8 Supplementary Material

We applied the code for the weight priority only for the adult dataset in the FOLD-SE case. For the rest of the cases, we assumed an equal weight of 1 for all features.

8.1 Dataset: Adult; Algorithm: FOLD-SE

Negative Decision: $\leq 50K$
Features and Feature Values used:

- Feature: marital_status
  1. married_civ_spouse
  2. never_married
- Feature: relationship
  1. husband
  2. wife
  3. unmarried
- Feature: sex
  1. male
  2. female
- capital_gain: [0, 99999]
- education_num: [1, 16]
- age: [17, 90]

Decision Rules We run the FOLD-SE algorithm to produce the following decision making rules:

```
label(X,'<=50K') :-
   not marital_status(X,'Married-civ-spouse'),
   capital_gain(X,N1), N1=<6849.0.
label(X,'<=50K') :-
   marital_status(X,'Married-civ-spouse'),
   capital_gain(X,N1), N1=<5013.0,
   education_num(X,N2), N2=<12.0.
```

- Accuracy: 84.5%
- Precision: 86.5%
- Recall: 94.6%

Causal Rules a) FOLD-SE gives Causal rules for the 'marital_status' feature having value 'never_married':

```
marital_status(X,'Never-married'):-
   not relationship(X,'Husband'),
   not relationship(X,'Wife'),
   age(X,N1), N1=<29.0.
```

- Accuracy: 86.4%
- Precision: 89.2%
- Recall: 76.4%

b) FOLD-SE gives Causal rules for the 'marital_status' feature having value 'Married-civ-spouse':

```
marital_status(X,'Married-civ-spouse'):-
   relationship(X,'Husband').
marital_status(X,'Married-civ-spouse'):-
   relationship(X,'Wife').
```

- Accuracy: 99.1%
- Precision: 99.9%
- Recall: 98.2%

c) For values of the feature 'marital_status' that are not 'Married-civ-spouse' or 'never_married' which we shall call 'neither', a user defined rule is used

```
marital_status(X,neither):-
   not relationship(X,'Husband'),
   not relationship(X,'Wife').
```

d) FOLD-SE gives Causal rules for the 'relationship' feature having value 'husband':

```
relationship(X,'Husband'):-
   not sex(X,'Male'),age(X,N1),not(N1=<27.0).
```

- Accuracy: 82.3%
- Precision: 71.3%
- Recall: 93.2%

e) For the 'relationship' feature value of 'wife', a user defined rule is used

```
relationship(X,'Wife') :- sex(X,'Female').
```

8.2 Dataset: Adult; Algorithm: RIPPER

Negative Decision: $\leq 50K$

Features and Feature Values used:

- Feature: marital_status
  1. married_civ_spouse
  2. never_married
  3. divorced
- Feature: relationship
  1. husband
  2. wife
  3. own_child
  4. not_in_family
  5. unmarried

- Feature: education
  1. hs_grad
  2. some_college
- Feature: occupation
  1. farming_fishing
  2. adm_clerical
  3. machine_op_inspct
  4. other_service
- Feature: workclass
  1. never_worked
  2. private
- Feature: native_country
  1. japan
  2. united_States
- Feature: sex
  1. male
  2. female
- capital_gain: [0, 99999]
- education_num: [1, 16]
- age: [17, 90]
- hours_per_week: [1, 99]
- capital_loss: [0, 4356]

