Credit Card and Loan Approval Machine Learning Writeup

Sajid Anjum
Raymond Bell
Han Le
April Gao

Table of Contents:

I. Introduction

In the twentieth century, banking underwent significant changes that made financial services more accessible to more people. Advancements in technology and government policies played key roles in this transformation. Banks began to offer new products and services, such as personal loans and credit cards, that allowed individuals to access credit and manage their finances more easily. The development of automated teller machines (ATMs) made banking more convenient, as people no longer had to wait in long lines to make deposits or withdrawals. Additionally, the creation of online banking platforms and mobile apps further increased accessibility, allowing customers to check their account balances, transfer funds, and pay bills from the comfort of their homes. These advancements helped to democratize banking, making it more accessible to people across different socioeconomic backgrounds.

However, no matter how accessible a bank makes its services, if the banker does not make sound decisions about which services to offer which customers, it will not be able to maintain solvency. Our goal is to develop a machine learning web app that helps a banker make decisions about whether to approve or deny credit card services or loan services. In addition, we would also like to use the predictions of our machine learning algorithm to give customers some more insight about how the bank makes its decisions, and what the bank looks for when it approves or denies customers access to services in a fair and impartial manner.

II. Research Questions

1. Which machine learning algorithm is best suited to correctly predicting whether credit cards are approved based on our data set? What about for Loans?
2. How did we optimize the machine learning algorithm?
3. What were the most important features of our machine learning algorithm, and how do those features affect the probability of credit card approval?
4. Some questions from the dataset:

    a. What is the relationship between home ownership and credit-card/loan approval?
    b. What is the relationship between total income and credit card approval?
    c. How does income vary with marital status?
    d. What are the most common family-sizes for credit card approval?
    e. What are the most common purposes for loan approval?

5. Do our model's predictions match up with trends that we observe in our raw data?

III. Data Cleaning, Processing and Machine Learning in Jupyter Notebook

i. Credit-Card Data Set:

We worked with two different datasets for this project as approval for credit cards and loans require different criteria. The first dataset was sourced from Samuel Cortinhas on Kaggle, and it is about credit card data in China. It consists of 9709 rows of individuals and 20 columns. The dataset was already very clean with no null values. We decided that to drop the `["Work_phone", "Phone", and "Email"]` columns as even though those columns may be predictive in China, we do not think they will have much predictive power in America. We acknowledge that using a Chinese dataset is going to have sub-optimal predictive power for an American clientele, but we feel that for the scope of this project, it is still instructive to carry out the analysis.

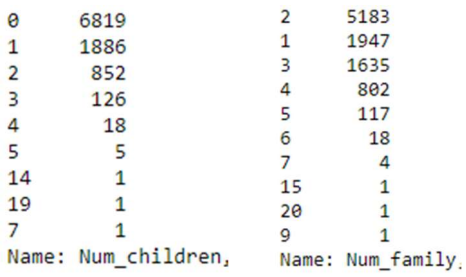
The final columns that we chose were:

```
['ID', 'Gender', 'Own_car', 'Own_property', 'Unemployed', 'Num_children',
'Num_family', 'Account_length', 'Total_income', 'Age', 'Years_employed',
'Income_type', 'Education_type', 'Family_status', 'Housing_type',
'Occupation_type', 'Target']
```

The explanations for all those columns can be found at the Kaggle page for that data:
https://www.kaggle.com/datasets/samuelcortinhas/credit-card-classification-clean-data

In order to process the data for our machine-learning model, we dropped the 'ID' column, and label-encoded the `['Own Car', 'Own Property', 'Unemployed']` columns as only two options were available for each. We employed a standard scaler for the `['Num_children','Num_family', 'Account_length', 'Total_income', 'Age', 'Years_employed']` columns, but it didn't make much difference so we didn't use it in our model. We one-hot encoded the `['Income_type', 'Education_type', 'Family_status', 'Housing_type', 'Occupation_type']` columns. The "Target" column was our prediction column, and we used that to train our classification models.

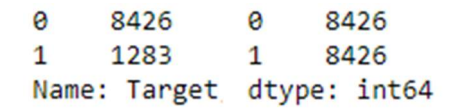
We also modified the 'Num_children' and 'Num_family' columns to collapse amounts of 3 and 4 respectively into the numbers 3 and 4. This is because there wasn't much data for family sizes larger than those numbers and we didn't want to skew our model. We understand that this is a limitation because clearly a family size of 10 is not the same as a family size of 5. However, there is no point in differentiating between the two sizes if our model does not have the data to do so.



```
0     6819
1     1886
2      852
3      126
4       18
5        5
14       1
19       1
7        1
Name: Num_children,
```

```
2     5183
1     1947
3     1635
4      802
5      117
6       18
7        4
15       1
20       1
9        1
Name: Num_family,
```

Our final dataset had 9709 rows and 40 columns. However, our dataset was imbalanced, so we used the SMOTE library to amend our data. The SMOTE library oversamples the imbalanced columns so that there are an equal number of Approved and Denied columns for the model to be trained on. On the right, we can see the value counts of our Target before and after using SMOTE on the dataset.



```
0    8426
1    1283
Name: Target  dtype: int64
```

```
0    8426
1    8426
Name: Target  dtype: int64
```

