How effective is your classifier?
Revisiting the role of metrics in machine learning

SANMI KOYEJO

Joint work with Ran Li, Xiaoyan Wang, Gaurush Hiranandani, Shant Boodaghians, and Ruta Mehta

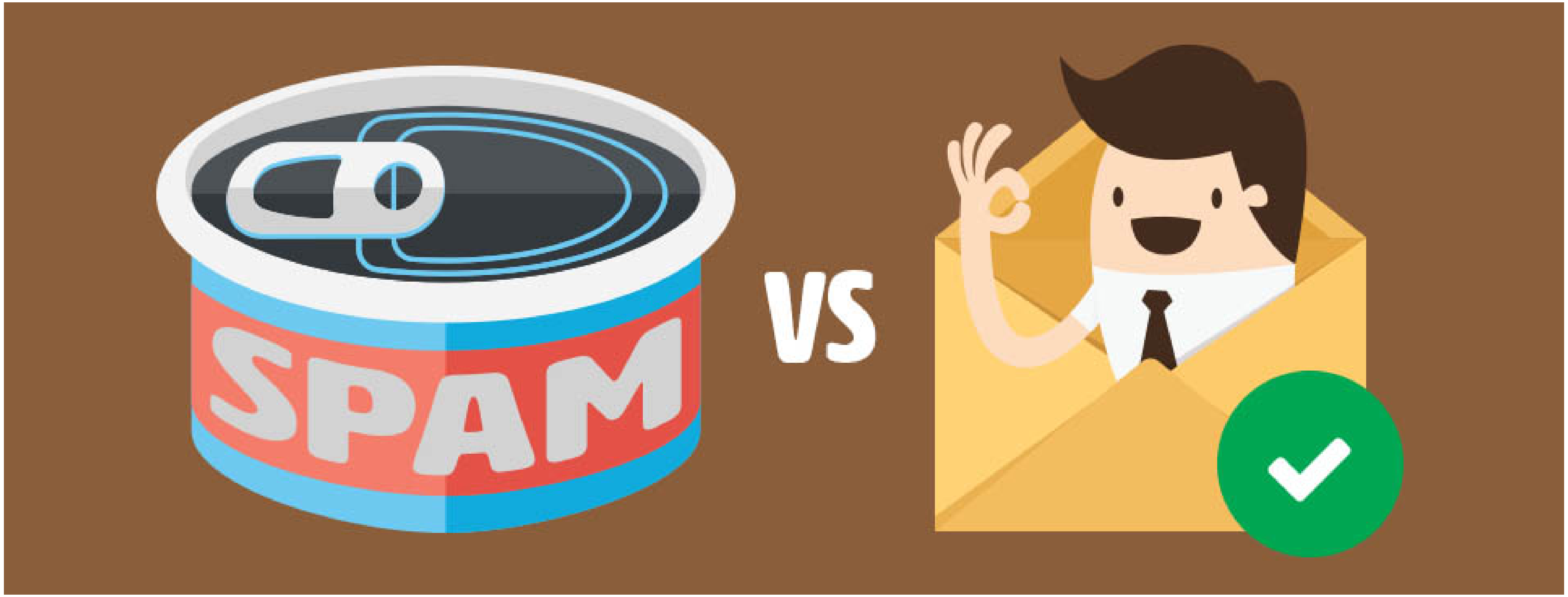
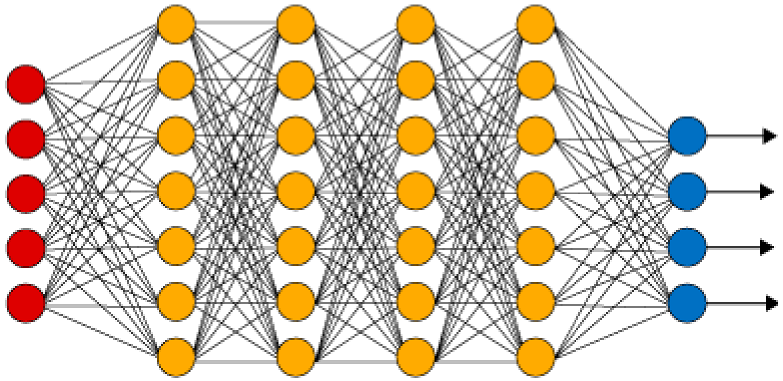


Image Source: https://davepannell.com/public/2016/03/Email-marketing-vs-spam.jpg



- Accuracy = 95%
- $$$

- Users complain that most real emails are labelled spam
- ~90% of all email is spam*
- Suggests that accuracy is the wrong metric as it gives equal weight to all errors

Image Source: https://becominghuman.ai/deep-learning-made-easy-with-deep-cognition-403fbe445351
*Source: Symantec circa 2008; https://www.theatlas.com/charts/NJipnKmq

HOW SHOULD YOU MEASURE PERFORMANCE?

It depends… on the relative cost/benefit of different kinds of errors.

The **metric** is a quantitative description of tradeoffs.

Error analysis for binary classification

		Ground truth	
		Spam	Not Spam
Predicted	Spam	TP	FP
	Not Spam	FN	TN

- Accuracy $= \mathrm{TP} + \mathrm{TN} = 1 - \mathrm{FP} - \mathrm{FN}$
- To improve user calibration, try evaluating and/or optimizing weighted accuracy e.g.

$$\phi(h) = 1 - 0.1\,\mathrm{FP} - \mathrm{FN}$$

The confusion matrix

		Ground truth	
		Y = 1	Y = 0
Predicted	h(x) = 1	TP	FP
	h(x) = 0	FN	TN

$$= \mathbf{C}(h)$$

Beyond Accuracy, more general metrics are nested functions

$$\phi(h) = \psi\big(\mathbf{C}(h)\big)$$

- Metrics are used to compare classifiers, or can be optimized directly
- The classifier performance metric can be approximated from data.

