Proposal for Potential Candidate Detection

Company XYZ

Yang Liu Kunz

TOC

Overview

1. The aim of this project is to use the credential/demographics/experience related data we collected from our training platform to find out the optimal solution to detect enrollees that are open for career change and connect them with our clients who are hiring.
2. KPI from the recruiting team: reach out to more than half of Target enrollees with minimum costs.
3. The dataset we used for analysis contains information of 18,359 unique enrollees, among them 2425 enrollees were open for a career change. The ratio of Non-Target enrollees to Target enrollees is 6:1 (imbalanced data).
4. After model training, the best model enabled us to reach out to 70.5% of Target enrollees in the test data set with minimum costs.

In real business case we should have a clear KPI such as:
1) Reach: as many potential Target enrollees reached as possible (cost doesn't matter)
2) Cost per Target: reach the most potential Target enrollees with the least financial cost
Here in this study we will aim for both

Dataset description

	enrollee_id	city	city_development_index	gender	relevent_experience	enrolled_university	education_level	major_discipline	experience	company_size	company_type	last_new_job	training_hours	target
0	23798	city_149	0.689	Male	Has relevent experience	no_enrollment	Graduate	STEM	3	100-500	Pvt Ltd	1	106	0
1	29166	city_83	0.923	Male	Has relevent experience	no_enrollment	Graduate	STEM	14	<10	Funded Startup	1	69	0
2	46	city_16	0.910	NaN	Has relevent experience	no_enrollment	Graduate	STEM	6	50-99	Public Sector	2	4	0
3	18527	city_64	0.666	Male	Has relevent experience	no_enrollment	Graduate	STEM	14	50-99	Pvt Ltd	1	26	0
4	21751	city_100	0.887	NaN	No relevent experience	no_enrollment	Masters	STEM	8	NaN	NaN	2	88	1

- Data sources: Company XYZ training platform database
- 18,359 observations of 14 variables(4numerical, 10 categorical)
  - Input variables:
    - Experience related variables: experience, last_new_job, relevant_experience
    - Employer related variables: company_type, company_size
    - Education related variabels: education_level, major_discipline, enrolled_university
    - Gender: gender
    - City related variables: city (name), city_development_index(float)
    - Training related variables: training_hours(int)
  - Target(int): open for career change or not (0=no, 1=yes)

*enrollee_id(int) is excluded from input variables, variable city is already hashed in original dataset

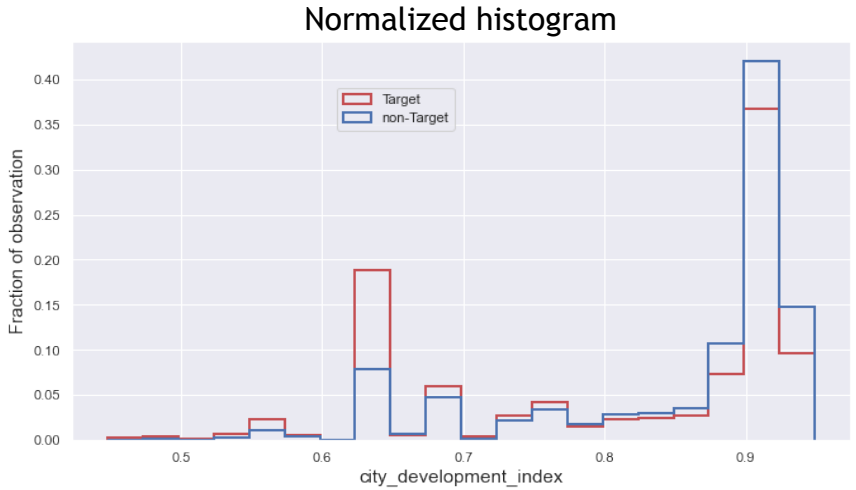
Data Wrangling

Group	Variable Feature	Variable Names	Missing Value Percentage	Process Method
Group 1	experience related variables	relevent_experience	0	data type correction
		experience	0.32	data type correction, missing value imputation (fill with mode)
		last_new_job	2	data type correction, missing value imputation (fill with mode)
Group 2	employer related variables	company_type,	27.45	data grouping, missing value imputation (fill with KNN imputation) *KNN imputation will automatically convert the variable's data type to float
		company_size	26.03	data grouping, missing value imputation (fill with KNN imputation)
Group 3	education related variables	education_level,	2.49	data grouping, data type correction, missing value imputation (fill with 0)
		major_discipline	15.46	data grouping, data type correction, missing value imputation (fill with 0 or mode) *when education_level equals to 0 (None) or 1 (high school or primary school), fill missing value with 0, then fill the rest with 0
		enrolled_university	1.86	fill with mode
Group 4	gender	gender	22.32	data type correction, missing value imputation (fill with mode)
Group 5	city related variables	city	-	data grouping, data type correction (dummy encoding)
		city_development_index	-	-
Group 6	training related variables	training_hours	-	-

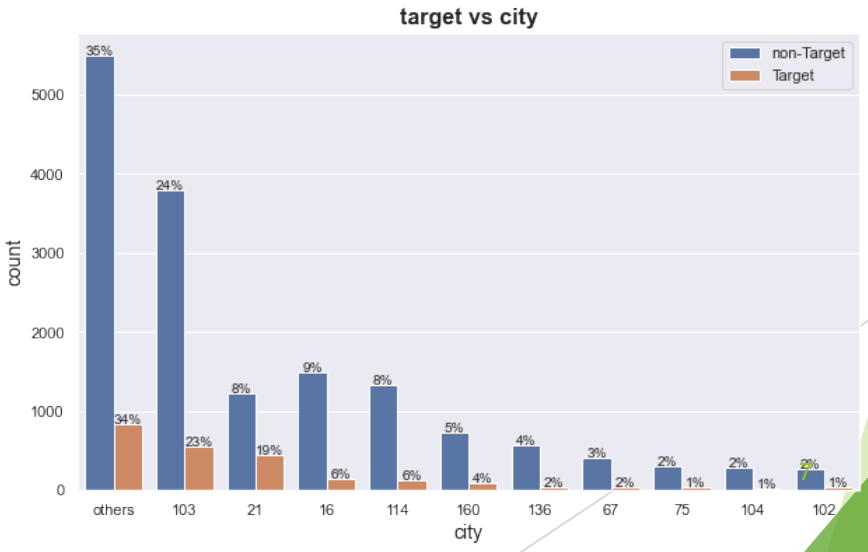
Key Findings

1 City vs. Target

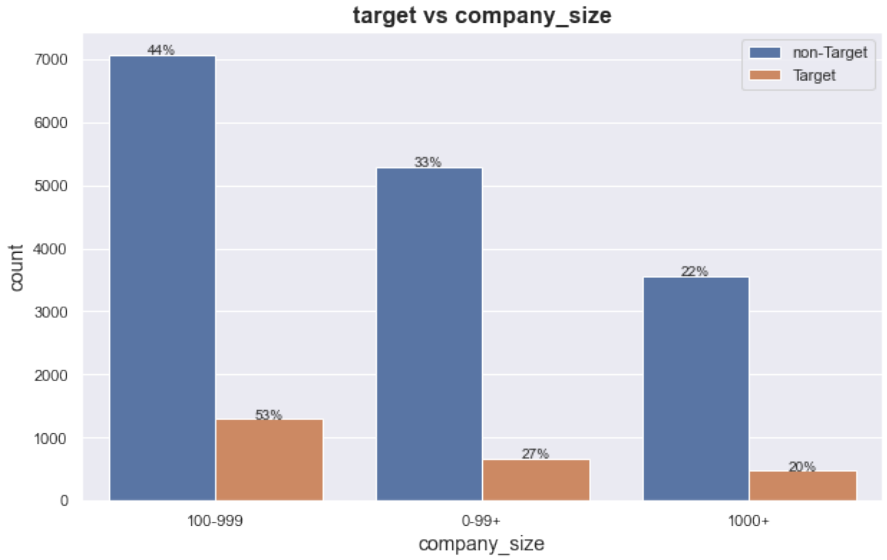
City	city_development_index	Total enrollee	Target enrollee
103	0.920	4358	563
21	0.624	1672	453
16	0.910	1654	154

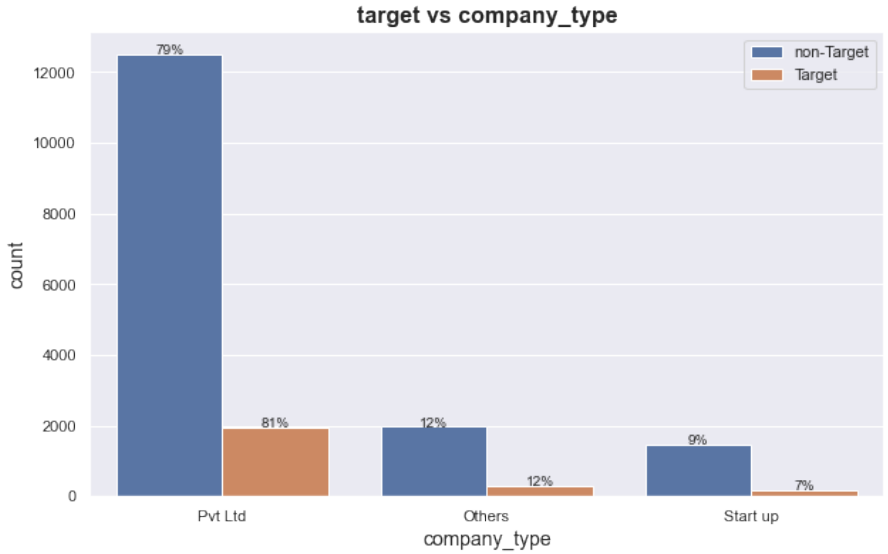


- 32% of the total enrollees, 30% of the Target enrollees are from cities where city_development_index ranges 0.910 ~ 0.920 (city 103, 16)
- 42% of the total enrollees, 48% of the Target enrollees are from city 103, 21, 16.



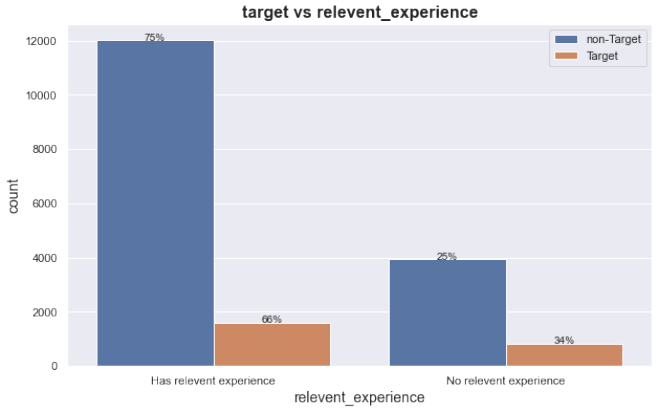
❷ Employer vs. Target



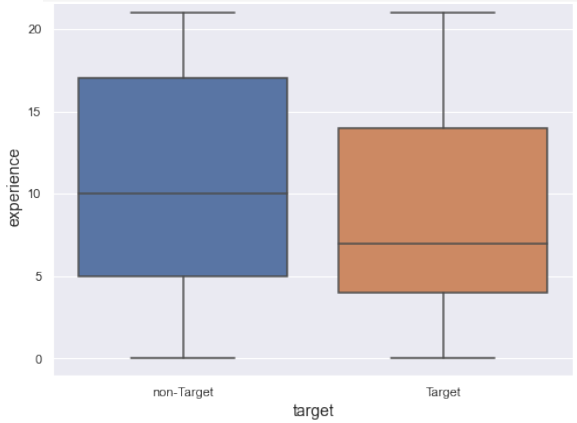


- 46% of the total enrollees, 53% of Target enrollees are from medium size company (100~999)
- 79% of the total enrollees, 81% of the Target enrollees are from private limited company

