Project Title: Forecast of Credit Card Approval Based on Vintage Analysis

Authors: Miaoting Feng, Can Song
Team: Crazy Thursday

1. -Executive Summary-

1.1 Decisions to be impacted and motivation

In general, scores of credit card applicants are calculated based on historical data collected from former clients. Once encountering large economic fluctuations, past models may lose their original power of prediction. The first decision that might be impacted is the results of applying for credit cards made by banks and organizations. Then a series of decisions related to credit systems, financial organizations and clients could receive further influence.

1.2 Business value

Instead of using human labor to manually accept or reject a credit card application, building predictive frameworks could help reduce large amounts of resources and save time and costs for companies and organizations. Besides, the establishment of the credit card approval prediction model and the exploration of the key factors influencing the customer credit not only contribute to guiding financial institutions to better choose customers and design data, but also can provide certain theoretical support for the credit decisions. In addition, it has a strong theoretical and practical significance.

2. -Data description/preprocessing

2.1 Data description

2.1.1 Data assets

The data set used in our project was released publicly on Kaggle and last updated two years ago in 2020. It has two tables that are connected by client number: application record and credit record.

The former table of application record mainly contains information about clients' assets, families, jobs, and personal lives and involves 18 features in total. The latter table of credit records is about the status of clients' balance in different months with 3 features. In other words, we could say that the first table would be the one with independent variables, which are also noted as the input features in our project. The second table contains the historical status of all clients, which is the dependent variable of "good" or "bad".

ID	CODE_GE	FLAG_OW	FLAG_OW	CNT_CHIL	AMT_INC(	NAME_IN(	NAME_ED	NAME_FA	NAME_HC	DAYS_BIR	DAYS_EM	FLAG_MO	FLAG_WO	FLAG_PH(	FLAG_EM/	OCCUPAT	CNT_FAM_
5008804	M	Y	Y	0	427500	Working	Higher ed	Civil marri	Rented ap	-12005	-4542	1	1	0	0		2
5008805	M	Y	Y	0	427500	Working	Higher ed	Civil marri	Rented ap	-12005	-4542	1	1	0	0		2
5008806	M	Y	Y	0	112500	Working	Secondary	Married	House / a	-21474	-1134	1	0	0	0	Security st	2
5008808	F	N	Y	0	270000	Commerci	Secondary	Single / n	House / a	-19110	-3051	1	0	1	1	Sales staff	1
5008809	F	N	Y	0	270000	Commerci	Secondary	Single / n	House / a	-19110	-3051	1	0	1	1	Sales staff	1
5008810	F	N	Y	0	270000	Commerci	Secondary	Single / n	House / a	-19110	-3051	1	0	1	1	Sales staff	1
5008811	F	N	Y	0	270000	Commerci	Secondary	Single / n	House / a	-19110	-3051	1	0	1	1	Sales staff	1
5008812	F	N	Y	0	283500	Pensioner	Higher ed	Separated	House / a	-22464	365243	1	0	0	0		1
5008813	F	N	Y	0	283500	Pensioner	Higher ed	Separated	House / a	-22464	365243	1	0	0	0		1
5008814	F	N	Y	0	283500	Pensioner	Higher ed	Separated	House / a	-22464	365243	1	0	0	0		1
5008815	M	Y	Y	0	270000	Working	Higher ed	Married	House / a	-16872	-769	1	1	1	1	Accountar	2
5112956	M	Y	Y	0	270000	Working	Higher ed	Married	House / a	-16872	-769	1	1	1	1	Accountar	2
6153651	M	Y	Y	0	270000	Working	Higher ed	Married	House / a	-16872	-769	1	1	1	1	Accountar	2
5008819	M	Y	Y	0	135000	Commerci	Secondary	Married	House / a	-17778	-1194	1	0	0	0	Laborers	2
5008820	M	Y	Y	0	135000	Commerci	Secondary	Married	House / a	-17778	-1194	1	0	0	0	Laborers	2
5008821	M	Y	Y	0	135000	Commerci	Secondary	Married	House / a	-17778	-1194	1	0	0	0	Laborers	2
5008822	M	Y	Y	0	135000	Commerci	Secondary	Married	House / a	-17778	-1194	1	0	0	0	Laborers	2
5008823	M	Y	Y	0	135000	Commerci	Secondary	Married	House / a	-17778	-1194	1	0	0	0	Laborers	2
5008824	M	Y	Y	0	135000	Commerci	Secondary	Married	House / a	-17778	-1194	1	0	0	0	Laborers	2
5008825	F	Y	N	0	130500	Working	Incomplet	Married	House / a	-10669	-1103	1	0	0	0	Accountar	2
5008826	F	Y	N	0	130500	Working	Incomplet	Married	House / a	-10669	-1103	1	0	0	0	Accountar	2
5008830	F	N	Y	0	157500	Working	Secondary	Married	House / a	-10031	-1469	1	0	1	0	Laborers	2
5008831	F	N	Y	0	157500	Working	Secondary	Married	House / a	-10031	-1469	1	0	1	0	Laborers	2
5008832	F	N	Y	0	157500	Working	Secondary	Married	House / a	-10031	-1469	1	0	1	0	Laborers	2
5008834	F	N	Y	1	112500	Working	Secondary	Single / n	House / a	-10968	-1620	1	0	0	0		2
5008835	F	N	Y	1	112500	Working	Secondary	Single / n	House / a	-10968	-1620	1	0	0	0		2
6153712	F	N	Y	1	112500	Working	Secondary	Single / n	House / a	-10968	-1620	1	0	0	0		2
5008836	M	Y	Y	3	270000	Working	Secondary	Married	House / a	-12689	-1163	1	0	0	0	Laborers	5
5008837	M	Y	Y	3	270000	Working	Secondary	Married	House / a	-12689	-1163	1	0	0	0	Laborers	5