Lots of real-world examples

$$\phi(h) = a_1\mathrm{TP} + a_2\mathrm{FP} + a_3\mathrm{FN} + a_4\mathrm{TN}$$

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}},\ \mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{FP}+\mathrm{TN}},\ \mathrm{Prec} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}},\ \mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{FN}+\mathrm{TP}},\ \mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FN}}.$$

$$\mathrm{AM} = \frac{1}{2}\left(\frac{\mathrm{TP}}{\pi} + \frac{\mathrm{TN}}{1-\pi}\right) = \frac{(1-\pi)\mathrm{TP} + \pi\mathrm{TN}}{2\pi(1-\pi)},\ F_\beta = \frac{(1+\beta^2)\mathrm{TP}}{(1+\beta^2)\mathrm{TP} + \beta^2\mathrm{FN} + \mathrm{FP}} = \frac{(1+\beta^2)\mathrm{TP}}{\beta^2\pi + \gamma},$$

$$\mathrm{JAC} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}+\mathrm{FP}} = \frac{\mathrm{TP}}{\pi+\mathrm{FP}} = \frac{\mathrm{TP}}{\gamma+\mathrm{FN}},\quad \mathrm{WA} = \frac{w_1\mathrm{TP} + w_2\mathrm{TN}}{w_1\mathrm{TP} + w_2\mathrm{TN} + w_3\mathrm{FP} + w_4\mathrm{FN}}.$$

$$\pi = \mathrm{TP} + \mathrm{FN},\ \gamma = \mathrm{TP} + \mathrm{FP}$$

Metrics in ranking and recommendation

“Results show that improvements in RMSE often do not translate into [top-N ranking] accuracy improvements. In particular, a naive non-personalized algorithm can outperform some common recommendation approaches and almost match the accuracy of sophisticated algorithms”

P. Cremonesi, Y. Koren, and R. Turrin. "Performance of recommender algorithms on top-n recommendation tasks." Recsys, 2010.

sklearn.metrics : Metrics

Regression metrics

See the Regression metrics section of the user guide for further details.

metrics.explained_variance_score (y_true, y_pred)	Explained variance regres
metrics.mean_absolute_error (y_true, y_pred)	Mean absolute error regre
metrics.mean_squared_error (y_true, y_pred[, …])	Mean squared error regre
metrics.mean_squared_log_error (y_true, y_pred)	Mean squared logarithmic
metrics.median_absolute_error (y_true, y_pred)	Median absolute error reg
metrics.r2_score (y_true, y_pred[, …])	R^2 (coefficient of determ

Multilabel ranking metrics

See the Multilabel ranking metrics section of the user guide for further details

metrics.coverage_error (y_true, y_score[, …])	Coverage error mea
metrics.label_ranking_average_precision_score (…)	Compute ranking-ba
metrics.label_ranking_loss (y_true, y_score)	Compute Ranking lo

Clustering metrics

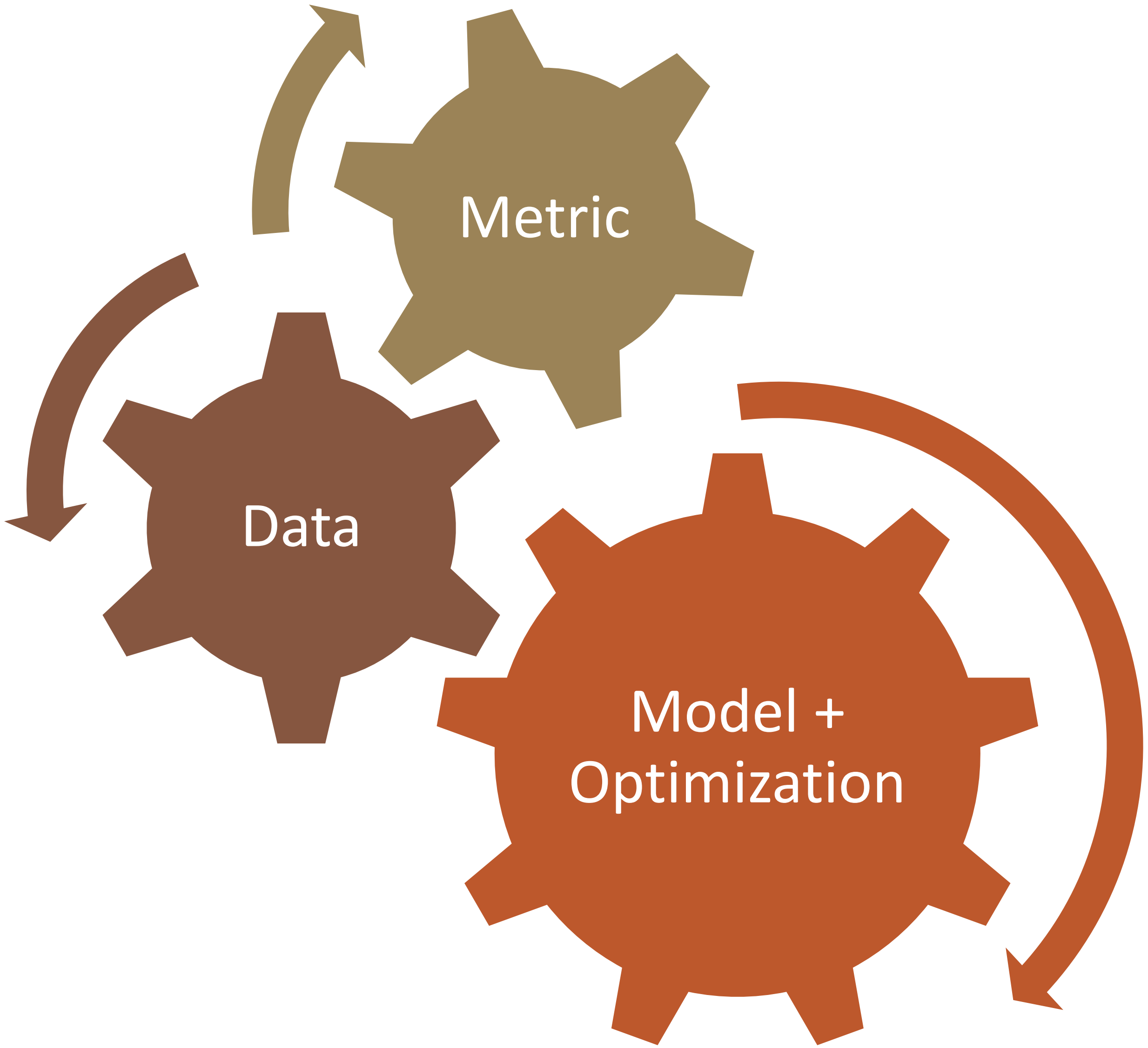
See the Clustering performance evaluation section of the user guide for further details.

