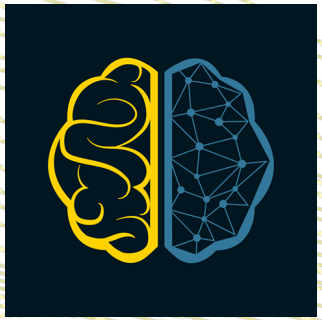
Explaining the model or Making black box transparent

an overview

Dmitry Larko, Sr. Data Scientist @ H2O.ai
dmitry@h2o.ai



November 28th, 2016

H2O.ai

About me

Dmitry Larko

Sr. Data Scientist at H2O.ai

San Francisco Bay Area, CA, United States

Joined 4 years ago · last seen in the past day

http://h2o.ai

Competitions Grandmaster

Home | Competitions (32) | Kernels (0) | Discussion (31) | Datasets (0) | More | Edit Profile

Competitions Grandmaster			Kernels Contributor			Discussion Contributor		
Current Rank	Highest Rank		Unranked			Unranked		
36 of 50,664	25							
9	7	6	0	0	0	0	4	12

Machine learning and model interpretation

- Machine learning studies algorithms that learn from data and make predictions
- Learning algorithms are about *correlations* in the data

- In contrast, in data science and data mining, understanding *causality* is essential
- Applying domain knowledge requires understanding and interpreting models

Usefulness of model interpretation

- Often, we need to understand individual predictions a model is making.

For example a model may

- Recommend a treatment for a patient or estimate a disease to be likely. The **doctor** needs to understand the reasoning.
- Classify a user as a scammer, but the user disputes it. The **fraud analyst** needs to understand why the model made the classification.

Usefulness of model interpretation cont.

- Understanding differences on a dataset level.
  - Why is a new software release receiving poorer feedback from customers when compared to the previous one?
  - Why are grain yields in one region higher than the other?

- Debugging models. A model that worked earlier is giving unexpected results on newer data.

Algorithmic transparency

- Algorithmic transparency becoming a requirement in many fields

- French Conseil d'Etat (State Council's) recommendation in „Digital technology and fundamental rights"(2014) : *Impose to algorithm-based decisions a transparency requirement, on personal data used by the algorithm, and* ***the general reasoning it followed****.*

- Federal Trade Commission (FTC) Chair Edith Ramirez: *The agency is concerned about 'algorithmic transparency /../'* (Oct 2015).
  FTC Office of Technology Research and Investigation started in March 2015 to tackle algorithmic transparency among other issues

Possible solutions

- **Interpretable models**. Alternative machine learning techniques that learn more structured, interpretable, or causal models
  - Learning simpler, more compact models, such as Bayesian Rule Lists (Letham, Rudin, McCormick, & Madigan, 2015)
  - Learning richer, more conceptual, generative models, such as Bayesian Program Learning (Lake, Salakhutdinov, & Tenenbaum, 2015)

- **Model induction.** Techniques that experiment with any given machine learning model—as a black box—to infer an approximate, explainable model
  - Visualizing algorithmic models using Glassbox , Partial plots
  - L.I.M.E. Explaining the Predictions of Any Classifier by Ribeiro, Singh, & Guestrin (2016)

- **Model deconstruction.** Techniques which build explanation using information about model architecture.
  - Visualizing and Understanding Convolutional Networks (Zeiler & Fergus, 2014)
  - Random Forest interpretation using treeinterpreter
  - Extracting interpretable information from tree ensembles using inTrees package

Agenda

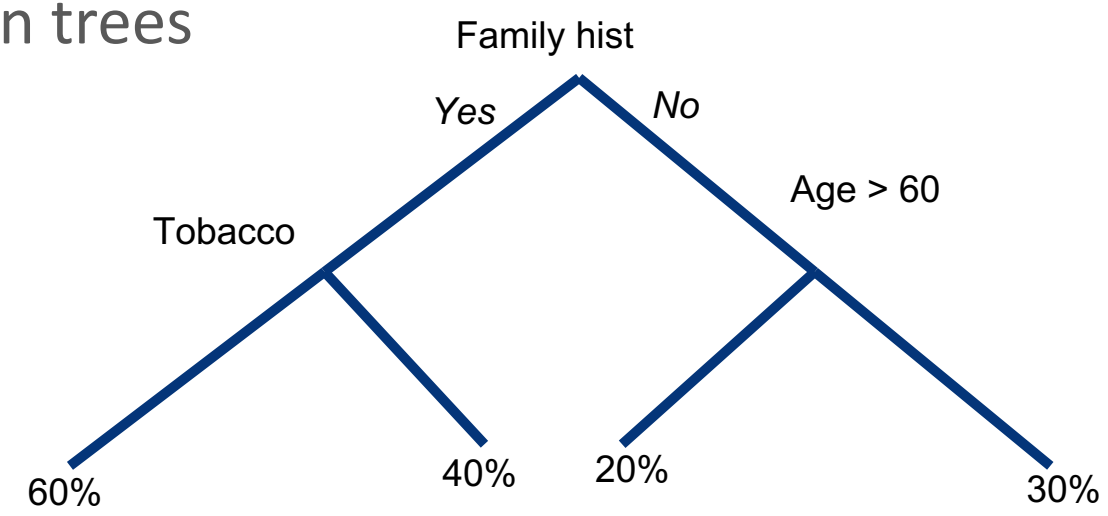
- **Interpretable models**. Alternative machine learning techniques that learn more structured, interpretable, or causal models
  - Learning simpler, more compact models, such as Bayesian Rule Lists (Letham, Rudin, McCormick, & Madigan, 2015)
  - Learning richer, more conceptual, generative models, such as Bayesian Program Learning (Lake, Salakhutdinov, & Tenenbaum, 2015)

- **Model induction.** Techniques that experiment with any given machine learning model—as a black box—to infer an approximate, explainable model
  - Visualizing algorithmic models using Glassbox or Partial plots
  - L.I.M.E. Explaining the Predictions of Any Classifier by Ribeiro, Singh, & Guestrin (2016)

- **Model deconstruction.** Techniques which build explanation using information about model architecture.
  - Visualizing and Understanding Convolutional Networks (Zeiler & Fergus, 2014)
  - Random Forest interpretation using treeinterpreter
  - Extracting interpretable information from tree ensembles using inTrees package

