SAND2012-9592P

Estimating Extrapolation Risk in Supervised Machine Learning

Should I trust *this* prediction?

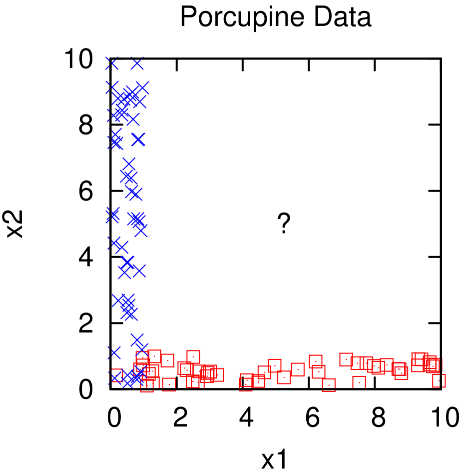
Art Munson, Philip Kegelmeyer

Sandia National Laboratories

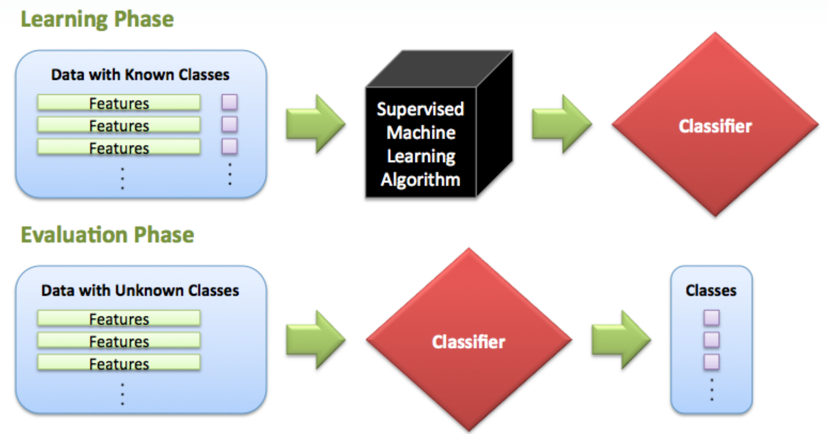
November 1, 2012

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

A Toy Example



Supervised Machine Learning from 10K Feet



Successful Applications:

- Bing (Microsoft)
- Kinect (Microsoft)
- Friend Recommendations (Facebook)

A Troubling Assumption

Machine learning assumes future data looks like past data.

What happens if:

- a new category appears?
- future data is noisier?
- a category evolves (e.g., malware)?

Answer: user gets a prediction, business as usual.

Can we detect when machine learning is extrapolating?

Focus: decision tree ensembles

Notation

- $X = (X_1, X_2, \ldots, X_m)$: the input feature space
- $\boldsymbol{x} = (x_1, x_2, \ldots, x_m) \in X$: a feature vector
- $Y = \{y_1, y_2, \ldots, y_c\}$: the set of possible classes

In supervised learning, the training data are labeled input-output pairs:

$$T = \left\{ (\boldsymbol{x}^{(1)}, y^{(1)}), \cdots, (\boldsymbol{x}^{(n)}, y^{(n)}) \right\}$$

Given T, a learning algorithm outputs a probability estimator

$$h : X \mapsto \Phi(Y)$$

where $\phi(Y) = (\Pr(Y = y_1), \ldots, \Pr(Y = y_c)) \in \Phi(Y)$.

Extrapolation Risk

Following Hooker (2004), define extrapolation risk for $\boldsymbol{x}$ as

$$R(\boldsymbol{x}) = \frac{f_U(\boldsymbol{x})}{f_U(\boldsymbol{x}) + f_T(\boldsymbol{x})}$$

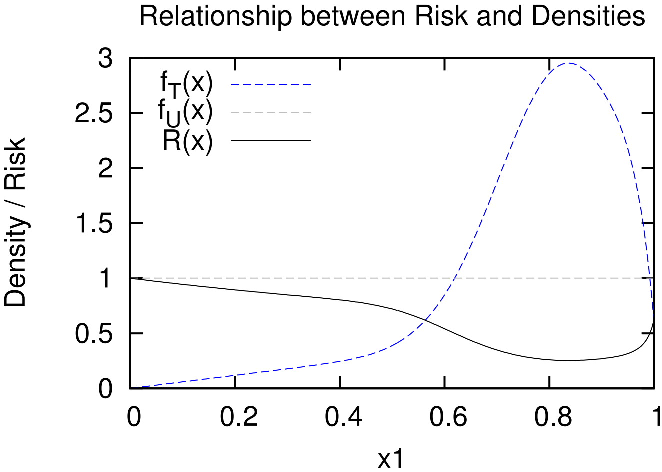
- $f_U(\boldsymbol{x})$: data density at $\boldsymbol{x}$ assuming a uniform distribution
- $f_T(\boldsymbol{x})$: data density at $\boldsymbol{x}$ assuming the same distribution that generated the training data