The sklearn.metrics.cluster submodule contains evaluation metrics for cluster analysis results. There are two forms of evaluation:

- supervised, which uses a ground truth class values for each sample.
- unsupervised, which does not and measures the 'quality' of the model itself.

metrics.adjusted_mutual_info_score (…[, …])	Adjusted Mutual Information between two clusterings.
metrics.adjusted_rand_score (labels_true, …)	Rand index adjusted for chance.
metrics.calinski_harabaz_score (X, labels)	Compute the Calinski and Harabaz score.
metrics.davies_bouldin_score (X, labels)	Computes the Davies-Bouldin score.
metrics.completeness_score (labels_true, …)	Completeness metric of a cluster labeling given a ground truth.
metrics.cluster.contingency_matrix (…[, …])	Build a contingency matrix describing the relationship between labels.
metrics.fowlkes_mallows_score (labels_true, …)	Measure the similarity of two clusterings of a set of points.
metrics.homogeneity_completeness_v_measure (…)	Compute the homogeneity and completeness and V-Measure scores at once.
metrics.homogeneity_score (labels_true, …)	Homogeneity metric of a cluster labeling given a ground truth.
metrics.mutual_info_score (labels_true, …)	Mutual Information between two clusterings.
metrics.normalized_mutual_info_score (…[, …])	Normalized Mutual Information between two clusterings.
metrics.silhouette_score (X, labels[, …])	Compute the mean Silhouette Coefficient of all samples.
metrics.silhouette_samples (X, labels[, metric])	Compute the Silhouette Coefficient for each sample.
metrics.v_measure_score (labels_true, labels_pred)	V-measure cluster labeling given a ground truth.

Metric selection is challenging, but it's a required step in modern ML



Metric choice has a large impact on real-world machine learning performance.

1

Given a complex metric, how can we efficiently construct classifiers that (approximately) optimize it?

2

Given a new classification problem, which metric should you use to measure performance?

One simple trick…

A RE-WEIGHTING STRATEGY

Multiclass classification

		Ground truth			
		Y = 1	Y = 2	…	Y=K
Predicted	h(x) = 1	C_{11}	C_{12}		C_{1K}
	h(x) = 2	C_{21}	C_{22}		C_{2K}
	⋮				…
	h(x) = K	C_{K1}	C_{K2}		C_{KK}

$$C(h)$$

Standard metric is Accuracy

$$\phi(h) = c_{11} + c_{22} + \ldots + c_{KK}$$

$$= \langle I, C(h) \rangle$$

$$s_i(x) \approx p(y = i|x)$$

e.g. logistic regression, RF, DNN, …

$$h(x) = \operatorname*{argmax}_{i \in [K]} s_i$$

Standard Prediction Strategy

$$\max_h \langle A, C(h) \rangle$$

$$s_i(x) \approx p(y = i | x)$$

e.g. logistic regression, RF, DNN, …

$$h(x) = \underset{i \in [K]}{\operatorname{argmax}} \, \mathbf{a}_i^\top \mathbf{s}(x)$$

Proposed Postprocessing Strategy

Narasimhan, H., et al. "Consistent multiclass algorithms for complex performance measures." ICML. 2015.