3 Experience vs. Target

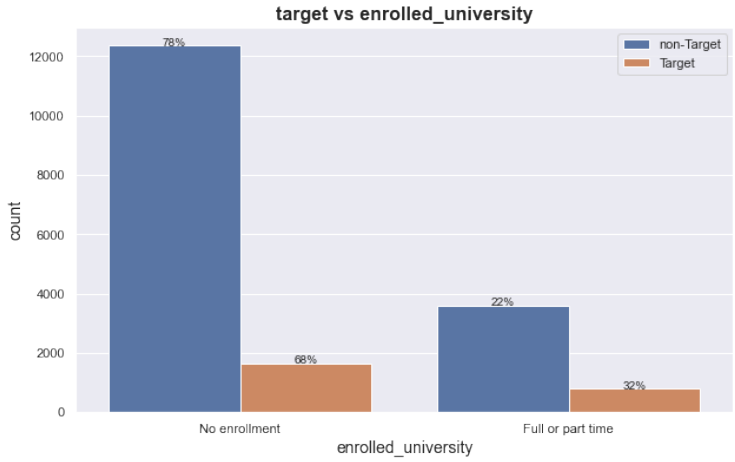


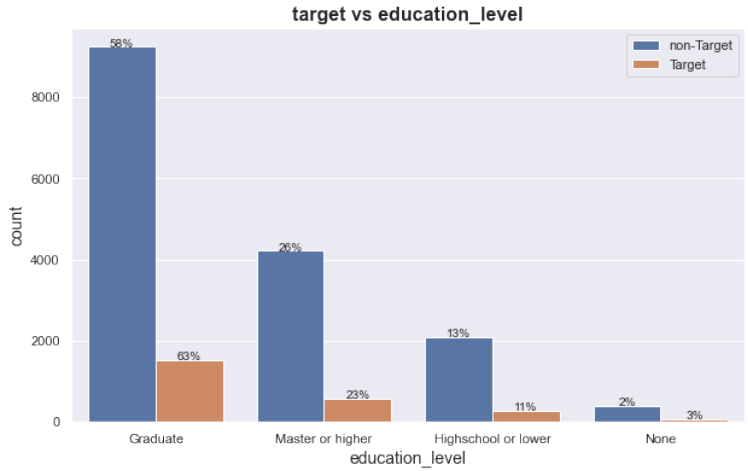
	count	mean	std	min	25%	50%	75%	max
target								
non-Target	15934.0	10.872725	6.763342	0.0	5.0	10.0	17.0	21.0
Target	2425.0	9.164124	6.625679	0.0	4.0	7.0	14.0	21.0

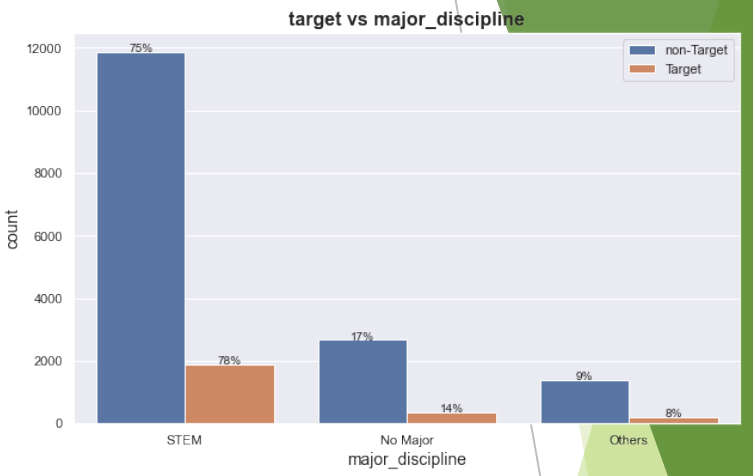


- 66% of the Target enrollees has relevant experience in data science/analytics field
- 45% of the Target enrollees have stayed in their current company for 1 or less than 1 year. 15% of them have stayed in current company longer than 4 years.
- non-Target enrollee compares to Target enrollee, has slightly longer average working years and wider range of working experience in years. (21 years means longer than 20 years)

4 Education vs. Target

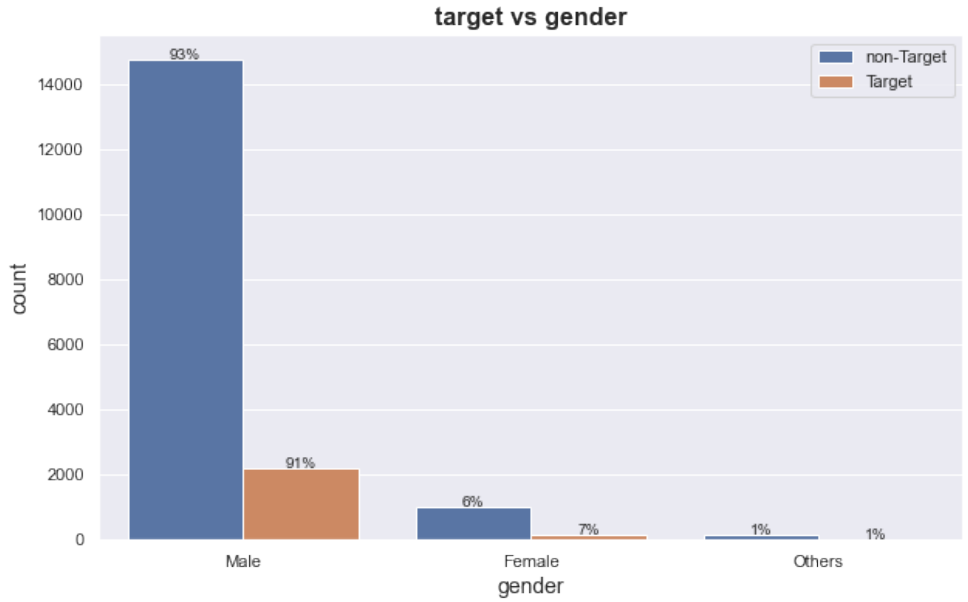






- 32% of the Target enrollees are enrolled in Full or part time university.
- Graduate degree is the most common degree among all enrollees, 63% of the Target enrollees have a Graduate degree
- Major STEM is the most common major among all enrollees, 78% of the Target enrollees have a STEM background

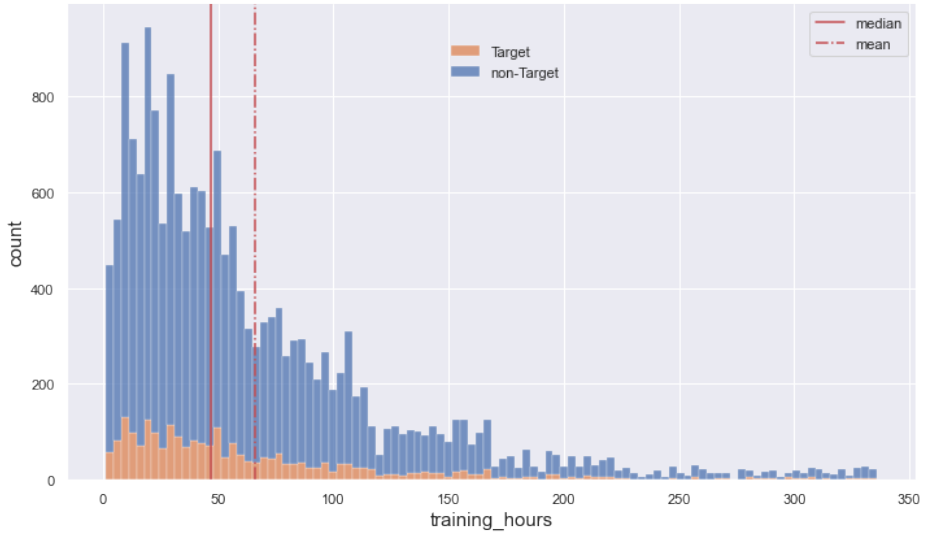
5 Gender vs. Target

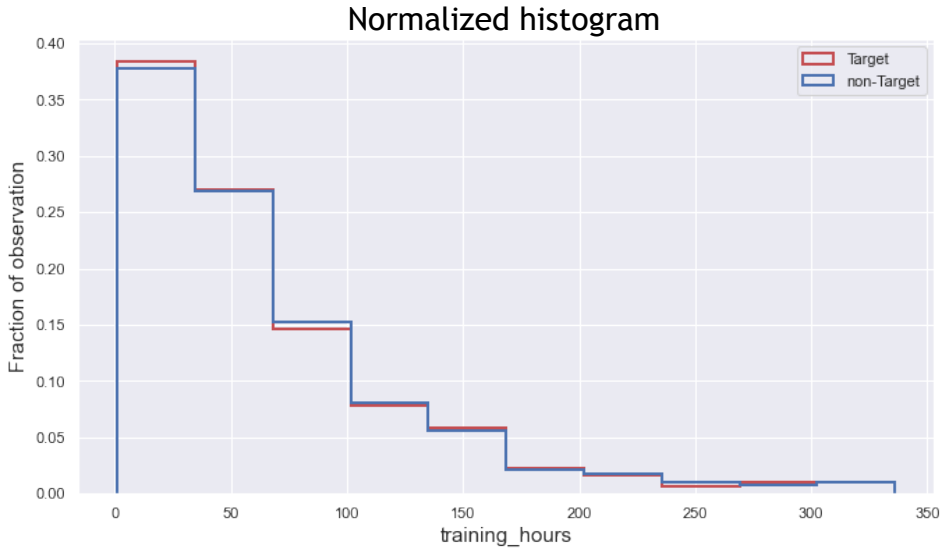


- Male is very dominant. 92% of the total enrollees are male
- 91% of the Target enrollees are male.

6 Training_hours vs. Target

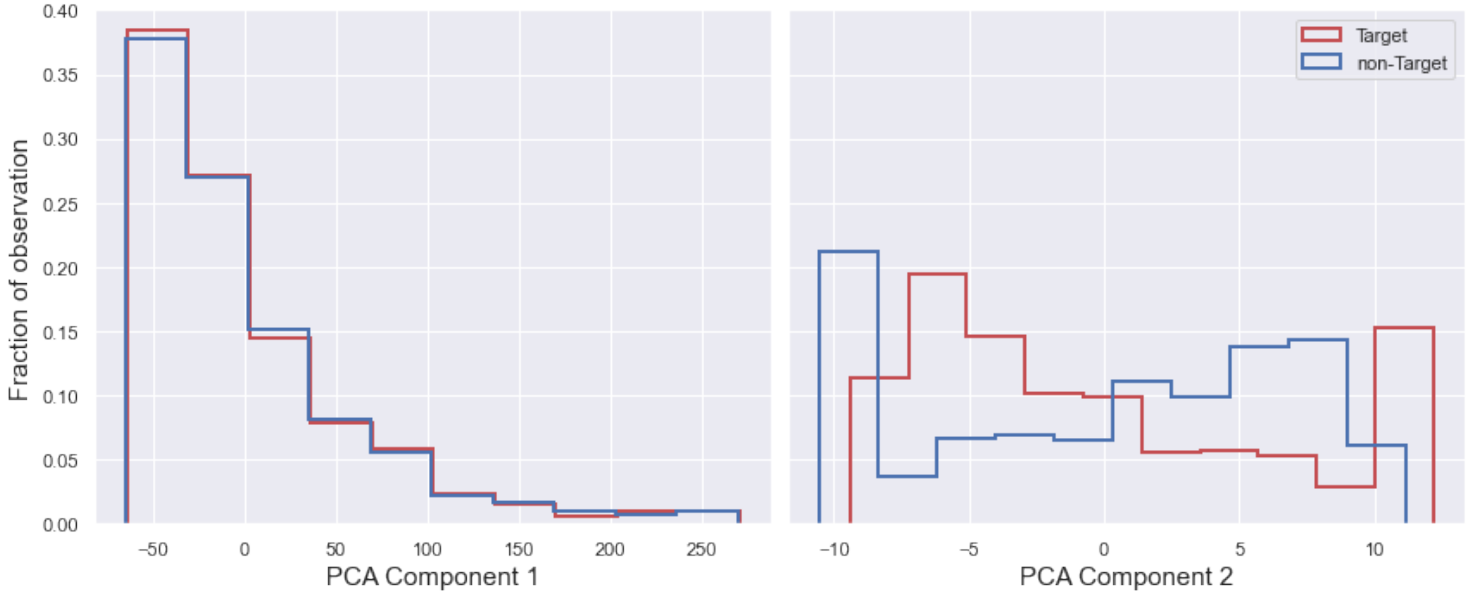
target	count	mean	std	min	25%	50%	75%	max
non-Target	15934.0	66.011485	60.908040	1.0	23.0	47.0	89.0	336.0
Target	2425.0	65.160000	60.743027	1.0	23.0	47.0	87.0	336.0





- Regardless of the class, the training time distribution is skewed right with a wide range of training time from 1 hour to 336 hours.
- Two classes share lots of similarities in terms of summary statistic and distribution. The average training hours for Target and non-Target enrollee only has less than 1hour difference.

