Credit Card Default Probability Prediction

Team 8
Ruoyi Chen, Hao Lin, Yifeng Wang, Yusen Wu

01

Problem Recognition

Credit card default rate is essential to banks and other financial Institutions.

- The use of credit has been one of the core activities in today‘s commercial setting, but the risk of credit default emerges incidentally.
- Machine learning model(s) could be a promising tool to identify people with high default risk to minimize potential bad-debt losses.

Who Cares about the Problem?

- Aid **financial institutions** in processing large amounts of applications
- Output model could be used for self-checks, saving **applicants** time and effort and help build a legitimate expectation

Our Goals:

- To predict whether or not the applicants will default as accurate as possible.
- To identify the influential factors of credit default.

02

Dataset Description & Feature Engineering

Dataset #1 describes the general information of credit card applicants

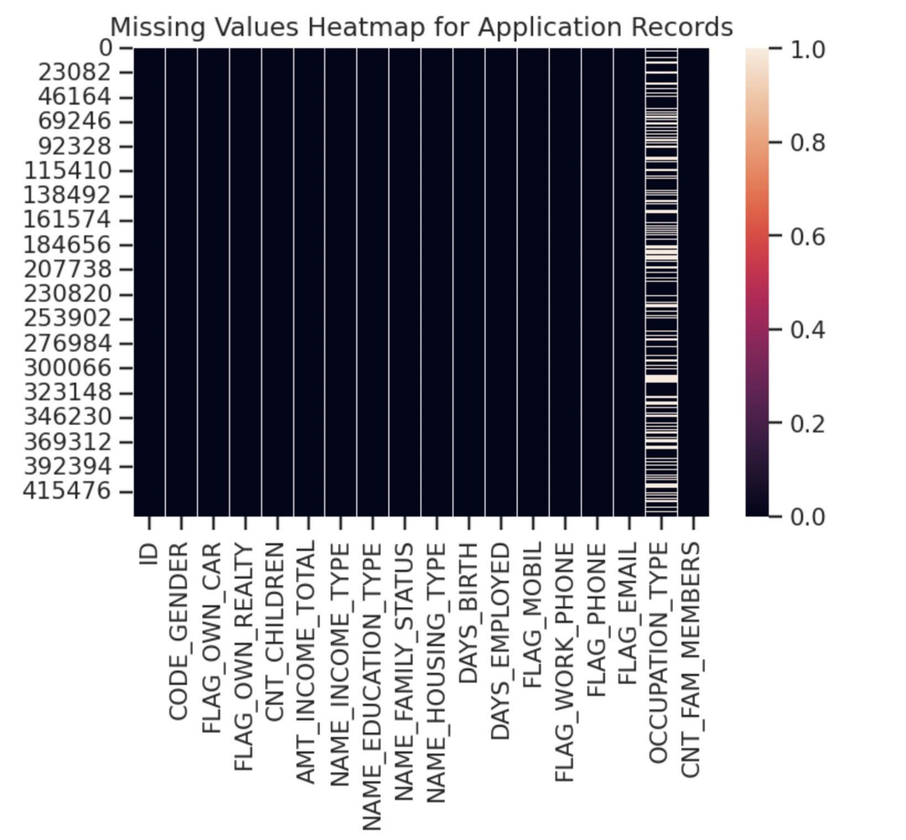
	ID	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	NAME_INCOME_TYPE	NAME_EDUCATION_TYPE	NAME_FAMILY_STATUS	NAME_HOUSING_TYPE	DAYS_BIRTH	DAYS_EMPLOYED	FLAG_MOBIL	FLAG_WORK_PHONE	FLAG_PHONE	FLAG_EMAIL	OCCUPATION_TYPE	CNT_FAM_MEMBERS
144137	5685745	F	N	N	0	157500.0	Working	Higher education	Married	House / apartment	-15281	-340	1	0	0	0	Accountants	2.0
330548	6339905	F	N	N	0	112500.0	Pensioner	Higher education	Married	House / apartment	-20807	365243	1	0	1	0	NaN	2.0
91412	5580075	F	Y	Y	0	270000.0	Working	Higher education	Single / not married	House / apartment	-18299	-153	1	0	0	0	Accountants	1.0

- 438,557 records*18 features
- 13 continuous variables
- 5 categorical variables

- Demographic information
- Property information
- Education information
- Family information

- **Missing value**
- No duplicate record
- No outlier

Dataset #1 - Cleaning & Feature Engineering



- About ⅓ of the data has missing value in occupation type
  New level 'Unknown' in occupation type

- Unemployed has the special value 365243 in their days_employed
  Replace special value with 0

  Add dummy variable 'Employed' to indicate employed or not

- FLAG_MOBIL only has 1 as its value
  Delete the whole column

Dataset #2 includes the credit records of the applicants

	ID	MONTHS_BALANCE	STATUS
505374	5061203	-42	0
719470	5096790	-39	C
540602	5065452	-28	X
210696	5017982	-21	X
1001115	5143489	-10	C

- 1,048,575 rows X 3 columns
- **Multiple monthly credit records referring to the same applicant in different record months**

- No missing value
- No duplicate record
- No outlier

Dataset #2 - Pivoting & Feature Engineering & Target Variable

	ID	first_record_time	record_counts	last_record_time	X_count	zero_count	C_count	one_count	two_count	three_count	four_count	five_count
0	5001711	-3	4	0	1	3	0	0	0	0	0	0
1	5001712	-18	19	0	0	10	9	0	0	0	0	0
2	5001713	-21	22	0	22	0	0	0	0	0	0	0
3	5001714	-14	15	0	15	0	0	0	0	0	0	0
4	5001715	-59	60	[illegible]	60	0	0	0	0	0	0	0

Pivoting

	ID	MONTHS_BALANCE	STATUS
505374	5061203	-42	0
719470	5096790	-39	C
540602	5065452	-28	X
210696	5017982	-21	X

Each applicant with only one record, with newly engineered feature:

- **First record time**
- **Last record time**
- **Credit record counts**
- **Default or not (Target Variable)**

People with record of past due over two month (about **1.6%**) will be classified as default to match US delinquency rate in Q3 2022 (about **1.86%**)

We merged the cleaned application record and pivoted credit record based on applicant ID to get our final dataset

