run

October 7, 2023

1 Using Machine Learning to predict Financial Crises

Author: Chris Reimann **Date created:** 2023/07/28 **Last modified:** 2023/10/07 **Description:** This notebook runs the experiments of the paper "Using Machine Learning to predict Financial Crises: An Evaluation of different Learning Algorithms for Early Warning Models".

1.1 Setup

Load the latest code from the repository and import it into the virtual machine.

[2]:
```
from prepareData import Data
from doExperiment import Experiment
import pandas as pd
```

1.2 Load Data

Construct datasets using specified indicator variables from the MacroHistory database. Nominal values of local currencies are transformed to GDP-ratios, while growth rates are computed for percentages and index values. For additional details see Chapter 2.3 of the paper.

[4]:
```
# Define Indicator Sets
iv_macro = ["rconsbarro", "iy", "money", "xrusd", "cpi", "ca"]
iv_credit = ["tloans", "debtServ", "yieldCurve", "ltd", "debtgdp",␣
 ↪"globaltloans", "globalyieldCurve"]
iv_ca = iv_credit + ["hpnom"]
iv_all = iv_macro + iv_ca

# Construct Datasets
df_macro = Data(indicators = iv_macro, crisisData = "MacroHistory").
 ↪getReady("Macro")
df_credit = Data(indicators = iv_credit, crisisData = "MacroHistory").
 ↪getReady("Credit")
df_ca = Data(indicators = iv_ca, crisisData = "MacroHistory").getReady("Credit␣
 ↪& Asset")
df_all = Data(indicators = iv_all, crisisData = "MacroHistory").getReady("All")
```

```
Macro: The final dataset contains 1591 observations with 63 distinct crisis
events.
Credit: The final dataset contains 1373 observations with 60 distinct crisis
```

```
events.
Credit & Asset: The final dataset contains 1159 observations with 46 distinct
crisis events.
All: The final dataset contains 1101 observations with 41 distinct crisis
events.
```

1.3 Construct Experiments

Specify models to be tested. Available models are Logit, KNeighbors, RandomForest, ExtraTrees, SVM and NeuralNet.

[6]:
```
# Specify Models to be tested
models = ["Logit", "KNeighbors", "RandomForest", "ExtraTrees", "SVM",␣
 ↪"NeuralNet"]

# Define Experiments for all Indicator Sets.
ex_macroIS = Experiment(df_macro, models, "InSample")
ex_creditIS = Experiment(df_credit, models, "InSample")
ex_caIS = Experiment(df_ca, models, "InSample")
ex_allIS = Experiment(df_all, models, "InSample")

ex_macro = Experiment(df_macro, models, "CrossVal")
ex_credit = Experiment(df_credit, models, "CrossVal")
ex_ca = Experiment(df_ca, models, "CrossVal")
ex_all = Experiment(df_all, models, "CrossVal")
```

1.4 Run Experiments

1.4.1 In-Sample

Compute ROC values for all models trained and tested on the whole dataset.

[4]:
```
ex_macroIS.run(disableTqdm = True)
ex_creditIS.run(disableTqdm = True)
ex_caIS.run(disableTqdm = True)
ex_allIS.run(disableTqdm = True)

resIS = pd.concat([ex_macroIS.auc, ex_creditIS.auc, ex_caIS.auc, ex_allIS.auc],␣
 ↪axis = 0)
resIS
```

[4]:

	Set	Model	AUC
0	Macro	KNeighbors	1.000000
1	Macro	RandomForest	0.975804
2	Macro	ExtraTrees	0.825952
3	Macro	SVM	0.773601
4	Macro	Logit	0.709806
5	Macro	NeuralNet	0.707416
6	Macro	Random Assignment	0.500000

```
0          Credit         KNeighbors  1.000000
1          Credit       RandomForest  0.983538
2          Credit         ExtraTrees  0.934343
3          Credit                SVM  0.918643
4          Credit          NeuralNet  0.851912
5          Credit              Logit  0.790217
6          Credit  Random Assignment  0.500000
0  Credit & Asset         KNeighbors  1.000000
1  Credit & Asset       RandomForest  0.992726
2  Credit & Asset         ExtraTrees  0.955384
3  Credit & Asset                SVM  0.943323
4  Credit & Asset          NeuralNet  0.864574
5  Credit & Asset              Logit  0.808380
6  Credit & Asset  Random Assignment  0.500000
0             All         KNeighbors  1.000000
1             All       RandomForest  0.993947
2             All         ExtraTrees  0.963143
3             All                SVM  0.949269
4             All          NeuralNet  0.927175
5             All              Logit  0.858661
6             All  Random Assignment  0.500000
```

1.4.2 Out-of-Sample

Run experiments in cross-validation setting.

