RDR for Explainable AI

By Nathan Driscoll (z5204935)

Supervisor: Dr Mike Bain

Assessor: Prof Paul Compton

Outline

- Explainable AI
- Ripple Down Rules
- Project outline
  - Motivations for the project
  - System Architecture diagram
- Demo
- Decision Path and feature importances
- Evaluation
- Further work

Explainable AI

- Explains the predictions of ML models
- Accurate ML models are “Black boxes” - often cannot be understood by humans
  - Example application – lending finance
- But “black box” classifiers must be explainable on demand for legal or other reasons
- Explainable AI attempts to solve this problem by creating an interpretable model which explains a ML model
  - This project uses rules in the form of RDR to create an explainable model

Ripple Down Rules (RDR)

- Ripple Down Rules (RDR) is a knowledge acquisition system which involves incrementally building up the knowledge in the system
- Knowledge is built up as a set of rules
  - A rule is added when the system does not deal with a case correctly
  - System built incrementally while it is in use
- Different types of RDR: Single classification (SCRDR) is used for this project

Project Proposal: RDR for Explainable AI

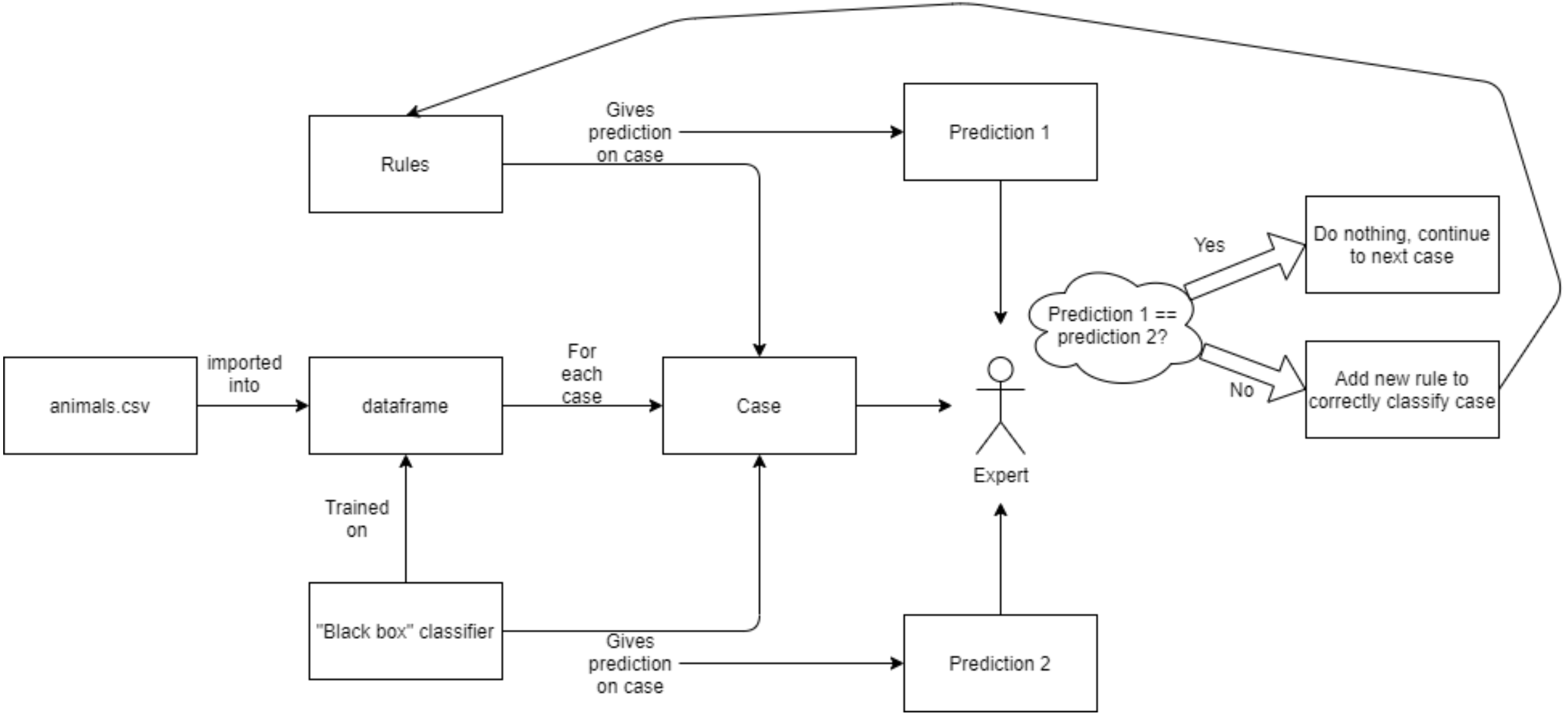
- **Hypothesis:** RDR can be used in place of LIME to explain predictions of a Black Box model
  - Build an explainability model using RDR using an expert human-in-the-loop
  - Rules created as part of RDR system form explainability model

