Machine Learning Engineer Nanodegree

Capstone Project Proposal (Nov 2017)

Jorge Pinzon

PROBLEM LOAN PREDICTION

Problem loans and its implications to the financial sector

It is not a secret that banks, and other financial institutions, make their profits by lending money (www.investopedia.com). Every time a bank lends money is taking a risk. The risk involves the fact that some borrowers will not return the money as agreed with the institution during the lending process. This situation originates “Problem loans,” or loans where the banks may lose money as they cannot recover the original amount, or by investing more in the repaying process (www.iedunote.com). According to the US Department of Treasury, banks can deal with problem loans in different ways, including: simple renewal or extension of the loan terms, extension/addition of credit, loan restructuration, or foreclosure. In this process known as loan workout the bank is forced to choose an alternative where recovery is maximized and risk and expenses are minimized (www.occ.treas.gov). In this background, why not using machine learning techniques on historical data to predict whether or not a loan will become a problem loan.

Problem Statement

Banks and financial institutions should use machine learning techniques during the loan application process to determine whether or not the potential borrowers will pay on time the loan. In this project, I propose to develop a neural network to predict such outcome. Currently publically available project on the topic have an accuracy of 20%, but have failed to incorporate some more modern techniques that may improve accuracy of the models. I am planning to use neural networks to predict a binary classifier, that can be measure be determining the accuracy in the predictions made on a testing subset of the data or through cross validation.

Datasets and Inputs

The data for this project comes from Lending Club, a peer-to-peer lending company from San Francisco, California. The company releases its statistics on loan applications and status, as well as on applications that were rejected, regularly. The most recent release corresponds to the second quarter of 2017. For this project, I will use data from 2017, first and second quarters, which has information on 139,311 loans. The data can be accessed on https://www.lendingclub.com/info/download-data.action. There are other compilations of other years in available in kaggle and data.world, but as far as U know, 2017 is only in the Lending Club website.

The 2017 data constitutes two separate files, one for each quarter. Each file has 145 columns with the same headers making a merge easy. Of the columns, at least 92 have more than 90% of the cells filled with non-values (nan) rendering these not useful for the model. The other 53 columns have more potential as they include information on the amount of the loan, income of the customer, use of the loan, previous defaults, whether or not the customer has a mortgage, and also the loan status. In the model loan status will be the target feature and a more complete selection of the other 52 columns will results in enough feature to build a predictive model.

Both data sets, a csv file with a preliminary cleaning, and the schema of the data can be found in the github (https://github.com/jpinzonc/Problem_loan_prediction).

Solution Statement

I plan to build a neural network classifier, this can be with an Artificial Neural Network or a Recursive Neural Network classifier. I am leaning towards the RNN classifier by using the date-based data to create a sequence, possible by adding a new feature that includes length of the loan, or possible the term of the loan. Similar to the RNN used to predict movie preference in https://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/, and that used by Ando et al., 2016 to predict credit fraud. I will divide the data into subset of training, testing, and validation to measure the reliability of the model and accuracy.

Benchmark Model

As shown in table 1, there are a few models that have use data from the Lending Club at different intervals of time and using different models. Most of the neural network attempts are from a kaggle dataset that included data from 2007 to 2015, in all cases they do no report accuracy. The only project that report a measure of accuracy is the R-based attempt, in which the author of the tutorial tried several machine learning models including: Logistic regression, SVM, xgb, but all with very low accuracy levels. None of the neural network models include RNN on it. Ando et al., 2016 used RNN to predict credit fraud and use it with and without LSTM and compare the results with SVM models. The RNN without LSTM resulted in the best performance for that particular dataset as it provided an F-score of 1. For this project, I can use one of the kaggle kernels to compare my model performance.

Table 1. Different projects building models for problem loan predictions publically available. These models the same source for their data (Lending Club), but different years.

