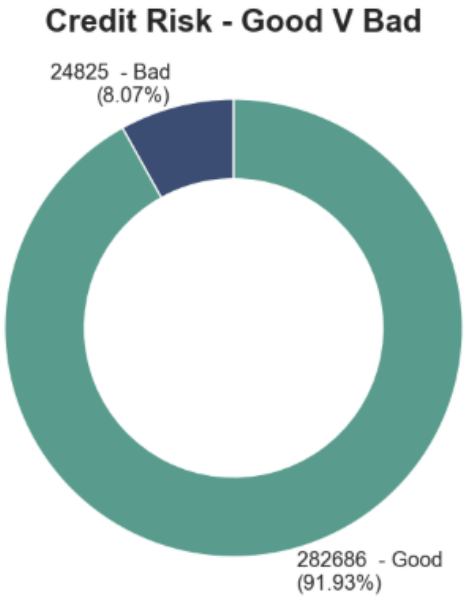
A Home Credit Risk Analysis & Broad Application of CRISP-DM

By: Geoffrey Nel



Contents

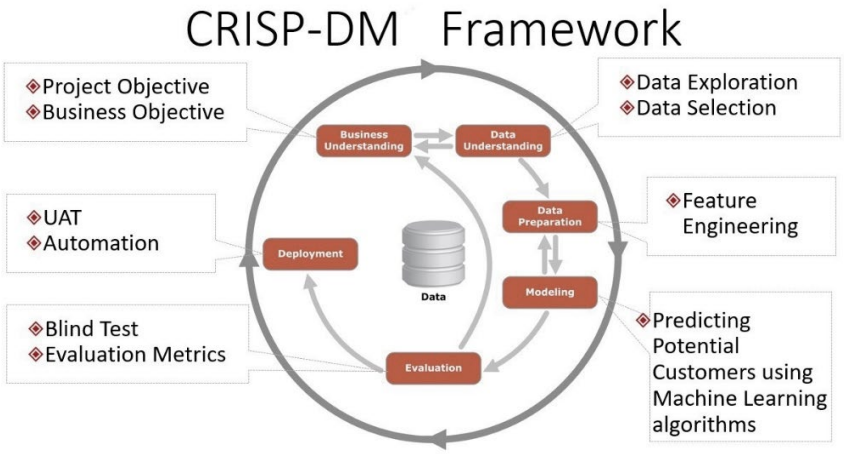
Introduction

The intention for this analysis is to better understand how CRISP-DM can be used in a credit risk modelling context. It is also to better understand credit risk analysis and how two seemingly disparate models, like LightGBM and a simple Artificial Neural Network, will compare when presented with the same consumer credit data.

CRISP-DM

Business Objective

CRISP-DM is a well-known data analytics framework. It is best used in mapping out data-driven business solutions where the core of the framework is centered around a business objective. In our case, we want to answer business concerns surrounding consumer credit risk; moreover, what attributes should we be most concerned with when determining a consumer's credit risk. If we are in the business of processing loans, then modeling credit risk is at the core of the business. The more capable a model is at discerning an applicant's credit risk, then there is greater benefit for the company in terms of both increasing revenue and cutting losses.



Understanding The Data

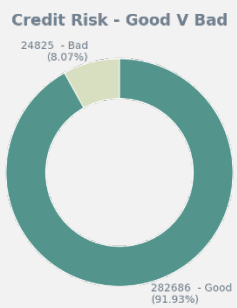
As stated above, the business objective is the first step within the CRISP-DM framework. The next step is in understanding the data. Since the eventual output results of the model are hinged on the type of data that can obtained, if there are issues with the data, like sparsity,

inaccuracies, or generally just inconsistent data, then the analysis will follow suit. If we can obtain well maintained and accurate data, where it also the case that the data is understandable through the eyes of either a subject matter expert or a statistician, then the analysis is much more likely to yield good results.

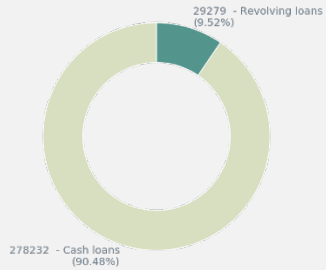
Looking at the pie plot illustrations on the subsequent page, we can see feature examples extracted from data that seems to be well suited for credit risk analysis. To start, there exists an historical record of training data with previously logged good and bad—low or high—risk consumers. It is noted that a bad consumer is known to have either defaulted on loan, had 120 or more days of unpaid loans, and or had a charge-off. Other fields include the types of loans, be it a revolving loan, a cash loan, if the borrower owns a vehicle, if they own equity in real estate or one of the many other features available in this dataset.

