TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

2IOI0 PROCESS MINING 2019

INTRODUCTION

The goal of this project was to predict the remaining cycle time of a request into an administrative system. Different models were designed and tested on the different datasets provided. We focussed on the datasets '**BPI Challenge 2012**' and '**BPI Challenge 2019**'. The best performing models were a Random Forest and our Naive++. Therefore, these models were implemented and optimized for the final dataset.

METHODS

Before training our models, we performed several pre-processing methods. For the 2012 dataset, we split the events into 3 processes: 'automatically declined within an hour', 'automatically cancelled after 31 days', and 'rest'. The data was already divided into 4 categories in the 2019 dataset. We divided the 2019 data into 15 **buckets**. Each bucket contains data from the first event until a specific timestamp. For each bucket a separate model is trained. This speeds up the running time immensely. Furthermore, we have removed outliers.

END CASES

We removed all incomplete cases to avoid undesired impact on the predictions. For this, we created a **generic algorithm** that looks at the last event of each case, and counts how often these events appear. It then finds the top five most occurring ending-events in the dataset. For the 2019 dataset, we found that 5% of the cases were incomplete. The top five final events for the 2019 dataset are '`Clear Invoice`', '`Record Goods Receipt`', '`Record Invoice Receipt`', '`Delete Purchase Order Item`', and '`Create Purchase Order Item`'. For the 2012 dataset, we looked at the top three most occurring ending events. These are '`A_ACTIVATED`', '`A_DECLINED`' and '`A_CANCELLED`'.

NAIVE ESTIMATOR

For both datasets, we first created a Naive prediction model. For each event, this model predicts the remaining time as the average time between the first and last event of a case minus the time already passed. Also, negative predictions are replaced with zero.

NAIVE++ ESTIMATOR

For the 2019 dataset, we created an improved version of our Naive estimator by differentiating between the 4 different categories. We also calculated the mean per bucket. So in the end we used 60 means to predict the remaining time.

NEURAL NETWORK ESTIMATOR

We then made a Neural Network prediction model for both datasets. For this, we used the library **Keras**. The network was optimized by adding layers (see figure 1). All layers in the figure were used for the 2012 dataset. For the 2019 dataset, only the layers with red outline were used. The '2012 model' was trained with 5 epochs and the '2019 model' with 2 epochs. This model uses a **on-the-fly technique** for training, because it updates the already trained model when adding new training data. We decided to stop improving this model and concentrate on our other estimator

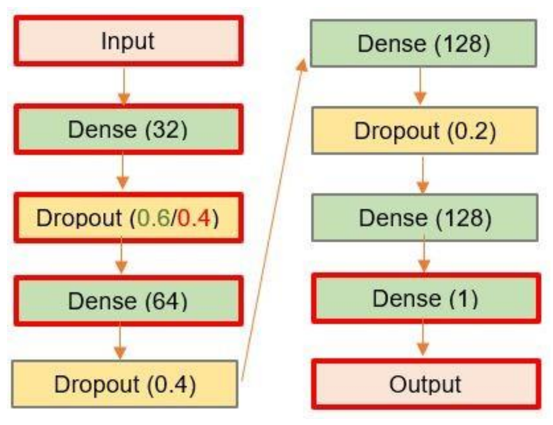

Figure 1. Layers for the Neural Network model

RANDOM FOREST ESTIMATOR

A Random Forest builds multiple decision trees and averages over them to get an accurate and stable prediction. It does this by adding randomness to the model: instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. The **most important features** for the 2019 dataset are '`event concept:name == Clear invoice`', '`event time:timestamp`', '`case Sub spend area text == Labels`', '`event Cumulative net worth (EUR)`', '`day of week`', '`day of month`', '`time spent`'. Tuning the hyperparameters can make the model faster and the prediction better. We used a **Grid Search** method to find the best hyperparameters for the 2012 dataset. For the 2019 dataset, we ran into memory issues due to the fact that we had to train a new model for each consecutive bucket. Therefore, we used less optimal parameters which still resulted in a lower RMSE.

MODEL EVALUATION

The **Root Mean Squared Error (RMSE)** is our chosen performance evaluation metric. It gives the measure of how far the predictions are from the actual output.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Formula 1. Calculation of the RMSE. y_i is the observed cycle time and $\hat{y}_i$ is the predicted cycle time for row i. n is the number of observations.

RESULTS

We have deemed the first 25% of the test data as unpredictable due to small training dataset.