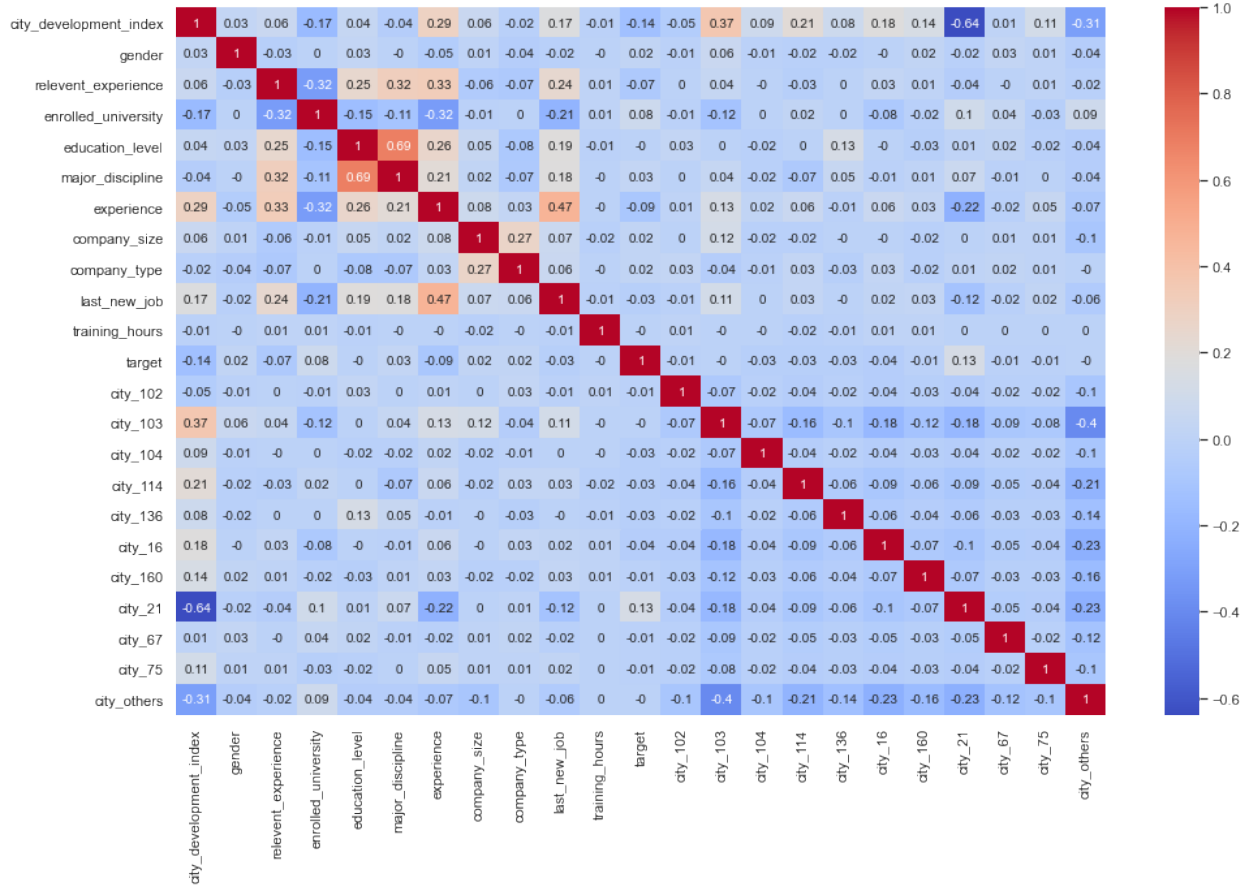
7 Feature Space Overlap



- Converted categorical variables to numerical variables
- Applied PCA reduction(2 components) to each class (enrollee_id excluded from variables)
- Note lack of distinction in distribution and significant amounts of overlap between 2 classes

*histogram is normalized

8 Multicollinearity



- Variables are not highly correlated
- As we dummy encoded variable city, therefore the correlation coefficient may not be statistically robust due to these binary values

Modeling and Analysis

Model Selection

Model Type	No.	Binary Classifier
Probabilistic Classifier	1	Logistic Regression
	2	Gaussian Naïve Bayes
Instance-based Classifier	3	k-Nearest Neighbors
Support Vector Machine	4	Support Vector Machine
Tree Based Classifier	5	Decision Tree
	6	Random Forest (Ensemble Bagging)
	7	Gradient Boosting (Ensemble Boosting)
Optimized Gradient Boosting	8	XGBoost (Ensemble Boosting)
	9	LightGBM (Ensemble Boosting)
	10	CatBoost (Ensemble Boosting)

1. To gain better result, we explored 10 models with our dataset, identified the one with the best performance to be the final model.

2. You can find pros and cons for each model on this list from appendix

Train/Test Split

	non-Target Count	Target Count	Target %
Full Dataset	15,934	2,425	13.209
Training set	11,154 (70%)	1,697	13.205
Testing set	4,780 (30%)	728	13.217

Split data to Training/Testing set → Train 10 baseline models on Training set, apply 5Folds Cross Validation → Test models on Training set and Testing set

Metrics

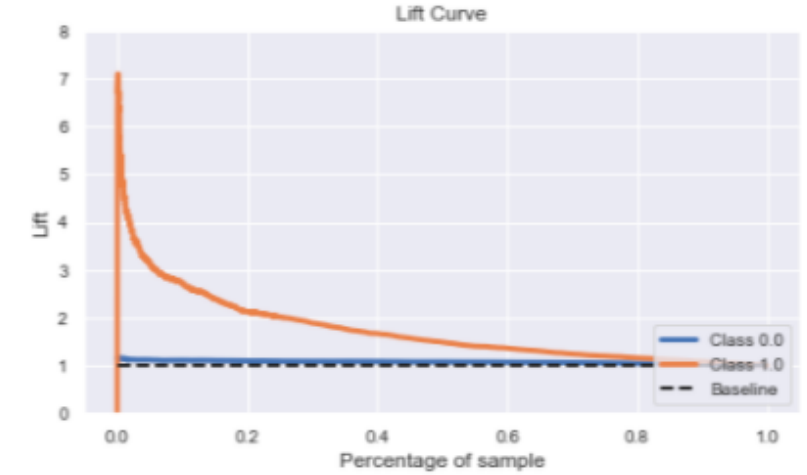
- Precision
- Recall
- F1score
- ROC_AUC
- Customized metrics:
  1) Target in top 20%: the number of positive classes the model is able to catch within the top 20% of the test dataset when you sort the test dataset by their prediction scores (predicted probabilities) in descending order.

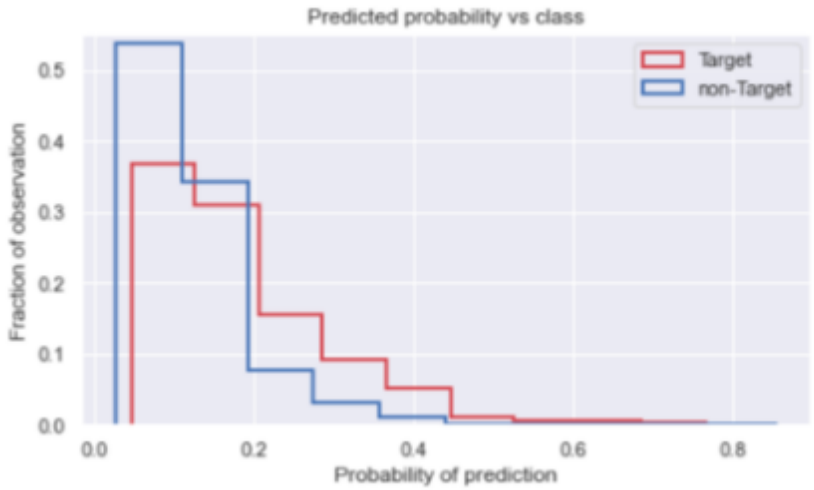
  2) Target in top 50%: the number of the positive classes the model is able to catch within the top 50% of the dataset when you sort the test dataset by their prediction scores (predicted probabilities) in descending order.
     *There are 728 positive classes (Target) in the Testing set

Best Model: Gradient Boosting

Training

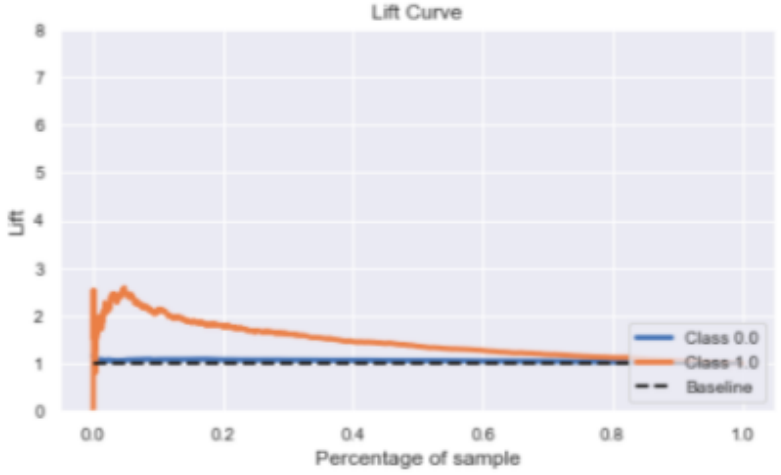
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
GB	0.886	0.018	0.036	0.705	719.0	1259.0

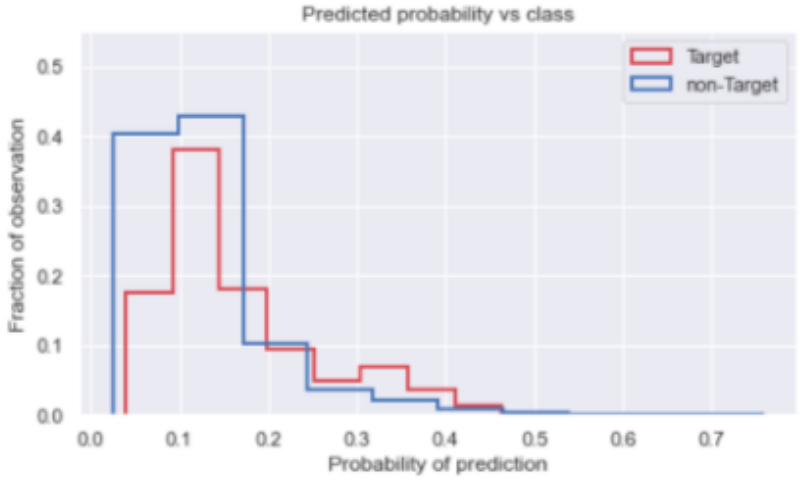




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
GB	0.25	0.003	0.005	0.645	259.0	493.0



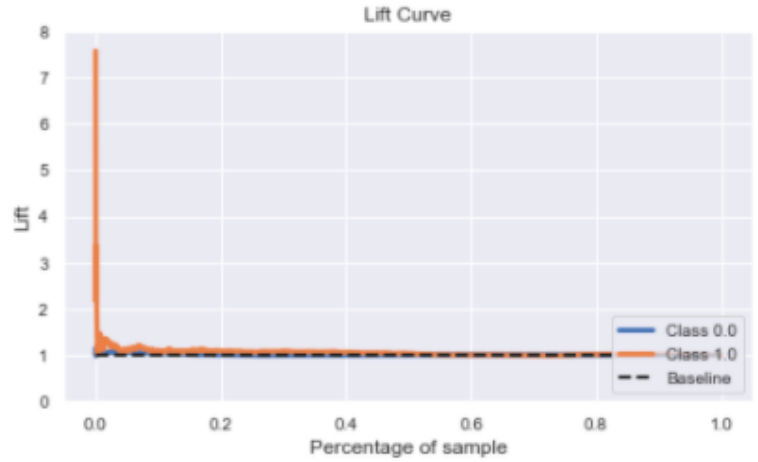


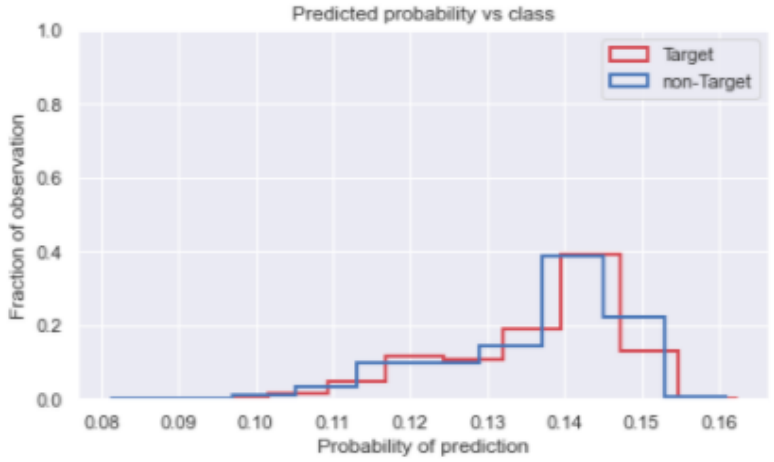
Worst Model: Support Vector Machine

The result for other models can be found in Appendix

Training

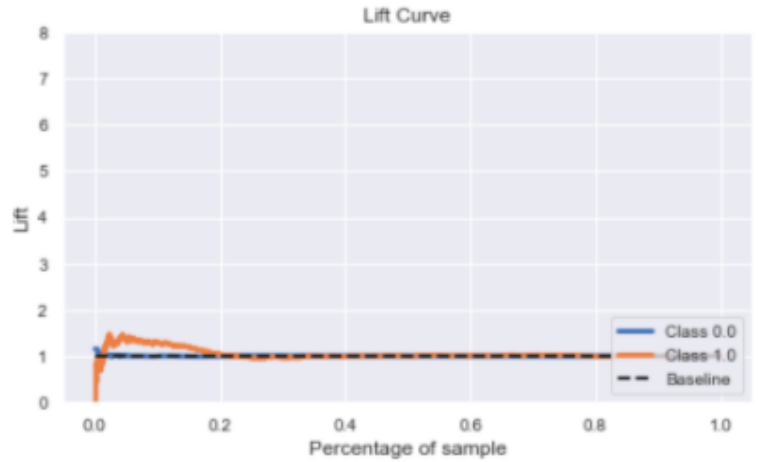
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
SVC	0.0	0.0	0.0	0.515	370.0	878.0

