IESEG School of Management

Statistical & Machine Learning: Individual Assignment 2024

PERAN Mathieu

Contents

1 Introduction to Machine Learning Predictive Algorithms

Machine learning algorithms are the backbone of predictive analytics. They allow us to make predictions about future events or understand patterns within data. This report explores five key predictive algorithms: Logistic Regression, Decision Trees, Random Forest, Gradient Boosting Machines (GBM), and Gaussian Naive Bayes.

1.1 Logistic Regression

Logistic Regression is a statistical method for analyzing a dataset in which there are one or more independent variables that determine an outcome. The outcome is measured with a dichotomous variable (in which there are only two possible outcomes).

1.1.1 General Description

Logistic Regression predicts the probability of the target variable being true based on the logistic function. It is widely used for binary classification problems in various fields from medical to social sciences and machine learning.

1.1.2 Function and Mathematics

Mathematically, Logistic Regression estimates the probabilities using a logistic function, which is an S-shaped curve that can take any real-valued number and map it between 0 and 1, but never exactly at those limits. The logistic function $\sigma(z)$ is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

where z is a linear combination of the input features $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n$, β values are coefficients, and e is the base of the natural logarithm.

1.1.3 Fitting Process

The fitting process, also known as training, involves selecting the best coefficients β for the logistic function. This is typically done through an optimization process, such as gradient descent, that aims to minimize the cost function, which is often a log-likelihood function in the case of Logistic Regression.

1.1.4 Pros and Cons

Pros:

- Easy to implement and efficient to train.
- Provides probabilities and fits a linear boundary which is interpretable.
- Can be regularized to avoid overfitting and extended to multiclass classification (Multinomial Logistic Regression).
- Coefficients of the model can be used to interpret the importance of each feature.

Cons:

- Assumes linearity between independent variables and the log odds of the dependent variable.
- Can struggle with complex relationships in data, which might be non-linear.
- Vulnerable to overfitting if there are a large number of features.
- Performance relies on the proper presentation of the feature space.

1.2 Decision Trees

Decision trees are versatile machine learning algorithms that can perform both classification and regression tasks. They work by breaking down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node holds a class label.

1.2.1 General Description of Decision Tree

A Decision Tree is a flowchart-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label or regression value (the decision taken after computing all attributes). The paths from the root to the leaf represent classification rules or regression conditions. In essence, it uses a tree structure to model the decisions and their possible consequences, including chance event outcomes, resource costs, and utility. Decision Trees can handle both categorical and numerical data and are simple to understand and interpret, making them useful for exploratory knowledge discovery.

1.2.2 Function and Mathematics

The core mathematics behind decision trees lies in choosing the best feature to split the data to maximize the homogeneity of the resultant subsets. This is done using criteria like Gini Impurity or Entropy in classification tasks and variance reduction in regression.

For classification, the Entropy for a split is given by:

$$H(T) = -\sum_{i=1}^{c} p_i \log_2 p_i$$

where $H(T)$ is the entropy of the set T, c is the number of classes, and p_i is the proportion of class i in the set.

The Gini Impurity is given by:

$$G(T) = 1 - \sum_{i=1}^{c} p_i^2$$

where $G(T)$ is the Gini Impurity of the set T.

1.2.3 Fitting Process

The algorithm begins with the entire dataset and selects the best feature to split the data into sub-nodes. The selection is based on the mathematical criteria mentioned above. This process is recursive and continues until certain stopping criteria are met, such as maximum tree depth or minimum node size.

1.2.4 Pros and Cons

Pros:

- Intuitive and easy to explain to non-technical stakeholders.
- Can handle both numerical and categorical data.
- Requires little data preprocessing.
- Identifies the most significant variables and the relation between them.

Cons:

- Prone to overfitting, especially if the tree is very deep.
- Can be unstable because small variations in data can result in a completely different tree.

- Biased towards classes with a higher frequency rate.
- Often less accurate compared to other more complex algorithms.

1.3 Random Forest

Random Forest is an ensemble learning method that constructs a multitude of decision trees during training and outputs the mode of the classes for classification or mean prediction for regression of the individual trees. It operates by building numerous decision trees at training time and aggregating their results to improve the overall performance of the model. By doing so, it mitigates the problem of overfitting associated with individual decision trees, offering a more robust and accurate prediction. Random Forest can handle a large number of input variables without variable deletion and is highly effective in high-dimensional spaces.

1.3.1 General Description

A Random Forest is an ensemble learning method, predominantly used for classification and regression, which operates by constructing a multitude of decision trees at training time. The main principle behind Random Forest is the wisdom of crowds; in other words, the combined decision of multiple models gives a more accurate and stable prediction than a single model.

