Loan Application Assistant: App Development

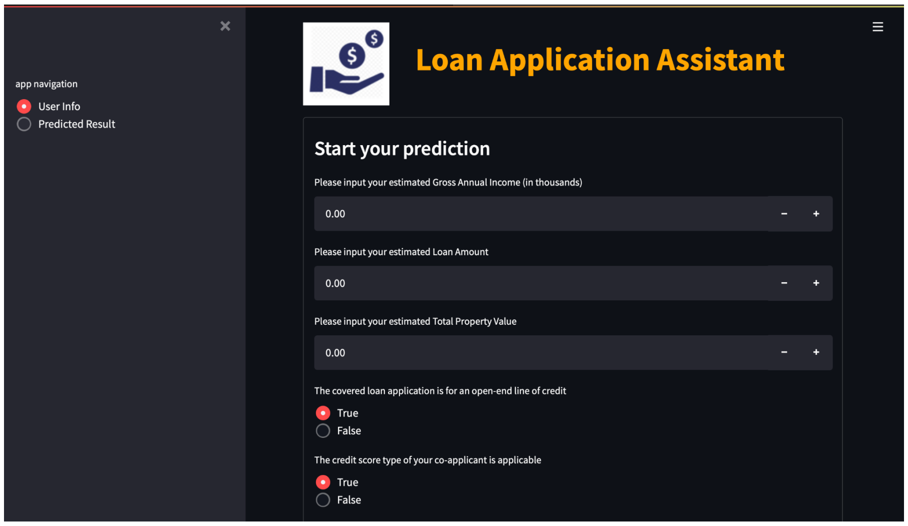
Xinzhu Wang xw486

Susan Wu fw249

Yuchen Tang yt388

Abstract

Today, loans are one of the most popular, reliable and efficient ways for people to borrow money. Applicants submitted their loan application with required personal information to the bank and the financial institutions decided whether to approve or reject the loan application based on the evaluation of the applicant's information.However, for those credit applicants, there are lacks of useful tools to help them to have a forecasting and implication analysis on their application decisions before they submit their application to the bank. In order to provide assistance to all the potential loan applicants, we implemented machine learning algorithms and sensitivity analysis to develop an APP that could accurately predict the success rate of a loan application and provide suggestions about the possible methods to boost the chance of success.



Start your prediction

Please input your estimated Gross Annual Income (in thousands)

0.00

Please input your estimated Loan Amount

0.00

Please input your estimated Total Property Value

0.00

The covered loan application is for an open-end line of credit

[✓ True] False

The credit score type of your co-applicant is applicable

[✓ True] False

Table of Contents

Introduction

Credit approval is a common application or process for individuals to gain eligibility for financing. With the rapid growth in machine learning and AI techniques, as well as the volume of the applications, the popularity of data-driven prediction models in the financial industry continues to increase(Lusinga, Mokoena, Modupe & Mariate, 2021). Financial institutions have taken steps to transform the traditional evaluation methods using credit-scoring systems into data-driven decision-making methods using big data and innovative Fin techs. However, there are few such advanced tools for the credit applicants' usage to know more about decisions of their credit applications in advance to hand in those applications. Easy accesses and diverse resources of datasets recording comprehensive customers' information provide possibilities to swap out this 'blind spot'. By analyzing and exploring a useful dataset that contains historical information of applicants, such as customers' property values, annual income and so on, relationships between those dependent variables and the target vairable, decisions on applications can be revealed and described in a numerical way, a linear regression model.

Based on those discussions on the innovative decision-making methods used in the financial industry, parts of following sections introduce details about the developed logistic regression model giving predicted probabilities of an application getting approved to assist credit applicants with their applications before they step into processes in banks or financial insitutions. The main algorithm is to 'learnt' important information from old data, then to construct a logistic regression model to provide accurate predictions on new data, website users' input information.

Addition to the development of the prediction model giving a predicting result, sensitivity analysis also plays a role in providing suggestions depending on users' information and their predicting results. The constructed prediction model and sensitivity analysis results finally implemented in Streamlit in which the usable website giving assistance to credit applicants was developed. How this method contributes to give suggestions about applications and how those algorithms are implemented in the developed website in Streamlit are addressed below.

Problem Identification & Stakeholder Analysis

From people around us, we recognized that the feelings of uncertainty prevented a lot of them from applying for a loan when they actually needed money. Some of them were worried about being rejected and the potential impacts brought by the refusal. Some of them gave up getting a loan from the bank as the application requires too much information for them to prepare and organize. As loan is one of the most well-regulated and diplomatic ways for people to quickly borrow the money that they require, we believe that a loan assistant could have a good market demand and potential. After further research on the area related to loan application, we realized that a number of people could have the chance to pass the application but ended up with denials. For example, only if the applicant lowered his applied loan amount by 10 percent could dramatically boost his chance of success. Unfortunately, the applicant had no way to acquire such information in advance. Therefore, we feel like a predictor to evaluate applicants' personal info and predict the probability of being approved before they submit their application to the bank could make their life much easier. Then, in order to further explore the problems from the perspective of our potential stakeholders, we conducted a stakeholder analysis by doing a survey.