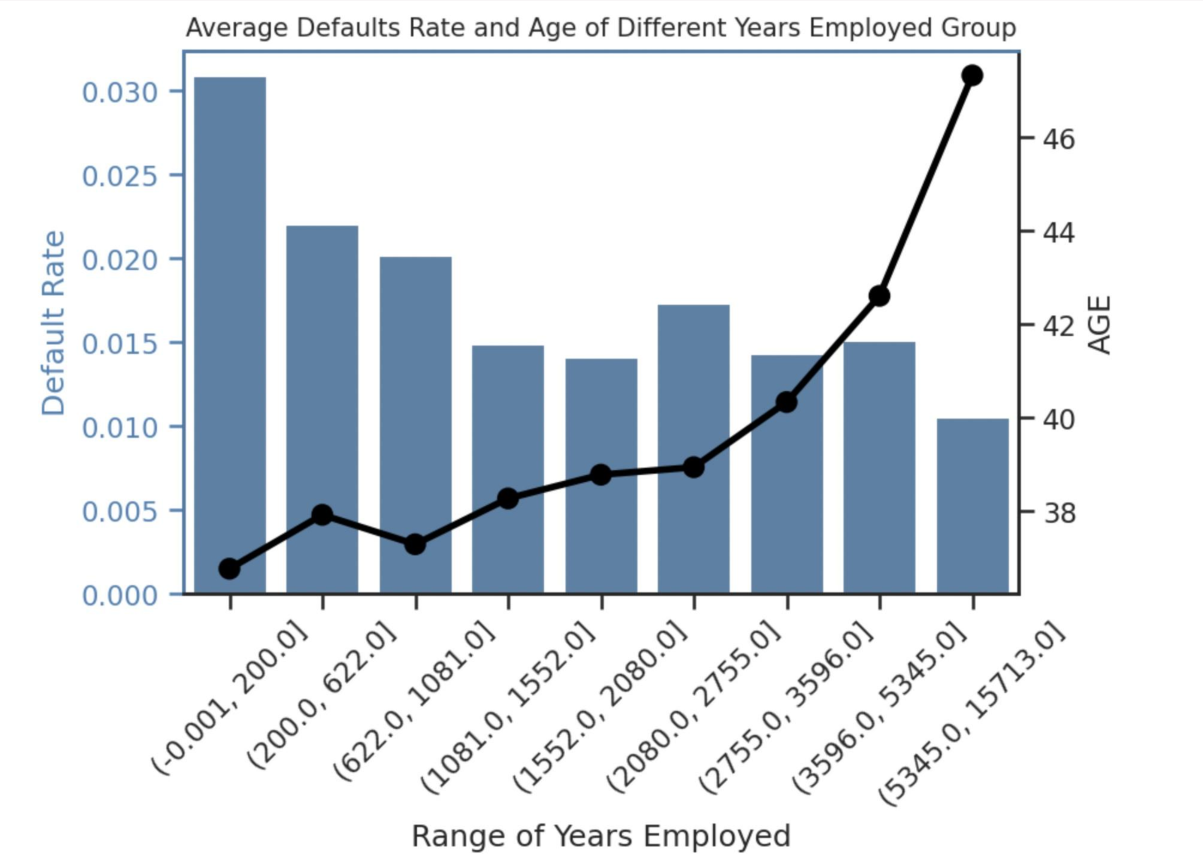
36457 records*20 features

Variable	Type	Description
CODE_GENDER	Categorical	Gender
NAME_INCOME_TYPE	Categorical	Income Category
NAME_EDUCATION_TYPE	Categorical	Education Level
NAME_FAMILY_STATUS	Categorical	Marital Status
OCCUPATION_TYPE	Categorical	Occupation
NAME_HOUSING_TYPE	Categorical	Way of Living
FLAG_WORK_PHONE	Categorical	Is there a work phone
FLAG_PHONE	Categorical	Is there a phone
FLAG_EMAIL	Categorical	Is there an email
FLAG_OWN_CAR	Categorical	Is there a car
FLAG_OWN_REALTY	Categorical	Is there a property
Employed	Categorical	Employed or not
Age	Numerical	Biological age
AMT_INCOME_TOTAL	Numerical	Annual Income
CNT_FAM_MEMBERS	Numerical	Family Size
CNT_CHILDREN	Numerical	# children
DAYS_EMPLOYED	Numerical	Days being employed
first_record_time	Numerical	Timestamp of the first credit record
last_record_time	Numerical	Timestamp of the last credit record
record_count	Numerical	Number of credit records
Default	Numerical	Default ot not

