Probability of Default (PD) Model Development

University of Texas @ Dallas, FTEC 6334 (Fall 2020)

Rounok Joardar, Hamesh Alcendor

Outline

Project Goal: *Create an effective Probability of Default (PD) model using ML techniques*

Stretch Goal: *Does probability of default decrease if per period payment is reduced?*

- **Data description**
- **Exploratory Data Analysis (EDA) and data preparation**
- **Feature engineering**
- **Model building and evaluation**
- **Model interpretation**
- **Summary**

Dataset: Loan Application and Default Info

MAIN DATASET (application.csv)

- Contains detailed application information about clients
- Target column is defined as: 1 = client with payment difficulties, 0 = all other cases
- **307,511 rows** – one row per client
- **120 columns** – one column per feature

CUSTOMER_ID_#

CREDIT BUREAU DATA (bureau.csv)

- Application information about clients previous loans from other institutions
- **1,716,428 rows, 17 columns**

CUSTOMER_ID_#

PREVIOUS LOAN DATA (bureau.csv)

- Status information about clients previous loans from this loan provider (Home Credit)
- **1,670,214 rows, 37 columns**

Bureau_balance.csv

- Monthly credit balance
- Behavioral data

POS_CASH_balance.csv

- Monthly balance of previous loans
- Behavioral data

Installment_payments.csv

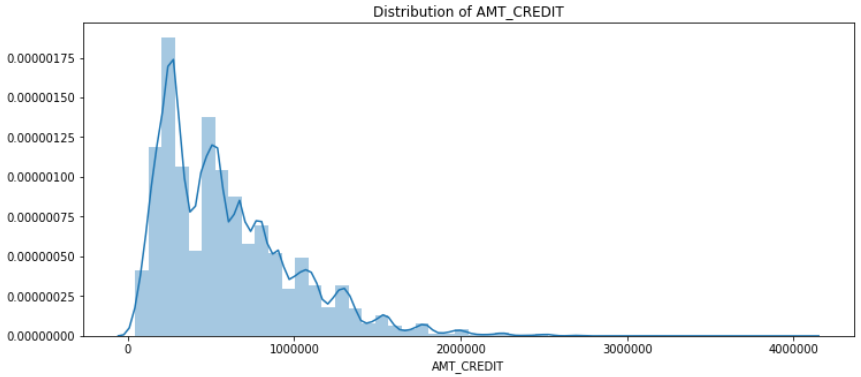
- Past payment data per installment
- Behavioral data

Credit_card_balance.csv

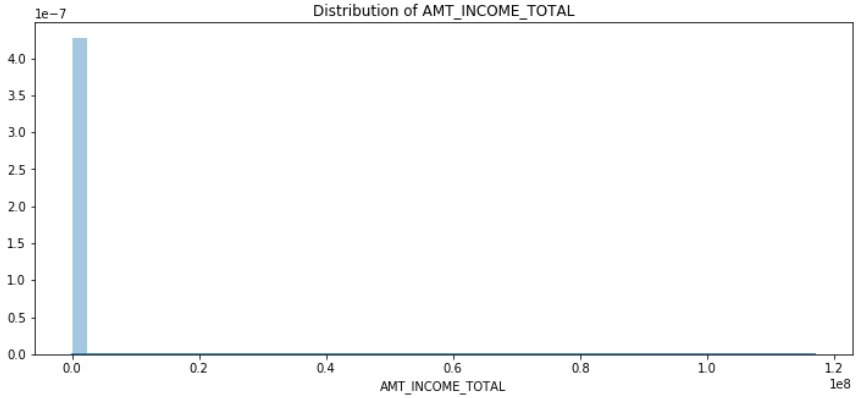
- Monthly balance of previous credit card loans
- Behavioral data

- Dataset provided by an organization called **Home Credit**, "*a global platform which centrally manages core strategy, technology, risk, product and funding functions for consumer finance operations*"
- Target data is highly asymmetrical – 8% ones, 92% zeros

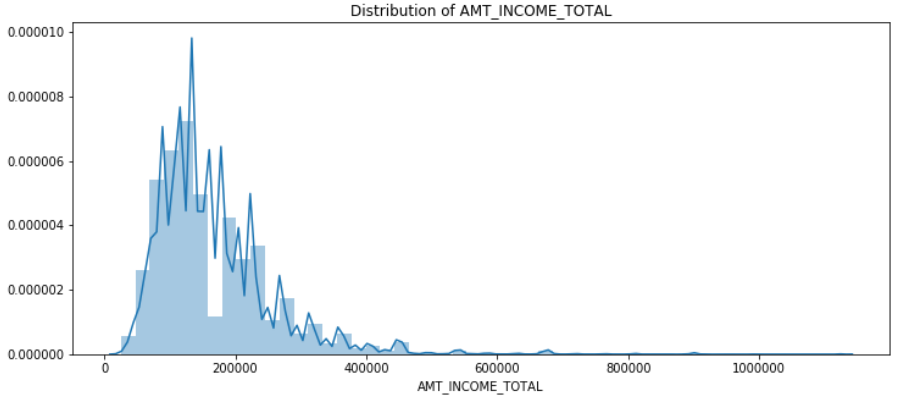
Exploratory Data Analysis



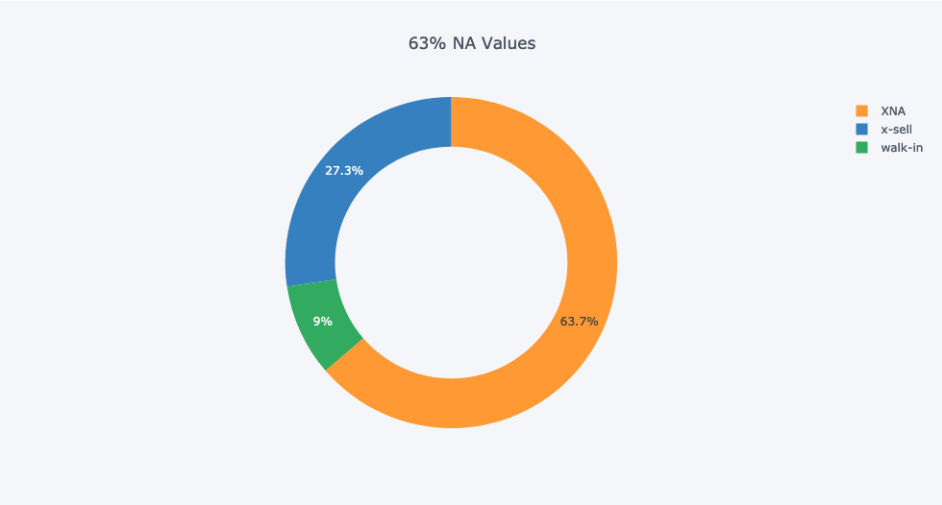
- **Distribution of each of the features was explored to check for data quality**
- **As expected, much of the data was left skewed**
- **Some features had to be further cleaned before they could be used**

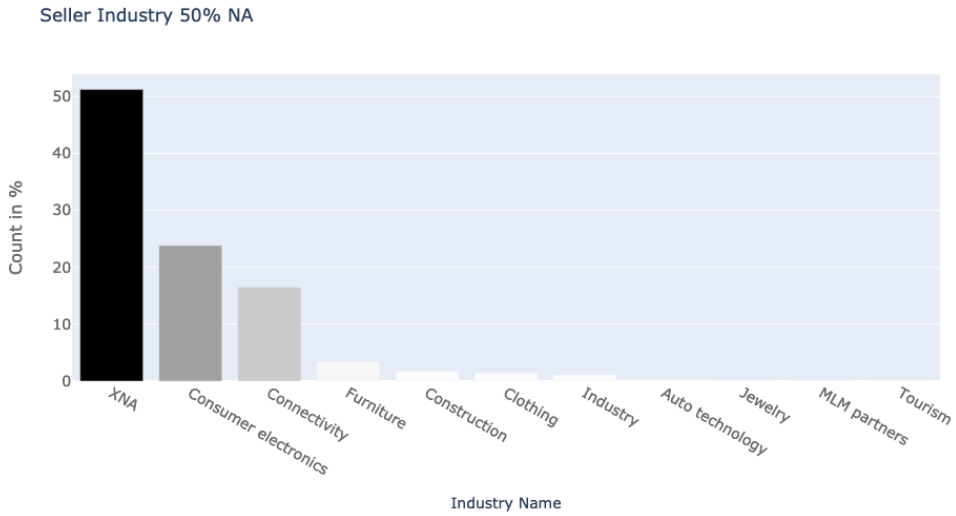


remove outliers

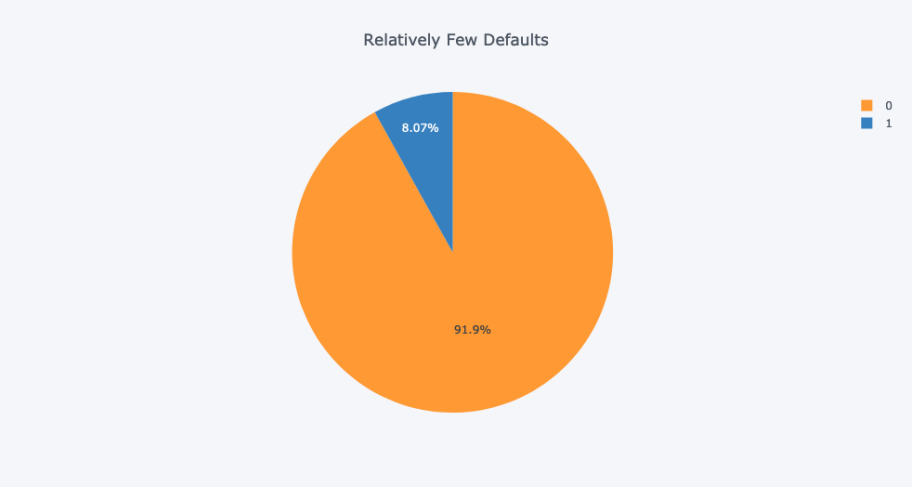


Exploratory Data Analysis





- **Many of the features are missing data for over half the applicants**



- **Less than 10% of the applicants default on their loans**
- **This will need to be addressed when splitting the train/test data**

Data Preparation and Feature Selection

Drop features with more than 25% missing values

```
dataIn.dropna(thresh=0.25*len(dataIn), axis=1)
```

↓

Create dummy variables for categorical features

```
for col in proc_df.dtypesdataIn.dtypes == object.index:
    make_my_dummy = dataIn.pop(col)
    tempDummy = pd.get_dummies(make_my_dummy, prefix=col5:)
    tempDummy.drop(tempDummy.columns0, axis=1, inplace=True)
    dataIn = pd.concat(dataIn, tempDummy, axis=1)
```

↓

Split data into train/test subsets (80/20)

```
X_train, X_test, y_train, y_test =
    train_test_split(X, y, stratify=y, test_size=0.2,
    random_state=37)
```

↓

Perform imputation to fill in remaining missing values

```
num_transformer = Pipeline(steps=(imputer,
                  SimpleImputer(strategy=median)),
                  (scaler, StandardScaler()))
```

↓

- **Run classifier (XGBM)**
- **Perform predictions**
- **View feature importances**
- **Select top 25 features**

```
clf = XGBClassifier(max_depth=maxDepth, learning_rate=learnRate,
n_estimators=numTrees, min_child_weight=childWt, scale_pos_weight=1,
random_state=33)
clf = clf.fit(X_train, y_train)
y_test_score = clf.predict_proba(X_test):, 1
testScore = roc_auc_score(y_test, y_test_score)
```

↓

- **Create model and predict using reduced feature set**
- **Compare before vs after ROC-AUC scores**

Data Preparation and Initial Feature Selection

Drop features with more than 25% missing values

→ *120 features reduced to 105*

↓

Create dummy variables for categorical features

↓

Split data into train/test subsets (80/20)

↓

Perform imputation to fill in remaining missing values

↓

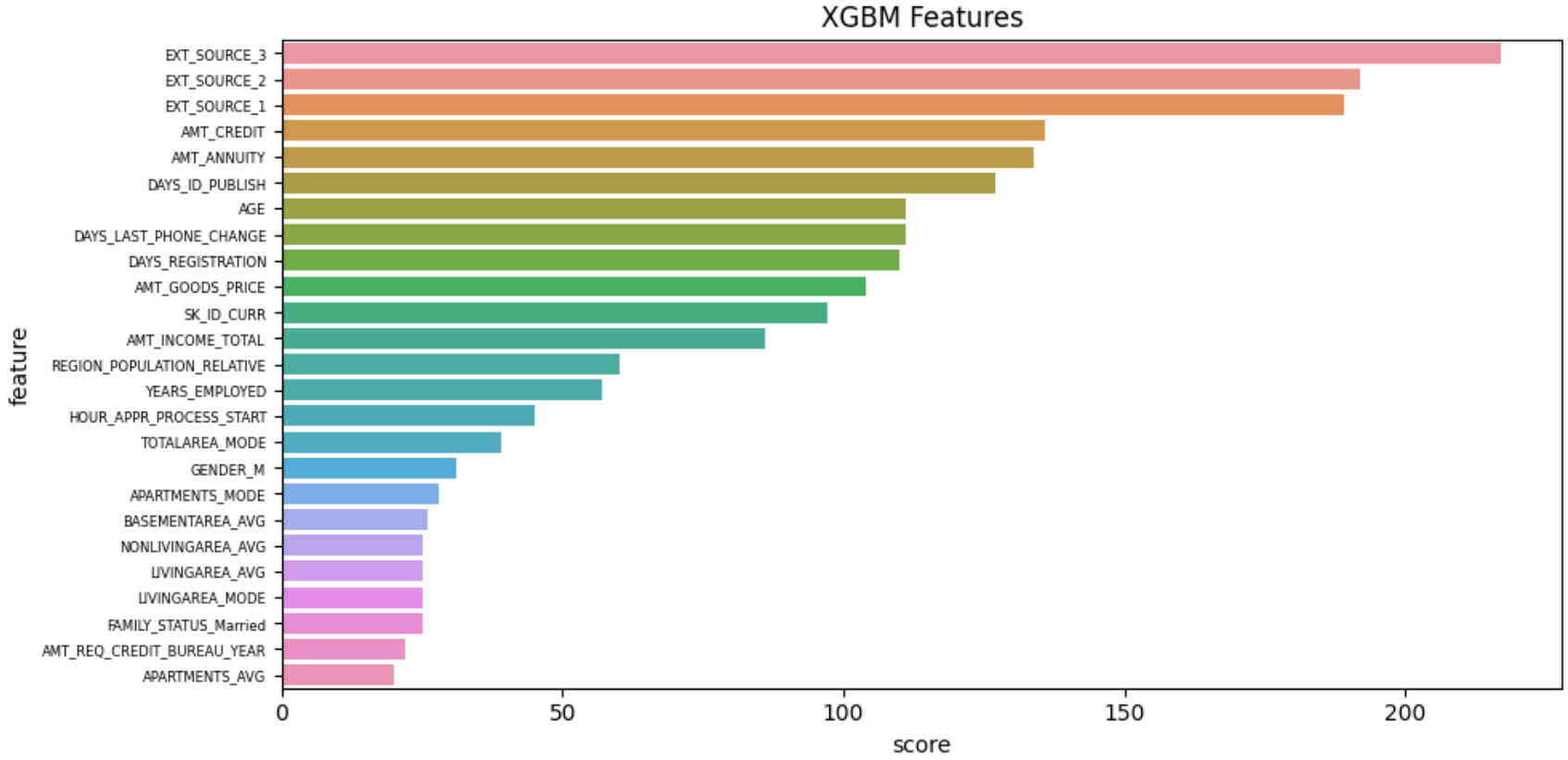
- Run classifier (XGBM)
- Perform predictions
- View feature importances
- Select top 25 features

→ *105 features reduced to 25*

↓

- Create model and predict using reduced feature set
- Compare before vs after ROC-AUC scores

→ **AUC score with all 120 features: 0.7557**
AUC score with top 25 features: 0.7452



Feature Engineering

Purpose: *Improve predictive power of ML models by introducing novel features*

- **Weight of Evidence and Information Value**
- **Include supplementary data available in credit bureau reports + clients previous application history**
  - **Example: number of previous loans taken out**
- **Append statistics such as mean/max/min/sum of supplemental information (aggregations)**
- **Add new features based on simple logic**
- **Eliminate highly correlated features**

Weight of Evidence (WoE) and Information Value (IV)

```
AMT_INCOME_TOTAL: array(2.565e+04, 1.170e+05, 1.800e+05, 1.170e+08),
AMT_CREDIT: array(  45000.,  337500.,  679500., 4050000.),
AMT_ANNUITY: array( 1615.5,  19552.5,  30717. , 258025.5),
AMT_GOODS_PRICE: array(  40500.,  292500.,  675000., 4050000.),
EXT_SOURCE_1: array(0.01456813, 0.39241613, 0.61765109, 0.96269277),
EXT_SOURCE_2: array(8.17361652e-08, 4.66980758e-01, 6.33707415e-01, 8.54999666e-01),
EXT_SOURCE_3: array(5.27265239e-04, 4.31191798e-01, 6.26304277e-01, 8.96009549e-01),
AGE: array(21., 37., 50., 69.),
YEARS_EMPLOYED: array( 0.,  2.,  6., 49.),
EDUCATION_TYPE_Higher education: array(0., 1.),
EDUCATION_TYPE_Incomplete higher: array(0., 1.),
EDUCATION_TYPE_Lower secondary: array(0., 1.),
EDUCATION_TYPE_Secondary / secondary special: array(0., 1.),
HOUSING_TYPE_House / apartment: array(0., 1.),
HOUSING_TYPE_Municipal apartment: array(0., 1.),
HOUSING_TYPE_Office apartment: array(0., 1.),
HOUSING_TYPE_Rented apartment: array(0., 1.),
HOUSING_TYPE_With parents: array(0., 1.),
OWN_CAR_Y: array(0., 1.),
OWN_REALTY_Y: array(0., 1.)}
```

	Variable_Name	Category	...	WOE		Information_Value
0	AGE	(20.999, 37.0	...	0.301379		0.075482
1	AGE	(37.0, 50.0	...	-0.035757	Merge	0.075482
2	AGE	(50.0, 69.0	...	-0.377744		0.075482
3	AMT_ANNUITY	(1615.499, 19552.5	...	-0.064670		0.006762
4	AMT_ANNUITY	(19552.5, 30717.0	...	0.112013		0.006762
5	AMT_ANNUITY	(30717.0, 258025.5	...	-0.055877		0.006762
6	AMT_CREDIT	(44999.999, 337500.0	...	-0.044895	Merge	0.031568
7	AMT_CREDIT	(337500.0, 679500.0	...	0.217496		0.031568
8	AMT_CREDIT	(679500.0, 4050000.0	...	-0.212747		0.031568

Numerical features broken into bins

- **Used for optimal binning of numerical features**
- **Helps with feature selection**
- **Commonly used in credit industry**
- **Regulatory concerns**

$$\text{WoE} = \log\left[\frac{\text{Relative frequency of 1s}}{\text{Relative frequency of 0s}}\right]$$

$$\text{IV} = \text{WoE} \times \sum\left[\text{Distrib(1s)} - \text{Distrib(0s)}\right]$$

Bins with small WoE merged with adjacent bins

Feature Engineered Data

Credit Bureau data (aggregations)

DAYS_CREDIT	min, max, mean, var
DAYS_CREDIT_ENDDATE	min, max, mean
DAYS_CREDIT_UPDATE	mean
CREDIT_DAY_OVERDUE	max, mean
AMT_CREDIT_MAX_OVERDUE	mean
AMT_CREDIT_SUM	max, mean, sum
AMT_CREDIT_SUM_DEBT	max, mean, sum
AMT_CREDIT_SUM_OVERDUE	mean
AMT_CREDIT_SUM_LIMIT	mean, sum
AMT_ANNUITY	max, mean
CNT_CREDIT_PROLONG	sum
MONTHS_BALANCE_MIN	min
MONTHS_BALANCE_MAX	max
MONTHS_BALANCE_SIZE	mean, sum

Installment pymt data (aggregations)

NUM_INSTALMENT_VERSION	nunique
DPD	max, mean, sum
DBD	max, mean, sum
PAYMENT_PERC	max, mean, sum, var
PAYMENT_DIFF	max, mean, sum, var
AMT_INSTALMENT	max, mean, sum
AMT_PAYMENT	min, max, mean, sum
DAYS_ENTRY_PAYMENT	max, mean, sum

Previous applications data (aggregations)

AMT_ANNUITY	min, max, mean
AMT_APPLICATION	min, max, mean
AMT_CREDIT	min, max, mean
APP_CREDIT_PERC	min, max, mean, var
AMT_DOWN_PAYMENT	min, max, mean
AMT_GOODS_PRICE	min, max, mean
HOUR_APPR_PROCESS_START	min, max, mean
RATE_DOWN_PAYMENT	min, max, mean
DAYS_DECISION	min, max, mean
CNT_PAYMENT	mean, sum

New features added

DAYS_EMPLOYED_RATIO	DAYS_EMPLOYED / DAYS_BIRTH (Age)
INCOME_CREDIT_RATIO	AMT_INCOME_TOTAL / AMT_CREDIT
INCOME_PER_PERSON	AMT_INCOME_TOTAL / CNT_FAM_MEMBERS
ANNUITY_INCOME_RATIO	AMT_ANNUITY / AMT_INCOME_TOTAL
PAYMENT_RATE	AMT_ANNUITY / AMT_CREDIT

119 new features added (total 798 columns after one-hot encoding of categorical features)

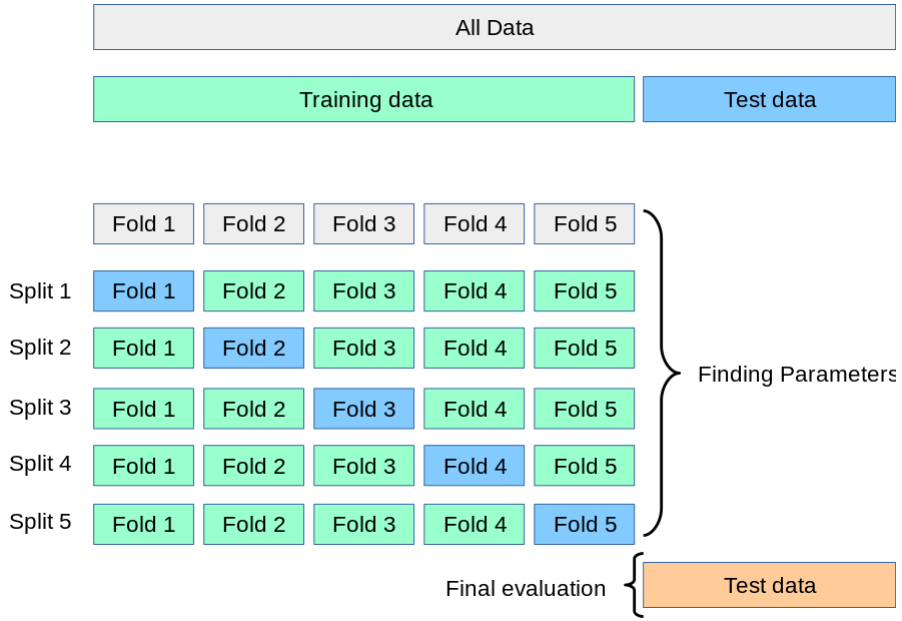
Modeling – Setup

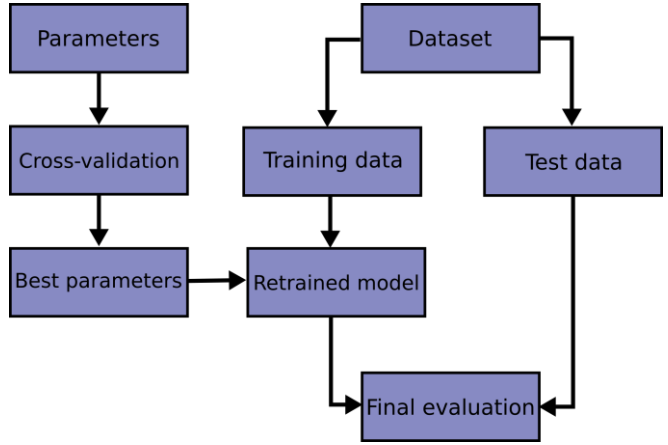
- **Classifiers used in this project:**
  - **Logistic regression (starting baseline)**
  - **Light Gradient Boosting (LGBM)**
  - **Extreme Gradient Boosting (XGBM)**
  - **Neural Network**

- **k-fold cross-validation used to measure and validate model quality (k = 10)**

- **Ensemble model and neural network model hyperparameters tuned using grid-search**
  - **Limited search range and reduced dataset size to conserve time**

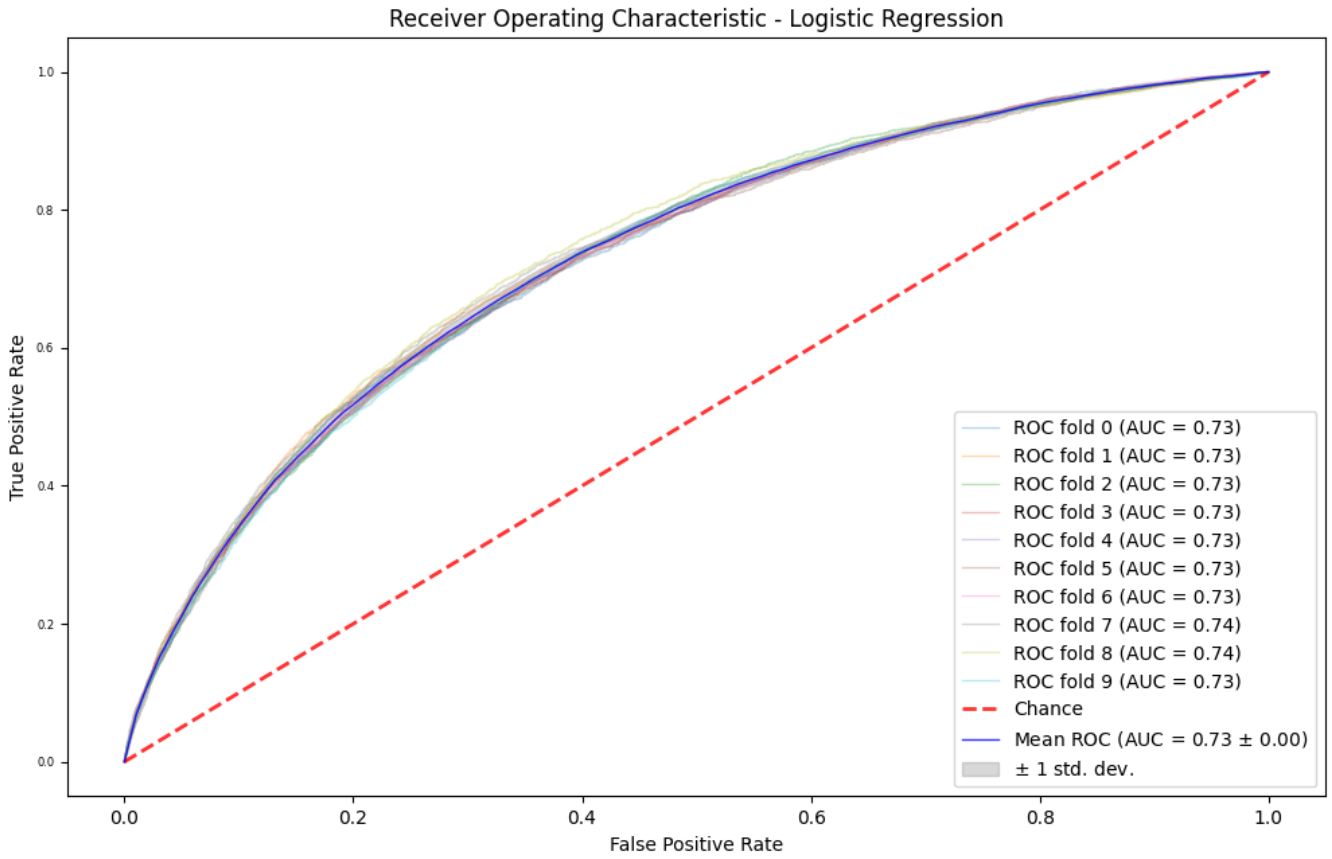
- **ROC-AUC score used as quality metric**






```
gridSearch = GridSearchCV(estimator=model, param_grid=param_grid, cv=StratifiedKFold(numFolds), scoring='roc_auc', verbose=100)
```

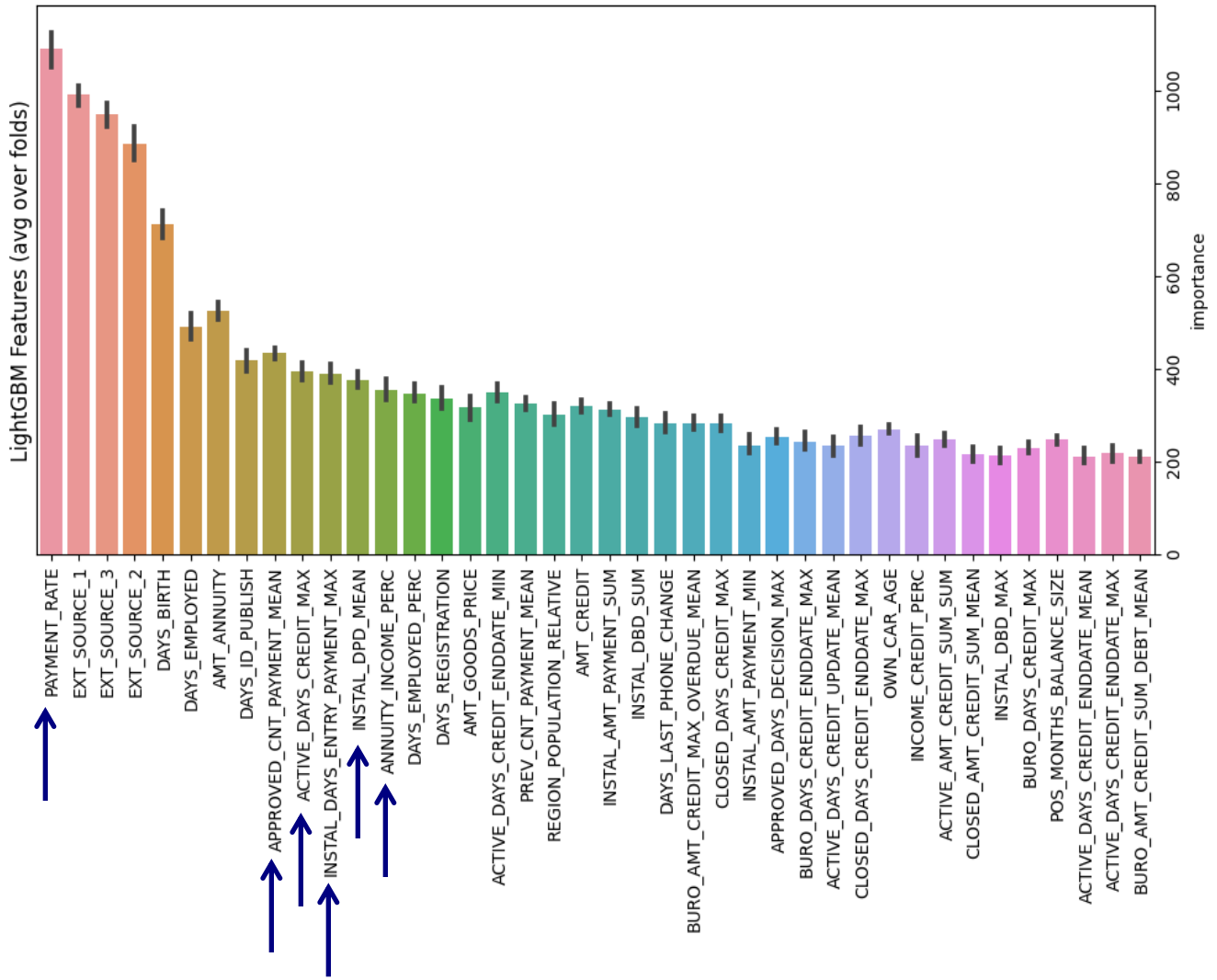
Modeling Results – Logistic Regression



- **Simple logistic regression model built as a starting baseline**
- **Used “top 25” features determined in previous stage – no feature engineering**
- **WoE used to classify numerical features**
- **10-fold cross-validation used to validate model**

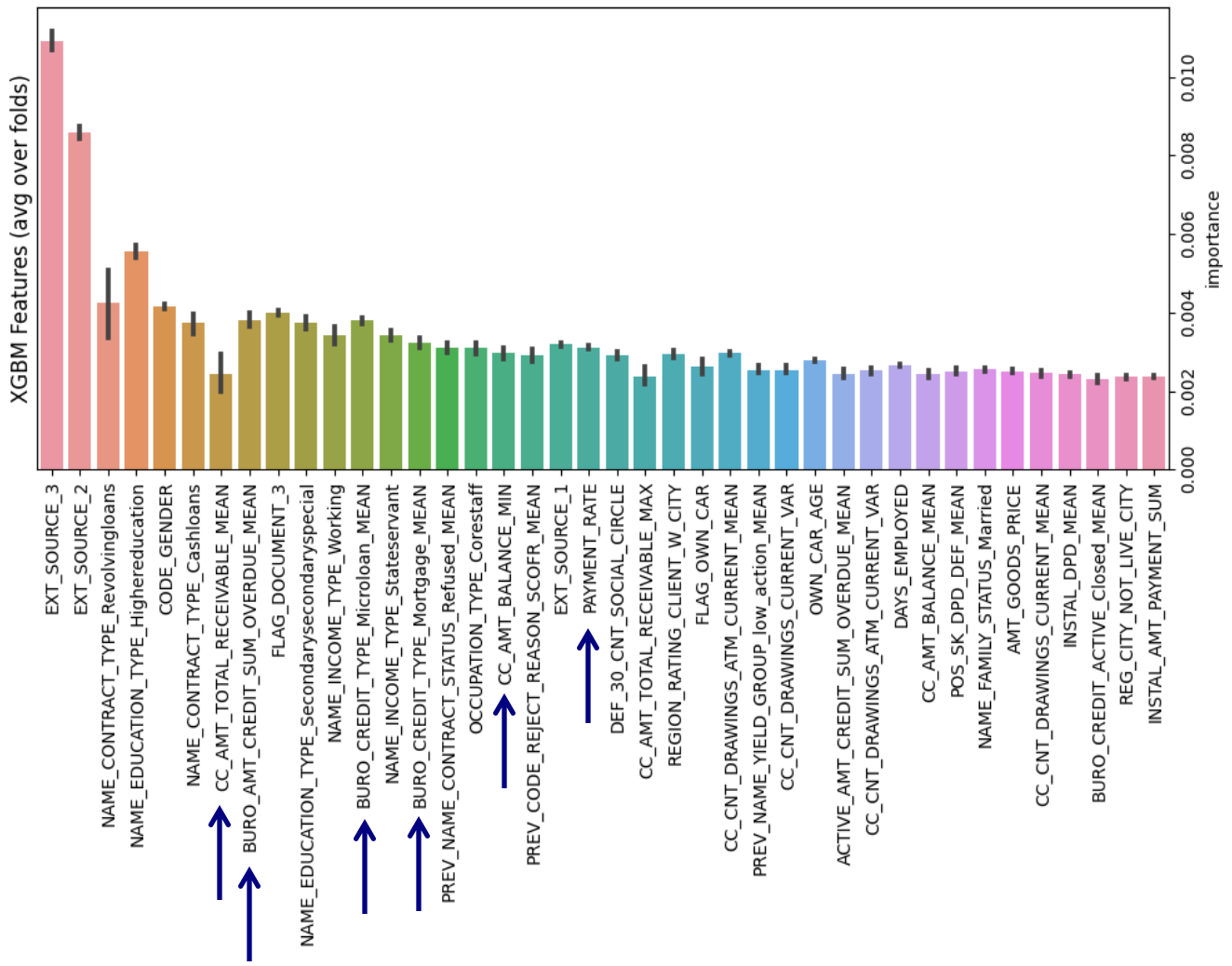
```
classifier = LogisticRegression(solver='lbfgs', C=1e5, max_iter=500, random_state=37)
```

Modeling Results – LGBM



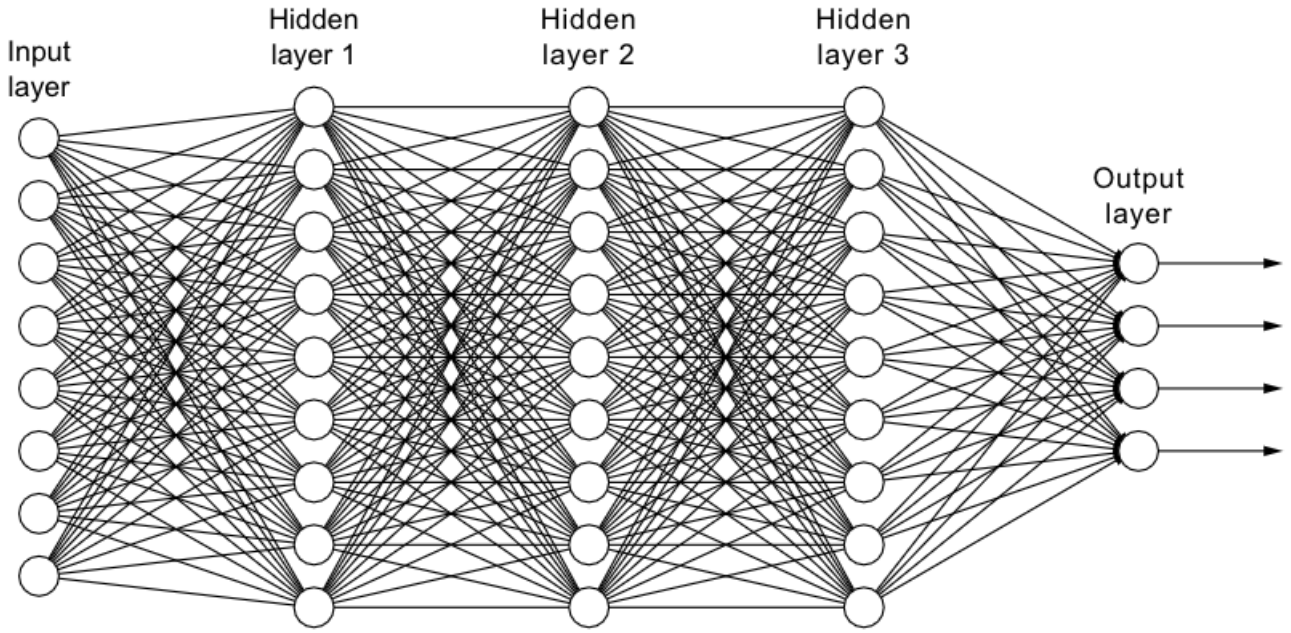
- **Results from 10-fold cross-validation**
- **Hyperparameter values used:**
  - `n_estimators = 10000`
  - `learning_rate = 0.02`
  - `num_leaves = 34`
  - `subsample = 0.9`
  - `max_depth = 8`
  - `min_child_weight = 40`
- **Top 40 features mostly from newly engineered features**
- **AUC score = 0.791635 (0.0059)**

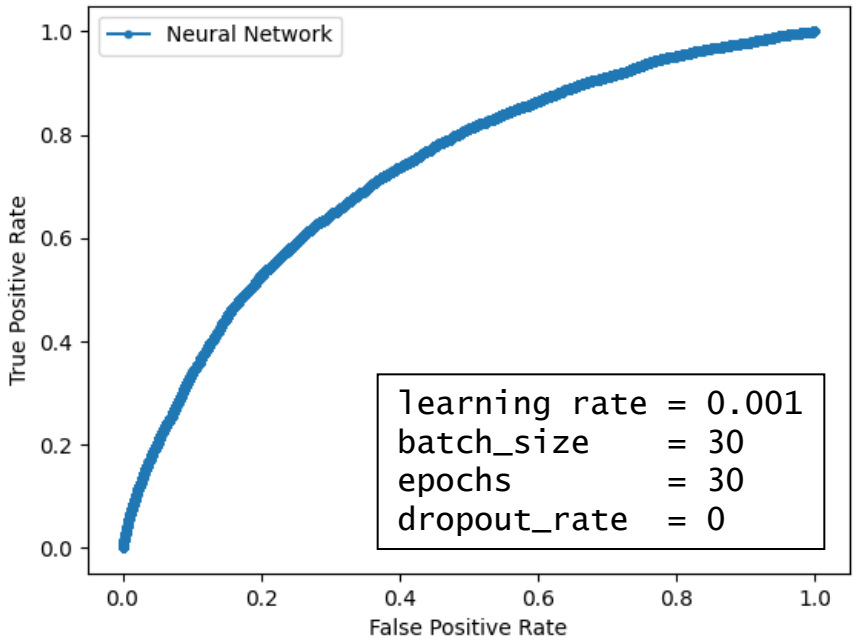
Modeling Results – XGBoost



- **Results from 10-fold cross-validation**
- **Hyperparameter values used:**
  - `n_estimators = 10000`
  - `learning_rate = 0.02`
  - `num_leaves = 34`
  - `subsample = 0.9`
  - `max_depth = 8`
  - `min_child_weight = 40`
- **Top 40 features mostly from newly engineered features**
- **AUC score = 0.793355 (0.0057)**

Modeling Results – Neural Network



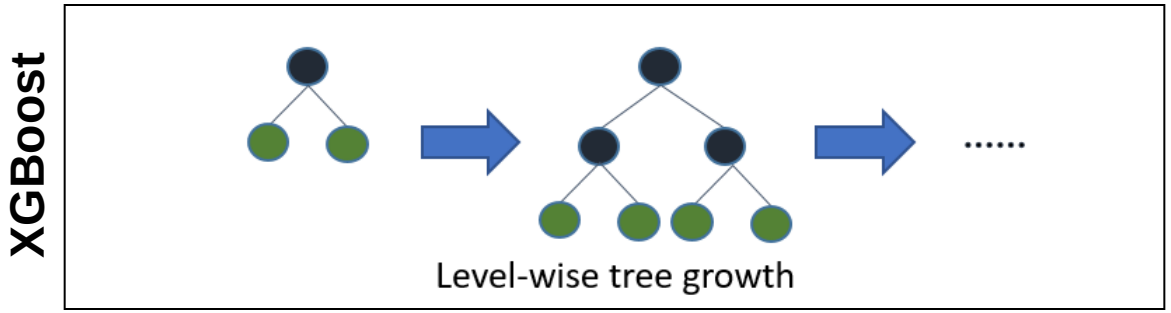


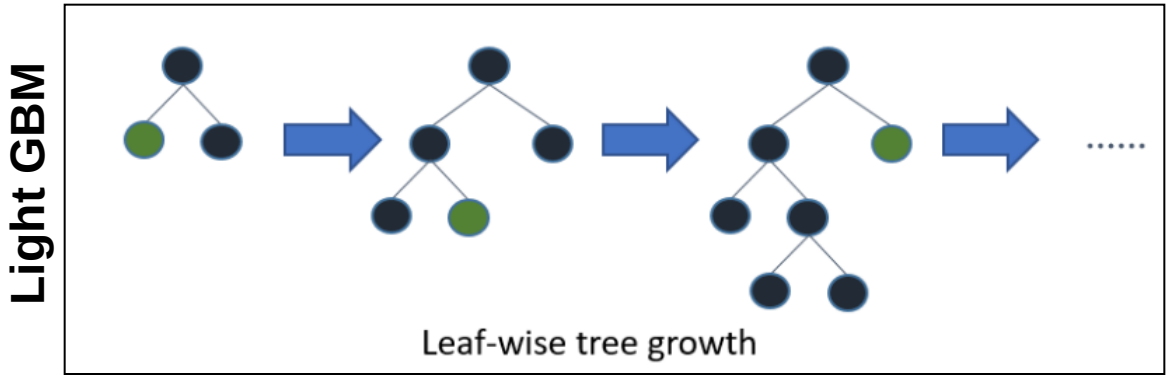
- **Used 3 fully connected hidden layers with 20 nodes per layer in final model**
  - **Smaller network (2 hidden layers, 6 nodes each) used for parameter tuning**
  - **Search grid consisted of 81 points, 5-fold cv search took 9 hours!**

- **Results:**
  - **AUC (train) = 0.737886 (σ = 0.0067)**
  - **AUC (test) = 0.741067 (σ = 0.0053)**

Modeling Results – Summary

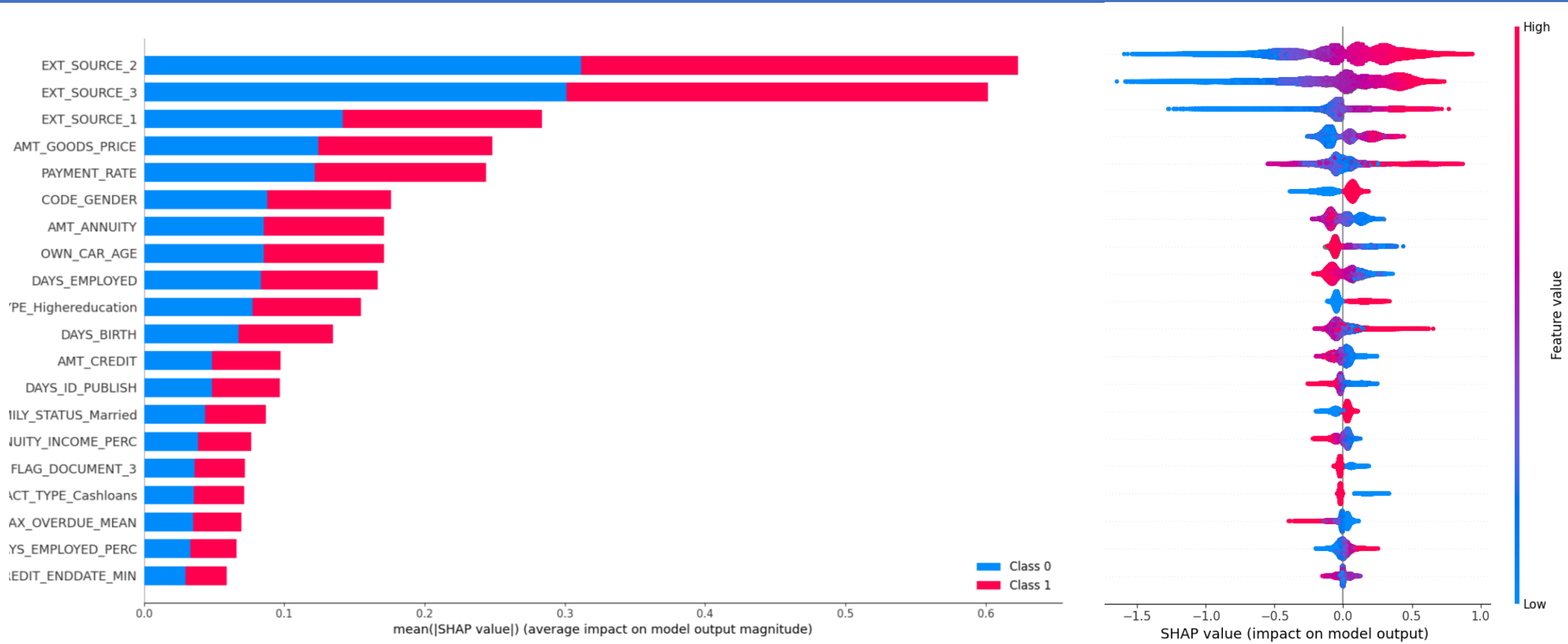
Model	Type of feature engineering	Number of new features	Execution time (min)	AUC-ROC Score
Logistic	WoE, IV	0	< 1	0.73
LGBM	New, Aggregation	119	**28.5**	0.791635
XGBoost	New, Aggregation	119	**390**	**0.793355**
Neural Network	New, Aggregation	119	31.2	**0.741067**





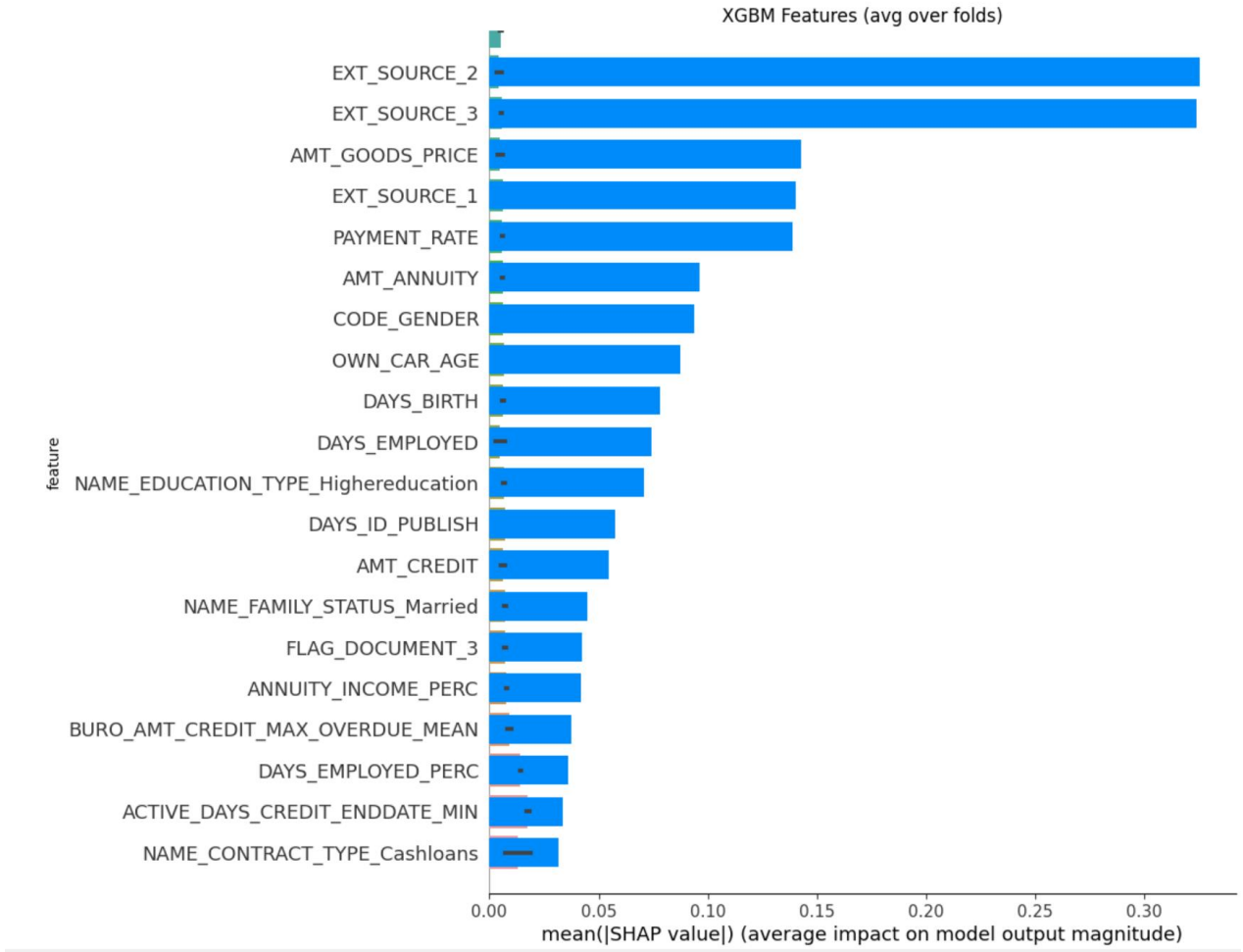
https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost

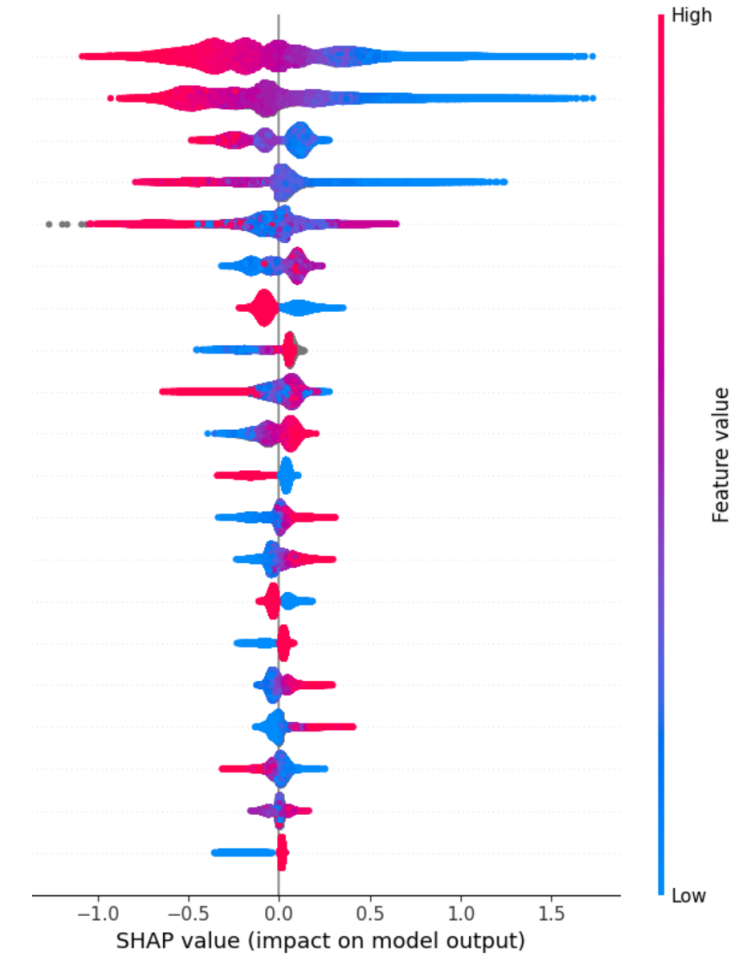
Model Interpretability – LGBM SHAP Values



- **Impact of predictor (i.e. features) on target assessed using SHAP values**
- **Variable importance plot obtained from LGBM model matches well with feature importance characteristic**
  - **15 out of top 20 are common to both characteristics (compare with slide 13)**

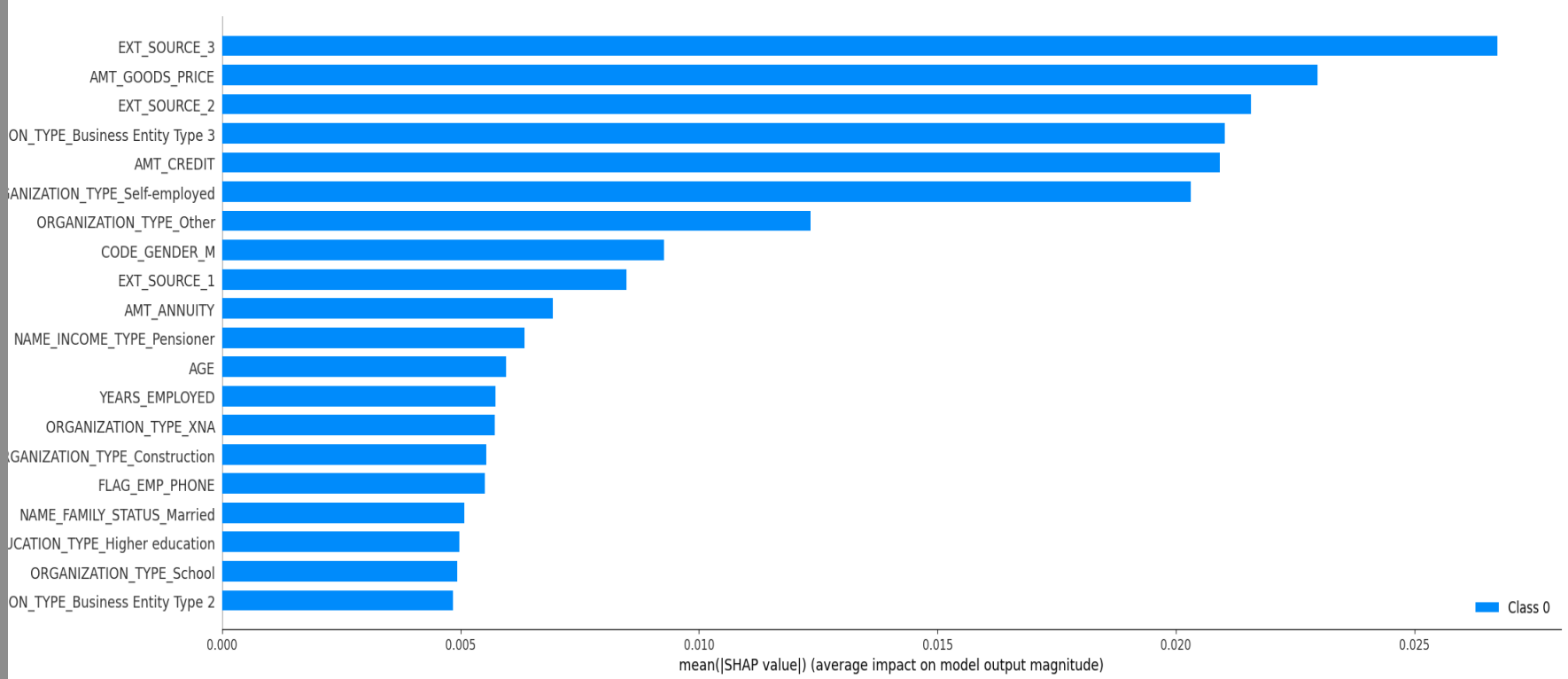
Model Interpretability – XGBoost SHAP Values

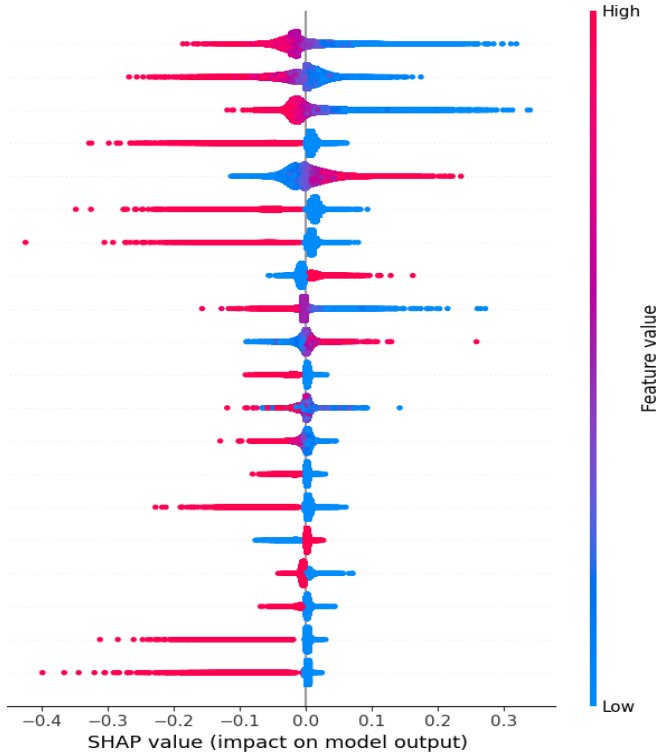




- **Impact of predictor (i.e. features) on target assessed using SHAP values**
- **Variable importance plot obtained from XGBoost model matches well with feature importance characteristic**
  - **15 out of top 20 are common to both characteristics (compare with slide 14)**

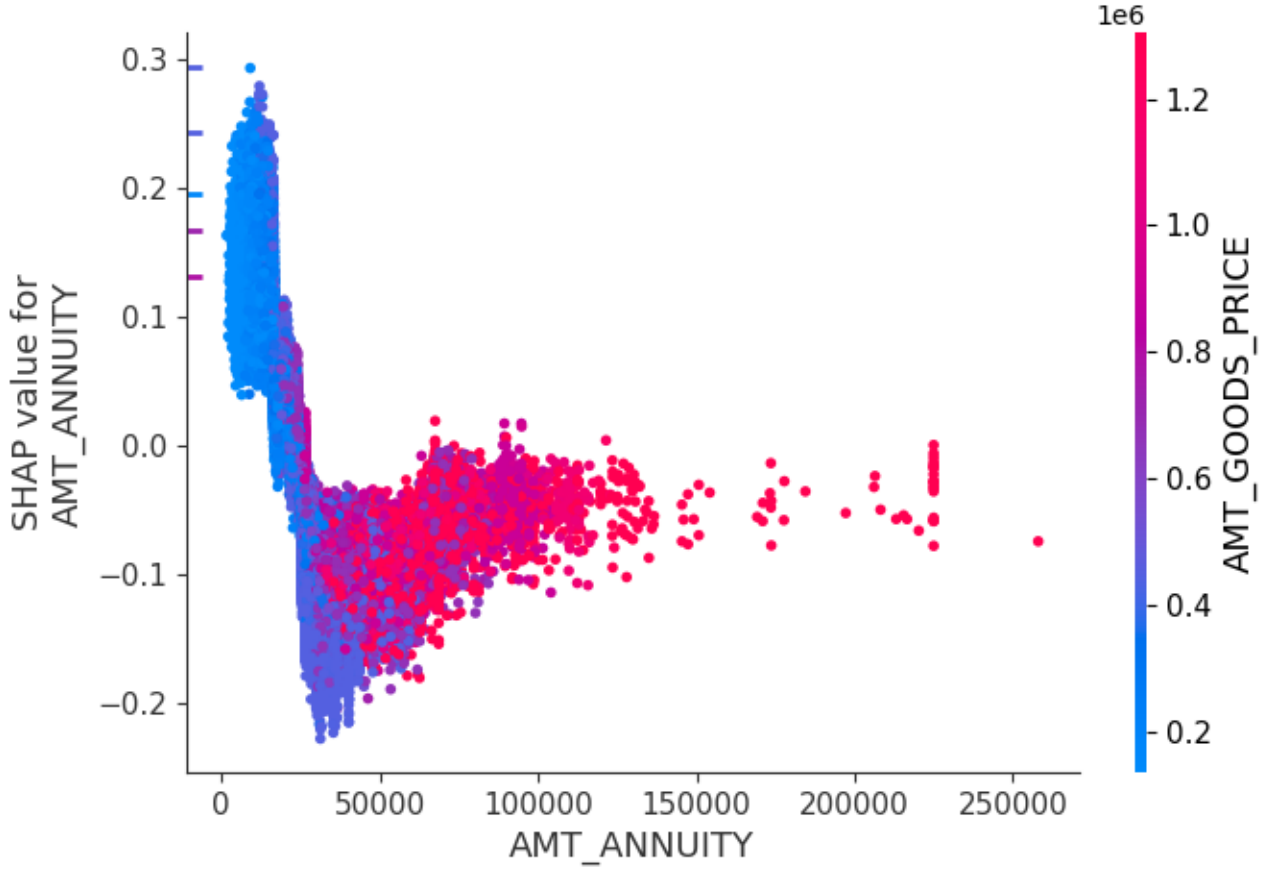
Model Interpretability – Neural Network Model SHAP Values





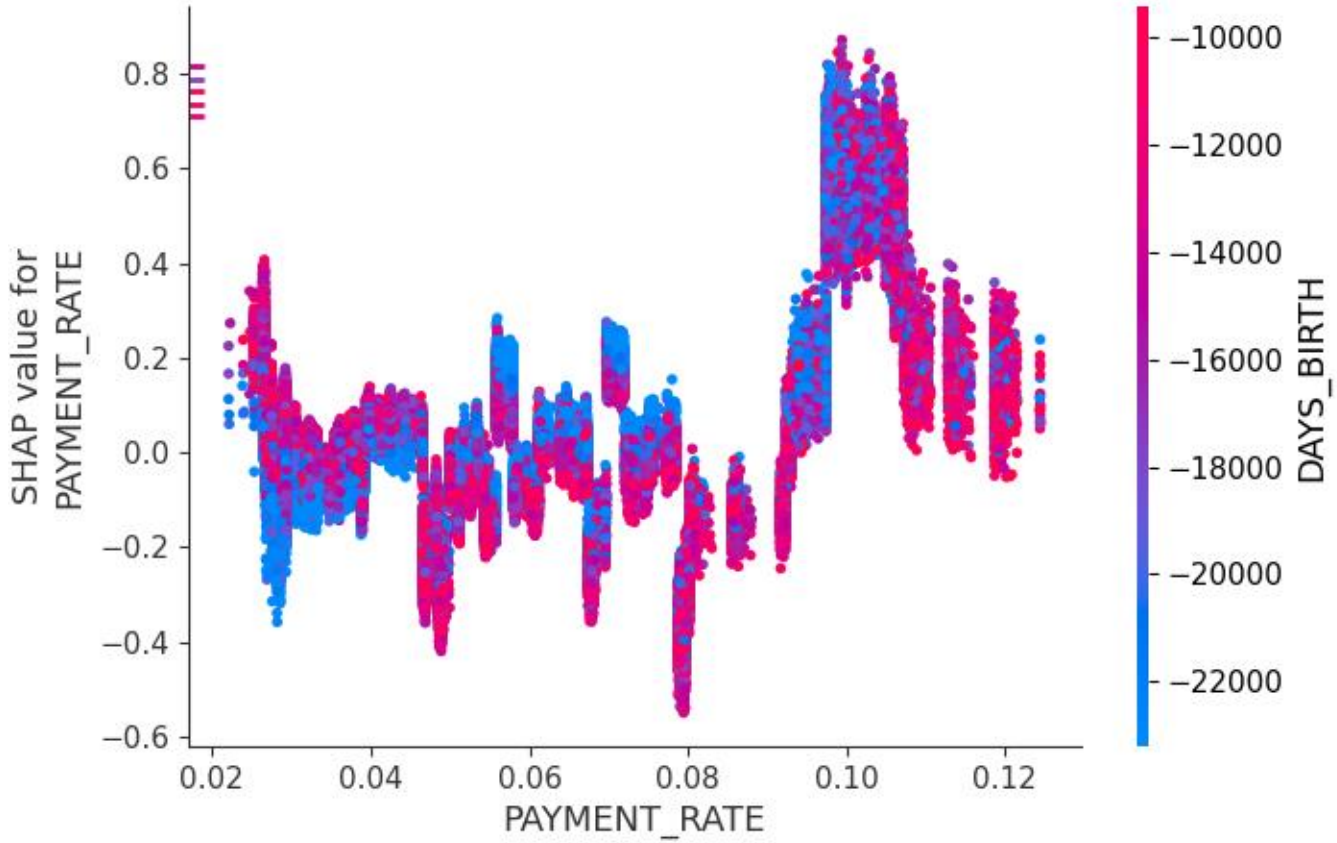
- **Impact of predictor (i.e. features) on target assessed using SHAP values**
- **Variable importance plot obtained from Neural Network model matches well with corresponding characteristics from LGBM and XGBoost models**
  - **11 of top 20 features in common with XGBoost**
  - **10 of top 20 features in common with LGBM**

Annuity Amount – PD Sensitivity



- **SHAP partial dependence plot shows the marginal effect of one feature on the predicted outcome**
- **If annuity amount can be reduced below 50K, default probability goes down**

Payment Rate – PD Sensitivity



- **SHAP partial dependence plot shows weak marginal effect of payment rate on probability of default**

Summary

- Three types of ML models were built to predict probability of default based on a dataset provided by Home Credit

- XGBoost reached the highest AUC score but took longest training time
- Neural network model had lowest AUC score
- LGBM took the shortest training time

- SHAP analysis was performed to demostrate interpretability of the models
- All 3 models showed similar feature importance
- New engineered features had a large influence on model accuracy

- Credit default rate was shown to have high sensitivity to annuity amount

Thank you!