A small experiment

$$\eta_k(x) \propto e^{\mathbf{w}_k^\top \mathbf{x}}$$

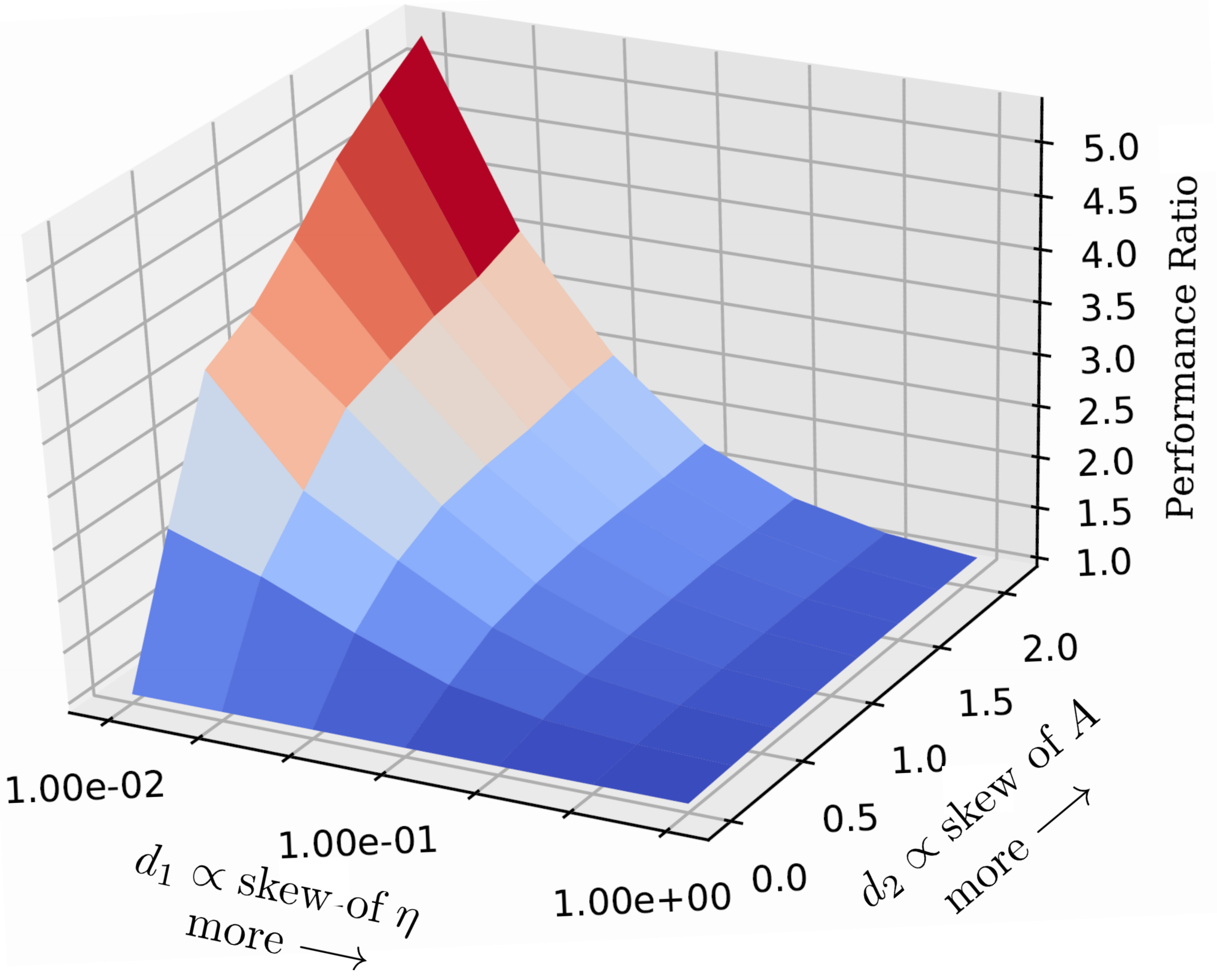
$$\mathbf{x} \sim \mathcal{N}(0, \mathbf{I}); \ \mathbf{w}_k = d_1 |k - K| \mathbf{1}$$

$$A_{j,j} = e^{-d_2 j}$$

1. Generate random data from model
2. Fit a logistic regression model
3. Post-process predictions

$$\text{Performance Ratio} = \frac{\text{Perf. of weighted postprocess}}{\text{Perf. of std. prediction}}$$

Simple re-weighting can have a huge effect!