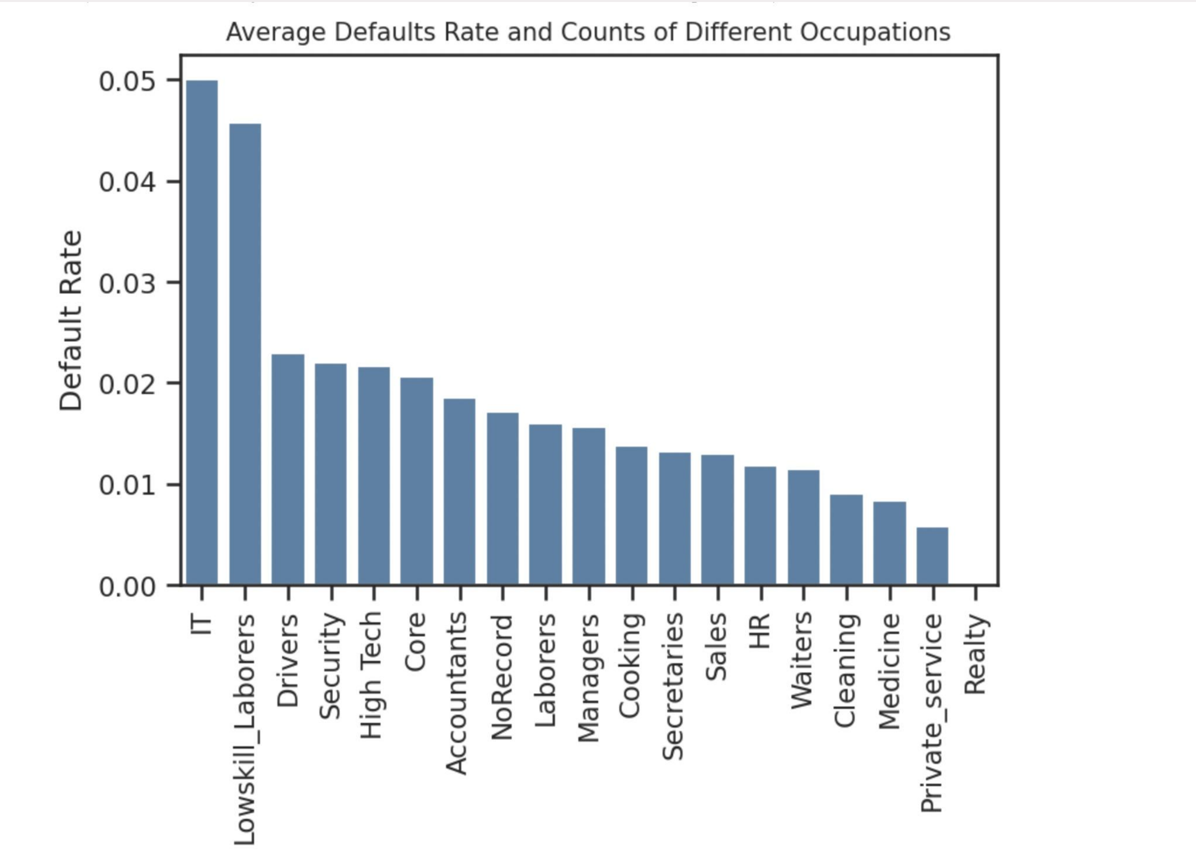
03

Exploratory Data Analysis

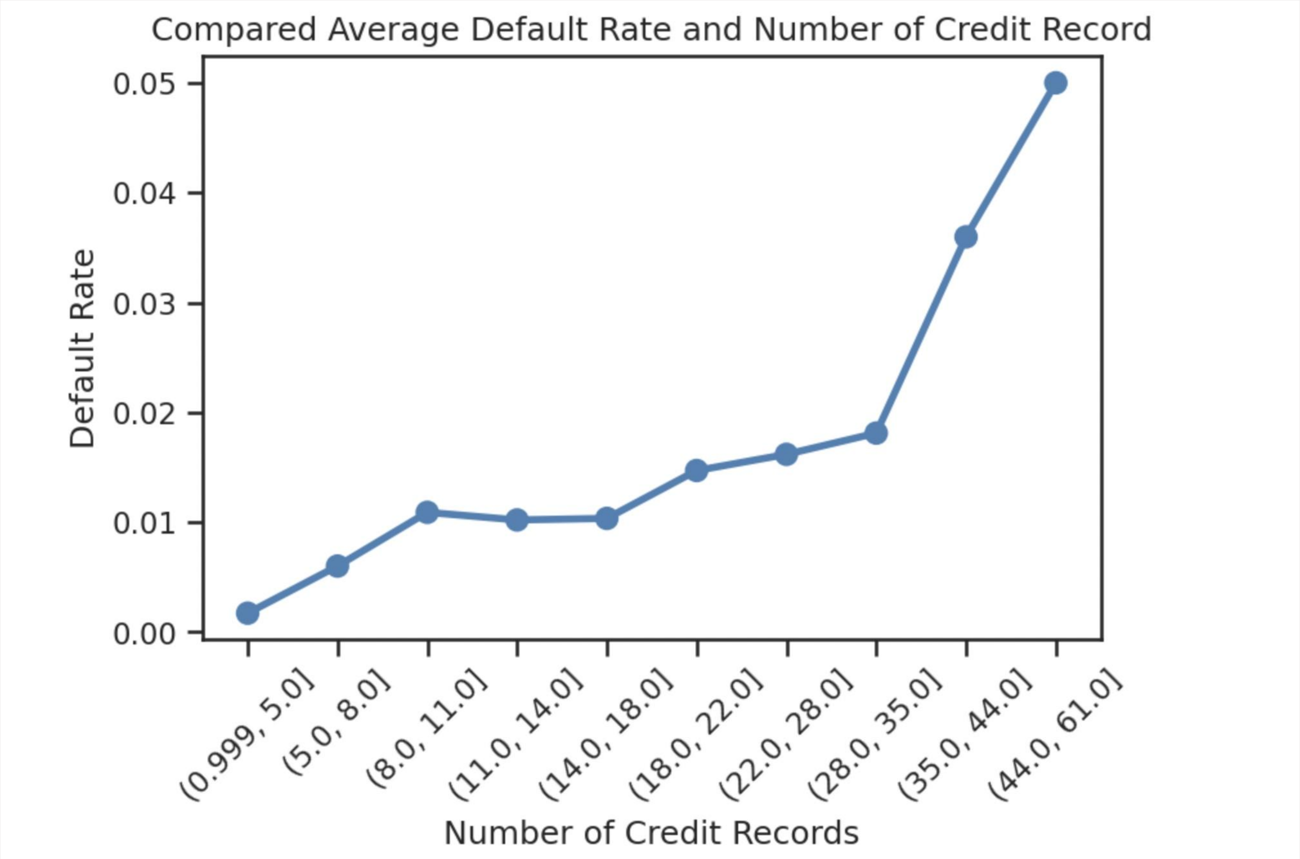
People with more days employed are less likely to default



IT and low skilled workers are more likely to default



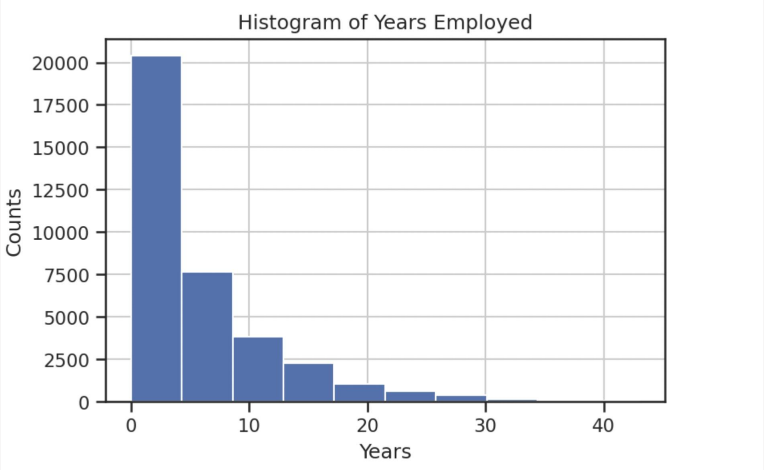
People with more credit records are more likely to default