Survey:

As we are developing an APP to help and support loan applicants to be better prepared before they submit the loan applications to the bank, as an applicant, what information or service would you expect from such an application?

Below are some of the answers from the Volunteers:

- I would like to know the probability of the credit approval with details of getting this probability and also if I can not apply for the full amount, give me some suggestions of increasing this probability if it's considerably low.
- My last loan application was rejected by Bank of America. Is the rejection going to affect the chance of the approval for the next time? It would be great if you can tell me my chance of getting approval on the loan application based on my current situation.
- I want to know the probability of getting rejected. If the chance is high, I would not continue my application as it always takes a lot of time but switch to other more efficient ways for money.
- It would be great if you could provide me with some pointed suggestions for the loan application

based on my personal situation rather than general instructions everywhere on the internet.

Through a further analysis of the answers given by the volunteers, we extracted two functionalities that are mentioned and expected by most of the potential users. First of all, almost everyone cared about the chance of getting approval from the bank and wanted to know the estimated probability in advance. Secondly, most of them were interested in the possible ways to increase their probability of success. Thus, based on the expectations of our potential stakeholders, we decided to design an APP that is able to predict the probability of getting approval, and provide specific suggestions on how to increase the chances of being approved based on every user's information.

Method

Dataset Description

The dataset is from an official website of the United States government, FFIEC.(link: https://ffiec.cfpb.gov/data-browser/data/2020?category=states&items=NY&leis=7H6GLXDRUGQFU57RNE97). The used dataset is 2018 HMDA US nationwide dataset, the financial institution would be Bank of America, National Association - B4TYDEB6GKMZO031MB27. There are originally 99 features that contain personal information of existing clients, such as income, race, pre-approval status and so on. In total, it contains 377468 samples. This large dataset contains sufficient data and information for us to further apply big data techniques, such as feature selections, cross-validations, of building linear models and testing the accuracy.

Data Cleaning

The final delivery is a website requiring information from users, so features, for example, 'denial reason', that users could not provide information dropped directly. After simply dropping those features by definitions, missing values were dealt with firstly, since it would lead to errors in further feature selection and modeling steps. Features were separated into 2 groups, numerical features and categorical features.

For numerical features, features containing missing values in more than a half of entries were also dropped directly. If those features remained and missing values in those features were replaced with some statistics, the statistical characteristics, mainly the distribution, would be largely modified and affect the accuracy of the model. For other missing value entries in each remaining column, they were replaced with column median. Entries with outliers in each column were also removed, since logistic regression is not considered to be robust, and outliers would largely affect the accuracy of the model.

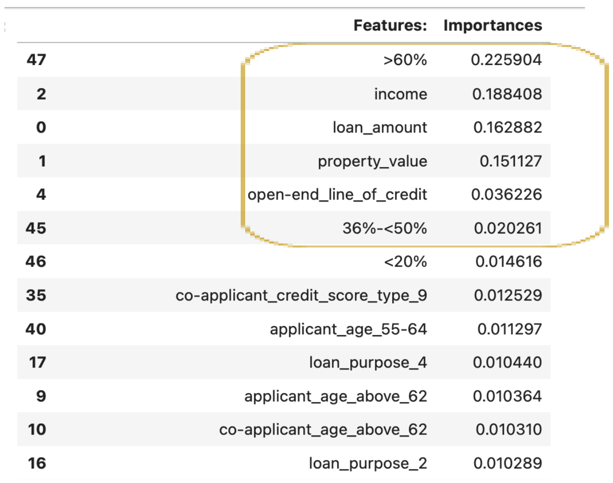
It is worth mentioning that there is a special case, 'income' column. This column contained lots of missing values that should be dropped typically. However, it's discovered that this feature has a considerable high importance to the target variable, so it's remained. For finding more optimal and accurate values to missing entries, Random Forest Regression was used to predict missing values. Correlations between other dependent variables, including categorical features that had been encoded, and 'income' features were calculated, and features with high correlations were chosen to give the prediction.

For categorical features, since the value type in each entry was string that could not be processed in the feature selection and modeling steps, 2 typical encoding methods were applied: one-hot encoding to multi-categorical features; boolean encoding to features only have 2 categories. This includes the target variable, 'action taken'. Then the cleaned dataset was splitted into dependent variables, X, and tergat variable, Y. At this stage, X had 356 features and 372656 data points; Y had 2 categories, approve and not approve encoding as 0 and 1. Feature selection was applied to X, and it's discussed in the next section.

Feature Selection

Feature selection plays a crucial role in data analysis, especially when the dataset has a high dimensionality. On one hand, filtering the dataset by removing the irrelevant information could simplify the model and reduce the amount of time spent on model training. On the other hand, a model is likely to have a higher risk of overfitting and lower accuracy without feature selection when the model is built upon too much redundant data and thus the predictive result is misled by the noise. The essence of feature selection is to measure and evaluate the relevance and importance of each feature to the target variable. Through feature selection, the redundant features in the original dataset are simply eliminated, while the relevant and critical ones are preserved to train the model. With a set of well-selected features, it's easier to construct a simple but precise model, and that's why we did the feature selection before model construction.