Interpretable models

- Traditionally, two types of (mainstream) models considered when interpretability is required
- Linear models (linear and logistic regression)
  - $Y = a + b_1 x_1 + \cdots + b_n x_n$
  - heart_disease = 0.08*tobacco + 0.043*age + 0.939*famhist + ... (from *Elements of statistical learning*)
- Decision trees



Decision trees

- Decision trees are understandable only when they are (very) small
  - Tree of depth *n* has up 2^n leaves and 2^n -1 internal nodes. With depth 20, a tree can have up to 1048576 leaves
- Additionally decision trees are high variance method – low generalization, tend to overfit

Random forests

- Can learn non-linear relationships in the data well
- Robust to outliers
- Can deal with both continuous and categorical data
- Require very little input preparation (see previous three points)
- Fast to train and test, trivially parallelizable
- High accuracy even with minimal meta-optimization

- Considered to be a **black box** that is difficult or impossible to interpret

Understanding the model vs the predictions

- Keep in mind, we want to understand why a particular decision was made. Not necessarily every detail of the full model

- As an analogy, we don't need to understand how a brain works to understand why a person made a particular a decision: simple explanation can be sufficient

- Ultimately, as ML models get more complex and powerful, hoping to understand the models themselves is doomed to failure
- **We should strive to make models explain their decisions**

Turning the black box into a white box

- In fact, random forest predictions **can** be explained and interpreted, by decomposing predictions into mathematically *exact* feature contributions

- Independently of the
  - number of features
  - number of trees
  - depth of the trees

Revisiting decision trees

- Classical definition (from *Elements of statistical learning*)

$$dt(x) = \sum_{m=1}^{M} c_m I(x \in R_m)$$

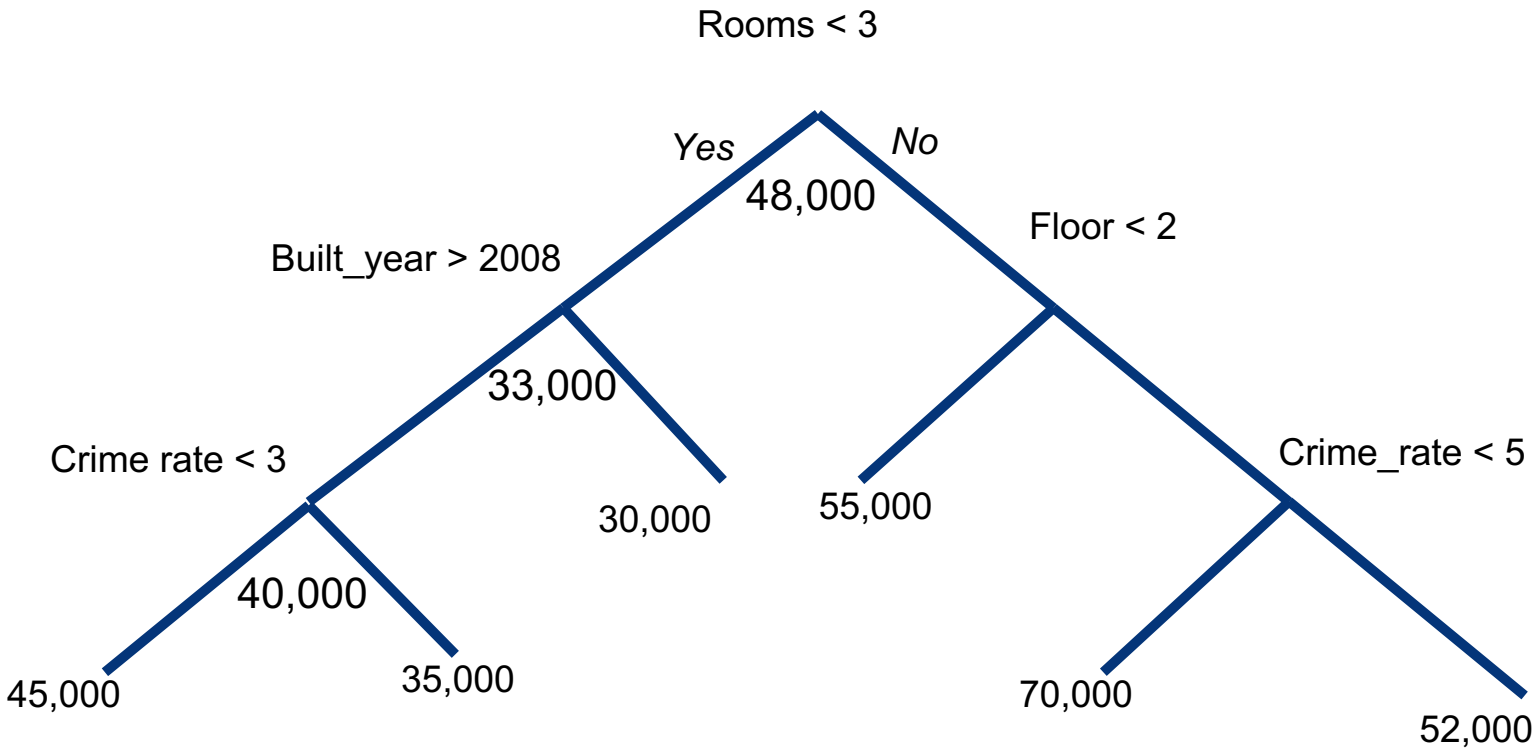
- Tree divides the feature space into M regions R_m (one for each leaf)
- Prediction for feature vector *x* is the constant c_m associated with region R_m the vector *x* belongs to