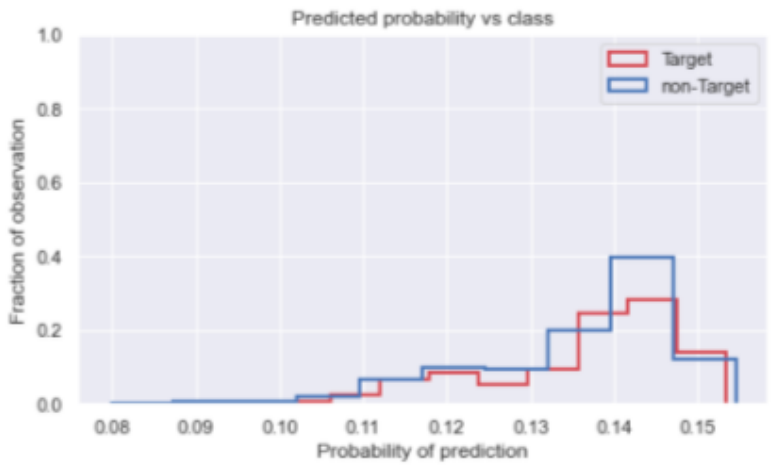




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
SVC	0.0	0.0	0.0	0.507	152.0	364.0





Overall Model evaluation on Testing Set

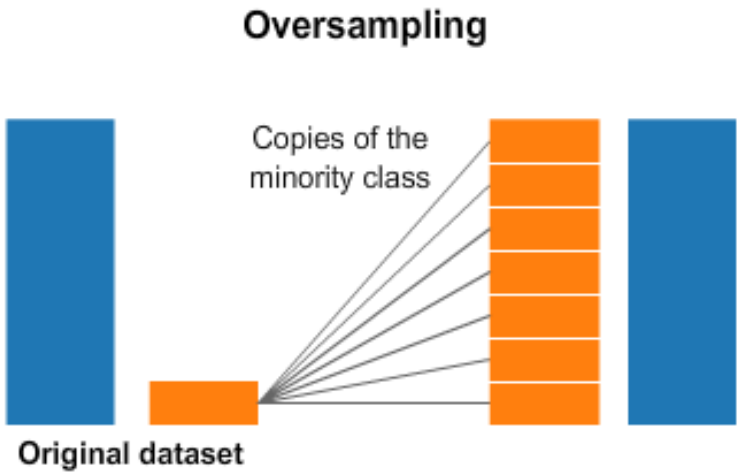
Model	Precision	Recall	F1score	ROC_AUC	Target in top 20%	Target in top 50%
LR	0	0	0	0.629	262	483
CatBoost	0.244	0.014	0.026	0.630	261	482
GB	0.286	0.003	0.005	0.645	259	493
GaussianNB	0.230	0.337	0.274	0.608	251	458
LightGB	0.333	0.016	0.031	0.618	237	473
XGB	0.220	0.037	0.063	0.599	225	456
RF	0.270	0.051	0.086	0.594	211	456
KNN	0.210	0.041	0.069	0.550	189	413
DT	0.179	0.220	0.197	0.533	186	391
SVC	0	0	0	0.507	152	364

1. As we expect the model to capture as many Target enrollees as possible from the Testing set, therefore instead of accuracy, we need to focus on the Recall score.

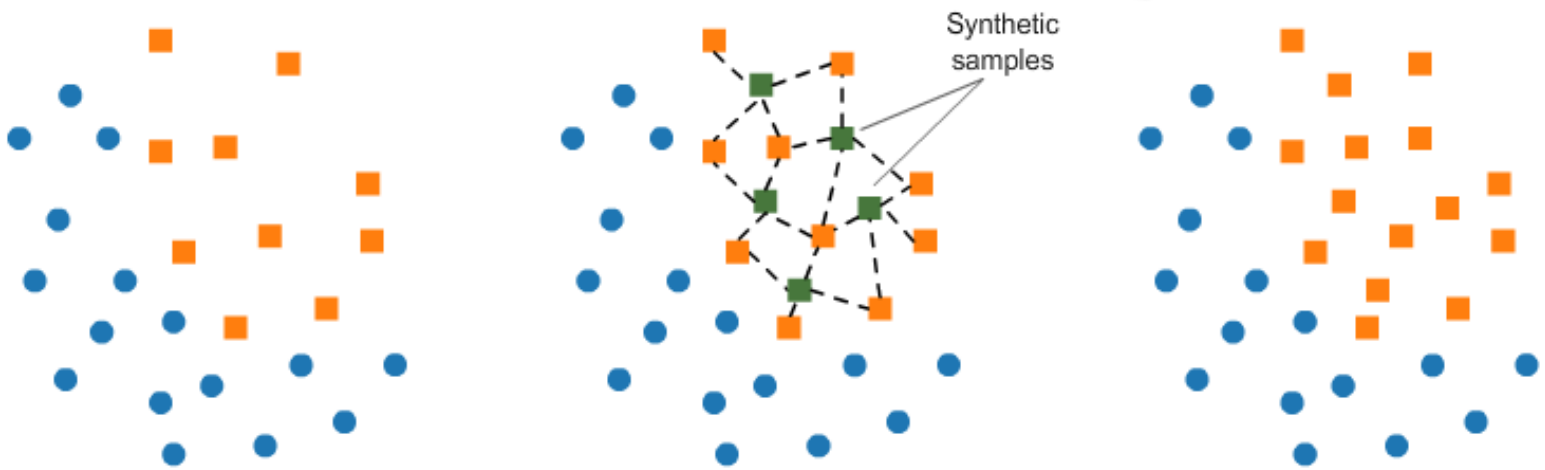
2. However as we can see most of the models have poor Recall due to the class imbalance issue in the dataset.

3. Therefore we tried to improve the model performance by oversampling

Synthetic Minority Oversampling Technique (SMOTE)



Example of SMOTE finds 4 nearest neighbors



Overall Model evaluation on Testing Set (after SMOTE)

Model	Precision	Recall	F1score	ROC_AUC	Target in top 20%	Target in top 50%
LR	0.192	0.521	0.281	0.623	253	480
Gaussian NB	0.151	0.674	0.246	0.598	251	441
KNN	0.144	0.455	0.218	0.541	182	391
SVC	0.153	0.647	0.247	0.58	212	440
DT	0.162	0.195	0.177	0.52	171	387
RF	0.237	0.065	0.102	0.588	206	450
GB	0.381	0.033	0.061	0.633	260	475
XGB	0.257	0.038	0.067	0.6	215	450
LightGB	0.309	0.029	0.053	0.628	236	484
CatBoost	0.315	0.04	0.071	0.623	247	481

*SMOTE was applied only to each test fold in the 5 Folds Cross Validation

Change in percentage after SMOTE

Model	Precision	Recall	F1score	ROC_AUC	Target in top 20%	Target in top 50%
LR	∞	∞	∞	-0.95	-3.44	-0.62
Gaussian NB	-34.35	100.00	-10.22	-1.64	0.00	-3.71
KNN	-31.43	1009.76	215.94	-1.64	-3.70	-5.33
SVC	∞	∞	∞	14.40	39.47	20.88
DT	-9.50	-11.36	-10.15	-2.44	-8.06	-1.02
RF	-12.22	27.45	18.60	-1.01	-2.37	-1.32
GB	33.22	1000.00	1120.00	-1.86	0.39	-3.65
XGB	16.82	2.70	6.35	0.17	-4.44	-1.32
LightGB	-7.21	81.25	70.97	1.62	-0.42	2.33
CatBoost	29.10	185.71	173.08	-1.11	-5.36	-0.21

Before SMOTE Target in top 20%	After SMOTE Target in top 20%		Before SMOTE Target in top 50%	After SMOTE Target in top 50%
262 (LR)	260 (GB)		493 (GB)	480 (LR)
261 (CatBoost)	253 (LR)		483 (LR)	475 (GB)
259 (GB)	251 (GNB)		482 (CatBoost)	441 (GNB)

Key Metrics: which one is the key metrics

Hypothesis 1: Recall is the key metrics
Best Recall: 0.674
Best model: GaussianNB after SMOTE
Confusion matrix:

	Predicted Target	Predicted non-Target
Target	TP: 491	FN:237
non-Target	FP: 2770	TN: 2010

Potential Targets approach list: 491+2770 = 3261
Cost per email sent: $1
Cost Per Target: 3261*1/491 = $6.6

Hypothesis 2: Targets in top 50% is the key metrics
Best Target in top 50%: 493
Best model: Gradient Boosting before SMOTE
Potential Targets approach list: 50% of the Testing set, 5508*0.5 = 2754
Cost per email sent: $1
Cost per Target: 2754/493 = $5.6

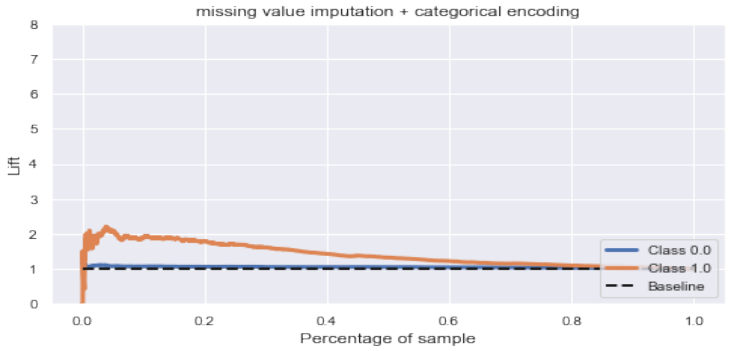
1) KPI for recruiting team:
reach out to more than half Target enrollees (728*0.5 = 364) with minimum costs

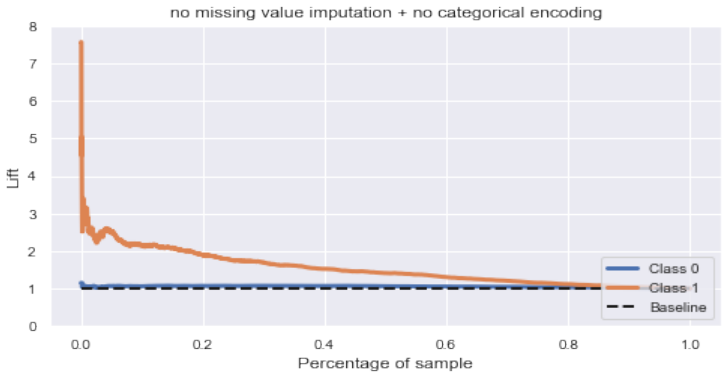
2) Conclusion:
Target in top 50%
is the key metrics for model comparison

