Final Report

Pankaj Patil

February 28, 2021

Abstract

The aim of the competition was to predict a target variable by using given structured data. Evaluation metric for the problem statement was AUC Score. 40% of the test dataset was available for public leaderboard while rest of the 60% data was private. A total of around 10k participants participated in the competition. My final public leaderboard rank as **60**

1 Introduction

1.1 Problem Statement

Your Client FinMan is a financial services company that provides various financial services like loan, investment funds, insurance etc. to its customers. FinMan wishes to cross-sell health insurance to the existing customers who may or may not hold insurance policies with the company. The company recommend health insurance to it's customers based on their profile once these customers land on the website. Customers might browse the recommended health insurance policy and consequently fill up a form to apply. When these customers fill-up the form, their Response towards the policy is considered positive and they are classified as a lead.

Once these leads are acquired, the sales advisors approach them to convert and thus the company can sell proposed health insurance to these leads in a more efficient manner.

Now the company needs your help in building a model to predict whether the person will be interested in their proposed Health plan/policy given the information about:

- Demographics (city, age, region etc.)
- Information regarding holding policies of the customer
- Recommended Policy Information

1.2 Data Dictionary

Variable	Definition
ID	Unique Identifier for a row
City_Code	Code for the City of the customers
Region_Code	Code for the Region of the customers
Accomodation_Type	Customer Owns or Rents the house
Reco_Insurance_Type	Joint or Individual type for the recommended insurance
Upper_Age	Maximum age of the customer
Lower_Age	Minimum age of the customer
Is_Spouse	If the customers are married to each other (in case of joint insurance)
Health_Indicator	Encoded values for health of the customer
Holding_Policy_Duration	Duration (in years) of holding policy (a policy that customer has already subscribed to with the company)
Holding_Policy_Type	Type of holding policy
Reco_Policy_Cat	Encoded value for recommended health insurance
Reco_Policy_Premium	Annual Premium (INR) for the recommended health insurance
Response (Target)	0 : Customer did not show interest in the recommended policy, 1 : Customer showed interest in the recommended policy

1.3 Data Exploration

First, I read the train and test data and combined it. This is done so that all the data processing I do will change both train and test data in a similar way. First, I looked at an overview of the data. I'm not mentioning the unique categories in the categorical column but that can be checked in the code file that is attached along with this document.

	Column	dtype
0	ID	int64
1	City_Code	object
2	Region_Code	int64
3	Accomodation_Type	object
4	Reco_Insurance_Type	object
5	Upper_Age	int64
6	Lower_Age	int64
7	Is_Spouse	object
8	Health_Indicator	object
9	Holding_Policy_Duration	object
10	Holding_Policy_Type	float64
11	Reco_Policy_Cat	int64
12	Reco_Policy_Premium	float64
13	Response	float64

The competition explicitly states that we cannot use ID column for prediction (which rules out using possible data leakage to improvise score). The Region_Code column shows up as an integer but we know that it is a categorical variable thus we'll transform that into object type. Same follows for Holding_Policy_Type column. Overall, we have 8 categorical columns and 3 numerical columns (excluding ID column).

Next, I checked the data for missing values.

Column	Count of missing values
ID	0
City_Code	0
Region_Code	0
Accomodation_Type	0
Reco_Insurance_Type	0
Upper_Age	0
Lower_Age	0
Is_Spouse	0
Health_Indicator	16718
Holding_Policy_Duration	28854
Holding_Policy_Type	28854
Reco_Policy_Cat	0
Reco_Policy_Premium	0
Response	21805
dtype: int64	

The missing values in Reponse variable actually correspond to the missing values in test data, so they're fine. Thus we are left with 3 columns with missing data. First, I look at the **Health_Policy_Duration**. My working hypothesis is

that missing values in this column will correspond to the person not having any existing insurance policy. This turned out to be true as all the missing values in **Health_Policy_Duration** are also missing **Health_Policy_Type**. Also, since only **39%** of the data was missing in these columns, it is better if we impute them and not drop them. I imputed both the columns with **-999**. This acts as a new category. For the column **Health_Indicator**, I created a new category "X0" and imputed missing values for this column. I checked whether the missing values in this column were also same as those missing in the other two columns but that was not the case. An assumption one might make is that this is a result of that particular person not choosing to disclose medical history to FinMan but that is just an assumption.