ID	MONTHS	STATUS
5001711	0	X
5001711	-1	0
5001711	-2	0
5001711	-3	0
5001712	0	C
5001712	-1	C
5001712	-2	C
5001712	-3	C
5001712	-4	C
5001712	-5	C
5001712	-6	C
5001712	-7	C
5001712	-8	C
5001712	-9	0
5001712	-10	0
5001712	-11	0
5001712	-12	0
5001712	-13	0
5001712	-14	0
5001712	-15	0
5001712	-16	0
5001712	-17	0
5001712	-18	0
5001713	0	X
5001713	-1	X
5001713	-2	X
5001713	-3	X
5001713	-4	X
5001713	-5	X

Figure 2.1-1 Table of Application Records (left) and Credit Records (right)

Shown in Figure 2.1-1 are part of the two tables from the raw data set.

2.1.2 Summary of datasets

The data set used in our project was released publicly on Kaggle and last updated two years ago (Song 2020). It has two tables connected by client number: application record and credit record with 18 features in the former table and 3 in the latter. Application records mainly contain information about clients' assets, families, jobs, and personal lives. The other is about the status of clients' balance. Below are the detailed features from two tables with the explanations (Table 2.1.2-1).

Table 2.1.2-1 Explanations of Features

Feature name	Explanation
ID	Client number
CODE_GENDER	Gender
FLAG_OWN_CAR	Is there a car
FLAG_OWN_REALTY	Is there a property
CNT_CHILDREN	Number of children
AMT_INCOME_TOTAL	Annual income
NAME_INCOME_TYPE	Income category
NAME_EDUCATION_TYPE	Education level
NAME_FAMILY_STATUS	Marital status
NAME_HOUSING_TYPE	Way of living
DAYS_BIRTH	Birthday
DAYS_EMPLOYED	Start date of employment
FLAG_MOBIL	Is there a mobile phone
FLAG_WORK_PHONE	Is there a work phone
FLAG_PHONE	Is there a phone

FLAG_EMAIL	Is there an email
OCCUPATION_TYPE	Occupation
CNT_FAM_MEMBERS	Family size
MONTHS_BALANCE	Record month
STATUS	Status

It is apparent that merely using manual efforts to classify a client is “good” or “bad” is impractical. Also, it may be difficult to provide customers and regulators with a reason for rejection or acceptance in this way. Constructing the predictive models based on historical data would not only help develop the analytics but also give more convincing results for credit card application.

2.2 Data preprocessing

Data cleaning is the process of correcting and deleting inaccurate records from a database or table. Broadly speaking, it consists of identifying and replacing incomplete, inaccurate, irrelevant, or otherwise problematic data and records.

Generally, organizations agree that data quality falls into six core dimensions or characteristics: data completeness, data accuracy, timelineness, uniqueness, consistency and validity. With effective cleaning methods applied, all data sets should be consistent and free of any errors that could be problematic during later use or analysis.

Observing our dataset of credit card approval, it is clear that the dataset is extracted from real-life cases and has no missing values.

However, we could see that this dataset has the data imbalance problem, which is common in this kind of binary classification dataset. We will deal with it in the model update part before constructing usable machine learning models. The method we will use is called Synthetic Minority Oversampling Technique, or SMOTE for short.

2.2.1 Vintage analysis

Based on the dataset of “credit record”, we construct vintage analysis to create labels. More detailly, we analyzed the overdue situation of more than 60 days and calculated the month on book (MOB) of each client. If one client has passed his/her due for over 60 days, he/she would be considered as “overdue”. Then we computed the overdue rate and made sure it was monotonically increasing.

Respectively, we analyze 1 day past due, 20 days past due, 60 days past due, 90 days past due, 120 days past due, 150 days past due. We could see that almost 87% of users have past due more than 1 day, which is too common, thus it's inappropriate to be a standard. Only 0.4% of accounts appear to have passed due more than 150 days. If we use that, we will leave many bad customers in our scrutiny.

A table was drawn to help determine what will be the most suitable standard of bad customers.

Table 2.2.1-1 Standard of Bad Customers

	situation	**bad customer ratio**

0	past due more than 1 day	0.87054
1	past due more than 30 days	0.11634
2	past due more than 60 days	0.0145
3	past due more than 90 days	0.0072
4	past due more than 120 days	0.00528
5	past due more than 150 days	0.00424

We also drew a graph of the vintage line chart, showing the cumulative overdue rate of more than 60 days.