Decision Rules We run the RIPPER algorithm to produce the following decision making rules:

```
[[marital_status=never_married ∧
   relationship=own_child ∧ age=<22.0.]V [marital_status=never_married
∧ capital_gain =< 9999.9. ]V [relationship=not_in_family ∧ capital_gain
=< 9999.9 ∧ education_num >= 7.0, education_num =< 9.0 ∧ hours_per_week
>= 35.0, hours_per_week =< 40.0 ∧ age >= 26.0, age =< 30.0.]V [relationship=not_in_family
∧ capital_gain =< 9999.9 ∧ education_num >= 7.0, education_num =< 9.0
∧ sex=Female.]V [relationship=not_in_family ∧ capital_gain =< 9999.9
∧ education=some_college ∧ sex=Female ∧ occupation=adm_clerical.]V
[relationship=not_in_family ∧ capital_gain =< 9999.9 ∧ hours_per_week
>= 35.0, hours_per_week =< 40.0 ∧ age >= 22.0, age =< 26.0.]V [relationship=not_in_family
∧ capital_gain =< 9999.9 ∧ education_num >= 7.0, education_num =< 9.0
∧ workclass=private ∧ occupation=machine_op_inspct.]V [relationship=not_in_family
∧ capital_gain =< 9999.9 ∧ education_num =< 7.0 ∧ sex=Female.]V [relationship=not_in_family
∧ capital_gain =< 9999.9 ∧ education=some_college ∧ occupation=other_service.]V
[relationship=unmarried.]V [relationship=not_in_family ∧ capital_gain
=< 9999.9 ∧ hours_per_week >= 35.0, hours_per_week =< 40.0 ∧ education_num
>= 7.0, education_num =< 9.0.]V [relationship=not_in_family ∧ capital_gain
=< 9999.9 ∧ age >= 26.0, age =< 30.0 ∧ hours_per_week >= 35.0, hours_per_week
=< 40.0.]V [education_num =< 7.0.]V [relationship=not_in_family ∧ capital_gain
=< 9999.9 ∧ hours_per_week >= 25.0, hours_per_week =< 35.0 ∧ workclass=private
```

```
∧ sex=Female. ]∨ [relationship=not_in_family ∧ capital_gain =< 9999.9
∧ hours_per_week =< 25.0 ∧ capital_loss =< 435.6 ∧ native_country=united_States
∧ workclass=private.]∨ [marital_status=divorced ∧ capital_gain =< 9999.9
∧ hours_per_week >= 35.0, hours_per_week =< 40.0 ∧ education=some_college.]∨
[education_num >= 7.0, education_num =< 9.0 ∧ marital_status=divorced
∧ relationship=own_child.]∨ [education_num >= 7.0, education_num =<
9.0 ∧ occupation=other_service ∧ age >= 37.0, age =< 41.0.]∨[education_num
>= 7.0, education_num =< 9.0 ∧ age >= 26.0, age =< 30.0.
```

- Accuracy: 72.42%
- Precision: 94.33%
- Recall: 67.74%

Causal Rules Due to the low precision and recall for causal rules we obtain, we use the causal rules of FOLD-SE as described in Section 8.1 to denote the causal dependency while using the decision making rule of RIPPER

8.3 Dataset: Titanic; Algorithm: FOLD-SE

Negative Decision: 0 (perished)
Features and Feature Values used:

- Feature: gender
  1. male
  2. female
- Feature: class
  1. 1
  2. 2
  3. 3

Decision Rules We run the FOLD-SE algorithm to produce the following rules:

```
survived(X,'0'):- not sex(X,'female').
survived(X,'0'):-
   class(X,'3'),sex(X,'female'),
   fare(X,N1),not(N1=<23.25).
```

The rules described above indicate if someone perished or not.

- Accuracy: 98.6%
- Precision: 97.8%
- Recall: 100%

8.4 Dataset: Titanic; Algorithm: RIPPER

Negative Decision: 0 (perished)
Features and Feature Values used:

- Feature: gender
  1. male
  2. female

Decision Rules We run the RIPPER algorithm to produce the following rules:
`[[ sex = male ]]`
The rules described above indicate if someone perished or not.

- Accuracy: 87.4%
- Precision: 89.15%
- Recall: 90.79%

8.5 Dataset: Cars; Algorithm: FOLD-SE

We relabel the dataset to 'positive' and 'negative' where 'negative' refers to used cars that are unacceptable for purchase.

Negative Decision: negative (reject/ used car is unacceptable for purchase)

Features and Feature Values used:

- Feature: persons
  1. 2
  2. 4
  3. more
- Feature: safety
  1. low
  2. med
  3. high
- Feature: buying
  1. low
  2. med
  3. high
  4. vhigh
- Feature: maint
  1. low
  2. med
  3. high
  4. vhigh

Decision Rules We run the FOLD-SE algorithm to produce the following rules:

```
label(X,'negative'):- persons(X,'2').
label(X,'negative'):- safety(X,'low').
label(X,'negative'):- buying(X,'vhigh'),
   maint(X,'vhigh').
label(X,'negative'):- not buying(X,'low'),
   not buying(X,'med'), maint(X,'vhigh').
label(X,'negative'):- buying(X,'vhigh'),
   maint(X,'high').
```

The rules described above indicate if the purchase of a car was rejected .

- Accuracy: 93.9%
- Precision: 100%
- Recall: 91.3%

8.6 Dataset: Cars; Algorithm: RIPPER

We relabel the dataset to ‘positive’ and ‘negative’ where ‘negative’ refers to used cars that are unacceptable for purchase.