After using SMOTE, we did a 0.75:0.25 train-test split on our data and used our dataset to train several different classifier machine learning models such as logistic, KMeans, and Tree-based algorithms. We also used a keras based neural network. The models that worked the best were two tree-based algorithms--RandomForestClassifer() and LGBMClassifier(). Both models has comparable ROC curves and confusion matrices, but the LGBM classifier showed less evidence for overfitting. Not only did the neural network model not perform as well, the tree-based algorithms did so

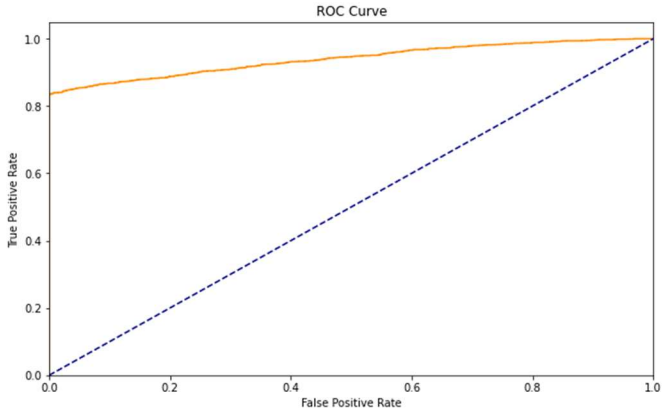
well that we figured the lack of explainability and extra processing power that neural networks demand were not necessary to solve our problem. Below are the results of our LGBMClassifer Model:

```
    METRICS FOR THE TESTING SET:
    ----------------------------
    [[2100    0]
 [ 353 1760]]

                  precision    recall  f1-score   support

           0       0.86      1.00      0.92      2100
           1       1.00      0.83      0.91      2113

    accuracy                           0.92      4213
   macro avg       0.93      0.92      0.92      4213
weighted avg       0.93      0.92      0.92      4213
```




```
AUC for the Model Test Set: 0.9390713947670881
```

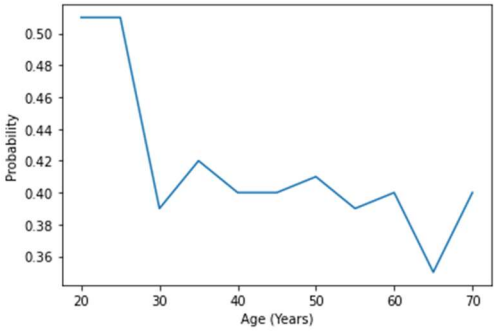
Our two most important numbers were “precision” and “recall” for the target of 1. A precision of 1 means that the model was able to identify every single credit-card default as a credit-card default, which is extremely useful for any bank. A recall of 0.83 means that the model declined 17% of customers who would probably have not defaulted on their loans. This means that it loses the legitimate business of one customer every time it rejects five customers, which is not bad, but also not ideal. Perhaps a higher recall at the expense of a lower precision may boost the profit margin slightly. However, there is no doubt that if our dataset rejects reality, this model is very useful.

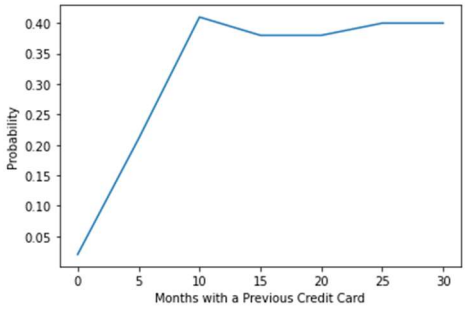
Lastly, we wanted to see which features of our model had the most predictive power. The top ten feature importances on our LGBMClassifer model are show on the right:

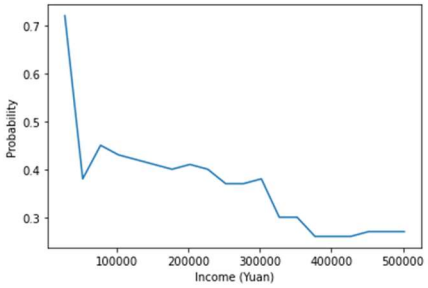
```
0                  (462, Age)
1          (425, Account_length)
2            (358, Total_income)
3          (342, Years_employed)
4                   (96, Own_car)
5           (95, Num_children)
6    (94, Family_status_Married)
7              (85, Num_family)
8                   (84, Gender)
9            (79, Own_property)
10     (70, Income_type_Working)
```

It is no surprise that age, income, employment, and family status are important predictors. What is surprising is that the second most important predictor is the length of time that the person had another credit card.

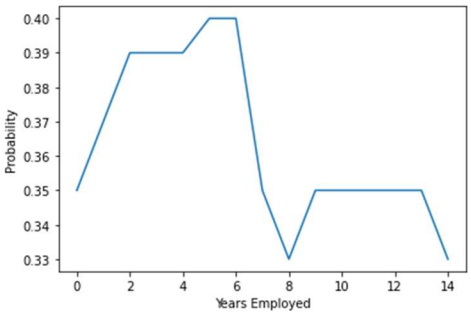
In order to determine how our model’s predictions vary with these features, we created a “most common customer” such that the values for all the numerical columns of this customer were the means of the dataset, and the values for all the categorical columns were modes. Then we varied just the age and saw how the probability varied, and did the same with a few other features. The graphs below are what we saw:

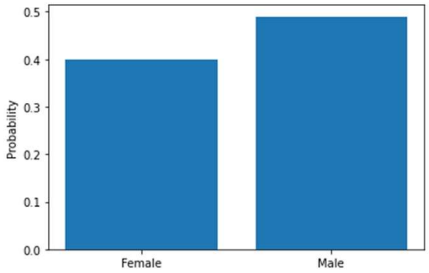


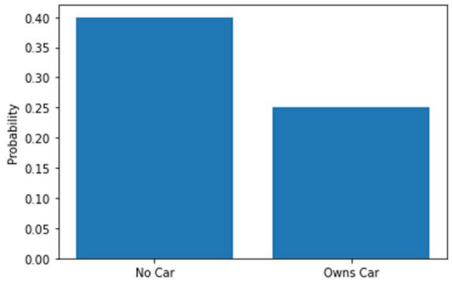




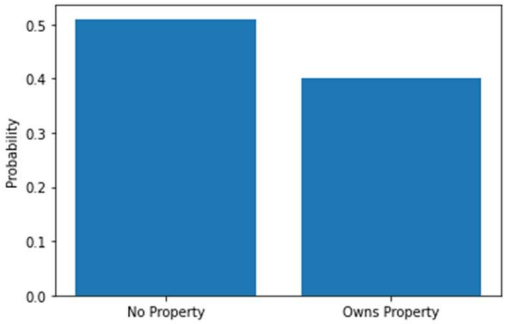
We see that you are less likely to be rejected the older you are. However, the probability of being rejected increases very quickly with the number of months a person has already owned a credit card up to about ten months, after which it flattens out. It seems that banks prefer to give new customers a chance at having a credit card. Lastly, the probability of being rejected falls precipitously with higher income, which is also to be expected.

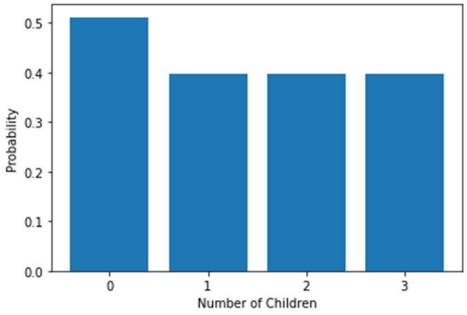






The years employed category varies very strangely with approval rating. The probability of being rejected rises dramatically the more a person works, then falls steeply, then sort of plateaus. We feel this maybe a reflection of the spending habits and application habits of people and how they vary versus how long they have worked. This is an interesting feature that would be difficult to notice without machine learning. The model also identifies males as a riskier investment than females and property and car owners as less risky investments. Lastly, the model does not consider a family with more children to be a riskier investment but does consider a person with no children to be a more risky investment.





ii. Loan Data Set

The second dataset that we worked with was from Arbaz Khan, also on Kaggle, and can be found at this link: https://www.kaggle.com/datasets/arbazkhan971/loan-approval-analysis

This dataset connected the status of a loan (Charged Off, Current, or Fully Paid) to several other facts about the people who took the loan out. We felt that we could use this data to build a machine learning model to predict whether a bank should approve or deny the loan.

This was a large dataset (39,716 rows and 111 columns) and was very unclean. We decided to use this for two reasons—to see whether we could use the large amount of information to make a better model and to take on the challenge of cleaning the data. First, we dropped all columns that had at least 30000 non-null values, which reduced our number of columns to 53. There also weren't very many rows with null values, so we dropped all rows that had null values and ended up with 36,431 rows.

```
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    35365 non-null  int64
 1   loan_amnt             35365 non-null  int64
 2   term                  35365 non-null  int64
 3   int_rate              35365 non-null  float64
 4   installment           35365 non-null  float64
 5   sub_grade             35365 non-null  int32
 6   emp_length            35365 non-null  int64
 7   home_ownership        35365 non-null  object
 8   annual_inc            35365 non-null  float64
 9   verification_status    35365 non-null  object
 10  issue_d               35365 non-null  int32
 11  loan_status           35365 non-null  int64
 12  purpose               35365 non-null  object
 13  dti                   35365 non-null  float64
 14  delinq_2yrs           35365 non-null  int64
 15  inq_last_6mths        35365 non-null  int64
 16  open_acc              35365 non-null  int64
 17  revol_bal             35365 non-null  int64
 18  revol_util            35365 non-null  float64
 19  total_acc             35365 non-null  int64
 20  pub_rec_bankruptcies  35365 non-null  float64
 21  fed_reserve_region    35365 non-null  object
 22  last_credit_pull_d_int  35365 non-null  int64
dtypes: float64(6), int32(2), int64(11), object(4)
memory usage: 5.9+ MB
```

Our goal was to use the 'loan_status' column as the Target for our machine learning model. However, a status of **Current** was irrelevant for our model, and only 1066 rows had the **Current** status, so we dropped those rows. This gave us a final row count of 35365 rows. Our next step was to whittle our columns down to a more manageable number and process the data. We settled on a final amount of 22 columns. They are listed to the right.

The remainder of the cleaning was mostly converting the datatypes from string to integer or float, and from datetime to integer. We made sure that there weren't too many minor categories in the categorical columns, and finally, we converted the states (48 different categories) into Federal Reserve regions (only 12 different categories) to make our dataset more machine-learning friendly.