[5]:
```
n = 100 # Specify number of cross-validation iterations

ex_macro.run(n)
ex_credit.run(n)
ex_ca.run(n)
ex_all.run(n)

resCrossVal = pd.concat([ex_macro.auc, ex_credit.auc, ex_ca.auc, ex_all.auc])
resCrossVal
```

```
Macro: Random Assignment:   0%|          | 0/100 [00:00<?, ?it/s]

Macro: Logit:   0%|          | 0/100 [00:00<?, ?it/s]

Macro: KNeighbors:   0%|          | 0/100 [00:00<?, ?it/s]

Macro: RandomForest:   0%|          | 0/100 [00:00<?, ?it/s]

Macro: ExtraTrees:   0%|          | 0/100 [00:00<?, ?it/s]

Macro: SVM:   0%|          | 0/100 [00:00<?, ?it/s]

Macro: NeuralNet:   0%|          | 0/100 [00:00<?, ?it/s]

Credit: Random Assignment:   0%|          | 0/100 [00:00<?, ?it/s]
```

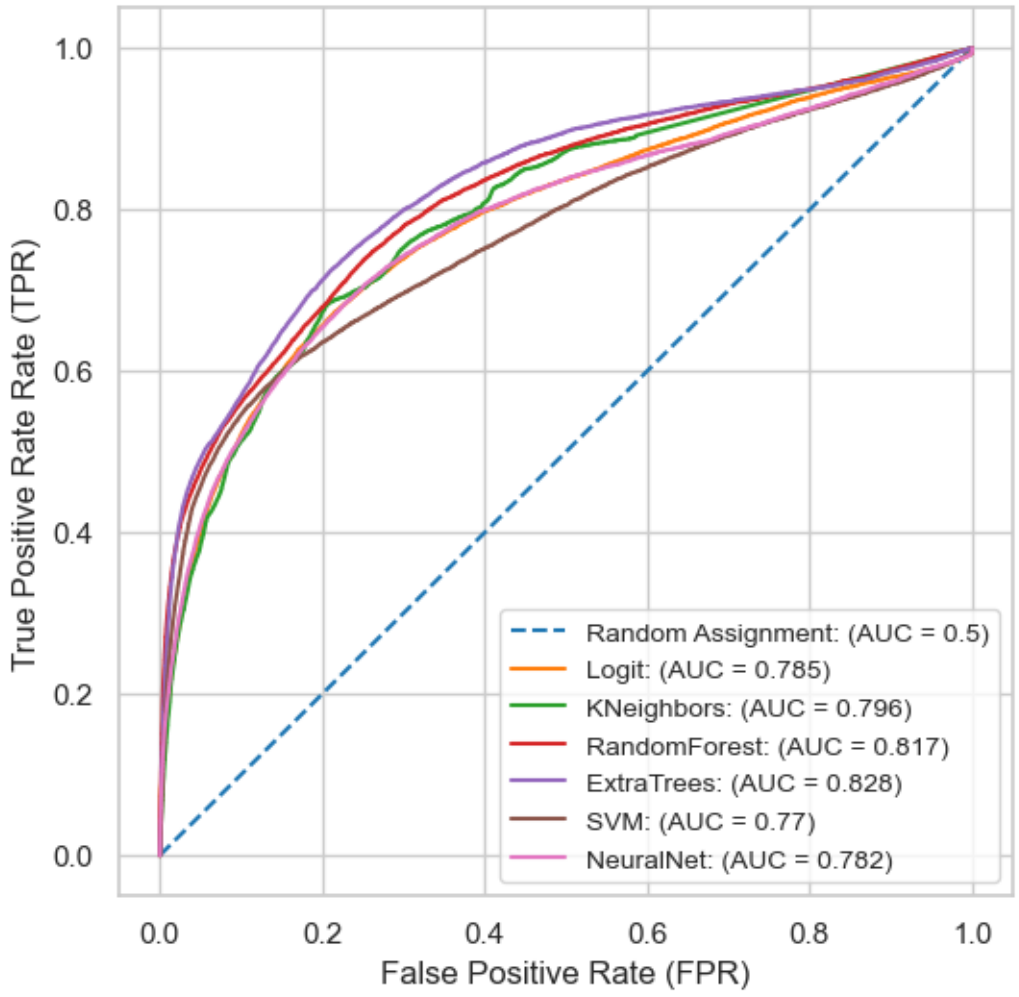
```
Credit: Logit:   0%|          | 0/100 [00:00<?, ?it/s]
Credit: KNeighbors:   0%|          | 0/100 [00:00<?, ?it/s]
Credit: RandomForest:   0%|          | 0/100 [00:00<?, ?it/s]
Credit: ExtraTrees:   0%|          | 0/100 [00:00<?, ?it/s]
