Workshop on Bayesian Methods in Practice

Integrating Decision Analysis and Data Science

John Mark Agosta
San José State University &
Fondata.ai

8 April 2025

SDP Annual Conference

John Mark Agosta

Ex-Microsoft, Azure Machine Learning
Lecturer San Jose State University
Founder Fondata.ai

PhD MS&E

BS Physics

Applied Researcher / Data Scientist
SRI International, Intel, Toyota, Microsoft

University - Industry Research projects
Initiated and led projects with
Stanford, CMU, U. Pittsburgh, Toronto, UC Riverside

Initiatives
Co-founded *Bayes Application Workshop,* 16 years running
Awarded Santa Fe Institute Business Fellowship

Singer
Slavyanka, Bay Area Slavic Chorus

Some familiar terms

- Machine Learning – Statistical estimation from data
  - *ML models produce predictions*
- Decision Analysis – Applied Decision Theory
  - *Decision models prescribe actions.*
- Bayes network and Probabilistic Graphical Models,
  - Directed A-cyclic graphs that factor joint probabilities, for computational purposes
- Influence Diagrams
  - Bayes networks with added decision and value variables.

“Intelligence” is Rational Choice

- A *decision* makes a tangible change; an allocation that is not revocable.
- A *rational decision* aligns actions to maximize a measure over outcomes
- *Outcomes* can be assigned values by which they can be compared
- *Predictions* are uncertainties over outcomes, expressed by probability

These are the 3 kinds of variables:

Outcome values

Conditional probabilities

Decisions, policies

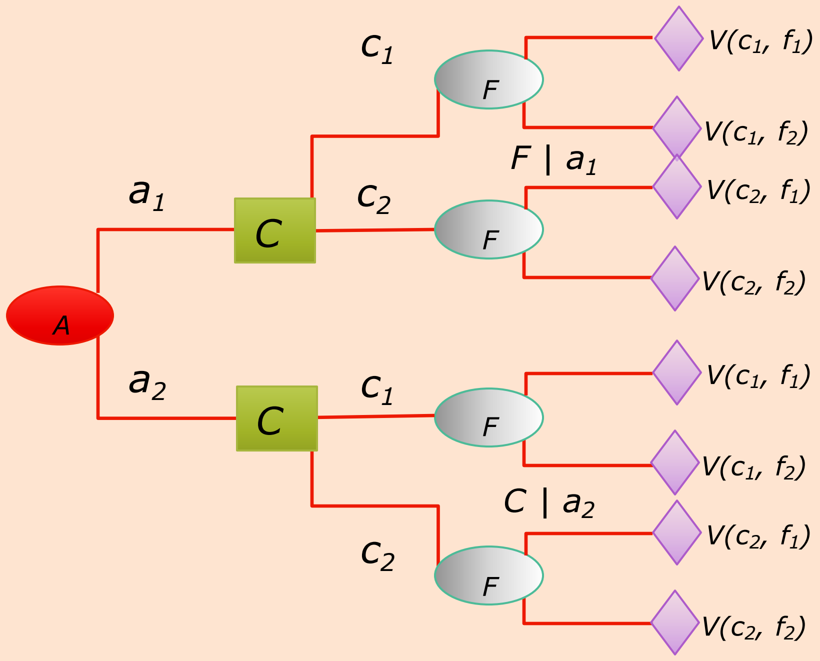
* Influence Diagrams, aka
- Bayes Networks,
- Probabilistic Graphical Models
- Structural Equation Models

have three kinds of variables.

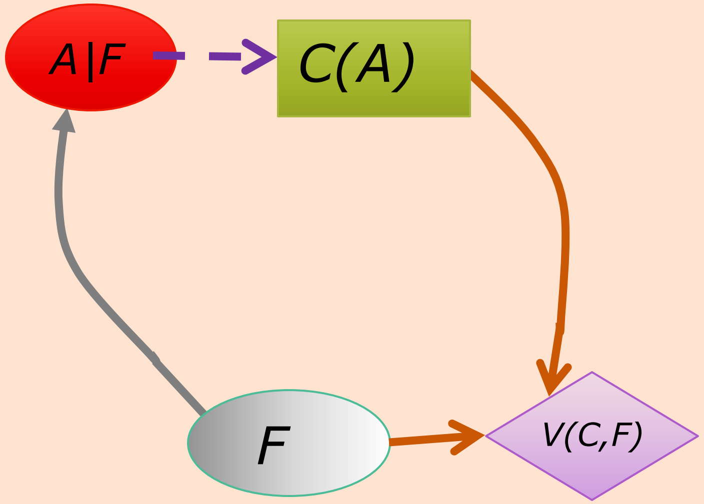
* Arcs show the influences among them

* Decision models, as one might draw out in a tree, can be formulated and solved by Influence Diagrams

Influence Diagrams are concise, causal, and computational



Tree, With Decision and Outcomes



Equivalent Influence Diagram

The marriage proposal: *A coupled influence diagram – statistical model*

- ”Used Car Buyer:” An influence diagram: “test before purchase”
- Informed by a data set of previous test results.
- Estimate a “classification tree” to predict test results from data
- Introduce model priors when learning the tree.
- Use the classification tree to create the influence diagram probability model.

Why?
Because conventional statistic methods don’t work

Why is this hard?