Negative Decision: negative (reject/ used car is unacceptable for purchase)

Features and Feature Values used:

- Feature: persons
  1. 2
  2. 4
  3. more
- Feature: safety
  1. low
  2. med
  3. high
- Feature: buying
  1. low
  2. med
  3. high
  4. vhigh
- Feature: maint
  1. low
  2. med
  3. high
  4. vhigh
- Feature: lugboot
  1. small
  2. medium
  3. big
- Feature: doors
  1. 2
  2. 3
  3. 4
  4. more

Decision Rules We run the RIPPER algorithm to produce the following rules:

```
[[persons=2] V
[safety=low] V
[buying=vhigh ∧ maint=vhigh] V
[lugboot=small ∧ safety=med ∧
   buying=high] V
[maint=vhigh ∧ buying=high] V
[buying=vhigh ∧ maint=high] V
[lugboot=small ∧ doors=2 ∧ persons=more]
   V
[safety=med ∧ lugboot=small ∧
```

```
   buying=vhigh] V
[safety=med ∧ maint=vhigh ∧
   lugboot=small] V
[safety=med ∧ doors=3 ∧ persons=4 ∧
   lugboot=med] V
[lugboot=small ∧ safety=med ∧
   maint=high ∧ buying=med]]
```

The rules described above indicate if the purchase of a car was rejected .

- Accuracy: 99.13%
- Precision: 99.58%
- Recall: 99.17%

8.7 Dataset: Voting; Algorithm: FOLD-SE

Negative Decision: 'republican'

Features and Feature Values used:

- Feature: physician_fee_freeze
  1. yes
  2. no
- Feature: budget_resolution
  1. yes
  2. no
- Feature: handicapped_infants
  1. yes
  2. no
- Feature: synfuels_corporation_cutback
  1. yes
  2. no
- Feature: mx_missile
  1. yes
  2. no

Decision Rules We run the FOLD-SE algorithm to produce the following rules:

```
label(X,'republican'):-
   physician_fee_freeze(X,'y'),
   not ab2(X,'True').
ab1(X,'True'):- budget_resolution(X,'y'),
   not handicapped_infants(X,'n').
ab2(X,'True'):-
   synfuels_corporation_cutback(X,'y'),
   not mx_missile(X,'n'),not ab1(X,'True').
```

The rules described above indicate if the vote was cast for a Republican.

1. Accuracy: 97.7%
2. Precision: 97%
3. Recall: 97%

8.8 Dataset: Voting; Algorithm: RIPPER

Negative Decision: 'republican'
Features and Feature Values used:

- Feature: physician_fee_freeze
  1. yes
  2. no
- Feature: synfuels_corporation_cutback
  1. yes
  2. no

Decision Rules We run the RIPPER algorithm to produce the following rules:

```
[[physician_fee_freeze=y ∧
   synfuels_corporation_cutback=n] V
[physician_fee_freeze=y]
```

The rules described above indicate if the vote was cast for a Republican.

- Accuracy: 96.5%
- Precision: 94.2%
- Recall: 97.05%

8.9 Dataset: Mushroom; Algorithm: FOLD-SE

Negative Decision: '*p*' (poisonous)
Features and Feature Values used:

- Feature: odor
  1. n
  2. f
- Feature: spore_print_color
  1. r
  2. b
- Feature: bruises
  1. f
  2. t
- Feature: stalk_root
  1. c
  2. r
  3. b
- Feature: gill_spacing
  1. c
  2. w

Decision Rules We run the FOLD-SE algorithm to produce the following rules:

```
label(X,'p'):- not odor(X,'n'),
   not ab1(X,'True'), not ab2(X,'True'),
   not ab3(X,'True').
label(X,'p'):- spore_print_color(X,'r').
ab1(X,'True'):- not bruises(X,'f'),
   stalk_root(X,'c').
ab2(X,'True'):- not bruises(X,'f'),
   stalk_root(X,'r').
ab3(X,'True'):- not gill_spacing(X,'c'),
   not bruises(X,'f').
```

The rules described above indicate if a mushroom is poisonous.

- Accuracy: 99.8%
- Precision: 100%
- Recall: 99.6%

8.10 Dataset: Mushroom; Algorithm: RIPPER

Negative Decision: '*p*' (poisonous)

Features and Feature Values used:

- Feature: odor
  1. f
  2. p
  3. c
- Feature: gill_size
  1. n
  2. b
- Feature: gill_color
  1. n
  2. b
- Feature: spore_print_color
  1. r
  2. b
- Feature: stalk_surface_below_ring
  1. y
  2. k
- Feature: stalk_surface_above_ring
  1. y
  2. k
- Feature: stalk_color_above_ring
  1. y
  2. c
- Feature: habitat
  1. l
  2. g
- Feature: cap_color
  1. e
  2. w

Decision Rules We run the RIPPER algorithm to produce the following rules:

```
[[odor=f] V
[gill-size=n ∧ gill-color=b] V
[gill-size=n ∧ odor=p] V
[odor=c] V
[spore-print-color=r] V
[stalk-surface-below-ring=y ∧
   stalk-surface-above-ring=k] V
[stalk-color-above-ring=y] V
[habitat=l ∧ cap-color=w]]
```

The rules described above indicate if a mushroom is poisonous.