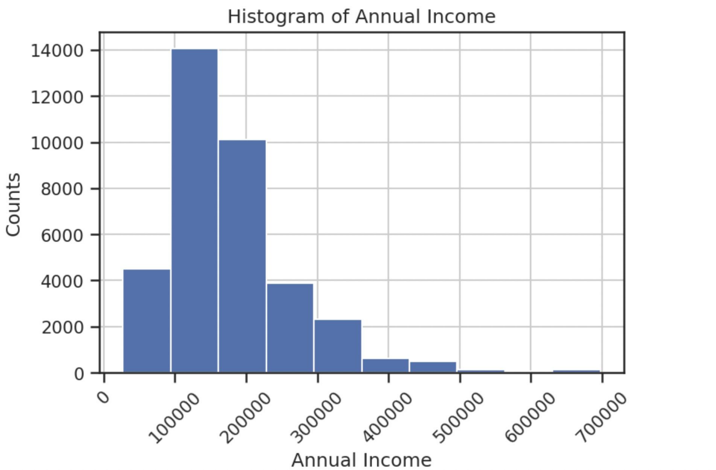


04

Dataset Post Processing for Modeling

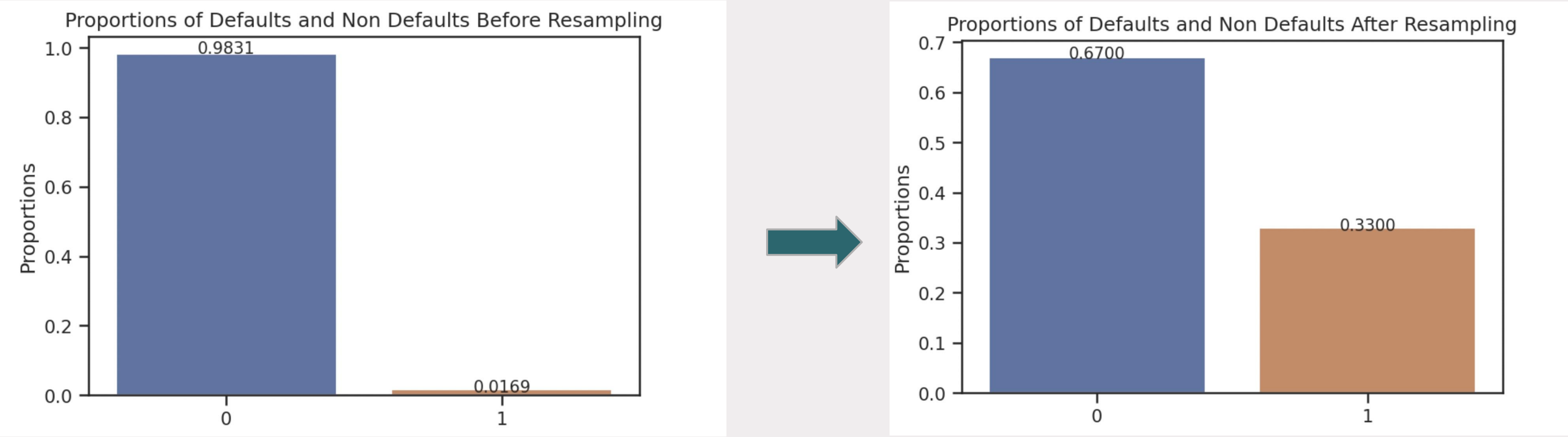
Our data is standardized and the train-test split was 80%/20%





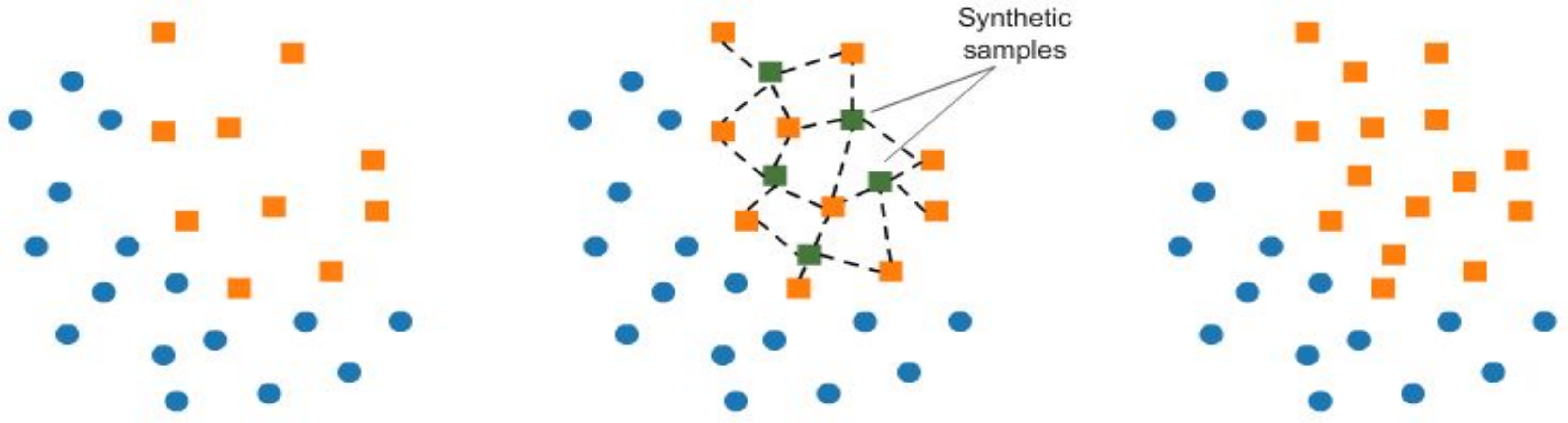
- Most continuous variables appear to be normally distributed and are thus best approximated by standardization
- **80%** of the data will be used for training and cross validation and **20%** will be used for testing
- All model parameters are evaluated by **GridSearchCV** using **5-fold cross validation**
- We will use **f1 score as our evaluation metric** to measure both precision and recall rate

SMOTE package is used to balance and resample the training data



- Synthetic Minority Oversampling Technique or SMOTE is used to synthesize new samples from the minority class
- SMOTE **constructs a latent space with k instances close to each other** and samples new data from the space
- **Validation and testing data will not be resampled for real world generalization**

A big takeaway: we should resample training data only and keep the validation and testing data as it is



If training and validation data are resampled altogether before cross validation, there will be serious **information leakage** for both datasets. Algorithms like KNN could take advantage of that and has cross validation accuracy high up to 100%!