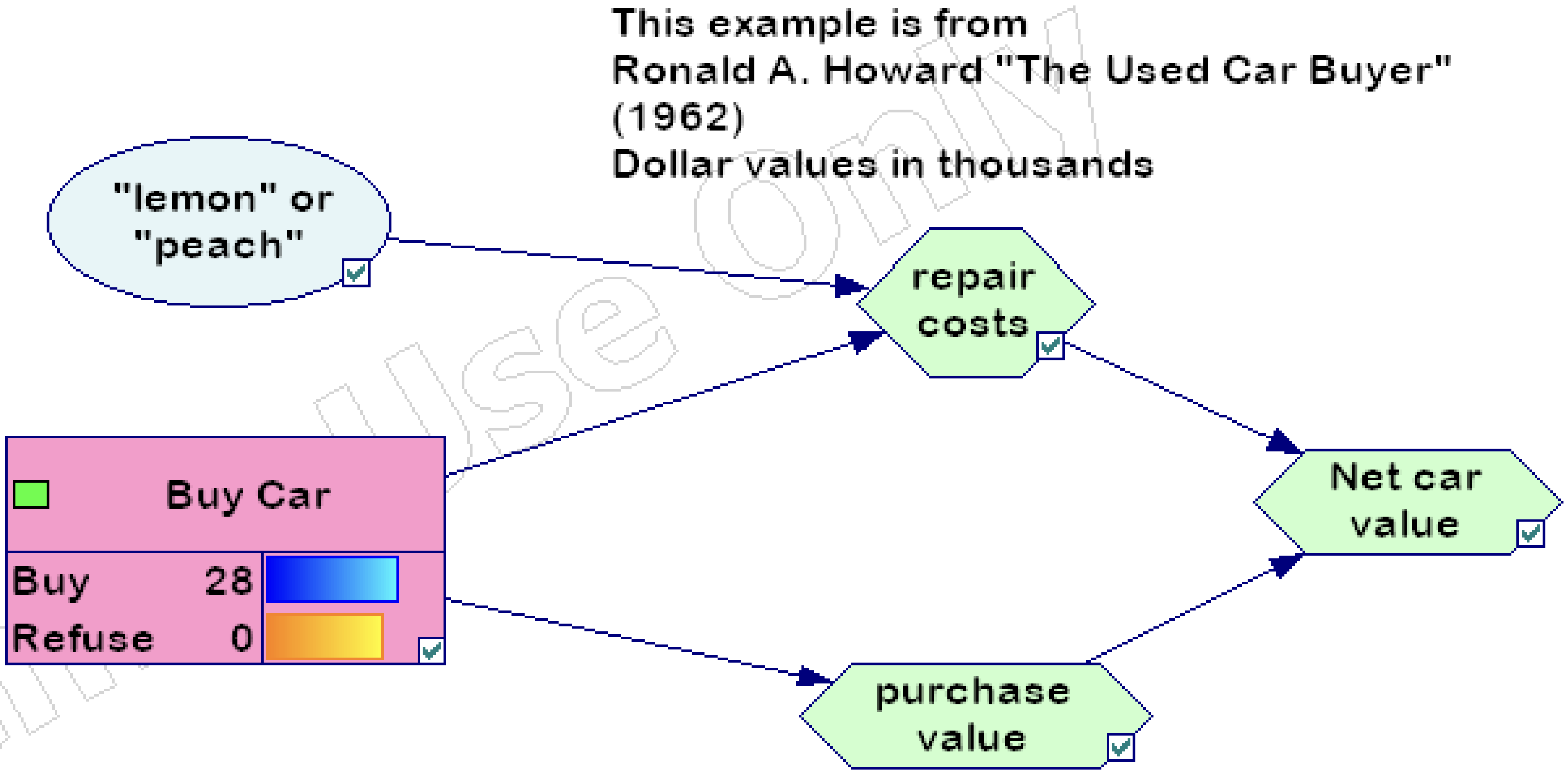
Statistical ”confidence” (p-values) are not the probabilities, needed and don’t work in an influence diagram

How to resolve this?

Creating a Bayesian model from data that can accept priors and obtain the required posterior probability of outcomes

THE USED CAR BUYER

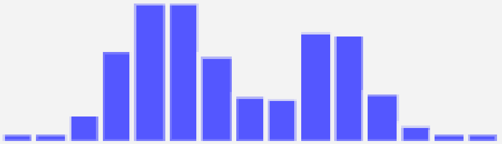
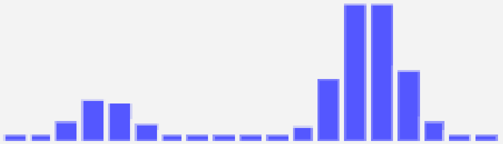
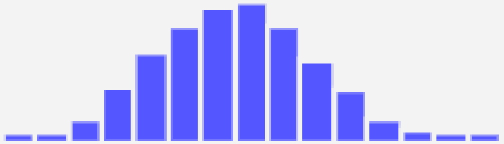
Ronald A. Howard
Department of Engineering-Economic Systems
Stanford University
1962.