- Accuracy: 100%
- Precision: 100%
- Recall: 100%

8.11 Dataset: Dropout; Algorithm: FOLD-SE

Negative Decision: 'Dropout'
Features and Feature Values used:

- Feature: debtor
  1. 0
  2. 1
- Feature: course
  1. 171
  2. 33
- Feature: curricular_units_2nd_sem_grade - [0, 18.57]
- Feature: admission_grade - [95, 190]

Decision Rules We run the FOLD-SE algorithm to produce the following rules:

```
label(X,'Dropout') :-
   curricular_units_2nd_sem-grade(X,N1),
   N1=<10.667.
label(X,'Dropout') :- not debtor(X,'1').
```

The rules described above indicate if someone is a dropout in college.

- Accuracy: 84%
- Precision: 74.9%
- Recall: 73.8%

8.12 Dataset: Dropout; Algorithm: RIPPER

Negative Decision: 'Dropout'
Features and Feature Values used:

- Feature: tuitionfeesuptodate
  1. 0
  2. 1
- Feature: debtor
  1. 0
  2. 1
- Feature: displaced
  1. 0
  2. 1
- Feature: scholarshipholder
  1. 0
  2. 1
- Feature: curricularunits2ndsem_approved - [0, 20]
- Feature: applicationmode - [1, 57]
- Feature: curricularunits2ndsem_enrolled - [0, 23]
- Feature: curricularunits2ndsem_evaluations - [0, 33]
- Feature: course - [3, 9991]
- Feature: mothersqualification - [1, 44]
- Feature: fathersqualification - [1, 44]
- Feature: curricularunits2ndsem_approved - [0, 20]
- Feature: age_at_enrollment - [17, 70]
- Feature: admissiongrade - [95, 190]
- Feature: mothersoccupation - [0, 194]
- Feature: previousqualification - [95, 190]

8.13 Decision Rules

We run the RIPPER algorithm to produce the following rules:

```
[[Curricularunits2ndsem-approved=<1.0 ∧
   Tuitionfeesuptodate=0 ∧ Debtor=0] V
[Curricularunits2ndsem-approved=<1.0 ∧
   Applicationmode=17.0-39.0] V
[Curricularunits2ndsem-approved=<1.0 ∧
   Curricularunits2ndsem-enrolled=5.0-6.0 ∧
   Curricularunits2ndsem-evaluations=<5.0] V
[Curricularunits2ndsem-approved=<1.0 ∧
   Course=9238.0-9500.0] V
[Curricularunits2ndsem-approved=<1.0 ∧
   Displaced=0 ∧
   Curricularunits2ndsem-enrolled=5.0-6.0 ∧
   Mothersqualification=<3.0] V
[Curricularunits2ndsem-approved=<1.0 ∧
```

```
   Displaced=0 ∧
   Fathersqualification=19.0-37.0 ∧
   Mothersqualification=19.0-37.0] V
[Tuitionfeesuptodate=0 ∧
   Curricularunits2ndsem-approved=1.0-3.0] V
[Curricularunits2ndsem-approved=<1.0 ∧
   Debtor=1 ∧
   Curricularunits2ndsem-evaluations=<5.0] V
[Curricularunits2ndsem-approved=<1.0 ∧ Displaced=0] V
[Curricularunits2ndsem-approved=1.0-3.0 ∧
   Curricularunits1stsem-approved=2.0-4.0 ∧
   Mothersqualification=19.0-37.0 ∧
   Ageatenrollment=>34.2] V
[Tuitionfeesuptodate=0 ∧
   Curricularunits1stsem-approved=2.0-4.0 ∧
   Mothersqualification=<3.0] V
[Tuitionfeesuptodate=0] V
   [Curricularunits2ndsem-approved=1.0-3.0 ∧
   Fathersqualification=19.0-37.0 ∧
   Admissiongrade=138.3-146.22] V
[Curricularunits2ndsem-approved=1.0-3.0 ∧
   Ageatenrollment=27.0-34.2] V
[Curricularunits2ndsem-approved=1.0-3.0 ∧
   Applicationmode=17.0-39.0 ∧
   Mothersoccupation=3.0-4.0] V
[Scholarshipholder=0 ∧
   Curricularunits1stsem-approved=2.0-4.0 ∧
   Curricularunits2ndsem-enrolled=5.0-6.0 ∧
   Previousqualification-grade=130.0-133.1]
```

The rules described above indicate if someone is a dropout in college.

- Accuracy: 84%
- Precision: 74.9%
- Recall: 73.8%