Operational view

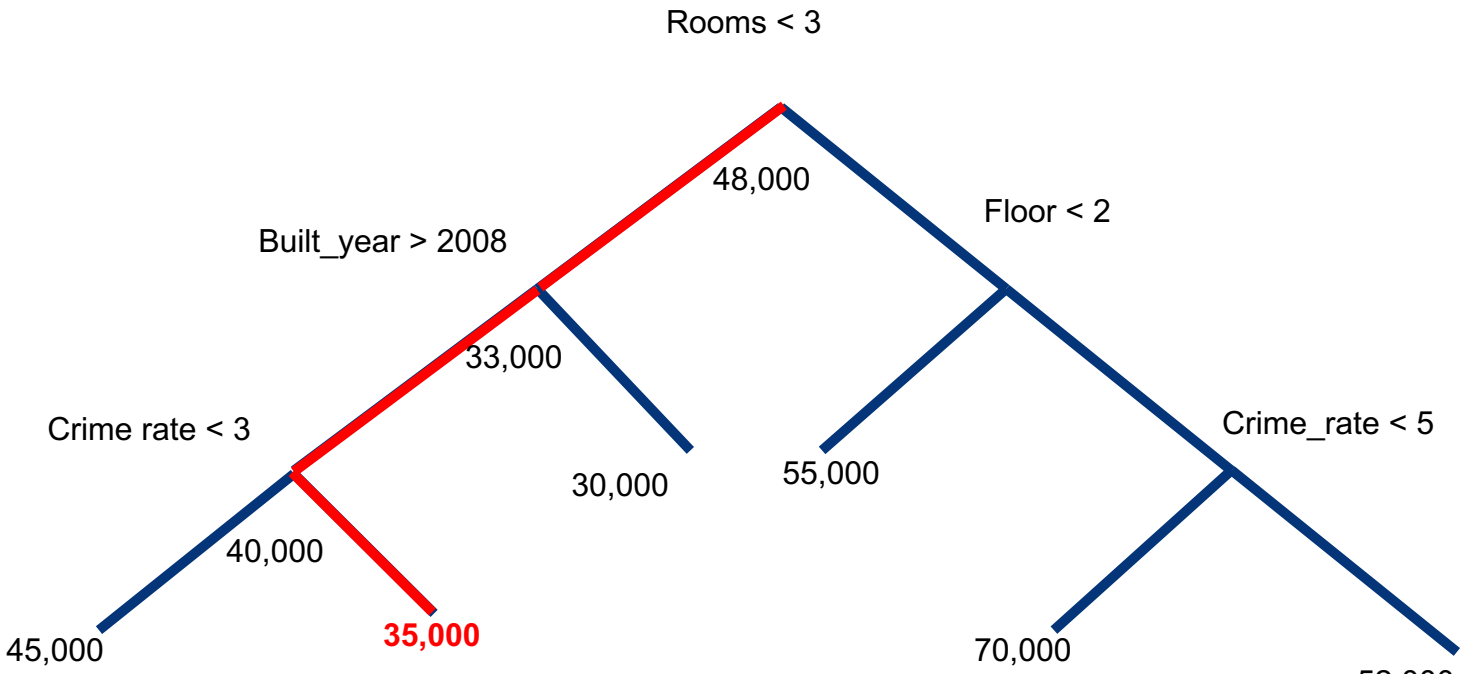
- Classical definition ignores the operational aspect of the tree.

- There is a **decision path** through the tree
- All nodes (not just the leaves) have a value associated with them
- Each decision along the path contributes something to the final outcome
- A feature is associated with every decision
- We can compute the final outcome in terms of feature contributions

Estimating apartment prices



Estimating apartment prices



X = [2 rooms; Built in 2010; Neighborhood crime rate: 5]
Path taken: Rooms < 3, Built_year > 2008, Crime_rate < 3
Prediction: 35,000

Decomposition for decision trees

- We can define the the decision tree in term of the bias and contribution from each feature.

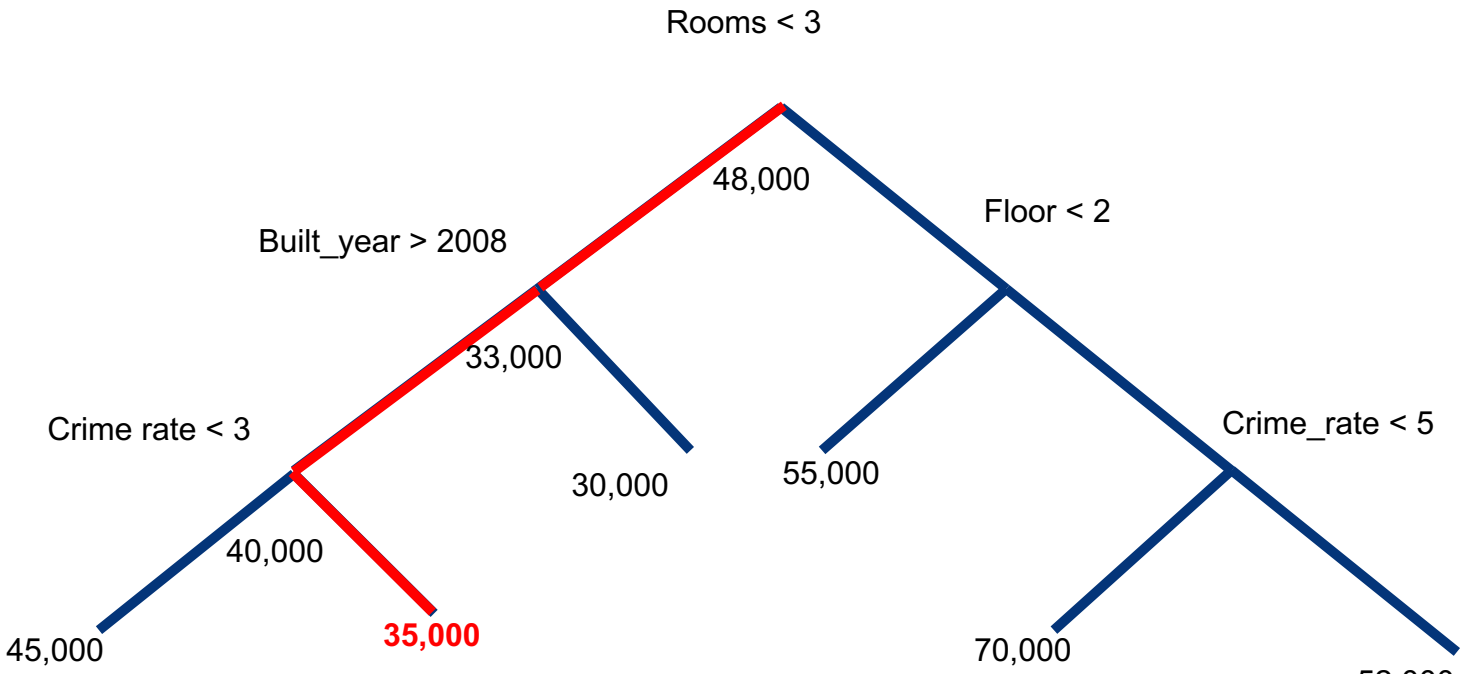
$$\mathrm{dt}(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \quad \Rightarrow \quad \mathrm{dt}(x) = bias + \sum_{i=1}^{N} contr(i, x))$$