Note:

- $R(\boldsymbol{x}) \in [0, 1]$
- $0 \longrightarrow$ safe
- $1 \longrightarrow$ high risk

$R(x)$ in One Dimension

$$R(\boldsymbol{x}) = \frac{f_U(\boldsymbol{x})}{f_U(\boldsymbol{x}) + f_T(\boldsymbol{x})}$$



Two Approaches to Estimating Prediction Risk

Builtin Risk (*BR*) — provided by classifier

- most classifiers can report prediction confidence
- free! (or almost)
- standard practice

Auxiliary Risk (*AR*) — provided by separate risk model

- need to train another model! (density/outlier)
- independent of classifier model
- relatively unexplored

Two Approaches to Estimating Prediction Risk

Builtin Risk (*BR*) — provided by classifier

- most classifiers can report prediction confidence
- free! (or almost)
- standard practice

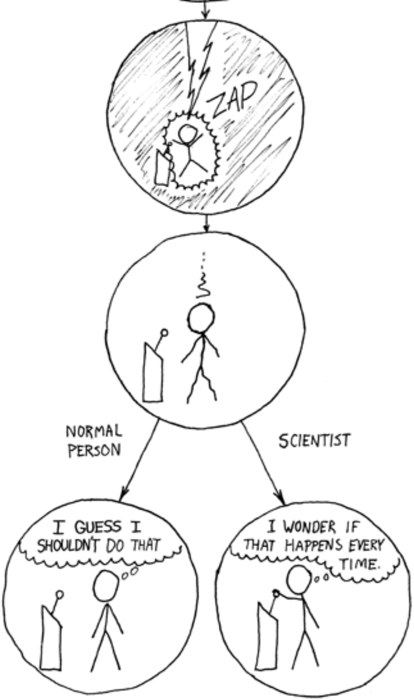
Auxiliary Risk (*AR*) — provided by separate risk model

- need to train another model! (density/outlier)
- independent of classifier model
- relatively unexplored

Sneak Preview: $BR(x)$ and $AR(x)$ have complementary strengths.
Only one aligns with $R(x)$.