1.3.2 Function and Mathematics

The power of Random Forests lies in the ability to reduce overfitting without substantially increasing error due to bias. It uses averaging to improve the predictive accuracy and control overfitting. The forest chooses the classification having the most votes (over all the trees in the forest) for classification, or average prediction for regression.

Mathematically, the variance of the combined models is given by:

$$\text{Var}(\bar{Y}) = \frac{\text{Var}(Y)}{n}$$

where $\bar{Y}$ is the mean prediction from all trees, and n is the number of trees.

The error of a Random Forest converges to a limit as the number of trees increases. The generalization error depends on the strength of the individual trees in the forest and the correlation between them.

1.3.3 Fitting Process

The fitting process of a Random Forest involves:

1. Bootstrapping datasets from the original data.
2. Growing decision trees from these bootstrapped datasets. At each node:
    - Select a random subset of features to determine the best split.
    - Split the node using the feature that provides the best split according to the objective function.
3. Each tree is grown to the largest extent possible without pruning.
4. Predictions are made by aggregating the predictions of the ensembled trees.

1.3.4 Pros and Cons

Pros:

- High predictive power and robustness to noise.
- Can handle a large number of features and identify the most important ones.
- Effective for datasets with a mixture of numerical and categorical features.
- Does not require feature scaling.

Cons:

- Complexity can lead to long training times.
- Less interpretable compared to individual decision trees.
- Can overfit on very noisy datasets.
- The size of the trees can lead to high memory consumption.

1.4 Gradient Boosting Machines (GBM)

Gradient Boosting Machines (GBM) is a powerful machine learning technique for both regression and classification problems. It builds the model in a stage-wise fashion like other boosting methods do, but it generalizes them by allowing optimization of an arbitrary differentiable loss function.

1.4.1 General Description

GBM works by sequentially adding predictors to an ensemble, each one correcting its predecessor. However, unlike other boosting methods that adjust the weights for every incorrect prediction, GBM tries to fit the new predictor to the residual errors made by the previous predictor.

1.4.2 Function and Mathematics

The mathematics behind GBM involve optimizing a loss function. At each iteration, trees are built to minimize a loss function, and a gradient descent procedure is used to minimize the error. Suppose $L(y, F(x))$ is a loss function where y is the true value and $F(x)$ is the predicted value. The algorithm approaches the minimization of L by iteratively choosing a function h that points in the negative gradient direction. This is analogous to a step of gradient descent for function approximation.

1.4.3 Fitting Process

The fitting process of GBM involves:

1. Start with an initial estimate which could be the mean (for regression) or log odds (for classification).
2. For each stage, fit a decision tree to the negative gradient of the loss function with respect to the current model.
3. Update the model with a shrunken version of the tree's predictions.
4. Iterate the above steps for a fixed number of iterations or until a minimum loss is achieved.

1.4.4 Pros and Cons

Pros:

- Highly effective predictive performance.
- Can handle various types of data and incorporate complex relationships.
- Robust to outliers and can model the impact of rare events in probability estimation.
- Automatically handles missing values and feature selection.

Cons:

- Prone to overfitting if the number of trees is not properly controlled.
- Computationally intensive and requires careful tuning of several parameters.
- Less interpretable compared to simpler models, such as linear regression or decision trees.
- Can be sensitive to noisy data and outliers.

1.5 Gaussian Naive Bayes

Gaussian Naive Bayes is a probabilistic machine learning model that is widely used in classification tasks. It is particularly well-suited for high-dimensional datasets and is based on the Bayes' theorem with an assumption of independence among predictors.

1.5.1 General Description

The Naive Bayes classifier combines the Bayes' theorem with a naive assumption that features are conditionally independent, given the class label. In the Gaussian variant, it is assumed that the continuous values associated with each feature are distributed according to a Gaussian distribution.

1.5.2 Function and Mathematics

In mathematical terms, the Gaussian Naive Bayes model applies the Bayes' theorem with the "naive" assumption of independence between every pair of features. For a given feature x and a class y, the probability distribution $P(x|y)$ is assumed to be Gaussian:

$$P(x|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x-\mu_y)^2}{2\sigma_y^2}\right)$$

where μ_y is the mean of the feature for class y, and σ_y^2 is the variance.

1.5.3 Fitting Process

The fitting process involves the following steps:

1. Calculate the prior probability for each class $P(y)$ based on the frequency of each class in the training set.
2. Calculate the mean μ and variance σ^2 for each feature per class.
3. Apply the Gaussian probability density function to compute $P(x|y)$ for each feature.
4. Multiply the individual probabilities and the class prior to get the unnormalized posterior probability $P(y|x)$.
5. Normalize to get the actual posterior probability and predict the class with the highest probability.