Next, I took a closer look at the numerical column **Holding_Policy_Duration**. The unique values in this column

```
print('Unique values in Holding_Policy_Duration column : {}'.format(df.
    Holding_Policy_Duration.unique()))
>>> Unique values in Holding_Policy_Duration column : ['14+' nan '1.0' '3.0' '5.0'
    '9.0' '14.0' '7.0' '2.0' '11.0' '10.0' '8.0'
 '6.0' '4.0' '13.0' '12.0']
```

So, not only we have to impute the missing values, we also have a set of entries corresponding to **14+**. I'll replace it with 15. Also, only 23% of the values in this column were missing, so it's safe to go ahead with imputation. Next, I move on to Data Visualization.

```
print('Missing value perc in Health_Indicator column : {}'.format(df.Health_Indicator
    .isna().sum()/df.shape[0]))
>>>Missing value perc in Health_Indicator column : 0.22999986242381718
```

2 Data Visualization

First, I checked the data for class imbalance. As shown in Figure 1, The target variable is imbalanced. Approximately 76% of the leads end up in no response. This provides a baseline metric for modelling. I dealt with class imbalance by using **Stratified K Fold cross validation** method.

Next I checked the correlation of the numerical features with the target variable. As shown in Figure 2, the correlation between target and numerical variables is very low. This might imply a low predictive power or could be an example of **Simpson's Paradox**. A solution to that would be to use binning of the numerical variables and see if that provides a better estimate.

Moving on, Now I draw a pairplot of numerical variables. As seen in Figure 5. There is not a nice linearly separable boundary for any numerical variable which can identify the target variable with greater accuracy. Another few points to note :
1. The Upper age is always greater than equal to Lower age (expected :))
2. The Age Distribution has peaks on lower and upper end. The Lower and Upper Ages are different only in case of a joint application.
3. Holding_Policy_Duration also has a peak on the lower end.
4. THere are outliers in the Premium column. While columns like Premium and Income are expected to have outliers, a good way to tackle them is by using log.

