DS7333 Quantifying the World: Case Study 7

Rick Fontenot, Feby Thomas Cheruvathoor

1. **Introduction**

­

The goal for this case study is to build a classification model for an anonymized dataset to minimize the client’s money lost due to incorrect predictions.

The client has shared that the loss associated with incorrectly predicting a target value of 1 when the true value is zero causes them to lose $100. Incorrectly predicting a value of 0 when the true value is 1 causes them to lose $250. For correct predictions, there is no money lost.

The course professor provided guidance that reducing the money lost per observation to the range of $5 is a reasonable goal to meet, but also challenged to push beyond that goal if possible.

Due to client privacy concerns, this dataset is anonymized therefore the lack of domain knowledge may hinder understanding of features and new feature creation. Model interpretability is not a goal of this study, only the performance and minimizing monetary loss.

1. **Methods**

**Data description**

This anonymized dataset includes 160,000 records with one target variable and 50 features that can be used for predictions.

The target variable is binary with the following sample sizes:

Class-0: 95,803 records

Class-1: 64,197 records

There are 50 features labeled “x0” through “x49”. There are 45 numeric features and 5 features containing strings. While some insight can be gained as to the type of information in the string features, there is no information as to exactly what the features represent.

**Processing data & Feature Creation**

Missing Values:

All features have at least 1 missing value, while the feature with the most missing has 41 or 0.026% missing values. Due to the relatively small number of missing values and lack of domain knowledge on what the features represent, missing values were dropped rather than imputed. After removing rows with missing values, there are 158,392 records remaining (1,608 records dropped)

Duplicates:

There were no duplicate observations in the dataset that required removal

Class Imbalance:

With a 60%/40% ratio, the original data is relatively balanced. There are sufficient number of samples in each class, so no resampling or class weights were applied in this analysis.

Data Types and feature creation:

The 45 numeric features were treated as floats. The 5 features with strings were studied and transformed as follows:

x24:

This feature has 3 unique values: “euorpe”, “asia”, and “america” which appear to be regions of the globe associated with each transaction. The spelling of “euorpe” was corrected to “europe” and the feature was one-hot encoded and the first category “america” was dropped to eliminate multicollinearity issues between the features.

x29:

This feature includes various formats of text abbreviations for months of the year. Before deciding whether this feature should be one-hot-encoded, the values were converted to the number of the month within the year (1-12) for exploratory analysis. Examining the proportion of the target variable equal to 1 within each month highlighted a trend over the course of the year:

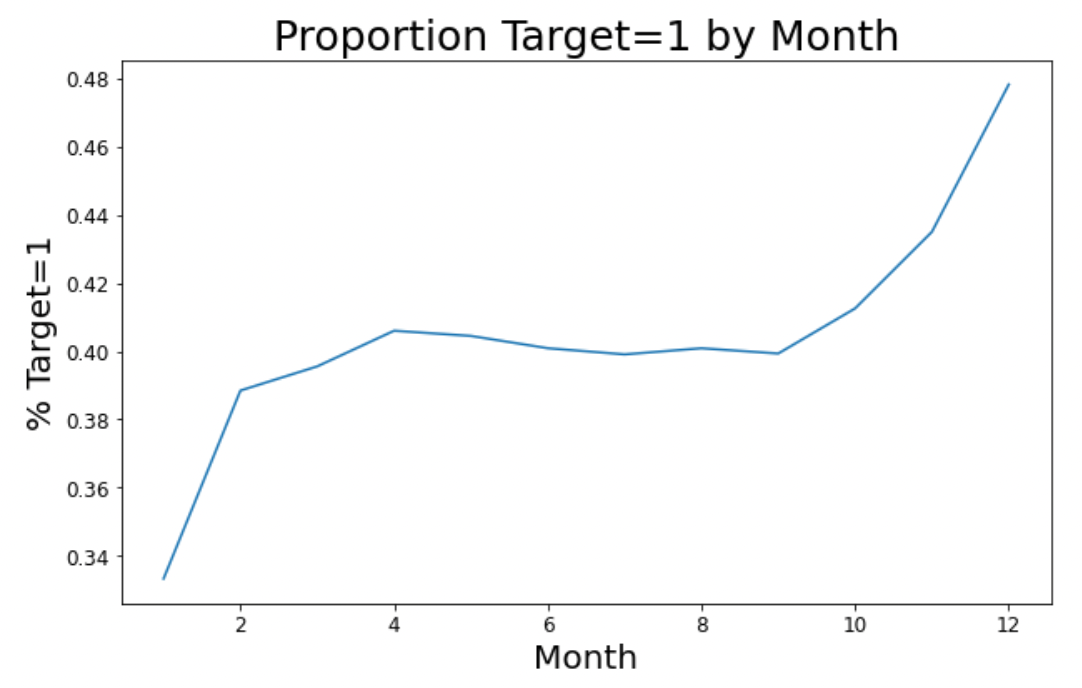


Figure 1: Target Class vs. Month # (feature x29)

Due to the relationship observed with the target variable. x29 was transformed into an ordinal variable of 1-12 based on the month of year rather than one-hot encoding.

x30:

This feature includes days of the week (mon-fri). Similar to x29 above, examining the proportion of the target variable equal to 1 highlighted a trend over the course of the week:

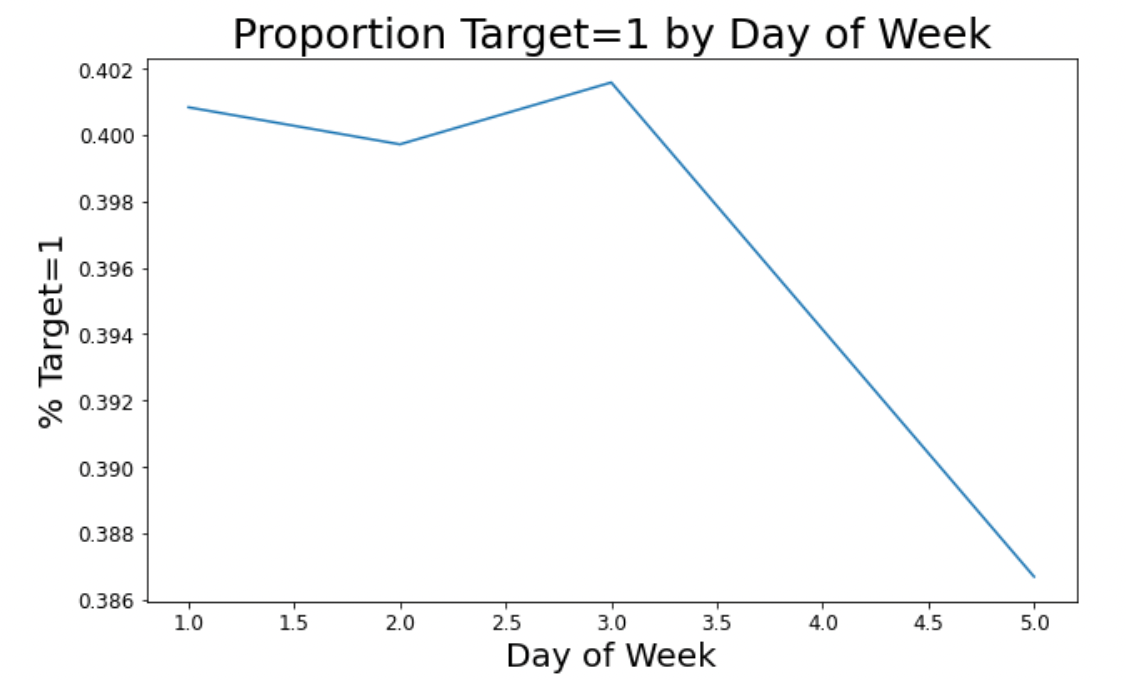


Figure 2: Target Class vs. Day of Week (feature x30)

Due to the relationship observed with the target variable. x30 was transformed into an ordinal variable of 1-12 based on the month of year rather than one-hot encoding.

x30:

This feature contains a string with numeric values followed by a “%”. Some of he numeric values are preceded by “-“ indicating a negative values and the percentages are small such as -0.02%. The “%” sign was removed, data type converted to float. The value was then multiplied by 100 to end up with values such as -2.0 prior to any future scaling or standardizing.

x37:

This feature contains numeric values preceded by a “$” indicating it represents currency. The “$” was removed and the data type converted to float.

