DATA DRIVEN BUSINESS DEVELOPMENT - LOAN DATA

EIRIK DUESUND HELLAND

Michelle Wiggins | https://www.kwikcashonline.com/quick-online-loans/ | https://creativecommons.org/licenses/by-sa/4.0

EXECUTIVE SUMMARY

A 35-year-old man arrives at a bank downtown. He lines up in the queue stretching through the reception and sighs at the fading opportunity of a swift visit when he sees the slow banker. She is counting the coins of an old lady at the counter. He thinks of his wife and children waiting at home and wishes that this process consumed less time and was not so nerve-wracking. He is there to apply for a loan.

Almost everyone must apply for a loan at some point. Billions of applications have been processed manually until now, consuming lots of time and resources. Is there a way to free up these resources, streamline the application process and create a better experience for the applicants?

This project has created a model for predicting the loan status of loan applications, using loan data. This model has been integrated into a user interface using streamlit to create a user-friendly experience, which makes the loan application process much faster.

The project demonstrates the automation of the loan application process and the added business value it provides. To implement this project in real-life-scenarios, the model is dependent on high quantities of quality data. If this is provided, then the time of the manual processing of loan applications might come to an end.

LIST OF FIGURES

CONTENT

INTRODUCTION

BACKGROUND

This paper is written as a study for the course TIN200 at NMBU. The idea and data material were provided as course material.

Data driven business development is currently a hot topic. The advances in programming and artificial intelligence have pushed us onto another step of the automation process that the industrial revolution started. We are now automating the previously labour-intensive task in banking, known as the loan application process.

This paper will address the problem of machines making intelligent, fair decisions that can be compared to decisions made by human loan officer. It will also address the useability of this system by non-coders, and the factors that will make this process better, such as good data and models.

This paper will streamline the process of loan applications and create and effective and robust solution, without the need of human interference.

PROBLEM STATEMENT

Applying for a loan is a tedious process that demands that the applicants are interviewed by a loan officer. This takes up unnecessary time and resources of both the customers and the bank. Is applied machine learning a way for the applicants to get instant response on their loan applications, without having to physically interact with the bank? Will this yield a positive result for the bank, freeing resources and doing as good of a job as the loan officers?

GOALS AND OBJECTIVES

The overall goal of this report is to explore the possibility of automating the approval process of loan applications using machine learning.

The following objectives has been set to achieve this:

- Research
    - Evaluate key concepts to be included in the theoretical framework of this paper.
    - Chart the standard application process of loans.
    - Research the existing field of Robot Process Automation.
    - Identify state-of-art solutions in Robot Process Automation. Analyse their evaluations to find elements of interest for this project.
    - Explore similar data sets to the one used in this project to consider further implementation of the model.
    - Investigate the business value of the development of similar systems.
- Development
    - Collect data.
    - Explore the data set.
    - Structure the RPA system.
    - Pre-process data.

    - Create model.
    - Create app for automatic processing of new loan data.
    - Deploy app.
- Evaluation
    - Evaluate data used for training and testing. Evaluate the similarity to other loan data.
    - Evaluate the model for predicting approval.
    - Evaluate the app and its implementations.
    - Evaluate the business value of this system.
    - Compare machine learning vs programming.
- Recommendations and further work

LIMITATIONS

This paper is limited by the data used and the short project scope. The principal of garbage in, garbage out takes full effect in every machine learning project. A model can only be as good as the quality of the data it uses to train. The project spans over three weeks and this will also impact decisions regarding improving the model, and the paper in general.

THEORY AND KEY CONCEPTS

This chapter presents data as a resource, automation in business processes, machine learning and the loan application process. The goal is to show the value that must be extracted from data to streamline business processes and increase efficiency and free up resources.

DATA DRIVEN BUSINESS DEVELOPMENT

The Economist published an article named “The world’s most valuable resource is no longer oil, but data”, informing us that the five most valuable listed companies in the world are data companies. Companies such as Google and Facebook have a tremendous value, but their most popular services are free (The world’s most valuable resource is no longer oil, but data, 2017). What is their value?

Their value lies in the data they accumulate about you, the consumer. Google and Facebook, revenue-wise, are mainly advertisement companies, and profits from selling ads to other companies (Sweney & Canon, 2021). Their effectiveness as advertisement companies comes from their ability to tailor ads to different users on their platform, therefore increasing the value of each advertisement. This shows that there is value in data.

Other businesses can take part of this discovered value, by utilizing their data to its fullest potential. Data gathered in a business can used internally to help the business to separate themselves in a competitive ecosystem, providing better products and services than their competitors. Data can also be used commercially by selling it to external businesses (Werger et al., 2020).

Data and physical assets have a lot in common. It can be bought and sold. It makes us more efficient and able to win, and keep, customers. Having it, without using it is a waste. It degrades over time and must be maintained. The big differences are that it does not become less when used and it is easily made accessible to everyone (Treder, 2019).

Data should be treated as every other asset. Give it a price, inventory it, maintain it, refine it and increase its value (Treder, 2019).

ROBOTIC PROCESS AUTOMATION

“RPA is the technological imitation of a human worker with the goal of automating structured tasks in a fast and cost efficient manner”(Aguirre & Rodriguez, 2017). RPA is not a physical robot, but a software created to do repetitive operational processes that usually are performed by humans. RPA can be used to automate processes based on structured data and deterministic outcomes. One of the huge advantages of RPA is that it can implemented on top of already existing systems, across platforms. They are also made to not require programming skills to use them (Aguirre & Rodriguez, 2017).

Some benefits of RPA are as following: Reduced costs by automating processes, increasing productivity, and needing less employees. Better customer experience by freeing up resources, giving more time to focus on customers. Lowering risk by removing the highest cause of errors, the humans. RPA can leverage your existing systems, so you don’t need to replace your infrastructure (Mitra, 2018).

MACHINE LEARNING

Machine learning is the application of algorithms that makes sense of data. By using self-learning algorithms, we can turn data into something much more valuable; knowledge, by spotting patterns and making predictions. Instead of relying on humans to make complicated models by making rules, machine learning offers a faster way of capturing knowledge and provides you with the ability to make data-driven choices (Raschka & Mijalili, 2019).

There are three different kind of machine learning. Supervised learning uses labelled data to train up a model, giving direct feedback to make predictions. Unsupervised learning has no labels or feedback, but it works by finding hidden structure in the data. Reinforcement learning is a decision process that works by rewarding wanted behaviour. Reinforcement learning knows nothing to start with, so decisions are random, but every time it does something right, its rewarded, and will therefore do it similarly the next time (Raschka & Mijalili, 2019).

LOAN APPLICATION PROCESS

The loan application summarized by a control-flow analysis:

- Application is submitted, but not necessarily finished.
- All applications are either internally pre-accepted, declined, cancelled by applicant, or contacted by call.
- Now that all applications are completed by call, the same decision process is initiated, and the application is finalized.
- Now remains approval, registration, and activation.

After this application process, the loan is offered, and a new process starts around the applicants answer (Molka et al., 2013).