We were left with only four categorical columns: `['home_ownership', 'verification_status', 'purpose', 'fed_reserve_region']`

We used pd.get_dummies to one-hot encode those columns, dropped our ID column, and our dataset was ready for machine learning.

However, our dataset was imbalanced. As with the credit card data, we used SMOTE to balance our dataset, did a 0.75:0.25 train-test split, and fed the dataset into various machine learning algorithms and neural networks. The best performing model was a RandomForestClassifier. The model was perfect!

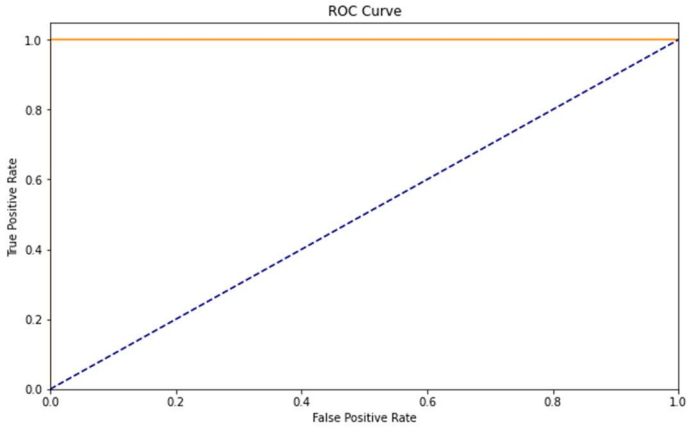
```
0    30423
1     4942
Name: loan_status,
```

```
    METRICS FOR THE TESTING SET:
    ----------------------------
    [[7621    0]
[   0 7591]]

                 precision    recall  f1-score   support

           0       1.00      1.00      1.00      7621
           1       1.00      1.00      1.00      7591

    accuracy                           1.00     15212
   macro avg       1.00      1.00      1.00     15212
weighted avg       1.00      1.00      1.00     15212
```



We think it is very suspicious that the model is so perfect, but this is what we got. Our list of the most important features is given on the right. The most important features were the length of the term, the interest rate, the dates of the last credit pull, the grade of the loan, and the income of the applicant. Unfortunately, we did not have time to do a detailed analysis on how the model varies with these features.

```
0                (0.11057980727851481, term)
1            (0.06764231458233043, int_rate)
2    (0.0474010012311593, last_credit_pull_d_int)
3          (0.04554884183480788, sub_grade)
4          (0.03868523631836471, annual_inc)
```

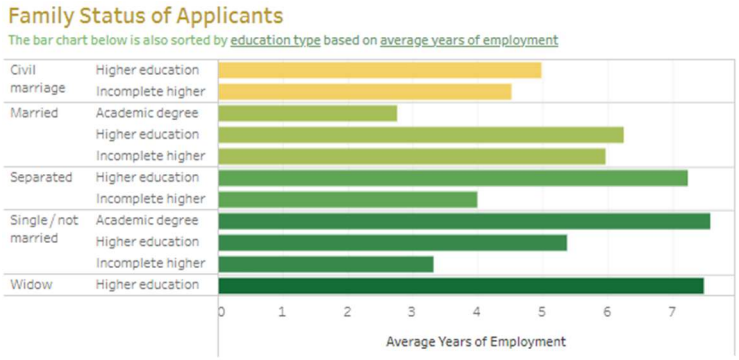
IV. Tableau

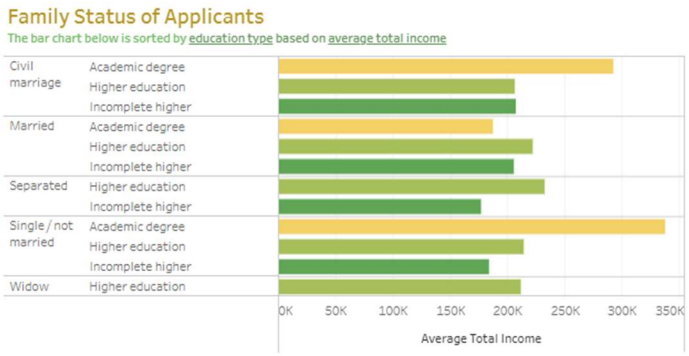
We created a total of four dashboards, two for each dataset.

i. Credit Card Dashboards

First dashboard: This dashboard explores the connections between marital status, education, number of years of employment, and the average total income to see how these are related within the group of people who apply for credit cards.

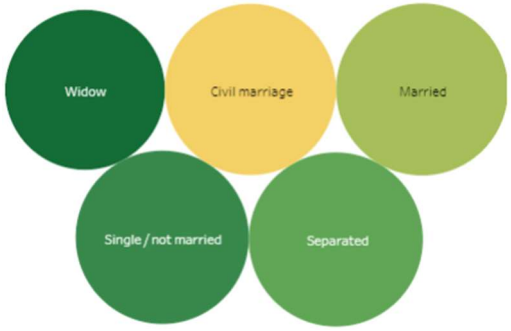
We created two horizontal bar charts that connected marital status and education to length of employment and average income respectively.

