MET CS 566

Analysis of machine learning algorithms for binary classification of fraudulent warranty claims

Presented By: Raghav Jindal, Shivam Bhardwaj and Tejaswi LNU

Introduction

Machine learning (ML) is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention. Finding fake warranty claims is a challenge for a business that offers warranty services to clients. Every day, the company receives a sizable number of warranty claims from clients, many of which are thought to be fake. For the company's financial health and to keep the faith of their loyal consumers, it is essential to recognize and reject false claims.

The business is interested in creating a machine learning model that can reliably categorize warranty claims as legitimate or fraudulent in order to solve this problem. The algorithm should be able to recognize trends and anomalies that suggest the probability of fraud by learning from previous data.

The objective of this project is to develop a machine learning model that can accurately classify warranty claims into two categories: genuine and fraudulent. The model should be able to analyze various features associated with each claim, such as product information, customer information, and claim details, to identify patterns that indicate potential fraud. To train and evaluate the machine learning model, the company will use a sizable dataset of previous warranty claims. The dataset contains details about the product, the client, and the kind of claim made. The algorithm should be able to learn from this information and distinguish between legitimate and fraudulent new warranty claims.

The success of this project will be measured by the accuracy of the model in correctly classifying warranty claims. A high-performing model will help the company to detect and reject fraudulent claims, which will result in significant cost savings and improved customer satisfaction.

Data Description

The dataset used for this analysis contains information on warranty claims, including the age of the product, the type of product, the warranty period, and the reason for the claim. The data contains 10,000 observations and 20 variables. AC_1003_Issue: The issue related to the air conditioner with code AC_1003, TV_2001_Issue: The issue related to the television with code TV_2001, TV_2002_Issue: The issue related to the television with code TV_2002, TV_2003_Issue: The issue related to the television with code TV_2003, Claim_Value: The amount of money being claimed for the service, Service_Centre: The location of the service center where the product is being serviced, Product_Age: The age of the product in years, Purchased_from: The source from which the product was purchased, Call_details: Details related to the service call, including date, time, duration, and outcome, Purpose: The reason for the service call, Fraud: A binary variable indicating whether or not the claim is suspected to be fraudulent.

Brief description of models used:

The project utilizes six different classification algorithms. These algorithms are listed as follows:

1. **Support Vector Machine (SVM) :** Support vector machine (SVM) is an example of a supervised learning method which is used for regression, outlier detection and

classification. In our research, we will be using SVM for classifying potential fraudulent cases of car insurance claims. SVM algorithm aims to develop a decision boundary or the best line that can help in segregating the n-dimensional space into multiple classes, helping the user to easily classify any new(seen or unseen) data point in the right category in future. This decision boundary is known as a hyperplane. SVM works by choosing vectors or the extreme points that can help in defining the hyperplane. Such extreme cases are termed as support vectors, therefore the algorithm is coined as a support vector machine. In our research, we have deployed Support Vector Clustering (SVC) to classify our labels.

2. **K-nearest neighbors (KNN) :** KNN is a non-parametric estimation method which can be used for both regression and classification problems. It can be utilized in a motley of pattern recognition and estimation problems. KNN algorithm evaluates the output value of any given input vector by examining the output values in the vicinity of similar k neighbors. The measure of similarity is usually determined using a function that determines distance. Various such distance calculating functions have been deployed to evolve KNN algorithms, such as Chebyshev distance, Mahalanobis distance, Euclidean distance, etc**.** The output of any given sample is evaluated by weighted averaging or normal averaging of its k nearest neighbors. The optimal value of k can be obtained with the help of a validation error curve. The algorithm iterates from 1 to the total number of points in training data and calculates the distance between each point in the training dataset and the given test data. The distances calculated by the algorithm are then sorted

in ascending order out of which the most frequent class in the top k rows of the sorted array is then returned as the predicted class.