The absence of a 'nice' boundary was expected from the corr relation plot

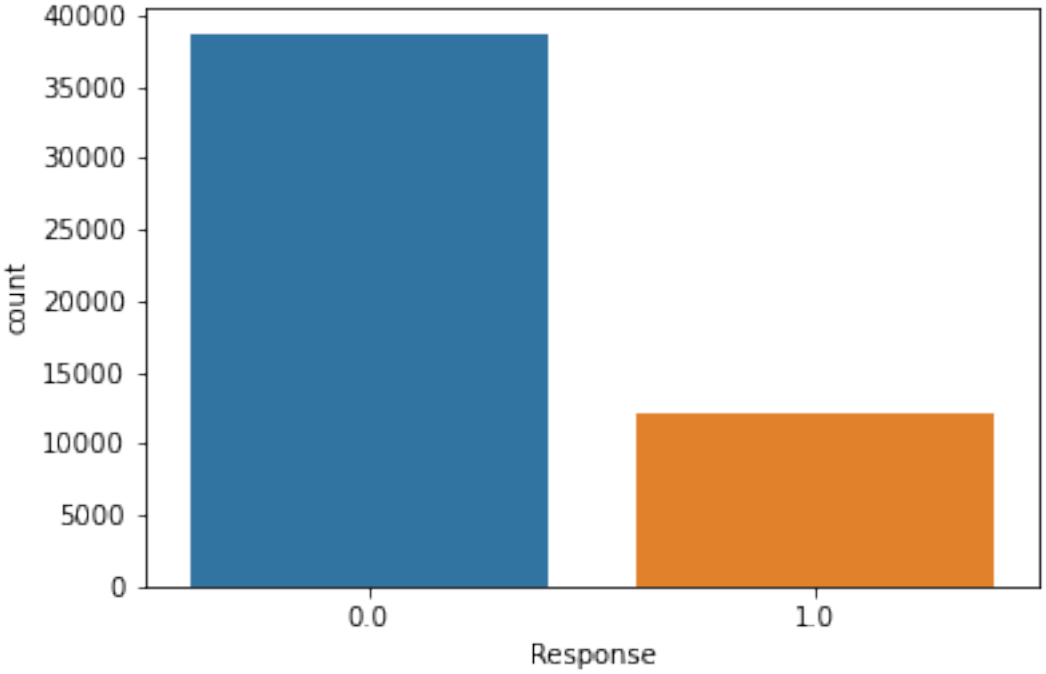

Figure 1: Target variable distribution

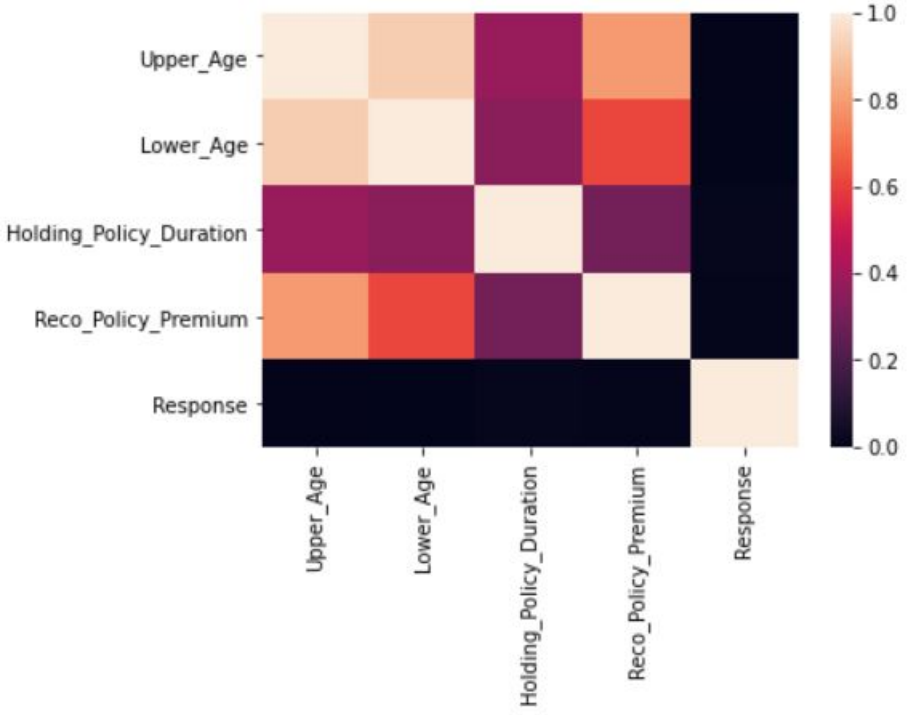

Figure 2: Correlation Plot

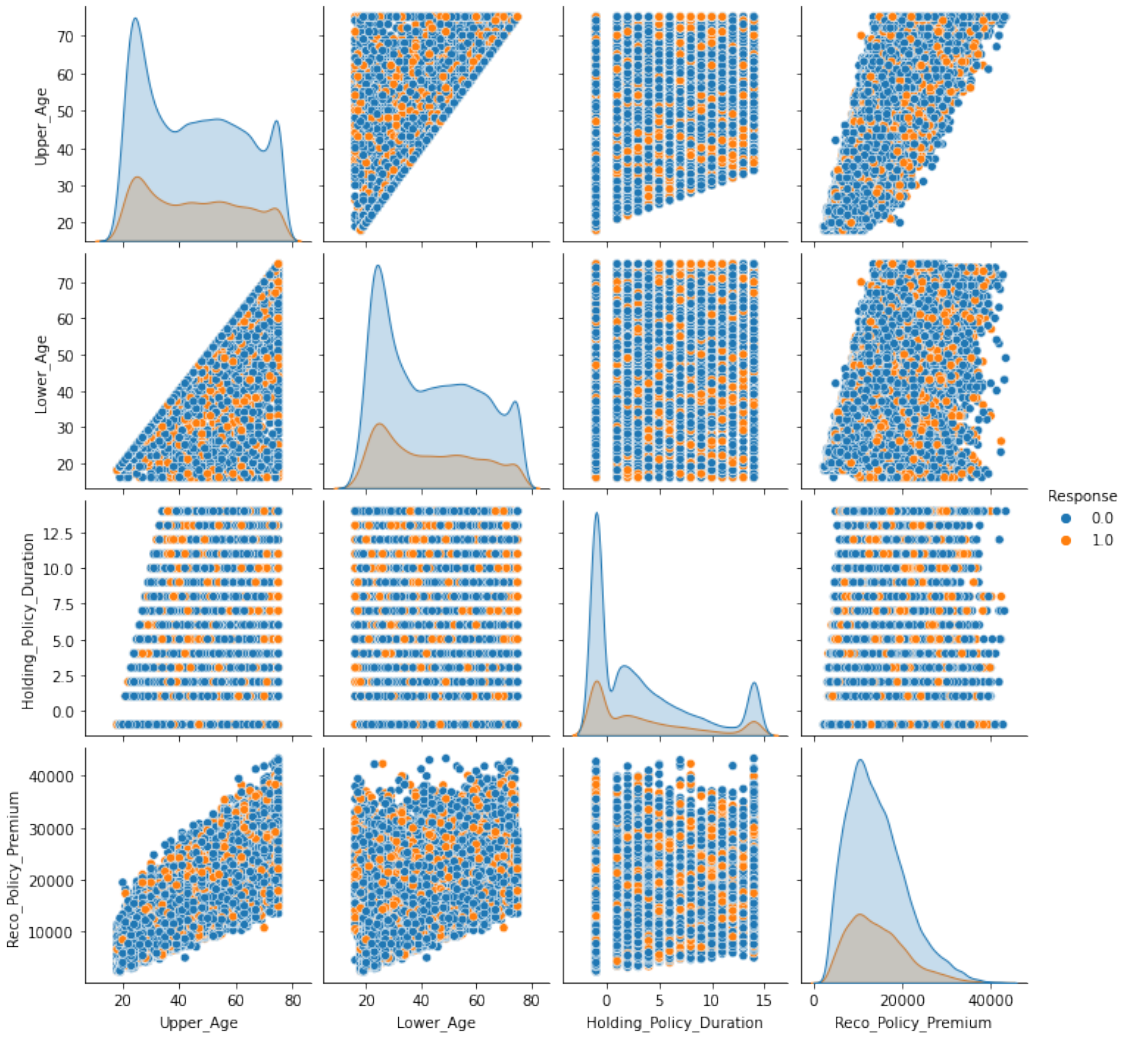

Figure 3: Pairplots of various numerical variables

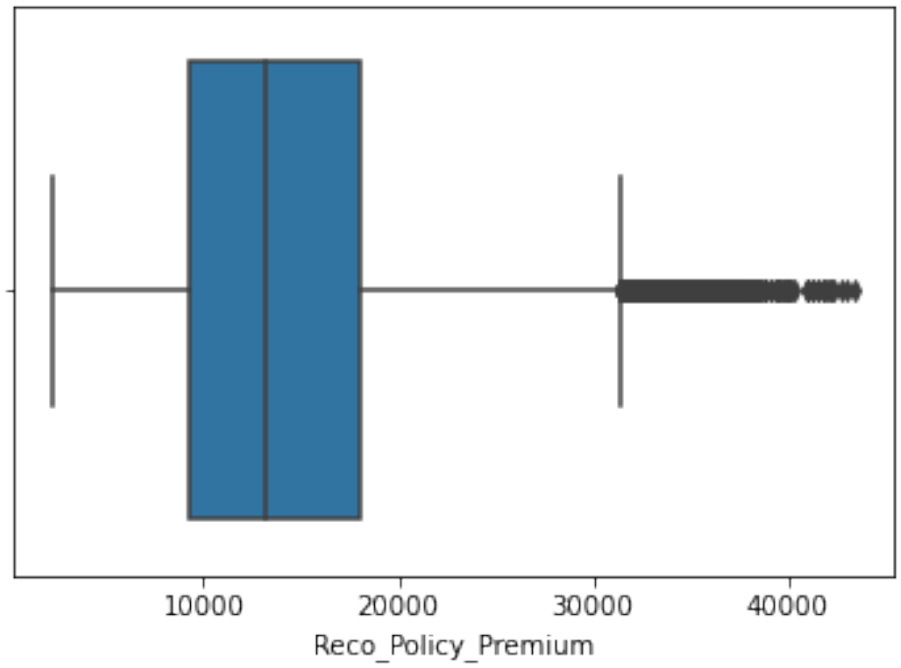

Figure 4: BoxPlot of Policy Duration feature