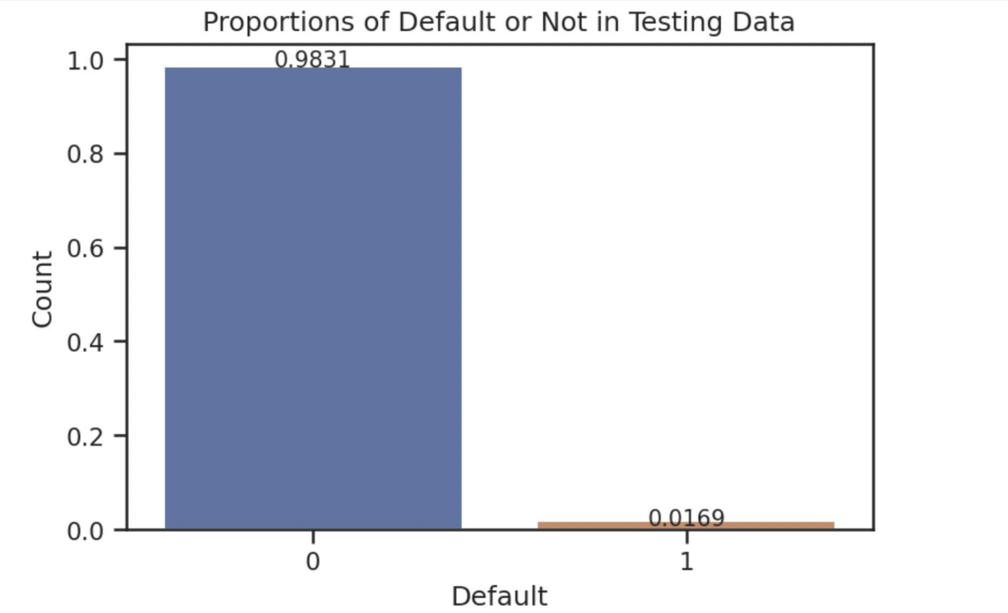
SMOTE component should combine with model as a special pipeline, which will resample training data alone and keep the validation and testing data unchanged in GridSearchCV

05

Predictive Modeling

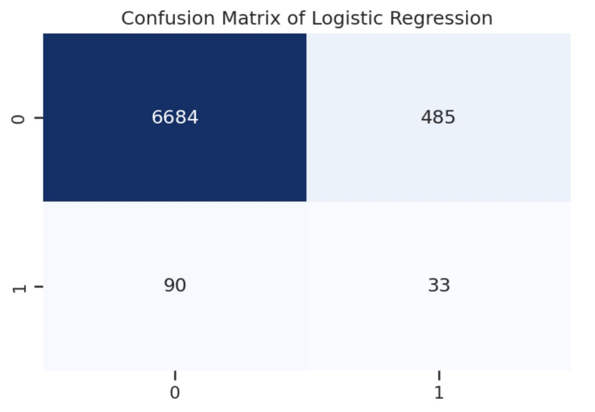
Zero-shot predicting gives a testing accuracy and f1 score of 98.32% and 0% as our baseline

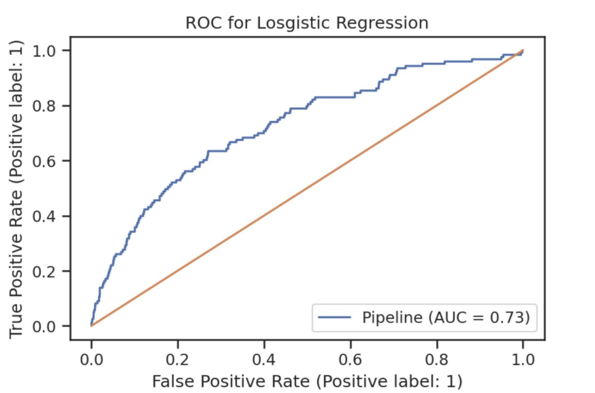
zero-shot predicting achieved 98.32% accuracy but 0% f1 score by predicting all instances as non default



Our zero-shot baseline is not that useful given the huge imbalances between classes in our data

Unregularized logistic regression achieves 10.17% testing f1 score



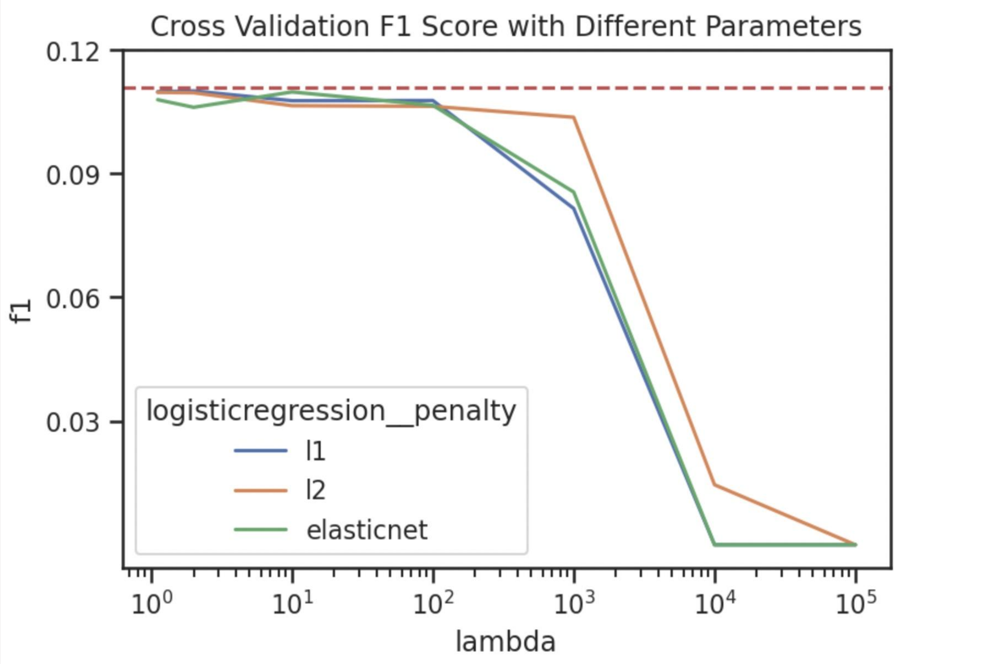


Performance	Not Tuned	Tuned
Cross Val F1	11.07%	11.07%
Training Acc	93.75%	93.75%
Testing Acc	92.00%	92.00%
Precision	6.27%	6.27%
Recall	26.83%	26.83%
F1-Score	10.17%	10.17%

