German Credit Risk Classification

Data Characteristics and Quality issues (milestone 1)

Firstly, we defined the characteristics of the data by doing EDA on the German credit risk dataset. We found that the data is kept in the same categorical format(AXXX).Despite that the original dataset came from UCI (the trusted source), there are no updates of records on the dataset.Although all relevant features are recorded,including both categorical and numerical (multivariate) data make it difficult to select features to train the model and the class feature is imbalanced.

Regarding the data quality issues, we identified four quality issues as follows:

1. Bias
   Imbalanced class could caused the prediction bias
2. Performance
   Overfitting, high accuracy, but low precision and recall
3. Fairness
   Prediction rely on specific people group (specific features)
4. Reliability
   Data's bias and unfairness caused the prediction unreliable

Defined goals and a suitable measure for the quality issues (milestone 1)

We defined two goals, first is to improve the prediction performance and the second is to reduce the prediction bias. To improve the prediction performance, we considered accuracy, precision, recall, f1-score, and confusion matrix. According to UCI Machine Learning Repository: Statlog (German Credit Data) Data Set, they said that "It is worse to class a customer as good when they are bad (5), than it is to class a customer as bad when they are good (1)." So, we are giving more concern on precision. Moreover, since the dataset is imbalanced, we decided to apply a data sampling method to handle this problem.

To reduce the prediction bias, we selected two measurements, i.e., Average odds difference (AOD) and Equal opportunity difference. The Average odd difference (AOD) is an average of difference in false positive rates and true positive rates between unprivileged and privileged groups. This metric must be close to zero (0) to ensure classification fairness. Equal opportunity difference (EOD) is a difference in true positive rates between unprivileged and privileged groups. Value of zero (0) implies the classification fairness.

Model exploration (milestone 1)

In this section, we experimented on various types of different ML models as follows:

- **Logistic Regression** - This is one of the supervised learning machine by Statistical Regression
- **GaussianNaiveBayes** - This is a probabilistic classification algorithm based on applying Bayes' theorem
- **Support Vector Machine** - This is supervised learning models with associated learning algorithms that analyze data for classification and regression analysis.

- **Random Forest** - This is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees
- **Extreme Gradient Boosting** - Implementation of the stochastic gradient boosting ensemble algorithm for classification and regression problems in machine learning competitions.
- **Ridge Regression** - Ridge regression is a method of estimating the coefficients of multiple-regression models in scenarios where the independent variables are highly correlated.

Model comparison (milestone 2)

Table 1 : model comparison results

Classifiers	Accuracy (%)	Training time(s)	Training Memory used (MB)	Testing time(s)	Testing Memory used (MB)	Model size (MB) (.pkl format)	Developer
XGBoosting	76.88	0.13	0.40	**0.01**	~ 0	0.20	Frong
LinearSVM	77.39	**0.03**	0.70	**0.01**	~ 0	**0.01**	Punch
Random Forest	76.88	0.2	1.68	0.02	~ 0	2.67	
GaussianNaive Bayes	73.4	0.043	**0.031**	0.033	0.211	**0.01**	Wendy
Linear Logistic Regression	**77.9**	0.050	0.242	0.050	0.020	**0.01**	
SMOTEENN with StandardScaler and RidgeClassifier	74.1	0.33	1.72	0.02	~ 0	2.89	Jincheng Zhang

Experiment dataset : German dataset, contains a total of 995 instances and 20 features.

Metrics :

1. **Prediction accuracy :** Logistic regression classifier achieved the highest accuracy as 77.9%, followed by LinearSVM (77.39%). XGBoosting and Random forest achieved the same accuracy as 76.88 % . The worst is the Gaussian Naive Bayes classifier which achieved 73.4% of accuracy.

2. **Training cost (Training time & memory usage) :** Linear SVM achieved the lowest training time as 0.031 seconds, followed by GaussianNaiveBayes (0.043 seconds) and Logistic Regression (0.050 seconds). The worst is the SMOTEENN classification which achieved 0.33 second. In terms of memory usage, The best one is GaussianNaiveBayes which achieved 0.031MB likewise the worst one is SMOTEENN at 1.72MB.
3. **Inference cost (Testing training time & memory usage) :** LinearSVM,GaussianNaiveBayes and logistic regression consume a small amount of memory on testset ,accounting for 0.01MB

respectively in comparison to other classifier models. In terms of training time, XGBoosting and LinearSVM have the shortest training time at 0.01second each. However, the most memory usage classifier out of six on the testset is Gaussian Naive Bayes and the classifier which takes the longest time to execute the testset is Logistic Regression.