This example is from
Ronald A. Howard "The Used Car Buyer"
(1962)
Dollar values in thousands
"lemon" or
"peach"
repair
costs
Buy Car
Buy 28
Refuse 0
Net car
value
purchase
value

Model *Posterior probability*

- P(”lemon” or “peach” | ig, carb, door test)

Simulated data from a history of past test results

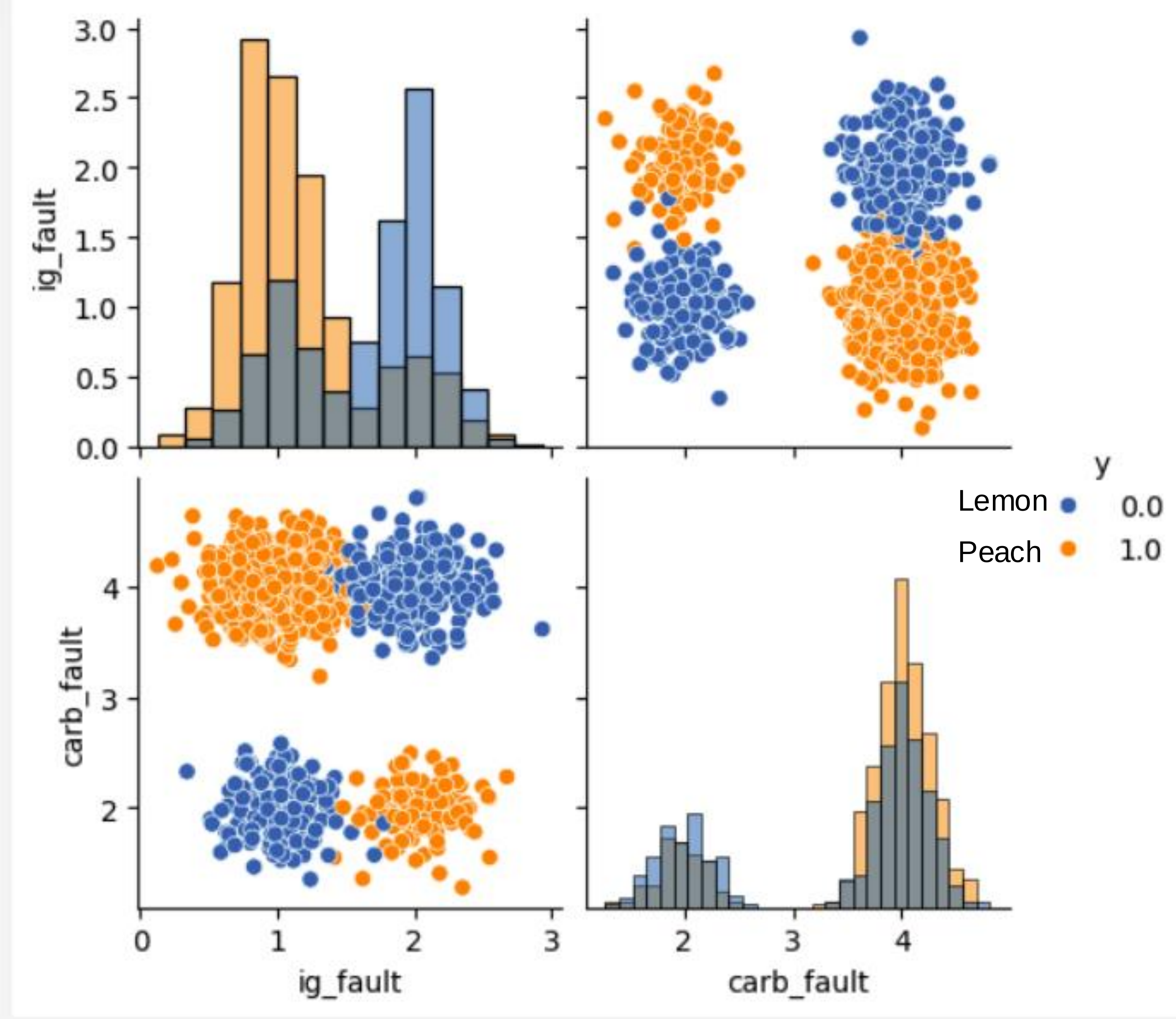
y	ig_fault	carb_fault	door_fault
Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)
False: 446 (45%)	Distinct: 1000 (100%)	Distinct: 1000 (100%)	Distinct: 1000 (100%)
True: 554 (55%)			
			
	Min 0.1277164816... Max 2.929026126...	Min 1.2781676054... Max 4.808618545...	Min 3.3546390533... Max 4.778792381...

	y	ig_fault	carb_fault	door_fault
0	True	0.94323283	3.7379584	4.6375065
1	True	0.9372964	4.3534083	4.076204
2	False	2.0919323	3.5214756	3.6438253
3	True			3.855231
4	True			4.4174094
5	False			4.0036902
6	False			3.7520921
7	True			4.5645795
8	True			3.9556243
9	True	2.106123	2.1948566	4.264723