Credit: SVM:   0%|          | 0/100 [00:00<?, ?it/s]
Credit: NeuralNet:   0%|          | 0/100 [00:00<?, ?it/s]
Credit & Asset: Random Assignment:   0%|          | 0/100 [00:00<?, ?it/s]
Credit & Asset: Logit:   0%|          | 0/100 [00:00<?, ?it/s]
Credit & Asset: KNeighbors:   0%|          | 0/100 [00:00<?, ?it/s]
Credit & Asset: RandomForest:   0%|          | 0/100 [00:00<?, ?it/s]
Credit & Asset: ExtraTrees:   0%|          | 0/100 [00:00<?, ?it/s]
Credit & Asset: SVM:   0%|          | 0/100 [00:00<?, ?it/s]
Credit & Asset: NeuralNet:   0%|          | 0/100 [00:00<?, ?it/s]
All: Random Assignment:   0%|          | 0/100 [00:00<?, ?it/s]
All: Logit:   0%|          | 0/100 [00:00<?, ?it/s]
All: KNeighbors:   0%|          | 0/100 [00:00<?, ?it/s]
All: RandomForest:   0%|          | 0/100 [00:00<?, ?it/s]
All: ExtraTrees:   0%|          | 0/100 [00:00<?, ?it/s]
All: SVM:   0%|          | 0/100 [00:00<?, ?it/s]
All: NeuralNet:   0%|          | 0/100 [00:00<?, ?it/s]
```