1. Proof of concept: create SCRDR system that works along side “black box” classifier. Firstly a pseudo “black box” classifier (decision trees) is used with known data (animal dataset)
2. Thesis C: get it working with actual “black box” (SVM, XGBoost) and other datasets

Motivations of RDR for Explainable AI

1. RDR captures an expert humans understanding
   - Allows incremental construction of explainability model
2. Print out of rules forms explainable model
   - Though after a certain level of complexity the rules are hard to understand and therefore -> not explainable anymore
3. It is a “no code” way of building Explainable AI
   - Allows programmers and non-programmers to create the rules

Implementation - system architecture



Demo

- Talk about decision tree (pseudo black box) -> actual black box

Full rule set that correctly classifies all cases

Number	Rule	Conclusion	TRUEBranch	FALSEBranch	Case	true_cases
1	`milk` == 1	mammal	mammal	2	aardvark	[]
2	`acquatic` == 1	fish	5	3	bass	[]
3	`feathers` == 1	bird	bird	4	chicken	[]
4	`backbone` == 0	mollusc	7	9	clam	[]
5	`backbone` == 0	mollusc	mollusc	6	crab	[2]
6	`feathers` == 1	bird	bird	8	duck	[2]
7	`no of legs` > 4	insect	10	mollusc	flea	[4]
8	`no of legs` > 0	amphibian	amphibian	12	frog	[2]
9	`eggs` == 1	reptile	13	None	pitviper	[]
10	`predator` == 1	mollusc	11	insect	scorpion	[4, 7]
11	`venomous` == 0	insect	insect	mollusc	ladybird	[4, 7, 10]
12	`fins` == 0	mollusc	mollusc	fish	seasnake	[2]
13	`no of legs` > 0	mollusc	mollusc	reptile	tortoise	[9]

Hints - Decision path and feature importances

- A user may need help in deciding what features to make rules on
  - Eg datasets containing features X1, X2, X3, … , X300, …
- Tools give hints on what features are important
  - Decision tree - decision path
  - Random forest – feature importances
  - XGBoost – feature importances

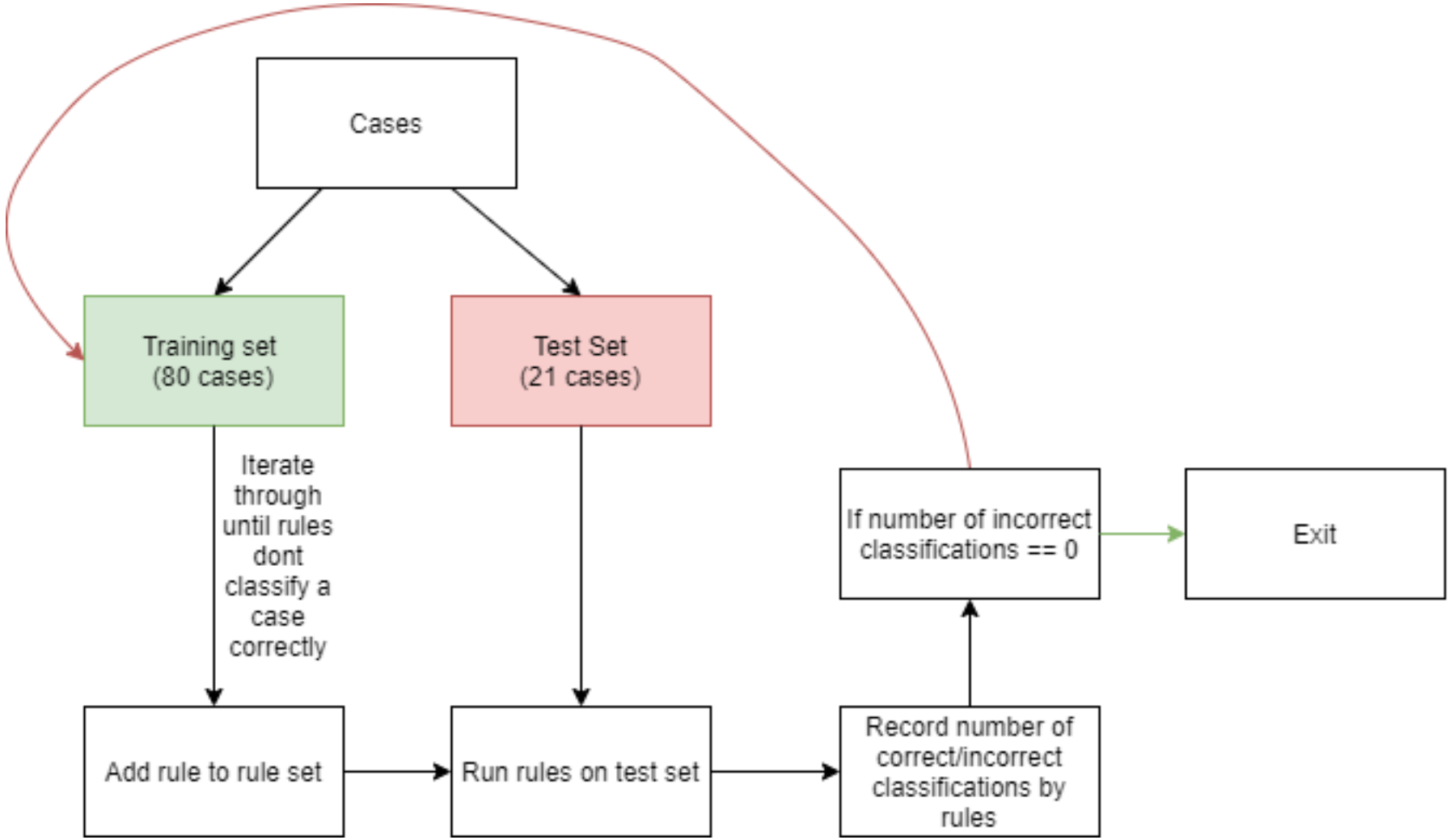
Evaluation – Number of rules

- There are 9 decision nodes in the decision tree

```
Total number of nodes in tree: 19
Number of leaf nodes in tree: 10
Number of decision nodes in tree: 9
```

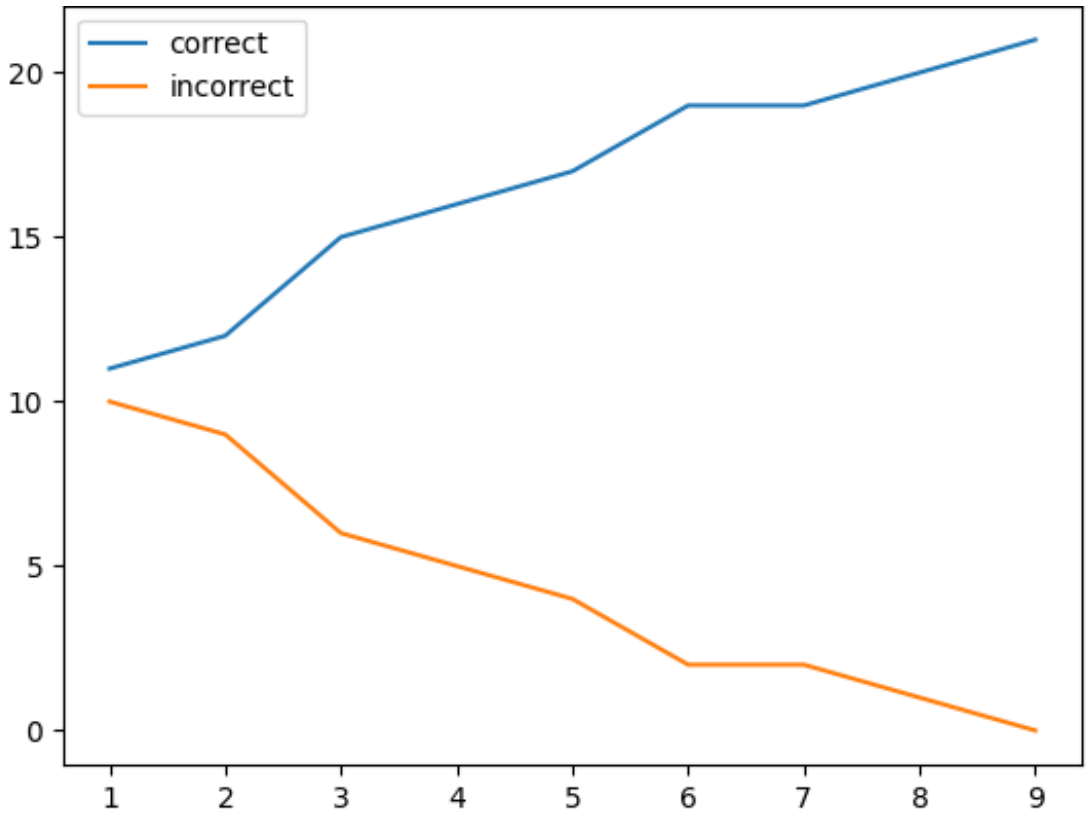
- The ruleset that I created has 13 rules

Evaluation


Cases
Training set
(80 cases)
Test Set
(21 cases)
Iterate through until rules dont classify a case correctly
If number of incorrect classifications == 0
Exit
Add rule to rule set
Run rules on test set
Record number of correct/incorrect classifications by rules

Evaluation - effect of adding rules

Number of rules correct/incorrect



Number of rules added

Further work

- Use new datasets
- Create a frontend for the system
- Integration of LIME
- Further evaluation – attempt to evaluate how explainable rules are
- Possible extension – train ML model, wrap RDR system around. Use it to lift accuracy by dealing with cases that ML doesn’t deal with correctly

Thank you ☺