Tyre Inc. R&D Project Report

Sommario

1. Introduction

The aim of the analysis was to develop a classification model to predict the failure of a tires test, given some environmental attributes. The dataset is composed of 15 attributes, 8 numerical and 7 categoricals, and the binary target that showcases whether the was a failure (1) or not (0). Given that the target variable is of a binary nature, the problem was reconducted to a classification one.
To attend to the objective, we developed and perfected 8 different models, that we'll further explain in the dedicated section (3. Classification Models).

2. Data Preparation & Exploration

In order to have a better understanding of the characteristics of the dataset, we performed a first exploration of the dataset, which enabled us to optimally feed it to the models.

2.1 Incomplete data

The first step of the process was to understand whether some attributes had missing or incomplete data. The only variable with missing information (NAs) was the "diameter" of the tire. A further analysis revealed that the missing values accounted for the 70% of the total observations. Because of this reason, we decided to drop the column as inspection, identification and substitution were not viable options.

2.2 Data Distribution & Outliers

To assess potential outliers and the possibility of data simplification, to improve the performances of the developed models, we check for known distributions among numerical attributes. Shapiro-Test was conducted to evaluate normality: having extremely low p-values for all the variables, we were forced to reject the Null Hypothesis of the test (H_0: data follow normal distribution).
A further analysis of the numerical attributes led us to discard some potential outliers, performing a boxplot analysis, because of other tests, as the $\mu \pm 3\sigma$ test, weren't pursuable alternatives in the absence of a normal distribution. However, the results obtained by our top performant models, weren't as good as the ones obtained without outlier elimination. For this reason and for better code readability, the outlier analysis is not displayed on the python file.

2.3 Gini Index

To continue the analysis, we converted the type of the categorical variables from the native integers to categoricals. Furthermore, we computed the Gini Index for each categorical attribute, to evaluate the heterogeneity of each variable, identifying that "add_layers" was the most homogeneous one, meaning that a single class is much more frequent than the others, thus having little impact on the target variable. For this reason, to simplify the model, this variable was dropped. Even in this case, results were worse than ones without dropping the attribute, forcing us to keep it.

2.4 Standardization

To complete the analysis on the numerical variables, we decided to the MinMax Standardization technique, to overcome the different orders of magnitude of the attributes, that can eventually have a negative impact on the classification models.

2.5 Imbalanced Data

Oversampling and downsampling are techniques that can be used to adjust the class distribution in a dataset in order to address imbalanced classes.

This can be a problem when training a classification model because the model may be biased towards the more prevalent class, resulting in poor performance on the minority class. For these reasons, we developed both an oversampled and a downsampled dataset. All the models were trained with the 3 datasets (original, oversampled and downsampled), that were divided in the corresponding training and test sets. Although the models were trained with the respective datasets, the performances were assessed only on the test set of the original dataset.

3. Classification Models

To identify the best prediction outcome, we fitted 8 different models. On the following table, the F-1 test of the different models is displayed:

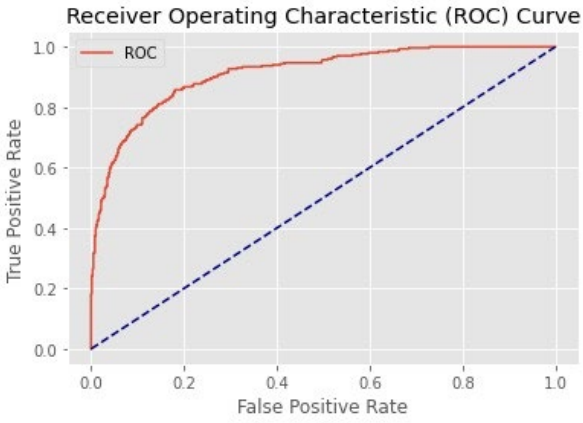
Classification Model	Original Dataset		Oversampled Dataset		Downsampled Dataset	
	Train Set	Test Set	Train Set	Test Set	Train Set	Test Set
KNN	0.652	0.433	0.938	0.713	0.614	0.588
Classification Tree	0.786	0.557	0.869	0.667	0.723	0.656
Random Forest	0.873	0.532	0.937[1]	0.757[1]	0.727	0.670
Adaboost	0.615	0.554	0.798	0.674	0.622	0.6
Logistic Regression	0.562	0.537	0.643	0.608	0.647	0.620
SVM	0.554	0.531	0.925	0.768	0.655	0.632
Multi-Layer Perceptron Classifier	0.656	0.542	0.689	0.619	0.656	0.637
Naive Bayes	0.590	0.560	0.591	0.571	0.599	0.572

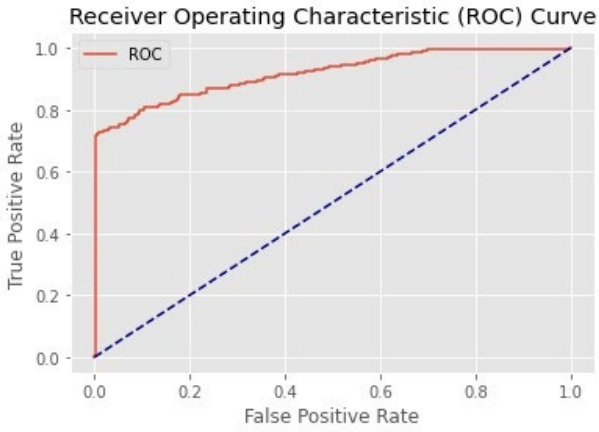
[1] Results referred to the optimal parameters setting

As shown by the table, “Random Forest” and “SVM” models were the best performing ones. For this reason, we decided to compare more performance metrics between the two.

RANDOM FOREST	precision	recall	f1-score	support
0	0.93	0.77	0.84	598
1	0.66	0.88	0.76	302
accuracy			0.81	900
macro avg	0.8	0.83	0.8	900
weighted avg	0.84	0.81	0.81	900

SVM	precision	recall	f1-score	support
0	0.91	0.82	0.86	598
1	0.7	0.84	0.77	302
accuracy			0.83	900
macro avg	0.81	0.83	0.82	900
weighted avg	0.84	0.83	0.83	900





Both the ROC Curves have AUC = 0.92

4. Model Selection

We believed that both of these models are suitable for our prediction purposes. However, we decided to go for the “**Random Forest**” model for different reasons.

Firstly, we believed that it’s more important to correctly identify the most numbers of failure, as we think to be much more costly to not identify a tire failure (for safety reasons). Thus, the higher True Positive Rate of the “Random Forest” supposed an advantage over the “SVM” model. Moreover, “Random Forest” models provide a measure of feature importance, which can be useful for understanding which features are most important for making predictions. “SVMs” do not provide this information. Again, for safety purposes, we believed it was important to clearly identify attributes that have high incidence in the test failure.

Finally, is worth noting that “Random Forest” models are generally less prone to overfitting compared to “SVMs”. This is because “Random Forest” models use an ensemble of decision trees, which tend to reduce the variance in the model. “SVMs”, on the other hand, can be more prone to overfitting if the model is overly complex or if the dataset is small.