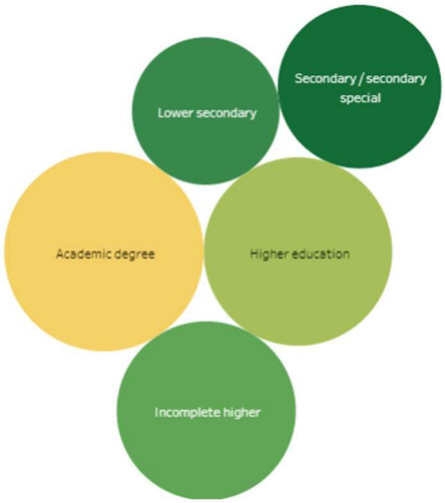




Some key trends:

1. The widowed tend to have worked for a very long time, tend to have very high education, and have a solid middle-class income. This is not surprising as you would expect the widowed to be older than the average population.
2. Single people and people involved in a civil marriage and with an academic degree tend to have the highest incomes. Since this dataset is taken from China, we think that people adopting for traditional marriages would more likely be away from modern city centers and marry earlier, while those opting for civil marriages or remaining single would more likely be closer to modern Chinese metropolises and thus have higher incomes. Thus, it seems that adding location to this dataset could be very revealing.
3. The bubble chart provides a good way to quickly look at the relative sizes of the different categories of education level and marital status within our dataset. Since the bubbles are mostly the same size, this dataset is very good for using as a machine learning model.





We are able to change within the two bubble categories by using a parameter filter converting “Education Type” to “Family Status”. However, our captions on our dashboard are only for education status. We will fix this in future.

4. From the bar charts, we can also see that the candidate with an academic degree makes the highest income within the group and almost double the income of the candidate who did not complete their education.

Second Dashboard:

Our second dashboard continues with the theme of our first board in relating marital status and income, but this time using a boxplot to see the distributions. We also have a few filters on the right where we sort by.

Application Status: we can choose to see approved only, or denied only, or both together.

Average Income: we can show all the values or narrow down the range.

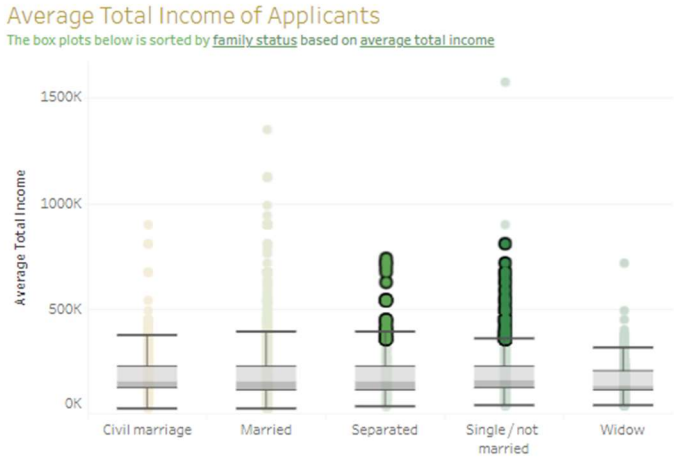
Family Size: we can choose the exact family size or a range of family sizes to review all the sizes on the same visualization.

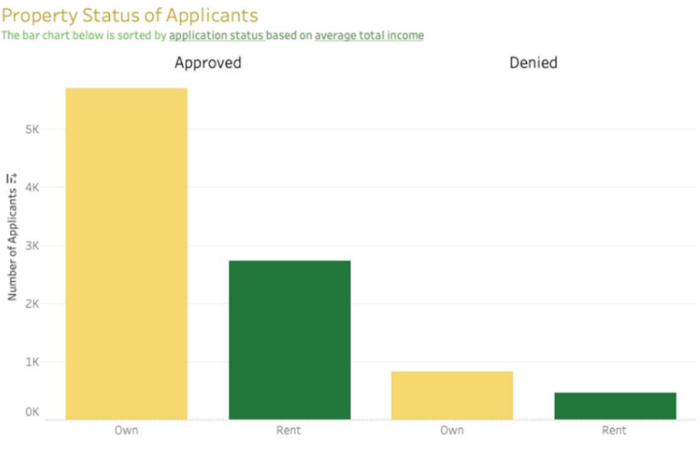
Family Status: we can select all or only candidates that are married, single, separated, or widow.

Ownership: we can choose both or either whether the candidates own properties or not.

We notice two things from the boxplot:

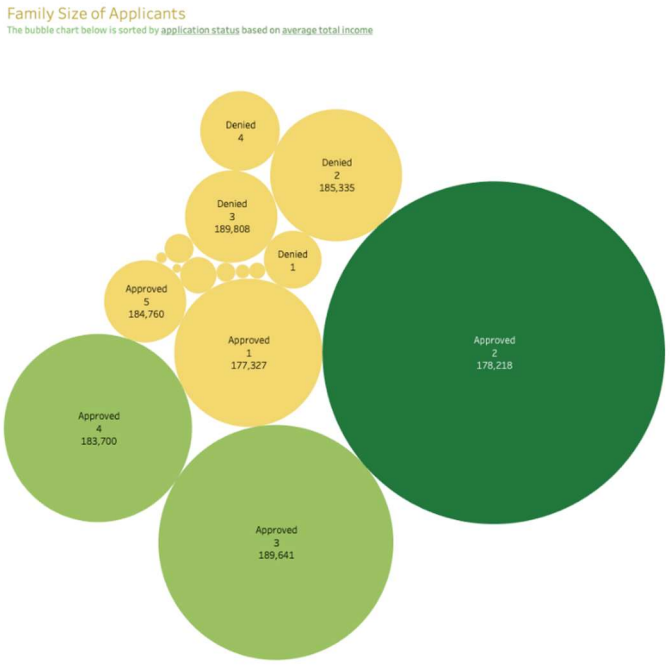
a. The average incomes of applicants in traditional and civil marriages are similar, while applicants in a traditional marriage are more likely to have very high salaries. It seems that unadjusted for education, there isn't much difference between being involved in a traditional or civil marriage.
b. Aside from outliers, all the boxplots are of a similar size and shape, showing that this data is very suitable for using to make a machine learning model.





We also sorted applicants by whether they owned or rented property. The data shows that of those who applied for a credit card, many more people own property than rent, but the percentage of approval is not really that different.

Lastly, we made a bubble chart comparing family size to income and number of applicants. The bubble chart is sorted by the application status (approved vs. denied) based on the applicant's average Income and family size. The bubble size reflects the number of candidates. The bigger the bubbles are, the more applicants are approved or denied. Using the family size of 2,3, and 4 (most giant bubbles), more candidates receive approvals than denials for their credit applications. For those with the same family size, those receiving approvals are making less than those who received denials. A few key points concluded from the bubbles are that Income is not the sole determinant driving the decision, as even though you can make good money, there is a chance you get denied. There must be other criteria factoring into the decision on credit card applications. Moreover, the family size seems irrelevant to the decision.



ii. **Loan Dashboards**

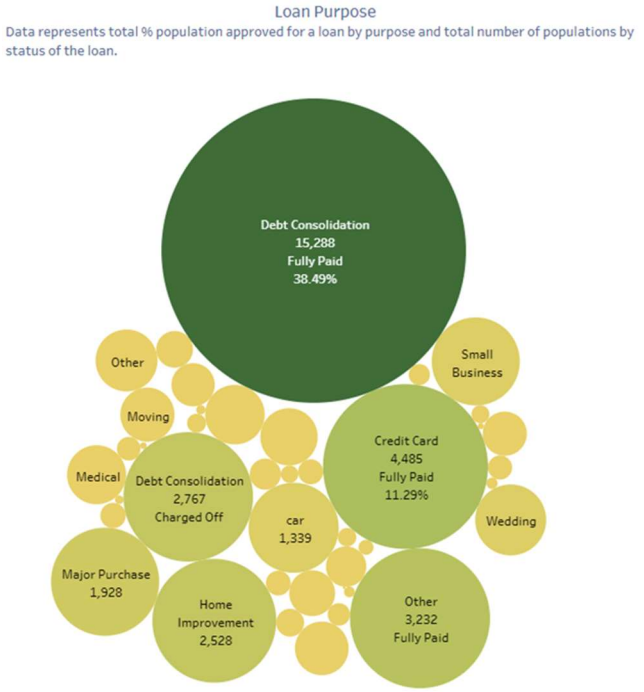
We created three Tableau Dashboards for the Loan application dataset.

Dashboard one: This is a combination of bubble charts and bar charts showing the relationship and the driving factors of loan Status, Loan Amount and Loan Purpose.

The bubble chart “Loan Purpose”:

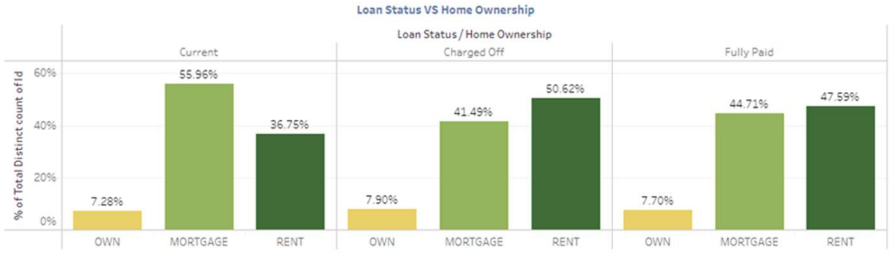
We grouped the applicant by loan purpose then divided by total number of the applicants. It returned the percentage of the loan total by purpose. We then filtered the chart by loan status.

We notice that most people take on loans for debt consolidation or to pay off credit cards, which really seems to be a very similar thing.



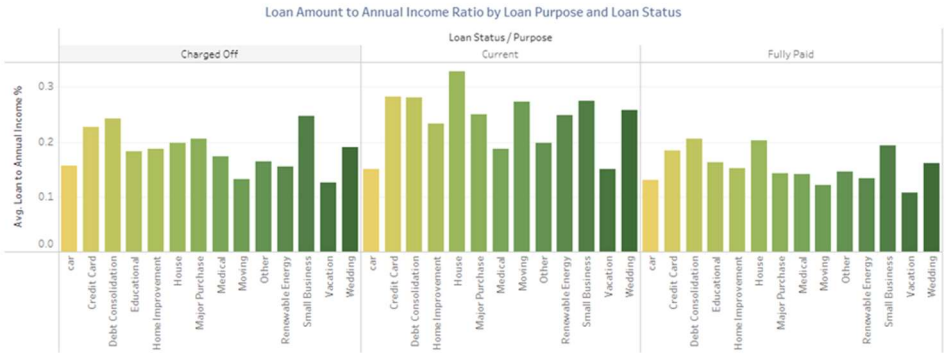
Bar Chart “Loan Status VS. Home ownership”

This chart breaks down the loan applicants by their Home Ownership (own, Mortgage, or Rent) then filtered by their loan status. With this chart, we can see people who own their house not necessarily most current or less likely to default on a Loan. Mortgage applicants are 56% current 42% Charged off and 45% Fully paid on their loan. Homeowners are about 7%-8% regardless of the loan status..

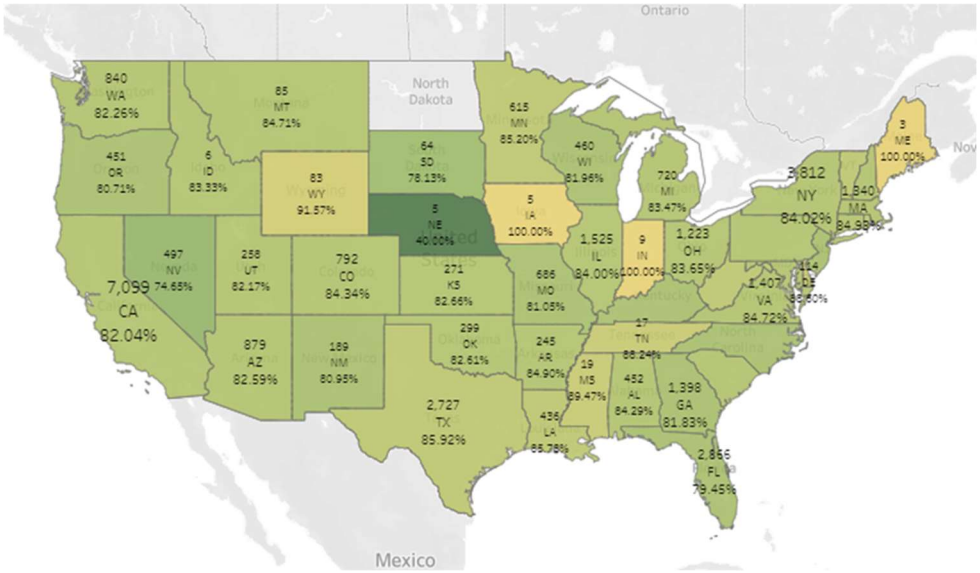


Bar Chart: “Loan Amount to Annual income Ratio by Loan Purpose and Loan Status”

This is chart shows some interesting aspects of the loan applicants. We are trying to see whether the ratio of total debt to annual income truly plays a role in determining whether a loan is successfully paid off. Small business loans, debt consolidations, and credit card debt makes up most of the applicants, but their debt to income ratio is also high. However, this is also the same population that has their loan fully aid. From this dataset we can say debt to income ratio not the golden driving factor in order to determine if a loan is fully paid.



The second dashboard is a map of the loan applicant population by state. It shows the total number of loan applicants in each state as well as the percentage of how many loans are successfully paid off. The color of the map is scaled by the percentage of fully paid to state total. It also shows the charge off, current, and fully paid percentage by state. We notice that applicants in most states successfully pay off between 80-85% of their loans, which means that if our model successfully predicts more than this percentage, we may have a future in the banking industry. When comparing states with more than 500 applicants, Texans appear to be the best at fully paying off their debt with an 85.92% success rate.

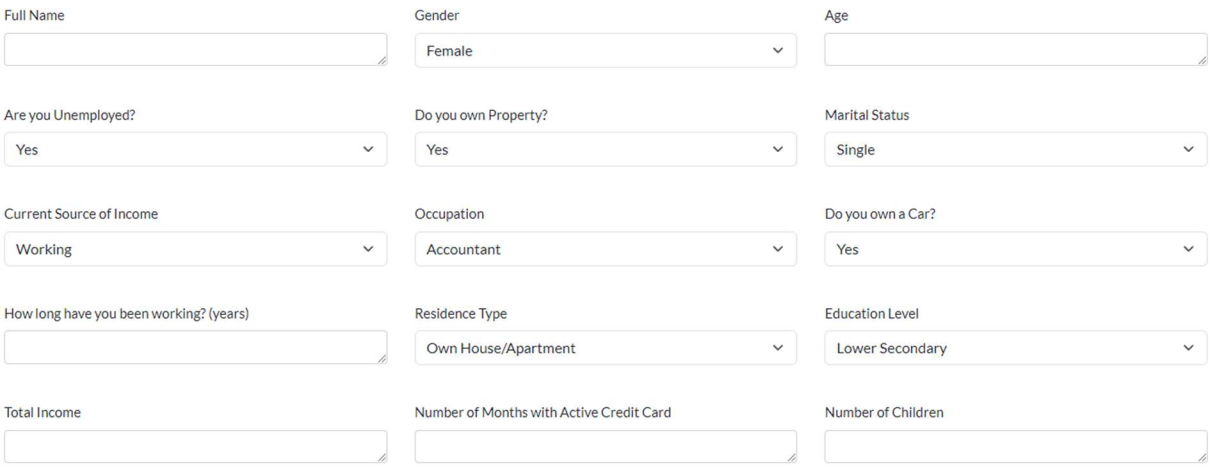


The third dashboard is also a map. The color of the map only depends on the total number of applicants who are successful or who fail, which is not that interesting. However, the tool tip gives the user a chance to scroll over the map and instantly visualize key data points by state. It would have been nice to be able to combine both maps into one, but it was difficult to process the data on Tableau, so we did the percentage map processing on Python and uploaded a new dataset to create that map.