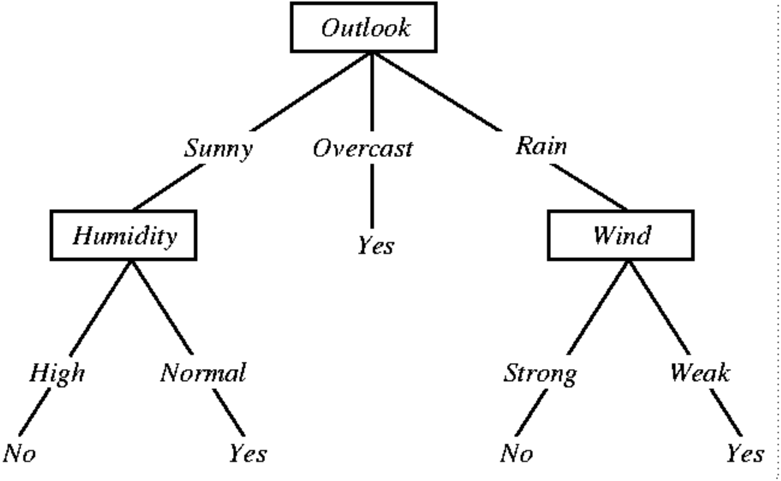
$BR(x)$ for Decision Tree Ensembles

Detour:
Decision Trees
and Ensembles



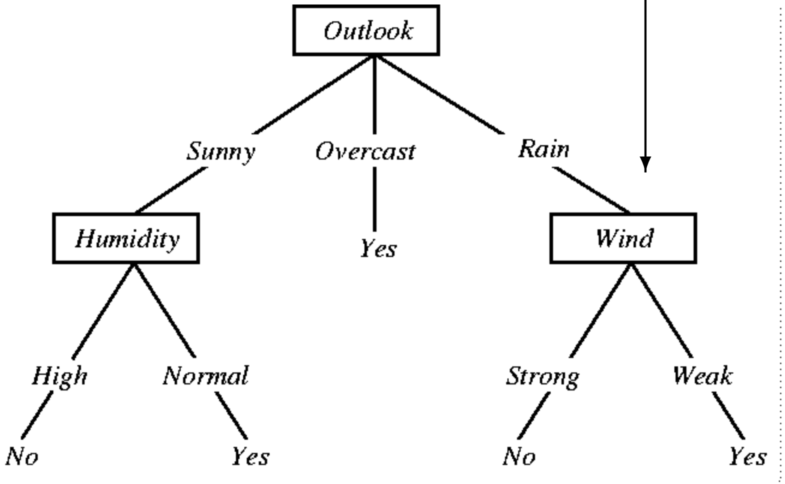
Source:
http://xkcd.com/242/

Decision Tree Review

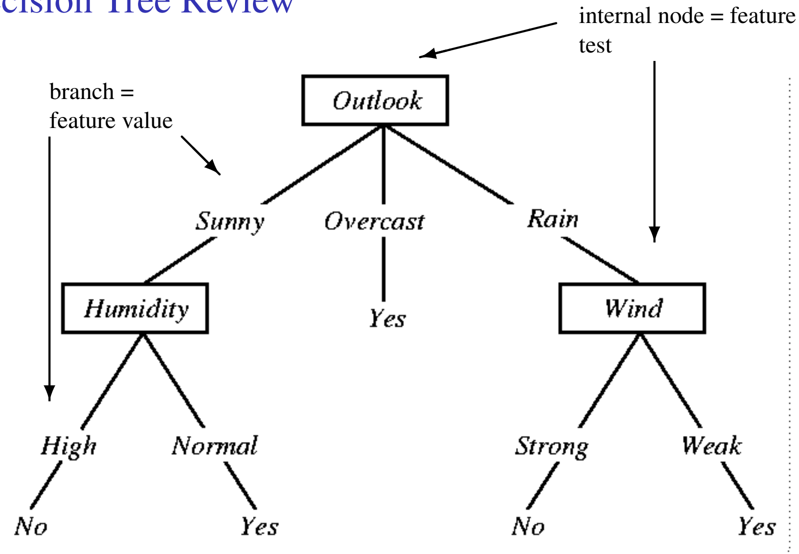


Decision Tree Review

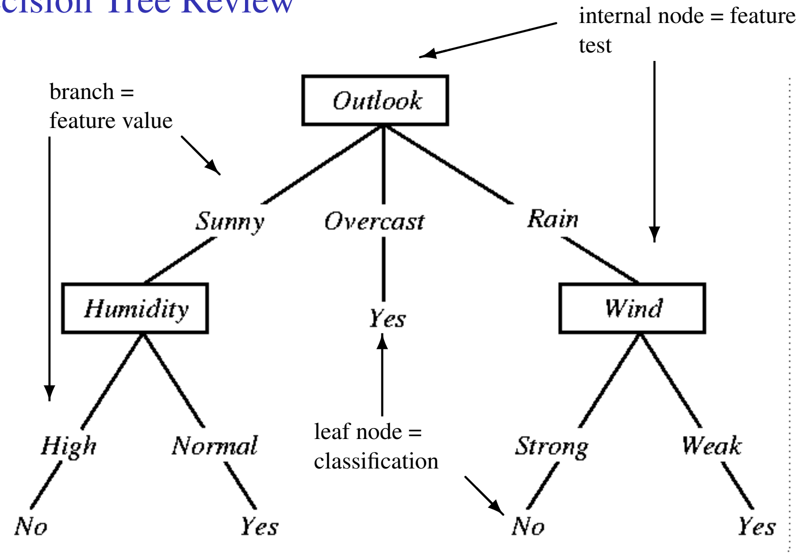
internal node = feature test



Decision Tree Review



Decision Tree Review



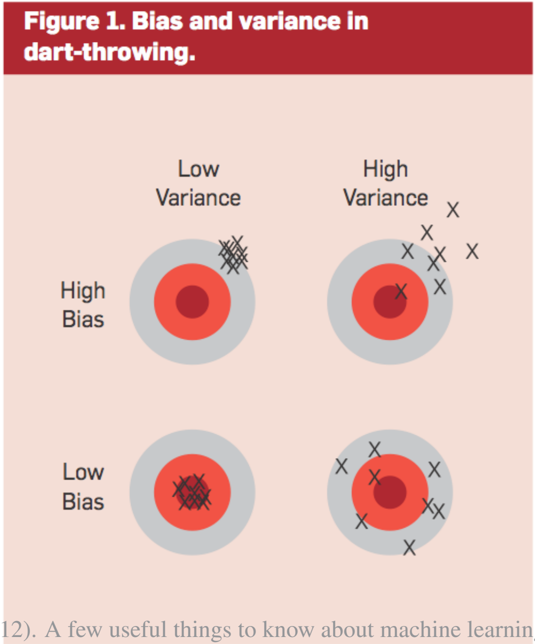
Decision Tree Strengths & Weaknesses

Strengths:

- Handle numeric and categorical features.
- Missing values are okay.
- Invariant to monotonic feature scaling.
- Robust to noisy training labels.
- Fast.
- Low bias.

Weaknesses:

- High variance.

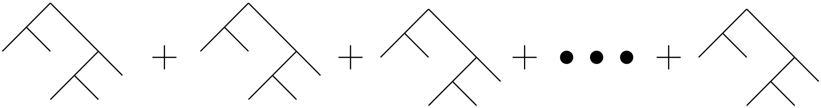


Domingos (2012). A few useful things to know about machine learning. Communications of the ACM 55(10):78–87.

Review of Simple Ensemble Learning

Bagging: simple ensemble learning algorithm [1]:

- draw random sample of training data
- train a model using sample (e.g. decision tree)
- repeat N times (e.g. 25 times)