Best Hyperparameters: {solver = ‘saga’}

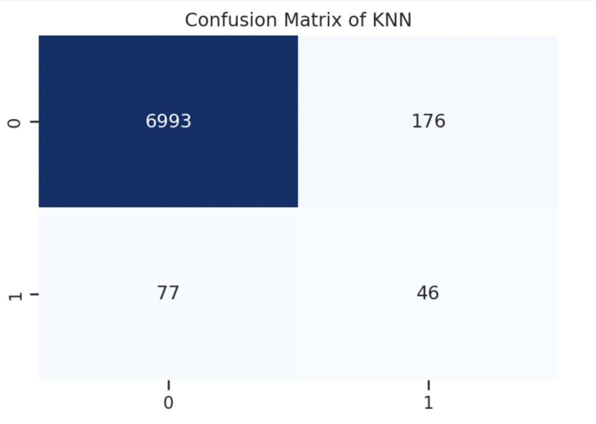
Unregularized logistic regression performs better than the regularized one, as the model might still be **underfitting**

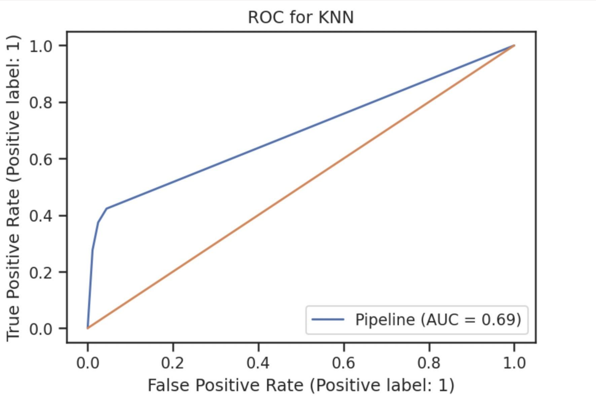
Parameter tuning result of regularized logistic regression



- As the model is still **underfitting**, implementing more regularization worsen model performance on cross validation f1 score
- No much difference found among different regularization penalty methods

Tuned K nearest neighbors achieves 25.57% testing F1 Score



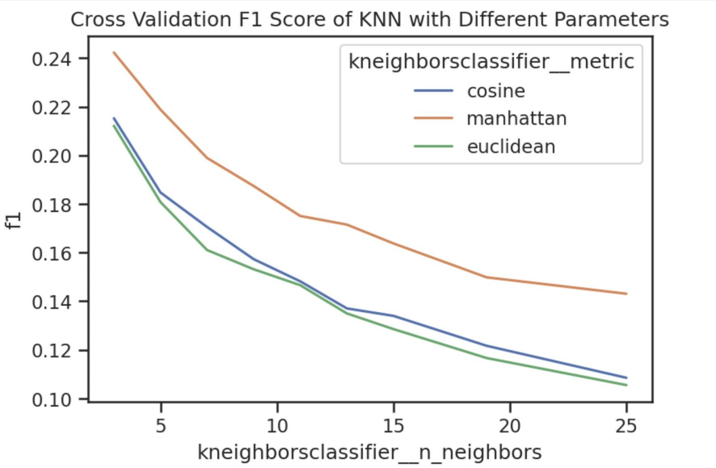


Performance	Not Tuned	Tuned
Cross Val F1	18.07%	24.22%
Training Acc	96.48%	98.70%
Testing Acc	93.69%	96.41%
Precision	11.62%	19.65%
Recall	41.46%	36.59%
F1-Score	18.15%	25.57%

Best Hyperparameters: {neighbors = 3, metric = 'manhattan'}

Tuned KNN has a great improvement on recall rate and precision, suggesting nonlinear pattern in our data structure

Parameter tuning result of K nearest neighbors

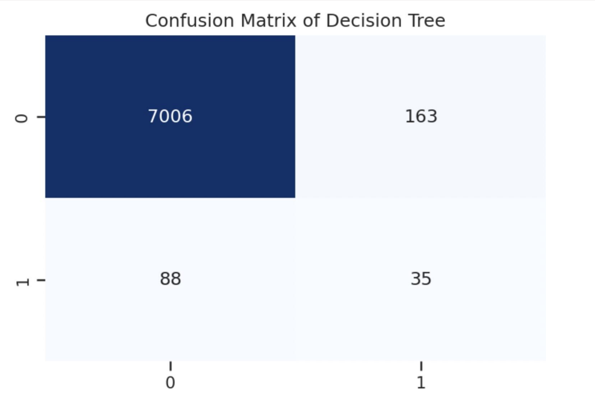


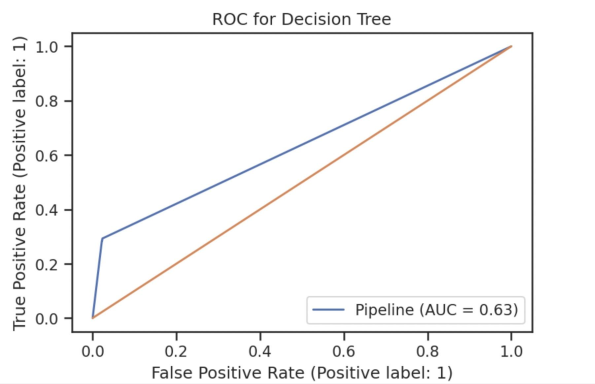
$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

Manhattan Distance Formula

- KNN model with less n_neighbors perform better than those with more neighbors, suggesting a strong need of **nonlinear fit** for our data.
- Manhattan distance metric tends to perform better than cosine and euclidean, suggesting that the **absolute difference** among features is important for classifying default behavior

