

CS583A: Course Project Avin sharma, Himanshu Kumar May 19, 2019

## Summary

---

We participated in an active with late submission competition of Santander Customer Transaction Prediction where the goal was to identify which customers will make a specific transaction in the future, irrespective of the amount of money transacted. The data provided for this competition has the same structure as the real data they have available to solve this problem. The data given was composed of 200,000 rows and 200 anonymized features, there is no intuition or explanation given for the data provided. The final model we choose is **Light GBM**. Light GBM is a gradient boosting framework that uses tree based learning algorithm. We implement the model using lightGBM package and run the code on a MacBook Pro with one Intel i7 CPU and 32 GB memory. Performance is evaluated on the classification accuracy, kaggle score and ranking. In the public and private leaderboard, our score is 0.9227123; We are in top 10%. Though the result on the public leaderboard is not available due to late submission glitch. But we can prove it through a screenshot attached in the jupyter notebook.

## Problem Description

---

### Problem

In this challenge, we should help this bank identify which customers will make a specific transaction in the future, irrespective of the amount of money transacted. The data provided for this competition has the same structure as the real data we have available to solve this problem.

### Data

The data provided for this competition has the same structure as the real data they have available to solve this problem. The data given was composed of 200,000 rows and 200 anonymized features, there is no intuition or explanation given for the data provided.

### Challenges

The data is highly unbalanced, There are 10.049% target values with 1 and rest are 0.

## Solution

---

### Model

The model we finally choose is the Light GBM which is a gradient boosting framework that uses tree based learning algorithm. Light GBM grows tree vertically while other algorithm grows trees horizontally meaning that Light GBM grows tree leaf-wise while other algorithm grows level-wise. It will choose the leaf with max delta loss to grow. When growing the same leaf, Leaf-wise algorithm can reduce more loss than a level-wise algorithm. Light GBM is prefixed as 'Light' because of its high speed. Light GBM can handle the large size of data and takes lower memory to run. It is advised to use LGBM on large datasets with 10,000+ rows. Since, Light GBM is sensitive to overfitting and can easily overfit small data.

## Implementation

We implemented the LGBMClassifier and it took 30 mins to train the model. Settings. The loss function is binary cross-entropy. Challenge was to get the hyperparameters such as learning rate, n\_estimators, num\_leaves, reg\_alpha, reg\_lambda. We didn't use the grid-search but by studying these parameters and changing it a little bit from their default values, we were able to get the good results.

## Compared Methods

---

**Simple Neural Network1** : We used the 2 Dense layer with activation as sigmoid and loss as binary crossentropy. The training and validation accuracies are respective 90.68% and 90.55%. Though the model is very modest but there is a twist in data preparation, which helped us in getting good accuracy with such a simple model. We standardized each variable with mean 0 and variance 1.





































**Simple Neural Network2** : Since the data is highly unbalanced, we thought of applying K-stratified cross validation, with above Neural network. The training and validation accuracies are respective 91.50% and 91.40%. We can see there is an improvement from the previous model. So our hypothesis of using K -stratified was correct.

## Outcome

---

In the public and private leaderboard, our score is 0.9227123; We are in top 10%. Though the result on the public leaderboard is not available due to late submission glitch. But we can prove it through a screenshot attached in the jupyter notebook.

Submission and Description	Private Score	Public Score
<b>Deep Learning Project (version 7/7)</b> an hour ago by <a href="#">Avin Sharma</a> From "Deep Learning Project" Script	<b>0.92085</b>	<b>0.92248</b>

48	▼ 4	Konstantin Yakovlev		0.92180	48	1mo
49	▲ 8	D&G	    	0.92175	153	1mo
50	▼ 4	Chris Deotte		0.92170	54	1mo
51	▼ 3	Going up!		0.92156	109	1mo
52	▲ 1	[ods.ai] elite_random_b	    	0.92147	72	1mo
53	▼ 4	[ods.ai] Vladimir Larin		0.92127	43	1mo
54	—	WispZero		0.92121	59	1mo
55	▼ 3	Adana Big Bang	   	0.92118	93	1mo
56	▲ 14	god & dog	 	0.92111	31	1mo
57	▼ 2	blending doesnt work	 	0.92104	48	1mo
58	▼ 2	magic is all you need		0.92103	59	1mo
59	▲ 5	Just Me		0.92096	68	1mo
60	▲ 7	shiyi001	  	0.92077	4	1mo
61	—	fakeplastictrees		0.92058	44	1mo
62	▼ 4	[ods.ai] Atanas Atanasov		0.92052	29	1mo
63	▼ 4	Yura Trubitsyn		0.92048	15	1mo
64	▲ 8	Upupup	 	0.92048	43	1mo
65	▲ 1	[ods.ai] valilenk		0.92047	9	1mo
66	▲ 3	Dmitriy Ukrainskiy		0.92040	10	1mo
67	▲ 4	Li-Der		0.92035	91	1mo