Estimating Client Needs

Business Case 2

T. Bucci, F. Cipriani, G. Corbo, D. Fabroni, M. Lucchini

Politecnico di Milano

March 31, 2022

Table of Contents

Goal: Recommendation System

We intend to estimate some investments needs for these customers using Data Science techniques.

Data:

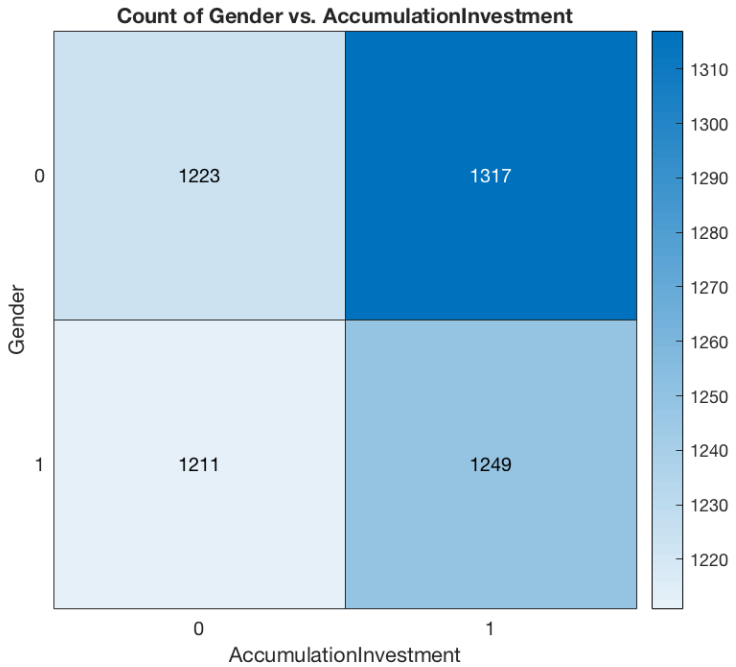
Needs:

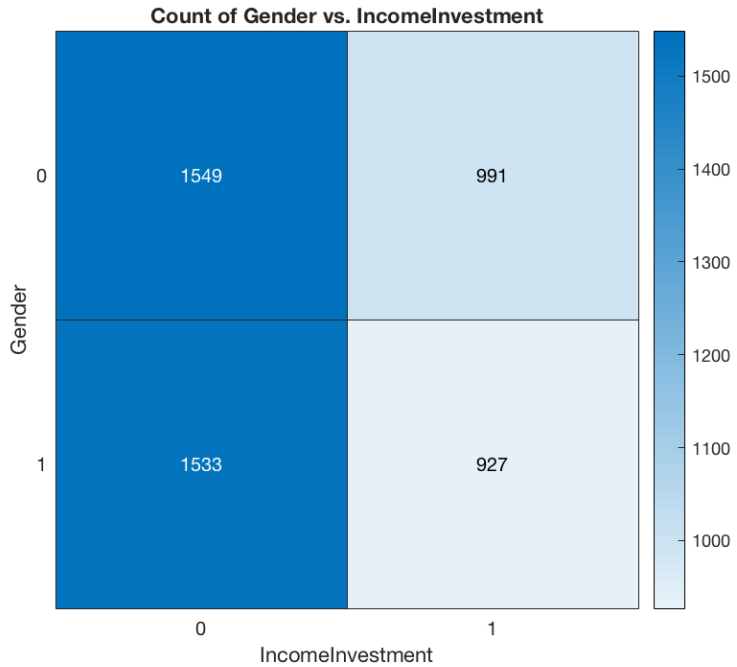
```
ID    Age    Gender    FamilyMembers    FinancialEducation    RiskPropensity
Income    Wealth    IncomeInvestment    AccumulationInvestment
```

Products:

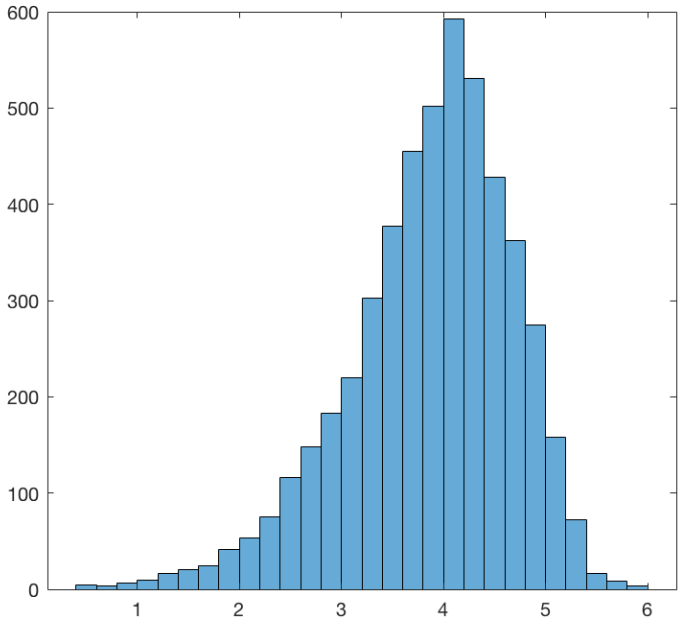
```
IDProduct    Type    Risk
```

No significant sex difference

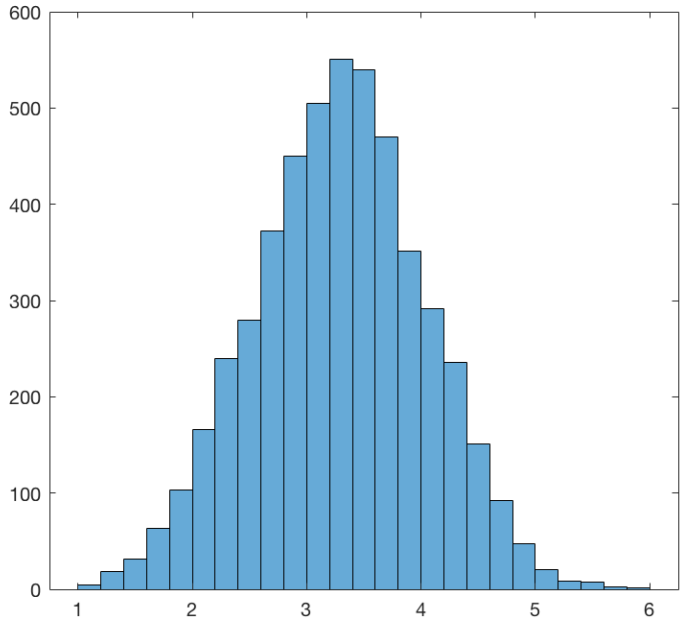




Data Transformation



(a) log(Income)