To precisely estimate the importance of each feature to the target variable, we implemented the random forest regressor to fit the raw dataset which returned a feature_importances_ property that can be accessed by retrieving the relative importance scores for each input feature. In this way, we can acquire a clear visual representation of the importance of each feature within the data. Below are part of the features with the top importance. We finally selected the top six features in the yellow box as we realized that there would be little improvement on the model accuracy by adding features with importance less than 0.02.



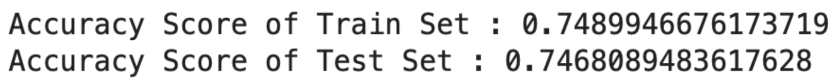
	Features:	Importances
47	>60%	0.225904
2	income	0.188408
0	loan_amount	0.162882
1	property_value	0.151127
4	open-end_line_of_credit	0.036226
45	36%-<50%	0.020261
46	<20%	0.014616
35	co-applicant_credit_score_type_9	0.012529
40	applicant_age_55-64	0.011297
17	loan_purpose_4	0.010440
9	applicant_age_above_62	0.010364
10	co-applicant_age_above_62	0.010310
16	loan_purpose_2	0.010289

Selected Features Definition:

- *>60%: Current Debt to Income Ratio>60%*
- *Income: Gross Annual Income (in thousands)*
- *loan_amount: Expected Loan Amount*
- *property_value: Total Property Value*
- *open-end_line_of_credit: whether the covered loan application is for an open-end line of credit*
- *36%-<50%: Current Debt to Income Ratio between 36% and 50%*

Logistic Regression Model

Taking the chance of success as the target value, we implemented the logistic regression model to predict the result (success / fail) of the loan application and the corresponding probability of each result. To check the performance of the model and ensure no overfitting, we conducted a cross validation and divided the dataset into a train set and a test set at a ratio of 7 to 3 for the model construction. First, we fit the model to the train set based on the six selected features [>60%, income, loan_amount, property_value, open-end_line_of_credit, 36%-50%] and got an accuracy score of 0.749. Then, we implemented the model to predict based on the test set and received an accuracy score of 0.747.



```
Accuracy Score of Train Set : 0.7489946676173719
Accuracy Score of Test Set : 0.7468089483617628
```

The trivial difference between the score of the train set and of the test set guaranteed that there was no overfitting. However, frankly speaking, the accuracy scores around 75 percent were a little below our expectation. Therefore, more methods that could potentially boost the accuracy of the model would be explored and developed in the future.

Sensitivity Analysis

The modeling step gives us a predictive model that helps us determine whether we can grant a loan to a user. At the same time, we would like to use our system and predictive model to give the user some suggestions based on different users' information to help them increase the probability of successful applications. This requires analyzing the impact of user information on the output of the prediction model. This is where sensitivity analysis comes into play. Sensitivity analysis determines how different values of an independent variable affect a particular dependent variable under a given set of assumptions. In other words, sensitivity analyses study how various sources of uncertainty in a mathematical model contribute to the model's overall uncertainty. In cases where the exact boundaries of the variables are available, sensitivity analysis is often used to report the input variables that have the greatest/minimal impact on the key simulation results. This can be a useful preliminary step for optimization analysis. Sensitivity analysis can also be

used to help manage risk in a systematic way as part of energy performance contract analysis.
In our project, the inputs to the model are information about the user's income, savings, and desired loan amount, and the outputs are whether the loan application is successful and the probability of success, and we need to analyze the impact of this user information on the outcome of the application. There are two types of approaches for sensitivity analysis of the model, Local approaches and Global approaches. Local approaches, i.e., one-factor at a time sensitivity analysis, vary one factor at a time while keeping other factors constant. One-factor sensitivity analysis assumes that the other factors are constant when calculating the impact of a given uncertainty on the project economics, but in practice this assumption is difficult to make. There may be two or more uncertainties that are changing at the same time, so it is difficult to accurately reflect the impact of different variables on the model output, and therefore a multi-factor sensitivity analysis is necessary. The applicable scenarios of different analysis methods are shown in Table 1. Our model has a total of nine inputs, and each input variable is continuous and not directly correlated. To facilitate the calculation and analysis, we choose the Variance-based method.

Table 1 Sensitivity Analysis Techniques

Type	Model independent?	Sample source	No. factors	Factor range	Multi-factor variation
Morris	yes	levels	20-100	global	yes
Variance	yes	distributions	< 20	global	yes
Factorial	yes	levels	> 100	global	yes
Monte Carlo	yes	distributions	< 20	global	yes
Local SA	yes	levels	< 100	local	no

The most commonly used Variance-based method is Sobol's method. Following the general steps, we first determine the bounds of each variable, which can be done here by directly calling the Python API. After determining the variance bounds, we randomly generate 2000,000 samples that obey a uniform distribution, that is the Sobol's sequence. The Sobol' sequence is a popular quasi-random low-discrepancy sequence used to generate uniform samples of parameter space. For example, the distributions of income and borrowing amount are shown in Figure 1 and Figure 2, respectively.

