Report Project Credit_P02

Ana Paula Juárez, Valeria Araiza, Abril Galán, Ricardo Cárdenas

Introduction

Credit risk analysis is a critical tool for evaluating a borrower's ability to repay a loan in the future. In this report, a credit risk analysis is presented using a dataset containing information about borrowers, including their credit history, income, employment, and other factors.

Exploratory data analysis

The pandas library was used to load the data from a CSV file, and the "id" column was removed as it did not provide relevant information for the analysis. The variables of interest for credit risk analysis were identified, and a descriptive analysis of the data was conducted to understand the distribution and characteristics of the variables.

Regarding the numerical variables, it was observed that most of the variables were skewed to the right, indicating the presence of outliers or a non-normal distribution. On the other hand, categorical variables were converted to numerical variables using the ordinal encoding technique.

Variable transformation

Variable transformation techniques were used to improve the quality of the analysis and the accuracy of the model. The ColumnSelectorTransformer, BinningTransformer, and WOETransformer transformers were used to select relevant columns, group numerical variables into categories, and transform variables through the WoE odds transformation.

The ColumnSelectorTransformer and BinningTransformer transformers were used to select relevant columns and group numerical variables into categories, respectively. On the other hand, the WOETransformer transformer was used to transform variables through the WoE odds transformation. The goal of the WoE odds transformation is to provide a measure of the strength of association between each variable and the output variable.

Variable transformation analysis

The variable transformations were analyzed using the WOETransformer transformer. It was observed that some variables were highly associated with the output variable, while others were not. To visualize this, bar charts were used to show the distribution of the variables and their WoE. It was observed that variables such as interest rate, loan duration, borrower rating, and income level were highly associated with the output variable.

Predictive model

The GradientBoostingClassifier classification model was used to predict the output variable "status." The F1-score metric was used to evaluate the model's performance, and the ROC curve was used to evaluate the model's ability to correctly classify observations.

The model was trained using the transformed variables, and the cross-validation technique was used to optimize the model parameters. It was observed that the model had moderate performance with an F1 score of approximately 70%.

Conclusion

Credit risk analysis is a critical tool for evaluating a borrower's ability to repay a loan in the future. A dataset containing information about borrowers was used, and variable transformations were performed to improve the quality of the analysis.

Subsequently, statistical modeling techniques, such as logistic regression, were applied to develop a predictive model of credit risk. Different performance metrics, such as accuracy, sensitivity, and specificity, were evaluated to measure the effectiveness of the model in classifying borrowers as high or low risk.

It is important to note that credit risk analysis is an ongoing process, and models should be reviewed and updated periodically to ensure their accuracy and relevance. Additionally, it is essential that lenders and borrowers understand the meaning of different risk metrics and how they are used in loan decision-making.

In summary, credit risk analysis is a valuable tool for evaluating a borrower's ability to repay a loan and minimizing the risk of default. By applying statistical modeling techniques and performance metrics, lenders can make informed decisions and mitigate the risks associated with loan granting.

Detailed Report of Some Code:

First of all, we download the data given to us and get the correlation. We do not want to use any variables that have a lot of correlation neither the ones that have little correlation between them.

For the WOE we select the variables to keep and modify them so we can read them in a correct way. Then we train the model.

```
woe_t.fit(x_train_b, y_train)
```

Python

```
WOETransformer(columns=['loan_amnt', 'term', 'int_rate', 'grade', 'emp_length', 'home_ownership', 'annual_inc', 'verification_status', 'purpose', 'dti', 'delinq_2yrs',
'inq_last_6mths', 'open_acc', 'pub_rec', 'revol_util', 'total_acc', 'initial_list_status', 'out_prncp', 'total_pymnt', 'total_rec_int', 'status'], target_mappings={0: 'good', 1:
'bad'})
```

```
woe_t.transform(x_train_b).head()
```

Python

	loan_amnt	term	int_rate	grade	emp_length	home_ownership	annual_inc	verification_status	purpose	dti	delinq_2yrs	inq_last_6mths	open_acc	pub_rec	revol_util
0	0.039104	0.132276	0.245609	0.363488	0.103018	-0.161971	-0.192434	-0.173580	0.262242	-0.167252	0.000165	0.000447	-0.002391	-0.000012	-0.101005
1	0.039104	-0.294989	-0.392335	-0.055709	-0.103593	-0.161971	-0.192434	0.054494	0.226328	0.228539	0.000165	0.000447	-0.002391	-0.000012	0.177735
2	0.039104	0.132276	-0.392335	-0.055709	0.103018	-0.161971	-0.192434	0.167166	-0.807898	0.228539	0.000165	0.000447	-0.002391	-0.000012	-0.101005
3	0.039104	0.132276	0.245609	-0.055709	0.103018	-0.161971	-0.192434	0.054494	-0.264567	0.061188	0.000165	0.000447	-0.002391	-0.000012	0.177735
4	0.039104	-0.294989	0.245609	0.363488	-0.103593	-0.161971	0.129966	0.054494	-0.264567	0.061188	0.000165	0.000447	-0.002391	-0.000012	-0.101005

