Name: Sagar Das

Email address: sagardas7.1998@gmail.com

Contact number: 9007789474

Anydesk address: 169 667 754

Years of Work Experience: **1**

Date: **15th May 2022**

Self Case Study -1: Credit Card Approval Prediction

Overview

1. A credit score is a number between 300–850 that depicts a consumer's creditworthiness. The higher the score, the better a borrower looks to potential lenders. A credit score is based on credit history: number of open accounts, total levels of debt, and repayment history, and other factors.
2. Banks receive a lot of applications for issuance of credit cards. Many of them are rejected for many reasons, like high-loan balances, low-income levels, or too many inquiries on an individual's credit report which determines the credit score.
3. Manually analyzing these applications is error-prone and a time-consuming process. Luckily, this task can be automated with the power of machine learning and pretty much every bank does so nowadays. In this project, we will build an automatic credit card approval predictor using machine learning techniques, just like the real banks do.

4. Manually analyzing these applications is error-prone and a time-consuming process. Luckily, this task can be automated with the power of machine learning and pretty much every bank does so nowadays. In this project, we will build an automatic credit card approval predictor using machine learning techniques, just like the real banks do.
5. The dataset consists of:
    a. *application_record.csv*

        This table contains details about the background of the customer like annual income, type of employment, number of days employed, occupation type, number of childrens, number of family members, education qualification, owns a car or not, owns a real estate or not, type of housing.
    b. *credit_record.csv*

        This table contains the data related to a customers(id) past credit records which contains the month of the extracted data and the status of the due. Status explains about the number of days that his debt is due.
        i. id - client number
        ii. month_balance - The month of the extracted data is the starting point, backwards, 0 is the current month, -1 is the previous month, and so on
        iii. status - 0: 1-29 days past due 1: 30-59 days past due 2: 60-89 days overdue 3: 90-119 days overdue 4: 120-149 days overdue 5: Overdue or bad debts, write-offs for more than 150 days C: paid off that month X: No loan for the month

6. Drawbacks:
    a. So, we will have to build our training data from these tables and predict if the applicant is a "good" or "bad" applicant. But the definition of the "good" or "bad" is not given. We ourselves will have to determine if a customer is "good" or "bad" and then label them and accordingly train them.
    b. The dataset is an imbalanced type of dataset, so we need to keep in mind the same while preparing our data and training them.

7. The task of this problem set is to determine whether or not a customer is a good or a bad. Here a customer is one who is applying for a credit card. Based on the details of the customer and his past details, we will have to determine if the credit card should be approved or not. This is basically a **binary classification task**.
8. The metrics for the task that can be used:
    a. Precision - False Positive Rate should be as low as possible as we do not want to provide credit cards to bad customers. So Precision is important.
    b. Recall - Having high False Negative Rate is bad as not authorizing credit cards to good customers will have a bad impact on business revenue.
    c. **F1 Score** - As both Precision and Recall both are important, we can use F1 Score as our metric.
    d. AUC-ROC - This being a binary classification problem, we can use AUC - ROC score as a performance metric also.

Research-Papers/Solutions/Architectures/Kernels

1. **https://www.kaggle.com/code/samuelcortinhas/credit-cards-data-cleaning**

   Observations:

   - Only one of the feature has a missing value : occupation_type
   - There are 6 binary categorical variables i.e. each of the features are flag variables. Ex - whether a customers own a car or not
   - For creating the target variable, he says a user is of "high risk" if and only if they are late on payments more than 30 days otherwise "low risk"

   Takeaways -

   - For the binary categorical feature they use 0 to implement negative and 1 to implement positive in all six cases
   - From the Days from birthday, determine the age which is a better feature and easy to interpret

2. **https://www.kaggle.com/code/caesarmario/credit-card-approval-prediction-w-pycaret**

   Observations -

   - He had fixed imbalance data problem using fixImabalance = True while setting up

Takeaways -

- Used PyCaret for the coding.
  Results of comparing the models:

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
lr	Logistic Regression	0.9996	1.0000	0.9996	1.0000	0.9998	0.9578	0.9597	0.9590
lightgbm	Light Gradient Boosting Machine	0.9993	0.9998	0.9996	0.9997	0.9996	0.9191	0.9222	0.6250
ada	Ada Boost Classifier	0.9992	0.9988	0.9995	0.9997	0.9996	0.9092	0.9136	1.2510
gbc	Gradient Boosting Classifier	0.9990	0.9997	0.9992	0.9998	0.9995	0.9014	0.9049	5.4810
xgboost	Extreme Gradient Boosting	0.9990	0.9999	0.9994	0.9996	0.9995	0.8918	0.8957	14.6230
catboost	CatBoost Classifier	0.9989	0.9998	0.9992	0.9997	0.9994	0.8870	0.8920	17.2320
dt	Decision Tree Classifier	0.9973	0.8728	0.9984	0.9988	0.9986	0.7150	0.7227	0.3950
rf	Random Forest Classifier	0.9965	0.9969	0.9999	0.9965	0.9982	0.3540	0.4198	1.7460
et	Extra Trees Classifier	0.9964	0.9814	0.9998	0.9966	0.9982	0.3660	0.4380	1.5620
knn	K Neighbors Classifier	0.9914	0.7599	0.9937	0.9976	0.9957	0.3338	0.3508	0.4570
ridge	Ridge Classifier	0.9357	0.0000	0.9365	0.9988	0.9666	0.0879	0.1841	0.0570
lda	Linear Discriminant Analysis	0.9355	0.9402	0.9364	0.9988	0.9665	0.0878	0.1839	0.2460
nb	Naive Bayes	0.8017	0.9567	0.8013	0.9995	0.8894	0.0315	0.1184	0.0540
svm	SVM - Linear Kernel	0.5541	0.0000	0.5544	0.5977	0.5693	0.0012	0.0077	1.0680
qda	Quadratic Discriminant Analysis	0.1945	0.5297	0.1914	0.9970	0.3207	0.0006	0.0107	0.1300
dummy	Dummy Classifier	0.0046	0.5000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0410

- Even without using the occupation as a feature, he has produced a good score in the performance metrics
- Most performance metrics had the best score for Logistic Regression

3. **https://www.kaggle.com/code/umerkk12/credit-card-predictive-analysis**

   Observations -

   - He found some duplicates and dropped them
   - He deleted the "occupation" feature because it had a lot of null values
   - Used label encoder for categorical features
   - Used Normalization for Numerical Features

- Handled imbalance data using SMOTE
- He used multiple classifiers like - KNN, Logistic Regression, SVC, DT, Random Forest and XGBoost

Takeaways -

- Best results that he got was using XGBoost with 91% accuracy
- He did not perform any hyper parameter tuning, doing that might result in having much better performance

4. **https://www.kaggle.com/code/rikdifos/credit-card-approval-prediction-using-ml#Algorithms**

   Observations -

   - For creating the target variable, he says a user is of bad if and only if they are late on payments more than 30 days otherwise good
   - Used a concept of Weight of Evidence(WoE) that measures the variable's ability to predict

   Takeaways -

   - Among many algorithms that has been used, XGBoost performed the best with 93% accuracy and performed much better than Logistic Regression and SVM

5. https://www.kaggle.com/code/tanulkumarsrivastava/99-4-accuracy-credit-card-approval-model

   Observations -

   - Used a very good and sensible technique to determine the target variable
   - Used Label encoding for the categorical features
   - Converted number of the days into readable format
   - Creating a new relationship using education and occupation in order to fill up the missing occupation feature values

   Takeaways -

   - Feature Engineering on point specially, the building of the target variable
   - While other dropped the Occupation feature, he filled the occupation feature brilliantly using the relationship that he built
   - Got the best accuracy of 99.4% using Decision Tree

6. **Understanding CATBoosting**
   - From the research I have found that a lot of people used CATBoosting for training the data. It was a new concept for me. I went through some tutorials and got a good idea about the same.
   - “CatBoost” name comes from two words “Category” and “Boosting”

- Advantages:

    It doesn't need much hyperparameter tuning, can handle categorical data without converting categories into numbers and provides best in class accuracy.

- References:
    - https://catboost.ai/
    - https://www.youtube.com/watch?v=s8Q_orF4tcI&t=1s
    - https://www.analyticsvidhya.com/blog/2017/08/catboost-automated-categorical-data/
    - https://en.wikipedia.org/wiki/Catboost

First Cut Approach

Based on the research and readings that I have done. I will follow the below steps -

1. The most important part of this problem statement is the labeling of the data. Using proper labels improves the accuracy of the models(as we see in case of 5th research point who used a much better and different kind of labeling from others)
2. Feature Engineering plays a very important role. People who did not use proper feature engineering and preprocessing failed to get accuracy more than 90%
3. For encoding the categorical variables, most of them used label encoding. Although it gives us fair results, I can try using one hot encoding, as the categorical features do not have a lot of categories. I

want to try one hot encoding because Label encoding gives a sense of ordinality which is not logical in these cases.

4. I will perform hyper-parameter tuning(Random Search) for the best results.
5. As mentioned earlier, we can use the F1-Score as well as AUC-ROC score for this task. We can use both to have a good comparison.
6. Followed by that, we will get the feature importance for each of the features in order to conclude which are the top features that determines whether a customer is good or bad.