Standardization and Scaling:

The features of the raw dataset are generally centered at a mean of zero or one but standard deviations vary and range as high as 1,000. All features including month and weekday were scaled using the sklearn StandardScaler tool to center mean at 0 and standard deviation to one prior to modeling.

**Training Split & Cross Validation Strategy**

In order to train, tune, and determine training stop points for models such as neural networks and XGBoost, 10% of the data was split into a validation set (stratified shuffled data to get random observations with similar proportion of records in the target classes).

An additional 10% of the dataset was split into a test set for an unbiased measure of the loss function after model tuning. The remaining 80% of the dataset is used for tuning. (both of these splits also utilized stratified shuffled data to get random observations with similar proportion of records in the target classes).

The remaining 80% of the dataset was used for training models.

A cross validation object was defined with 5 folds of stratified shuffle splits to utilize in the training of each model.

**Loss Function and scoring metric**

Based on the information provided by the client a custom function and scoring metric were created to evaluate models. The scoring metric was utilized in hyperparameter searches and early stopping criteria for models.

incorrect\_0 = 100

incorrect\_1 = 250

def calculate\_cost(y\_true, y\_pred):

cm = confusion\_matrix(y\_true, y\_pred)

losses = np.array([[0, incorrect\_0], [incorrect\_1, 0]])

cost = cm \* losses #Total cost varies by data size

return cost.sum()/cm.sum() #Returns Cost/Observation

scoring=make\_scorer(calculate\_cost, greater\_is\_better=False)

**Random Forest modeling (Model 1)**

Using all transformed and scaled features as described above, a random forest model was built using the sklearn RandomForestClassifier tool. The following random cross-validated grid search was performed with 5-folds for 20 models selected out of the possible 288 combinations:

'criterion':['gini','entropy'],

'max\_depth':[20,25,30,40],

'min\_samples\_split':[10,8,6,4],

'min\_samples\_leaf':[4,2,1],

'max\_features':['auto','sqrt','log2']

The final model identified by lowest money loss included the following parameters:

min\_samples\_split = 8,

min\_samples\_leaf = 1,

max\_features = 'sqrt',

max\_depth = 30,

criterion = 'entropy'

The final model was fit on training data and prediction probabilities were made on the validation set. A search for the optimal probability threshold was performed to minimize the money loss on the validation set.

Using the model fit to training data and the discriminant threshold identified by the validation set, predictions were made on the test set and the test money loss was recorded for comparison to other models.

**XGBoost modeling (Model 2)**

An XGBoost model was built using the sklearn XGBClassifier tool. The following random cross-validated grid search was performed with 5-folds for 40 models selected out of a large range of combinations:

'n\_estimators':[50, 100, 150, 200],

'max\_depth':[5,10,15,20,25,30],

'min\_child\_weight': [1, 5, 10],

'colsample\_bytree': [0.6, 0.8, 1.0],

'gamma': [0,0.1,0.2,0.4,0.8,1.6,3.2,6.4,12.8,25.6,51.2,102.4, 200],

'subsample': [0.6, 0.8, 1.0],

'learning\_rate': [0.01, 0.03, 0.06, 0.1, 0.15, 0.2, 0.25, 0.300000012, 0.4, 0.5, 0.6, 0.7],

'reg\_alpha': [0,0.1,0.2,0.4,0.8,1.6,3.2,6.4,12.8,25.6,51.2,102.4,200],

'reg\_lambda': [0,0.1,0.2,0.4,0.8,1.6,3.2,6.4,12.8,25.6,51.2,102.4,200]

'early\_stopping\_rounds': 5,

'eval\_metric': 'logloss',

'eval\_set': [[validation\_features, validation\_labels]]