$$\max_h \psi\big(C(h)\big)$$

$$s_i \approx p(y = i|x)$$

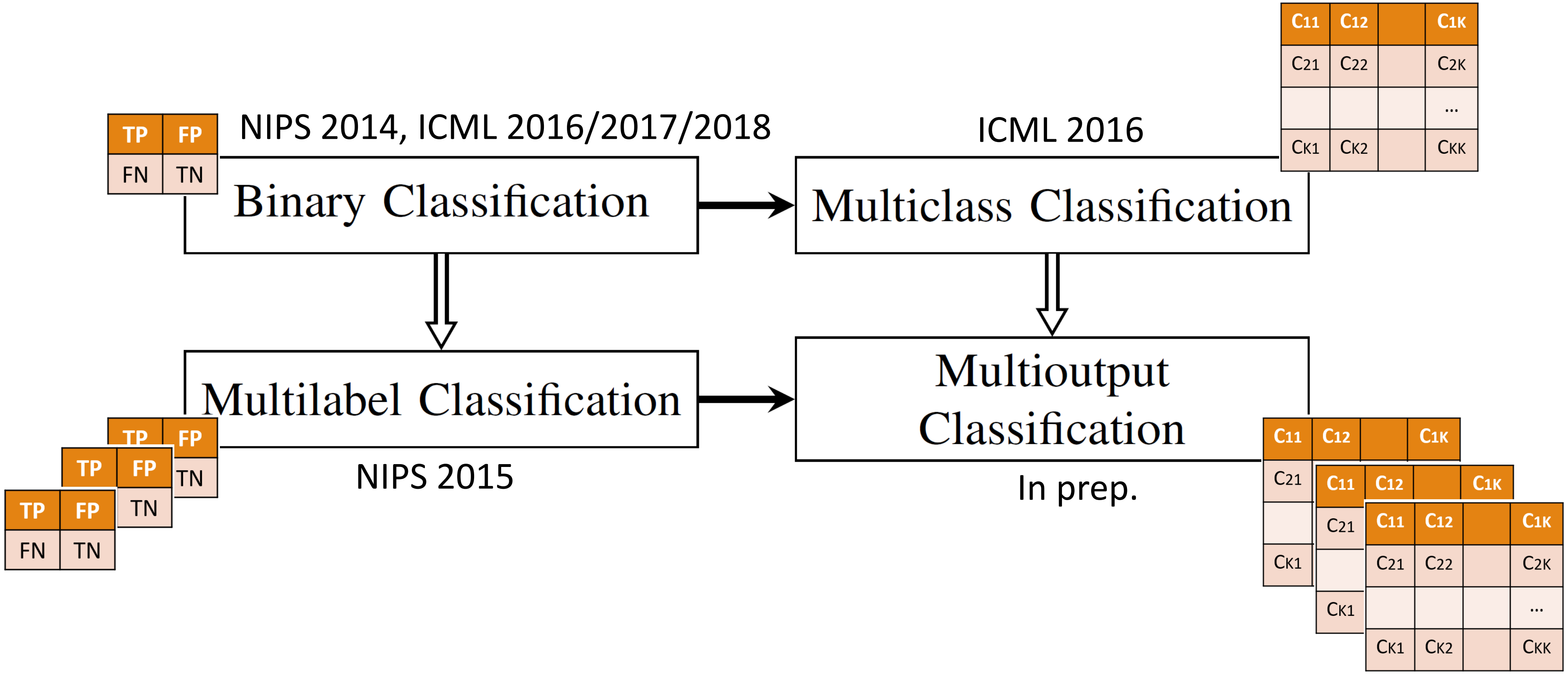
any calibrated classifier

$$h(x) = \operatorname*{argmax}_{i\in[K]} \mathbf{b}_i^\top \mathbf{s}$$

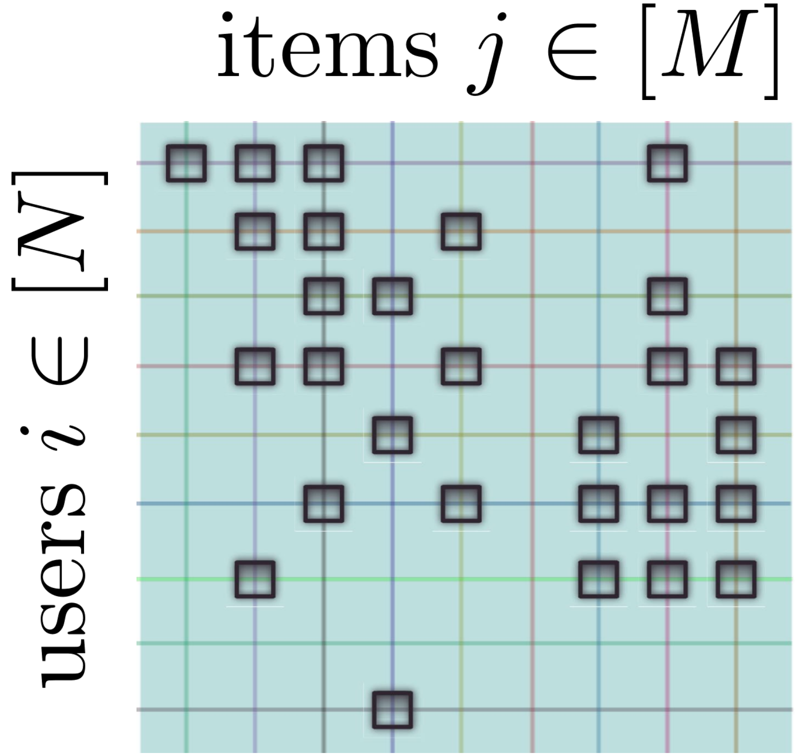
$$B = \nabla\psi|_{C=C^*}$$

Same strategy works for more complex metrics

Applies to more general settings



An application to recommender systems



User assigns rating to each item.

$$r_{i,j} \in [K]$$

Solve this as simultaneous (over items) multiclass classification problem i.e. multioutput classification

$$\phi(h) = \sum_{i=1}^{K} \sum_{j=1}^{K} |i - j| C_{i,j}$$

Postprocessed OrdRec

$$s_i(x) \approx p(y = i|x)$$

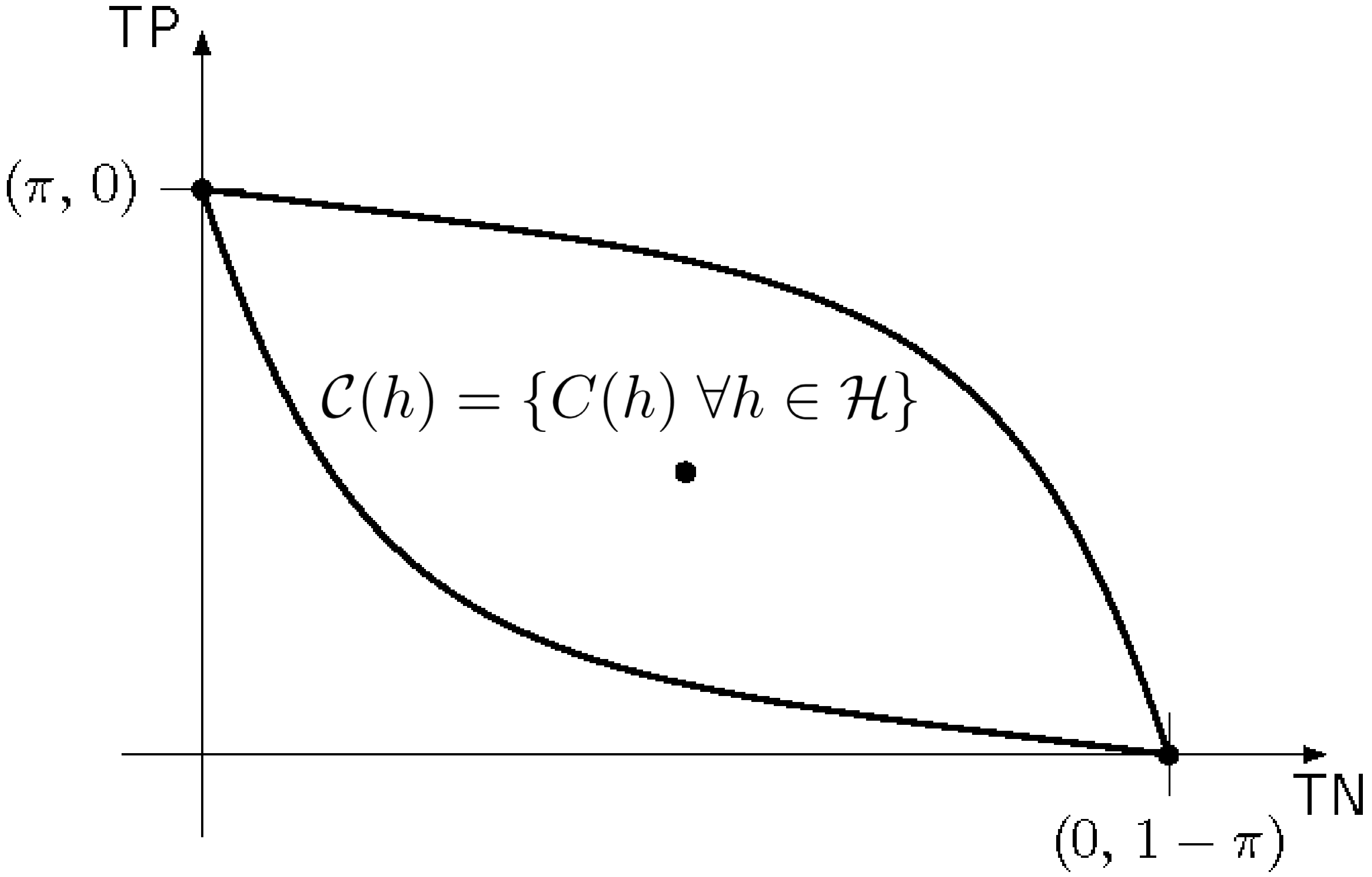
$$\operatorname*{argmax}_{i \in [K]} \mathbf{s}_i(x) \qquad \operatorname*{argmax}_{i \in [K]} \mathbf{a}_i^\top \mathbf{s}(x)$$