1.5.4 Pros and Cons

Pros:

- Simple to implement and can handle both binary and multiclass classification problems.
- Efficient on large datasets and performs well with an assumption of feature independence.
- Requires a small amount of training data to estimate the necessary parameters.
- Works well with categorical and continuous input features.

Cons:

- The assumption of feature independence is rarely true in real-world applications which can limit its performance.
- Not suitable for regression tasks as it only predicts categorical outcomes.
- Can be outperformed by more complex models like SVM or Random Forests when the feature independence assumption does not hold.
- Sensitive to data distribution: if the Gaussian assumption does not fit the data well, the model's performance can degrade significantly.

2 Benchmark Experiment

2.1 Experimental Setup Explanation

The experimental setup for evaluating various machine learning models is designed with precision to ensure consistency, repeatability, and fairness. This section delves into the rationale behind each component of our experimental setup.

2.1.1 Use of a Pipeline

Rationale: The pipeline mechanism streamlines the process of data transformation, feature selection, handling class imbalance, and model fitting. It ensures data integrity and consistent application of preprocessing steps and model evaluations.

- *Efficiency and Consistency:* Encapsulating sequences of transformations and model fitting into a single object, pipelines maintain consistency across training and validation phases, preventing data leakage.

- *Simplicity in Cross-Validation:* Pipelines, when combined with cross-validation, automate the fitting and transforming process appropriately for each fold, adhering to the principle of not leaking validation data into the model training process.

2.1.2 Variable Selection

Rationale: Effective variable selection improves model performance by reducing overfitting and enhancing interpretability. By removing irrelevant or less important features, models can become more efficient and potentially achieve higher performance on unseen data.

- *Numerical Features:* 'SelectKBest' with a scoring function like 'f_classif' selects statistically significant features, potentially improving model performance by retaining features with a significant relationship with the target variable.
- *Categorical Features:* The use of SelectFromModel with a RandomForestClassifier in preprocessing categorical data serves a dual purpose. Firstly, it reduces the dimensionality of the data by selecting only the most informative features based on their importances calculated from the tree ensemble. This can lead to a model that is less prone to overfitting, especially when the dataset contains a large number of categorical features that may introduce noise. Secondly, it enhances model interpretability by focusing on the most relevant predictors, making it easier to understand the driving factors behind the model's decisions.This method is particularly useful for datasets with high-dimensional categorical data, where naive inclusion of all dummy variables (from one-hot encoding) could lead to model complexity and overfitting. The RandomForestClassifier is effective in this role due to its ability to handle large feature spaces and its intrinsic feature importance measures, which provide a straightforward criterion for feature selection.

2.1.3 Cross-Validation

Rationale: Cross-validation is essential for assessing predictive performance and model generalization on unseen data.

- *Robust Model Evaluation:* Utilizing k-fold cross-validation provides a robust estimate of model performance by using every data subset for both training and validation, thus making better use of the dataset.

2.1.4 SMOTE (Synthetic Minority Over-sampling Technique)

Rationale: Addressing class imbalance through SMOTE helps in improving model sensitivity towards the minority class by creating synthetic examples.

- *Improving Model Sensitivity:* SMOTE counteracts majority class bias by ensuring adequate representation of the minority class, potentially improving recall without severely affecting precision.
- *Enhanced Generalization:* Generating synthetic samples prompts the model to learn more generalized class representations, beneficial for performance on unseen data.

Conclusion: This comprehensive framework, comprising pipelines, variable selection, cross-validation, and SMOTE, is crafted to address challenges like preprocessing consistency, feature relevance, robust model evaluation, and class imbalance. Each component plays a crucial role in developing performant and generalizable models.

Evaluation Metrics:

Multiple metrics were employed to evaluate model performance, including Accuracy, AUC (Area Under the ROC Curve), Precision, Recall, and F1 score. Using a variety of metrics provides a holistic view of model performance, especially in the presence of class imbalance.

3 Analysis of Machine Learning Models

This analysis delves into the performance of five distinct machine learning models: Logistic Regression, Decision Tree, Random Forest, GBM (Gradient Boosting Machines), and Gaussian Naive Bayes. Each model was meticulously evaluated across various metrics, including Validation Accuracy, AUC, Precision, Recall, and F1 Score, under a uniform experimental setup that involved variable selection, cross-validation, SMOTE for handling class imbalances, and pipelines for efficient data processing. The following provides a comprehensive analysis of each model, highlighting its strengths, limitations, and potential areas for further research.