4. **Model size :** LinearSVM,GaussianNaiveBayes and logistic regression have a minimum with model size around 0.01MB as .pkl format file. The maximum model size is the ridge classifier as 2.89MB.

According to the result the best model is Linear Logistic Regression. To select the best model, we first prioritize the model accuracy. The Logistic Regression model has achieved a considerable accuracy of 77.9% as compared to other classifiers' accuracy score within our analysis. Moreover, its' small-sized can achieve the highest accuracy score which is a reasonable choice.

Metric measurement (milestone 2)

Performance measurement

Table 1 : model performance comparison after applied sampling techniques

Model	Sampling method	Accuracy (%)	Precision (%)	**Recall (%)**	**F1-score (%)**
XGBoosting	Under-sampling	67.84	71.74	67.84	68.99
	Over-sampling	73.37	73.77	73.37	73.55
	Non-sampling	**76.88**	**75.97**	**76.88**	**76.21**
LinearSVM	Under-sampling	72.36	75.47	72.36	73.27
	Over-sampling	72.86	74.29	75.38	73.93
	Non-sampling	**77.39**	**76.72**	**77.39**	**76.95**
Random Forest	Under-sampling	69.35	74.25	69.35	70.57
	Over-sampling	75.38	74.29	75.38	74.58
	Non-sampling	**76.88**	**75.67**	**76.88**	**74.85**
KNNs	Under-sampling	72.4	72.8	72.4	72.6
	Over-sampling	**76.4**	**80.6**	**76.4**	**77.3**
	Non-sampling	75.9	74.4	75.9	73.5
SMOTEENN with StandardScaler and RidgeClassifier	**Under-sampling**	**74.16**	**75.12**	**77.21**	**76.95**
	Over-sampling	75.27	73.98	75.44	73.94
	Non-sampling	75.98	73.27	74.29	75.66

In terms of models' performance, the sampling techniques are mostly not effective on the models, except for two models such as KNN and RidgeClassifier.So we can conclude that sampling techniques might not be the suitable choice to improve the performance.

Fairness measurement

For the fairness measurement, we have measured by using two fairness metrics, i.e., Average odds difference (AOD) and Equal opportunity difference. Moreover, we have applied two techniques called reweighing and reject option classification, in order to mitigate the bias.

Table 2 : Classification fairness measurement and comparison

Models		**Reweighting**		**Reject option classification**	
		Average odds difference	Equal opportunity difference	Average odds difference	Equal opportunity difference
Logistic Regression	Before	-0.2508	-0.2974	-0.2828	-0.2874
	After	-0.2115	-0.2088	-0.2225	-0.2088
Gaussian Naive Bayes	Before	-0.2451	-0.2759	-0.2746	-0.2759
	After	-0.2030	-0.2203	-0.1815	-0.1743
SVM	Before	-0.3458	-0.4058	-0.3458	-0.4058
	After	-0.1089	0.0000	0.0595	0.0487
Random Forest	Before	-0.2059	-0.2029	-0.2059	-0.2029
	After	-0.0218	-0.0341	-0.0947	-0.0860
XGBoost	Before	-0.1661	-0.2289	-0.2278	-0.2318
	After	-0.0078	0.0877	0.0937	0.1015
Ridge Regression	Before	-0.2351	-0.2514	-0.2435	-0.2967
	After	-0.1954	-0.2167	-0.1687	-0.2165

The table 2 represents the fairness measurement and comparison among classification models. The result indicates that the biases were reduced after we applied reweighing and reject option classification methods. After applying bias mitigation techniques, the average odd difference (AOD) values are closely to zero. Similarly to equal opportunity difference (EOD), after applying bias mitigation techniques, the values are also close to zero. The zero or nearly zero of the fairness metrics imply that the classification results are fairly classified.