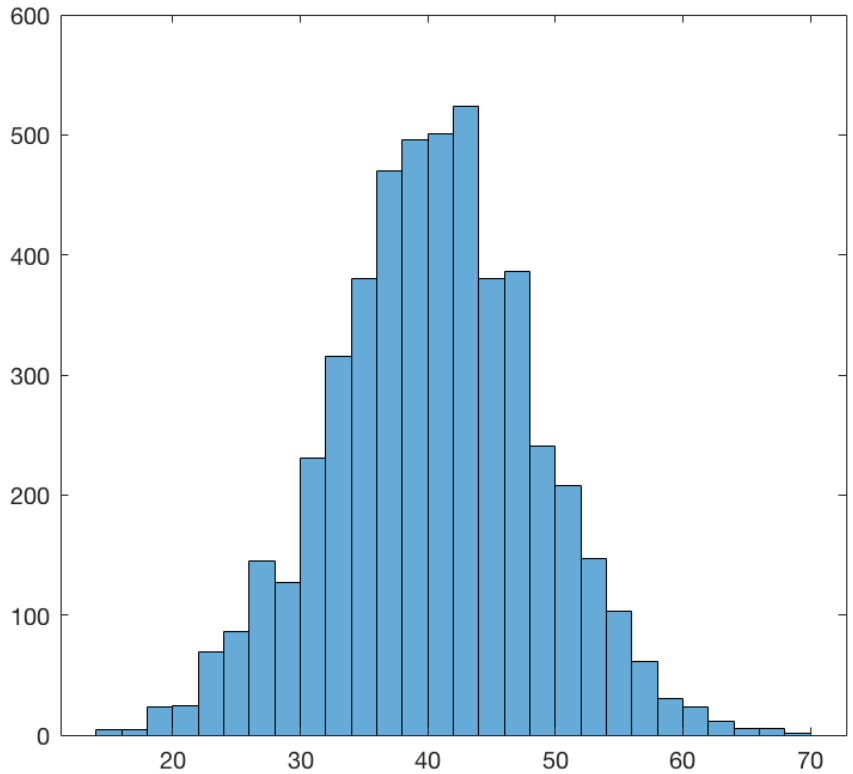


(b) $\text{Income}^{0.3}$

Figure: Comparison of transformations, we'll go with the second.

Data Transformation

Using a boxcox transformation, $\text{NewAge} = (\text{OldAge}^{0.9} - 1)/0.9$.



Data Transformation

$$\text{NewFinancialEducation} = \text{OldFinancialEducation}^{0.8}$$
$$\text{NewRiskPropensity} = \text{OldRiskPropensity}^{0.65}$$
$$\text{NewIncome} = \text{OldIncome}^{0.3}$$

Cross validation

```
nObs = size(Data, 1);
rng(10)
idxPermutation = randperm(nObs);
X = X(idxPermutation,:); % random permutation
train =  0.75 ;
cross =    0 ;

nObsTrain = round(train*nObs);
nObsCross = round(cross*nObs);

XTrain = X(1:nObsTrain,:);
XCross = X((nObsTrain+1):(nObsTrain+nObsCross), :);
XTest = X(nObsTrain+1+nObsCross:end,:);
```

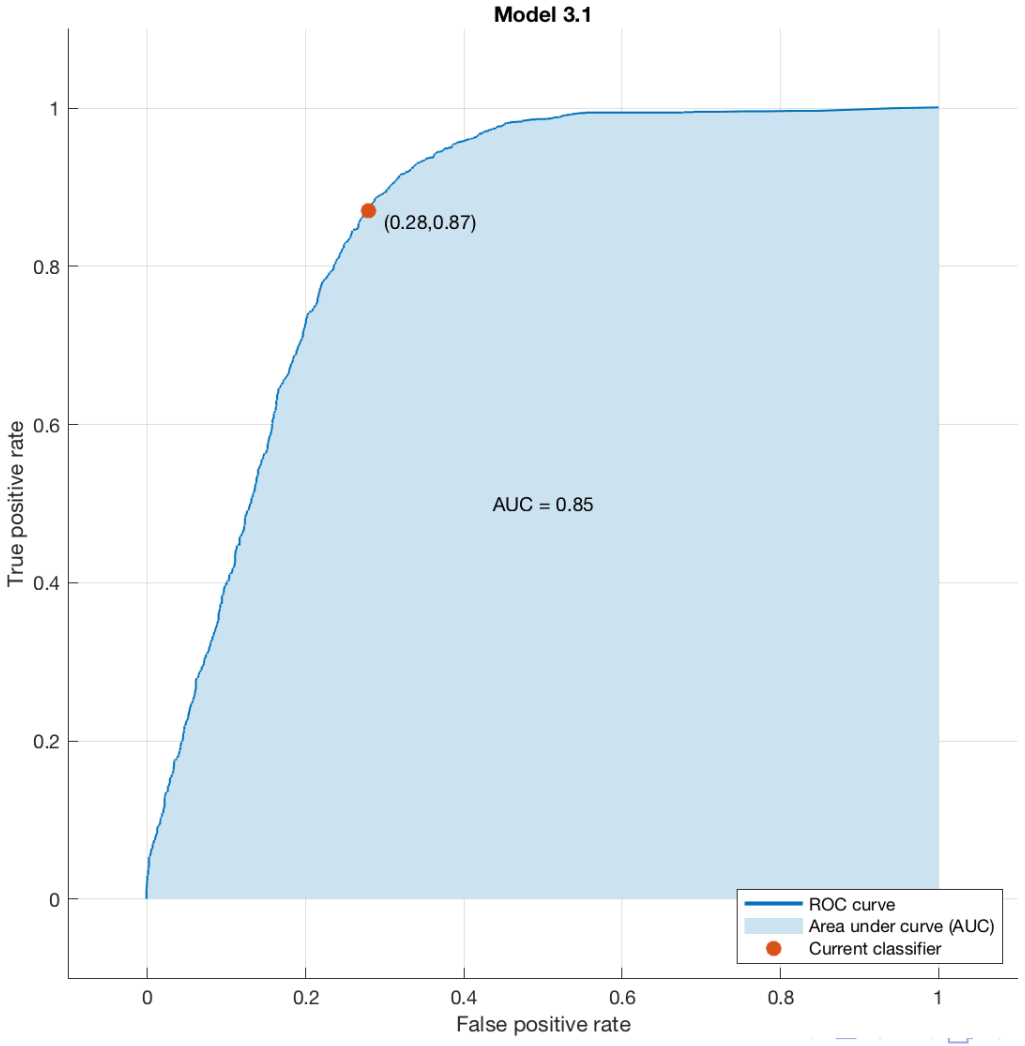
Added Y permutation equal to X

```
YInc = Data.IncomeInvestment;
YInc = YInc(idxPermutation);

yInvIncTrain = YInc(1:nObsTrain);
yInvIncCross = YInc((nObsTrain+1):(nObsTrain+nObsCross));
yInvIncTest = YInc(nObsTrain+nObsCross+1:end);

YAcc = Data.AccumulationInvestment;
YAcc = YAcc(idxPermutation);

yInvAccTrain = YAcc(1:nObsTrain);
yInvAccCross = YAcc((nObsTrain+1):(nObsTrain+nObsCross));
yInvAccTest = YAcc(nObsTrain+nObsCross+1:end);

varNames = {'Age', 'Gender', 'Family', 'FinEdu', 'Risk', 'Income', 'Wealth'};
XTrainTable=table(XTrain(:,1), XTrain(:,2), XTrain(:,3), XTrain(:,4),
XTrain(:,5), XTrain(:,6),XTrain(:,7), 'VariableNames',varNames);
XTestTable=table(XTest(:,1), XTest(:,2), XTest(:,3), XTest(:,4),
XTest(:,5), XTest(:,6),XTest(:,7), 'VariableNames',varNames);
```

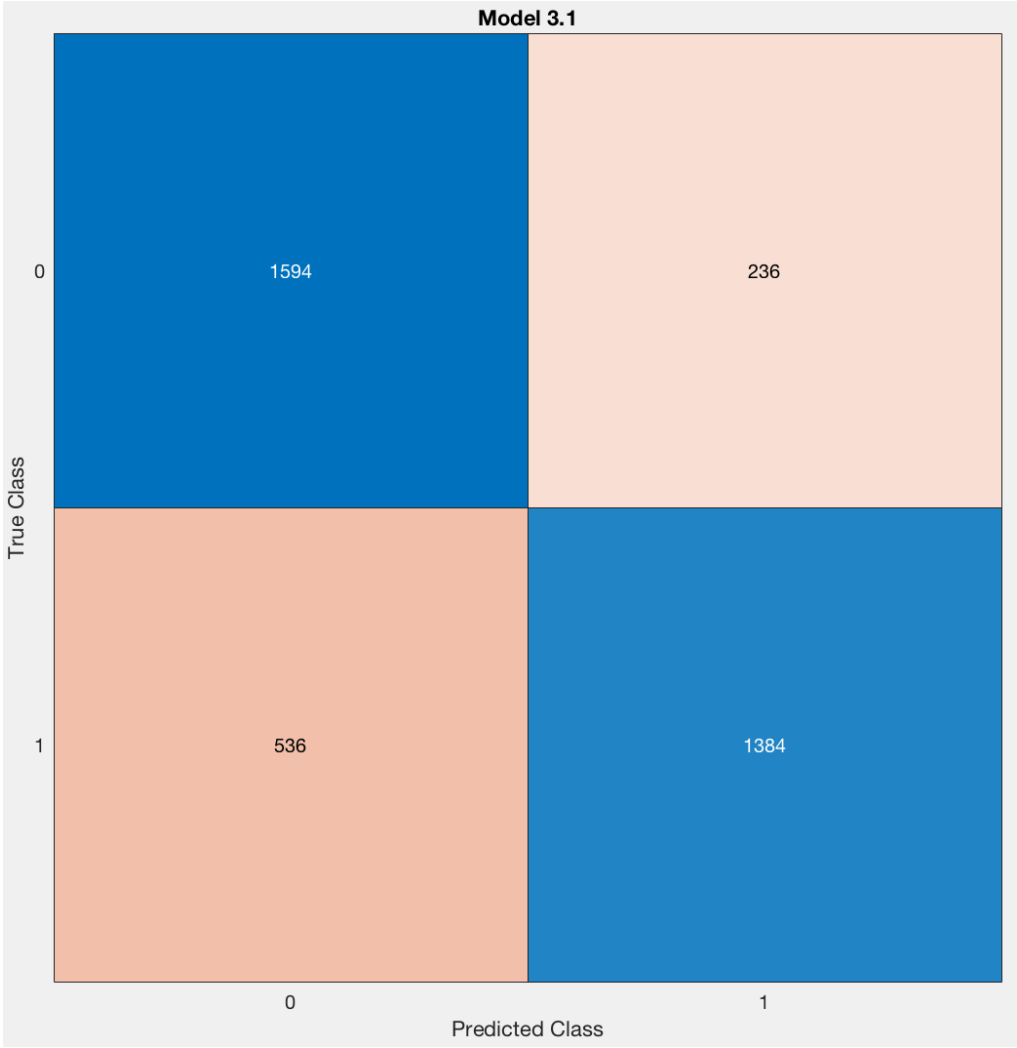
Classification Learner: model training

1.4 KNN Last change: Fine KNN	Accuracy: 65.7% 7/7 features
1.5 KNN Last change: Medium KNN	Accuracy: 69.8% 7/7 features
1.6 KNN Last change: Coarse KNN	Accuracy: 68.9% 7/7 features
1.7 KNN Last change: Cosine KNN	Accuracy: 69.8% 7/7 features
1.8 KNN Last change: Cubic KNN	Accuracy: 69.1% 7/7 features
1.9 KNN Last change: Weighted KNN	Accuracy: 69.7% 7/7 features
2 Ensemble Last change: Boosted Trees	Accuracy: **79.4%** 7/7 features
3.1 Ensemble Last change: Boosted Trees	Accuracy: **79.4%** 7/7 features
3.2 Ensemble Last change: Bagged Trees	Accuracy: 76.6% 7/7 features
3.3 Ensemble Last change: Subspace Discrimin...	Accuracy: 62.9% 7/7 features
3.4 Ensemble Last change: Subspace KNN	Accuracy: 70.0% 7/7 features
3.5 Ensemble Last change: RUSBoosted Trees	Accuracy: 75.0% 7/7 features

