**Documentation**

**Data Sampling Methods:**

Data sampling is a statistical analysis technique used to select, manipulate and analyze a representative subset of data points to identify patterns and trends in the larger data set being examined. It enables data scientists, predictive modelers and other data analysts to work with a smaller, more manageable subset of data, rather than trying to analyze the entire data population. With a representative sample, they can build and run analytical models more quickly, while still producing accurate findings.

**UnderSampling:**

\* **Definition** : UnderSampling involves reducing the number of majority class samples in the dateset to balance the class distribution .

\* **How it works** : UnderSampling reduces the dateset size by discarding instances from the majority class .This approach aims to create a balanced dateset where the number of instances in each class is approximately equal .

\* **When to use:** UnderSampling is useful when the dateset is large, and the majority class overwhelms the minority class, leading to biased model predictions. It helps to mitigate the impact of class imbalance by creating a more balanced training dateset.

**OverSampling:**

\* **Definition** : Oversampling is a technique used to address class imbalance by increasing by the number o minority class sample in the datasets. This is typically done by generating synthetic sample that are similar to the existing minority class instances

\* **How it work** : The most common method for oversampling is Synthetic Minority Over-sampling Technique (SMOTE) . SMOTE creates new minority class samples by interpolating between existing minority class samples . it randomly selects a minority class instance and find it's k-nearest neighbors . Then , it generates new instances along the line segments connecting the original instance to its neighbors .

\* **When to use:** Oversampling is suitable when the datasets has a significant class imbalance, and the minority class is underrepresented. It helps to improve the classifier's ability to learn patterns from the minority class and can lead to better model performance.

**Class Weights :**

\* **Definition** : Class wights are used during model training to assign different weights to different class based on their importance . In the context f handling class imbalance , higher weights are assigned to the minority class to make it more influential during model training .

\* **How it work** : Class weights are incorporated into the loss function of the machine learning algorithm .During training , the algorithm assigns higher penalties for misclassifying minority class instances compared to majority class instances .

\* **When it use** : Class weights are simpler alternative to oversampling and under-sampling . They can be used when the class imbalance is moderate , and the dataset size is sufficient to represent both classes adequately . Class weights are especially useful when implementing algorithms that support class weighting , such as logistic regression , decision trees ,and support vector machines.

**Sampling Method Choice:**

\* The choice between oversampling, undersampling, and class weights depends on the specific characteristics of the dataset and the machine learning algorithm being used. While oversampling and undersampling directly modify the dataset's class distribution, class weights adjust the model's learning process during training. Oversampling and undersampling may lead to increased dataset size or information loss, respectively, while class weights modify the learning process without altering the dataset.

**Models Description :**

1. **Decision Tree:**

**Description**: Decision trees are a type of supervised learning algorithm that is used for both classification and regression tasks. They work by recursively splitting the data based on features that result in the most homogeneous subsets.

**How it work** :Decision trees work by recursively partitioning the data into smaller subsets based on the values of input features. At each step, the algorithm selects the feature that best separates the data into homogeneous classes (nodes) using a specific criterion, such as Gini impurity or entropy. This process continues until a stopping criterion is met, such as reaching a maximum tree depth or having nodes with minimum samples.Decision trees work by recursively partitioning the data into smaller subsets based on the values of input features. At each step, the algorithm selects the feature that best separates the data into homogeneous classes (nodes) using a specific criterion, such as Gini impurity or entropy. This process continues until a stopping criterion is met, such as reaching a maximum tree depth or having nodes with minimum samples.

**Hyperparameters to tune**:

**max\_depth:** Maximum depth of the tree.

**min\_samples\_split:** Minimum number of samples required to split an internal node. **min\_samples\_leaf:** Minimum number of samples required to be at a leaf node.

**max\_features:** Number of features to consider when looking for the best split.

1. **Random Forest Classifier:**

**Description**: Random Forest is an ensemble learning method that constructs a multitude of decision trees during training and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

**How it work:**

Random Forest is an ensemble learning method that builds multiple decision trees during training. Each tree is trained on a random subset of the training data and a random subset of the features. The final prediction is made by aggregating the predictions of all the individual trees (classification: by taking the mode of the predictions, regression: by taking the mean).