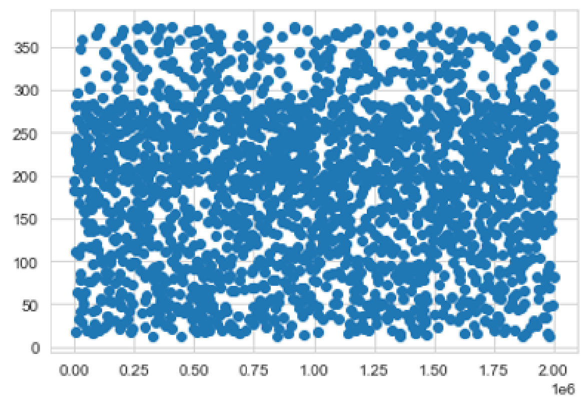


Figure 1 Distribution of the income

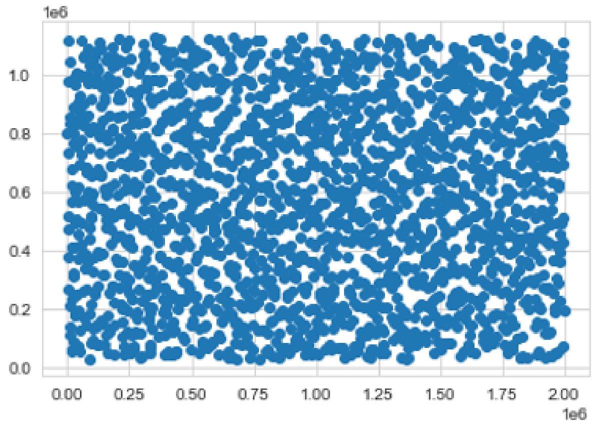


Figure 2 Distribution of the loan amount

The third step is to let the model we have trained predict the generated random samples. We randomly selected 1000 samples and plotted their distributions as shown in Figure 3. It can be seen that the output of most of the samples results in failure and a few in success.

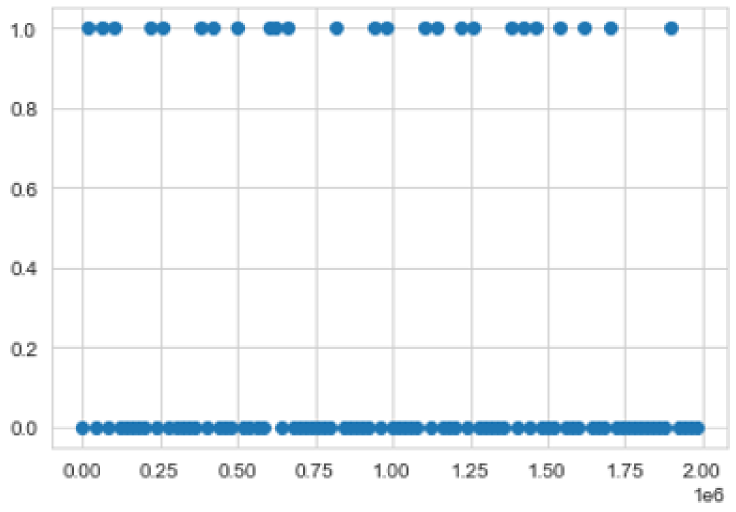


Figure 3 Distribution of the prediction results

The last step is to analyze the relationship between the variables by the prediction results. The specific method is to first calculate the total variance: $V(Y)$, then make Monte Carlo estimates of the conditional variances, and finally calculate the sensitivity indices. This step, we use SALib, a commonly used Python algorithm library for sensitivity analysis, which helps us to automatically calculate Sobol sensitivity index of all orders. The total order and first-order SIs are shown in Table 2.

Table 2 The total order and first-order SI

Variable	ST	ST_conf	S1	S1_conf
>60%	0.922703	0.007250	0.607221	0.009380
income	0.129915	0.003790	0.007862	0.003047
loan_amount	0.161063	0.004570	0.012117	0.003578
property_value	0.257645	0.005087	0.034136	0.003916
open-end_line_of_credit	0.197843	0.004688	0.019708	0.003998
36%-50%	0.066143	0.002689	0.001659	0.002223
co-applicant_credit_score_type_9	0.040258	0.002193	0.000520	0.001736
co_applicant_age_above_62	0.046075	0.002293	0.000509	0.001904
lien_statues	0.040781	0.002055	0.000377	0.001506

Also, we plot the order SIs as histograms, as shown in Figure 4. From the graph, we can see that the variable >60% has the most influence on the final result, followed by property value. In addition, since we are using a global sensitivity analysis, we can derive the relationship between different variables by the 2nd order SI. So by 2nd order SI, we plotted the correlation between each variable as an image, as shown in Figure 5.