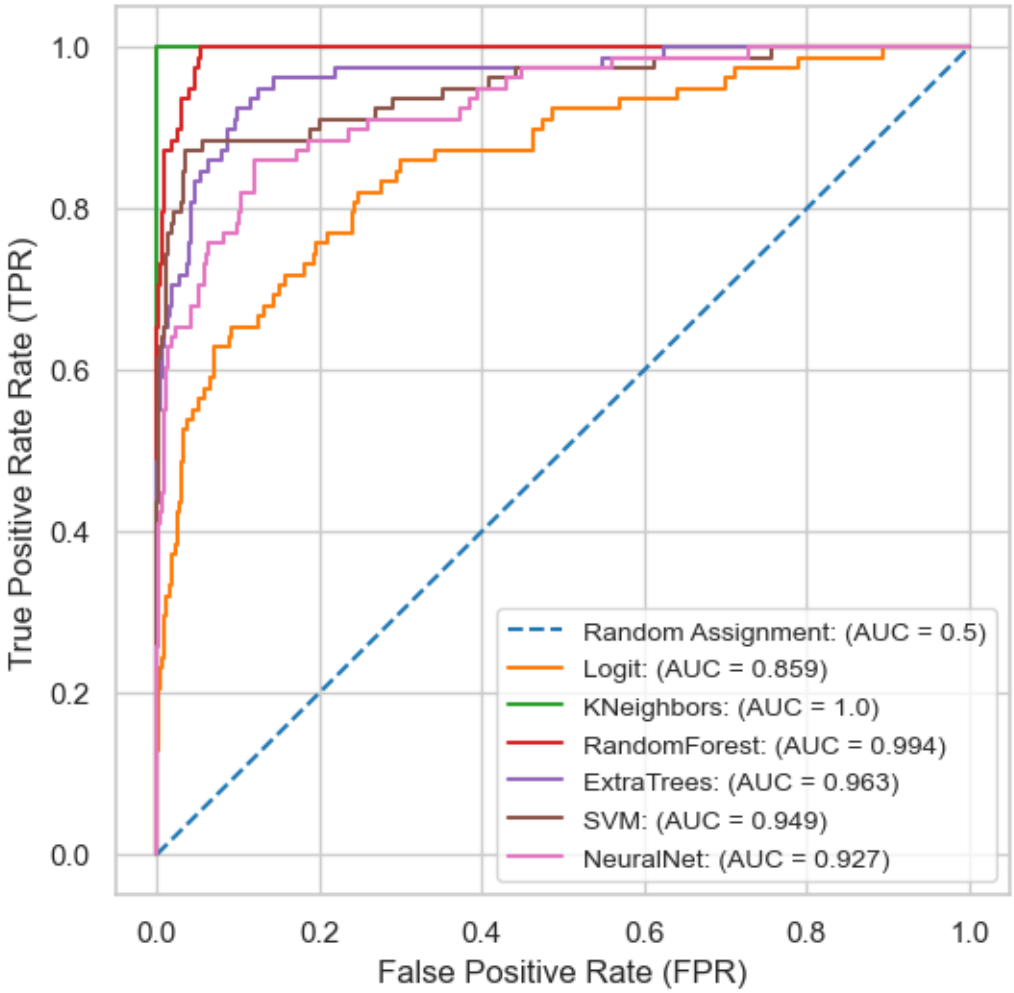
[5]:

	Set	Model	AUC
0	Macro	RandomForest	0.704588
1	Macro	Logit	0.679660
2	Macro	NeuralNet	0.663692
3	Macro	ExtraTrees	0.656247
4	Macro	KNeighbors	0.603511
5	Macro	SVM	0.516354
6	Macro	Random Assignment	0.500000
0	Credit	ExtraTrees	0.825699
1	Credit	RandomForest	0.824423
2	Credit	KNeighbors	0.799671
3	Credit	NeuralNet	0.772014
4	Credit	Logit	0.755909
5	Credit	SVM	0.751870
6	Credit	Random Assignment	0.500000
0	Credit & Asset	ExtraTrees	0.839913
1	Credit & Asset	RandomForest	0.832740

```
2  Credit & Asset          KNeighbors  0.798748
3  Credit & Asset           NeuralNet  0.772401
4  Credit & Asset               Logit  0.763955
5  Credit & Asset                 SVM  0.751551
6  Credit & Asset  Random Assignment  0.500000
0             All          ExtraTrees  0.828041
1             All        RandomForest  0.816599
2             All          KNeighbors  0.795852
3             All               Logit  0.785161
4             All           NeuralNet  0.781879
5             All                 SVM  0.769958
6             All  Random Assignment  0.500000
```

[6]:
```
ex_all.rocGraph()
ex_allIS.rocGraph()
```

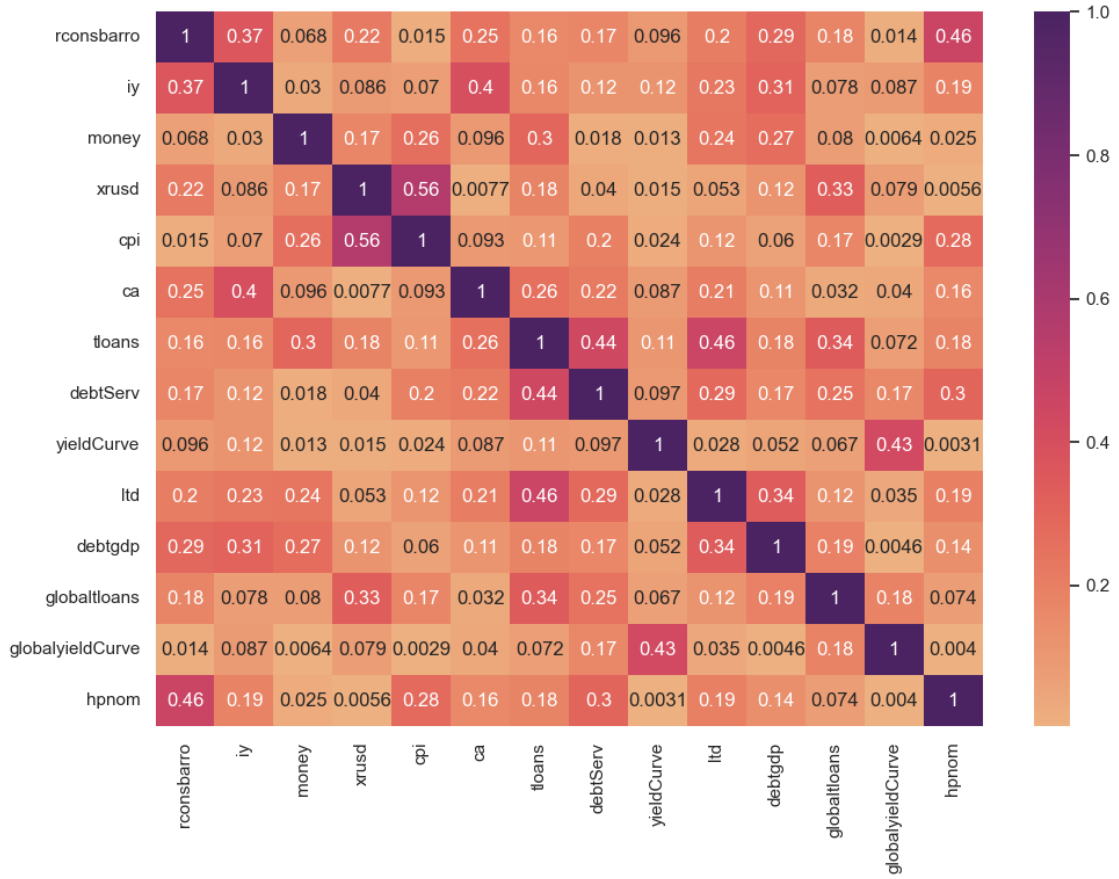




1.5 Explainability

Compute logistic regression coefficients and AEL for best performing black-blox model (Random Forest / Extremely Randomized Trees).

[7]:
```
df_all.correlationMatrix()
```



[8]:
```
df_all.vif()
```

[8]:

	variable	VIF
0	rconsbarro	3.139997
13	hpnom	2.501668
6	tloans	2.482708
12	globalyieldCurve	2.288070
4	cpi	2.273961
9	ltd	1.839108
2	money	1.764422
11	globaltloans	1.743562
3	xrusd	1.732725
8	yieldCurve	1.540389
1	iy	1.459956
7	debtServ	1.444777
10	debtgdp	1.355034
5	ca	1.302359

[9]:
```
ex_all.logitCoef()
```

```
Optimization terminated successfully.
         Current function value: 0.179190
         Iterations 8
const is significant at 1%
rconsbarro is significant at 1%
money is significant at 5%
xrusd is significant at 5%
cpi is significant at 1%
debtServ is significant at 5%
yieldCurve is significant at 1%
ltd is significant at 1%
globaltloans is significant at 1%
globalyieldCurve is significant at 1%
hpnom is significant at 10%
```