Tuned Decision Tree achieves a 21.81% testing f1 score



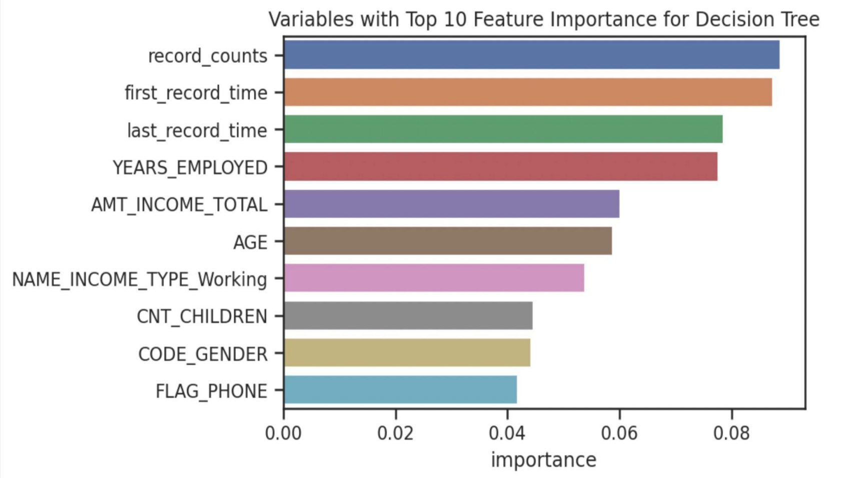


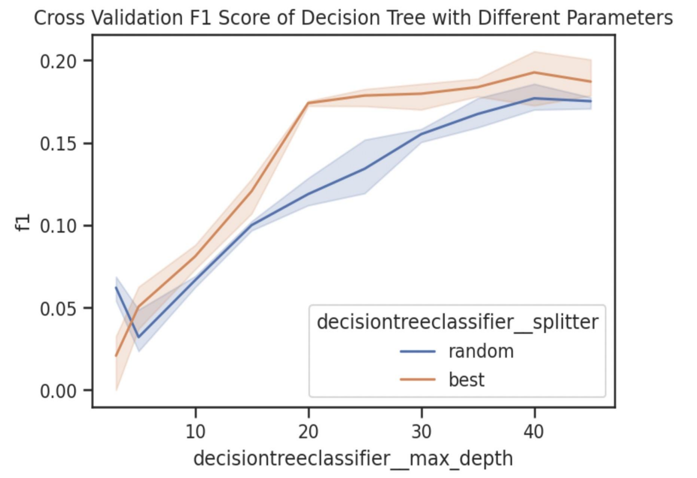
Performance	Not Tuned	Tuned
Cross Val F1	17.86%	20.54%
Training Acc	99.93%	99.89%
Testing Acc	96.43%	96.56%
Precision	15.23%	17.68%
Recall	24.39%	28.46%
F1-Score	18.75%	21.81%

Best Hyperparameters: {max_depth = 40, Splitter = 'Best'}

Tuned decision tree performs better than logistic regression but worse than KNN

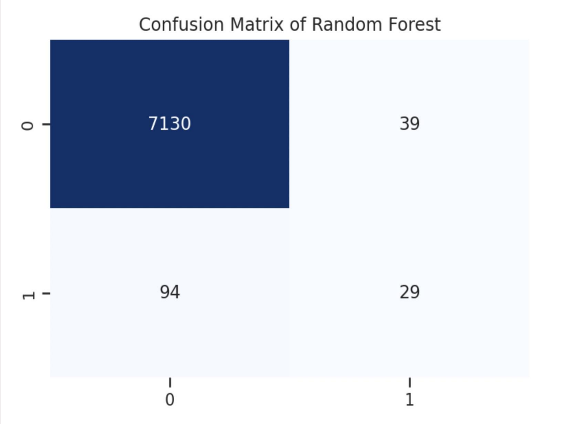
Variable Importance and parameter tuning result of Decision Tree

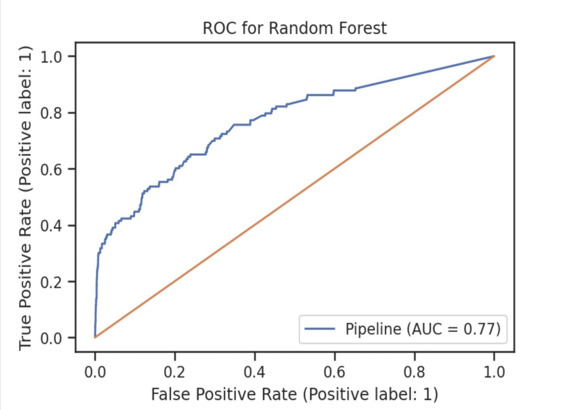




- Decision Tree model with larger max_depth and best split tends to perform better
- Our engineered variables rank top 3 in variable importance of decision tree - **record_counts, first_record_time, last_record_time**!

Tuned random forest achieves 30.37% testing f1 score



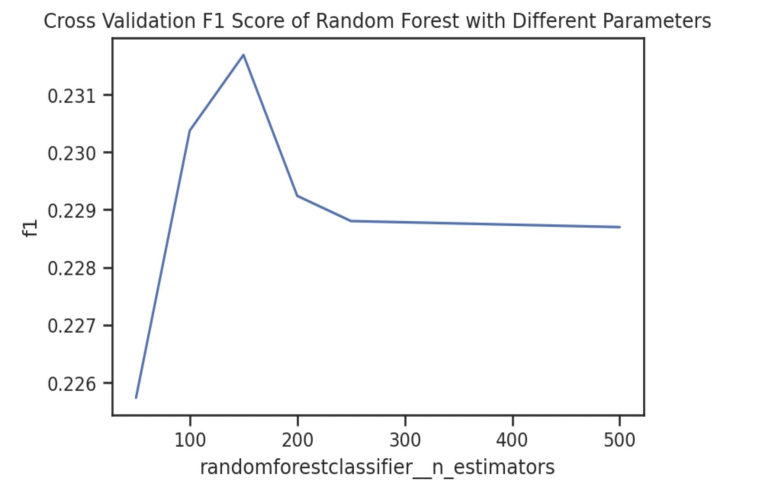


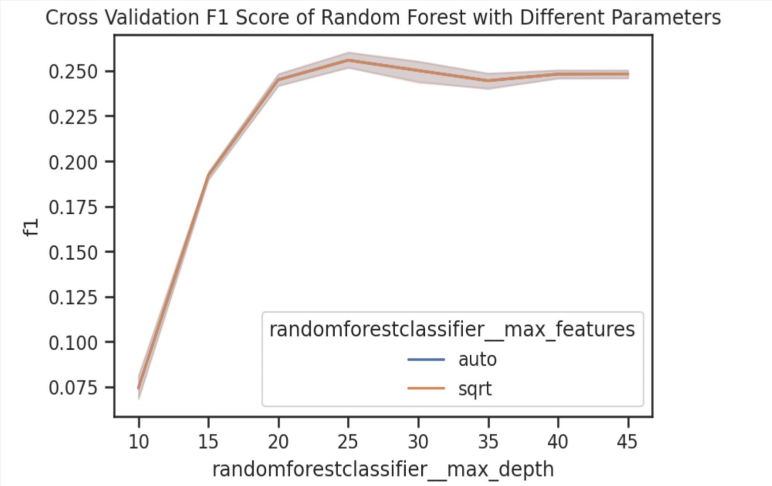
Performance	Not Tuned	Tuned
Cross Val F1	19.59%	28.07%
Training Acc	99.09%	99.82%
Testing Acc	98.16%	98.18%
Precision	36.59%	42.65%
Recall	12.20%	23.58%
F1-Score	18.29%	**30.37%**

Best Hyperparameters: {max_depth = 25, n_estimators = 150}

Random forest has the **highest f1 score and precision**, which is a big improvement compared to Decision Tree. The model is much more **robust** against noise compared to the other models

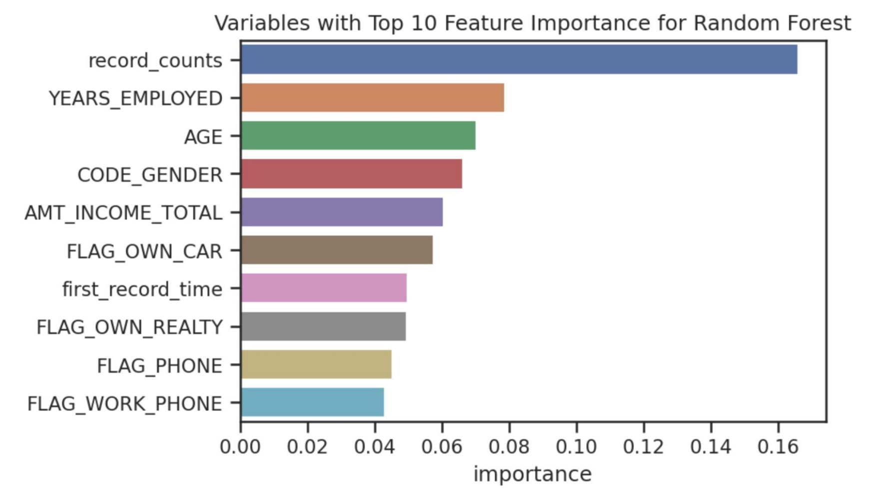
Parameter tuning result of random forest





- Random forest model with max_depth of 25 for each tree and 150 trees tends to perform better than others
- Max features setting is not very important, which is similar to Decision Tree models

Variable importance of random forest



- Our engineered variable **record counts** still ranks top in feature importance
- Dummy variables with high importance are similar in both random forest and decision tree

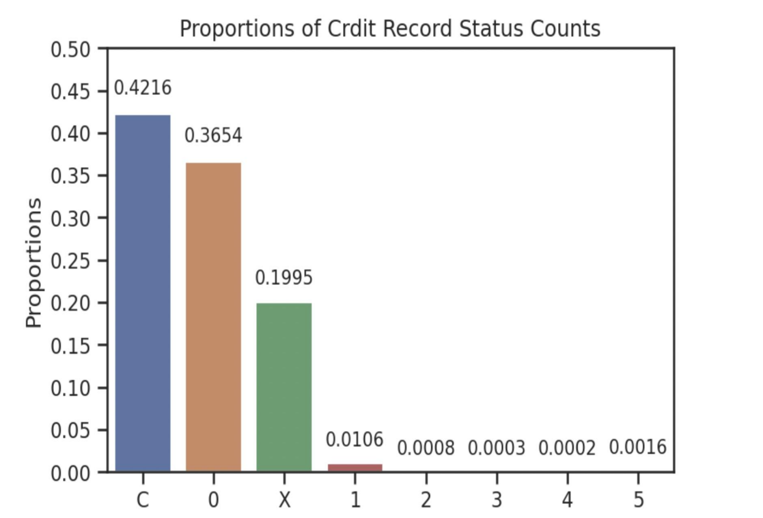
Model Performance Overview

Performance	Logistic Regression	KNN	Decision Tree	Random Forest	Neural Network
Cross Val F1	11.07%	24.22%	20.54%	28.07%	N/A
Nested CV F1	**10.31%**	**25.72%**	**21.70%**	**28.32%**	N/A
Training Acc	93.75%	98.70%	99.89%	99.82%	99.44%
Testing Acc	92.00%	96.41%	96.56%	98.18%	96.61%
Precision	6.27%	19.65%	17.68%	42.65%	19.31%
Recall	26.83%	36.59%	28.46%	23.58%	31.71%
F1-Score	10.17%	25.57%	21.81%	30.37%	24.00%

05

Obstacles

Class imbalance is the toughest issue to predict default behaviours



The extremely low default class has led to the following two problems:

1. **The choice of standards to classify a person as default or not**
   To approximate real world credit default rate, we need to classify less people as default resulting in more imbalanced data.
2. **The choice of resampling technique and the correct way to implement it**
   Resampling training and validation data together gives inconsistent cross validation results about model performance

Generalization of our models will be good but the performance is still limited for real world applications

1. **Tuning and testing on data without resampling help model generalize**
   All models are tested and cross-validated on data without resampling: performance estimate should be good and can be generalized to unseen real world data

2. **Cross validation gives unbiased and fair results**
   All models parameters are 5-fold cross validated

3. **Performance of all models is limited due to lack of data**
   Even with the best model, we can only achieve f1 score of 30.37% due to lack of default data. All of our models might not be able to satisfy real world needs

About 76% of default people might still get their credit card if our best model is the final judge

06

Takeaways

1. More data is needed to learn credit default behavior
2. Number of credit records, days employed, and etc. are all important factors to consider when measuring default probabilities
3. Model with higher ability of non-linear fit may perform better in predicting default behavior
4. Sampling method should be implemented on training data only

THANKS!