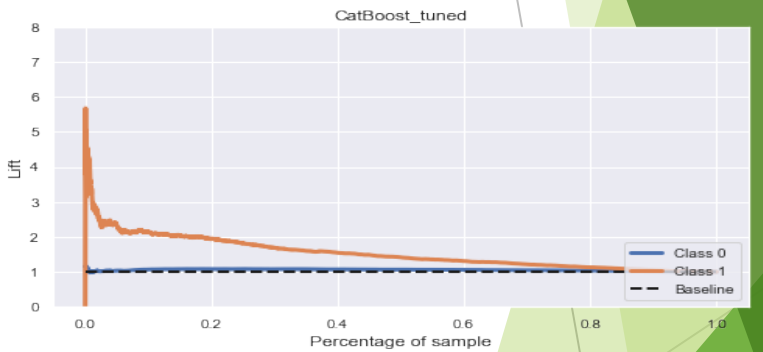
Experiment: CatBoost handles missing values and categorical variables

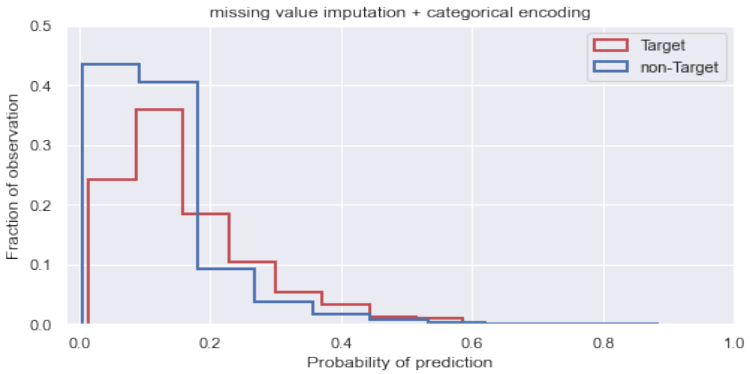
CatBoost	Precision	Recall	F1score	ROC_AUC	Target in top 20%	Target% in top 20%	Target in top 50%	Target% in top 50%
Data processed	0.244	0.014	0.026	0.630	261	35.9%	482	66.2%
Data Not processed (baseline model)	0.500	0.005	0.011	0.662	275	37.8%	514	70.6%
Hyperparameter tuned	0.571	0.005	0.011	0.664	283	38.9%	515	70.7%

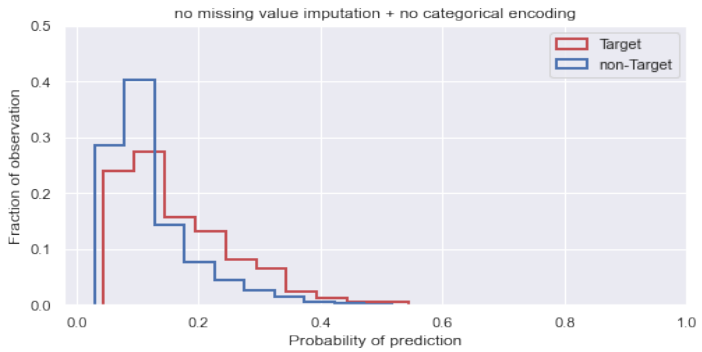
we applied hyperparameter tuning via random search Cross Validation (3 Folds) to the model and obtained our optimized model

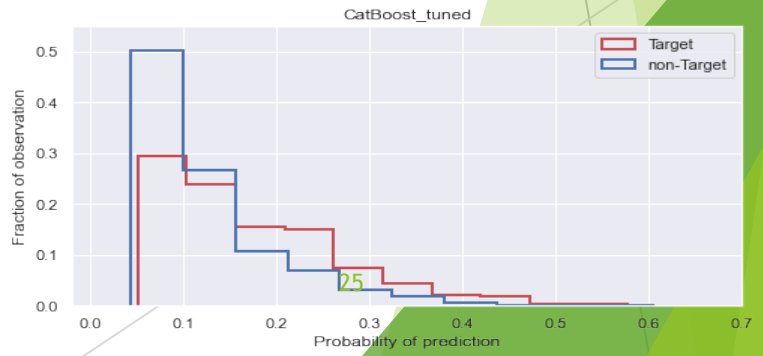






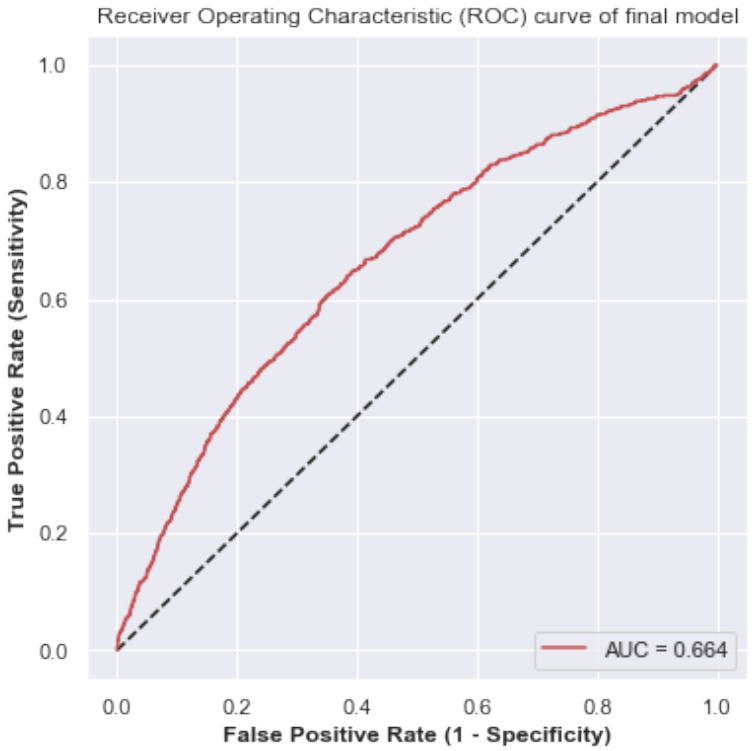


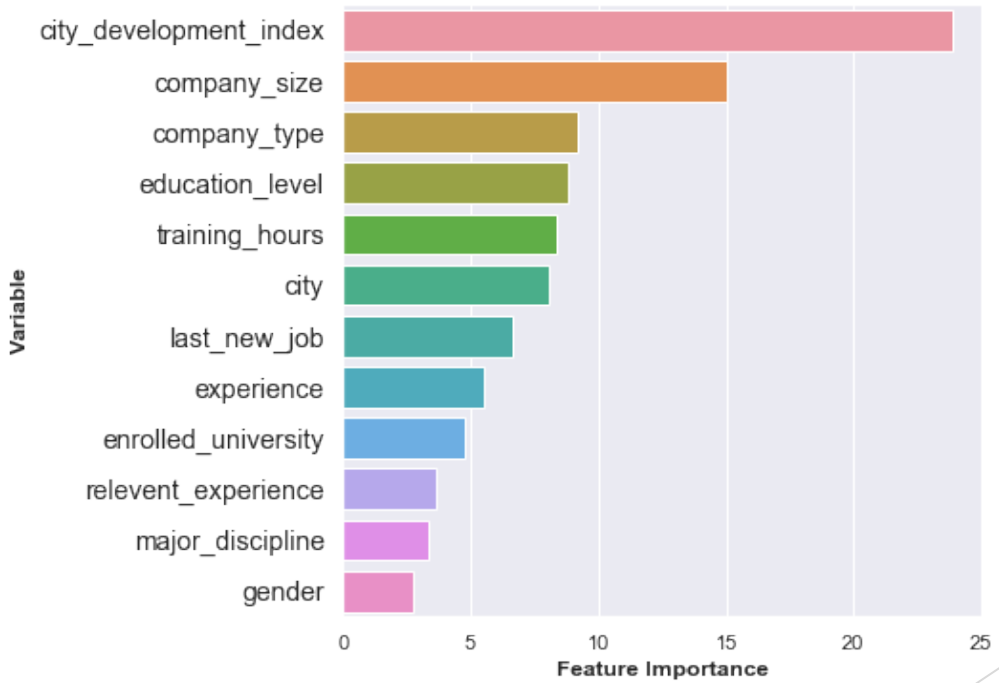




Final Model

model_final = CatBoostClassifier(iterations=988, depth=2, learning_rate=0.07165)





Summary and Recommendation

- Imbalanced data and feature overlap problem posed difficulties in model training. Resulted in low Recall and low Precision cross all models.
- When Recall and Precision are not ideal, we can use customized metrics like Targets captured in top 50% to achieve the KPI Target.
- If we use random picking to sample the same number of enrollees as the approach list (2754) , we could get a list that contains about 363~365 Target enrollees. Our best model, tuned CatBoost outperformed this result by 1.4 times.
- To solve the high degree of feature overlap between the two classes in this dataset, a good solution is to add more features/variables in the dataset if possible. Features like age, income and job title/level information.

Appendix

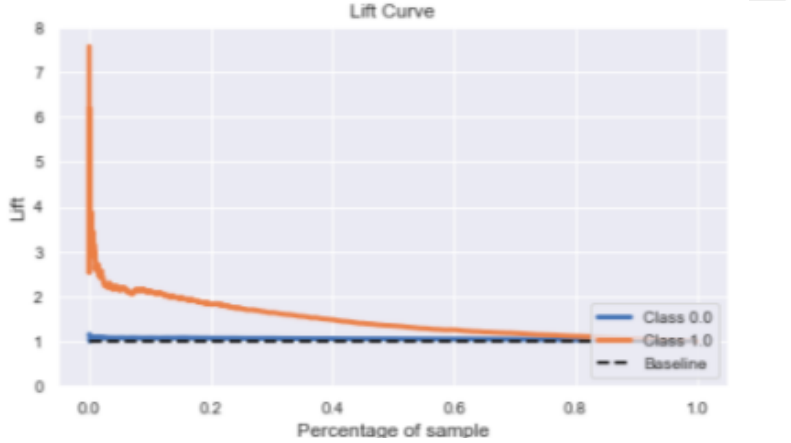
Model pros and cons

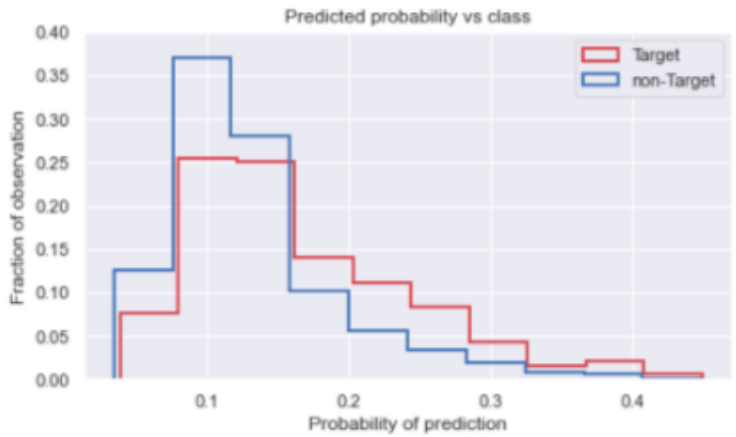
No.	Binary Classifier	Advantages	Disadvantages
1	Logistic Regression	• East to interpret • Small number of hyperparameters • Overfitting can be addressed though regularization	• May overfit when provided with large numbers of features • Can only learn linear hypothesis functions • Input data might need scaling • May not handle irrelevant features well
2	Gaussian Naïve Bayes	• No training involved • Very little parameter tuning is required • Features do not need scaling	• Assumes that the features are independent, which is rarely true
3	k-Nearest Neighbors	• No training involved, easy to implement • Only one hyperparameter	• Need to find optimal number of K • Slow to predict • Outlier sensitivity
4	Support Vector Machine	• Accuracy • Works well on smaller cleaner datasets • Is effective when umber of dimensions is greater than the number of samples.	• Isn't suited to larger datasets as the training time can be high • Less effective on noisier datasets with overlapping classes
5	Decision Tree	• Easy to interpret • Work with numerical and categorical features. • Requires little data preprocessing • Performs well on large datasets • Doesn't require normalization	• Overfitting • Unable to predict continuous values • Doesn't work well with lots of features and complex large dataset
6	Random Forest (Ensemble Bagging)	• Excellent predictive power • Requires little data preprocessing • Doesn't require normalization • Suitable for large dataset • Plenty of optimization options	• Overfitting risk • Parameter complexity • Limited with regression
7	Gradient Boosting (Ensemble Boosting)	• Excellent predictive accuracy • Can optimize on different loss functions • Provides several hyperparameter tuning options that make the function fit very flexible.	• Overfitting risk • Computationally expensive • Parameter complexity
8	XGBoost (Ensemble Boosting)	• Regularization to prevent overfitting • Computational efficiency and often better model performance • Can handle missing value	• Difficult to interpret and visualize • Parameter complexity • Time consuming when dataset is large
9	LightGBM (Ensemble Boosting)	• High speed, high accuracy • Can handle missing value and categorical value • Low memory usage • Compatibility with large dataset	• Parameter complexity • Overfitting risk
10	CatBoost (Ensemble Boosting)	• Can handle missing value and categorical value • Work well with both small and large dataset • Can monitor loss function	• Prevent overfitting • Normally doesn't need to tune hyperparameters to gain better result

