the decision boundary between classes is non-linear, Logistic Regression will struggle to accurately classify the data.

Underfitting:-

When Logistic Regression is applied to non-linear data, it may result in underfitting, where the model is too simplistic to capture the underlying patterns in the data. In such cases, the model may fail to adequately separate the classes and achieve poor performance on both training and test data.

Limited Expressiveness:-

Logistic Regression has limited expressiveness compared to non-linear models such as Decision Trees, Support Vector Machines with non-linear kernels, or Neural Networks. These models are better suited for capturing complex relationships in non-linear data by learning more flexible decision boundaries.

While Logistic Regression may not be suitable for directly classifying non-linear data, there are ways to make it more flexible or to preprocess the data to make it more amenable to linear methods:

Feature Engineering :-

Transforming the input features or adding polynomial features can sometimes make the data more linearly separable, allowing Logistic Regression to perform better.

Kernel Tricks :- In some cases, Logistic Regression can be combined with kernel methods to implicitly map the input features into a higher-dimensional space where the data becomes linearly separable. However, this approach essentially transforms Logistic Regression into a non-linear model similar to Support Vector Machines with non-linear kernels.

Overall, while Logistic Regression is a powerful and interpretable linear classification algorithm, it may not be the best choice for handling non-linear data without appropriate preprocessing or modifications. In such cases, non-linear classification algorithms are typically more effective.

Question - Differentiate between Adaboost and Gradient Boosting.

Statistics MACHINE LEARNING ASSIGNMENT 5

Answer -: AdaBoost (Adaptive Boosting) and Gradient Boosting are both ensemble learning techniques used for classification and regression tasks. While they share some similarities, they also have key differences in their algorithms and how they build the ensemble of models.

Here's a comparison between AdaBoost and Gradient Boosting:

Base Learners

- AdaBoost: AdaBoost trains a sequence of weak learners (e.g., decision trees with only a few nodes) sequentially. Each subsequent weak learner focuses more on the instances that were misclassified by earlier models.

- Gradient Boosting: Gradient Boosting also trains a sequence of weak learners, typically decision trees. However, each tree is trained to correct the errors made by the previous models. Instead of focusing on misclassified instances, Gradient Boosting minimizes the loss function directly by adjusting the predictions of the ensemble.

Training Process

- AdaBoost: In AdaBoost, each weak learner is trained using a weighted version of the training data, where the weights are adjusted at each iteration based on the performance of the previous models. Misclassified instances are given higher weights to focus on them during subsequent iterations.

- Gradient Boosting: In Gradient Boosting, each weak learner is trained on the residuals (the differences between the predictions of the ensemble and the true labels) of the previous models. The next weak learner is then trained to predict the residuals, effectively reducing the error made by the ensemble.

Loss Function

- AdaBoost: AdaBoost typically uses the exponential loss function, which penalizes misclassifications exponentially. It focuses on improving classification accuracy by adjusting the weights of misclassified instances.

- Gradient Boosting: Gradient Boosting can use various loss functions depending on the task, such as squared loss for regression or logistic loss for classification. It directly optimizes the loss function by minimizing the residuals of the ensemble.

Learning Rate

- AdaBoost: AdaBoost introduces a learning rate parameter that controls the contribution of each weak learner to the final ensemble. A smaller learning rate makes the algorithm more robust but requires more iterations to converge.

- Gradient Boosting: Gradient Boosting also uses a learning rate parameter, but it affects the step size of each weak learner in the direction of the gradient. A smaller learning rate results in slower but more stable convergence.

Overall, while both AdaBoost and Gradient Boosting are powerful ensemble techniques, they have different approaches to building the ensemble and updating the model parameters. Gradient Boosting, in particular, has become popular due to its flexibility, robustness, and ability to handle complex datasets and loss functions.

Question :- What is bias-variance trade off in machine learning ?