- Similar to linear regression

  prediction = bias + feature$_1$contribution + … + feature$_n$contribution

  but on a prediction level, not model level
- Does not depend on the size of the tree or number of features
- Works equally well for
  - Regression and classification trees
  - Multivalued and multilabel data

Estimating apartment prices



X = [2 rooms; Built in 2010; Neighborhood crime rate: 5]
Path taken: Rooms < 3, Built_year > 2008, Crime_rate < 3
Prediction: 35,000
Price = 48,000 – 15,000(Rooms) + 7,000(Built_year) – 5,000(Crime_rate) = 35,000

Deeper inspections

- We can have more fine grained definition in addition to pure feature contributions
  - Separate negative and positive contributions
  - Contribution from interactions (floor == 1 & has_terrace → 3000)
  - etc

- Number of features typically not a concern because of the long tail. In practice, top 10 features contribute the vast majority of the overall deviation from the mean

From decision trees to random forests

- Prediction of a random forest is the average of the predictions of individual trees

$$\mathrm{RF}(x) = \frac{1}{J}\sum_{j=1}^{J} dt_j(x)$$

- From distributivity of multiplication and associativity of addition, random forest prediction can be defined as the average of bias term of individual trees + sum of averages of each feature contribution from individual trees

$$\mathrm{RF}(x) = \frac{1}{J}\sum_{j=1}^{J} bias_j(x) + (\frac{1}{J}\sum_{j=1}^{J} contr_j(1,x) + \cdots + \frac{1}{J}\sum_{j=1}^{J} contr_j(n,x))$$

- prediction = bias + feature_1contribution + … + feature_ncontribution

Summary

There is a very straightforward way to make random forest predictions more interpretable,
leading to a similar level of interpretability as linear models — not in the static but dynamic sense. Every prediction can be trivially presented as a sum of feature contributions, showing how the features lead to a particular prediction. This opens up a lot of opportunities in practical machine learning and data science tasks:

- Explain to an analyst why a particular prediction is made
- Debug models when results are unexpected
- Explain the differences of two datasets (for example, behavior before and after treatment), by comparing their average predictions and corresponding average feature contributions.

Links

Blog posts:

- http://blog.datadive.net/interpreting-random-forests/
- http://blog.datadive.net/random-forest-interpretation-with-scikit-learn/
- http://blog.datadive.net/random-forest-interpretation-conditional-feature-contributions/

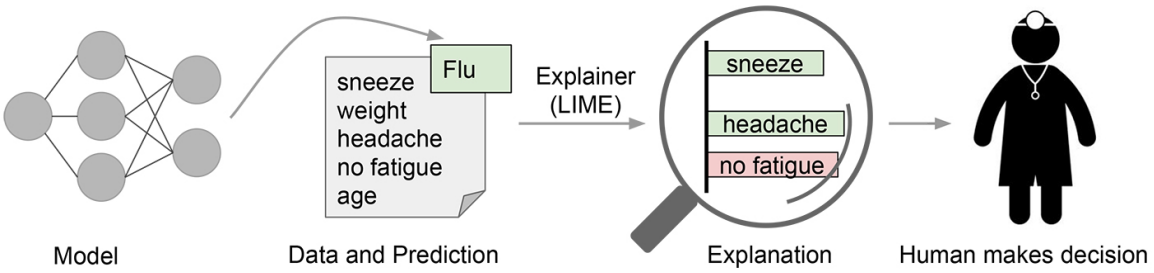
Presentation:

- http://www.slideshare.net/andosa/interpreting-machine-learning-models

GitHub:

- https://github.com/andosa/treeinterpreter

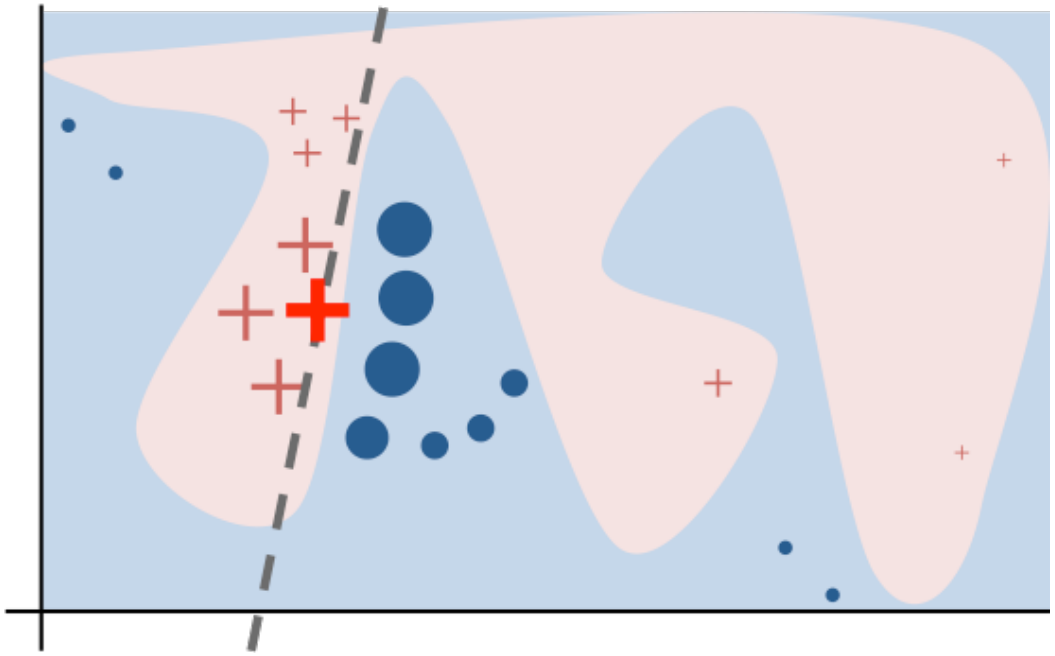
L.I.M.E.



Local Interpretable Model-Agnostic Explanations (LIME), a technique to explain the predictions of *any* machine learning classifier.