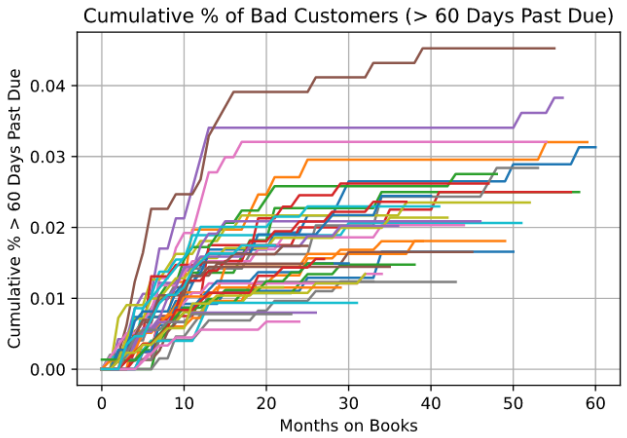

Figure 2.2.1-1 Graph of vintage analysis

2.2.2 Basic analysis of given features

1) Binary features

First is the binary features, which are “Gender”, “Having a car or not”, “Having house reality or not”, “Having a phone or not”, “Having an email or not” and “Having a Work phone or not”. We did some basic analysis, such as the number of “bad” or “good”, the share of each category, the distribution of “good” and “bad”, and finally the result is shown in the table.

Table 2.2.2-2 Table of “Gender”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	Gender	0	6500	6322	178	0.618341	0.027385	0.619804	0.570513
1	Gender	1	4012	3878	134	0.381659	0.0334	0.380196	0.429487

Table 2.2.2-3 “Having a car or not”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	Car	0	5910	5733	177	0.562215	0.029949	0.562059	0.567308
1	Car	1	4602	4467	135	0.437785	0.029335	0.437941	0.432692

Table 2.2.2-4 “Having house reality or not”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	Reality	0	3826	3693	133	0.363965	0.034762	0.362059	0.426282
1	Reality	1	6686	6507	179	0.636035	0.026772	0.637941	0.573718

Table 2.2.2-5 “Having a phone or not”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	phone	0	7364	7155	209	0.70053	0.02838	0.701471	0.669872
1	phone	1	3148	3045	103	0.29947	0.03272	0.298529	0.330128

Table 2.2.2-6 “Having an email or not”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	email	0	9462	9180	282	0.90011	0.0298	0.9	0.903846
1	email	1	1050	1020	30	0.09989	0.02857	0.1	0.096154

Table 2.2.2-7 “Having a Work phone or not”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	wkphone	0	7549	7340	209	0.71813	0.02769	0.719608	0.669872
1	wkphone	1	2963	2860	103	0.28187	0.03476	0.280392	0.330128

2) Continuous features

Besides, we construct the analysis of the continuous features – “Children numbers”, “Annual income”, “Age”, “Working years”, and “Family size”. The following tables and figures show the basic analysis results.

Table 2.2.2-8 Table of “Children Numbers”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	ChldNo	0	6718	6515	203	0.639079	0.030217	0.638725	0.650641
1	ChldNo	1	2463	2395	68	0.234304	0.027609	0.234804	0.217949
2	ChldNo	2More	1331	1290	41	0.126617	0.030804	0.126471	0.13141

Table 2.2.2-9 “Family Size”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distributio n Good	Distribution Bad
0	famsizegp	1	1597	1539	58	0.151922	0.036318	0.150882	0.185897
1	famsizegp	2	5515	5355	160	0.524639	0.029012	0.525	0.512821

2	famsizegp	3more	3400	3306	94	0.32344	0.027647	0.324118	0.301282

For visualization, histograms are drawn to show the frequency of the rest categories.

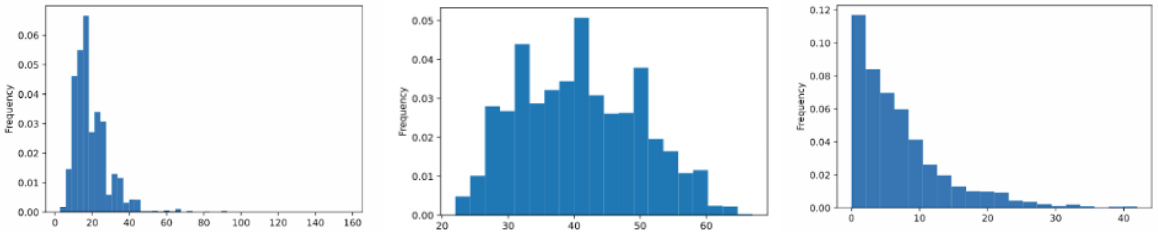

Figure 2.2.2-1 Histograms (left to right: “Annual Income”, “Age”, “Working Years”)

3) Categorical features

Also, we derive the analytic results of all the rest of categorical features.