3. **Logistic regression :** The logistic model (or logit model) is a statistical model that models the probability of an event taking place by having the log-odds for the event be a linear combination of one or more independent variables. In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (the coefficients in the linear combination). Formally, in binary logistic regression there is a single binary dependent variable, coded by an indicator variable, where the two values are labeled "0" and "1", while the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). Logistic regression is used in various fields, including machine learning, most medical fields, and social sciences. In this project, we will be using logistic regression for binary classification of fraud car insurance claims. The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function, hence the name. The unit of measurement for the log-odds scale is called a *logit*, from the logistic *unit*, hence the alternative names.

4. **Random Forest :** A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Random forests or random decision forests is an ensemble learning method for classification, regression and other tasks that operates

by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance. The first algorithm for random decision forests was created in 1995 by Tin Kam Ho using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.

5. **Naive Bayes Classifier:** The family of straightforward "probabilistic classifiers" known as "naive Bayes classifiers" in statistics is based on the application of Bayes' theorem with strong (naive) independence assumptions between the features (see Bayes classifier). Despite being among the simplest Bayesian network models, they can reach great levels of accuracy when used in conjunction with kernel density estimation. The number of parameters required for naive Bayes classifiers is linear in the number of variables (features/predictors) in a learning problem, making them extremely scalable. Instead than using an expensive iterative approximation, which is how many other types of classifiers are trained, maximum-likelihood training can be accomplished by evaluating a closed-form expression, which requires linear time. Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. For training such classifiers, there isn't just one technique, but rather a

family of algorithms built on the premise that, given the class variable, the value of one feature is independent of the value of every other feature. For instance, if a fruit is red, round, and roughly 10 cm in diameter, it may be regarded as an apple. Despite any potential correlations between the variables of color, roundness, and diameter, a naive Bayes classifier considers each of these features to contribute independently to the likelihood that this fruit is an apple. It is possible to operate with the naive Bayes model without embracing Bayesian probability or applying any Bayesian techniques because parameter estimation for naive Bayes models frequently employs the maximum likelihood method. Naïve Bayes classifiers have performed admirably in a variety of challenging real-world circumstances despite their naive design and ostensibly oversimplified assumptions.

6. **Decision Tree Classifier :** In a decision tree, which resembles a flowchart, each internal node represents a feature (or property), a branch represents a decision rule, and each leaf node displays the outcome. In a decision tree, the root node is the topmost node. It gains the capacity to split data into subsets based on attribute values. The method employed to partition the tree is called recursive partitioning. Making decisions is aided by this framework, which resembles a flowchart. It is an exact portrayal of how people think, like a flowchart. This makes decision trees easy to understand and interpret.The decision tree is a type of ML algorithm known as a white box. Black box algorithms like neural networks do not have this functionality; rather, internal decision-making logic is shared. When compared to the neural network algorithm, it trains more quickly. The amount of records and number of attributes in the given data determine the temporal complexity of

decision trees. The decision tree is a non-parametric or distribution-free strategy that does not rely on the assumptions of a probability distribution. High dimensional data can be handled by decision trees accurately.

The dataset was split into training and testing sets in a ratio of 80:20. The models were then built on the training data and tested on the testing data. The performance of each model was evaluated using accuracy as the metric. The following table summarizes the accuracy scores of each model.

Algorithm	**Accuracy**	**Time Complexity**	**Space Complexity**
Logistic Regression	0.922	O(n^2)	O(n)
Decision Tree	0.994	O(nlogn)	O(n)
Random Forest Classifier	0.922	O(nlogn)	O(n)
Gaussian Naive Bayes	0.867	O(n)	O(n)

Linear Support Vector Classifier	0.922	O(n^3)	O(n^2)
K-Neighbours Classifier (k=3)	0.988	O(n)	O(n)

From the above table, it can be observed that the K-neighbors classifier with k=3 has the highest accuracy score of 0.988. It also has the lowest time and space complexity, making it the best model for this dataset.

Conclusion

In conclusion, the Decision Tree is the best algorithm for predicting genuine warranty claims, based on its high accuracy and low time and space complexity. However, further analysis can be done to optimize the hyperparameters of the other algorithms to improve their performance.

Bibliography

1. Python documentation : https://docs.python.org/3/
2. Scikit-learn documentation : https://scikit-learn.org/stable/
3. Pandas documentation : https://pandas.pydata.org/pandas-docs/stable/