- **Local** refers to local fidelity - i.e., we want the explanation to really reflect the behavior of the classifier "around" the instance being predicted
- **Interpretable** - explanation in terms of interpretable representations, so human can make sense of it
- It is able to explain any model without needing to 'peak' into it, so it is **model-agnostic**

LIME visualization



The bright red cross is the instance being explained (let's call it X):

- sample instances around X
- weight them according to their proximity to X (weight here is indicated by size).
- learn a linear model (dashed line) that approximates the model well in the vicinity of X, but not necessarily globally

LIME framework

Interpretable Data Representations

interpretable explanations need to use a representation that is understandable to humans, regardless of the actual features used by the model

$$x \in R^d \longleftrightarrow x' \in \{0,1\}^{d'}$$

original representation — interpretable representation

Fidelity-Interpretability Trade-off

$$\xi(x) = \underset{g \in G}{\operatorname{argmax}} \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

$g \in G$	G is a class of potentially interpretable models, The domain of g is $\{0,1\}^{d'}$, i.e. g acts over absence/presence of the interpretable components
$\Omega(g)$	measure of complexity
$f(x)$	probability (or a binary indicator) that x belongs to a certain class
π_x	proximity measure
$\mathcal{L}(f, g, \pi_x)$	measure of how unfaithful g is in approximating f in the locality defined by π_x

Sampling for Local Exploration

$$\xi(x) = \underset{g \in G}{\operatorname{argmax}} \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

Explainer must be **model-agnostic,** so we minimize $\mathcal{L}(f, g, \pi_x)$ without any assumptions about f

- sample instances around x' by drawing nonzero elements of x' uniformly at random:

$$x' \in \{0,1\}^{d'} \Rightarrow z' \in \{0,1\}^{d'}$$

- using z' recover the sample in the original representation $z \in R^d$
- use z to obtain $f(z)$
- use $f(z)$ to label z' data, so we create dataset $Z = \{z', f(z)\}$ and learn g

Now it's locally faithful explanation where locality is captured by π_x

Sparse Linear Explanations

$g \in G$ G be the class of linear models, such that $g(z') = w_g z'$

$\mathcal{L}$ locally weighted square loss

$\pi_x(z) = \exp\left(-\frac{D(x,z)^2}{\sigma^2}\right)$, an exponential kernel defined on some distance function D (e.g. cosine distance for text, $L2$ distance for images) with width σ

$$\mathcal{L}(f, g, \pi_x) = \sum_{z,z' \in Z} \pi_x(z)(f(z) - g(z'))^2$$

Sparse Linear Explanations

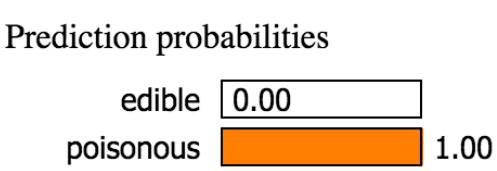
Algorithm 1 Sparse Linear Explanations using LIME

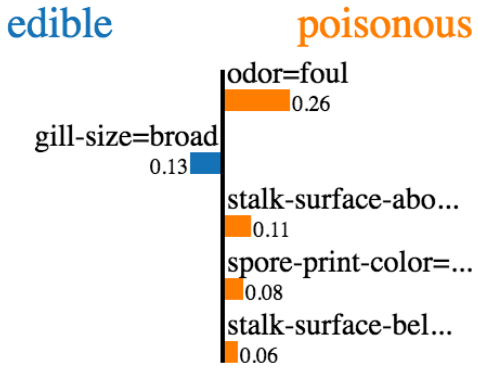
Require: Classifier f, Number of samples N
Require: Instance x, and its interpretable version x'
Require: Similarity kernel π_x, Length of explanation K

$\mathcal{Z} \leftarrow \{\}$
for $i \in \{1, 2, 3, ..., N\}$ **do**
  $z'_i \leftarrow sample\_around(x')$
  $\mathcal{Z} \leftarrow \mathcal{Z} \cup \langle z'_i, f(z_i), \pi_x(z_i) \rangle$
end for
$w \leftarrow$ K-Lasso$(\mathcal{Z}, K)$ ▷ with z'_i as features, $f(z)$ as target
return w

Tabular data

Dataset: Mushroom Data Set
From: UCI Machine Learning repository dataset. Link
Goal: Predict poisonous mushrooms
Size: 8124 rows, 22 features (all categorical)



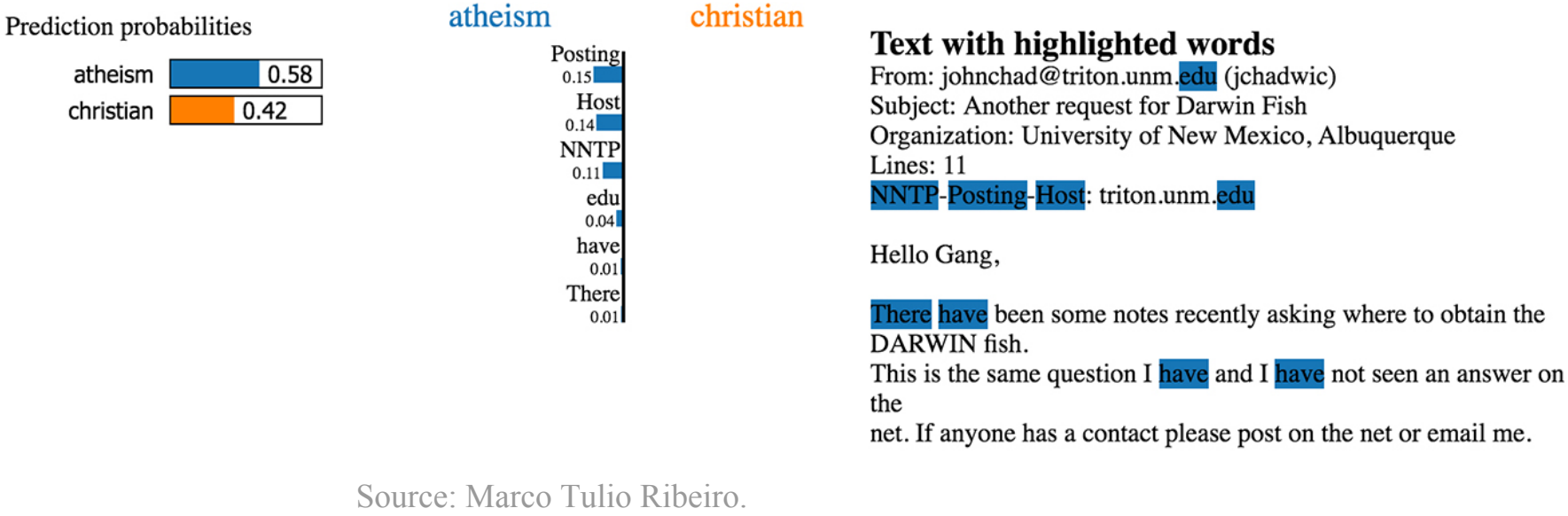


Feature	Value
odor=foul	True
gill-size=broad	True
stalk-surface-above-ring=silky	True
spore-print-color=chocolate	True
stalk-surface-below-ring=silky	True

Source: Marco Tulio Ribeiro.