Answer :- The bias-variance trade-off is a fundamental concept in machine learning that describes the trade-off between the bias of a model and its variance. It refers to the balance between the model's ability to capture the underlying patterns in the data (bias) and its sensitivity to fluctuations or noise in the training data (variance).

breakdown of the bias-variance trade-off:

Bias

- Bias measures the error introduced by approximating a real-world problem with a simplified model. A high bias model is overly simplistic and fails to capture the true relationships between features and target variable. It tends to underfit the data, resulting in poor performance on both the training and test datasets.

Statistics MACHINE LEARNING ASSIGNMENT 5

Variance

- Variance measures the model's sensitivity to fluctuations in the training data. A high variance model captures the noise or random fluctuations in the training data and may fail to generalize to unseen data. It tends to overfit the training data, performing well on the training dataset but poorly on the test dataset.

The bias-variance trade-off arises because reducing bias typically increases variance and vice versa. Here's how it works:

High Bias, Low Variance

When a model has high bias and low variance, it means that the model is too simple and is unable to capture the underlying patterns in the data. This leads to underfitting, where the model performs poorly on both the training and test datasets.

Low Bias, High Variance :

Conversely, when a model has low bias and high variance, it means that the model is complex and can capture the underlying patterns in the data well. However, it is sensitive to fluctuations in the training data and may capture noise or random fluctuations, leading to overfitting. As a result, the model performs well on the training dataset but poorly on the test dataset.

The goal in machine learning is to find the right balance between bias and variance to minimize the overall error of the model. This often involves selecting an appropriate model complexity, regularization techniques, and hyperparameter tuning to achieve a model that generalizes well to unseen data while capturing the underlying patterns in the data.

Question -: Give short description each of Linear, RBF, Polynomial kernels used in SVM.

Answer -: The Commonly Used Kernels In Support Vector Machines (Svm):

Statistics MACHINE LEARNING ASSIGNMENT 5

Linear Kernel

- The Linear Kernel Is The Simplest Kernel Function Used In Svm.

- It Calculates The Dot Product Between Feature Vectors In The Original Input Space, Essentially Representing A Linear Decision Boundary.

- The Linear Kernel Is Suitable For Linearly Separable Datasets Or Datasets Where A Linear Decision Boundary Is Appropriate.

Rbf (Radial Basis Function) Kernel:-

- The Rbf Kernel Is A Popular Non-Linear Kernel Used In Svm.

- It Computes The Similarity Between Feature Vectors Based On The Euclidean Distance Between Them.

- The Rbf Kernel Can Capture Complex, Non-Linear Relationships In The Data And Is Effective For Datasets With Non-Linear Decision Boundaries.

- The Rbf Kernel Has A Parameter Called Gamma (Γ), Which Controls The Influence Of Each Training Example. Higher Values Of Gamma Lead To More Complex Decision Boundaries And Can Result In Overfitting If Not Properly Tuned.

Polynomial Kernel:-

- The Polynomial Kernel Computes The Similarity Between Feature Vectors Using A Polynomial Function.

- It Introduces Additional Polynomial Features To The Input Space, Allowing The Svm To Capture Non-Linear Relationships.

- The Polynomial Kernel Has A Parameter Called Degree, Which Determines The Degree Of The Polynomial Function Used.

- The Polynomial Kernel Is Useful For Datasets With Non-Linear Decision Boundaries, But It Can Be Sensitive To The Choice Of The Degree Parameter. Lower Degrees May Lead To Underfitting, While Higher Degrees May Lead To Overfitting.

In Summary, The Choice Of Kernel Function In Svm Depends On The Nature Of The Data And The Complexity Of The Decision Boundary Required. The Linear Kernel Is Suitable For Linearly Separable Datasets, While The Rbf And Polynomial Kernels Are Effective For Capturing Non-

Linear Relationships In The Data. Proper Tuning Of Kernel Parameters Is Important To Achieve Optimal Performance And Avoid Overfitting Or Underfitting.