Table 2.2.2-10 “Income Type”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	inctp	Commercial associate	2981	2888	93	0.283581	0.031198	0.283137	0.298077
1	inctp	State servant	1083	1054	29	0.103025	0.026777	0.103333	0.092949
2	inctp	Working	6448	6258	190	0.613394	0.029467	0.613529	0.608974

Table 2.2.2-11 “Occupation Type”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	occyp	Laborwk	4329	4190	139	0.411815	0.032109	0.410784	0.445513
1	occyp	hightecwk	2024	1961	63	0.192542	0.031126	0.192255	0.201923
2	occyp	officewk	4159	4049	110	0.395643	0.026449	0.396961	0.352564

Table 2.2.2-12 “House Type”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	houtp	Co-op apartment	73	71	2	0.006944	0.027397	0.006961	0.00641
1	houtp	House / apartment	9284	9010	274	0.883181	0.029513	0.883333	0.878205
2	houtp	Municipal apartment	354	342	12	0.033676	0.033898	0.033529	0.038462
3	houtp	Office apartment	79	76	3	0.007515	0.037975	0.007451	0.009615
4	houtp	Rented apartment	156	153	3	0.01484	0.019231	0.015	0.009615

Table 2.2.2-13 “Education”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	edutp	Higher education	3042	2948	94	0.289384	0.030901	0.28902	0.301282
1	edutp	Incomplete higher	388	371	17	0.03691	0.043814	0.036373	0.054487
2	edutp	Lower secondary	97	92	5	0.009228	0.051546	0.00902	0.016026
3	edutp	Secondary / secondary special	6985	6789	196	0.664479	0.02806	0.665588	0.628205

Table 2.2.2-14 “Marriage Condition”

	Variable	Value	All	Good	Bad	Share	Bad Rate	Distribution Good	Distribution Bad
0	famtp	Civil marriage	819	794	25	0.077911	0.030525	0.077843	0.080128
1	famtp	Married	7639	7429	210	0.726693	0.027491	0.728333	0.673077
2	famtp	Separated	615	608	7	0.058505	0.011382	0.059608	0.022436
3	famtp	Single / not married	1210	1152	58	0.115107	0.047934	0.112941	0.185897
4	famtp	Widow	229	217	12	0.021785	0.052402	0.021275	0.038462

2.2.3 Outlier detection

We applied 9 outlier detection methods in total to this credit card data set, as shown in the following part.

1) Normal score (Deviation with respect to the mean)
   Normal score (Van der Waerden) test examines the location of one sample or equality of locations for two or multiple samples regardless of the distributions of the numeric data based on Van der Waerden rank scores. The normal score test provides high efficiency compared to other nonparametric test methods based on ranks of data when the normality assumptions are nearly or in fact satisfied. It is also robust to the violation of normality assumptions.

2) Median Absolute Deviation (Deviation with respect to the median)
   Median Absolute Deviation is a robust measurement of the sample deviation of univariate numerical data. At the same time, it can also represent the overall parameters estimated by the mad of the sample.

3) Interquantile range score
   The interquartile range is a measure of where the "middle fifty" is in a data set. Where a range is a measure of where the beginning and end are in a set, an interquartile range is a measure of where the bulk of the values lie.

4) Depth-based Approach
   Depth-based outlier detection is based on certain specific kinds of geometric spreads of the data. It does not use underlying probability distribution. For this technique to work the anomalies are expected to lie in the edges of the space. The data objects are organized in convex full layers. The outermost layer is at depth = 1 and it would increase by 1 as it moves towards the center. all points that are on the outer layers are outliers.

5) Outlier detection using Mahalanobis Distance

This method is used when given a certain kind of statistical distribution. Then Mahalanobis distance is applied to compute the distance between the variable and the mean value of all points. This distance follows a Chi-squared distribution with d degrees of freedom (d as the data dimensionality). It could be used with multiple distributions and both parametric and non-parametric models.

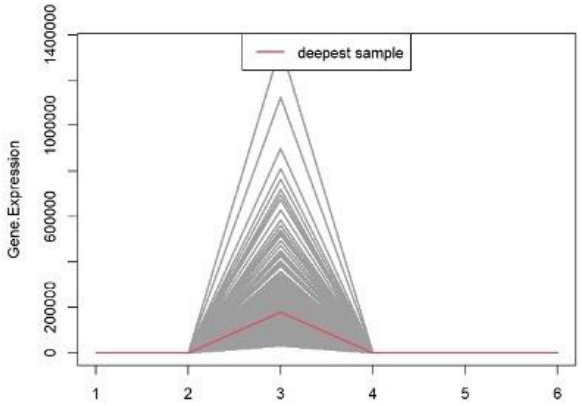

Figure 2.2.3-1 Result of Outlier Detection using Mahalanobis distance

6) Outlier detection using k Nearest Neighbors Distance method

The proposed kNN method detects outliers by exploiting the relationship among neighborhoods in data points. The farther a data point is beyond its neighbors, the more possible the data is an outlier.

7) Outlier detection using kth Nearest Neighbor Distance method

The kth nearest neighbor distance is used for setting the pruning threshold. Outliers are the n objects presenting the highest distance values to their respective kth nearest neighbor.

8) Outlier detection using Robust Kernel-based Outlier Factor (RKOF) algorithm

Local Outlier Factor (LOF) is based on the density estimate theory. But it is not accurate enough to detect outliers in the complex and large databases. The proposal of the variable kernel density can be estimated to address this disadvantage and improve the robustness to the variations of the parameter.