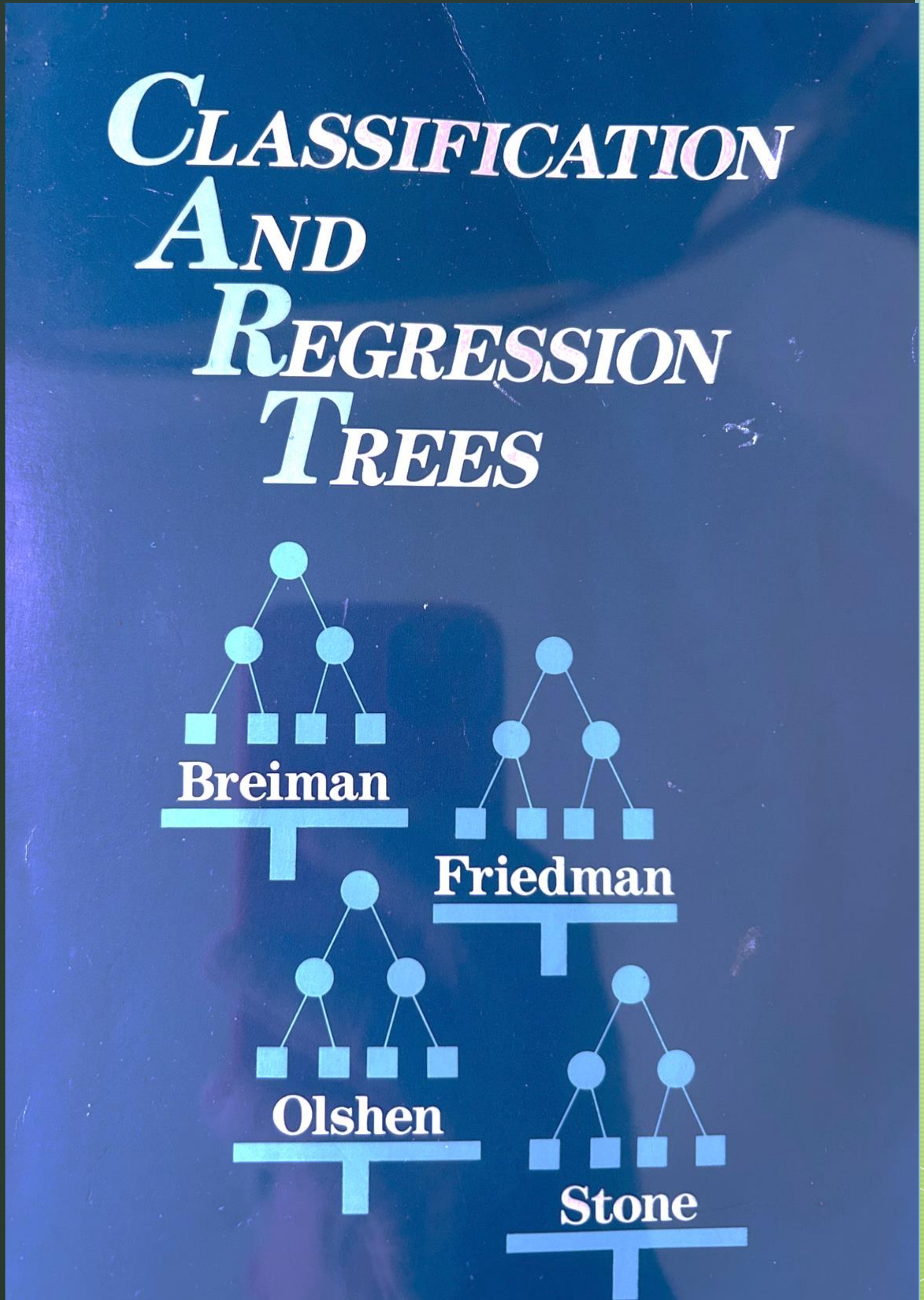
Correlation matrix

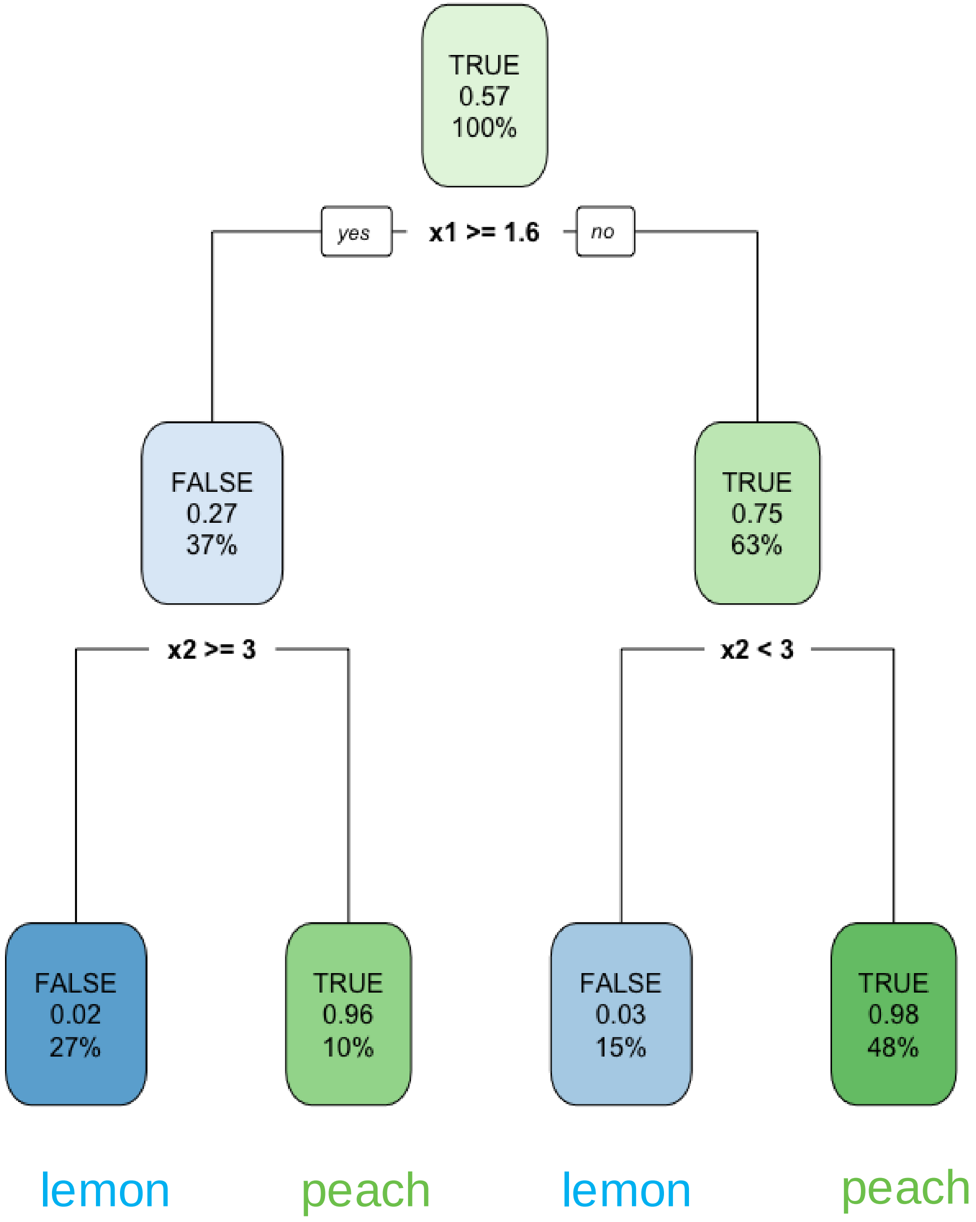
	y	ig_fault	carb_fault	door_fault
y	1.001	-0.450	0.153	0.000
ig_fault	-0.450	1.001	-0.049	0.049
carb_fault	0.153	-0.049	1.001	-0.024
door_fault	0.000	0.049	-0.024	1.001