Koren, Yehuda, and Joe Sill. "OrdRec: an ordinal model for predicting personalized item rating distributions." *Recsys* 2011.

AVERAGE	ORDREC	C-ORDREC
MICRO	0.8603±0.0010	**0.8640±0.0009**
MACRO	0.8577±0.0032	**0.8643±0.0022**
INSTANCE	0.8565±0.0014	**0.8619±0.0011**

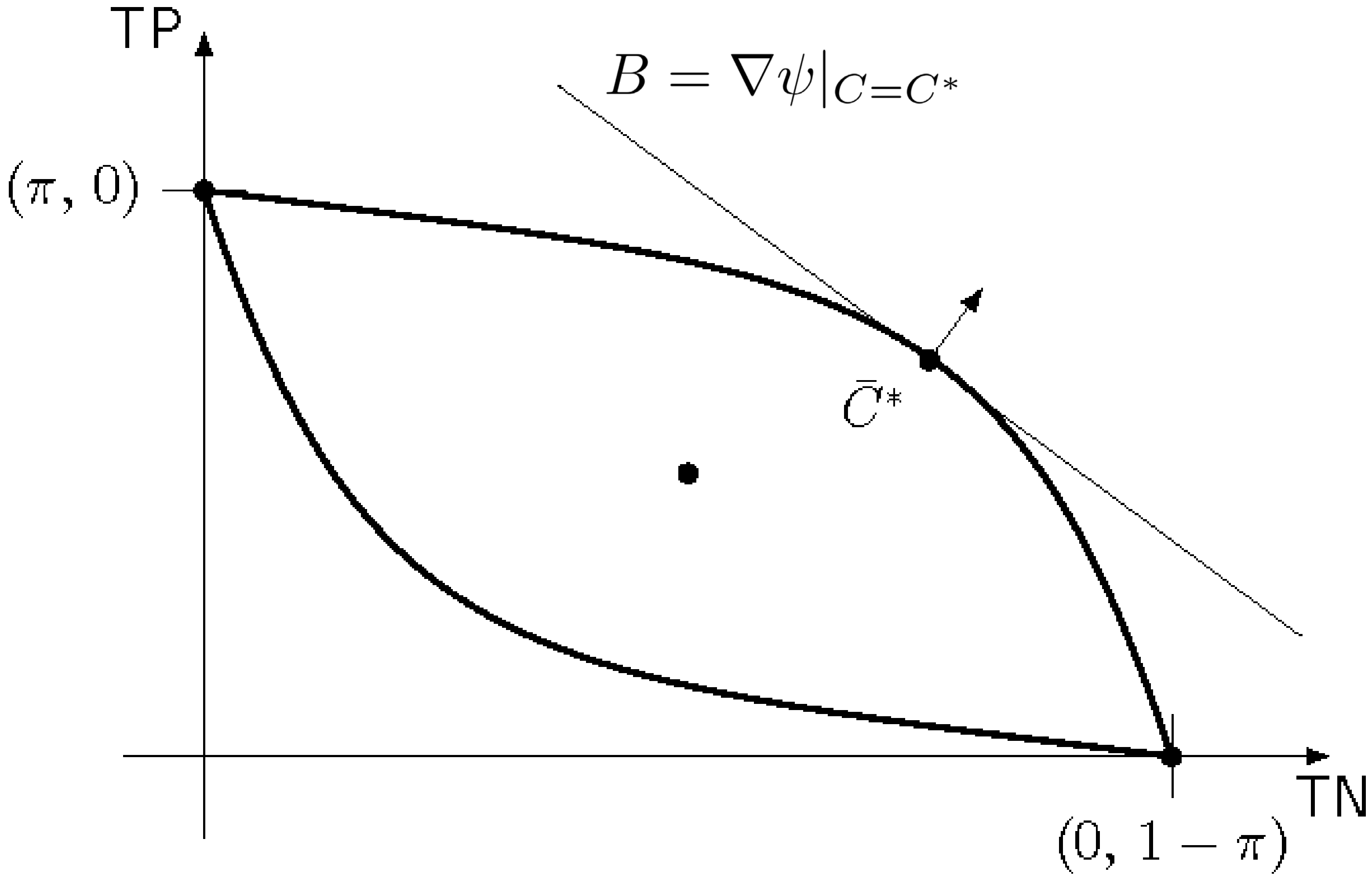
When & Why does re-weighting work?