- bagged predictions: average predictions of N models



Ensemble Learning Intuition

Ensemble machine learning: wisdom of crowds

Truth	**1**	**0**	**1**	**1**	**0**	**Accuracy**
Model 1	1	0	0	1	1	60%
Model 2	0	1	1	1	0	60%
Model 3	0	0	1	0	0	60%
Model 4	1	1	1	1	1	60%
Model 5	1	0	0	0	0	60%
Vote 1–5	1	0	1	1	0	100%

- No one model has to get it all right
- Performance of ensemble outperforms individuals
- Usually more reliable / robust
- Reduces variance

Back to $BR(x)$ for tree ensembles...

$BR(x)$: Vote Margin

Margin

Margin is the gap between the class with the most votes and the class with the 2nd most votes.

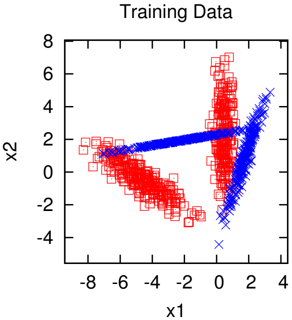
BR(x)

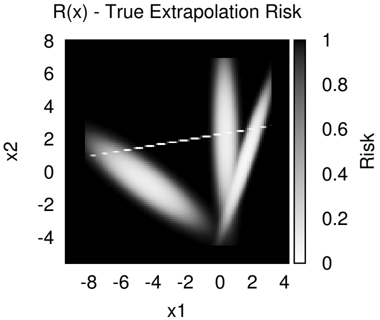
$BR(\boldsymbol{x}) = 1 - \text{margin}$

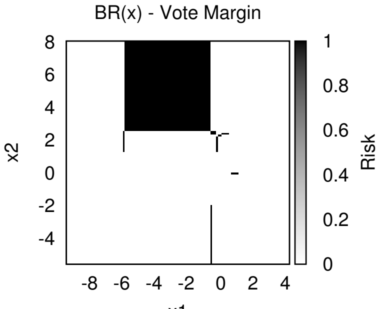
Example: suppose an ensemble with 100 trees votes:

	Class y_1	Class y_2	Class y_3	Margin	$BR(\cdot)$
$\boldsymbol{x}^{(1)}$	65	35	0	0.30	0.70
$\boldsymbol{x}^{(2)}$	30	25	45	0.15	0.85
$\boldsymbol{x}^{(3)}$	0	100	0	1.00	0.00