Logistic Regression: Training vs Testing

Training

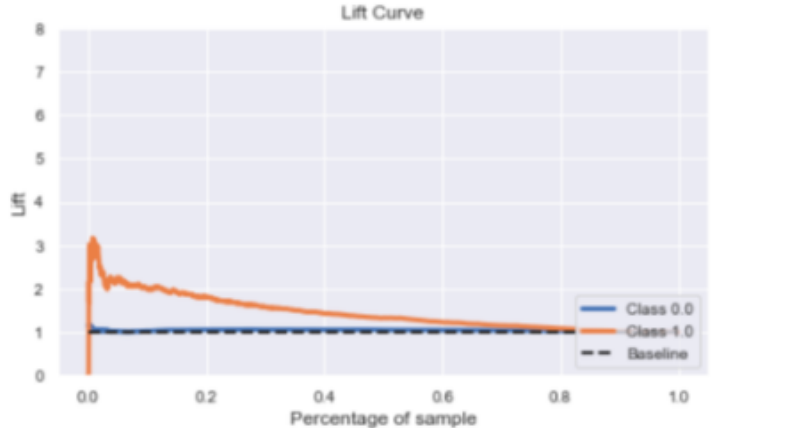
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
LogisticRegression	0.0	0.0	0.0	0.644	620.0	1140.0

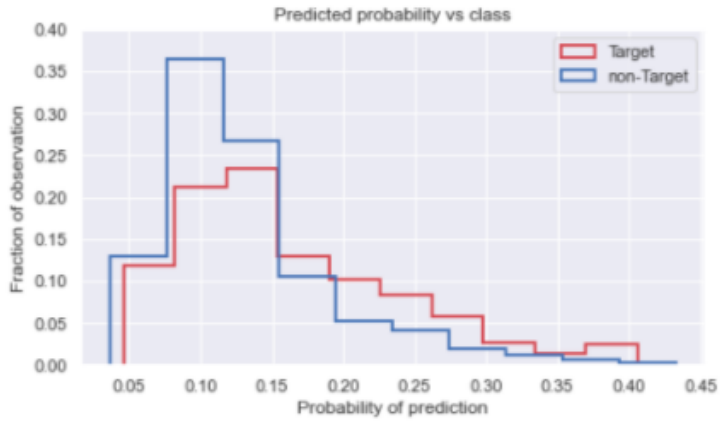




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
LogisticRegression	0.0	0.0	0.0	0.629	262.0	483.0



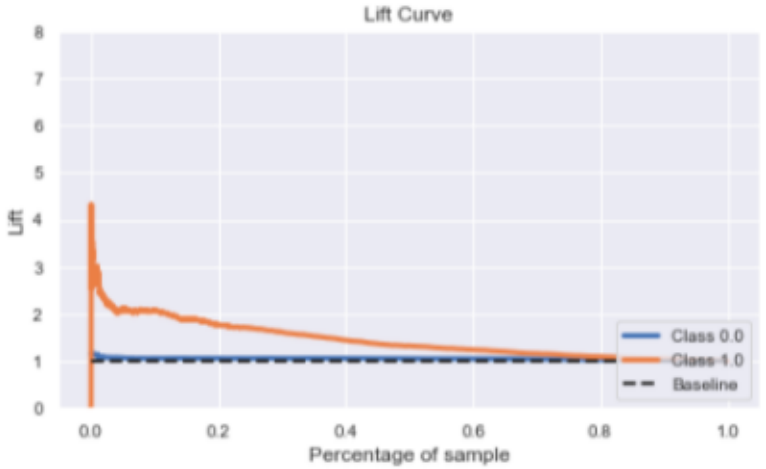


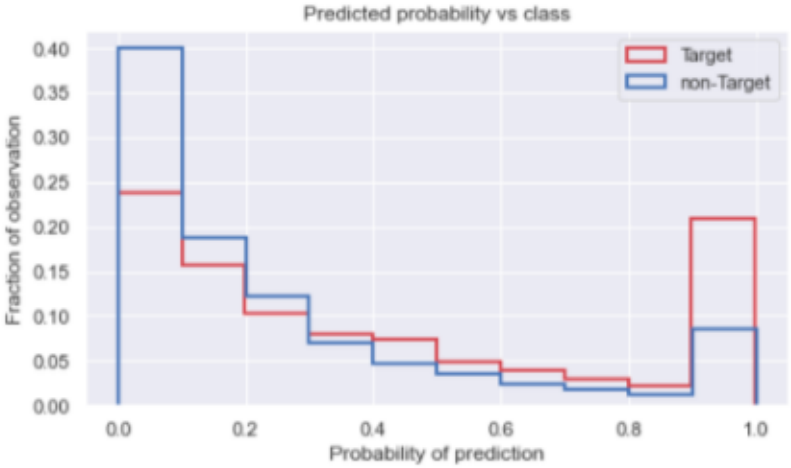
$$cumulative\ lift\ (0.2) = \frac{\%\ of\ positive\ classes\ in\ all\ deciles\ up\ to\ dec}{\%\ of\ positive\ classes\ in\ testing\ set} = \frac{262/1102}{0.132} \approx 1.80$$

Gaussian Naïve Bayes : Training vs Testing

Training

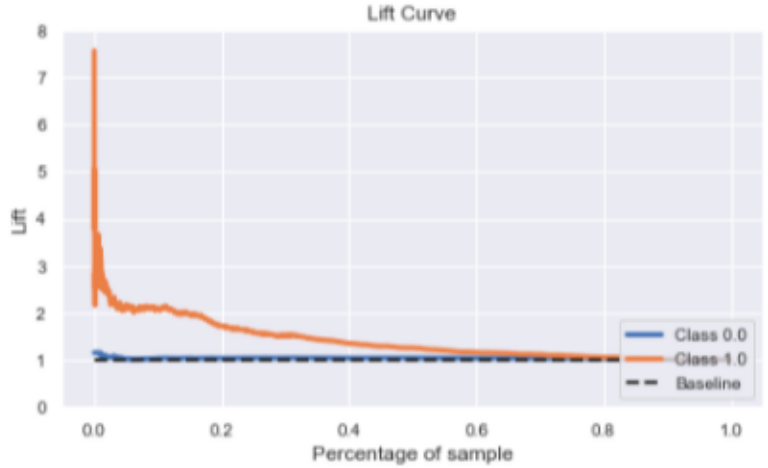
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
GaussianNB	0.234	0.346	0.279	0.633	599.0	1120.0

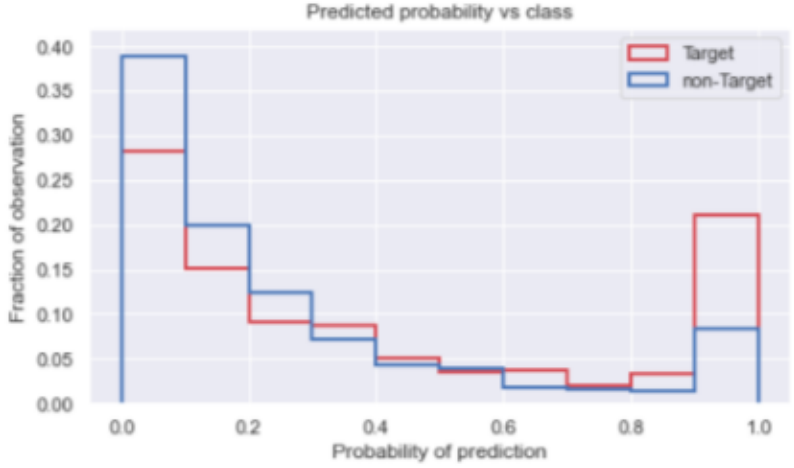




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
GaussianNB	0.23	0.337	0.274	0.608	251.0	458.0

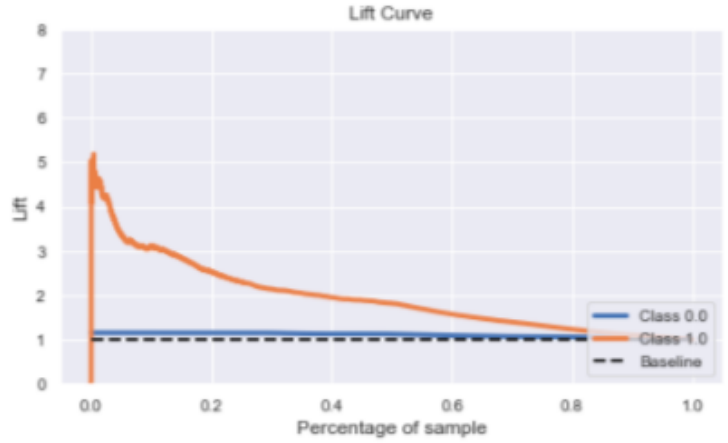


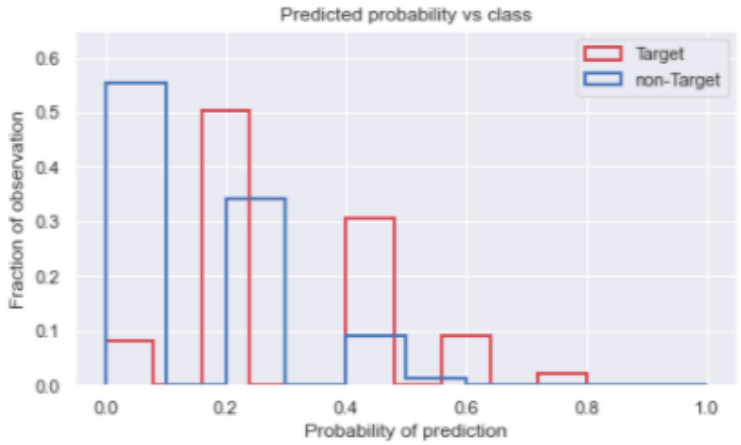


k-Nearest Neighbors : Training vs Testing

Training

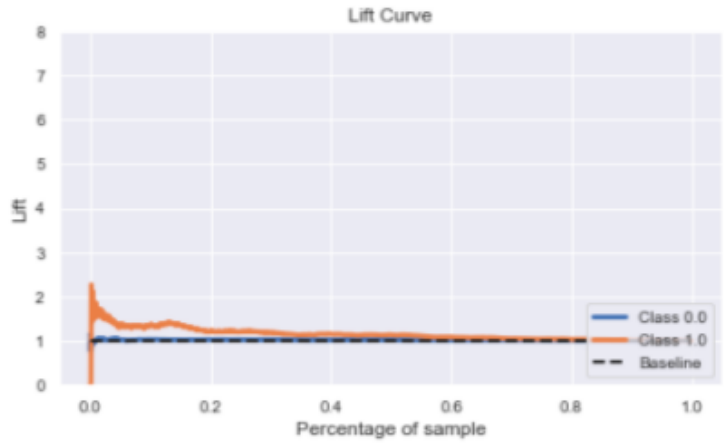
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
KNN	0.561	0.111	0.185	0.784	834.0	1545.0

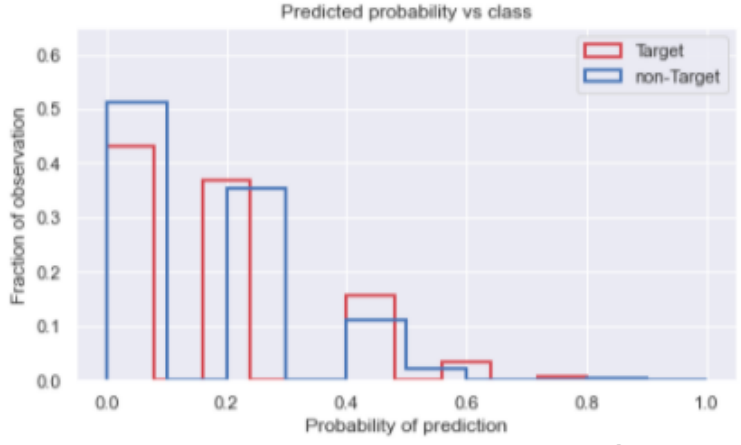




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
KNN	0.21	0.041	0.069	0.55	189.0	413.0

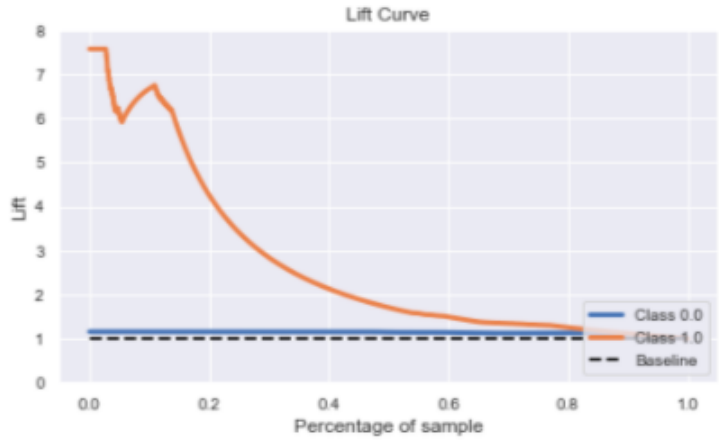


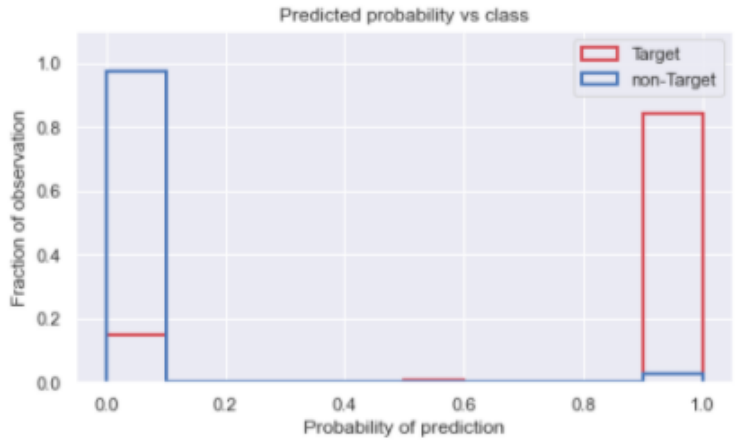


Decision Tree: Training vs Testing

Training

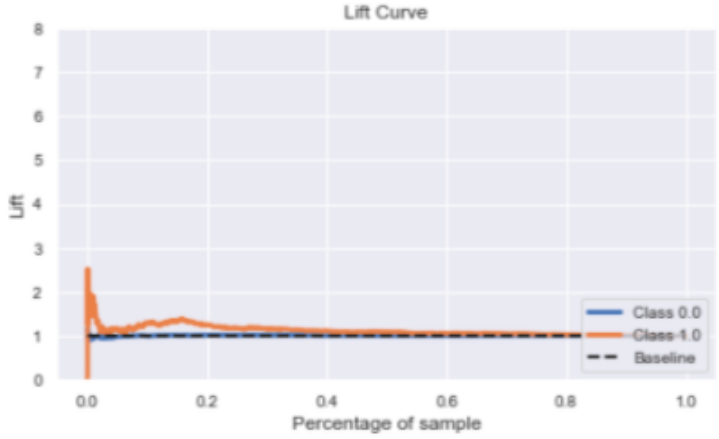
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
DT	0.819	0.843	0.831	0.911	1468.0	1683.0

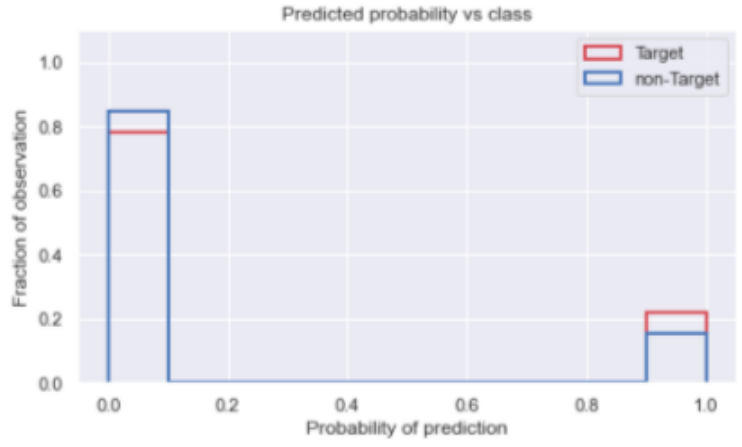




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
DT	0.179	0.22	0.197	0.533	186.0	391.0

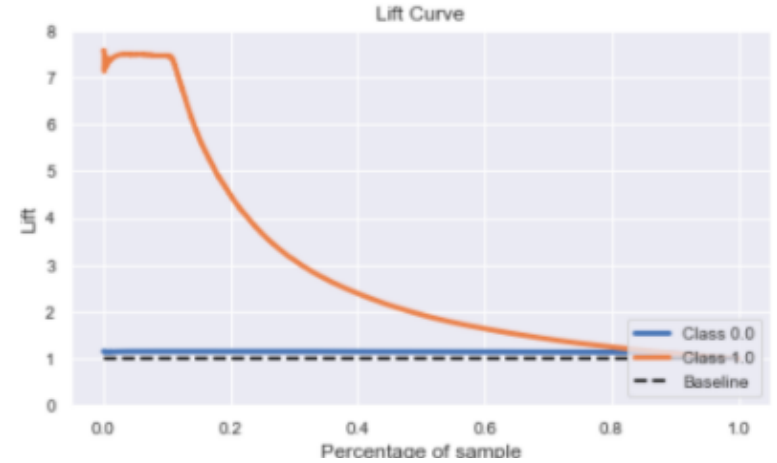


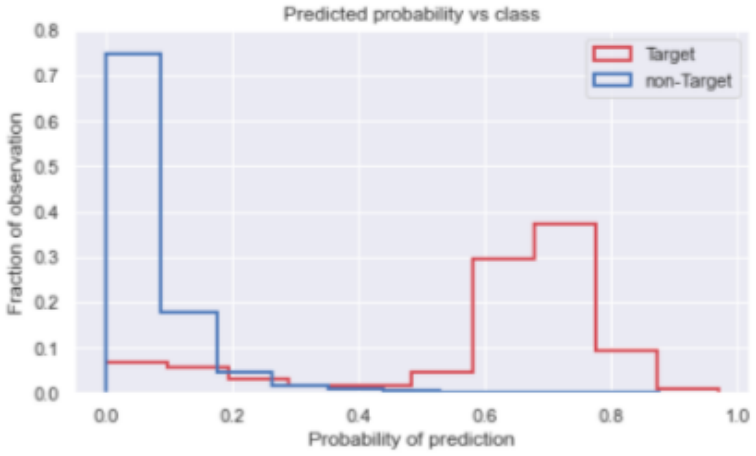


Random Forest: Training vs Testing

Training

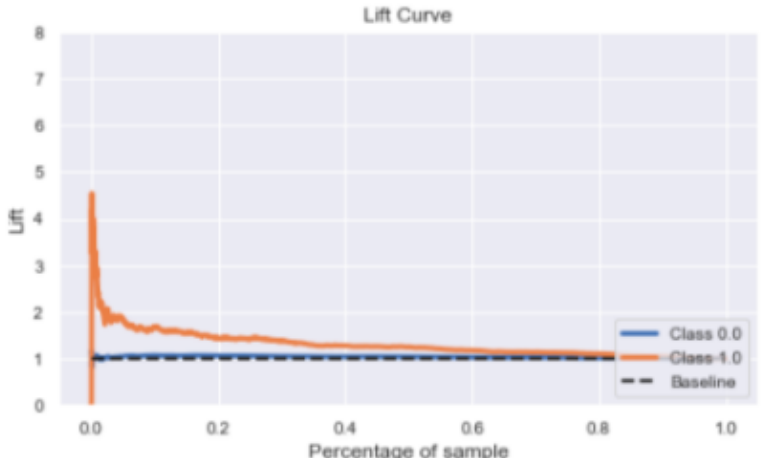
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
RF	0.968	0.81	0.882	0.96	1522.0	1646.0

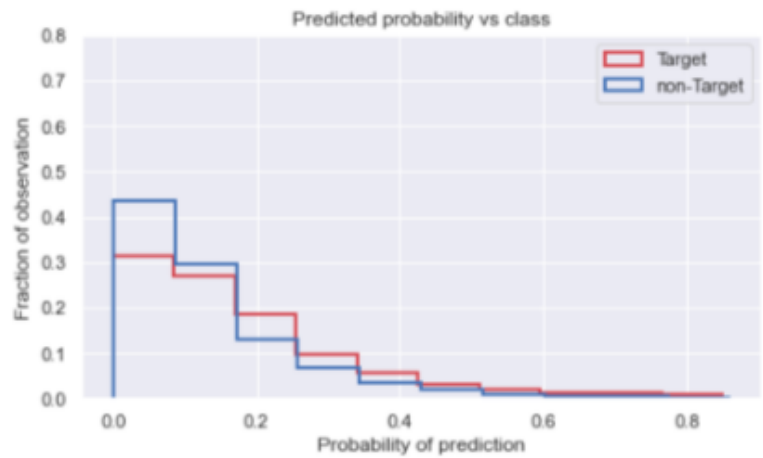




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
RF	0.27	0.051	0.086	0.594	211.0	456.0

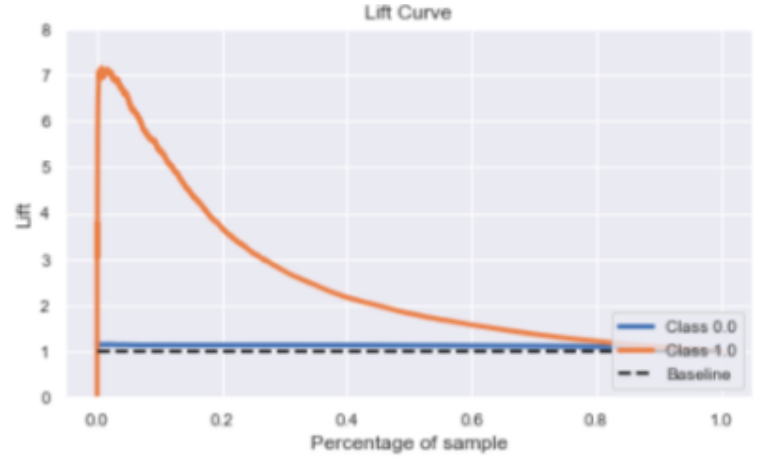


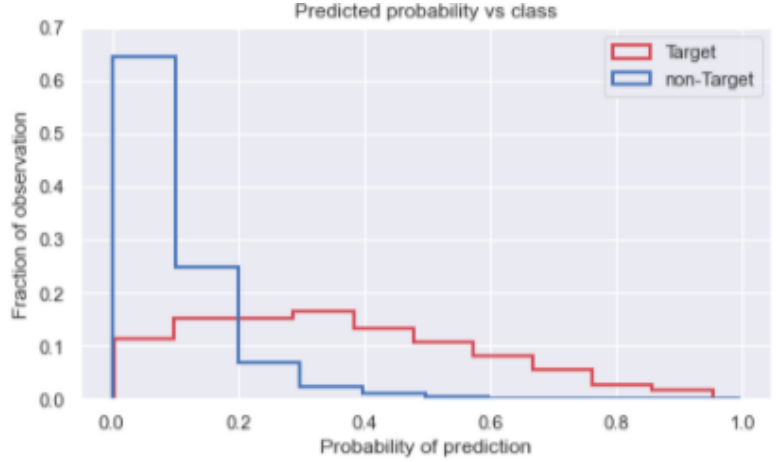


XGBoost: Training vs Testing

Training

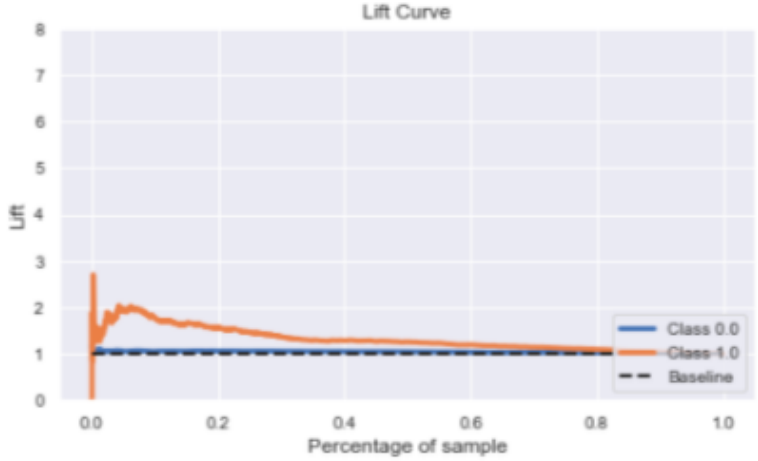
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
XGB	0.886	0.265	0.408	0.881	1247.0	1546.0

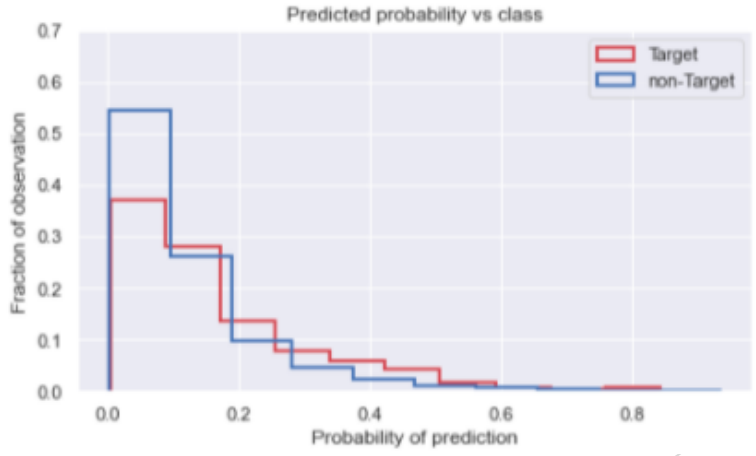




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
XGB	0.22	0.037	0.063	0.599	225.0	456.0

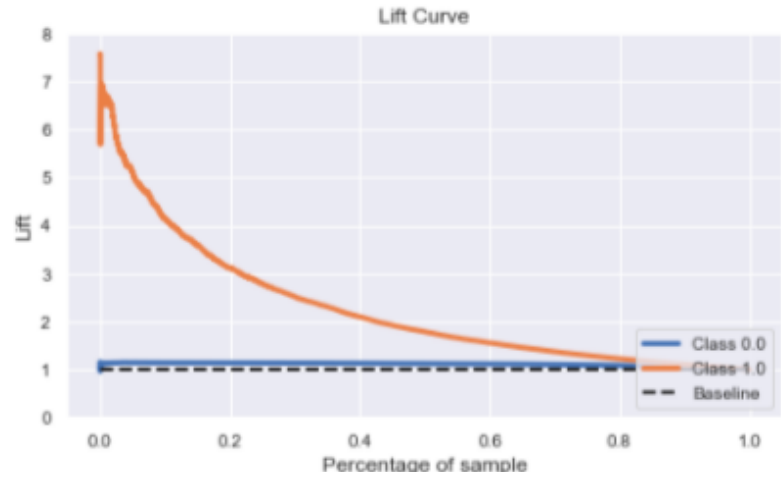


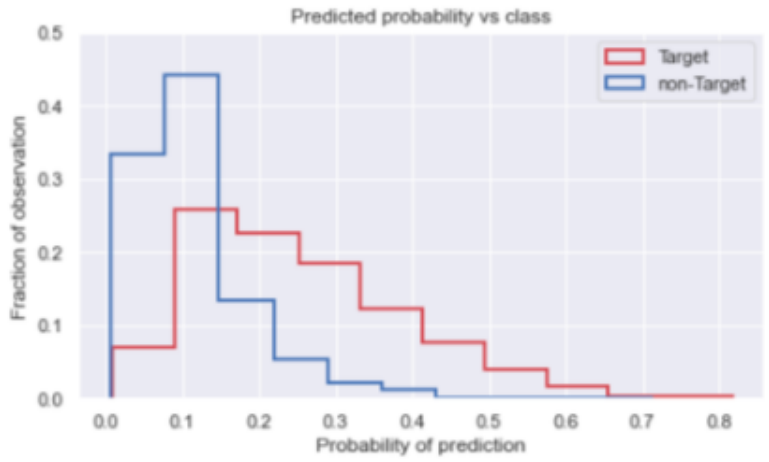


LightGB: Training vs Testing

Training

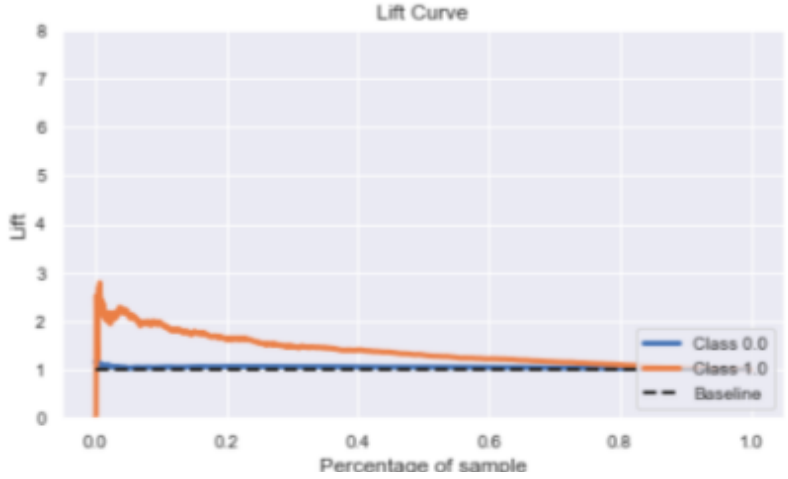
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
LightGB	0.86	0.058	0.108	0.838	1057.0	1513.0

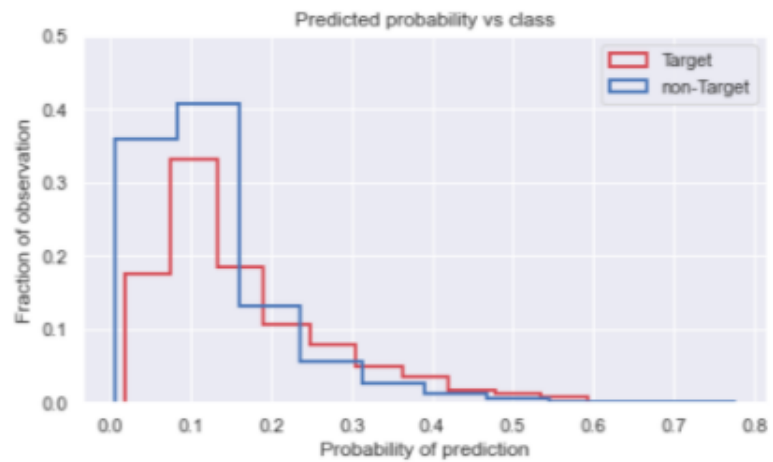




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
LightGB	0.333	0.016	0.031	0.618	237.0	473.0

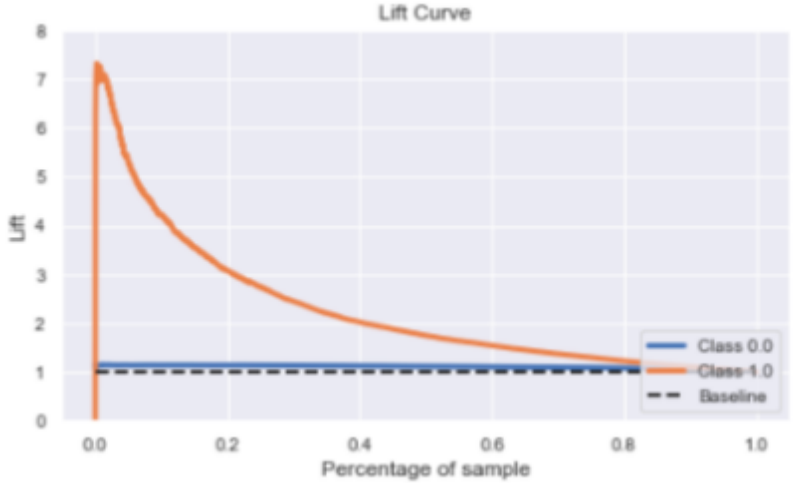


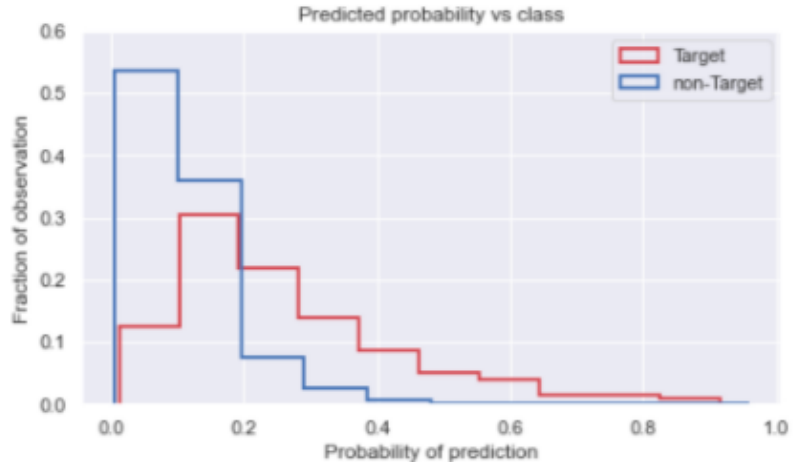


CatBoost: Training vs Testing

Training

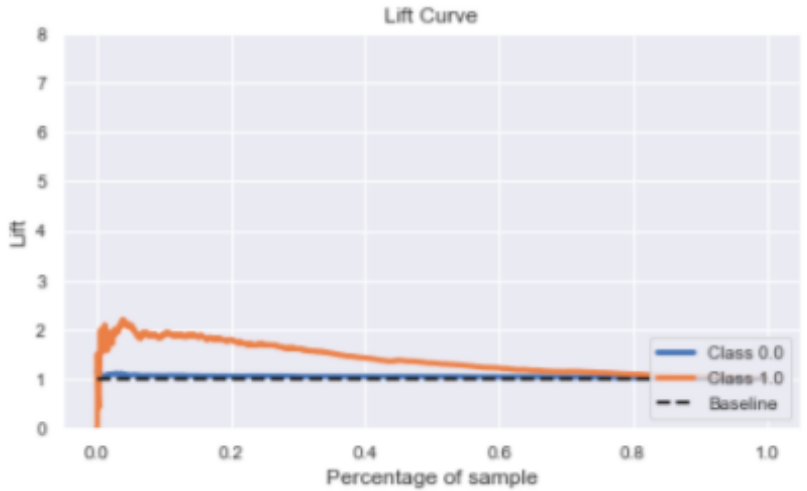
	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
CatB	0.921	0.11	0.196	0.829	1044.0	1480.0

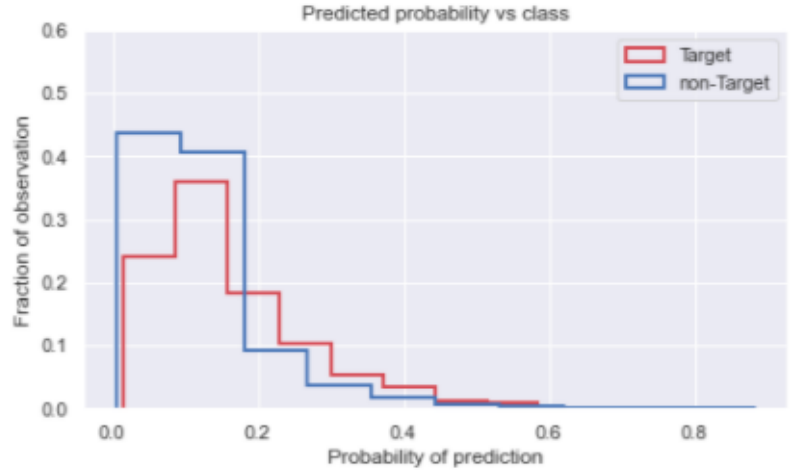




Testing

	precision	recall	f1score	roc_auc	Target_in_top20pct	Target_in_top50pct
CatB	0.244	0.014	0.026	0.63	261.0	482.0





Hyperparameter Tuning for CatBoost

1) Parameter Range:

```
parameters = {'depth' : sp_randInt(1, 5),
              'learning_rate' : sp_randFloat(),
              'iterations' : sp_randInt(800, 1100)}
```

2) Tuned result:

```
parameters = {'depth’ : 2,
              'learning_rate’ : 0.072
              'iterations’ :  988}
```

Link for dataset

- https://www.kaggle.com/aswathrao/hr-analysis?select=train.csv