**Hyperparameters to tune**:

**n\_estimators**: Number of trees in the forest.

**max\_depth**: Maximum depth of the trees.

**min\_samples\_split:** Minimum number of samples required to split an internal node. **min\_samples\_leaf:** Minimum number of samples required to be at a leaf node.

1. **Gradient Boosting Classifier:**

**Description**: Gradient Boosting is an ensemble technique that builds a strong model by sequentially adding weak learners (typically decision trees) to correct the errors of the previous model. Each tree tries to correct the mistakes of the previous one.

**How it work:**

Gradient Boosting works by sequentially adding weak learners (typically decision trees) to an ensemble, with each one trying to correct the errors of the previous model. The algorithm starts with an initial model (e.g., a single decision tree) and then iteratively fits new models to the residual errors made by the existing ensemble. The final prediction is obtained by summing the predictions of all the models.

**Hyperparameters to tune:**

**n\_estimators:** Number of boosting stages.

**learning**\_**rate**: Rate at which the model learns from mistakes.

**max\_depth:** Maximum depth of the individual trees.

**subsample**: Fraction of samples used for fitting the individual base learners.

1. **Support Vector Classifier (SVC):**

**Description**: Support Vector Classifier is a supervised learning algorithm used for classification tasks. It attempts to find the hyperplane that best separates the classes in the feature space.

**How it work:**

SVC works by finding the optimal hyperplane that separates the classes in the feature space. It does this by maximizing the margin between the closest data points (support vectors) of different classes. In cases where the data is not linearly separable, SVC uses a kernel trick to map the input features into a higher-dimensional space where the classes become separable. Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid.

**Hyperparameters to tune:**

**C**: Regularization parameter. Controls the trade-off between a smooth decision boundary and classifying the training points correctly.

**kernel**: Specifies the kernel type to be used in the algorithm.

**gamma**: Kernel coefficient. Defines how far the influence of a single training example reaches.

**Evaluation Metrics :**

1. **Precision**:

Precision is the ratio of true positive predictions to the total number of positive predictions made by the model. It measures the accuracy of positive predictions. A high precision indicates that the model has a low false positive rate.

**Interpretation**: Precision is important in situations where false positives are costly or undesirable. For example, in medical diagnosis, we want to minimize false positives to avoid unnecessary treatments.

1. **Recall (Sensitivity):**

Recall is the ratio of true positive predictions to the total number of actual positive instances in the data. It measures the model's ability to correctly identify positive instances.

**Interpretation**: Recall is important in situations where false negatives are costly or undesirable. For example, in fraud detection, we want to minimize false negatives to catch as many fraudulent transactions as possible.

1. **F1 Score:**

The F1 score is the harmonic mean of precision and recall. It provides a single metric that balances both precision and recall.

Interpretation: The F1 score is useful when you want to consider both false positives and false negatives. It's a good overall measure of a model's performance, especially when the class distribution is imbalanced.

1. **ROC Curve (Receiver Operating Characteristic Curve):**

The ROC curve is a graphical plot that illustrates the performance of a binary classification model across different threshold settings. It plots the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

**Interpretation**: The ROC curve helps visualize the trade-off between true positive rate and false positive rate. A model with higher area under the ROC curve (AUC) generally indicates better performance.

1. **Learning Curve:**

A learning curve plots the model's performance (e.g., accuracy or error) on the training and validation sets as a function of the training dataset size. It helps visualize how model performance improves or stabilizes with more training data.

**Interpretation**: A learning curve can help diagnose issues such as overfitting (gap between training and validation performance) or underfitting (poor performance on both training and validation sets).

1. **Confusion Matrix**:

A confusion matrix is a table that summarizes the performance of a classification model by comparing actual and predicted class labels. It breaks down predictions into four categories: true positives, true negatives, false positives, and false negatives.

**Interpretation**: The confusion matrix provides insights into the model's errors, including the types of misclassifications it makes. It's particularly useful for understanding class imbalance and the model's bias towards certain classes.