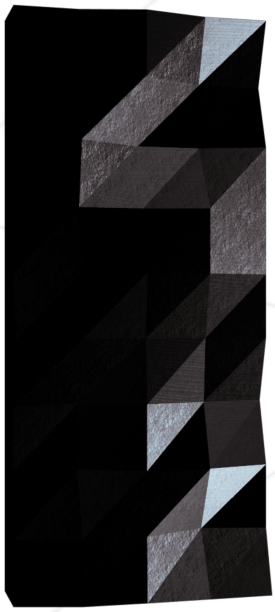


Project 01 – Churn prediction

Intro to Neural Networks. Churn prediction

Summary: This project is an introduction to artificial neural nets: fully-connected neural nets, hidden layers, activation functions, back-propagation, dropout.

Contents

Chapter I

Preamble

Do you know that your brain is a natural neural net? It consists of 90 billion neurons, on average, that are interconnected with each other creating 100-1,000 trillion synaptic connections. When one neuron is activated it transfers its signal or does the opposite – inhibits transferring the signal – to another neuron. In order to do this a combination of chemicals and electricity is used.

Some research says that there is a hierarchy between neurons in our brain. Some of them are responsible for recognizing specific basic figures. If they "see" it, they transfer the signal to the next neuron in the hierarchy. This next neuron might be responsible for recognizing "A". If it sees it, then it transfers the signal further to another neuron that might be responsible for "Apple".

This information was inspiring for people who were occupied in the artificial intelligence field and they decided to take some of the insights from human brains to create artificial neural networks.