	Logistic Regression	Decision Tree	Random Forest	GBM	Gaussian Naive Bayes
CV Accuracy	0.746375	0.835687	0.894375	0.898813	0.735125
Validation Accuracy	0.836250	0.830000	0.893500	0.895000	0.730250
Validation AUC	0.752821	0.626586	0.763632	0.786968	0.760590
Validation Precision	0.358650	0.296763	0.559829	0.576577	0.248996
Validation Recall	0.561674	0.363436	0.288546	0.281938	0.682819
Validation F1	0.437768	0.326733	0.380814	0.378698	0.364921

Figure 1: Metrics summary

3.1 Logistic Regression

Logistic Regression showed a commendable balance in its performance, particularly excelling in Recall. Its simplicity and interpretability make it a solid baseline for binary classification problems, although it tends to assume linear relationships between features.

Limitations and Further Research: The model's performance might improve with the exploration of non-linear feature transformations or regularization techniques to address potential overfitting and enhance model generalizability.

3.2 Decision Tree

The Decision Tree demonstrated reasonable accuracy, yet struggled with lower AUC, indicating its limitations in effectively separating the classes. It remains highly interpretable, allowing for easy extraction and understanding of decision rules.

Limitations and Further Research: This model is prone to overfitting, especially without careful tuning of its parameters. Techniques such as tree pruning, setting maximum depth, or integrating ensemble strategies could improve its predictive accuracy and robustness.

3.3 Random Forest

Random Forest topped the charts in terms of accuracy, benefiting from its ensemble approach which helps in reducing variance and avoiding overfitting problems common to Decision Trees.

Limitations and Further Research: While effective, Random Forest models can become computationally expensive and may suffer from increased complexity when incorporating a large number of trees. Optimizing tree count and feature selection parameters could enhance efficiency without compromising performance.

3.4 Gradient Boosting Machines (GBM)

GBM stood out with high metrics across the board, illustrating its capability in handling both bias and variance effectively through the boosting of weak learners.

Limitations and Further Research: The model's sensitivity to outlier and noise could be mitigated through advanced regularization methods. Further exploration into loss functions and learning rates might also yield improvements in model performance.

3.5 Gaussian Naive Bayes

Notably, Gaussian Naive Bayes achieved high Recall but the lowest precision, suggesting it is effective in identifying positive classes but at the cost of a higher false positive rate. Its efficiency in processing large datasets quickly is a significant advantage.

Limitations and Further Research: The assumption of feature independence is a major limitation. Integrating approaches that can model feature correlations or using hybrid models could potentially boost its performance.

4 Conclusion

This report has meticulously evaluated the performance of five critical machine learning algorithms—Logistic Regression, Decision Trees, Random Forest, Gradient Boosting Machines (GBM), and Gaussian Naive Bayes—across various metrics such as Accuracy, AUC, Precision, Recall, and F1 Score. Through a robust experimental setup that included variable selection, cross-validation, and handling of class imbalances using SMOTE, each model was tested to uncover its strengths and weaknesses in a controlled and fair environment.

4.1 Key Findings

Our analysis showed that the ensemble methods, particularly Random Forest and GBM, consistently outperformed single-model approaches such as Logistic Regression and Decision Trees in terms of Accuracy and AUC. The ability of these ensemble methods to aggregate multiple weak learners into a more robust and stable model significantly reduces variance and helps avoid the overfitting problems that are often seen with single decision trees.

Random Forest not only demonstrated the highest overall accuracy but also maintained good balance across other metrics, benefiting from the ensemble's ability to average decisions from multiple deep decision trees. GBM, on the other hand, showed superior performance by effectively minimizing bias and variance through its boosting technique, which sequentially corrects the predecessors' errors.

4.2 Performance Considerations

The performance of Gaussian Naive Bayes and Logistic Regression, while lower in Precision and F1 Score, highlighted important considerations. Gaussian Naive Bayes, known for its simplicity and efficiency in handling large datasets, was limited by its assumption of feature independence, often leading to suboptimal precision. Logistic Regression, despite good Recall and AUC, faced limitations due to its inherent assumption of linear relationships between features.

4.3 Strategic Implications

The selection of an algorithm should be guided not only by its statistical performance but also by the business or operational context in which the model will be applied. For instance, Logistic Regression or Decision Trees may be preferred in scenarios where interpretability is crucial, despite their relatively lower performance metrics compared to ensemble methods.

4.4 Future Directions

Future research could focus on hybrid models that combine the interpretability of simpler models with the predictive power of ensemble techniques. Moreover, advancements in regularization and feature engineering are likely to enhance the performance of all models discussed here.

In conclusion, while no single model universally outperforms others across all evaluated metrics, the strategic selection of a model based on specific performance criteria and business needs can greatly enhance the effectiveness of predictive analytics projects. The insights gained from this benchmarking exercise emphasize the importance of a nuanced approach to model selection, underlining the need for a comprehensive understanding of the strengths and limitations of each algorithm within various application domains.