No.	Language	Library	Website	Model	Years/Notes
1	Python	Tensorflow	https://www.kaggle.com/vigilanf/building-a-neural-net-to-predict-default	DNN Regressor	2007-2015
2	Python		https://www.kaggle.com/jlrsource/predicting-loan-status-with-python	Decision Tree Classifier	2007-2015
3	R		https://www.kaggle.com/anuragpatil94/artificial-neural-network-ann-on-loandata	ANN	2007-2015
4	Python	Keras	https://www.kaggle.com/gagrawal/neural-net-with-keras	ANN	2007-2015
5	Python	Tensorflow	https://www.kaggle.com/mina20/building-a-neural-net-to-predict-default		2007-2015
6	Python	Tensorflow	https://www.kaggle.com/johnnogent/building-a-neural-net-to-predict-default	DNN Regressor	2007-2015
7	Python	Tensorflow	https://www.kaggle.com/forrestlmj/building-a-neural-net-to-predict-default	DNN Regressor	2007-2015
8	Python	Tensorflow	https://www.kaggle.com/tobitog/building-a-neural-net-to-predict-default	DNN Regressor	2007-2015
9	Python	Keras	https://www.kaggle.com/gagrawal/neural-net-with-keras	ANN	2007-2015
10	Python	Tensorflow	https://www.kaggle.com/mloflee/building-a-neural-net-to-predict-default	DNN Regressor	2007-2015
11	R		https://rstudio-pubs-static.s3.amazonaws.com/203258_d20c1a34bc094151a0a1e4f4180c5f6f.html	ML Algorithms (logistic reg, SVM, xgb, etc)	
12	Python		https://www.dataquest.io/blog/machine-learning-preparing-data/		Data Exploration

Evaluation Metrics

For the metrics in the model I should use the F-score on predictions from the test dataset, the main reason for this is to make a comparison with Ando et al., 2016 possible. The F-score provides a balance between precision and recall, and therefore complements these two measures, and reduces the possibility of ending in an accuracy paradox when evaluating the model.

When comparing with the kernel from kaggle, it will be necessary to either include the model's performance measurement but also modify to code to use the F-score too.

Project Design

This project will have four steps: data mining, data cleaning, model building, and model comparisons.

Data mining. Already done. Corresponded to the actual search and downloading of the data. It was performed manually, for initial exploration purposes. This step allowed to examine the possibilities with the project and the scope of the proposed model. The files were already loaded into Python, using pandas, and merged into a single data frame that was save as a csv file (loans_2017.csv).

Data cleaning. The data contains information 139,311 loans. The information includes 145 different columns with different levels of useful information. Initial exploration indicates the presence of rows that are not loans, but descriptors of the dataset, there are also several columns with multiple NaN, some do not have any data at all. These columns need to be filter out. The data has been cleaned up to this point.

Once the previous cleaning steps are done, it will be necessary to assess the relevance of each resulting feature for the model. For example, ‘emp_title’ is the title reported by the customer during the application. This column may not be as relevant as ‘annualinc’ which contains the annual income of the applicant.

After filtering out non-relevant columns, there may be the need to create new features. For example: ‘inqLast6Mths’ and ‘inq_last_12m’ can be combine into one feature. And there is a possibility for more combinations.

Then, non-numerical features should be coded with numbers and numerical features should be normalize across the dataset. These two steps will allow the model to be built as it may not take non-numerical values and prevent minimum local optima problems and increase the speed of the analysis.

Finally, the data is split into training, test, and validation subsets. This step will allow to have independent sets for training, testing and validating the model.

Model building. I will initially start with the construction of a model with three layers, an input, a hidden and an output layer. Train and evaluate the model performance.

Then based on the results of the initial training, I will optimize the model by either, modifying/creating/removing features, increasing hidden layers, and or changing parameters in the algorithms (e.g. activation, compilation, dropout).

Model comparisons. Finally, once the model is built, I will compare the results with those in the benchmark models. Using the F-score or accuracy depending on the comparison model.