For faulty cars, measurements “ignition_fault” and “carburetor_fault” tend to coincide



A Classification Tree learns the probability distribution of the outcome from data





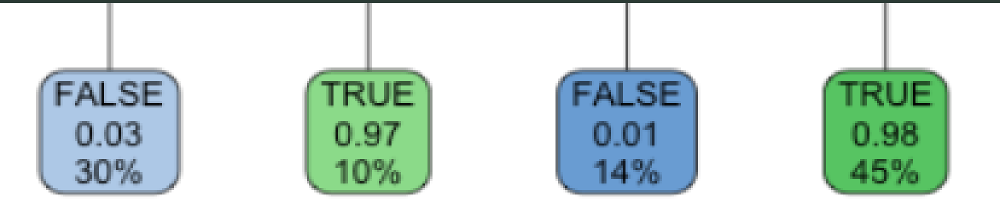
Classification tree for simulated data.

- The splits learned by this tree become the states, (e.g. the partitions) that define the observable variables for the decision node.
- The probabilities learned by the tree are used in the decision model to compute expected value (and hence VOI) for the decision.
- The probabilities learned by the Tree depend on the prior assumed for the unobserved state variable.
- The tree ignores irrelevant variables – you get variable relevance for free.

CART output

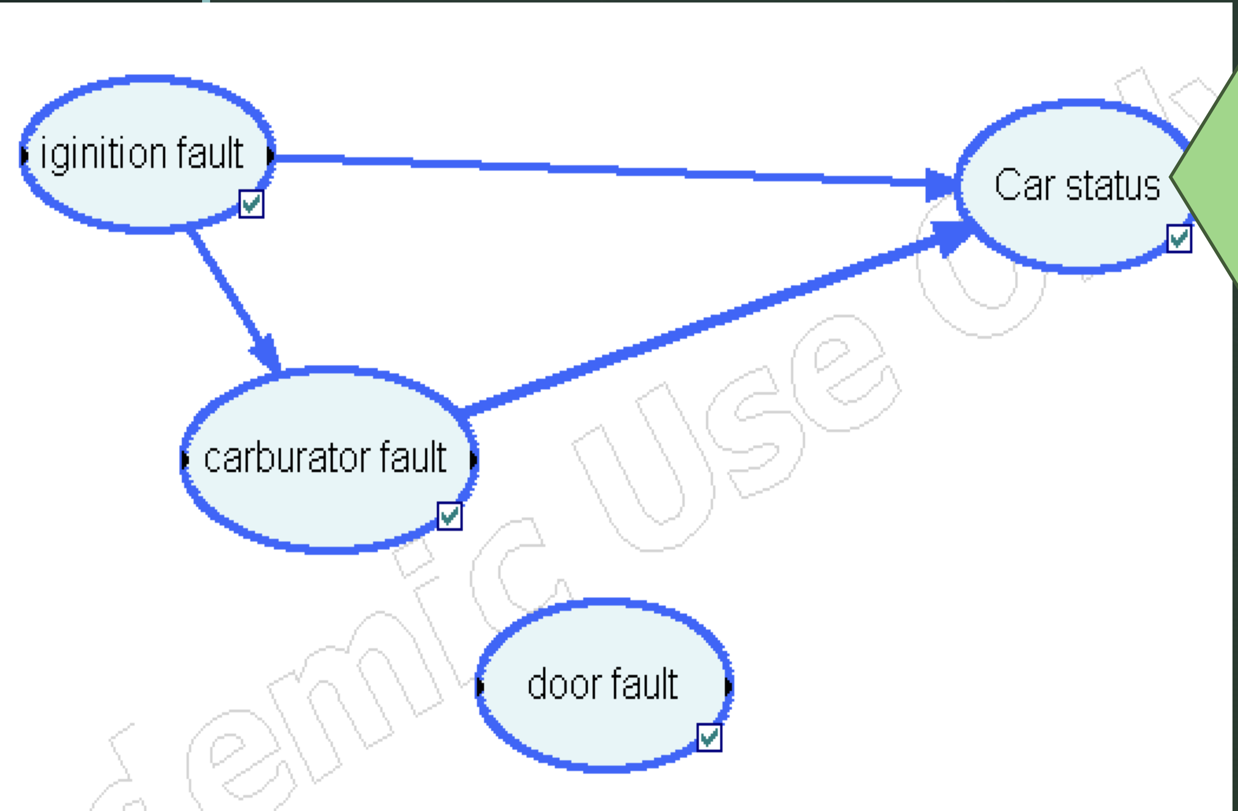
```
Node number 1: 1000 observations,    complexity param=0.432967
  predicted class=TRUE   expected loss=0.455  P(node) =1
    class counts:   455   545
   probabilities: 0.455 0.545
  left son=2 (415 obs) right son=3 (585 obs)
  Primary splits:
      ig_fault   < 1.56532   to the right, improve=113.108700, (0 missing)
      carb_fault < 3.227275  to the left,  improve=  9.451384, (0 missing)
      door_fault < 3.757347  to the left,  improve=  3.923901, (0 missing)
  Surrogate splits:
      carb_fault < 4.435948  to the right, agree=0.594, adj=0.022, (0 split)
      door_fault < 3.333539  to the left,  agree=0.586, adj=0.002, (0 split)

Node number 2: 415 observations,    complexity param=0.221978
  predicted class=FALSE  expected loss=0.2626506  P(node) =0.415
    class counts:   306   109
   probabilities: 0.737 0.263
  left son=4 (310 obs) right son=5 (105 obs)
  Primary splits:
      carb_fault < 2.848778  to the right, improve=145.050600, (0 missing)
      door_fault < 3.783904  to the left,  improve=  3.424465, (0 missing)
      ig_fault   < 1.591579  to the left,  improve=  1.125461, (0 missing)

Node number 3: 585 observations,    complexity param=0.3010989
  predicted class=TRUE   expected loss=0.2547009  P(node) =0.585
    class counts:   149   436
   probabilities: 0.255 0.745
  left son=6 (147 obs) right son=7 (438 obs)
  Primary splits:
      carb_fault < 3.014921  to the left,  improve=198.663000, (0 missing)
      ig_fault   < 1.304715  to the right, improve=  2.376436, (0 missing)
      door_fault < 4.539541  to the right, improve=  0.976095, (0 missing)
  Surrogate splits:
      door_fault < 4.539541  to the right, agree=0.752, adj=0.014, (0 split)
      ig_fault   < 0.4313022 to the left,  agree=0.750, adj=0.007, (0 split)
```