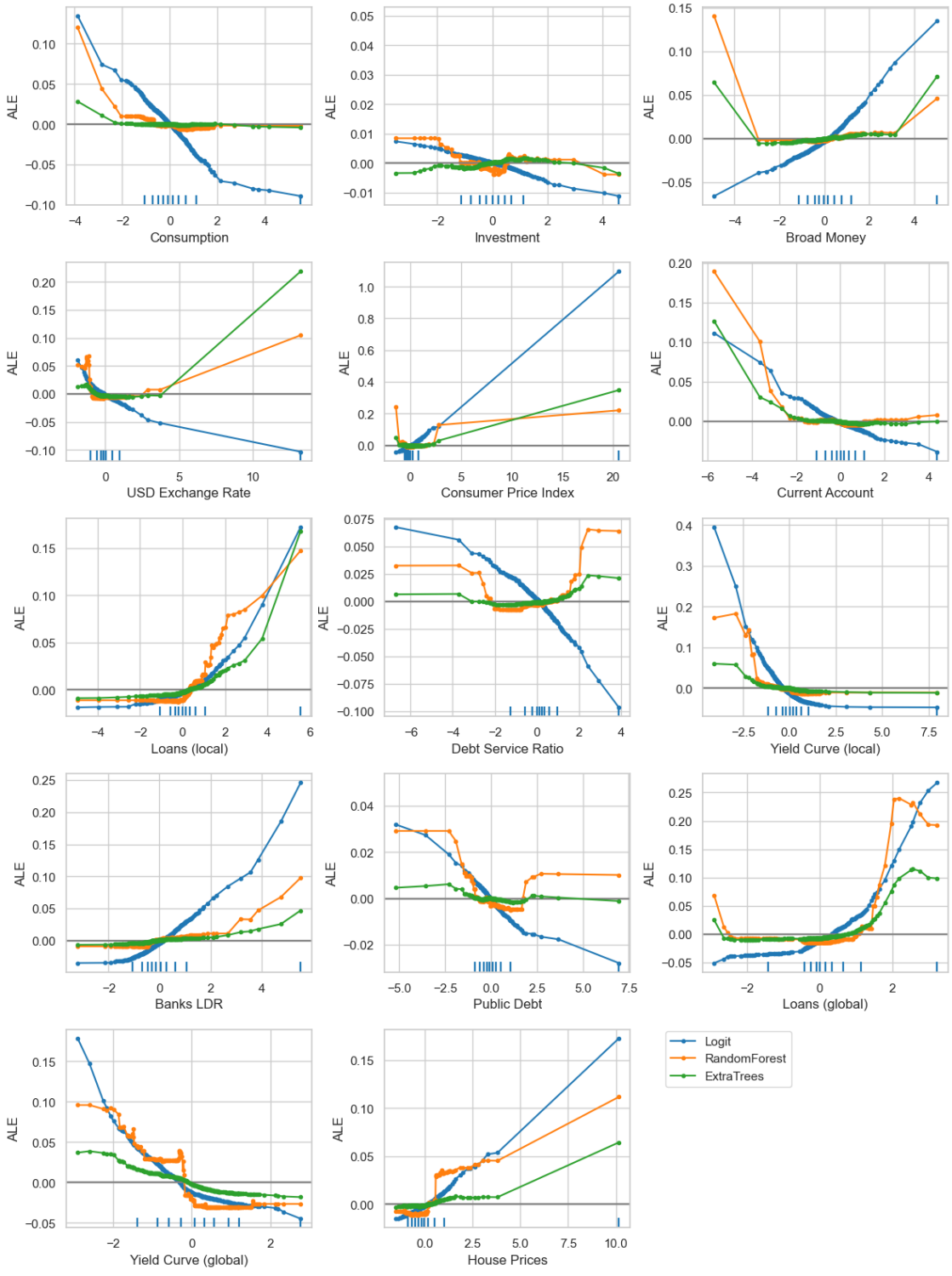
[9]:

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
const	-3.697644	0.228878	-16.155525	1.038316e-58	-4.146237	-3.249052
rconsbarro	-0.629918	0.191872	-3.283011	1.027045e-03	-1.005980	-0.253856
iy	-0.051658	0.169415	-0.304922	7.604257e-01	-0.383706	0.280389
money	0.389246	0.174602	2.229332	2.579181e-02	0.047032	0.731459
xrusd	-0.410281	0.191527	-2.142155	3.218099e-02	-0.785668	-0.034895
cpi	0.748978	0.171188	4.375175	1.213349e-05	0.413456	1.084501
ca	-0.258403	0.158231	-1.633073	1.024535e-01	-0.568530	0.051724
tloans	0.223606	0.180472	1.239010	2.153419e-01	-0.130112	0.577324
debtServ	-0.401652	0.176307	-2.278137	2.271840e-02	-0.747208	-0.056096
yieldCurve	-0.787079	0.176864	-4.450203	8.578928e-06	-1.133725	-0.440433
ltd	0.531080	0.173788	3.055914	2.243756e-03	0.190462	0.871697
debtgdp	-0.170815	0.192149	-0.888970	3.740192e-01	-0.547421	0.205791
globaltloans	0.756404	0.172335	4.389149	1.137953e-05	0.418633	1.094174
globalyieldCurve	-0.583369	0.160648	-3.631355	2.819367e-04	-0.898233	-0.268505
hpnom	0.236903	0.121942	1.942752	5.204617e-02	-0.002099	

```
hpnom                0.475905
```