Classification Learner: Boosted Tree ROC



Classification Learner: Boosted Tree confusion matrix



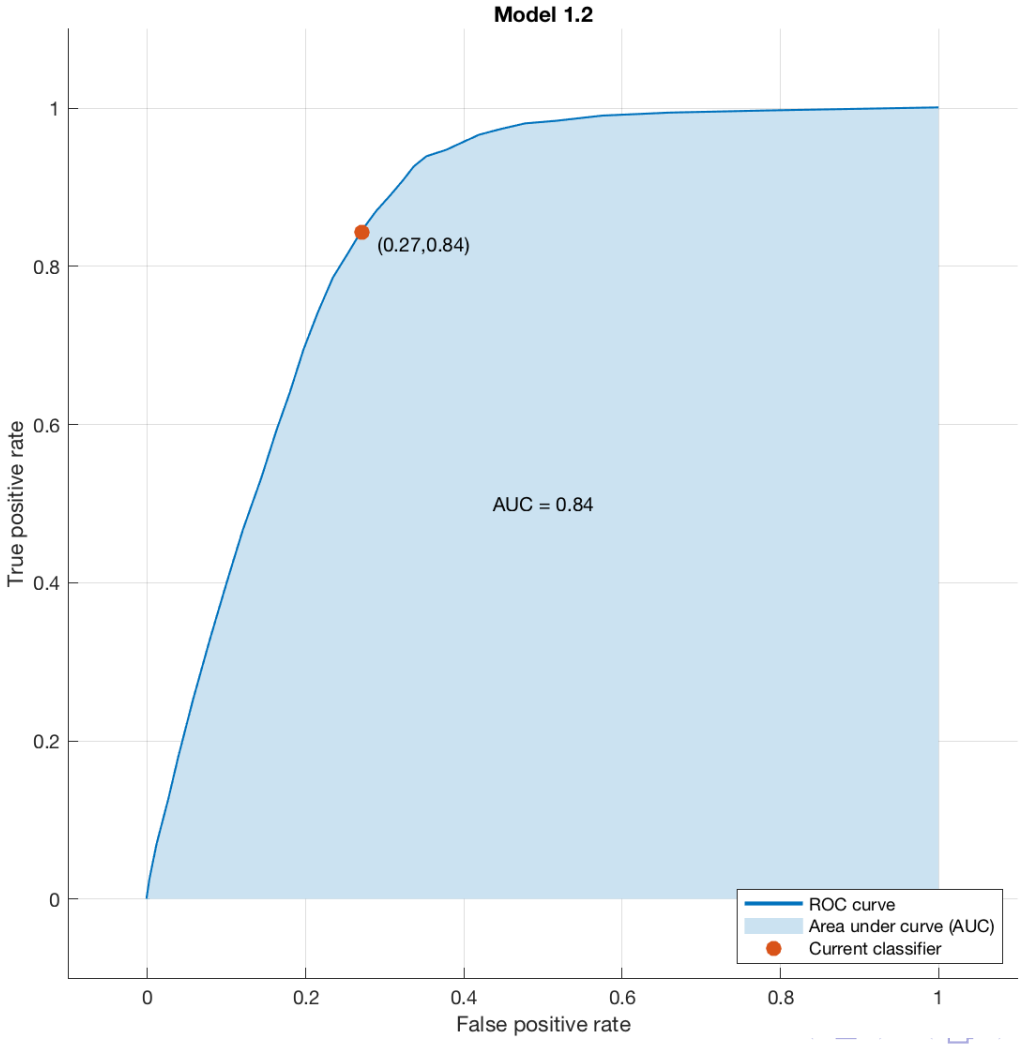
Reducing data

```
Xsmall = [rescale(IncomeWealthRatio) rescale(pAge) rescale(pFin) rescale(pIncome) rescale(pWealth)];
Xsmall = Xsmall(idxPermutation,:);
XsmallTrain = Xsmall(1:nObsTrain,:);
XsmallTest = Xsmall(nObsTrain+1:end,:);
XsmallTrainTable=table(XsmallTrain(:,1), XsmallTrain(:,2), XsmallTrain(:,3),
XsmallTrain(:,4), XsmallTrain(:,5), 'VariableNames',xnames);
```