Synthetic Data Results



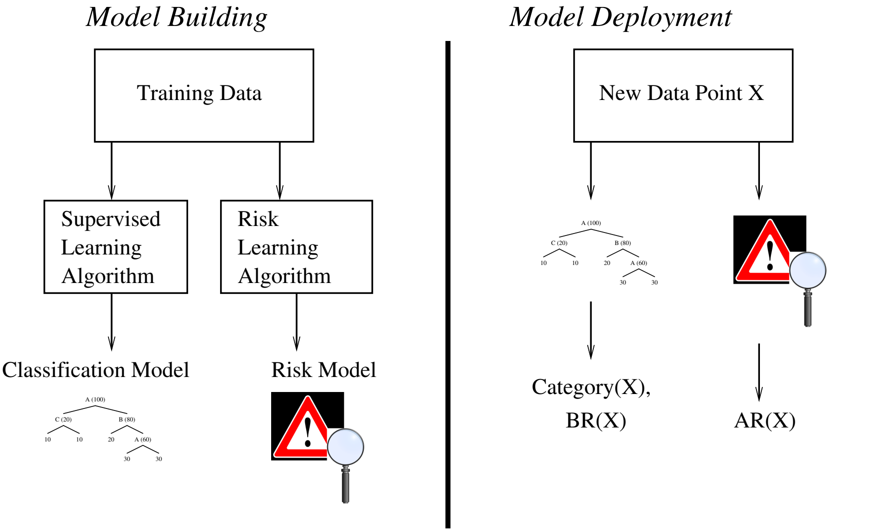




Summary:
$BR(\boldsymbol{x})$ mainly useful for detecting uncertainty caused by instability.

$\ldots$ let's try $AR(\boldsymbol{x})$ $\ldots$

$AR(x)$: Big Picture

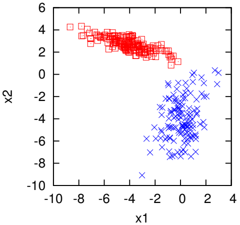


$AR(\boldsymbol{x})$: Building Block

Hooker (2004) proposed *confidence and extrapolation risk trees* (CERT) for estimating extrapolation risk.

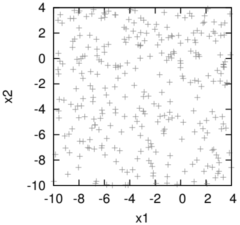
- Idea: frame as classification problem.

Foreground Class
(all train data)



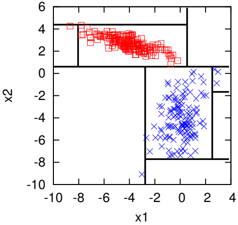
Vs.

Background Class
(uniform)



$\Longrightarrow$

Result:



- $\Pr(Y = \text{background} \mid \boldsymbol{x}) \approx R(\boldsymbol{x})$
- Could use any classification algorithm that estimates probabilities.

$AR(\boldsymbol{x})$: Building Block (2)

Problem:

$$\begin{aligned} \text{High dimensions} &\implies \text{sparsely sampled background} \\ &\implies \text{high variance} \end{aligned}$$

Solution: don't sample!

- Decision tree learning minimizes entropy of subregions r:

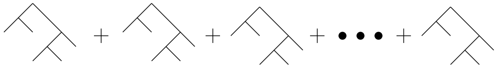
$$\begin{aligned} \text{Entropy}(r) &= -\sum_{i=1}^{c} p(y_i) \log_2 p(y_i) \\ &= -p(\text{foreground}) \log_2 p(\text{foreground}) \\ &\quad - p(\text{background}) \log_2 p(\text{background}) \end{aligned}$$

  with

$$p(y_i) = \Pr(Y = y_i \mid r) = \frac{\# \, y_i \in r}{\# \text{ foreground} \in r + \# \text{ background} \in r}$$

- Compute # background analytically.