THE GEOMETRY OF CONFUSION



$$\mathrm{TP} + \mathrm{FN} = \pi,\ \mathrm{TN} + \mathrm{FP} = 1 - \pi$$

- Set of feasible confusion matrices is a bounded convex set
- Optimization properties will depend on how gradient field of the metric interacts with the feasible set
- Any monotonic metric will be optimized at the boundary



- All points on the boundary are determined by the support function
- This characterization is exhaustive i.e. characterizes ALL metrics that are consistently optimizable via linear post-processing

$$s_i(x) \to p(y = i|x)$$

$$B \to \nabla\psi|_{C=C^*}$$

$$\operatorname*{argmax}_{i\in[K]} \mathbf{b}_i^\top \mathbf{s} \to h^*(x)$$

This classification strategy is consistent

Binary classification with general metrics

$$s(x) \approx p(y = i | x)$$

Logistic regression w/ MLE
Holder densities w/ kernel approx.

$$g_\delta(x) = \mathrm{sign}\left(s(x) - \hat{\delta}\right)$$

Plug-in classifier

$$\hat{h}_n(x) = \underset{\delta \in [0,1]}{\mathrm{argmax}}\ \phi_n\left(g_\delta\right)$$

Threshold search

$$|\phi(h^*) - \phi(\hat{h}_n)| \leq O\left(\frac{\log n}{n}\right)$$

Yan, **K.**, Zhong, Ravikumar (2018)

Which metric should you use?

THE BINARY CLASSIFICATION CASE

Recall: Lots of real world examples

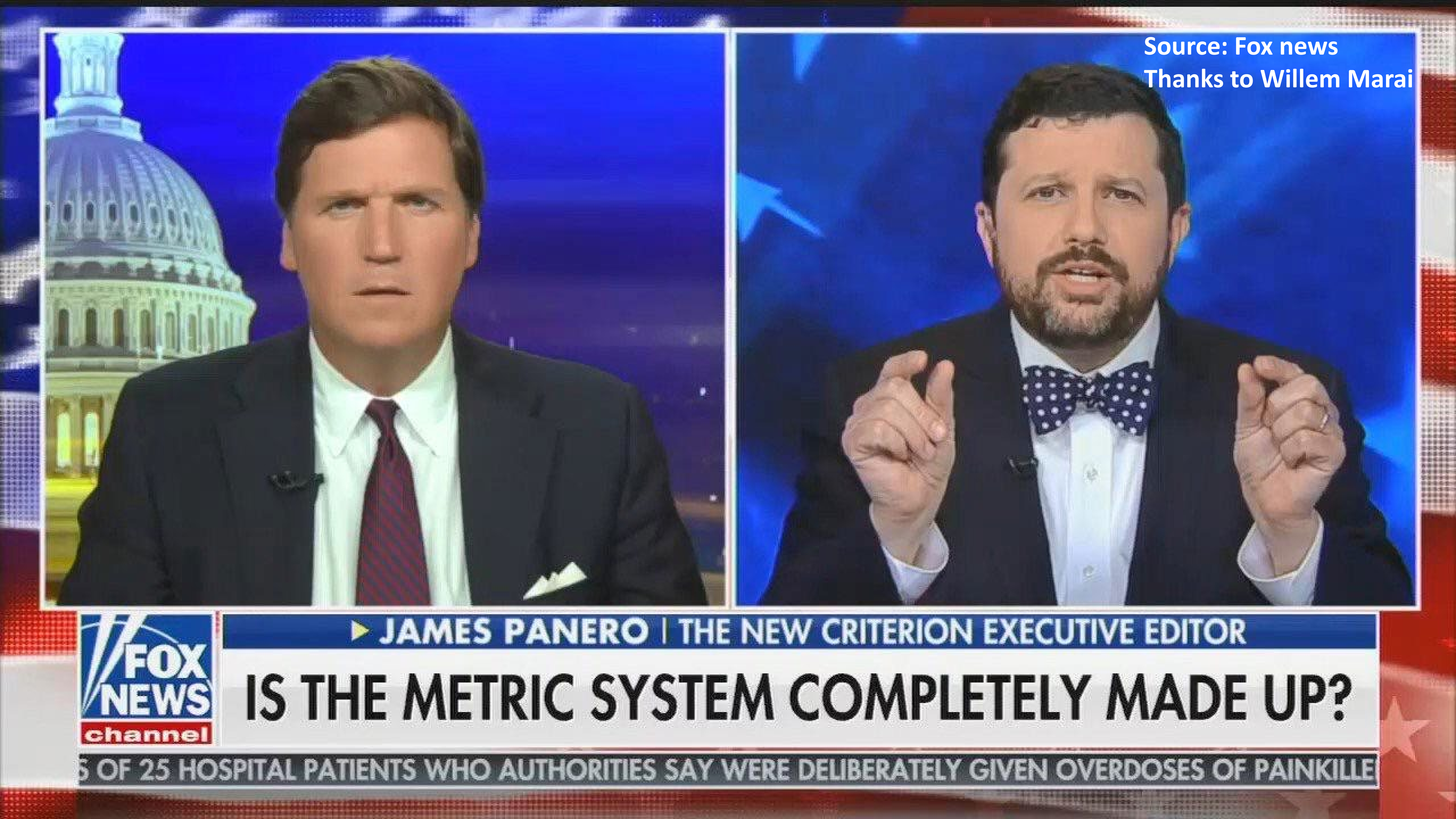
$$\phi(h) = a_1\mathrm{TP} + a_2\mathrm{FP} + a_3\mathrm{FN} + a_4\mathrm{TN}$$