The leaves of the Classification Tree create the *Car Status* conditional probability table.



	ig_fault.carb_fault	not_ig_fault.carb_fault	ig_fault.not_carb_fault	not_ig_fault.not_carb_fault
TRUE	0.02	0.98	0.98	0.03
FALSE	0.98	0.02	0.02	0.97

iginition fault	no_fault_found		found_faulty	
carburator fault	no_fault_fo..	found_faulty	no_fault_fo..	found_faulty
Peach	0.03	0.98	0.98	0.02
Lemon	0.97	0.02	0.02	0.98

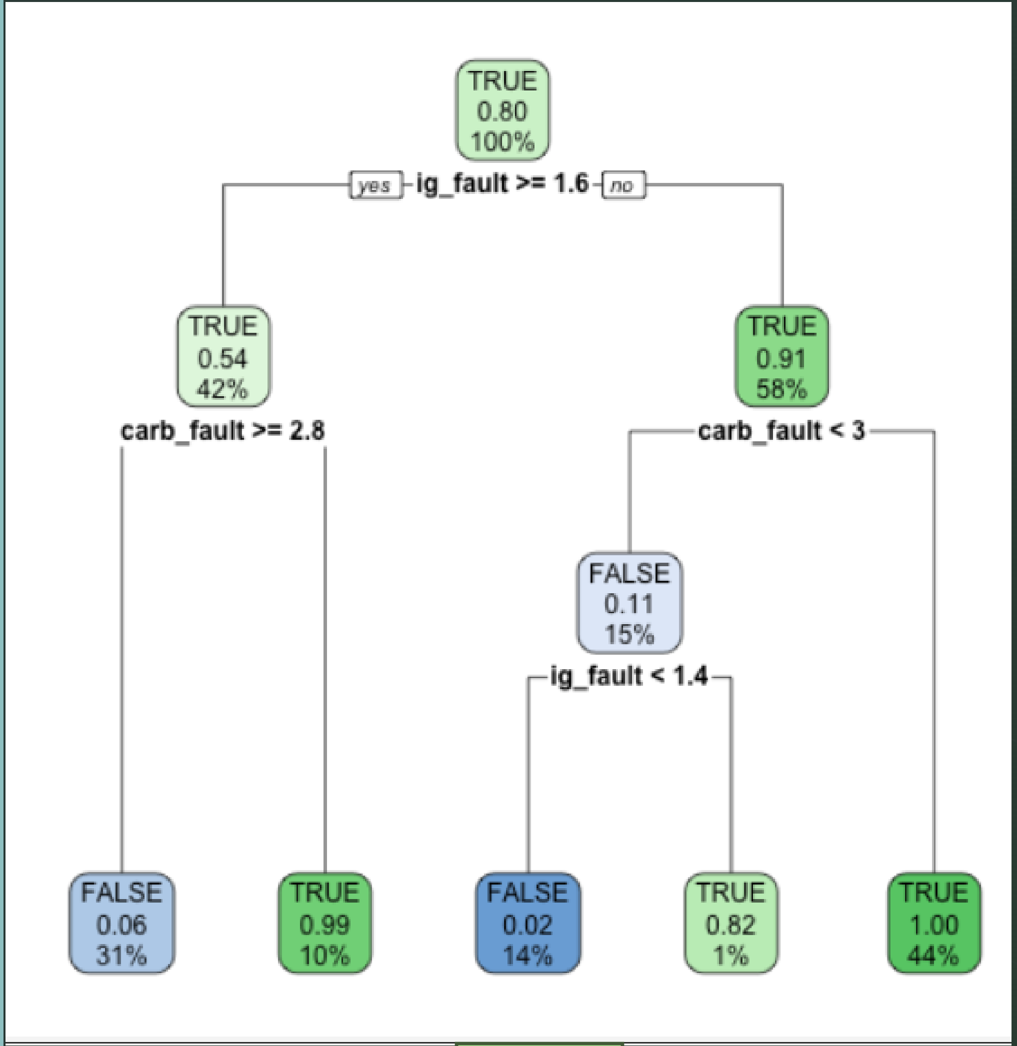


What if the prevalence in the historical training data

does not equal the elicited prior?