+ Code + Markdown

In this next part, we do the WOE mappings for the variables in this way.

```
display(woe_t.woe_mappings['loan_amnt'])
display(woe_t.woe_mappings['int_rate'])
```

	loan_amnt	good	bad	woe	info_val
3	(20_000, inf)	0.208178	0.221960	-0.064107	0.000884
2	(10_000, 20_000)	0.391321	0.393105	-0.004549	0.000008
1	(1000, 10000)	0.397364	0.382125	0.039104	0.000596
0	(-inf, 1000)	0.003137	0.002809	0.110509	0.000036

	int_rate	good	bad	woe	info_val
2	(20, inf)	0.075529	0.172245	-0.824408	0.079734
1	(15, 20)	0.264751	0.391947	-0.392335	0.049903
3	(9, 15)	0.484441	0.378943	0.245609	0.025911
0	(-inf, 9)	0.175279	0.056865	1.125699	0.133299

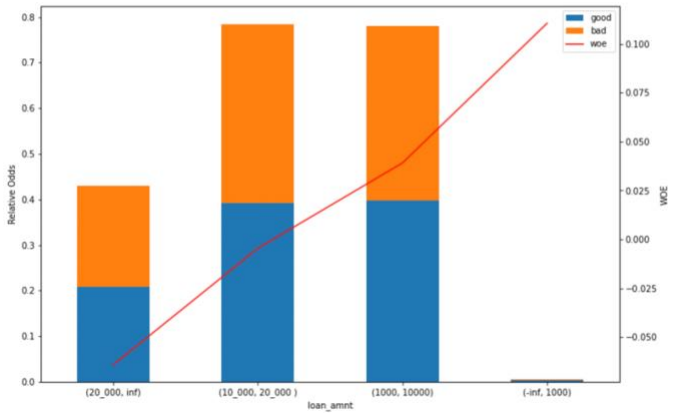
```
display(woe_t.woe_mappings['grade'])
display(woe_t.woe_mappings['emp_length'])
```

	grade	good	bad	woe	info_val
6	G	0.005766	0.018051	-1.141287	0.014021
5	F	0.024157	0.062581	-0.951888	0.036575
4	E	0.069300	0.136535	-0.678134	0.045594

Then we plot the variables so we can view them and see if the data given to us is relevant, for example in this next graph the last column we can see that really that data is not so relevant but we can use it because the value is not 0.

```
woe_t.woe_mappings["loan_amnt"].plot(
    x="loan_amnt", y="woe", ax=ax, secondary_y=True, mark_right=False, c="r"
)

ax.right_ax.set_ylabel("WOE")
ax.set_ylabel("Relative Odds");
```



In the pipelines first part, we use this part of the code to see really which variables we can use and which are not really relevant to use.

Pipelines

```
for i in cols_to_keep:
      print(i,":",woe_t.woe_mappings[i]["info_val"].sum())
```

```
loan_amnt : 0.0015238548895052046
term : 0.0388936868050649
int_rate : 0.28884705522202625
grade : 0.29038265423533693
emp_length : 0.005992216634496431
home_ownership : 0.021665476346556134
annual_inc : 0.04700314579685071
verification_status : 0.020873361093898757
purpose : 0.03691862165777514
dti : 0.02296389281541246
delinq_2yrs : 7.509273055445297e-05
inq_last_6mths : 0.0007456252076791674
open_acc : 0.00022913741959972793
pub_rec : 1.4513534174038643e-10
revol_util : 0.018216589614104516
total_acc : 0.004812165109308639
initial_list_status : 0.024766325013152096
out_prncp : 0.705433211931269
total_pymnt : 0.4855710319693385
total_rec_int : 0.015548546094269854
status : 0.0
```

This are the columns we chose in base of the numbers given on the top.

```
new_cols_to_keep = ['int_rate','grade','annual_inc','purpose','pub_rec','out_prncp','total_pymnt']
x_train_b = x_train_b.loc[:, new_cols_to_keep]
```

```
x_train_b
```

	int_rate	grade	annual_inc	purpose	pub_rec	out_prncp	total_pymnt
0	(9, 15)	B	(3.000e+03, 6.0000e+04)	credit_card	(-inf, 30)	(-inf, 1000)	(1000, 15_000)
1	(15, 20)	C	(3.000e+03, 6.0000e+04)	car	(-inf, 30)	(-inf, 1000)	(1000, 15_000)
2	(15, 20)	C	(3.000e+03, 6.0000e+04)	small_business	(-inf, 30)	(-inf, 1000)	(1000, 15_000)
3	(9, 15)	C	(3.000e+03, 6.0000e+04)	other	(-inf, 30)	(-inf, 1000)	(15_000, 35_000)
4	(9, 15)	B	(1.00000e+04, 1.0000e+05)	other	(-inf, 30)	(-inf, 1000)	(1000, 15_000)
...	...	...	...	...	...	...	...
465940	(9, 15)	C	(1.0000e+05, inf)	debt_consolidation	(-inf, 30)	(10_000, 30_000)	(15_000, 35_000)
465941	(15, 20)	D	(1.00000e+04, 1.0000e+05)	debt_consolidation	(-inf, 30)	(-inf, 1000)	(1000, 15_000)
465942	(15, 20)	D	(3.000e+03, 6.0000e+04)	debt_consolidation	(-inf, 30)	(10_000, 30_000)	(15_000, 35_000)
465943	(-inf, 9)	A	(1.00000e+04, 1.0000e+05)	credit_card	(-inf, 30)	(-inf, 1000)	(1000, 15_000)
465944	(15, 20)	D	(3.000e+03, 6.0000e+04)	other	(-inf, 30)	(1000, 10_000)	(1000, 15_000)

465945 rows × 7 columns