Text

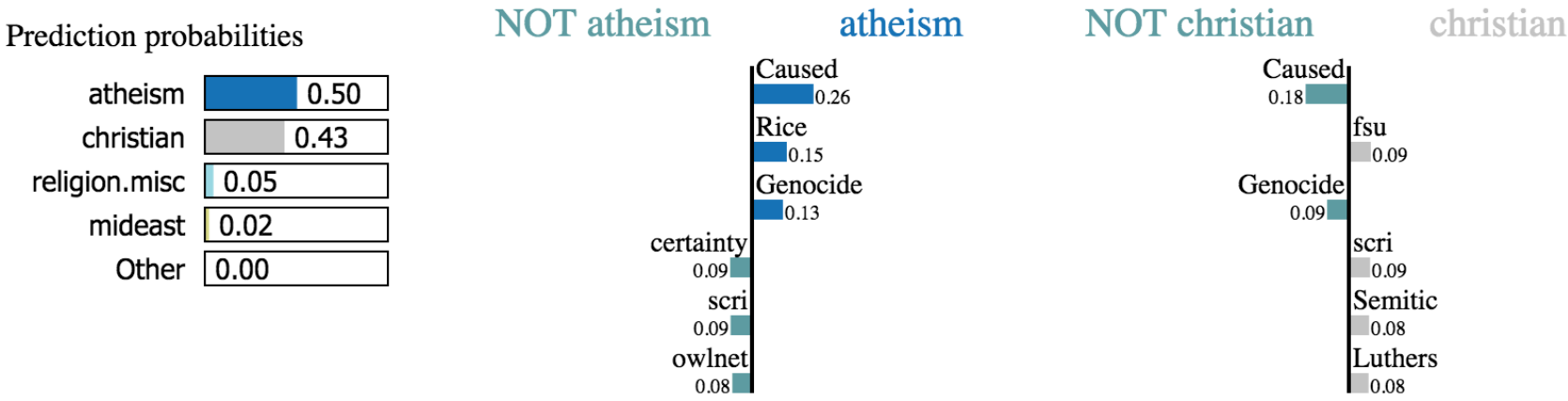
Dataset: 20Newsgroups Data Set
From: UCI Machine Learning repository dataset. Link
Goal: Predict poisonous mushrooms
Size: 20000 text messages



Source: Marco Tulio Ribeiro.

Text (multi class)

Dataset: 20Newsgroups Data Set
From: UCI Machine Learning repository dataset. Link
Goal: Predict poisonous mushrooms
Size: 20000 text messages



Source: Marco Tulio Ribeiro.

