MSc. Data Science & Artificial Intelligence

Introduction to Machine Learning

Dr. Michel Riveill & Dr. Diane LINGRAND

Final project: Petfinder, predicting adoption

Author: Joris LIMONIER
joris.limonier@hotmail.fr
Due: January 15, 2022

Contents

List of Figures

List of Tables

1 Problem description

The problem we are trying to solve consists of predicting whether an animal will be adopted from a shelter within 30 days, given several pieces of information on this animal. This problem is a clean and reduced version of a Kaggle competition dating back from 2019.

2 Exploratory Data Analysis

We would like to get some basic information of the data set before diving into the machine learning solution.
The training set has shape (8168×16) and the test set has shape (250×16), where the column names and data types are summarized in Table 1.

CATEGORICAL		NUMERICAL	TEXT	IMAGE
Type	MaturitySize	Age	Description	Images
Gender	FurLength	Fee		
Breed	Vaccinated			
Color1	Dewormed			
Color2	Sterilized			
Color3	Health			

Table 1: Data types per column

Overall, the data set is very clean as it contains 0 NaN values.

3 Problem Solution

3.1 Overview

The problem can be solved by using a pipeling, which is represented in Figure 1. The pipeline consists of two steps:

- Data Pre-processing
- Classification

We will detail these two steps in the following subsections.

3.2 Data Preprocessing

We saw in section 2 that the data is already fairly clean (*e.g.* in terms of NA's). We still need to process the columns, which we do depending on the type of data they contain. As per Figure 1, we use:

- A Categorical Preprocessor
- A Numerical Preprocessor

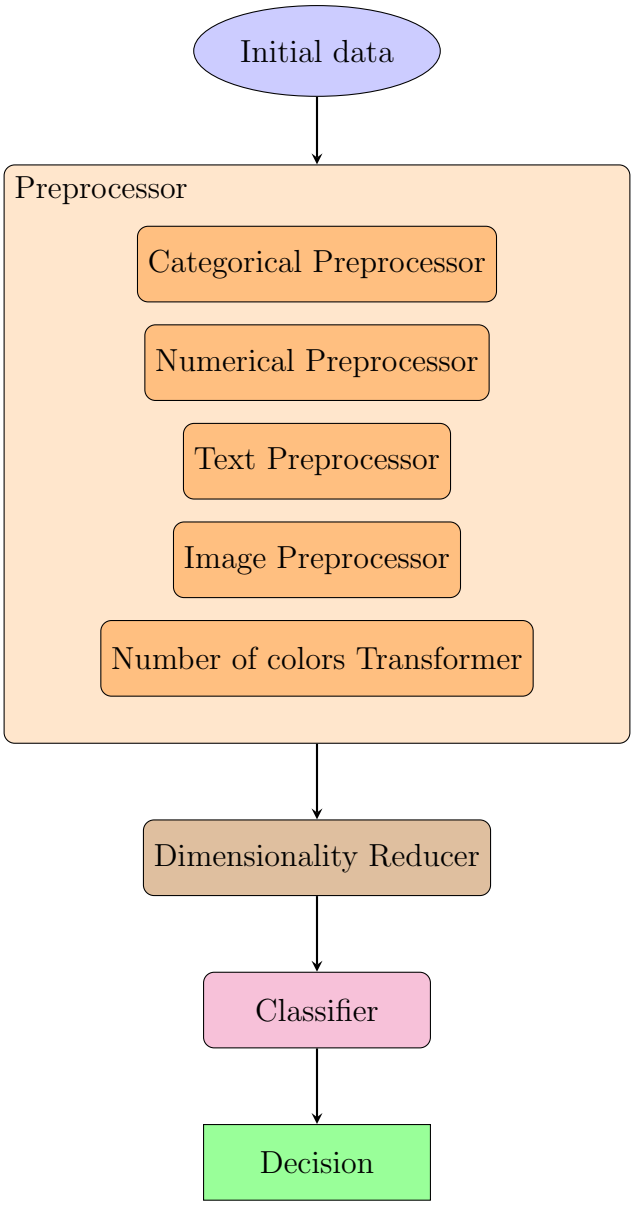

Figure 1: Complete pipeline diagram

- A Text Preprocessor
- A Image Preprocessor
- A Transformer for the number of colors

We detail the above in their respective paragraphs.

Categorical Preprocessor The Categorical Preprocessor is composed of a OneHotEncoder, which encodes categorical features as a one-hot numeric array. That is, for each of the categories, we create a new column with 1 if the initial column is of that category and 0 otherwise.

Numerical Preprocessor The Numerical Preprocessor contains a StandardScaler. Its role is to center and scale each of the numerical variables in order to remove differences

in orders of magnitude.

Text Preprocessor The Text Preprocessor contains a TfidfVectorizer, which converts the raw comments from our text feature to a matrix of Term Frequency-Inverse Document Frequency (TF-IDF) features. We used a CountVectorizer initially, but it gave worse results than the TF-IDF, which is why we kept the latter. The CountVectorizer counts the occurences of each word in the *Description* column. A possible explaination of why the TfidfVectorizer performs better is that the features it produces not-only contain information about the frequency of each term, but they also account for the frequency of each word within the whole document.

Image Preprocessor The Image Preprocessor consists of a custom BOF_extractor. It extracts Scale-Invariant Feature Transforms (SIFTs) and computes Bag Of Features (BOF) on the images from the *Images* column.

Number of colors Transformer The Number of colors Transformer uses FunctionTransformer to compute how many colors the animal has (*i.e.* colors different from "Unknown"). It then sets the number of colors as a feature.
In addition to the preprocessors and transformers presented above, we perform Dimensionality Reduction:

Dimensionality Reducer The Dimensionality Reducer consists of a TruncatedSVD transformer, which applies Latent Semantic Analysis to reduce the number of features in the data set.

3.3 Classification

3.3.1 General approach

Our approach for the Classification part separates in two parts

1. Try various classifiers and evaluate their performance. Then, keep the best 5 classifiers.

2. For the best 5 classifiers, fine-tune their hyperparameters and find the best one.

3.3.2 Select Models

The algorithms tested are presented in Table 2. We computed their accuracy following GridSearchCV (on the train data), then computed their prediction on the test data. We did not test XGBoost because it was too slow.

Classifier	TestAccuracy	Train Accuracy
RandomForestClassifier	0.592	0.625
BernoulliNB	0.584	0.593
GradientBoostingClassifier	0.572	0.633
AdaBoostClassifier	0.572	0.605
MLPClassifier	0.564	0.608
GaussianNB	0.552	0.569
GaussianProcessClassifier	0.548	0.514
SVC	0.508	0.525
SGDClassifier	0.504	0.516
KNeighborsClassifier	0.496	0.513
DecisionTreeClassifier	0.476	0.556

Table 2: Accuracies on first prospect

3.3.3 Fine-tune hyperparameters

As shown in Table 2, the five best algorithms are GradientBoostingClassifier, RandomForestClassifier, AdaBoostClassifier, MLPClassifier and BernoulliNB. The next step is to select several hyperparameters for each of them and test them with GridSearchCV.

4 Evaluation & critical view

,,

,,,,

Glossary

BOF Bag Of Features. 3

SIFT Scale-Invariant Feature Transform. 3

TF-IDF Term Frequency-Inverse Document Frequency. 3