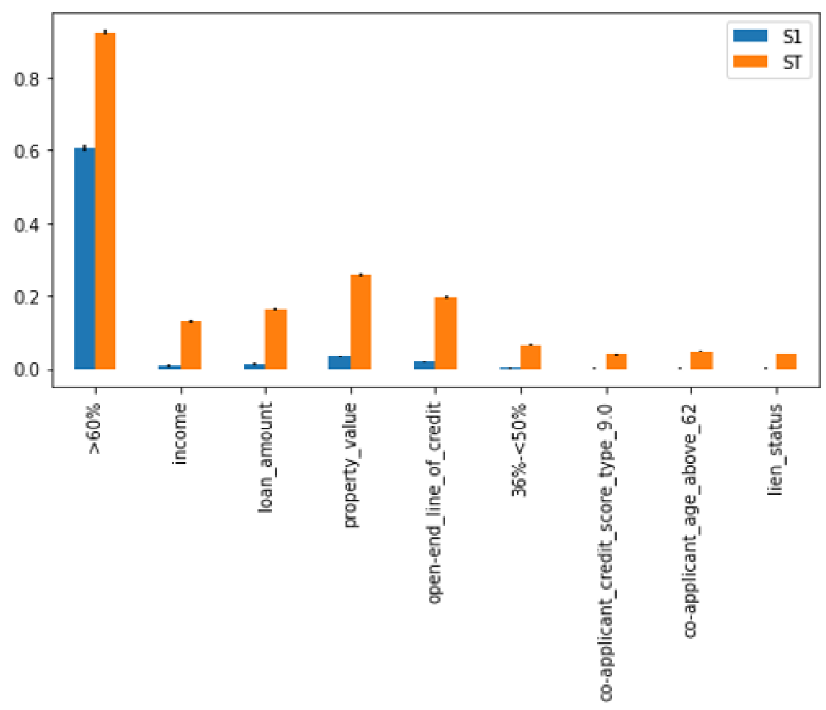


Figure 4 Histogram of the SI

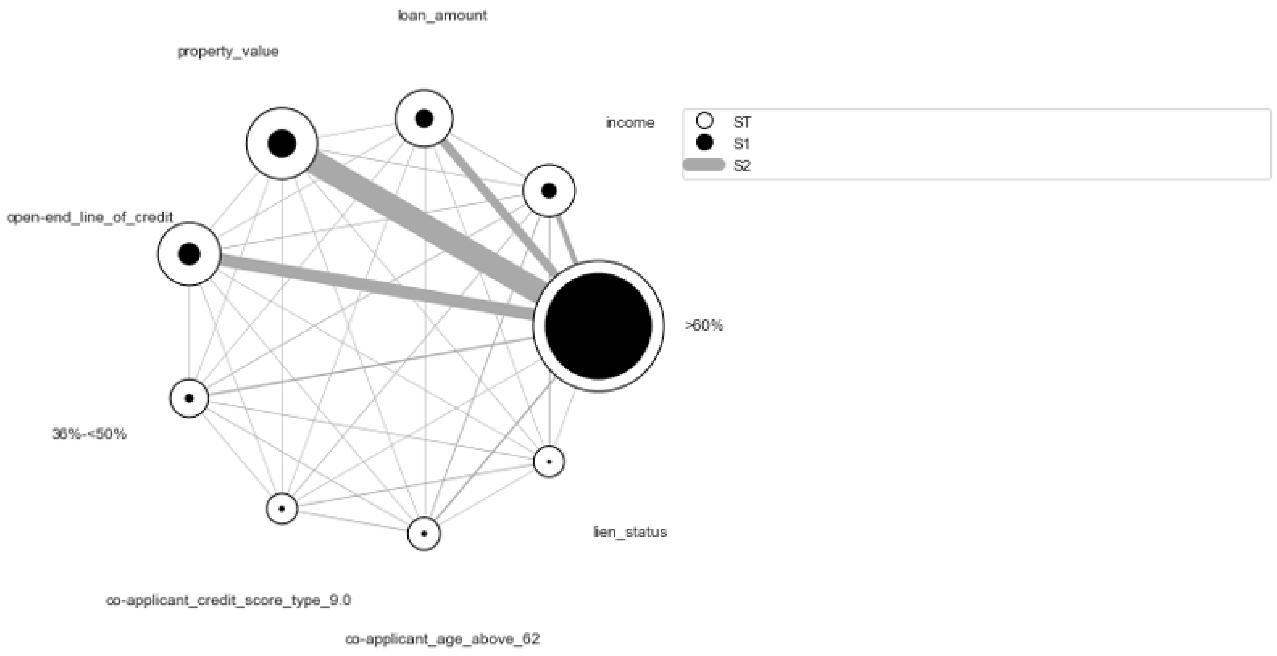


Figure 5 Relationship between each variable

From the figure, we can see that the variable *>60%* is most closely associated with the variable *property value*, followed by *loan amount* and *open-end_lien_of_credit*. while the other variables have relatively weak effects on the final results. Through the relationship between these variables and the final results, in our final APP, we can give some suggestions, such as decreasing the debt to income ratio, selecting the type of loan that is covered by open-end line of credit, and acquiring higher property value, which could potentially help the user to apply for the loan.

Sensitivity analysis is a powerful analytical tool that helps us to better understand the model we get by fitting the data so that the model is no longer a black box. It also helps us to better adjust the model parameters and choose the more appropriate variables among a large number of parameters to make the model achieve better performance. This can better help organizational leaders to make appropriate decisions.

Streamlit Application

By implementing all the models and analysis done before, we designed an interactive loan application evaluation system through Streamlit. The requirements of the system can be basically divided into the following steps: receiving loan application information submitted by the user, predicting the success of probability, and providing suggestions to improve the loan application success rate based on the prediction results and the user's specific loan application information.

According to the baseline of software engineering, we need to conduct requirement analysis first. The first thing is that the user can input the loan application information, and the system needs to receive and process the loan application information, which requires a user interface to be built into our application to receive the user input. Secondly, predicting the application success rate, which is the core function of our application. To fulfill this function, it is time consuming to use the original dataset to retrain the model every time we open the application, so we read in the already trained prediction model and keep it in memory to be called at any time. In this way, once the user submits his or her information, the system can automatically pre-process the input data to fit into the model and output the predicted value. Finally, based on the prediction result and the user's loan application information, we would provide users with possible ways that could improve the success rate of loan application. For this function, we need to subdivide the user input and model output into various cases based on the conclusion obtained from the sensitivity analysis, in order to provide personalized suggestions to the user according to different user input and model output results. The use case diagram is shown in Figure 6.

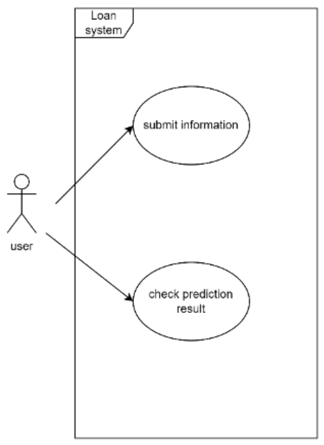


Figure 6 Use case diagram

With the requirements analysis, we now had a general idea of the expected functionalities of the system, so we proceeded to the next step, system design, to depict the system in a more specific way based on the required functions. Since the system does not require to define complex entities or store data, we adopt a process-oriented software design approach here in order to simplify the system design. First, the system is divided into two modules, one is the user loan information processing module, and the other is the model prediction module. The user loan information processing module is used to process the loan application information inputted by users and execute data input and pre-processing, while the model prediction module is used to load the prediction model, perform the prediction, output the prediction result and the corresponding type of recommendation. In the user loan information processing module, a form page is created to guide the users to input and submit their personal information, which supports the pre-processing of data in the system backend. In the model prediction module, model loading needs to be completed at system startup. Then, by retaining the model in computer memory, the model could be automatically implemented to make predictions based on the input data from the user loan information processing module, and finally output the prediction results to the user. A flow chart of the system is shown in Figure 7.

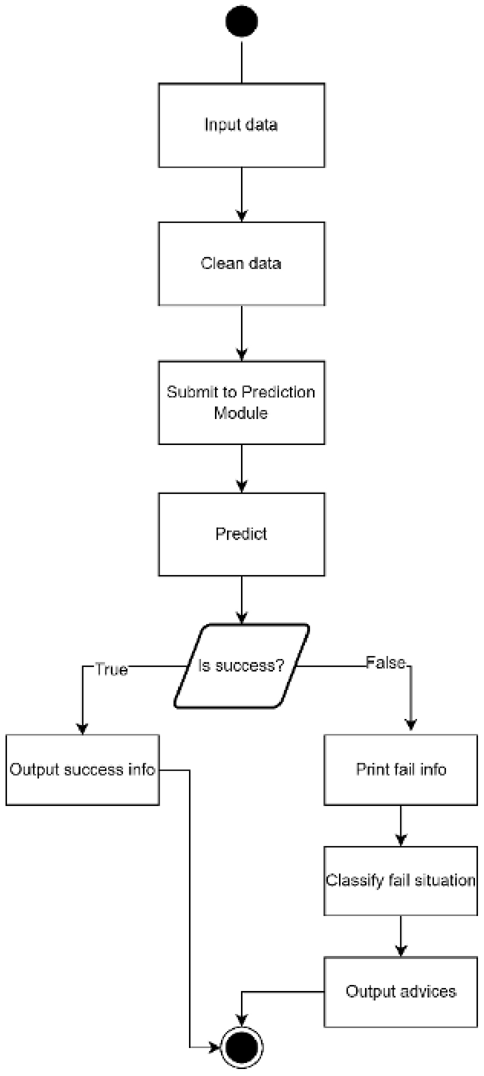


Figure 7 System flow chart

Once the system design framework was finalized, we started to develop the application. The first thing is the technology selection. We chose Python as the development language and completed the whole application development based on the Streamlit framework. Streamlit is an open-source Python library that makes it easy to create and share fancy, custom web apps for machine learning and data science. Through Streamlit framework, we can quickly develop a web application, no longer need to develop the front-end and back-end of the web page separately, which can lower the development difficulty and shorten the development cycle.