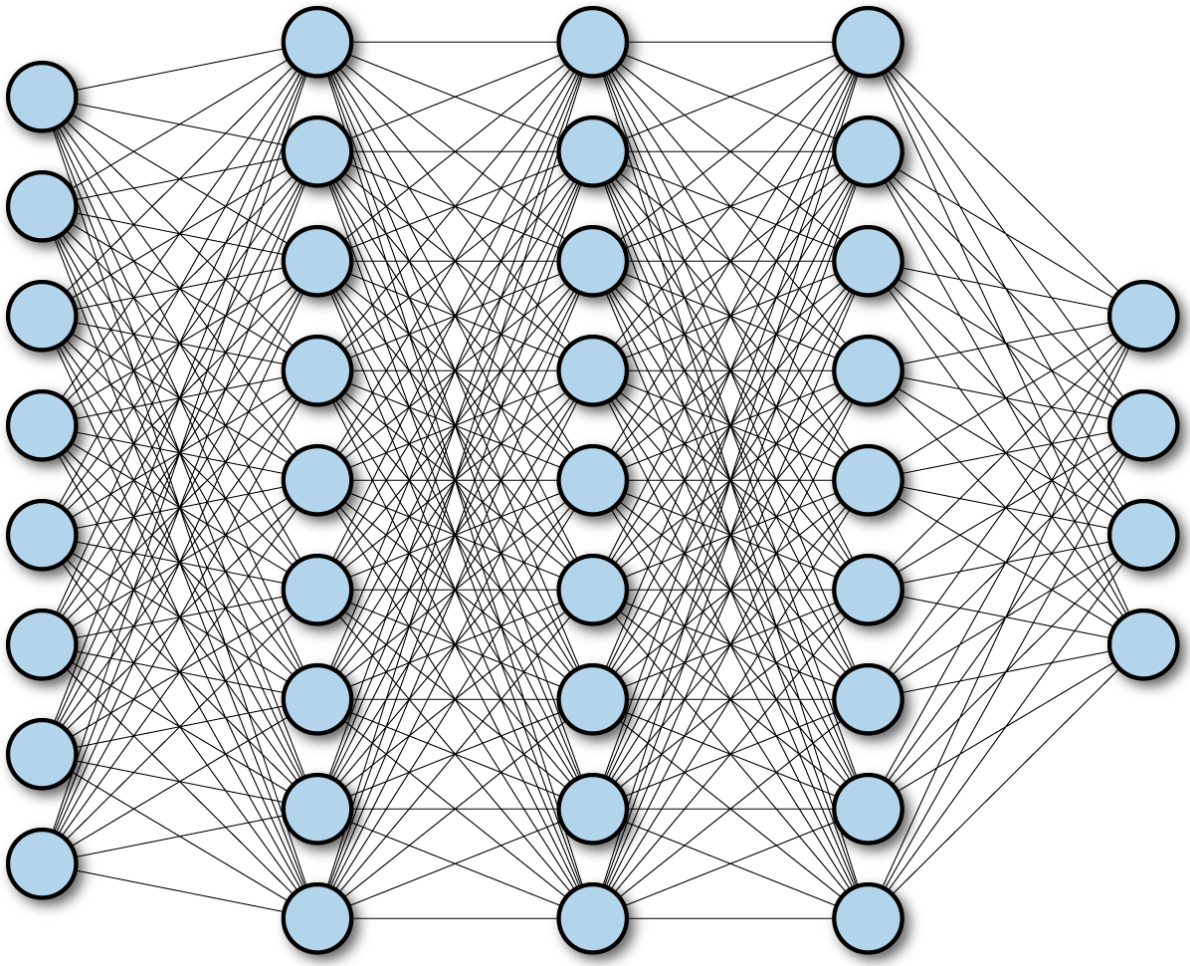
Chapter II

Introduction

Artificial neural networks are not that complex as natural ones. They are only inspired by brains. But still, they are a bit more complicated than classic machine learning algorithms.

In general, nothing changes – it is a subset of machine learning algorithms. Thus, it requires data to make predictions. It can be used for classification and regression tasks if we talk about classic machine learning tasks.

In this project, you will work only with fully-connected neural networks (FCNN). Those networks consist of neurons where each of them connected to all the other neurons in the previous and the next layer.



The first layer is called the "input layer". Each of the neurons in this layer takes the value of a feature. For example, if you have 15 features in your dataset, you will have 15 neurons in the input layer.

The next layers are called hidden layers. Each of them consists of neurons that make a non-linear transformation of the input they get. The input is the sum of the product of values and weights from the previous layer. For example, for the first neuron in the first hidden layer, you will need to multiply each feature value to a weight and then calculate the sum. The neuron from the hidden layer will pass that sum through an activation function, for example, a sigmoid (like in logistic regression) and return the value to the next layer. The neurons on the next layer will do the same thing.

The last layer is the output layer. It actually predicts something. For example, if you have a classification task where you have 4 classes, you will have 3 (n-1) neurons in the output layer.

In this example, you may think about the neural network as an ensemble full of logistic regressions. To train a FCNN means to find optimal values of weights that minimize the error.

There is the forward propagation mechanism – when you make the calculations (multiplications of weights and values, and activation functions applications and making predictions). And there is the backward propagation mechanism – when you have the predictions, you calculate the error and then adjust the weights (for example, via stochastic gradient descent) from the last layers to the first. Going back and forth, you are training a neural network.

Chapter III

Goals

The goal of this project is to give you a first approach to neural networks. You will try to train a multilayer perceptron (FCNN) using several libraries as well as making the same network using NumPy.

Chapter IV

Instructions

- This project will only be evaluated by humans. You are free to organize and name your files as you desire.
- Here and further we use Python 3 as the only correct version of Python.
- The norm is not applied to this project. Nevertheless, you are asked to be clear and structured in the conception of your source code.
- Store the datasets in the subfolder data

Chapter V

Mandatory part

a. Task

In this project, you will work on a churn prediction. You will need to predict which customers are going to stop being customers of the bank. You will need to use a multilayer perceptron for your final prediction.

- Baseline. Naive classifier where you use the most popular class for predictions.
- Random forest. Solve the task with the random forest as another baseline solution, use grid search to find optimal hyperparameters.
- Scikit-learn. Solve the task using MLPClassifier.
- Keras. Solve the task using Keras from the TensorFlow library.
- TensorFlow. Solve the task using the TensorFlow library.
- NumPy. Implement the best architecture that you achieved earlier but with NumPy using matrix calculations. You need to train the model and use it for inference (predictions).

b. Dataset

You will work with the dataset of one of the Russian banks. It contains different data about their customers: financial information, their age, services that they used, and the target – whether they left the bank in the next three months. There are two files: train and test. You will use the train data to fit the models and make predictions for the test dataset.