$$P(\text{“lemon”} \mid \text{data}) \neq P(\text{“lemon”} \mid \mathcal{E})$$

Not a problem!



, , carb_fault

	ig_fault	not_ig_fault
TRUE	0.06	1
FALSE	0.94	0

, , not_carb_fault

	ig_fault	not_ig_fault
TRUE	0.99	0.02
FALSE	0.01	0.98

Updating the class prior

Using the car “lemon-peach” prior from the influence diagram when re-training the test classifier modifies the prediction probabilities, incrementally improving its performance.

Adjusting the priors

Bayes to the rescue!

Execute Cell (^Enter)

```
# Apply the adjustment to the joint
p = joint_p * adjustment.expand(2,2,2)
# Check the new prior
new_joint_p.sum(axis=(0,1))
```

[137] ✓ 0.0s

```
tensor([0.2000, 0.8000])
```

```
new_joint_p
```

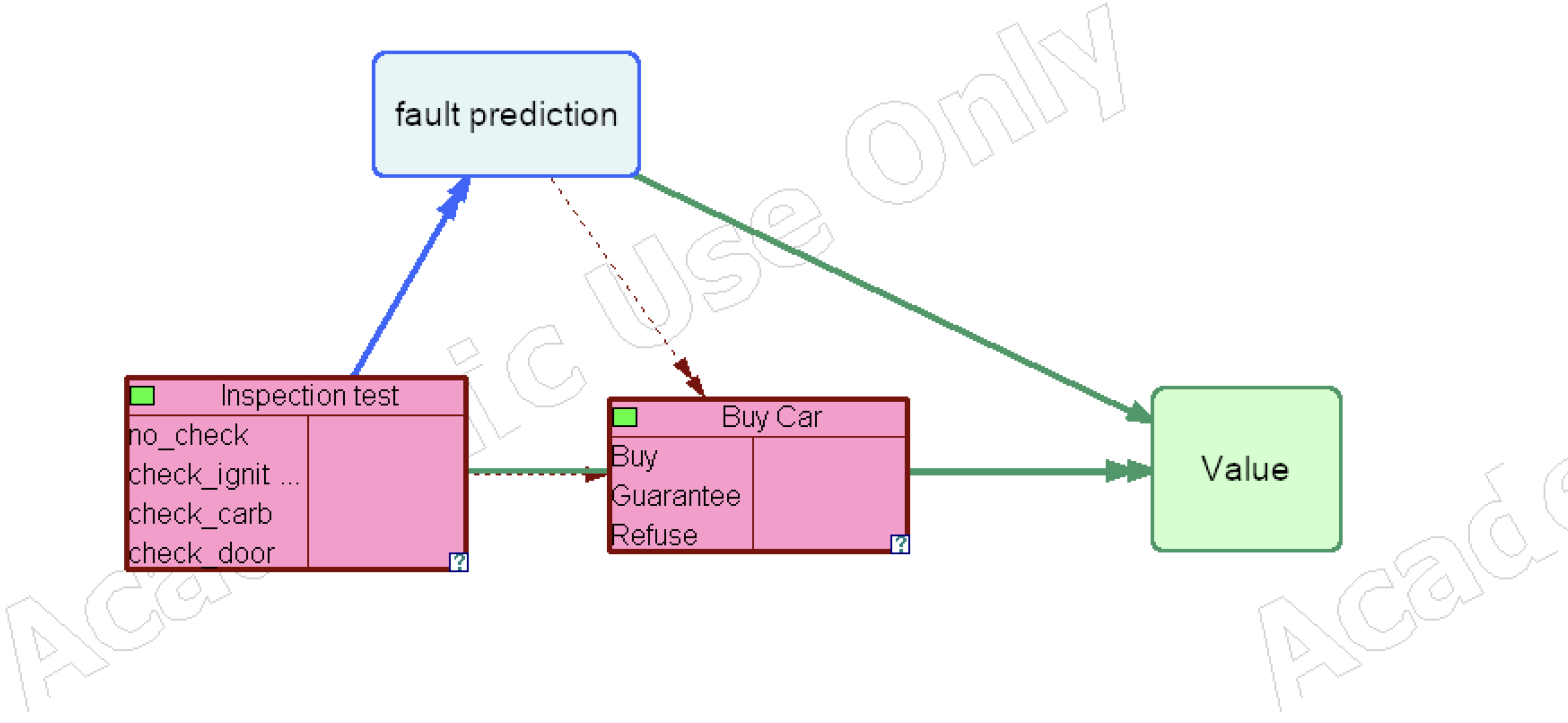
[116] ✓ 0.0s Open 'new_joint_p' in Data Wrangler

```
tensor([[[0.1336, 0.0088],
         [0.0009, 0.1512]],

        [[0.0031, 0.6327],
         [0.0624, 0.0073]]])
```