During the development process, one technical challenge that we encountered is to perform page jumping, as we expected one page for the user to fill in their personal information and another page to present the prediction result. Since Streamlit is a single-page application framework, it does not provide interfaces for page jumping, but only a sidebar function to switch the functions on the page. Therefore, we implemented the MultiPage class with a sidebar interface, which supports loading different pages and executing different page rendering procedures. The class diagram of the MultiPage class is shown in Figure 8. The variable *pages* is used to store each individual page, which can be defined as a dictionary with titles and rendering functions. When the application is initialized, we first initialize a MultiPage instance, then initialize each page as a Python dictionary type variable storing the title and rendering function, and then add these dictionary type variables to the MultiPage class instance by calling the *add_page* method. The *run* method calls the Streamlit's sidebar API, which returns the page corresponding to the user's click on the sidebar button, and then calls the page rendering function in the run method, which completes the page jumping function.

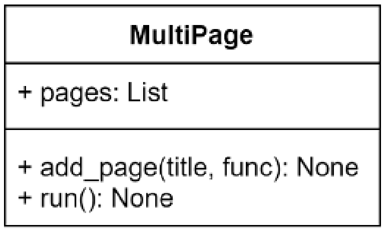
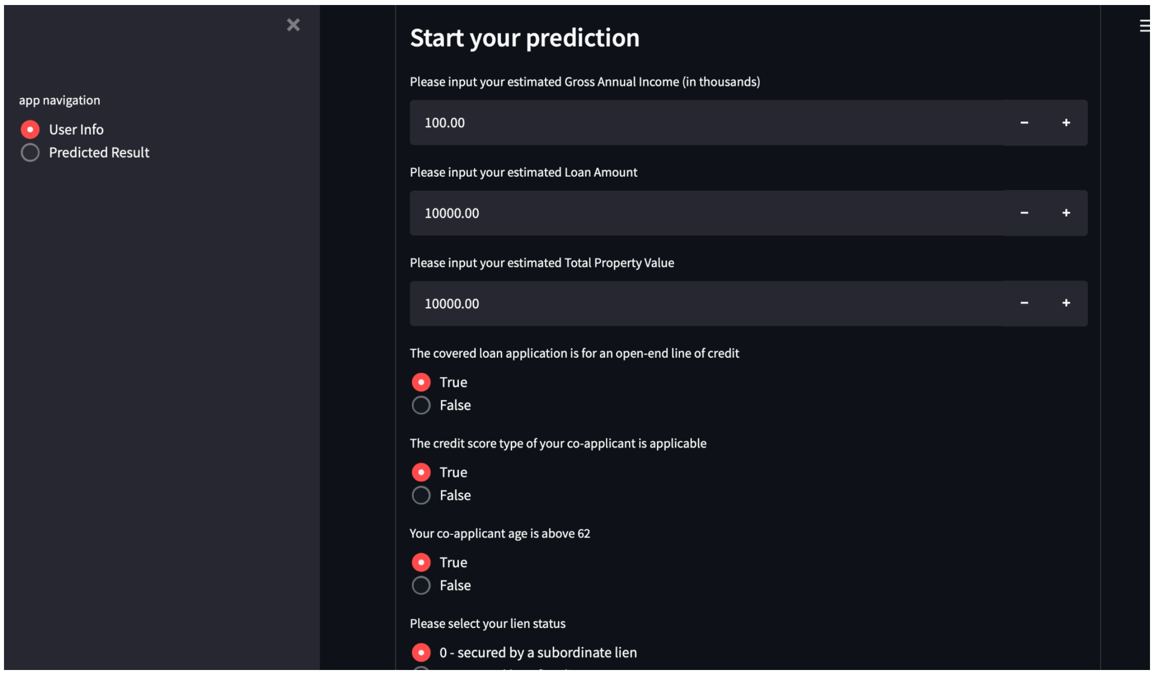


Figure 8 The MultiPage class diagram

However, due to the characteristic of the single-page framework, once the page was switched, all the variables on the previous page would be cleared from memory, which led to data loss in the user loan information processing module. In order to solve this problem, we first implemented a global variable interface module by using Python's native global variable keyword and importing this module on every page to achieve the effect of sharing global variables across multiple pages. However, we found that this method did not achieve the desired effect, and the global variables were also lost after switching pages. By checking the official documentation of Streamlit, we found that Streamlit provides a session API which supports communication between multiple pages. By implementing the API, the memory would not get lost during page break.