9) Outlier detection using generalized dispersion

It computes LOO dispersion matrix for each observation (dispersion matrix without considering the current observation) and based on the bootstrapped cutoff for score (difference between determinant of LOO dispersion matrix and det of actual dispersion matrix), labels an observation as outlier.

2.2.4 IV and WoE

We have introduced two concepts and applied them to our project: WoE (Weight of Evidence) and IV (Information Value).

Weight of Evidence (WoE):

$$woe_i = ln\frac{P_{yi}}{P_{ni}} = ln\frac{y_i/y_s}{n_i/n_s}$$

woe_i is the I category's WOE value. P_{yi} is the proportion of the positive samples in this category to all positive samples. P_{ni} is the ratio of negative samples (n_i) in this class to all negative samples (n_s).

Information Value (IV):

$$IV_i = (P_{yi} - P_{ni}) \times woe_i$$

The IV values of the various types are the difference between the conditional positive rate and the conditional negative rate multiplied by the WOE value of the variable. The total IV of the variable can be understood as the weighted sum of the conditional positive rate and the conditional negative difference:

$$IV = \sum_{i}^{n} IV_i$$

The IV value measures the variable's ability to predict. Here is the relationship between IV value and predictive power, from which we will get more information from our features.

Table 2.2.4-1 Relationship between IV value and predictive power

IV	Ability to predict
<0.02	Almost no predictive power
0.02~0.1	weak predictive power
0.1~0.3	Moderate predictive power
0.3~0.5	Strong predictive power
>0.5	Predictive power is too strong, need to check variables

Here is the IV result for each variable:

Table 2.2.4-2 IV results

variable	IV
Agegp	0.0659351
famtp	0.0431371
worktmgp	0.0402215
Reality	0.0274407
Gender	0.0252035
edutp	0.0103618
houtp	0.0073275
famsize	0.00615614
occyp	0.00482047
Incgp	0.002422
wkphone	0.00204243
ChldNo	0.00112145
Phone	0.00054805

inctp	5.1593e-05
email	1.73436e-05
car	4.54248e-06

From the result above, we can find that all the IVs are less than 0.1, which means they have weak predictive power. And some of them are even less than 0.02, we can think those variables almost have no predictive power. Therefore, we should do more data processing to our dataset to help them with better performance.

First, we try to split the dataset, for example, we have split the “Age” variable into “Age_lowest”, “Age_low”, “Age_high”, “Age_highest”, “Age_medium”. For “Family Size”, we split them into “1”, “2” and “3 more”. Such an operation is already implemented on every variable.

However, as we know our dataset is still imbalanced, we will use other ways to deal with this problem, which can be seen in the following part.

2.2.5 SMOTE for imbalanced data

An imbalanced classification problem is an example of a classification problem where the distribution of examples across the known classes is biased or skewed. The distribution can vary from a slight bias to a severe imbalance where there is one example in the minority class for hundreds, thousands, or millions of examples in the majority class or classes.

Imbalanced classifications pose a challenge for predictive modeling as most of the machine learning algorithms used for classification were designed around the assumption of an equal number of examples for each class. This results in models that have poor predictive performance, specifically for the minority class. This is a problem because typically, the minority class is more important and therefore the problem is more sensitive to classification errors for the minority class than the majority class.

There is an approach to address imbalanced datasets is to oversample the minority class. The simplest approach involves duplicating the examples in the minority class, although these examples do not add any new information to the model, this way can balance the class distribution. An improvement on duplicating examples from the minority class is to synthesize new examples from the minority class. This is a type of data augmentation for tabular data and can be very effective.

The most widely used approach to synthesizing new examples is called the Synthetic Minority Oversampling Technique, or SMOTE for short.

SMOTE works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line. Specifically, a random example from the minority class is first chosen. Then k of the nearest neighbors for that example are found (typically k=5). A randomly selected neighbor is chosen and a synthetic example is created at a randomly selected point between the two examples in feature space. The graph below could roughly show the idea of SMOTE.

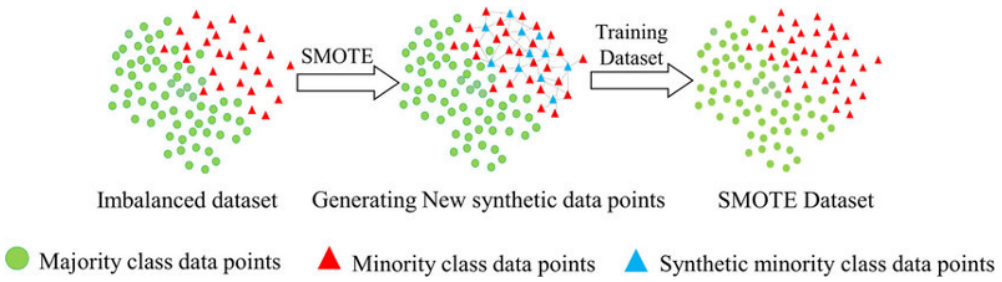

Figure 2.2.5-1 Rough demonstration of SMOTE

The approach is effective because new synthetic examples from the minority class are created that are plausible, that is, are relatively close in feature space to existing examples from the minority class. But a general downside of the approach is that synthetic examples are created without considering the majority class, possibly resulting in ambiguous examples if there is a strong overlap for the classes. After over sampling, the number between 1 and 0 is balanced, which could be seen from the confusion matrix.