Next we do a similar exercise with categorical variables but here we divide them on the basis of target variable.

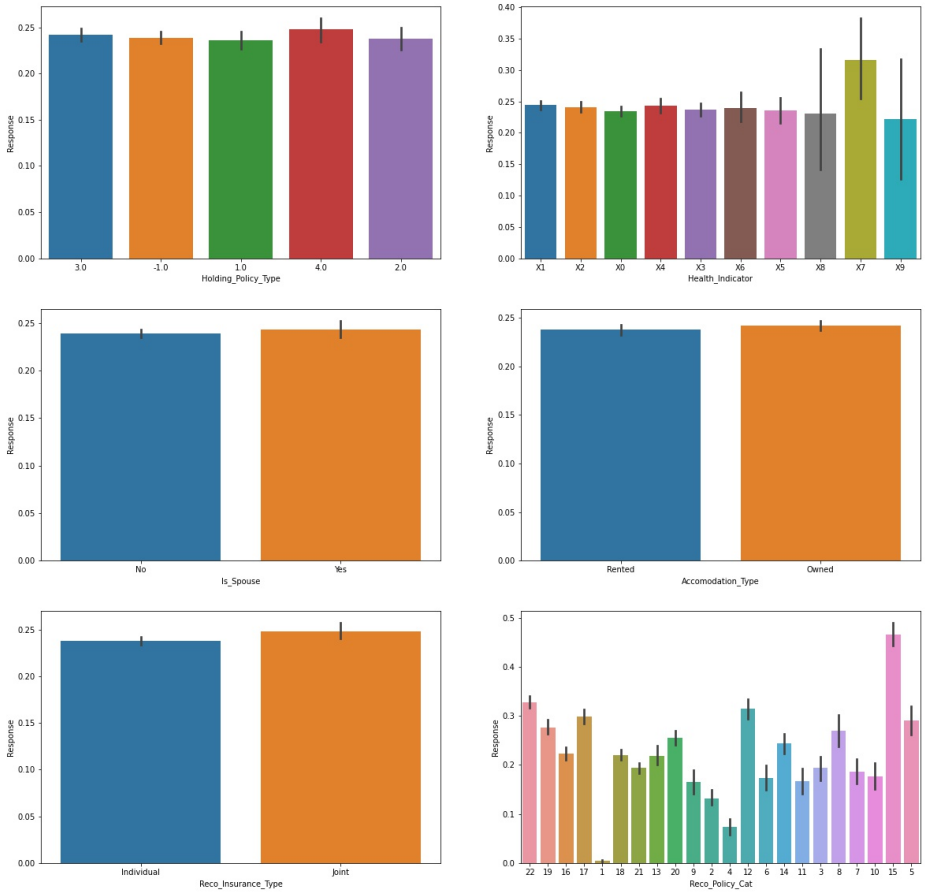

Figure 5: Pairplots of various numerical variables

This shows some interesting insights :

1. Health_Indicator category X7 has highest success rate
2. Reco_Policy_Cat category 15 has highest success rate
3. Reco_Category_Cat category 1,4 have highest failure rate
4. Variables like Is_Spouse, Accommodation_Type and Reco_Insurance_Type are not very useful in predicting response variable

3 Modelling

For this section, I'm mentioning below what didn't work and then I'll just mention what worked. The detailed analysis of complete set of combinations tried is beyond the scope of this document.