The user interface of the finalized application is shown in Figure 9 and Figure 10. In the loan application page, the users first enter the loan information as required and click the submit button. Once the information is correctly input, the system will show a successful submission message and guide the user to switch to the next page in the sidebar to check the predicted result. If the application is successful, the page will indicate that the loan application is successful, and if it fails, the page will show the probability of success and display different suggestions for the user's loan information.


app navigation

[✓ User Info] Predicted Result

Start your prediction

Please input your estimated Gross Annual Income (in thousands)

100.00

Please input your estimated Loan Amount

10000.00

Please input your estimated Total Property Value

10000.00

The covered loan application is for an open-end line of credit

[✓ True] False

The credit score type of your co-applicant is applicable

[✓ True] False

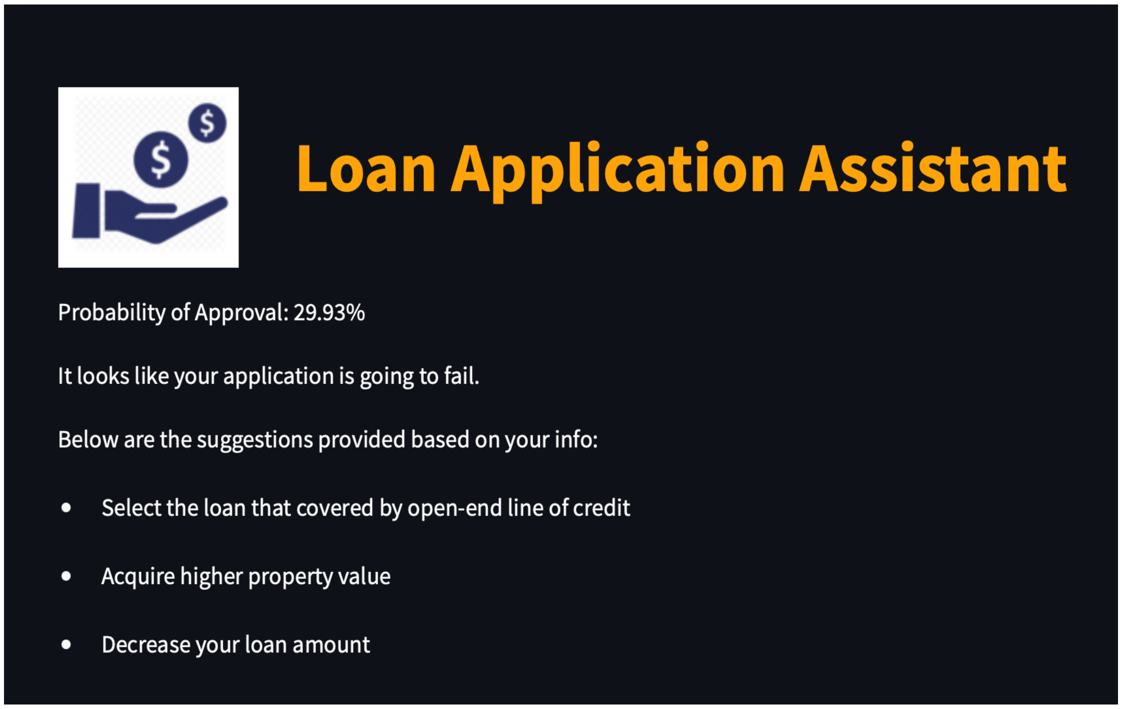
Your co-applicant age is above 62

[✓ True] False

Please select your lien status

[✓ 0 - secured by a subordinate lien]

Figure 9 Application Page1


Loan Application Assistant

Probability of Approval: 29.93%

It looks like your application is going to fail.

Below are the suggestions provided based on your info:

- Select the loan that covered by open-end line of credit
- Acquire higher property value
- Decrease your loan amount

Figure 10 Result Page2

Conclusion

This paper discusses how the finalized logistic regression model and sensitivity analysis result were made. From the stakeholder analysis, the most common needs of users, probability of getting approved and suggestions to increase the chance, were identified, and were transformed into use cases of the website. Those use cases determined the requirement of linear regression model type, logistic regression, and using sensitivity analysis. With constructed model and sensitivity analysis results, the credit application assistant website has been developed successfully. However, there are some limitations to the model due to the time constraint. The accuracy of the logistic model is not significantly high. Although both train and test accuracy are similar and balanced, which means there are no overfitting or underfitting issues, the overall accuracy, approximately 0.76, is not too bad, but is also not considerably high compared to the accuracy of Random Forest Regression using the same features, approximately 0.97, which means current remaining features can play important roles in identifying whether the application would be approved or not. However, the desired prediction result is probability that can’t be obtained from this Random Forest Regression. Further actions, such as more explorations on the features, different feature selection methods, or using k-fold cross validation to find optimal hyperparameters in the logistic regression, can be considered to improve the overall performance of the model.

References

Lusinga, M., Mokoena, T., Modupe, A., & Mariate, V. (2021). Investigating Statistical and Machine Learning Techniques to Improve the Credit Approval Process in Developing Countries. *2021 IEEE AFRICON, AFRICON, 2021 IEEE*, 1–6. https://doi-org.proxy.library.cornell.edu/10.1109/AFRICON51333.2021.9570906