The Norwegian newspaper “Finansavisen” published an article last year about the long processing times of loan applications. You could expect to wait up to three weeks for processing in some Norwegian banks, at the time (Parr, 2020).

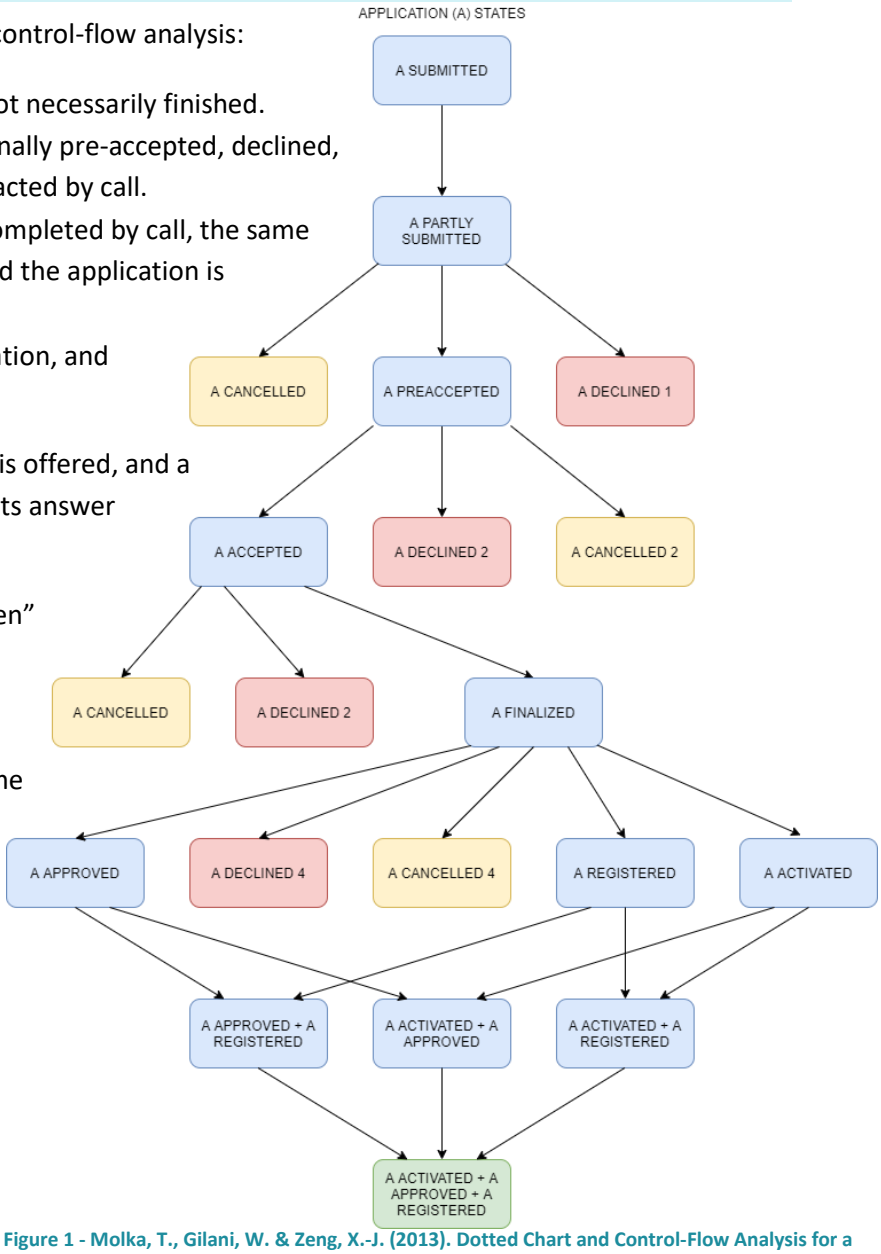


Figure 1 - Molka, T., Gilani, W. & Zeng, X.-J. (2013). Dotted Chart and Control-Flow Analysis for a Loan Application Process. Business Process Management Workshops, Tallinn, Estonia, pp. 219-220. Berlin, Heidelberg: Springer.

STATE-OF-THE-ART ANALYSIS

Some of the recent papers from the Business Process Management conference in 2020 gives an insight to the relevant state of art of the field of study RPA. The proceeding includes papers such as “A Conversational Digital Assistant for Intelligent Process Automation”, “How to Trust a Bot: An RPA User Perspective” and “From Robotic Process Automation to Intelligent Process Automation”(*Business Process Management: Blockchain and Robotic Process Automation Forum*, 2020).

The papers explore the use of machine learning combined with RPA interfaces to streamline the experiences of business processes. An example is how “A Conversational Digital Assistant for Intelligent Process Automation” solves the inexperience with the use an RPA with digital assistant, much like the ones we have at home, or in our phones. An example used is the simplification of the loan process. The assistant can help a loan officer without experience with machine learning, to automate the process of approving a loan, just by telling the digital assistant some key information (Rizk et al., 2020).

DEVELEOPMENT

This chapter contains the development of the automation process of loan applications.

DESIGN DESCRIPTION

CONCEPT OVERVIEW

CONCEPT

An application for instant response on loan applications.

DETAILS

This app will use a model created using machine learning to decide the outcome of loan applications. This will streamline the whole process and be beneficial for both the bank and the customers.

BENEFITS

- Freeing resources for the bank by streamlining a time-consuming process.
- Giving the customers instant feedback on their loan applications.

ISSUES

- Hard to understand how the model decides the outcome.
- Lack of customer service.

SYSTEM ARCHITECTURE

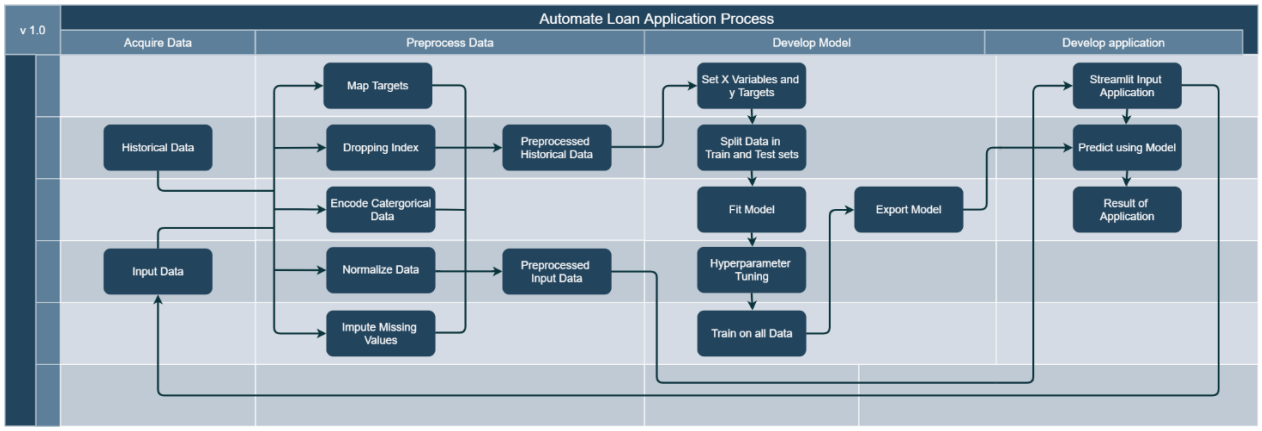


Figure 2 – System Architecture Flow Chart.