$AR(\boldsymbol{x})$: CERT Ensemble



Extend Hooker's work by applying bagging to CERT:

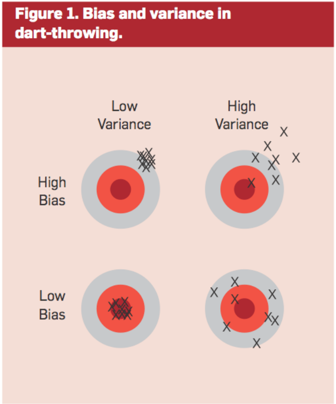
- draw random sample of foreground data
- train CERT model using sample
- repeat 100 times
- ensemble prediction is $AR(\boldsymbol{x})$

AR(x)

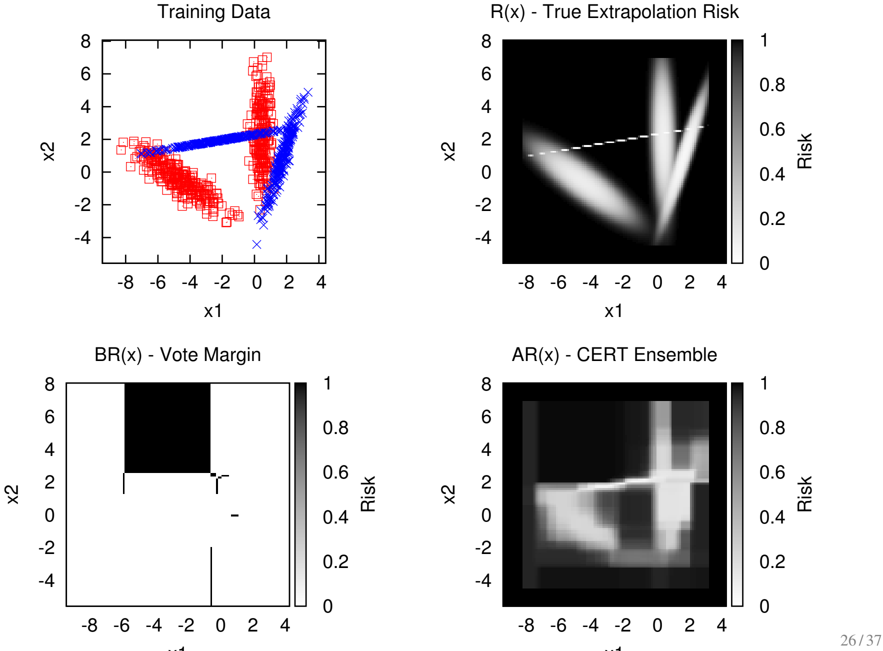
$$AR(\boldsymbol{x}) = \frac{1}{100} \sum_{t=1}^{100} \Pr_t(Y = \text{background} \mid \boldsymbol{x})$$

Questions about $AR(\boldsymbol{x})$

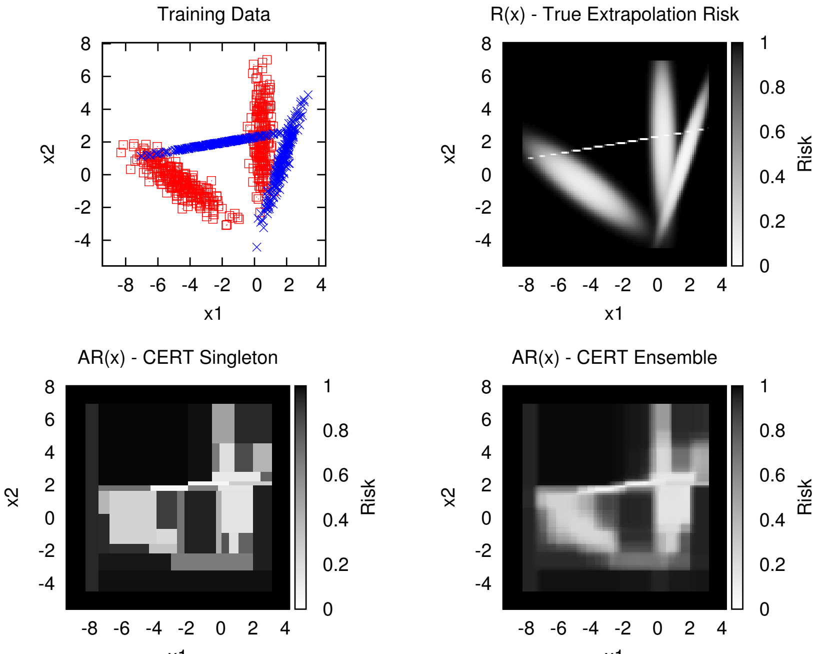
1. Better risk estimation by using ensemble?
   - Bagging reduces variance. . .
   - . . . but no variance in background data.
2. Does $AR(\boldsymbol{x})$ work?