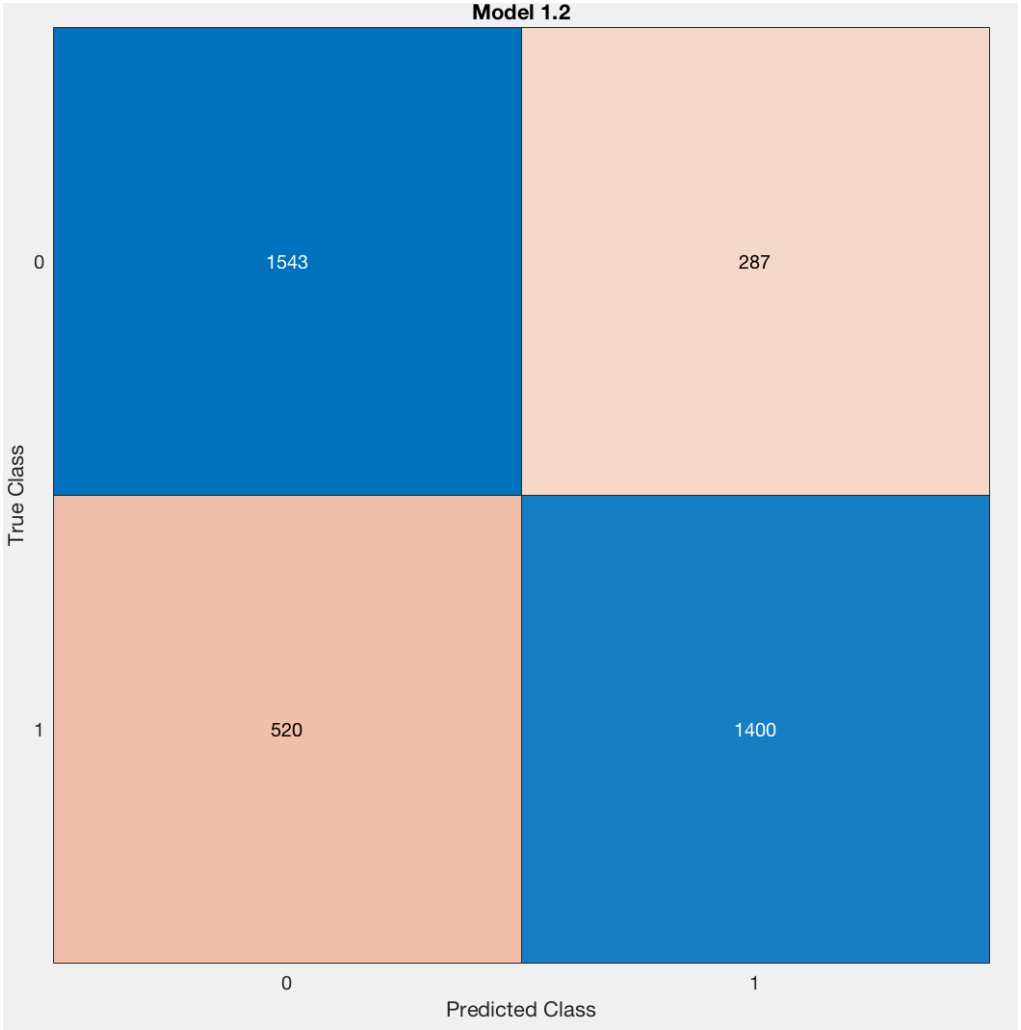
Classification Learner: model training

1.1 ☆ Ensemble	Accuracy: 78.3%	
Last change: Boosted Trees	5/5 features	
1.2 ☆ Ensemble	Accuracy: **78.5%**	
Last change: Bagged Trees	5/5 features	
1.3 ☆ Ensemble	Accuracy: 65.9%	
Last change: Subspace Discrimin...	5/5 features	
1.4 ☆ Ensemble	Accuracy: 73.4%	
Last change: Subspace KNN	5/5 features	
1.5 ☆ Ensemble	Accuracy: 76.3%	
Last change: RUSBoosted Trees	5/5 features	
2.1 ☆ Tree	Accuracy: 78.0%	
Last change: Fine Tree	5/5 features	
2.2 ☆ Tree	Accuracy: 76.1%	
Last change: Medium Tree	5/5 features	
2.3 ☆ Tree	Accuracy: 68.0%	
Last change: Coarse Tree	5/5 features	
2.4 ☆ KNN	Accuracy: 70.5%	
Last change: Fine KNN	5/5 features	

Classification Learner: Bagged Tree ROC



Classification Learner: Bagged Tree confusion matrix



Recommending products

Using Bayesian optimization we decide to use the Bagged Tree model after evaluating 30 alternatives.

Product recommendation based on Risk Propensity