Table 3 : Classification performance comparison before and after applied bias mitigation techniques

Models		Reweighting				Reject option classification			
		Accuracy	Precision	Recall	F1-score	Accuracy	Precision	Recall	F1-score
Logistic Regression	Before	0.6042	0.7846	0.5000	0.6108	0.6975	0.9077	0.5413	0.6782
	After	0.6085	0.7826	0.5294	0.6316	0.6685	0.8788	0.5321	0.6629
Gaussian Naive Bayes	Before	0.6048	0.7903	0.4804	0.5976	0.7021	0.9091	0.5505	0.6857
	After	0.6097	0.7937	0.4902	0.6061	0.6425	0.8594	0.5046	0.6358
SVM	Before	0.6066	0.8113	0.4216	0.5548	0.6066	0.8113	0.4216	0.5548
	After	0.6232	0.8511	0.3922	0.5369	0.5980	0.7714	0.5294	0.6279
Random Forest	Before	0.6158	0.8103	0.4608	0.5875	0.6158	0.8103	0.4608	0.5875
	After	0.6097	0.7937	0.4902	0.6061	0.6011	0.8000	0.4314	0.5605
XGBoost	Before	0.5913	0.7925	0.4118	0.5419	0.6761	0.8906	0.5229	0.6590
	After	0.5619	0.7660	0.3529	0.4832	0.6190	0.7882	0.6381	0.7053
Ridge Regression	Before	0.6891	0.7962	0.4839	0.6034	0.6423	0.8761	0.4937	0.6285
	After	0.6132	0.7973	0.4768	0.6184	0.6295	0.8538	0.4752	0.6121

Table 3 above represents the classification performance of individual models, before and after applying bias mitigation techniques. As a result, most of the models achieved better performance in terms of F1-score, except for the XGBoost models whose F1-score significantly decreased after applying reweighting techniques. When considering precision, most of the models achieved better scores after the bias reduction processes. However, when considering recall, the results indicate around 3 - 5% drop in some models such as SVM, random forest, and XGBoost.

Results analysis and possibility improvement

In terms of performance, we consider accuracy, precision, recall and F1-score by testing on different models. As a result, there are no significant differences between models' accuracy. After the bias reduction process, the average models' accuracy is around 60%. Not to note that linear regression trained with reject option classification and support vector machine combined with reweighting (but obtained lowest F1-score and recall) are outnumbered by other classifiers. Similarly to precision, recall and F1-score, there are no huge differences between the models.

In the view of fairness, we consider two metrics, i.e., average odd difference (AOD) and equal opportunity difference (EOD). Moreover, we are using two bias mitigation methods as described in the first section. According to our experimental results, the bias is potentially reduced for testing various classifiers. The most significant change is in the SVM model. The values in both farieness metrics (AOD and EOD) of two methods are close to zero which indicate the fairness.

In order to improve the quality of classifiers, we suggest that the bias reduction process is the most essential requirement if the data is biased. According to our experiment, without reducing bias, the quality of classifiers are not much improved even after applying other methods such as data sampling. In contrast, there are performance improvements in classifiers trained with bias-reduced dataset. Moreover, there is also the need in the feature collection and selection process to make the classification more accurate. Furthermore, we suggest experimenting with various types of machine learning techniques to find the appropriate classifiers for the credit prediction task.

Solution or best practices to mitigate the quality issues throughout the system's life cycle

For the solution or best practices, we are giving suggestions in two perspectives, i.e., from the theory and from our experience. These perspectives are described below:

Theory perspective :

- Make the Intelligent system to be generalize to the situations that hasn't faced before and measure how well it adapts(Generalization)
- Make sure the intelligent to produce the right type of mistakes which do not harm to others (In our case the accepted mistake is that the model predicts a customer as bad when they are good (false negative))
- Make sure to check the mistakes that the AI system made not to focus on specific users. It means that it must balance the types of mistakes it makes. (In our case, the system classifies as bad for a person who is female)
- It is also important to have the right amount of data which depends on how certain you need to be for evaluating processes
- Setting up the right operating points (precision and recall points) is also an important practice to check which models performs well

Experience perspective :

- Always monitor and check for bias and anomalies of data within the system's life cycle and the model must be retrained regularly.
- In the EDA process, there is a need for collaboration between customer and data scientist, in order to understand the data and gather hidden insight.
- Attention is needed on data's characteristics that must be protected and data privacy.
- Avoid black-box models.
- Having clear system's goal(s) and measurable metrics.

References

- https://github.com/Trusted-AI/AIF360/tree/master/examples
- https://www.mathworks.com/help/risk/explore-fairness-metrics-for-credit-scoring-model.html
- https://medium.com/sfu-cspmp/model-transparency-fairness-552a747b444
- https://aif360.readthedocs.io/en/latest/?badge=latest
- https://medium.com/analytics-vidhya/machine-learning-is-requirements-engineering-8957aee55ef4

Presentation slide

https://docs.google.com/presentation/d/1GtR7VuwWv4QZi6aa3xVgpplTcVet8DE7FyEPgofXgLw/edit?usp=sharing