You can find the dataset in the project page : "p01_bank_data.zip"

The description of the fields:

Variable:	Description:
AGE	Age (months)
AMOUNT_RUB_ATM_PRC	The fraction of transactions with MCC to the whole amount of the transactions in the period (rub)
AMOUNT_RUB_CLO_PRC	
AMOUNT_RUB_NAS_PRC	
AMOUNT_RUB_SUP_PRC	
APP_CAR	Ownership of car
APP_COMP_TYPE	Type of employer
APP_DRIVING_LICENSE	Driving license
APP_EDUCATION	Education
APP_EMP_TYPE	Type of occupation
APP_KIND_OF_PROP_HABITATION	Type of habitation property
APP_MARITAL_STATUS	Marital status
APP_POSITION_TYPE	Position type
APP_REGISTR_RGN_CODE	Code region
APP_TRAVEL_PASS	International passport
AVG_PCT_DEBT_TO_DEAL_AMT	Average percentage of debt to deal amount (average annuity)
AVG_PCT_MONTH_TO_PCLOSE	Average percentage of the credit term left
CLNT_JOB_POSITION	Job position
CLNT_JOB_POSITION_TYPE	Job position type
CLNT_SALARY_VALUE	Salary
CLNT_SETUP_TENOR	Months of being a customer
CLNT_TRUST_RELATION	Trust relation
CNT_ACCEPTS_MTP	Number of accepts in different campaign channels
CNT_ACCEPTS_TK	
CNT_TRAN_ATM_TENDENCY1M	Trend of transactions number by the MCC type (1 and 3 months)
CNT_TRAN_ATM_TENDENCY3M	
CNT_TRAN_AUT_TENDENCY1M	
CNT_TRAN_AUT_TENDENCY3M	
CNT_TRAN_CLO_TENDENCY1M	
CNT_TRAN_CLO_TENDENCY3M	
CNT_TRAN_MED_TENDENCY1M	
CNT_TRAN_MED_TENDENCY3M	
CNT_TRAN_SUP_TENDENCY1M	
CNT_TRAN_SUP_TENDENCY3M	

CR_PROD_CNT_CC	Number of product used in the period (by the category products)
CR_PROD_CNT_CCFP	
CR_PROD_CNT_IL	
CR_PROD_CNT_PIL	
CR_PROD_CNT_TOVR	
CR_PROD_CNT_VCU	
DEAL_GRACE_DAYS_ACC_AVG	Grace metrics
DEAL_GRACE_DAYS_ACC_MAX	
DEAL_GRACE_DAYS_ACC_S1X1	
DEAL_YQZ_IR_MAX	Max and min interest rate for revolvers and annuities
DEAL_YQZ_IR_MIN	
DEAL_YWZ_IR_MAX	
DEAL_YWZ_IR_MIN	
ID:	Unique ID:
LDEAL_ACT_DAYS_ACC_PCT_AVG	Metrics of activity in the period (credit contracts)
LDEAL_ACT_DAYS_PCT_AAVG	
LDEAL_ACT_DAYS_PCT_CURR	
LDEAL_ACT_DAYS_PCT_TR	
LDEAL_ACT_DAYS_PCT_TR3	
LDEAL_ACT_DAYS_PCT_TR4	
LDEAL_AMT_MONTH	Other product metrics in the period (credit contracts)
LDEAL_DELINQ_PER_MAXYQZ	
LDEAL_DELINQ_PER_MAXYWZ	
LDEAL_GRACE_DAYS_PCT_MED	
LDEAL_TENOR_MAX	
LDEAL_TENOR_MIN	
LDEAL_USED_AMT_AVG_YQZ	
LDEAL_USED_AMT_AVG_YWZ	
LDEAL_YQZ_CHRG	
LDEAL_YQZ_COM	
LDEAL_YQZ_PC	
MAX_PCLOSE_DATE	Number of months until planned credit close date (max with annuities)
MED_DEBT_PRC_YQZ	Median of debt percentage for annuities and revolvers
MED_DEBT_PRC_YWZ	
PACK	Service package