V. Website App

We used a Flask App with a Python back end to build our website. Our website has five main sections and eleven pages in all. We designed our website on a Bootstrap grid with a Bootswatch theme.

a. **Home Page:** The home page introduces our bank and allows you to quickly connect to the forms and Tableau dashboards web pages. The About, DataTables, and References pages can be accessed from the Navigation Bar.
b. **Forms:** We created two forms—one for credit card approval and one for Loan Approval. The forms were built upon bootstrap and take the inputs are values in our datatable. We did not build error corrections into our forms, so the data that we enter into the forms has to be in the correct format and within the correct range. If we had time, we could have built this into the data input. Our form looks like this:



After you press the submission button (not shown), the data is read into JavaScript using the D3 library. We then use Ajax from JQuery to send the data to Flask, where the data is read in as a dictionary from a JSON format. The values of the dictionary are then converted into a DataFrame, and the datatypes and values are changed to reflect those in the dataset. We then one-hot encode the categorical columns, add the missing categorical columns, set those values to 0, and reorder the columns so that they will match the order that is needed for the machine learning model. Finally, we unpickle our LGB model, feed the data into it, obtain a prediction and a probability, and feed that back to the website to get a probability.

Unfortunately, we did not have a chance to connect the loan data to the machine learning model that we built.

c. **Tableau**: We created four Tableau dashboards—two for the credit card, one for the loan data, and one map. I used Tableau's API to load the dashboards onto our website.
d. **Data Tables:** Our data tables for both data sets were really large, so we took a random sample of 1000 rows from each dataset. Then, I used the DataTable library from JavaScript to upload the data to our website
e. **About Us**: This section consists of the pictures from everyone in our group and some brief information about who we are.
f. **References:** This section talks about where we obtained our datatables from and points the reader to some links for more research
g. **Machine Learning Model:** This model talks a little bit about how the machine learning model works, how it makes predictions, and what the applicant should do so that they will have better luck next time.

VI. Conclusions and Future Work:

Conclusions:

1. Which machine learning algorithm is best suited to correctly predicting whether credit cards are approved based on our data set? What about for Loans? How did we optimize the machine learning models

   The tree models worked the best for the classification algorithms that we made. For credit cards, both the LGBMClassifier and the RandomForestClassifier gave spectacular results, but we picked the LGBMClassifier because it showed less evidence of overfitting. We did not try to optimize the algorithms too much, or even to use neural networks as the extra effort did not seem worth it based on how well the models are already working.

   For our loan data, the RandomForestclassifier fit perfectly, so we did not need to check any other models.

2. What were the most important features of our machine learning algorithm, and how do those features affect the probability of credit card approval?

   Age, number of months with a previous credit card, total income, and total years of employment are not surprisingly the categories that had the largest impact on the approval algorithm. For more details on how the probability of approval is affected by variation in these columns, please see section **III.**

3. Some questions from the dataset:

   a. What is the relationship between home ownership and credit-card/loan approval?
   While the majority of applicants were homeowners, it turns out that home ownership did not have a massive impact on approval.

   b. What is the relationship between total income and credit card approval?
   This was one of the most important factors, but very far from being determining.

   c. How does income vary with marital status?
   The average income does not vary much, but high earners tend to be either married or single. It is rare for separated or widowed people to be very high earners.

   d. What are the most common family-sizes for credit card approval?
   A family size of two was the most common. We think they are mostly young couples.

   e. What are the most common purposes for loan approval?
   Most people want loans for debt consolidation or move credit card debt into a loan debt.

4. Do our model's predictions match up with trends that we observe in our raw data?

   Our model shows that home-ownership, income, and family size all affected the probability of approval, but no single variable was the most determining factor. It turns out that using machine learning algorithms makes a big difference toward making predictions in this field, as even though a single factor does not determine whether a credit card or loan is approved, when we combine all the factors we can get a very good idea about what type of person is likely to successfully pay off their cards and loans.

Limitations and Future Work:

1. The biggest limitation was time. This was a very complex project, and we could easily have spent one other week on it to apply all the skills that we learnt in this course. As we mentioned earlier, we were not able to make a live webapp deployment of the loan data model even though we were able to create the model.

2. One of the reasons we chose the loan dataset was that it had location data. Even though we constructed maps, we would like to delve deeply into the location data to show how variation in location or type of environment (city/suburb/rural) connects with access to finance and ability to pay off debt.

3. The machine learning models ran very quickly despite the fact that they were trained on very large datasets. It is clear that there is room for much more data to make even better predictions. The better the predictions that banks can make, the more likely it is that every single person will have access to finance, and eventually, this could lead to banks personalizing loans and credit cards for perhaps every single individual in a way that would make defaulting on a credit card or a loan a very rare occurrence. We feel that the ability to get financing is imperative to go ahead with your ambitions in America, and every single person should have access to financing if they can pay it back responsibly and use it effectively.

4. Lastly, we would like to see if counseling or having access to a financial planner can lead to better decision making and making smarter choices with access to finance.

Acknowledgments: We would like to thank Professor Alexander Booth, Professor Farzad HosseinAli and SMU for giving us a wonderful bootcamp experience over the past six months. We feel that we have learned an amazing number of skills that seem especially relevant to the modern world and were instrumental in helping us create this machine learning web application. We would also thank all the other members of our cohort for creating a friendly and welcoming learning environment.