3. -Modeling approach-

3.1 Feature reconstruction

From the IV and WoE results calculated above, it is apparent that we cannot directly use the given 18 features for model fitting and prediction because some of them have small effects on the final decision of credit card approval. Instead, we applied their minimum and maximum. Some features with multiple values like the size of family were reduced to two values and we added one feature about proportion - famtp_Single / not married. The new features we used are listed in the table below.

Table 3.1-1 Reconstructed features

	Feature		Feature		Feature
1	Gender	11	gp_worktm_highest	21	houtp_Rented apartment
2	Reality	12	gp_worktm_low	22	houtp_With parents
3	ChldNo_1	13	gp_worktm_medium	23	edutp_Higher education
4	ChildNo_2More	14	occyp_hightecwk	24	edutp_Incomplete higher
5	wkphone	15	occyp_officewk	25	edutp_Lower secondary
6	gp_Age_high	16	famsizegp_1	26	famtp_Civil marriage
7	gp_Age_highest	17	famsizegp_3more	27	famtp_Separated
8	gp_Age_low	18	houtp_Co-op apartment	28	famtp_Single / not married
9	gp_Age_lowest	19	houtp_Municipal apartment	29	famtp_Widow
10	gp_worktm_high	20	houtp_Office apartment		

In total, we have the reconstructed feature space with 29 new features. All these features will later be used in the modeling process for building fitted algorithms.

3.2 Models

3.2.1 Logistic regression

Logistic regression is a process of modeling the probability of a discrete outcome given an input variable. The most common logistic regression models a binary outcome; something that can take two values such as true/false, yes/no, and so on. Multinomial logistic regression can model scenarios where there are more than two possible discrete outcomes. Logistic regression is a useful analysis method for classification problems, where you are trying to determine if a new sample fits best into a category. As aspects of cyber security are classification problems, such as attack detection, logistic regression is a useful analytic technique.

The primary difference between linear regression and logistic regression is that logistic regression's range is bounded between 0 and 1. In addition, as opposed to linear regression, logistic regression does not require a linear relationship between inputs and output variables. This is due to applying a nonlinear log transformation to the odds ratio (will be defined shortly).

$$logistic\ function\ =\ \frac{1}{1+e^{-x}}$$

As opposed to linear regression where MSE is used as the loss function, logistic regression uses a loss function referred to as “maximum likelihood estimation (MLE)” which is a conditional probability.

3.2.2 Support vector machine (SVM)

The Support Vector Machine (SVM) method with soft margin is also suitable for predictions in this non-separable scenario. It aims at maximizing the margin while softly penalizing points that lie on the wrong side of the margin boundary. By using kernel functions in the input space, it will find out the proper decision boundary after solving the margin optimization problem with theorems of Lagrangian functions and KKT conditions.

Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane.

Support Vector Machine is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

3.2.3 Random forest

Since it is apparent that the original data set is not linearly separable, we implemented the Random Forest Classification method. It is an ensemble model that combines the predictions of a number of decision tree classifiers into a single prediction to combat the overfitting problem. This method is designed based on the idea of “bagging” (bootstrap aggregation) while enhancing the randomness of the member decision trees.

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model’s prediction.

3.2.4 Hyperparameter tuning

As our focus is on better model performance, we can tune their hyperparameters to find more fitted ones and thus improve the classification models' power. With certain possible values of hyperparameters, the Grid Search method can work to sample those values. Then combining the application of the 5-fold cross-validation method, accuracy would function as the evaluation criteria to judge the model.

3.3 Cross validation

Overfitting occurs when the model does so well on the training data that it even fits noise and outliers while it would perform poorly on out-of-training data. The existence of the overfitting problem demonstrates that this machine learning model has low bias but high variance. After fitting the models and tuning the hyperparameters, it is possible for us to confront the overfitting problem.

In our model, we used the k-fold cross validation and set the value of K as 5, considering that 5-fold is a commonly used method for validation.

3.4 Performance metrics

Accuracy of the model is the first and necessary metric to evaluate the model's performance.

Considering that the value of accuracy could be affected by the imbalance of the original dataset, we will also use the confusion matrix and a series of related metrics for more precise evaluation, although the SMOTE method has been applied to solve the data imbalance problem. The other metrics include: precision, sensitivity/recall, specificity, and F1 score.