The final model identified by lowest money loss included the following parameters:

use\_label\_encoder=False,

subsample = 0.6,

reg\_lambda = 12.8,

reg\_alpha = 3.2,

n\_estimators = 150,

min\_child\_weight = 5,

max\_depth = 25,

learning\_rate = 0.15,

gamma = 0.8,

colsample\_bytree = 1.0

The early stopping criteria for the final model included:

eval\_set=[[validation\_features, validation\_labels]],

eval\_metric='logloss',

early\_stopping\_rounds=5

The final model was fit on training data and prediction probabilities were made on the validation set. A search for the optimal probability threshold was performed to minimize the money loss on the validation set.

Using the model fit to training data and the discriminant threshold identified by the validation set, predictions were made on the test set and the test money loss was recorded for comparison to other models.

**Neural Network modeling (Model 3)**

A neural network model was built using the Tensorflow tools. The model structure included 5 sequential Dense layers with each followed by a dropout layer to address potential overfitting. Using the HParam tool, a full search of all parameter combinations were performed in the following ranges:

HP\_NUM\_UNITS = hp.HParam('num\_units', hp.Discrete([100, 300]))

HP\_DROPOUT = hp.HParam('dropout', hp.Discrete([0.0, 0.1, 0.2])) #[0.0, 0.2]

HP\_OPTIMIZER = hp.HParam('optimizer', hp.Discrete(['adam'])) #['adam', 'sgd']

HP\_ACTIVATION = hp.HParam('activation', hp.Discrete(['gelu', 'selu', 'swish', 'relu', 'tanh']))

The final model identified by lowest money loss included the following parameters:

num\_units = 300

dropout = 0.1

optimizer = adam

activation = selu

The final model includes a batch size of 50 with a maximum 100 epochs and early stopping criteria as validation set loss with a patience of 3 epochs. Weights from the best epoch prior to stopping were restored.

The final model was fit on training data and prediction probabilities were made on the validation set. A search for the optimal probability threshold was performed to minimize the money loss on the validation set.

Using the model fit to training data and the discriminant threshold identified by the validation set, predictions were made on the test set and the test money loss was recorded for comparison to other models.

**Ensembling**

Using the final models for each of the 3 types created, the training fits were used to predict probabilities for the validation set so that predictions were out-of-fold. Predictions on test set probabilities were also created for each the training fit models.

Validation probabilities for each of the 3 models were fed as features into ensemble models, then the weighted predictions produced were evaluated for optimal discriminant threshold based on the money loss function.

After creating the validation ensemble fit with optimal validation discriminant thresholds, ensemble predictions were made on the test set and the money loss was recorded for comparison to other models.

Linear Regression, Logistic Regression, and Neural Network were all evaluated as the ensemble model type for comparison. The regression models used default parameters (no regularization performed).

The Neural Network for ensembling included the relu activation and adam optimizer with 5 dense layers containing 300 neurons, each followed by a dropout layer with 50% dropout to minimize risk of overfitting. Without another unbiased set of data to use for early stopping, the training termination was based on the training loss.

1. **Results**

A summary of the training time and performance for each of the individual models and the three ensemble models is below:



Figure 3: Comparison of model performances

The neural network performed as the best individual model for money lost as well as the type of model to use for ensembling. Below is a comparison of the confusion matrices for these two best models showing how the slight improvement in misclassifications significantly improves the money lost per transaction.

Neural Network (individual model) Ensemble (NN model)

