Udacity – Machine Learning Engineer Nanodegree – Capstone Proposal – Predict Survival on the Titanic

Domain Background

Based on the problems I will be facing in the near future in my Data Science consultant job at Avanade, I decided to look for a "traditional" classification problem where a class has to be predicted based on a relational dataset as my capstone project. With a project like this in my portfolio, I have a first template for the machine learning workflow behind this kind of very common machine learning problem, which I can then reuse to solve future use cases.

My search for a suitable problem led me to Kaggle and I found the "Titantic: Machine Learning from Disaster" challenge, which I find quite interesting to solve (https://www.kaggle.com/c/titanic).

The goal of this challenge is to use machine learning to predict which passengers survived the Titanic shipwreck. As such, it will be a binary classification task and part of the supervised learning domain. In supervised learning, we train a model with a (historic) dataset containing the correct labels. Based on these provided labels and the input features, different algorithms exist that can then find the inherent relation between the input features and the target variable. The model can then be used to predict the outcome for new data that does not have labels yet.

Since classification problems based on relational datasets are often very similar in nature, solving this problem will help me to understand and build the complete machine learning workflow for such a problem, which I can then reuse in one way or the other for future endeavors, making this project very relevant for my personal goals.

Problem Statement

The problem to be solved is to create a well-performing machine learning model workflow that can be used to predict whether a titanic passenger survived or not. I will set myself the goal to achieve 90% accuracy. In addition, I want to create a well-rounded, reproducible pipeline to train many different models and allow for hyperparameter tuning to select the model with the highest prediction quality.

Datasets and Inputs

The dataset is obtained from Kaggle and contains details of a subset of the passengers on board (891 to be exact). This data includes passenger information such as the name of the passenger, age, gender, socio-economic class etc. The dataset also contains the label which I will try to predict with my model: whether the passenger survived or not. I will have to conduct some initial explorative data analysis to see which columns from the dataset could help me in predicting this target variable.

Solution Statement

My proposed solution is to train different supervised learning models that are suited for binary classification, such as Support Vector Machines (SVMs) and XGBoost models. I will set up a machine learning pipeline to train and test different models and conduct hyperparameter tuning to receive the best-performing model in terms of prediction quality.

Benchmark model

As a benchmark model, I will first train an untuned Logistic Regression model on all the features available in the dataset as this is the most simple algorithm that can be used for binary classification (although not necessarily the one with the lowest prediction performance). This will allow me to assess more complex algorithms and also the quality of the features that I will engineer.

Evaluation Metrics

There are different possible evaluation metrics for a binary classification task. The most straightforward one is accuracy, which just measures the percentage of data points that we have predicted correctly. The accuracy is calculated as follows:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}}$$

where TP refers to true positives, TN to true negatives, P to overall positives and N to overall negatives. True positive in this case means that the model has predicted positive (this will be survived) and the real label is also positive whereas the overall positives also include the cases where the model has predicted negative (not survived) but the real label is positive (survived), i.e. the false positives.

There are also other evaluation metrics which are more suited in the case of an imbalanced dataset and when you want to prioritize predicting one class correctly over the other. These metrics include precision and recall and can be calculated as follows:

Recall: $$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

Precision: $$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

where TP: true positives, P: positives, FN: false negatives, FP: false positives (source: https://en.wikipedia.org/wiki/Precision_and_recall)

In my case I will just focus on accuracy as the dataset Is quite balanced and this is also the target metric for Kaggle.

Project Design

As a first part of this challenge, it will be necessary to conduct some exploratory data analysis (EDA) on the previously described dataset to get a feeling for the data available and find potential features that can be used to predict the target variable (Survival “Yes” or “No”). Part of this EDA will for example be visualizing data to understand their distribution and find missing values, as well as calculating correlations with the target variable.

After EDA, the feature engineering task has to be completed. This includes selecting already existing promising features from the dataset as well as potentially building new features by combining the data at hand to get even more powerful features. In addition, the data also has to be cleaned and processed in order to bring it in a format that is compatible with the machine learning models I want to train. This includes for example scaling as well as one-hot-encoding of categorical variables.

As a next step, a first baseline model will be built as described in the “Benchmark model” section.

It will also be necessary to have an evaluation metric to compare different models as described in the section “Evaluation Metrics”. This metric can then be used to evaluate many different models that will be build in the next step and find the most performant one in terms of prediction quality.

For the actual model training part, I will try to apply hyperparameter tuning based on this evaluation metric for different algorithms in order to get the model with the highest prediction quality. Based on research in the domain of binary classification with (small) relational datasets, it seems that the Support Vector Machine (SVM) or XGBoost algorithms could be promising candidates to solve this problem (e.g. source: https://dl.acm.org/doi/abs/10.1145/2939672.2939785).

In order to fairly evaluate our trained models, we will need to split the dataset into 3 parts: a training set, a testing set and a validation set. The training set will be used to train the models and fit the model parameters by minimizing a cost function. The validation set will be used to tune the model’s hyperparameters by testing out different possible hyperparameters and selecting the ones that perform best on the validation set. It is necessary to use data that the model has not seen in model training for this to avoid overfitting. Finally, the test set will be used to make a fair final evaluation on how the selected model would perform on new, unseen data.

As a result of this workflow I will thus train different models and select the one with the best performance on the validation set. I will also be able to then give an indication for how the model performs in general based on the test set.