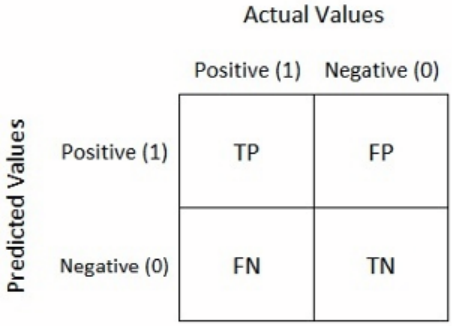

Figure 3.4-1. Confusion Matrix

The formulas of the metrics are shown below:

$$accuracy = \frac{TP+TN}{n}$$

$$precision = \frac{TP}{TP+FP}$$

$$specificity = \frac{TN}{TN+FP}$$

$$recall = \frac{TP}{TP+FN}$$

$$F1\ score = \frac{2}{\frac{1}{precision}+\frac{1}{recall}} = \frac{2*precision*recall}{precision+recall}$$

4. -Results-

Before truly training the models, the original dataset has to be split into the training set and the test set. Here, we will use the train_test_split function in Python and let the size of the test set be 30% of the whole dataset. Then comes the modeling part.

4.1 Hyperparameter tuning

To find the most fitted model, our first goal is to tune the hyperparameters. The input space includes the processed dataset with the 29 newly selected features. Considering that our problem of credit card is in fact a binary classification problem, the algorithms we used in the modeling process include Logistic regression, SVM and Random forest classification.

4.1.1 Logistic regression

For the Logistics regression algorithm, the hyperparameters we need to tune are the inverse of regularization strength - C - and the algorithm to use in the optimization problem - solver. Here we first tuned the hyperparameter C. After trying different values of C, there appeared no big difference for the accuracy of the linear model.

Table 4.1.1-1 Tuning “C” in the linear model

C	1	2	3	4	5
Accuracy	0.6485	0.64857	0.64857	0.64857	0.64871

The results do not differ much, so we set c=0.8. Then we tuned the hyperparameter “solver” in this algorithm. Commonly used solvers in the Logistic regression algorithm are ‘newton-cg’, ‘lbfgs’, ‘liblinear’, ‘sag’ and ‘saga’. The results are shown in the table below.

Table 4.1.1-2 Tuning “solver” in the linear model

Solver	newton-cg	lbfgs	liblinear	sag	saga
Accuracy	0.64844	0.64844	0.6485	0.64844	0.64844

As is shown in Table 4.1.1-2, there is no difference between the results. Thus, the solver we chose in this linear model is simply the default one - ‘lbfgs’. Using the hyperparameter pair we chose, the modeling results are demonstrated in the following graph of confusion matrix. The accuracy of this linear model is 0.64844.

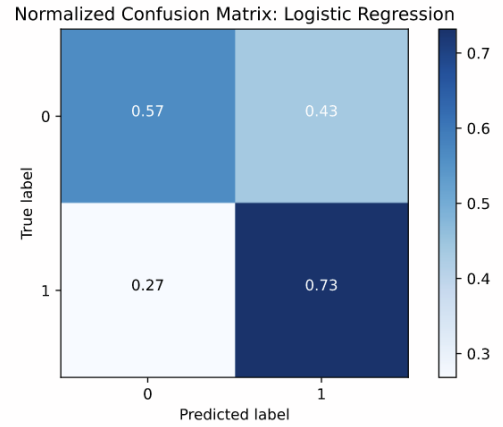

Figure 4.1.1-1 Normalized confusion matrix of Logistic regression

4.1.2 Random forest

There are three hyperparameters for us to tune in the Random forest algorithm - number of estimators, maximum depth and minimum sample leaf. Using the grid search, we have the following tables showing the accuracy results.

Table 4.1.2-1 Tuning 'n_estimators' in the Random forest algorithm

n_estimators	50	100	150	200	250	300
Accuracy	0.83889	0.83423	0.83713	0.83531	0.83713	0.83322

The best accuracy is reached when the number of estimators is 100, although the result may fluctuate a little in the Random forest algorithm.

Table 4.1.2-2 Tuning 'max_depth' in the Random forest algorithm

max_depth	1	5	10	15	20	25	30	40
Accuracy	0.62112	0.73422	0.81764	0.85285	0.86492	0.86842	0.86883	0.86809

We could find that when max depth increases, the accuracy will also increase. But after max depth reaches 20, the accuracy fluctuates near 0.87. So here we use max_depth=20.

Table 4.1.2-3 Tuning 'min_samples_leaf' in the Random forest algorithm

min_samples_leaf	1	5	10	15	16	20
Accuracy	0.89978	0.8894	0.87793	0.86822	0.87099	0.85844

There is no big difference between the results when we change the min samples leaf. Combining the hyperparameter tuning results above, we chose: n_estimators=100, max_depth=20, min_samples_leaf=16. Thus, the corresponding accuracy score is 0.86094. Also, the confusion matrix is drawn in Figure 4.1.2-1.

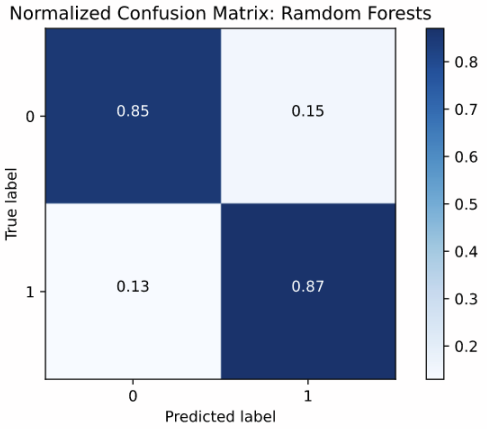

Figure 4.1.2-1 Normalized confusion matrix of Random forest classifier

4.1.3 SVM

There are two hyperparameters in total in the model of SVM that need us to tune - the inverse of regularization strength and the kernel. Tuning these two hyperparameters using the idea of grid search, we computed the following two tables.