Feature Review

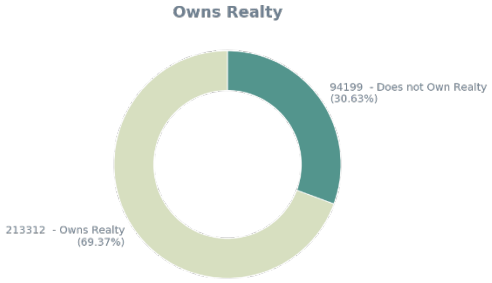
Credit Risk



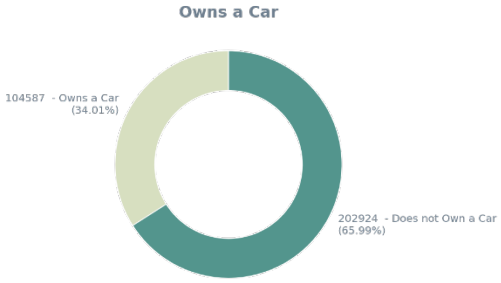
Loan Types



Realty



Auto

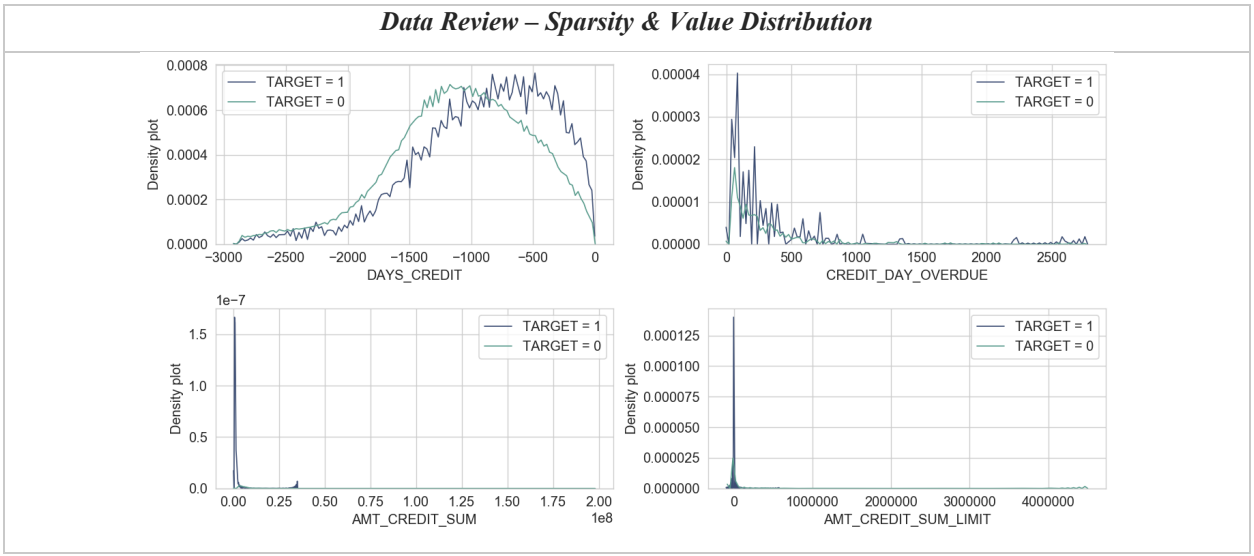


Description: Above we can see four features. The response variable, being the credit risk, good and bad; then the types of loans, if the person owns realty, and if they own a car.

Data Preparation

Next, CRISP-DM acknowledges the importance of data preparation. Often times data is not acquired in a way that it is simply plug-and-play. Data munging or cleaning is typically required, and it is a large portion of the data analysis process. It is not uncommon to spend more than 70-90% of the project time on this phase. As noted in the second step, just as a lack of data or incorrect data can be an issue, data that is laden with governance or quality issues, can be just as bad.

Given the credit risk project case, we can see that the chosen dataset contains varied data. The data is generally good and know the source from which it came, but we do also see signs of sparsity and many zero values within some of the data fields. This is noted in the amount of credit sum and amount credit sum limit variables below. These zero value counts are acceptable if there are no underlying data quality issues, although due to their lack of variance, it is unlikely that they will provide considerable predictive power when modelling. It may be required to consult a SME on how best to impute missing or zero-type values if it is also determined that the variable is of high importance.



Modelling

The subsequent step after data preparation is modelling. Here note that there is a vast degree of models to choose from. Additionally, there are countless ways to implement these

models. As an example we find that there are simple naive Bayes and regression models and there are complex deep learning or even Bayesian optimization models; there are simple single layer neural networks and there are multilayer CNN and RNN based neural networks; there are old and simple association ruled-based models that can help determine which items a particular person might buy in tandem with another item at the grocery store, and there are novel and complex general adversarial networks that can help create iterate design topologies for organic and structurally integral engineering components.

In the case of credit risk, the two models that we have chosen to use are one, a densely constructed artificial neural network and a gradient boosted model. The architecture for the neural network is in the immediate illustration below.

Neural Net Model Architecture

```
input_num = Input(shape=(21,), dtype='float32')
outputs = Sequential()#([*model_out, input_num])
outputs = Concatenate(axis=1)([*model_out, input_num])

#outputs = Sequential()(outputs)
outputs = (Dense(512))(outputs)
outputs = (Activation('relu'))(outputs)
outputs = (Dropout(.35))(outputs)
outputs = (Dense(256))(outputs)
outputs = (Activation('relu'))(outputs)
outputs = (Dropout(.15))(outputs)
outputs = (Dense(128))(outputs)
outputs = (Activation('relu'))(outputs)
outputs = (Dropout(.15))(outputs)
outputs = (Dense(1))(outputs)
outputs = (Activation('sigmoid'))(outputs)

model = Model([*model_in, input_num], outputs)

model.compile(loss='binary_crossentropy', optimizer='adam')#,
```

Description:

The neural net model was built using three main dense layers, each with a dropout and rectified linear activation. The final layer is the sigmoid prediction output.

Continuing the topic of modelling, one concern in developing a successful and stable model that can stand a production environment, is how the model is developed. To briefly touch on this, given our project case, credit risk, we have maintained a typical 80/20 train and test split. We have decided not to continue through with final testing set, although this is normally an important step in finalizing the modelling process.

It is often the case that the model either undergoes a cross fold validation process and or a train-validation-test process. One consideration in how to proceed in this aspect is the dataset

size. Given that our data is roughly 30,000 datapoints, there is enough data and a higher degree of variance within each variable where we should expect for find independent variables with a sufficient degree of predictive power. If it turns out that a model only has 1000 or even a 100 data points, with far fewer than 189 features, then this would be a major concern. In the following illustration we can more clearly see the data used in this analysis.

Train and Test Split – Dimensions and Percent	
X-Train	***X-Test***
Dim: (246008, 189); Perc: 80.0 %	*Dim: (61503, 189); Perc: 20.0 %*
Y-Train	***Y-Test***
Dim: (246008,1); Perc: 80.0 %	*Dim: (61503,1); Perc: 20.0 %*

One last consideration that we will discuss in terms of modeling. When constructing a neural network for language processing, it is understood that the word embeddings are an integral part of the process. Differing embeddings, or the lack thereof, can drastically impact the results. A similar thought process holds in terms of categorical features within a tabular style dataset.

For this project, an encoding, followed by an embedding layer is used to 'encode' the categorical values in the data. These categorical values are given a particular numeric representation that can then be parsed into the model.

The presumption is, insofar as embeddings are concerned, that similar categorical features aren't just represented by a random number, but are given numerically representative values that place them within close proximity, or a similarly calculated path, so that similar categories are also numerically similar. This becomes clearer when given a multidimensional space for which all the categories are mapped to. Given the credit risk data, a reasonably good embedding would map similar occupations close to each other, while diametrically opposed occupations would lie in opposite directions.

It can be noted that there are models like CatBoost that have inherent capabilities with categorical data, although, when working with raw neural networks, it is common practice to

work through preliminary embedding transformations prior to passing the data into the model. Both the embedding and the encoding code is noted below.

Encoding Categorical Features

```
# Generator To Parse Cat
generator = (c for c in X_train.columns if X_train[c].dtype == object)

# Label Encoder
for c in generator:
    lbl = LabelEncoder()
    lbl.fit(list(X_train[c].values))  #+ list(X_test[c].values))
    X_train[c] = lbl.transform(list(X_train[c].values))
```

```
cuted in 3.86s, finished 06:29:02 2020-06-07
```

```
# Setting Cat Embedding
embed_cols = []
len_embed_cols = []
for c in col_vals_dict:
    if len(col_vals_dict[c]) > 2:
        embed_cols.append(c)
        len_embed_cols.append(len(col_vals_dict[c]))
        print(c + ': %d values' % len(col_vals_dict[c])
              )  #Look at value counts to know the embedding dimensions
```

Categorical Variable Examples:

CODE_GENDER: 3 values

OCCUPATION_TYPE: 19 values

ORGANIZATION_TYPE: 58 values

HOUSETYPE_MODE: 4 values

WALLSMATERIAL_MODE: 8 values

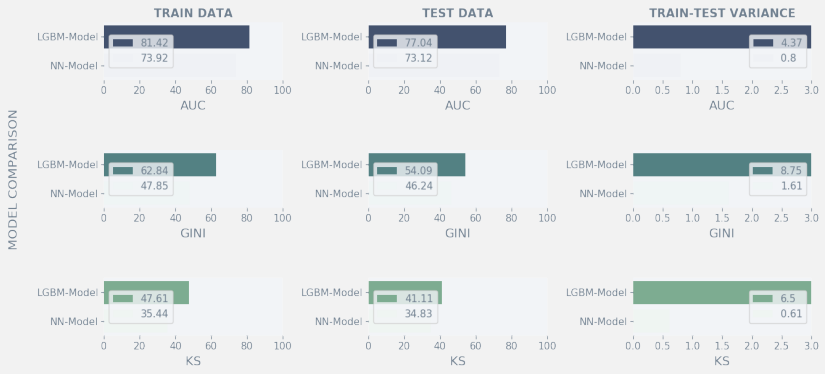
EMERGENCYSTATE_MODE: 3 values

Evaluation

Next, we will move to the evaluation process. We will be reviewing several pages of output from our model, examining differences between the models that we have used, as well as the difference between the same model when using different input data.

Primary Model Metrics – Comparisons

Models NN1 & Lgbm1



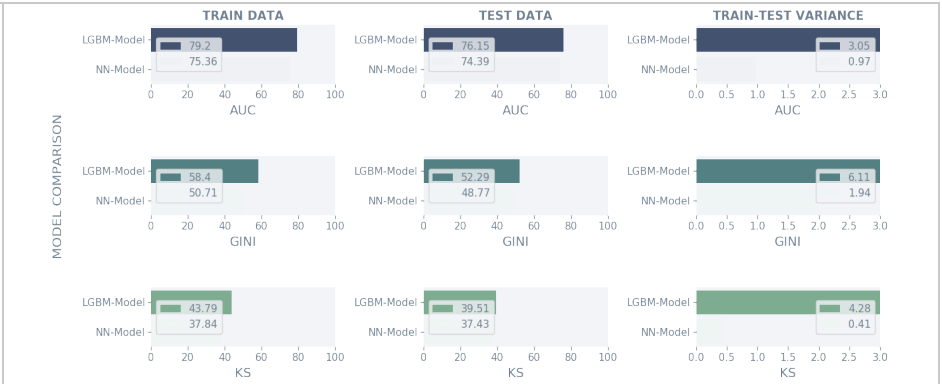
Description:

Left & right, we comparison of both initial models and their resulting AUC, GINI, and KS scores.

Description:

Left & right, we see three columns for evaluating the model; on the train

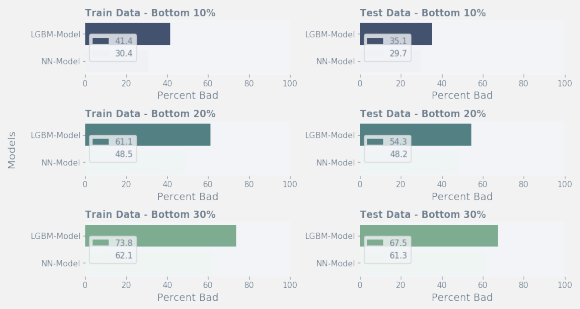
Models NN2 & Lgbm2

data, the test data, and the variance between them.	

Looking at the model output above, we see in the top illustration that LightGBM is the top model for all three metrics, AUC, GINI, and KS, leading to the assumption that this is the better model. In looking deeper and in particular, the third column, the variance between the train and the test model, there is a discernably high variance. When creating these measurements, we made the initial assumption that variance would remain below 3%. Looking at the numbers above, we see that the LightGBM variance that exceeds this.

Bottom Rank Order Bad Percent – Model Comparisons

Models NN1 & Lgbm1



Models NN2 & Lgbm2

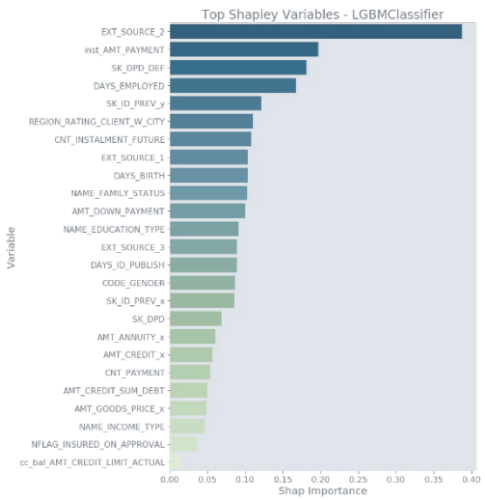
Models NN1 & Lgbm1

For the primary models, noted by NN1 and Lgbm1, these models received all variables, amounting to a total of 189 input variables. In the case of the second models, noted by NN2 and Lgbm2, the intention is was to withdraw some of the potential overfitting issues, while observing the general differences in how this would effect the models. For this, the variables were narrow down to the top 21 numeric variables and top 4 categorical variables. These feature selections are illustrated by the feature importance and the correlation matric below.

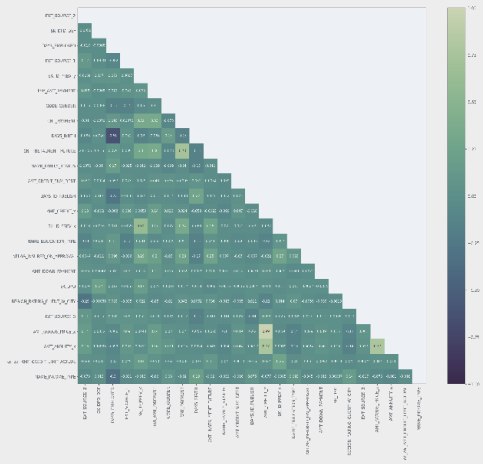
With this, we can see that the variance for the Lgbm2 model drops by a few percentage points across all metrics, where there is also a slight degradation in the overall metrics. One insight that was discovered here, is that the opposite is true for the neural network. It appears to increase in variance, yet all metrics seem not to decrease, but increase up to 3 percentage points.

Feature Review

Top Features by Importance



Correlation Matrix – Top Features



Description:

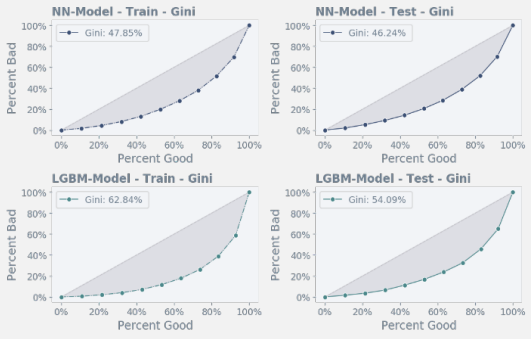
Above we can see a correlation plot of the top features as discerned by a Shapley feature importance analysis. This correlation plot is on the top left, while the feature importance is listed on the right in descending order.

The same trend continues throughout our analysis. We can clearly see that through all sets of metrics, even when review the rank ordering, that the neural network is benefitting from the narrowed feature set, where this same feature set is diminishing the LightGBM results. If we look at the KS between both the NN2 and Lgbm2 model, which is visible in the left most bottom plot of illustrations below, we see that the two models are coming together on this metric.

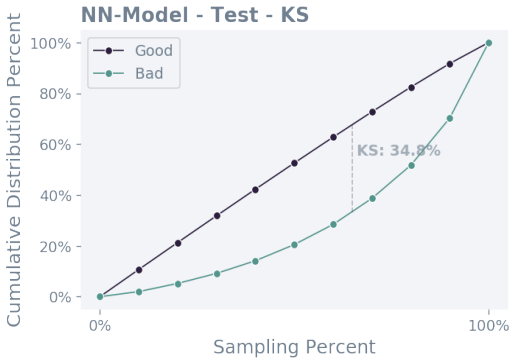
The reason why the Kolmogorov-Smirnov statistic is used is because it is a key metric in determining a models ability to separate good and bad consumer credit profiles; the greater the vertical distance between the good and the bad curves, the better the model is at discerning between these populations.

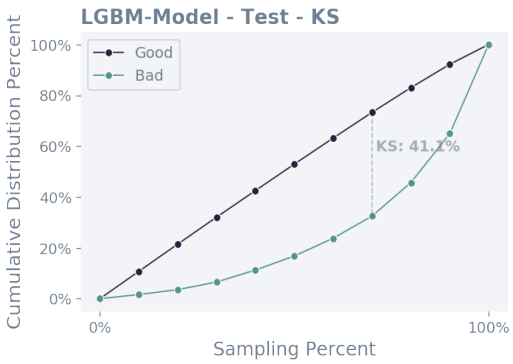
Model Rank Gini and KS – Comparisons

Models NN1 & Lgbm1

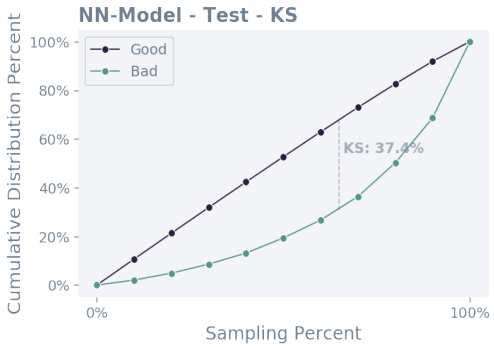


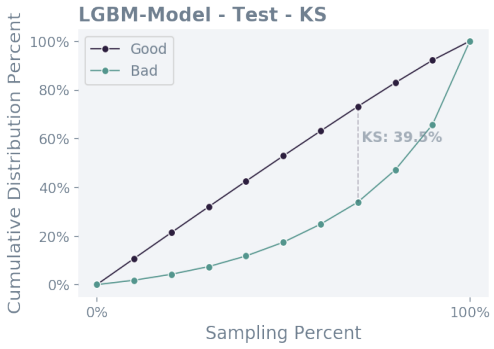
Models NN1 & Lgbm1



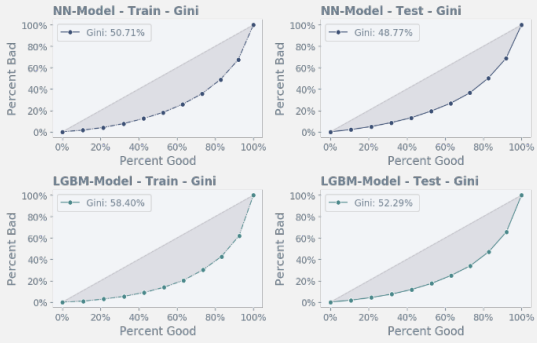


Models NN2 & Lgbm2





Models NN2 & Lgbm2



The last and final phase of the CRISP-DM process that needs to be discussed is that of deployment. In our case, a deployed credit risk model would essentially meet this requirement, and according to the CRISP-DM framework, this is the final step, however, certain models are further monitored and evaluated over the course of the months and years following deployment.

It is feasible to cycle in updates and changes, yet as far as a project phase is concerned, the deal is closed after deployment.

Challenges

This project has included several challenges, beginning with programming. In the same regard that data preparation can be time consuming within the CRISP-DM framework, programming the data inputs for a machine learning model, and most notably neural network models—due to their scaling and input data requirements—can consume a considerable amount of time. A majority of the analysis process was concerned with building a successful neural network pipeline.

A subsequent challenge was in finding a neural network that would ‘bite’ on the data. Out of the five neural networks that were assessed, two showed no results, two showed poor results, and then there is the one that was presented in this analysis.

Concerns

A major concern is on the target variable imbalance. There are certain techniques available when running predictions on unbalanced datasets, like the use of SMOTE in balancing the dataset prior to inputting the data. This analysis was done as is where the balance was combated by choice of metrics, like area under the curve, which will maintain consistency in imbalanced datasets.

Insights

In general, we see that both LightGBM and a variation of an artificial neural network, can work in credit risk prediction modelling. Research shows that gradient boosting models are already in use throughout the credit risk industry, however, neural networks are seemingly scarce. One of the major issues with neural networks is the monotonic feature explainability, where along with strict regulations and outright performance results, the neural network models may not be ready for plug-and-play implementation.

The most pertinent features discovered in this analysis are, SK_DPD_DEF, which is days past due, EXT_SOURCE_2, which is an unknown external metric, and AMT_PAYMENT, or the previous credit installment payment.

Regarding the business implementation or the incorporation of moving forward with these models, we can firmly state that since both models performed well, they both would be candidates for moving forward. There would be additional work needed regarding refinement and feature interpretability, yet overall feasibility and performance aspects have been met through this analysis.