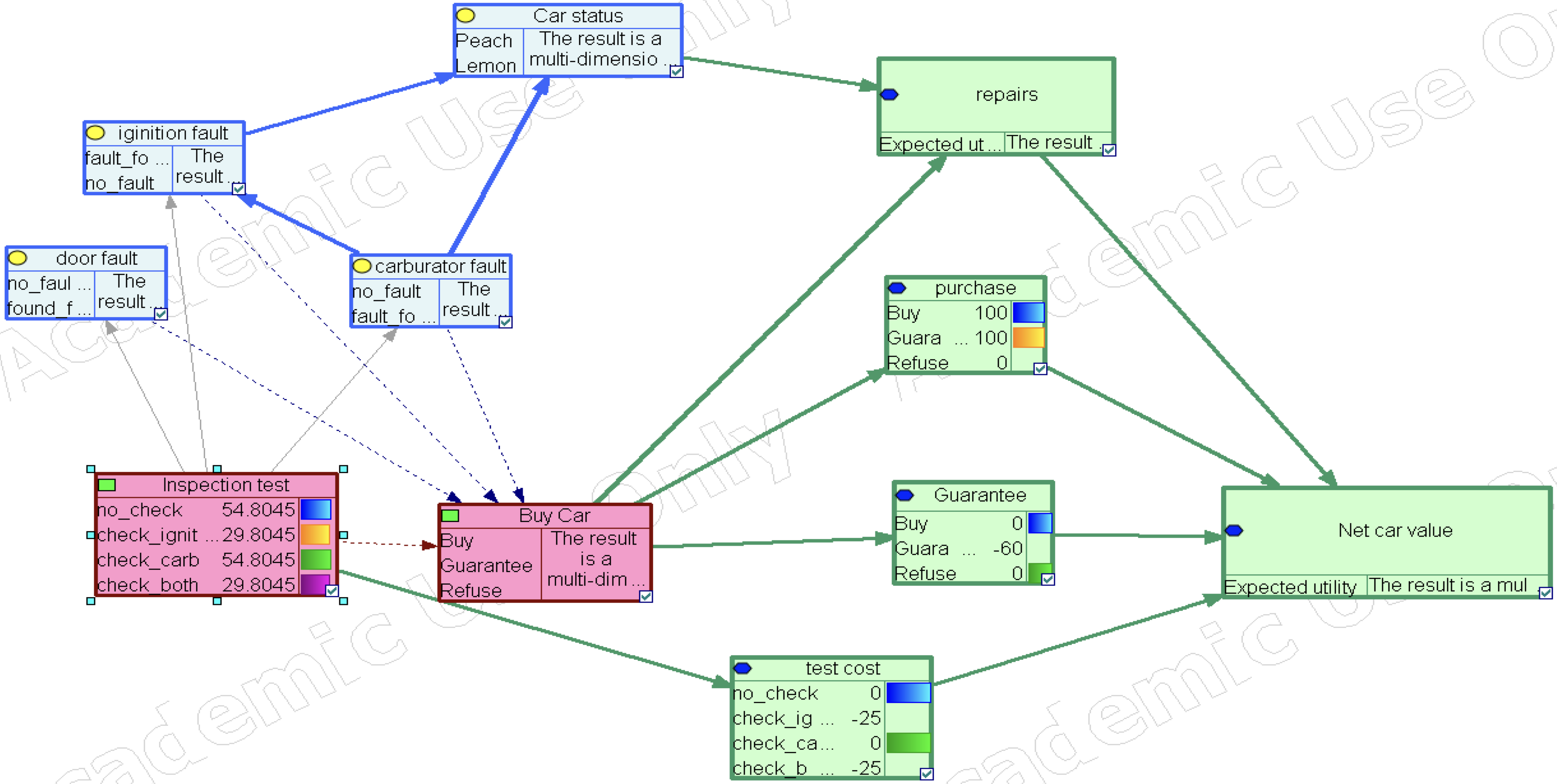
```
# Condition to get p(y | g, c)
# Note, this is just the conditional probabilities at the node leaves.
y_norm = new_joint_p.sum(2)
Py_given_gc = new_joint_p / y_norm.expand(2,2,2).permute(1,2,0)
Py_given_gc
```

[92] ✓ 0.0s Open 'Py_given_gc' in Data Wrangler

```
tensor([[[0.9382, 0.0618],
         [0.0058, 0.9942]],

        [[0.0048, 0.9952],
         [0.8948, 0.1052]]])
```



The “test then buy” influence diagram by adding the prediction model to the *Used Car Buyer* value model



Expanding the prediction and value sub-models
to show the full influence diagram

Applications:
Where is Decision Modeling Useful?

Decisions with significant uncertainty:

- Product management: Selection of new features
- Cloud automation: Model deployment
- Operations: Optimal supply chains
- Healthcare: Treatment choices

The take-away

Derived from a common mathematical framework,

Data Science predictive models and

decision analysis models

can combine

Bayesian statistics and

decision theory.

Abstract

- Decision Analysis (DA) and Data Science teams often operate at cross-purposes, using distinct frameworks. The aim of this work is to demonstrate how current machine learning methods can be applied in a principled way to Decision Analysis. For example, machine learning methods do not make the prior explicit, but directly estimate posterior predictive distributions. Estimating likelihoods instead avoids this confounding. Extending well-known textbook examples, such as value of information analysis, with estimation of likelihoods illustrates how this is possible. The result is a novel analytical approach that combines software tools that solve influence diagrams with Bayesian statistical methods. This advances the computational aspects of Decision Analysis, aligning it with contemporary evidence-based practices.