2012 DATASET

Predictor	RMSE (days)	RMSE (days) no future data
Naive	8.9	
Random Forest	7.7	12.9
Neural Network	9.3	9.3

2019 DATASET

Predictor	RMSE (days)	RMSE (days) no future data
Naive	90.0	
Naive ++		32.4
Random Forest	55.7	55.4
Neural Network	876.3	496.5

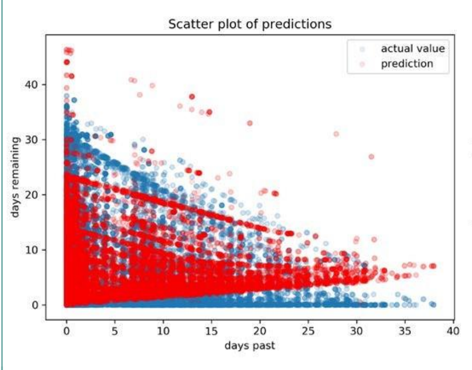

Figure 2. Scatter plot of predictions using the Neural Network estimator on the 2012 data

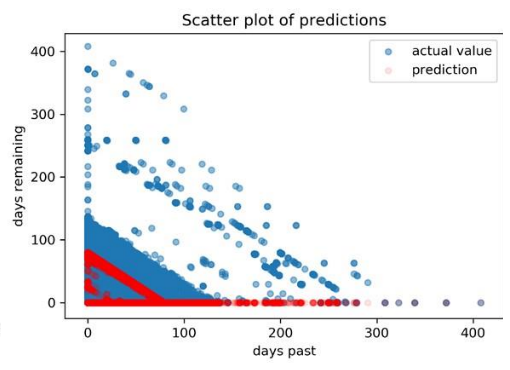

Figure 3. Scatter plot of predictions using the Naive++ estimator on the 2019 data

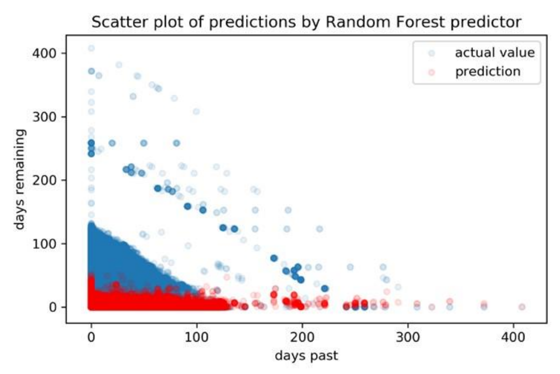

Figure 4. Scatter plot of predictions using the Random Forest estimator on the 2019 data

RESOURCE USES

We have measured memory usage (RAM, in Megabytes) and running time (in seconds) for training both predictors, and the actual predicting process. Training the Naive predictor is not that intensive: it peaks at approximately 2200 MBs of RAM and takes 60 seconds to run on the full dataset (see figure 5). Our Random Forest model, however, uses a lot of RAM: it peaks at 12000 MBs of RAM and training takes 8000 seconds, a little over two hours (see figure 6). After having trained both models, predicting for all cases and events takes 700 seconds, and peaks at around 8000 MBs of RAM (see figure 7).

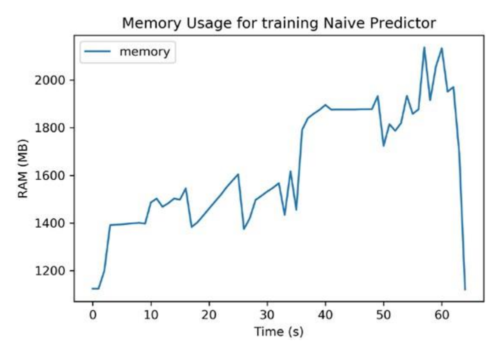

Figure 5. Memory Usage for training Naive Predictor

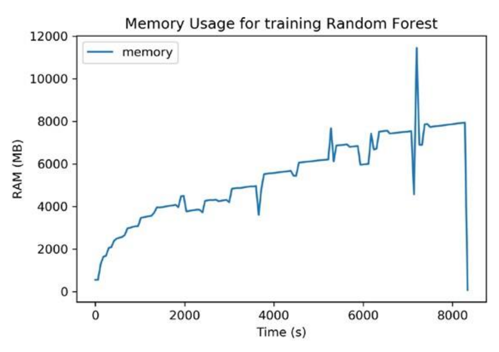

Figure 6. Memory Usage for training Random Forest

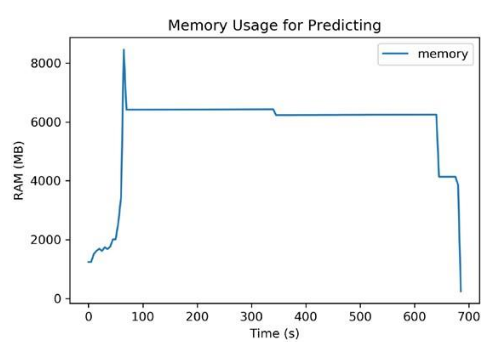

Figure 7. Memory Usage for predicting