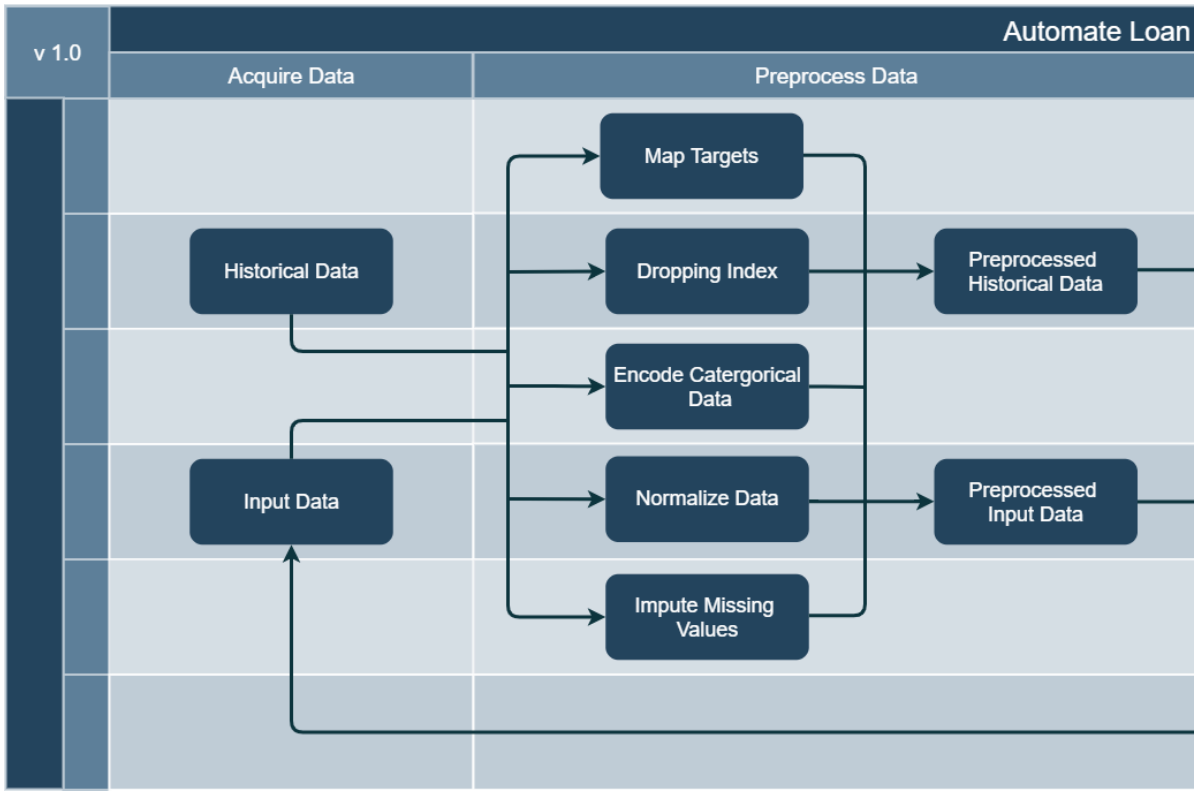


Figure 3 – System Architecture part 1.

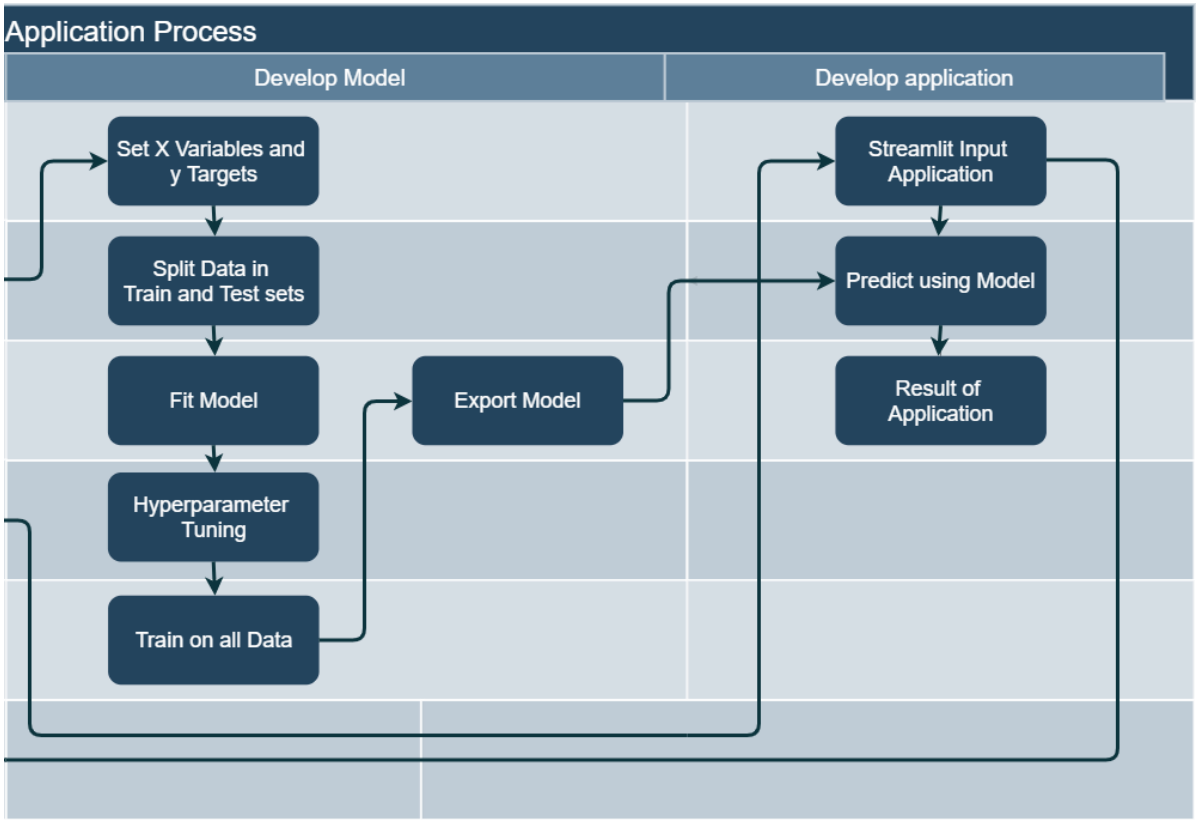


Figure 4 – System Architecture part 2.

EXPLORING THE DATA

The data used in this project was handed out as part of the TIN200 project material. It contains the information of 614 loan applicants and their loans, and their loans status.

"data_exploration.py" is a python script, found in the source code, that explores the attributes of the data. The script uses pandas, matplotlib and seaborn to display different characteristics such as missing values, distributions, and correlation.

Figure 5 shows the features the loan dataset contains. It contains information about the applicant's gender, marriage status, number of dependents, education, employee status, income, co-applicant's income, amount of loan applied for, terms of payback in months, if they have credit history and their property area.

Figure 5 also shows the number of missing values in the dataset. There are 22 missing values in the LoanAmount colum, and 14 missing values in the Loan_Amount_Term column, in the training dataset.

```
Shape of data: (614, 13)

Features:
Loan_ID
Gender
Married
Dependents
Education
Self_Employed
ApplicantIncome
CoapplicantIncome
LoanAmount
Loan_Amount_Term
Credit_History
Property_Area

Target:
Loan_status: ['Y' 'N']
NaN in train:
_____________________________
Loan_ID                       0
ApplicantIncome               0
CoapplicantIncome             0
LoanAmount                   22
Loan_Amount_Term             14
Loan_Status                   0
Gender_Male                   0
Married_Yes                   0
Dependents_1                  0
Dependents_2                  0
Dependents_3+                 0
Education_Not Graduate        0
Self_Employed_Yes             0
Credit_History_1.0            0
Property_Area_Semiurban       0
Property_Area_Urban           0
dtype: int64

NaN in test:
_____________________________
Loan_ID                       0
ApplicantIncome               0
CoapplicantIncome             0
LoanAmount                    5
Loan_Amount_Term              6
Gender_Male                   0
Married_Yes                   0
Dependents_1                  0
Dependents_2                  0
Dependents_3+                 0
Education_Not Graduate        0
Self_Employed_Yes             0
Credit_History_1.0            0
Property_Area_Semiurban       0
Property_Area_Urban           0
```

Figure 5 – Features and Missing Values.

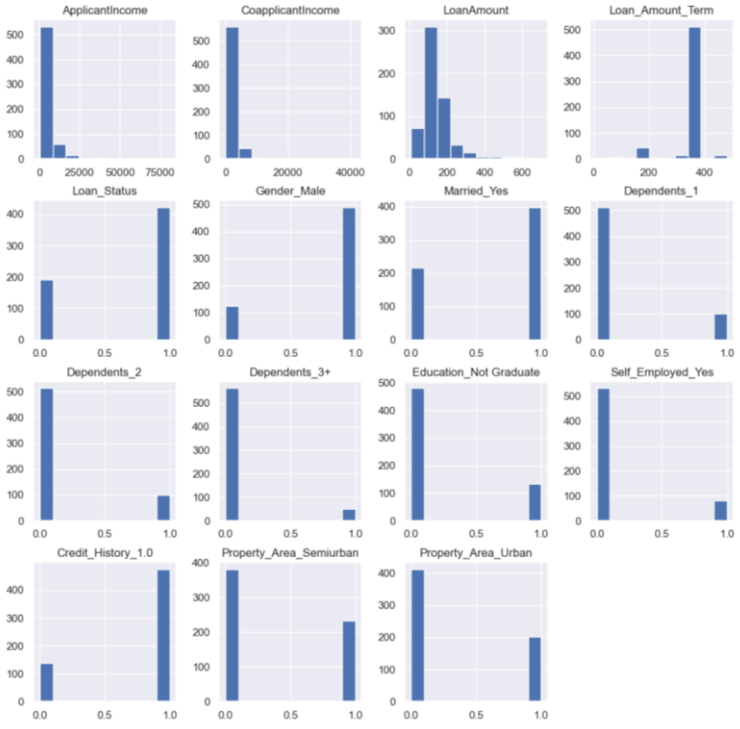


Figure 6 – Histogram showing the Data's Distribution.

Figure 6 shows the distribution of all the data. We can see that most of the columns are categorical and have only two different values, 0 or 1.

Error! Reference source not found. also shows the distribution of the variables in the dataset. The boxplot also gives us an idea of outliers in the dataset.

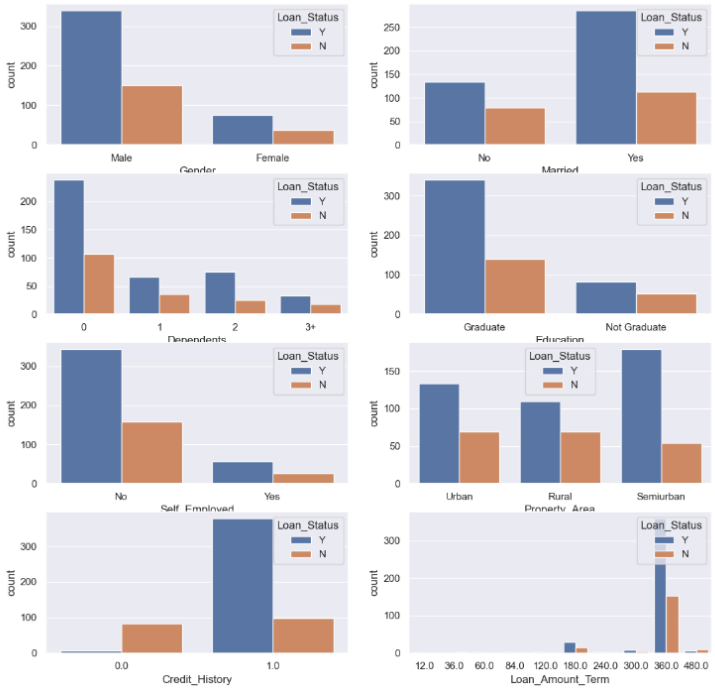
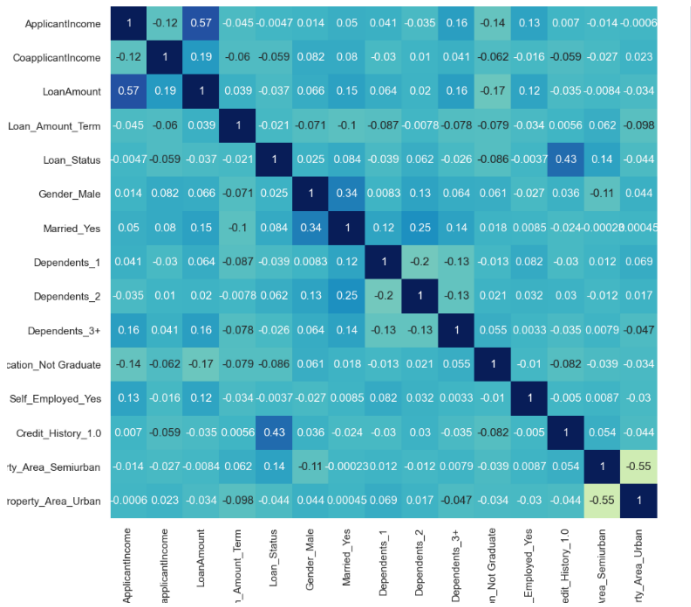


Figure 8 – Numerical Variables plotted vs Loan Status.

Figure 8 shows the distribution between the loan status of each of the numerical variables.



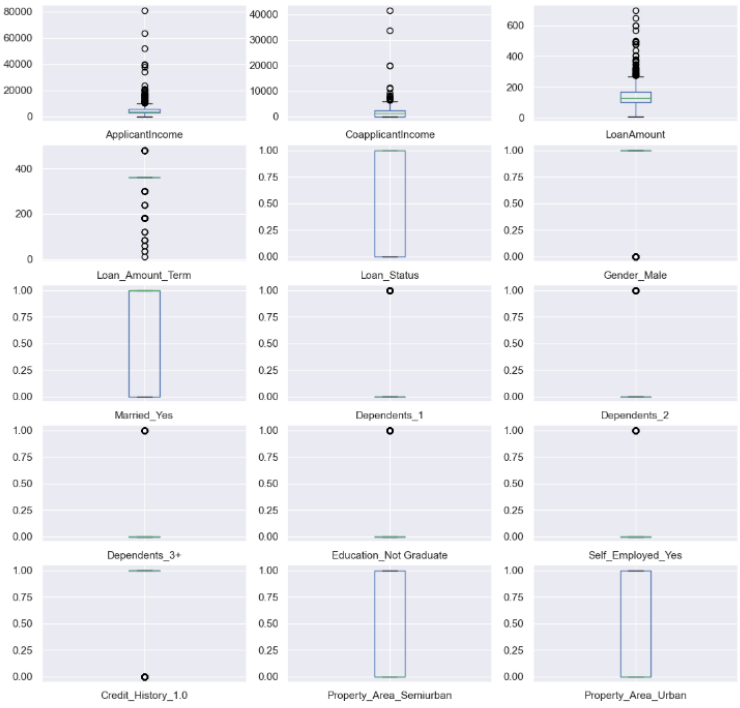


Figure 7 – Boxplot of Data.

Figure 9 shows the distribution between the loan status of each of the categorical variables.

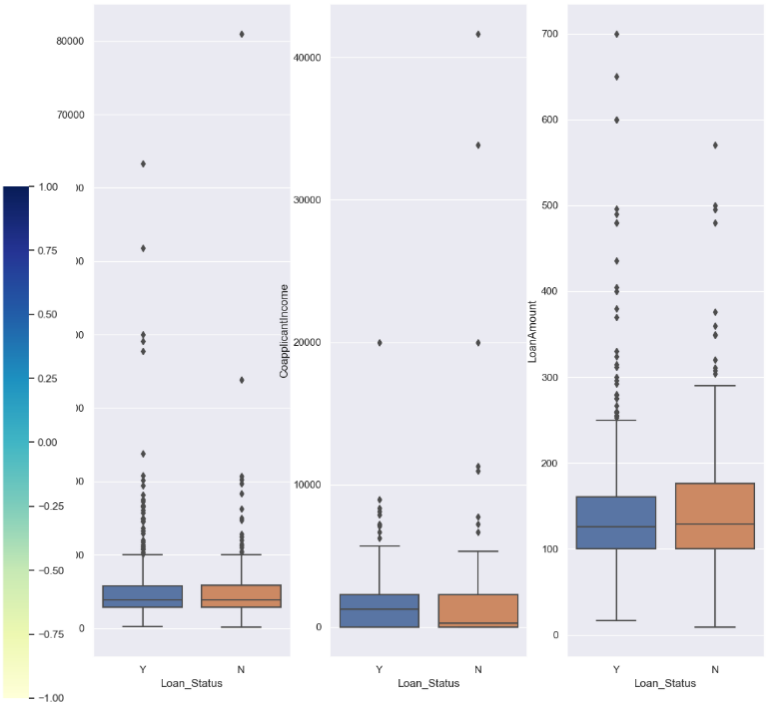


Figure 10 shows the correlation between all the variables.

SYSTEM DEVELOPMENT

PREPROCESSING

```python
# -*- coding: utf-8 -*-
"""
Created on Mon May 31 21:41:36 2021

@author: Eirik
"""

import pandas as pd
import numpy as np

from sklearn.impute import SimpleImputer
from sklearn.preprocessing import MinMaxScaler


# Reading data and making dataframe
data = pd.read_csv('train.csv')
test_data = pd.read_csv('test.csv')

# Making dataframes
df = pd.DataFrame(data)
test_df = pd.DataFrame(test_data)

#Target mapping - N: 0, Y = 1
target_mapping = {target: idx for idx, target in
                  enumerate(np.unique(df['Loan_Status']))}
df['Loan_Status'] = df['Loan_Status'].map(target_mapping)

# Dropping Loan_ID
df = df.drop(['Loan_ID'], axis=1)
test_df = test_df.drop(['Loan_ID'], axis=1)

# Create dummies
df_encoded = pd.get_dummies(data=df, drop_first=True,
                            columns=['Gender', 'Married', 'Dependents',
                                     'Education', 'Self_Employed',
                                     'Property_Area'])
test_df_encoded = pd.get_dummies(data=test_df, drop_first=True,
                                 columns=['Gender', 'Married', 'Dependents',
                                          'Education', 'Self_Employed',
                                          'Property_Area'])

# Normalizing data using MinMaxScaler
scaler = MinMaxScaler()
df_encoded = pd.DataFrame(scaler.fit_transform(df_encoded),
                          columns = df_encoded.columns)
test_df_encoded = pd.DataFrame(scaler.fit_transform(test_df_encoded),
                               columns = test_df_encoded.columns)

# Filling NaN data/Missing data with Simple Imputer
imputer = SimpleImputer(strategy='median')
df_encoded = pd.DataFrame(imputer.fit_transform(df_encoded), columns=df_encoded.columns)
test_df_encoded = pd.DataFrame(imputer.fit_transform(test_df_encoded), columns= test_df_encoded.columns)

# Save dataframes to csv
df_encoded.to_csv('preprocessed_train.csv', encoding='utf-8',index=None)
test_df_encoded.to_csv('preprocessed_test.csv', encoding='utf-8',index=None)
```

Figure 11 – Pre-Processing script.

Figure 11 shows how the data is imported from csv using Pandas and pre-processed. The target, “Loan Status”, which is either yes or no (Y or N), is mapped to 0 for no, and 1 for yes. The first column, “Loan_ID”, is removed because it is obviously not important for developing an accurate model. The categorical variables are encoded to multiple binary variables instead because the model does not know the difference between

categorical variables. The data is normalized using MinMaxScaler from the scikit-learn library. The missing values are imputed over the median of the other values in each column. The newly pre-processed data is saved as a csv file to be accessed by the model.

MODELLING

- Pre-processed data is imported.
- Data is split into y, targets, and X, features.
- The data is split random into 2/3 training data and 1/3 testing data. The testing data is later used to validate the model.
- XGBoostClassifier, or Extreme Gradient Boosting Classifier, is used to make the model.
- The model is fitted with the training data from the earlier split.
- The hyperparameters of the model is tuned using RandomizedSearchCV.
- A confusion matrix is created to show the accuracy of the model in an understandable way.
- The model is trained on all the data, without doing a train-test-split.
- The model is exported.

```python
# -*- coding: utf-8 -*-
"""
Created on Wed Jun  2 12:41:44 2021

@author: Eirik
"""

import pandas as pd
import matplotlib.pyplot as plt
import pickle
from scipy.stats import uniform, randint

from sklearn.model_selection import train_test_split
from sklearn.model_selection import RandomizedSearchCV
from sklearn.metrics import confusion_matrix
import xgboost as xgb

# Reading data and making dataframe
df = pd.read_csv('preprocessed_train.csv')

# Setting the y values (target)
y = df['Loan_Status'].values

# Setting up the x values (features)
X = df.drop(['Loan_Status'], axis=1).values

# Split data into training and test data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
                                                    random_state=1, stratify=y)
model = xgb.XGBClassifier(objective="binary:logistic", random_state=1,
                          use_label_encoder=False)
model.fit(X_train, y_train)

# Hyperparameter tuning
param_grid = {
    "colsample_bytree": uniform(0.7, 0.3),
    "gamma": uniform(0, 0.5),
    "learning_rate": uniform(0.03, 0.3),
    "max_depth": randint(2, 6),
    "n_estimators": randint(100, 150),
    "subsample": uniform(0.6, 0.4)
}

rs_model = RandomizedSearchCV(estimator=model,
                  param_distributions=param_grid,
                  scoring='roc_auc',
                  cv=10,
                  n_jobs=-1,
                  n_iter=200,
                  verbose=1)

rs_model = rs_model.fit(X_train, y_train)
print(rs_model.best_params_)
print('Test accuracy: %.3f' % rs_model.score(X_test, y_test))

# Creating confusion matrix
y_pred = rs_model.predict(X_test)
confmat = confusion_matrix(y_true=y_test, y_pred=y_pred)

fig, ax = plt.subplots(figsize=(2.5, 2.5))
ax.matshow(confmat, cmap=plt.cm.Blues, alpha=0.3)
for i in range(confmat.shape[0]):
    for j in range(confmat.shape[1]):
        ax.text(x=j, y=i, s=confmat[i, j], va='center', ha='center')

plt.xlabel('Predicted label')
plt.ylabel('True label')
plt.tight_layout()
plt.show()

# Training on all train data
rs_model = rs_model.fit(X, y)

# Saving model
filename = 'loan_application_model.sav'
pickle.dump(rs_model, open(filename, 'wb'))
```

Figure 12 – Model training script.

```
[21:28:57] WARNING: ..\src\learner.cc:1061: Starting in XGBoost
1.3.0, the default evaluation metric used with the objective
'binary:logistic' was changed from 'error' to 'logloss'. Explicitly
set eval_metric if you'd like to restore the old behavior.
Fitting 10 folds for each of 200 candidates, totalling 2000 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent
workers.
[Parallel(n_jobs=-1)]: Done  48 tasks      | elapsed:    3.2s
[Parallel(n_jobs=-1)]: Done 348 tasks      | elapsed:    9.7s
[Parallel(n_jobs=-1)]: Done 848 tasks      | elapsed:   20.7s
[Parallel(n_jobs=-1)]: Done 1548 tasks      | elapsed:   36.2s
[21:29:44] WARNING: ..\src\learner.cc:1061: Starting in XGBoost
1.3.0, the default evaluation metric used with the objective
'binary:logistic' was changed from 'error' to 'logloss'. Explicitly
set eval_metric if you'd like to restore the old behavior.
{'colsample_bytree': 0.7618710631033927, 'gamma':
0.4924722485783868, 'learning_rate': 0.13887550893978923,
'max_depth': 3, 'n_estimators': 105, 'subsample':
0.7972050064712304}
Test accuracy: 0.758
```

Figure 14 – Output from console after modelling.

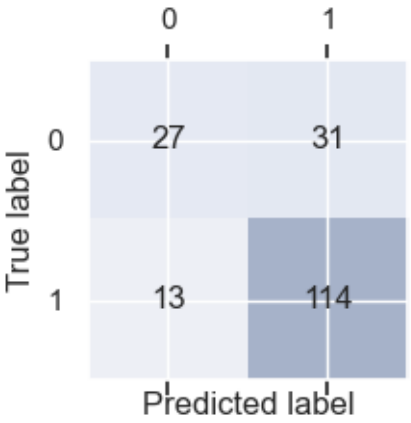


Figure 13 – Confusion Matrix.

Figure 14 and Figure 13 shows the accuracy of the model. The model has an accuracy of 79.7%. 114 of the approved loans have been successfully predicted. 13 of the approved loans have been misclassified as not approved. 27 of the not approved loans has been successfully predicted. 31 of the not approved loans has been misclassified as approved loans.

APPLICATION

Figure 15 shows the imports used in the application script.

```
# -*- coding: utf-8 -*-
"""
Created on Tue Jun  1 10:56:48 2021

@author: Eirik
"""

import streamlit as st
import pandas as pd
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import MinMaxScaler
import pickle
import numpy as np

```

Figure 15 – Imports for the application script.

The application is created using Streamlit. Streamlit is used to make an easy-to-use interface to interact with the model and to predict the outcome of a loan application. This can be done without having to directly interact with the Python script.

The inputted data is collected and added to a test dataset which does not contain target labels. This is done so that the inputted data will fit into a DataFrame with the same number of columns as the model takes.

The data is scaled the same way as the data the model is trained on, and eventual missing values are imputed.

The outcome of the application is displayed in the Streamlit app.

```python
# Reading data and making dataframe - Needed because of encoder
test_df = pd.read_csv('test.csv')

# Filling in information in streamlit
st.title('Loan application')
st.markdown('Please fill in information below: ')
gender = st.selectbox('Gender: ',('Male', 'Female'))
married = st.selectbox('Married? ',('No', 'Yes'))
dependents = st.selectbox('How many are dependent on you? ',('0','1','2','3+'))
education = st.selectbox('Education: ',('Graduate', 'Not Graduate'))
self_employed = st.selectbox('Are you self employed? ',('No', 'Yes'))
slider_a_income = st.slider('What is your income?',value=3000, min_value=0,
                            max_value=10000, step=100)
applicant_income = st.number_input('',value=slider_a_income)
slider_co_income = st.slider('What is your coapplicant`s income?' ,value=3500,
                             min_value=0, max_value=10000, step=100)
coapplicant_income = st.number_input('',value=slider_co_income)
slider_loan_amount = st.slider('How much do u want to loan?',value=200,
                               min_value=0, max_value=600, step=10)
loan_amount = st.number_input('',value=slider_loan_amount)
slider_loan_amount_term = st.slider('How long is the term of the loan in months?',
                                    value=360, min_value=0, max_value=700, step=10)
loan_amount_term = st.number_input('',value=slider_loan_amount_term)
credit_history = st.selectbox('Your credit history: ',('0', '1'))
property_area = st.selectbox('Your property area? ',('Urban', 'Rural',
                                                      'Semiurban'))

# Saving information
new_data = {
        'Gender': gender,
        'Married': married,
        'Dependents': dependents,
        'Education': education,
        'Self_Employed': self_employed,
        'ApplicantIncome': applicant_income,
        'CoapplicantIncome': coapplicant_income,
        'LoanAmount': loan_amount,
        'Loan_Amount_Term': loan_amount_term,
        'Credit_History': credit_history,
        'Property_Area': property_area
        }

# Dropping Loan_ID
test_df = test_df.drop(['Loan_ID'], axis=1)

# Making dataframe with inputted data
new_df = pd.DataFrame.from_records(new_data, index=[0], columns=new_data.keys())

# Adding new data to old dataframe because of encoding
test_df = test_df.append(new_df, ignore_index=True)

# Encoding to match model
test_df_encoded = pd.get_dummies(data=test_df, drop_first=True,
                                 columns=['Gender', 'Married', 'Dependents',
                                          'Education', 'Self_Employed',
                                          'Property_Area'])

# Normalizing data using MinMaxScaler
scaler = MinMaxScaler()
test_df_encoded = pd.DataFrame(scaler.fit_transform(test_df_encoded),
                               columns = test_df_encoded.columns)

# Filling NaN data/Missing data with KNNImputer
imputer = SimpleImputer(strategy='median')
test_df_encoded = pd.DataFrame(imputer.fit_transform(test_df_encoded),
                               columns= test_df_encoded.columns)

# Setting up X
X = test_df_encoded.values

# Load model
filename = 'loan_application_model.sav'
model = pickle.load(open(filename, 'rb'))
y_pred = model.predict(X)

# Saving predictions
pred_df = pd.DataFrame(y_pred, columns = ['Loan_Status'])
pred_df['Loan_Status'] = np.where(pred_df['Loan_Status']==0, 'No', 'Yes')

result = st.button('Check for loan approval')
if result:
    if pred_df.Loan_Status.iat[-1] == 'No':
        st.markdown('Your loan has been declined')
    else:
        st.markdown('Your loan has been approved')
```

Figure 16 – The rest of the application script.

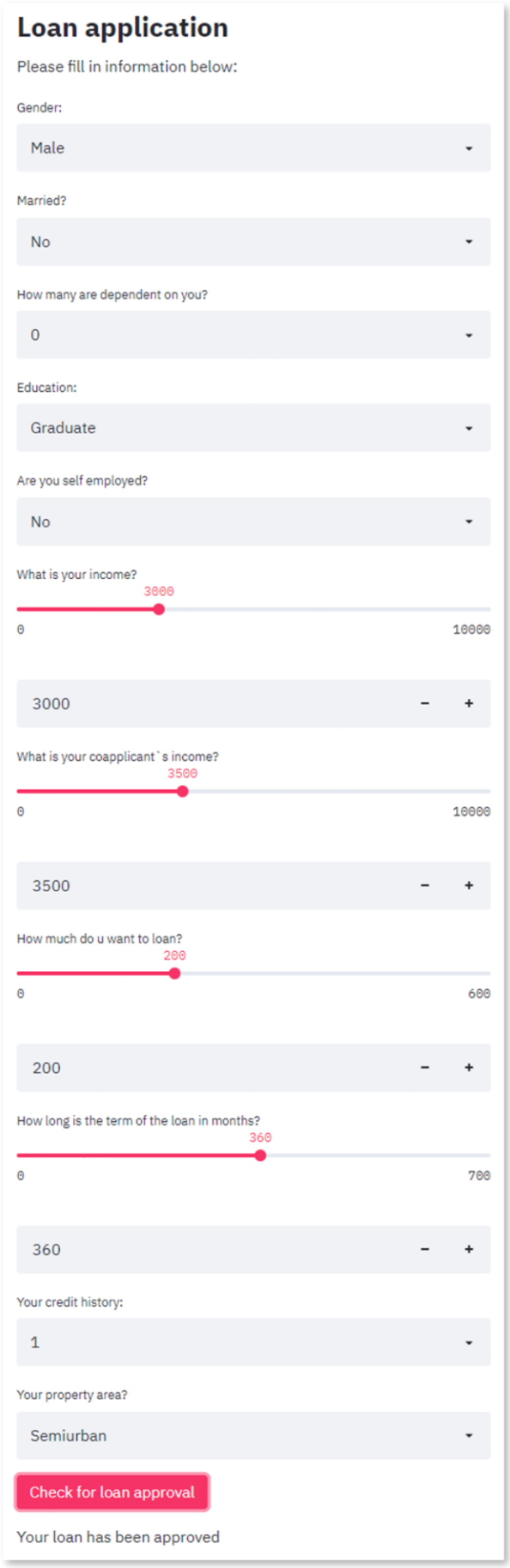


Figure 18 – Approved Loan in Streamlit

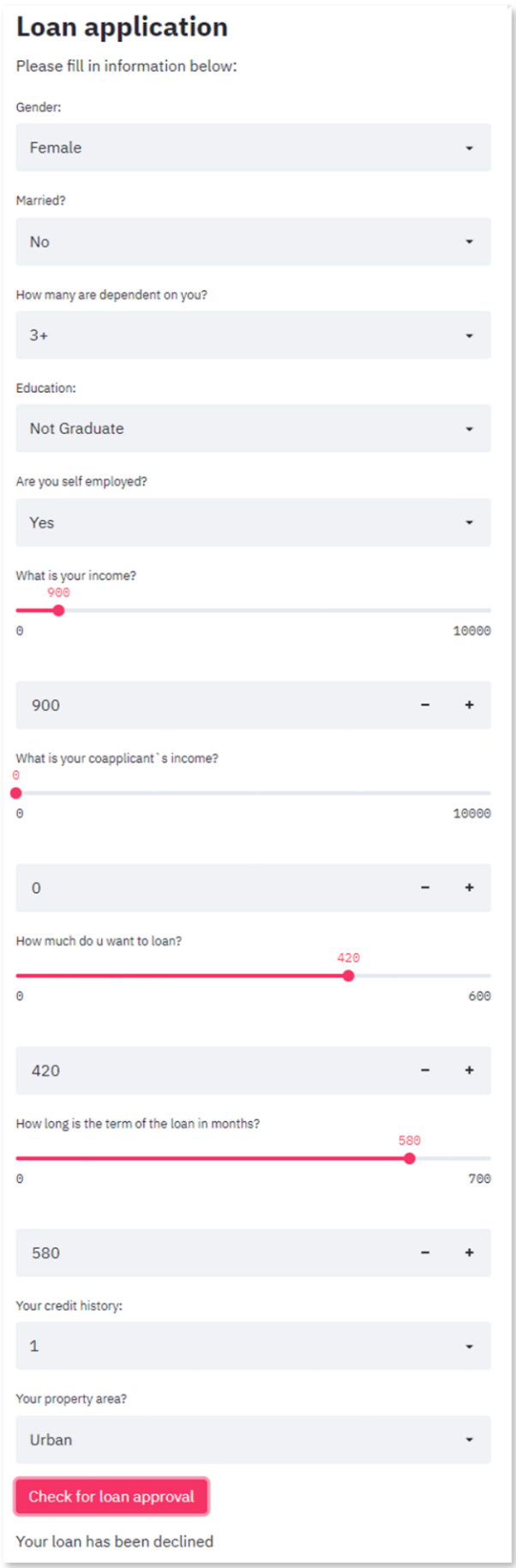


Figure 17 – Declined Loan in Streamlit.

EVALUATION AND RESULTS

This chapter presents the results of this study. It includes an evaluation of different choices made in this project.

DATA

The data used for this project was handed out as course material in TIN200 and was to be used despite unfortunate features that might have been discovered during the exploration of the data. The project was based on the concept of the data being good enough to create a working model. If this project were to be implemented in a real-life-scenario, the data would have been reconsidered. Only the train data handed out was used for this project, as the test data was created by filling out the form in the streamlit module.

The data exploration shows that the data contains 614 loan applications, including 13 features each. The data also includes a target label telling the status of the loan. Some of the applications are missing some features, either the “LoanAmount” or the “Loan_Amount_Term”. These missing values was imputed using the median, but in hindsight, they might as well have been removed. It does not make sense to apply for a loan, but not give the amount you apply for, or the period of back payment. A solution to remove such data would be to dismiss all applications that is not filled out completely.

A quick overview in the chapter: Exploring the data, shows us by looking at the distributions and the confusion matrix, that the feature with the biggest impact on the loan status seems to be “Credit_History”.

MODEL

The model is trained by the help of XGBoostClassifier with an accuracy of 79.7%. More classifiers were tested, some giving better results on the train-test-split, but to avoid overfitting, this classifier was chosen. This was also decided after testing predictions with inputted data which gave an unrealistic big value to the credit history when using other models. This error lies in the data used for training, but preventing overfitting makes the solution more robust, and more applicable to other data.

Figure 13 – Confusion Matrix. shows that we have some room for improvement since 31 loans which was not approved by a loan officer, was approved by the machine. One of the issues of this paper is that we cannot know for sure if the loan officers have made the right decision or not. A model is never better then the people who labels the targets. The better the data, the better the model. This model works very good for demonstrative purposes.

IMPLEMENTATION

The model is implemented in a scrip that is run through the Streamlit module made for python. This module makes it easy to visualise code, and can be used to make an easy-to-use interface, which does not demand any computer knowledge. This is excellent for this demonstration.

The Streamlit module creates an app with a fill-in form that is easy to navigate and understand. After the form is filled in, it gets processed in the background and tested by the model. This takes a few milliseconds, and from there you can see if your application has been approved or declined.

BUSINESS VALUE

We can conclude that this solution, correctly implemented, will give added business value to any bank. We have freed up a lot of capacity and resources by automating the loan process. We have given the customers an effective solution which gains everyone.

We can imagine that this solution can be implemented fast, but it needs good supervision, and an opportunity to solve problems ad hoc. Customer service must also be maintained.

CONCLUSION AND RECOMMENDATIONS

The goal of this project was to research the process of loan applications, create a model which can automate the loan application process, and make a user interface to easily use the model.

This paper explored the basics of the state of the loan application process as is. It discovered that it is a tedious process which in some cases takes up to three weeks, taking a hold of resources and time.

A model was created using course material data, to predict the outcome of loan applications. This model had an accuracy of 79.7%, which meant that in almost 4 of 5 cases, the machine would do the same as the loan officer. This was considered good enough for demonstrative purposes.

The model was implemented to a user interface using Streamlit, making a working demonstration on how an easy fill out form could give instant response to a loan application.

The project is considered a success in the way it demonstrates the possible application of machine learning in the automation of the loan application process. It is easy to use, and incredibly fast. The project demonstrates a clear business value potential.

FURTHER WORK

The following points should be considered done in the continuation of this project:

- Cooperate with a bank to get real life, quality data that would be authentic in a loan application process. The quantity should be increased, and other factors should be considered such as inflation, to future proof the model.
- The model should be trained, tuned, and tested to perfection. A hybrid between a normal programmed program and a machine learning model should be considered, to make the prediction more robust to special cases.
- Integration between the chosen interface, and the existing interfaces of the bank, to make it easy to use this model for non-coders. First step would be to implement something like pyinstaller, which can make python scripts into executables, so you do not need python on your machine to run the model.

ATTACHMENTS

SOURCE CODE

https://github.com/hellund/loan_application_processing.git

REFERENCES

Aguirre, S. & Rodriguez, A. (2017). *Automation of a Business Process Using Robotic Process Automation (RPA): A Case Study*. Applied Computer Sciences in Engineering, Cartagena, Colombia, pp. 65-71: Springer International Publishing.

Business Process Management: Blockchain and Robotic Process Automation Forum. (2020). Business Process Managemen, Seville, Spain. Cham, Switzerland: Springer.

Mitra, M. (2018). *Robotic Process Automation(RPA) and Benefits*. Web: Mantra Labs. Available at: https://www.mantralabsglobal.com/blog/robotic-process-automationrpa-and-benefits/ (accessed: 10.06).

Molka, T., Gilani, W. & Zeng, X.-J. (2013). *Dotted Chart and Control-Flow Analysis for a Loan Application Process*. Business Process Management Workshops, Tallinn, Estonia, pp. 219-220. Berlin, Heidelberg: Springer.

Parr, O. S. (2020, 1. juli 2020). Travle boligkjøpere stresser bankene. *Finansavisen*. Available at: https://finansavisen.no/nyheter/bolig/2020/07/01/7540549/rekordaktivt-boligmarked-gir-lanerush-dnb-beklager-lang-ventetid (accessed: 10.06.2021).

Raschka, S. & Mijalili, V. (2019). *Python Machine Learning*. Third ed. Birmingham, UK: Packt Publishing Ltd.

Rizk, Y., Isahagian, V., Boag, S., Khazaeni, Y., Unuvar, M., Muthusamy, V. & Khalaf, R. (2020). *A Conversational Digital Assistant for*

Intelligent Process Automation. Business Process Management: Blockchain and Robotic Process Automation Forum, pp. 85–100: Springer Nature Switzerland AG 2020.

Sweney, M. & Canon, G. (2021, 28. april 2021). Alphabet: revenue soars for Google owner as Covid lockdown boom continues. *The Guardian*. Available at: https://www.theguardian.com/technology/2021/apr/27/alphabet-google-revenue-quarterly-earnings (accessed: 10.06.2021).

Treder, M. (2019). *Becoming a data-driven Organisation*

Unlock the value of data. 1 ed. Berlin, Heidelberg: Springer Vieweg.

Werger, K., Kenedy, J., Peckham, D., Mather, S., Ginsberg, R., Jooste, A., Robinson, A. & Knappenberger, D. (2020). Data valuation: Understanding the value of your data assets. 12. Available at: https://www2.deloitte.com/content/dam/Deloitte/global/Documents/Finance/Valuation-Data-Digital.pdf (accessed: 10.06.2021).

The world's most valuable resource is no longer oil, but data. (2017, May 6th 2017). *The Economist*. Available at: https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data (accessed: 10.06.2021).