What didn't work :

- Frequency Encoding of Categorical variables
- Clustering of Categorical variables
- Creating new categorical variables from existing ones
- Models like XGBoost, RandomForest
- LabelEncoding + OneHotEncoding

In this particular data set, all the feature engineering and transformations i did yielded around 0.68 AUC in cross validation and around 0.69 score on public leader board. In fact, the model which gave me highest AUC was the baseline model built using catboost. I didn't do any transformations in it except creation of 2 new numerical variable **Premium_Per_Age** and Average_Age. While their contribution in model improvement was not massive, it still performed slightly better than the baseline model. The training vs test loss for one of the folds in Stratified K fold cross validation is shown below.

Next I performed grid search for catboost hyperparameters using 10 fold cross validation and used the finetuned model for final output.

Figure 6: Grid Search Results AUC vs iterations

Based on the grid search, Final parameters I used was depth = 10, learning rate = 0.01, l2 regression coeff = 1

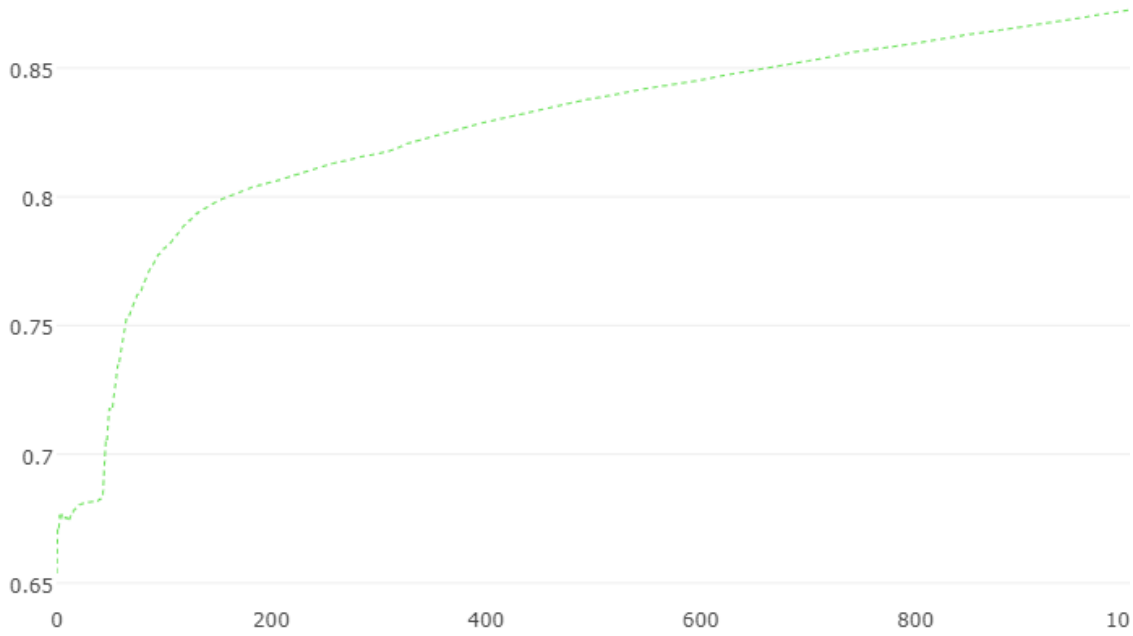

Figure 7: Final Model Results AUC vs iterations

While there is a lot to be explored and addressed in the dataset, for given time constraint, I believe this approach has yielded good results. As Data Scientists, our struggle is always to balance time against the level of detail analysis. This is the one which we all have to master.

4 Results

Public Leaderboard Rank : 60*

*- subject to movement as more submissions are made

Public Leaderboard AUC : 0.8117978635