Table 4.1.3-1 Tuning 'C' in SVM

C	0.05	0.1	1	5	10
Accuracy	0.64277	0.64311	0.64311	0.64311	0.64311

The accuracy does not change much when the value of C is bigger than 0.1. So here we just use C=0.8. Then we continued to tune the hyperparameter of the kernel.

Table 4.1.3-2 Tuning 'kernel' in SVM

kernel	linear	poly	rbf	sigmoid
Accuracy	0.64297	0.83275	0.8602	0.5054

As we can see, the model has better performance when the kernel is polynomial or rbf. So we chose the kernel as rbf. Using the tuned hyperparameter pair, the accuracy score is 0.8579. The confusion matrix is shown in the following figure.

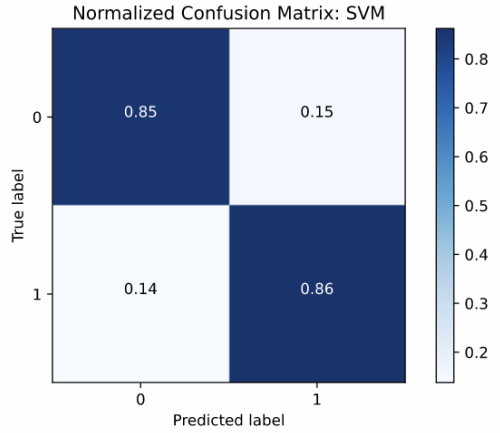

Figure 4.1.3-1 Normalized confusion matrix of SVM classifier

4.2 Cross validation results

Here we set the number of K-fold as 5, so 5-fold cross-validation was implemented to three classification models stated above. The cross validation function used here is the cross_val_score function from sklearn. For further comparison, we choose the f1-score as the scoring function in this validation process. The corresponding computation results are shown in the table below.

Table 4.2-1 Cross validation results

	Logsitic regression	**Random forest**	**SVM**
accuracy score	0.69	0.83	0.85
standard deviation	0.01	0.01	0.01

4.3 Performance comparison

Our initial intention was to let the logistic regression model be the basis model for comparison and compare performances of different algorithms to find the best one. The following table clearly shows the comparison of three classification models with all computational results we got before.

Table 4.3-1 Comparison of models

Model	**Logistic regression**	**Random forest**	**SVM**
Hyperparameters	C=0.8, solver=lbfgs	n_estimators=100, max_depth=20, min_samples_leaf=16	C=0.8, kernel=rbf
Accuracy	0.69	0.83	0.85

Observing the result table, we could find that the Random forest classifier and SVM classifier both have much better performances than the Logistic regression model, with accuracy as 83% and 85% respectively after implementing the 5-fold cross-validation. But the SVM model outperformed 2% more than the Random forest model, which means the SVM model has the best performance of all three models when dealing with the credit card application dataset.

5. -Conclusions and insights-

Considering the pressure of market competition, financial organizations have paid great attention to the field of credit card and credit system. Customers want to benefit from the approval of credit cards for future consumption, while the banks need to control the number of credit card approval for budgets and profit.

In this report, we briefly analyzed the credit card dataset given on Kaggle. Firstly, the vintage analysis was conducted to find out the standard of “bad” customers in the original dataset. We also drew the graph to compare different standards (built based on the past due dates) and eventually found that 60 days is the most suitable choice. This means if the credit card is past due for over 60 days, the customer will be classified as a “bad” customer. Then we conducted basic statistical analysis of three types of features - binary features, continuous features and categorical features. Two concepts, IV and WoE, were introduced to find out the importance of each original feature. In the modeling process, the method of SMOTE was applied to the whole dataset to fix the problem of data imbalance. Then using multiple feature construction methods, we created the new feature space with 29 features and later used them to fit models. During the model fitting process, we tuned the corresponding hyperparameters and then found the relatively most fitted models. The accuracy of each model with 5-fold cross-validation implemented is: 69% for Logistic regression, 83% for Random forest and 85% for SVM. It is apparent that the fitted SVM classification model has the best performance among all.

The final result of our project shows that in the field of credit card approval, the financial organizations could consider the Support vector machine classifier as a potentially useful tool. The method of modeling the credit card approval could not only help banks decide the applications from customers, but also help build the credit system for future analysis and usage. This can contribute to finding characteristics of each customer and maybe predict their future behavior related to personal credits. To conclude, this forecast of credit card approval based on historical data has great practical significance for financial organizations and customers.

6. -References-

[1] Y. X. G. W. Ruiting Mei, “Study on Analysis and Influence Factors of Credit Card Default Prediction Model,” Statistics and Application, p. Vol.05 No.03:13, 2016.9.20.

[2] X. Song, “Credit Card Approval Prediction,” 2020.3.24. Available: https://www.kaggle.com/rikdifos/credit-card-approval-prediction. [visiting date: 2022/5/7].