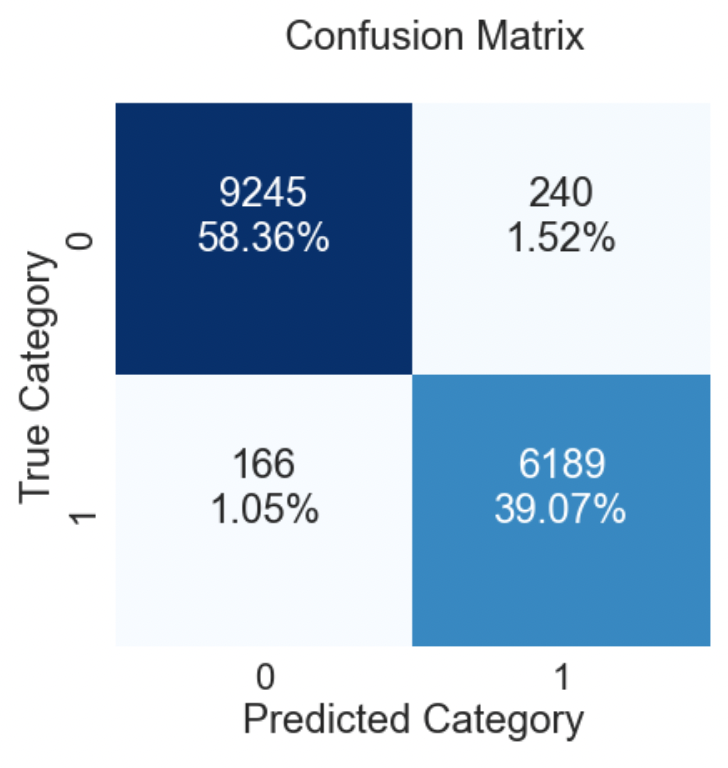
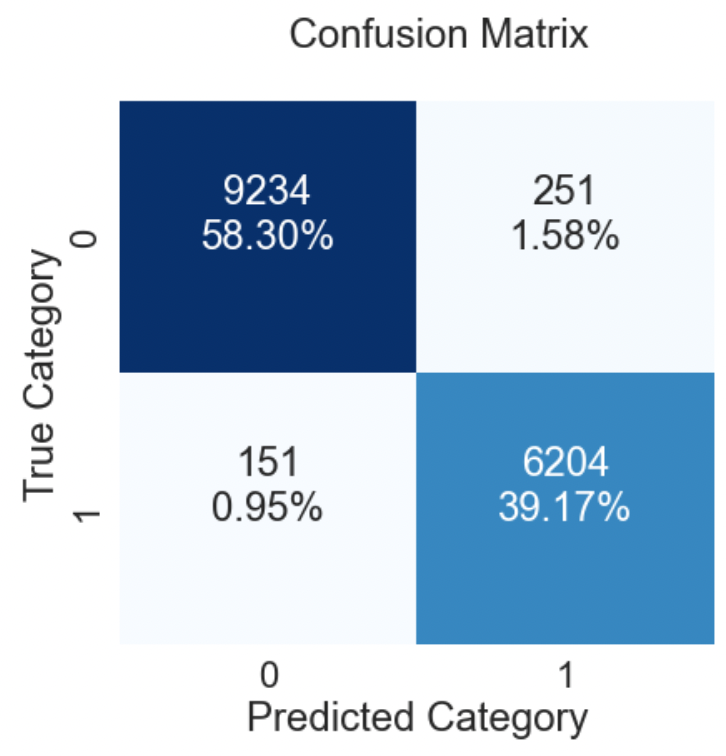
 

Figure 4: Confusion Model comparison for best models

1. **Conclusions**

Random Forest and XGBoost are powerful non-linear models, but even tuned versions of these models did not meet the goal of getting the monetary loss to $5.00. The individual neural network model took longer to tune and train but the prediction results for monetary loss were significantly better and beat the goal.

While the individual Random Forest and XGBoost models did not perform as well, they still provided value as inputs to the ensemble model which further reduced the monetary loss beyond the individual Neural Net model.

The best ensemble model **reduced the money lost per transaction to $3.97** which is well below the goal for this study.

Just as evaluating multiple types of individual models is important in the base analysis, exploring multiple types of models in the ensembling portion proved to be valuable as well.

Further studies with the client’s insight on the features would be interesting. Even without disclosing what the data represents, their domain knowledge about relationships of the features to the target, and interactions between features could provide additional feature creation.

**Appendix**

1. **Code**

A rendered notebook containing code for the base analysis can be accessed at:

<https://nbviewer.org/github/rickfontenot/QTW/blob/main/Case%20Study%207/case7_rick.ipynb>