Synthetic Data Results



Ensembles improve CERT



Case Study I: Detect Novel NYT Topic

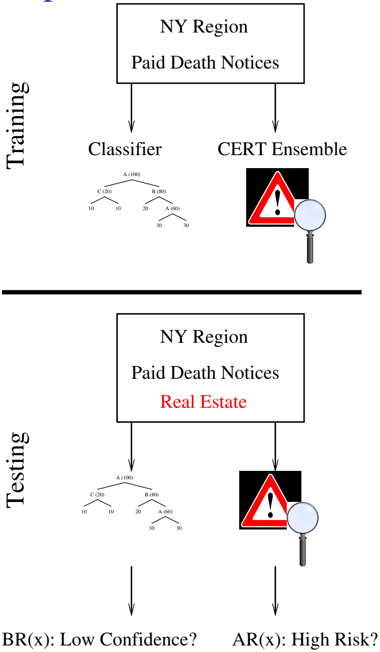
Experiment: Detect Novel NYT Topic

Data:

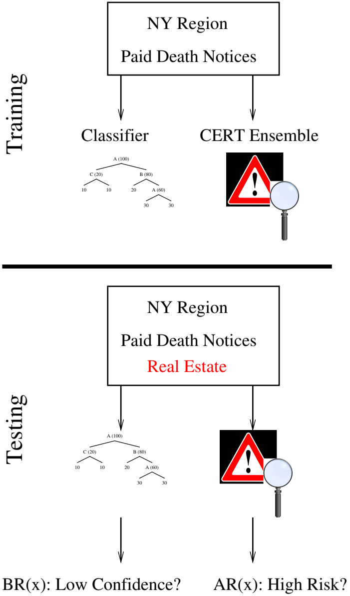
- 22,926 NYT articles
  - 48.9% NY Region
  - 48.6% Paid Death Notices
  - 2.4% Real Estate
- 9 numeric features (LSA)
- 1/2 train, 1/2 test

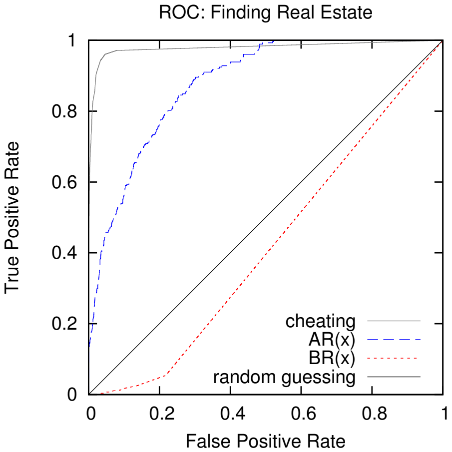
Experiment Design:

- Real estate topic omitted from training.
- Find real estate in testing?
  - $BR(\boldsymbol{x})$: vote margin
  - $AR(\boldsymbol{x})$: CERT ensemble



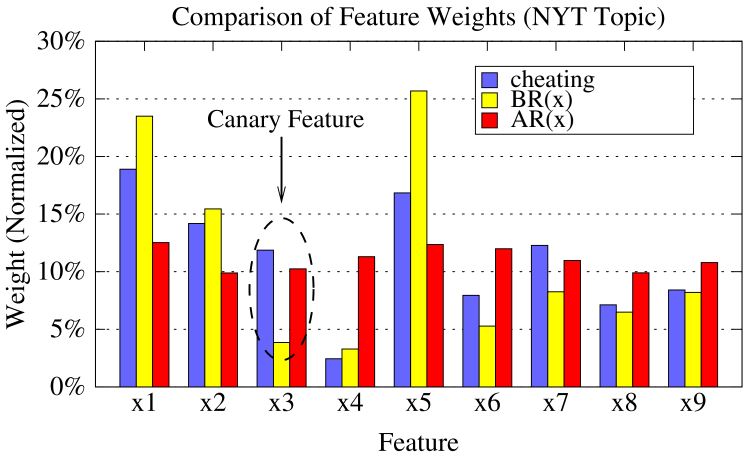
Take Away #2: Auxiliary Risk Model Needed





Why is $BR(\boldsymbol{x})$ worse than random?

Classification model ignores feature x_3
— which is important for finding the novel class.



Case Study II: Predict if EXE is Malware

Use Case: Predicting Reliability of Malware Classifier

Scenario: predict when model classifying EXE files might be wrong.[1]

Data:

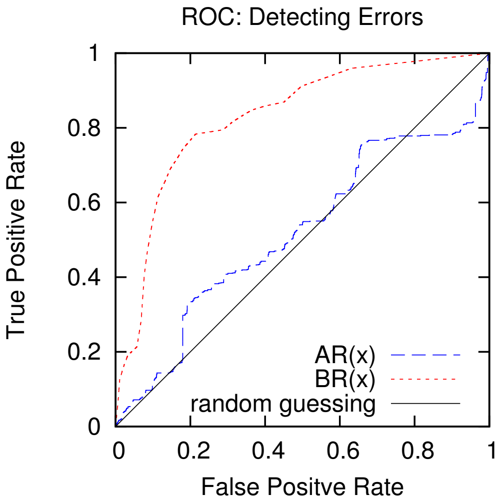
- Training Data: 2010
  - 18,588 examples
  - 44.8% malware
- Testing Data: 2011
  - 16,432 examples
  - 79.3% malware
- Extracted Features
  - 57 categorical features
  - 63 numeric features

Setup:

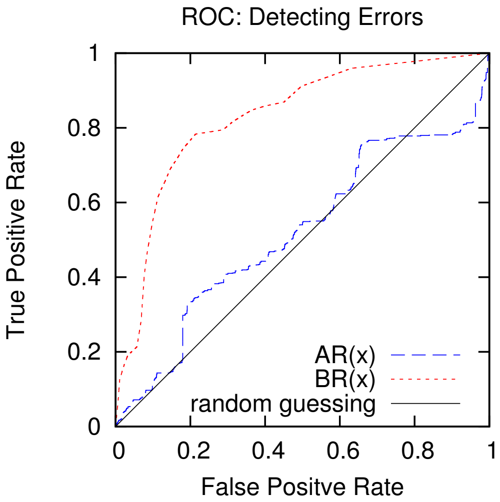
- Train classifier to predict goodware or malware.
- Train auxiliary risk model.
- Does classifier make mistakes on high risk test points?

[1] Data from Ken Chiang, Michael Karres, and Levi Lloyd.

Sometimes, $BR(\boldsymbol{x})$ is what you need!



Sometimes, $BR(\boldsymbol{x})$ is what you need!



Hypothesis: more errors from class overlap than outlier data

Future Work: Combining $BR(\boldsymbol{x})$ and $AR(\boldsymbol{x})$

Algorithm 1: Simple Risk Combination Baseline

```
if BR(x) is high then
    declare prediction risky;
else
    if AR(x) is high then
        declare prediction risky
    else
        declare prediction safe
```

Conclusions & Next Steps

- Builtin and auxiliary risk measures are complementary.
  - $BR(\boldsymbol{x})$ useful for finding unstable predictions.
  - $AR(\boldsymbol{x})$ good for detecting extrapolation risk.
- Ensembles improve CERT's risk assessments.

Future work:

- Further validation on real data sets.
- Try other risk learning algorithms: density estimation, outlier detection.
- Benefit from combining?

Questions?

mamunso@sandia.gov

Bibliography I

Leo Breiman.
Bagging predictors.
Machine Learning, 24(2):123–140, 1996.

Giles Hooker.
Diagnosing extrapolation: Tree-based density estimation.
In Won Kim, Ronny Kohavi, Johannes Gehrke, and William DuMouchel, editors, *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 569–574, New York, NY, USA, 2004. ACM.

Parikshit Ram and Alexander G. Gray.
Density estimation trees.
In Chid Apté, Joydeep Ghosh, and Padhraic Smyth, editors, *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 627–635, New York, NY, USA, 2011. ACM.

Backup Slides

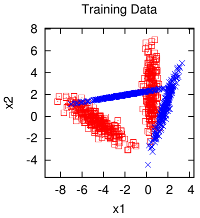
Aside

Cool. But wouldn't it be better to do density estimation from first principles?

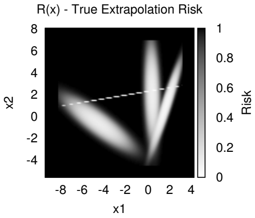
CERT vs DET

Compare density estimation trees [3] to CERT. Default params.

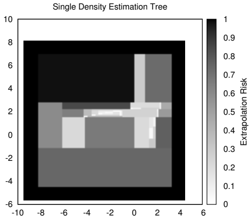
(a) Training Data (1000 pts)



(b) Oracle



(c) DET



(d) CERT