Name: Nisrin Dhoondia
Email: nisrin.dhoondia@gmail.com
LinkedIn: https://www.linkedin.com/in/nisrin-dhoondia-javadeveloper/

Health Insurance Lead Prediction

1) A brief on the approach, which you have used to solve the problem.

This is a binary classification problem. To do the prediction I have used supervised machine learning models and also Deep learning models using Tensorflow and Keras.

I had used many models but the evaluation metric roc_auc_score was just hanging between 0.50 and less than 0.54 on Analytics Vidhya though I did my best with
hyperparameter tuning,
best feature selection using XGBClassifier and SelectFromModel,
finding best feature through correlation matrix and by mean of all independent variables with groupby on dependent variable,
without using SMOTE and also using SMOTE for balancing the dependent variable binary values,
using class_weight arguments in the model for balancing the dependent variable binary values records,
with MinMaxScaler normalization
dropping the null values rows and also filling the null values with most frequent and mode.
But the roc_auc_score is between 0.50 and less than 0.54.
These are the models I have used to train the dataset and get prediction results:

Model	remarks
LogisticRegression	Used many times 1) with different max_iter values 2) with selected feature found with mean 3) with different C-Statistic. Cannot submit all C values csv file because of 10 submissions per day.
DecisionTreeClassifier	with GridSearchCV
RandomForestClassifier	with GridSearchCV
XGBClassifier	1) with GridSearchCV 2) with GridSearchCV and feature selection using SelectFromModel
ExtraTreesClassifier	with arguments value
Sequential	Deep Learning ANN model using Tensorflow and Keras 1) With different optimizer adam and RMSprop 2) With different epochs
CNN	Conv2D a Deep Learning CNN model using

	Tensorflow and Keras I am still working on it. Hope to finish before submission ends. As I had only used it in image dataset and this is first time trying with numeric dataset.
LGBMClassifier	with arguments value
BaggingClassifier	With GridSearchCV and RepeatedStratifiedKFold But it was taking too long so had to go without using RepeatedStratifiedKFold and with very much reduce the parameter values too.

The dataset has no duplicates record but null values. Initially I dropped these rows and also without using SMOTE to balance dataset didn't gave good roc_auc_score. So I filled the null values in Health_Indicator and Holding_Policy_Duration columns using SimpleImputer from sklearn with most frequent strategy and also for Holding_Policy_Type using mode.
I used mode in Holding_Policy_Type because people go with policy which is most popular and same goes for Holding_Policy_Duration. For Health_Indicator I used most frequent strategy too as most of the policy holder are at same health level mostly.

I had dropped ID column as it wasn't important independent variable for prediction. Since I had dropped from the original dataset first I created a copy so that I can use it for prediction file.

While converting object dtype to numeric dtype first I checked whether the unique values are same in train and test dataset. As while converting object dtype to numeric dtype if both the train and test dataset doesn't have same unique values it will create a discrepancy in these columns values which may affect the prediction results of the model. Though these dataset has same unique values in each of the object dtype columns in both train and test, then though I had combined both train and test to perform the conversion of object dtype to numeric dtype using astype('category').cat.codes as this doesn't create extra columns like get_dummies. Later again separated train and test dataset.

In feature engineering, by creating a new column Mean_Age having the mean of the values in each of the Upper_Age and Lower_Age columns datapoints (row-wise) that is mean of upper age and lower age at row 0 index for 0 index row datapoint in Mean_Age column and so on. And then dropping the Upper_Age and Lower_Age columns.

Used Synthetic Minority Oversampling Technique (SMOTE) for balancing the dataset dependent variable. Here actually we have to first split the train dataset into train and test datasets as we want to ensure that our model generalizes well to unseen data by avoiding having exact same observations to be present in both the test and train datasets after SMOTE but here in this competition I am using the entire train dataset to train the models so I am not splitting into train and test datasets.

For normalizing the dataset data I mostly use MinMaxScaler as it normalize the data between 0 and 1. And it doesn't disturb the shape of the distribution of the original values.

Here in this competition I am using the entire dataset to train the models and not dividing it into train and test datasets. As mostly in competition everybody does that and are getting good scores. I have yet not got good score, maybe all the models I have used are getting overfitted and maybe if I had did train_test_split or a StratifiedShuffleSplit from sklearn.model_selection may have reduced the data and all the model may have get trained to generalize

2) What data-preprocessing / feature engineering ideas really worked? How did you discover them?

Answer same as above in question 1

3) What does your final model look like? How did you reach it?

I had used many models but the evaluation metric roc_auc_score was just hanging between 0.50 and less than 0.54 on Analytics Vidhya submission platform though I did my best. Till now only LogisticRegression model with max_iter 1000 and 5000 both giving me same roc_auc_score 0.537203001847805, which is my highest score on Analytics Vidhya submission platform.
I am submitting my Notebook with LogisticRegression model with max_iter 1000 then.