[10]:
```
ex_allIS.ALE(["Logit", "RandomForest", "ExtraTrees"], range(0,14))
```

```
LogisticRegression(max_iter=1000, penalty='none', random_state=1)
AUC: 0.8586610522094392
RandomForestClassifier(max_depth=6, n_estimators=1000, random_state=1)
AUC: 0.993946913301752
ExtraTreesClassifier(max_depth=6, n_estimators=1000, random_state=1)
AUC: 0.9631425921748502
```


ALE
Consumption
Investment
Broad Money
USD Exchange Rate
Consumer Price Index
Current Account
Loans (local)
Debt Service Ratio
Yield Curve (local)
Banks LDR
Public Debt
Loans (global)
Yield Curve (global)
House Prices
Logit
RandomForest
ExtraTrees

1.6 Robustness Checks

[7]:
```
# Robustness Check: ESRB Crisis Data
df_alt1 = Data(iv_all, crisisData = "ESRB").getReady("ESRB")
ex_alt1 = Experiment(df_alt1, models, "CrossVal")
ex_alt1.run(n = 100)
ex_alt1.auc
```

```
ESRB: The final dataset contains 426 observations with 22 distinct crisis
events.

ESRB: Random Assignment:   0%|          | 0/100 [00:00<?, ?it/s]

ESRB: Logit:   0%|          | 0/100 [00:00<?, ?it/s]

ESRB: KNeighbors:   0%|          | 0/100 [00:00<?, ?it/s]

ESRB: RandomForest:   0%|          | 0/100 [00:00<?, ?it/s]

ESRB: ExtraTrees:   0%|          | 0/100 [00:00<?, ?it/s]

ESRB: SVM:   0%|          | 0/100 [00:00<?, ?it/s]

ESRB: NeuralNet:   0%|          | 0/100 [00:00<?, ?it/s]
```

[7]:

	Set	Model	AUC
0	ESRB	RandomForest	0.896315
1	ESRB	ExtraTrees	0.867543
2	ESRB	KNeighbors	0.798209
3	ESRB	Logit	0.770237
4	ESRB	SVM	0.764440
5	ESRB	NeuralNet	0.743398
6	ESRB	Random Assignment	0.500000

[8]:
```
# Robustness Check: Laeven & Valencia Crisis Data
df_alt1 = Data(iv_all, crisisData = "LaevenValencia").getReady("LaevenValencia")
ex_alt1 = Experiment(df_alt1, models, "CrossVal")
ex_alt1.run(n = 100)
ex_alt1.auc
```

```
Yugoslavia, SFR not found in regex

LaevenValencia: The final dataset contains 674 observations with 18 distinct
crisis events.

LaevenValencia: Random Assignment:   0%|          | 0/100 [00:00<?, ?it/s]

LaevenValencia: Logit:   0%|          | 0/100 [00:00<?, ?it/s]

LaevenValencia: KNeighbors:   0%|          | 0/100 [00:00<?, ?it/s]

LaevenValencia: RandomForest:   0%|          | 0/100 [00:00<?, ?it/s]

LaevenValencia: ExtraTrees:   0%|          | 0/100 [00:00<?, ?it/s]

LaevenValencia: SVM:   0%|          | 0/100 [00:00<?, ?it/s]
```

```
LaevenValencia: NeuralNet:   0%|          | 0/100 [00:00<?, ?it/s]
```

[8]:

	Set	Model	AUC
0	LaevenValencia	KNeighbors	0.867680
1	LaevenValencia	ExtraTrees	0.867093
2	LaevenValencia	NeuralNet	0.863735
3	LaevenValencia	SVM	0.853616
4	LaevenValencia	RandomForest	0.851058
5	LaevenValencia	Logit	0.837715
6	LaevenValencia	Random Assignment	0.500000

[ ]: