Machine Learning Engineer Nanodegree

Capstone Proposal

Raphael Bürki
September 3rd, 2018

Domain Background

The goal of the proposed project is to help to reduce churn (the number of customers who switch away from one supplier to another) for a large Swiss automotive retail and maintenance company with about 70 branches. These branches sell and service cars of the brands Volkswagen, Audi, SEAT, SKODA and Volkswagen Commercial Vehicles (number of brands sold / serviced varies for each subsidiary).

Automotive retail is under huge pressure. The earnings from selling cars are not enough to cover the cost, so the company is losing money with every sale (it sells about 40'000 new cars a year). It's only hope to recover these losses and to make profits comes from the recurring aftersales service and maintenance works that every car generates for years (the older the car, the higher the earnings per event). But customers are free to choose where to service their cars and so it is crucial for the company to turn the car buyers into loyal service customers. Unfortunately, a lot of them churn soon.

(Until recently I was responsible for the company's marketing department and guessing about ways to target customers with high churn risk directly. Before we could act on this I left the company (in good terms). The company has supplied me with anonymized customer data that I can use for this project, in return I will share my results with them.)

Problem Statement

Every two years or 30'000 km of mileage a car must appear for mandatory service. With every service cycle a growing proportion of erstwhile customers does not show up anymore in the company's repair shops. (Fig 1) The company now wants to contact all customers 3 month before the next mandatory service is due (it can interfere the period) with a reminder mailing and to offer a substantial discount only to those customers that are about to churn to motivate them to turn up and they stay loyal. Problem is, it doesn't know which of it's many customers to target with such a discount. If the discount is given to customers that would have showed up anyway, it will mean a big diminution of the company's earnings.

With help of predictive modelling this project wants to help to identify the customers most at risk for churn.

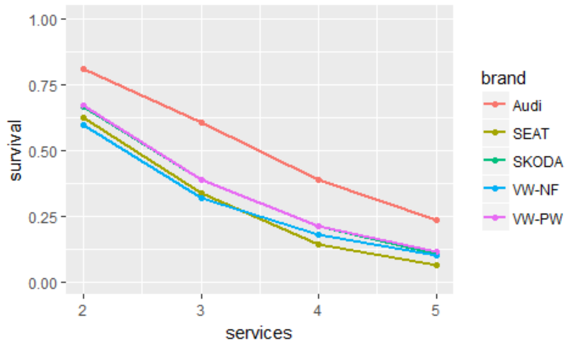

Fig 1: "survival rate" / customer loss from mandatory service event to next, average is roughly 35%.

Datasets and Inputs

The main dataset for this project is a CSV-file (cars.csv) containing anonymized data of 74'130 cars and their owners (for the rest of the paper we will just speak of cars because the car ID is the unique identifier of the occurrences in the dataset). The selected cars have been bought from the company and been in mandatory service at least once in the period from 2006 to 2017. Data gathering was done by the company's BI team. Because of limited resources / effort and a rather old-fashioned IT-architecture it was quite a difficult process and various information from different sources has been consolidated in the file. There are 77 features that can roughly be grouped in 4 areas:

- Features on a cars age, usage and service history
- Features on technical car specifications
- Features on a cars holder, either from the company's CRM-system and enriched / purchased demographic data from an external supplier
- Features on the company branch the customer is affiliated with

A big part of the features will probably be rather useless for our problem and the dataset must be thoroughly preprocessed before use. But especially the first group of features could, in combination with some of the other features, reveal interesting information on how to identify customers who are likely to turn away from the company.

There is a second small dataset containing information about the branches (67 observations with 16 features) that could be helpful: It contains the information on which branch offers service for which car brands (CSV-file, branches.csv). It is intended to use this information to calculate distance from customer's home address to next branch offering service work for the respective car brand.

Solution Statement

With help of the date of the last service event it was possible to label the cars in binary classes ACTIVE (active customers) and CHURN (lost customers): All cars which have not turned up for mandatory service after more than 24 + tolerance 3 month after their last service event can be classified as churn, because they should have turned up in the meantime. The other cars are still loyal (at least as far as the company can know).

With help of this classification it is possible to apply different supervised machine learning algorithms for binary classification to the provided historical data. The goal is to test several possible classifiers, select the one that is best suited and train a model that can predict which of the today's active customers are most at risk for churn with help of a classification scoring.

Benchmark Model

Today the company has no means to identify the customers at risk. It knows that overall it loses roughly 35% of customers from service event to service event (see Fig 1). If it contacts a selction of random customers offering them a retention discount for their next service visit, it will offer 65% of its discounts to customers that would have come anyway. This naïve approach is to be beaten by far.

Evaluation Metrics

One important evaluation metric has been hinted at in the section above: the false positive rate must be as low as possible (→ as few loyal customers that are offered discounts as possible). Therefore, a model's ability to *precisely* predict those that are about to churn is more important than the model's ability to *recall* all potential churners:

- precision: true positives / (true positives + false positives) - how many of the model's identified CHURN cars are really CHURN cars
- recall (sensivity): true positives / (true positives + false negatives) - how many of all the CHURN cars in the data set are correctly identified as CHURN cars

In this project the **F-beta score** will be used as main metric to compare results of different classifiers:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

It considers both precision and recall but gives more emphasis on precision if we select a beta of < 1. In this project beta will be set to 0.5, a value that is commonly used to emphasize precision (see https://en.wikipedia.org/wiki/F1_score for reference).

Project Design

The workflow for approaching a solution to the problem described is listed below in form of a checklist. There will be much emphasis on the data preprocessing / data preparation step to give the different algorithms a good base. Also, I will try to automate as many of those steps as possible with the help of functions so that I can some of the preparations choices as hyperparameters.

Workflow:

1. Frame the problem ✓
2. Select a performance measure ✓
3. Have a first peek at data
4. Put aside test set (with stratified sampling)
5. Perform Data exploration
   - types
   - usability for task
   - missing values
   - statistics: distributions, outliers (visual)
   - correlations (visual)
6. Prepare the data (with functions / pipelines)
   - Clean Data (outliers, missing values)
   - Handle categorical and text attributes
   - Feature selection
   - Feature engineering (discretization, decomposition, aggregation, …)
   - Feature scaling (standardize or normalize)
7. Short-list promising models
   - Train quick and dirty models from different categories (see below)
   - Measure and compare performance (with N-fold cross validation and fold means, sd)
   - Analyze most significant variables for each algorithm
   - Further (quick) round of feature selection
   - Select 3-5 most promising models
8. Fine-tune system
   - Fine tune hyperparameters using cross-validation
   - Try ensemble methods of best models
9. Measure performance on test set
10. Prepare Presentation of solution

Models that will be tested for this classification task:

- LogisticRegression
- GaussianNB (Naïve Bayes)
- KNeighborsClassifier
- SVC
- RandomForestClassifier
- XGBClassifier
- (Optional / for reference and if I have the time only: Sequential Neural Network (in Keras))