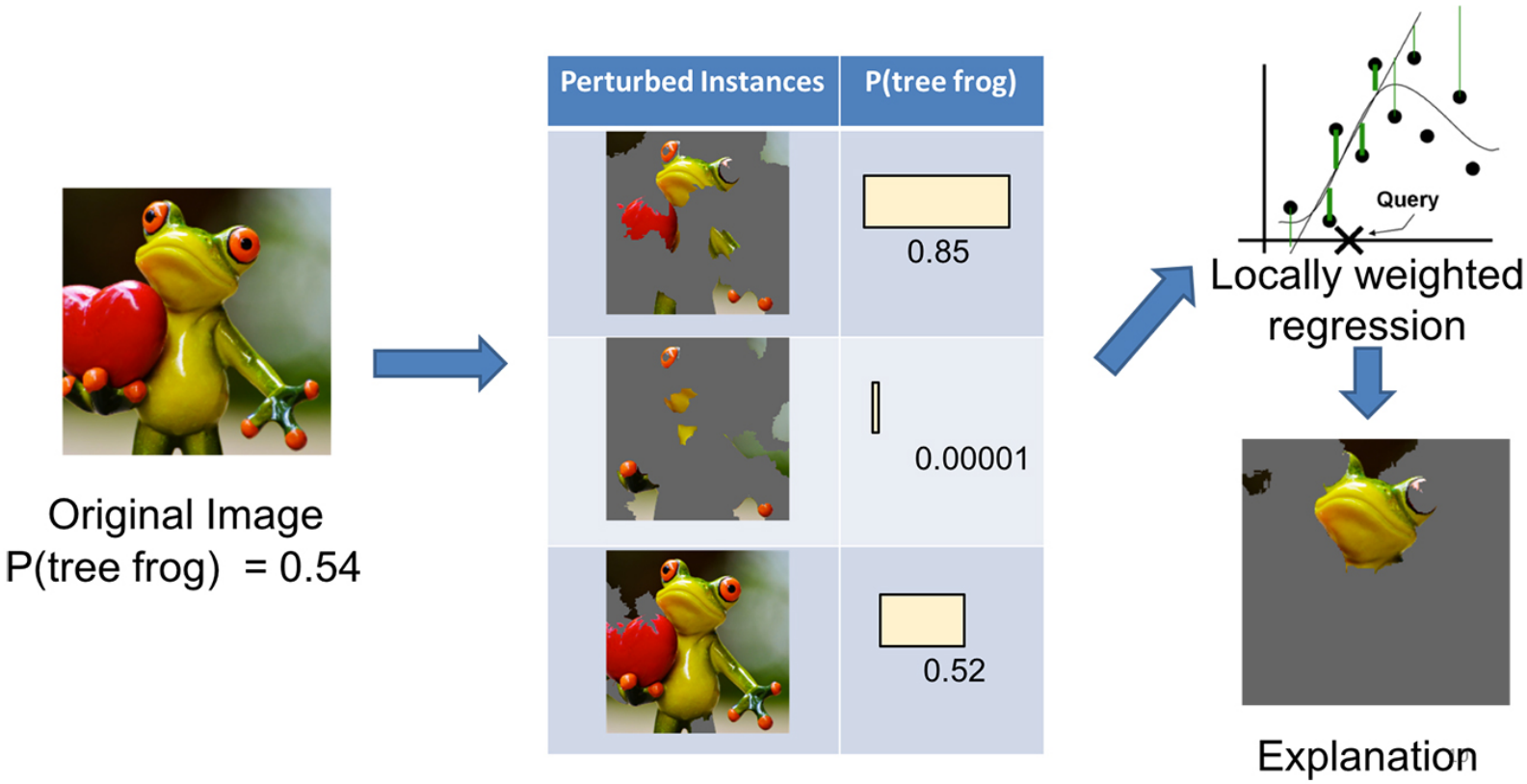
Image recognition

Original Image

Interpretable Components

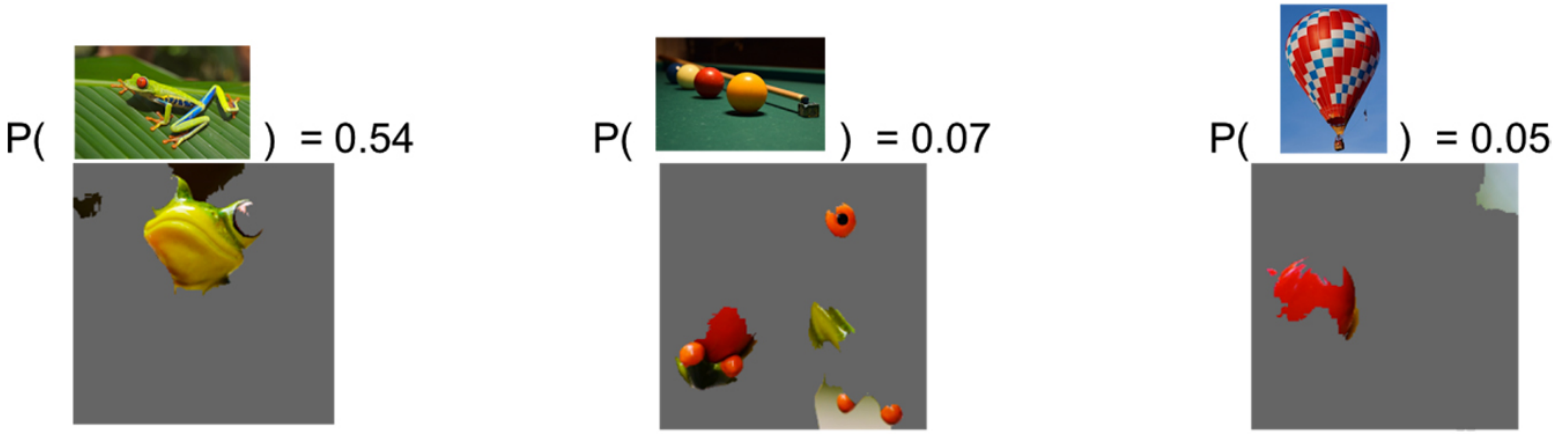
Sources: Marco Tulio Ribeiro, Pixabay.

Image recognition



Sources: Marco Tulio Ribeiro, Pixabay.

Image recognition



Explanation for a prediction from Inception. The top three predicted classes are "tree frog," "pool table," and "balloon." Sources: Marco Tulio Ribeiro, Pixabay (frog, billiards, hot air balloon).

Links

Blog posts:

- https://www.oreilly.com/learning/introduction-to-local-interpretable-model-agnostic-explanations-lime
- https://homes.cs.washington.edu/~marcotcr/blog/lime/

ArXiv:

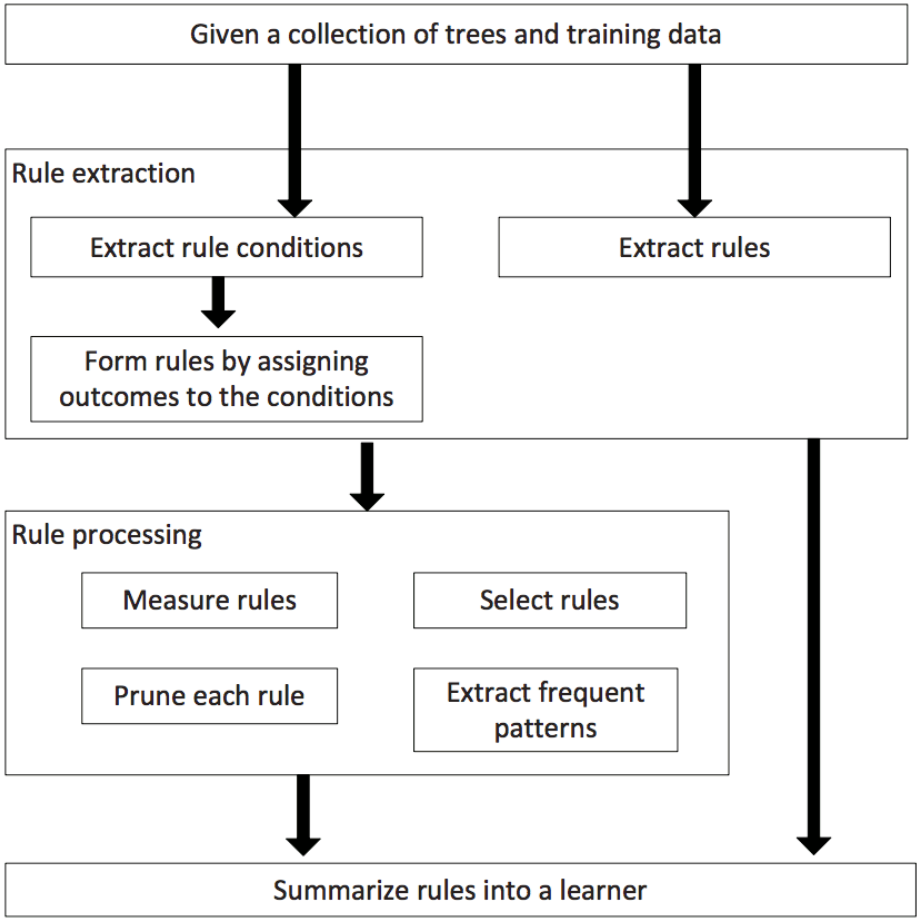
- https://arxiv.org/pdf/1602.04938v3.pdf

GitHub:

- https://github.com/marcotcr/lime

inTree

Extract interpretable information (rules) from a tree ensemble
Applicable to many tree ensembles such as as random forests, regularized random forests and boosted trees



Example

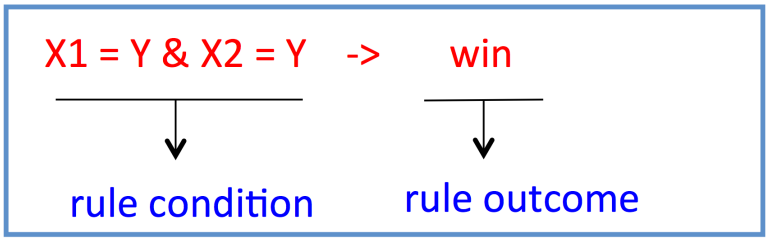
XOR:

X1	X2	result
Y	Y	lose
Y	N	win
N	Y	win
N	N	lose

plus 18 random variables X3 – X20

Build RRF (regularized random forest) with 100 trees

Extract conditions



1923 (1835 distinct) conditions are extracted from RRF (100 trees).

For example:

condition
X[,1] %in% c('N') & X[,2] %in% c('N') & X[,19] %in% c('N')
X[,1] %in% c('Y') & X[,2] %in% c('N') & X[,19] %in% c('N')

Assign outcomes

condition
X[,1] %in% c('N') & X[,2] %in% c('N') & X[,19] %in% c('N')
X[,1] %in% c('Y') & X[,2] %in% c('N') & X[,19] %in% c('N')

condition	pred
X[,1] %in% c('N') & X[,2] %in% c('N') & X[,19] %in% c('N')	lose
X[,1] %in% c('Y') & X[,2] %in% c('N') & X[,19] %in% c('N')	win

Measure rules

Frequency proportion of instances that satisfy a rule condition
Error the error rate of a rule
Length (Complexity) # variable-value pairs in a rule condition

condition	pred
X[,1] %in% c('N') & X[,2] %in% c('N') & X[,19] %in% c('N')	lose
X[,1] %in% c('Y') & X[,2] %in% c('N') & X[,19] %in% c('N')	win

len	freq	err	condition	pred
3	0.07	0	X[,1] %in% c('N') & X[,2] %in% c('N') & X[,19] %in% c('N')	lose
3	0.16	0	X[,1] %in% c('Y') & X[,2] %in% c('N') & X[,19] %in% c('N')	win

Prune rules

len	freq	err	condition	pred
3	0.07	0	X[,1] %in% c('N') & X[,2] %in% c('N') & X[,19] %in% c('N')	lose
3	0.16	0	X[,1] %in% c('Y') & X[,2] %in% c('N') & X[,19] %in% c('N')	win

irrelevant variables

len	freq	err	condition	pred
2	0.22	0	X[,1] %in% c('N') & X[,2] %in% c('N')	lose
2	0.24	0	X[,1] %in% c('Y') & X[,2] %in% c('N')	win

Select a compact rule set

len	freq	err	condition	pred
2	0.22	0	X[,1] %in% c('N') & X[,2] %in% c('N')	lose
2	0.24	0	X[,1] %in% c('Y') & X[,2] %in% c('N')	win

• • • 1000+ rules

len	freq	err	condition	pred	impRRF
2	0.22	0	X[,1] %in% c('N') & X[,2] %in% c('N')	lose	1
2	0.22	0	X[,1] %in% c('Y') & X[,2] %in% c('Y')	lose	0.97
2	0.24	0	X[,1] %in% c('Y') & X[,2] %in% c('N')	win	0.60
2	0.32	0	X[,1] %in% c('N') & X[,2] %in% c('Y')	win	0.42

Summarize rules (simplified tree ensemble learner)

len	freq	err	condition	pred
2	0.22	0	X[,1] %in% c('N') & X[,2] %in% c('N')	lose
2	0.24	0	X[,1] %in% c('Y') & X[,2] %in% c('N')	win

... 1000+ rules

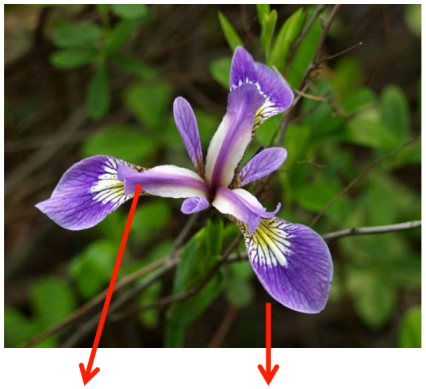
len	freq	err	condition	pred
2	0.32	0	X[,1] %in% c('N') & X[,2] %in% c('Y')	win
2	0.24	0	X[,1] %in% c('Y') & X[,2] %in% c('N')	win
1	0.44	0	X[,1]==X[,1]	lose

More readable format

len	freq	err	condition	pred
2	0.32	0	X1 %in% c('N') & X2 %in% c('Y')	win
2	0.24	0	X1 %in% c('Y') & X2 %in% c('N')	win
1	0.44	0	Else	lose

Real-life example: Iris data

setosa



virginica

versicolor

simplified tree ensemble learner (STEL)

len	freq	err	condition	pred
1	0.37	0	petal.width<=0.7	setosa.
3	0.31	0	petal.len>2.6 & petal.len<=4.85 & petal.width<=1.6	versicolor.
2	0.25	0	petal.len>4.85 & petal.width>1.7	virginica.
3	0.03	0	sepal.width<=3.1 & petal.len<=4.85 & petal.width>1.6	virginica.
3	0.02	0	sepal.width>2.25 & petal.width>0.8 & petal.width<=1.75	versicolor.
1	0.01	0	petal.len>4.85	virginica.
1	0.01	0	Else	versicolor.

Links

Website:

- https://sites.google.com/site/houtaodeng/intrees

ArXiv:

- http://arxiv.org/pdf/1408.5456v1.pdf

CRAN package:

- inTrees

What next?

- Another approach how to explain model for specific example: A Model Explanation System (MES)
- Principles of Explanatory Debugging to Personalize Interactive Machine Learning
- MIT: Making computers explain themselves
- MLconf 2016 “Interpreting Black-Box Models with Applications to Healthcare” talk and slides.

Thank you!

Q&A

Appendix

Bayesian Rule Lists

Rule List model of the well-known Titanic dataset:

IF *male* AND *adult* THEN survival probability: 21% (19% - 23%)
ELSE IF *3rd class* THEN survival probability: 44% (38% - 51%)
ELSE IF *1st class* THEN survival probability: 96% (92% - 99%)
ELSE survival probability: 88% (82% - 94%)

TBD: Add some high level explanation