PRC_ACCEPTS_A_AMOBILE PRC_ACCEPTS_A_ATM	% of accepts in channels / product groups
PRC_ACCEPTS_A_EMAIL_LINK	
PRC_ACCEPTS_A_MTP	
PRC_ACCEPTS_A_POS	
PRC_ACCEPTS_A_TK	
PRC_ACCEPTS_MTP	
PRC_ACCEPTS_TK	
REST_AVG_CUR	Average current account balances
REST_AVG_PAYM	Average salary account balances
REST_DYNAMIC_CC_1M	Trend of monthly average account balances per products (1 or 3 months)
REST_DYNAMIC_CC_3M	
REST_DYNAMIC_CUR_1M	
REST_DYNAMIC_CUR_3M	
REST_DYNAMIC_FDEP_1M	
REST_DYNAMIC_FDEP_3M	
REST_DYNAMIC_IL_1M	
REST_DYNAMIC_IL_3M	
REST_DYNAMIC_PAYM_1M	
REST_DYNAMIC_PAYM_3M	
REST_DYNAMIC_SAVE_3M	
SUM_TRAN_ATM_TENDENCY1M	Trend of transactions amount per MCC (1 months and 3 months)
SUM_TRAN_ATM_TENDENCY3M	
SUM_TRAN_AUT_TENDENCY1M	
SUM_TRAN_AUT_TENDENCY3M	
SUM_TRAN_CLO_TENDENCY1M	
SUM_TRAN_CLO_TENDENCY3M	
SUM_TRAN_MED_TENDENCY1M	
SUM_TRAN_MED_TENDENCY3M	
SUM_TRAN_SUP_TENDENCY1M	
SUM_TRAN_SUP_TENDENCY3M	
TARGET:	Actual churn in the next 3 months:
TRANS_AMOUNT_TENDENCY3M	Ratio between transaction sum in the last 3 months to the last 6 months
TRANS_CNT_TENDENCY3M	Ratio between transaction number in the last 3 months to the last 6 months
TRANS_COUNT_ATM_PRC	Ratio of MCC transactions to the all transactions in the period
TRANS_COUNT_NAS_PRC	
TRANS_COUNT_SUP_PRC	
TURNOVER_CC	Average turnover in credit cards

TURNOVER_DYNAMIC_CC_1M	Trend of monthly average turnovers in the period (1 or 3 months)
TURNOVER_DYNAMIC_CC_3M	
TURNOVER_DYNAMIC_CUR_1M	
TURNOVER_DYNAMIC_CUR_3M	
TURNOVER_DYNAMIC_IL_1M	
TURNOVER_DYNAMIC_IL_3M	
TURNOVER_DYNAMIC_PAYM_1M	
TURNOVER_DYNAMIC_PAYM_3M	
TURNOVER_PAYM	Average turnover of salary accounts

c. Implementation

You can work in Jupyter Notebooks. The notebooks should be well-formatted. You need to make a split on the train and test (20%) datasets with stratification. You may apply any preprocessing for the data: working with anomalies, missing values, feature generation, and selection. Use a grid search to find the best hyperparameters.

In the last part of the task when you will implement your neural network, please, use OOP principles.

At the end of your notebook(s), you need to create a table with the results of your research where you should display the name of the library, algorithms, hyperparameters, and the score (accuracy and AUC) of the models you used (including baseline solutions). Try to use dropout for model regularization.

d. Submission

After you finish working with the models, you need to save the final predictions in the CSV-file with only two fields: “ID” and “TARGET”. The order of IDs should be the same as in the test dataset you were given. The values of “TARGET” can be either the class or the probability.

You need to achieve AUC at least equal 0.8183 on the test dataset with a neural network solution. It will be calculated with an automated checker.

Your repository should contain one or several notebooks with your solutions as well as the prediction file.

Chapter VI

Bonus part

- Try to achieve a better AUC on the test dataset with a neural network solution – 0.83.
- Try to achieve an even better AUC on the test dataset with a neural network solution – 0.85.

Chapter VII

Submission and peer-correction

Submit your work on your Git repository as usual. Only the work on your repository will be graded.

Here are the points that your peer-corrector will have to check:

- there are baseline solutions,
- there are all 4 required implementations of neural networks,
- the score achieved on the test dataset.