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}},\ \mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{FP}+\mathrm{TN}},\ \mathrm{Prec} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}},\ \mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{FN}+\mathrm{TP}},\ \mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FN}}.$$

$$\mathrm{AM} = \frac{1}{2}\left(\frac{\mathrm{TP}}{\pi} + \frac{\mathrm{TN}}{1-\pi}\right) = \frac{(1-\pi)\mathrm{TP} + \pi\mathrm{TN}}{2\pi(1-\pi)},\ F_\beta = \frac{(1+\beta^2)\mathrm{TP}}{(1+\beta^2)\mathrm{TP} + \beta^2\mathrm{FN} + \mathrm{FP}} = \frac{(1+\beta^2)\mathrm{TP}}{\beta^2\pi + \gamma},$$

$$\mathrm{JAC} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}+\mathrm{FP}} = \frac{\mathrm{TP}}{\pi + \mathrm{FP}} = \frac{\mathrm{TP}}{\gamma + \mathrm{FN}},\quad \mathrm{WA} = \frac{w_1\mathrm{TP} + w_2\mathrm{TN}}{w_1\mathrm{TP} + w_2\mathrm{TN} + w_3\mathrm{FP} + w_4\mathrm{FN}}.$$

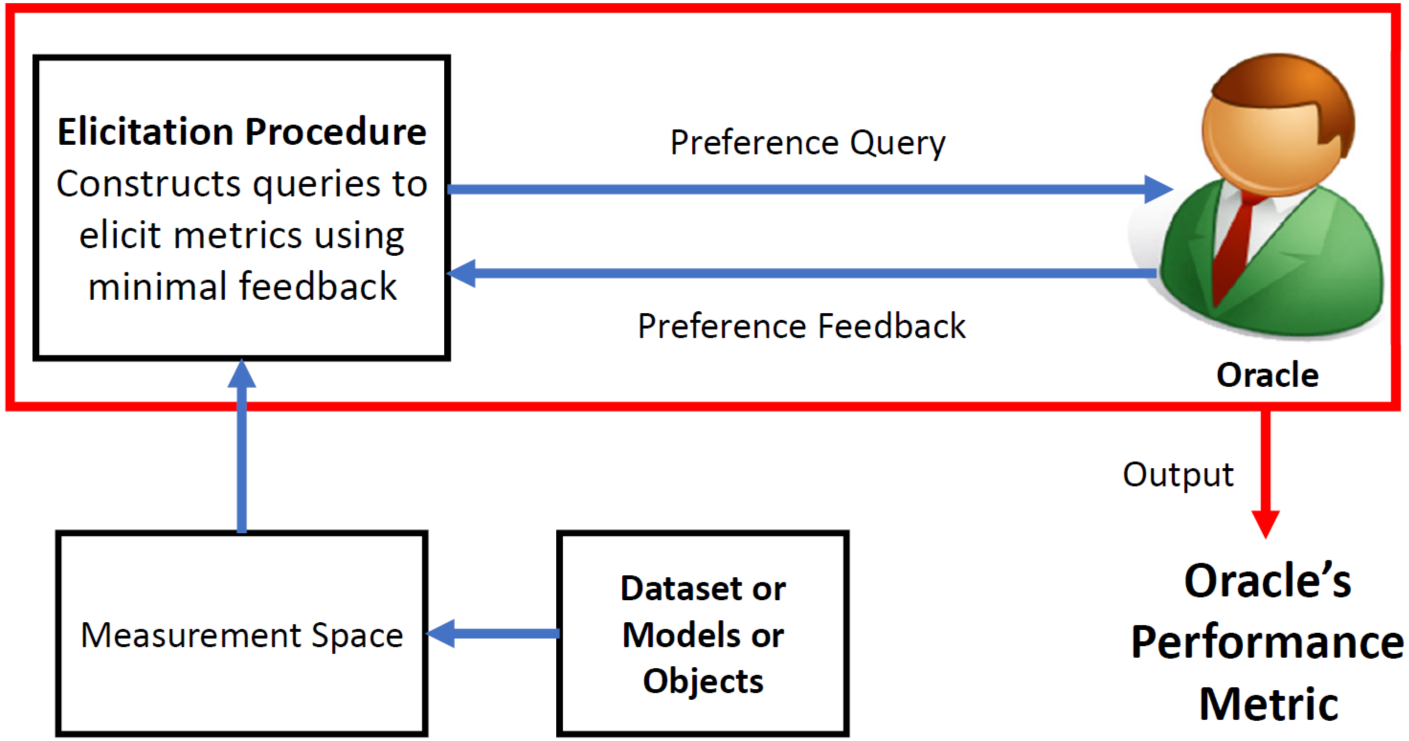
Source: Fox news
Thanks to Willem Marai



Limited formal guidance

Academia:
Use the standard metric in your application area
- Accuracy
- Top-K accuracy
- F1 measure

Industry:
Hire a consultant or economist
- User survey
- A/B tests

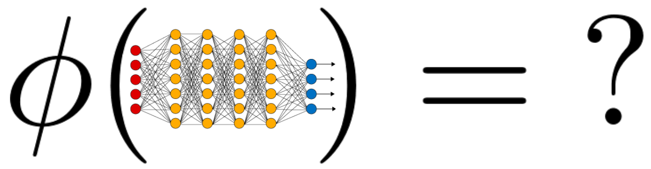


Metric Elicitation

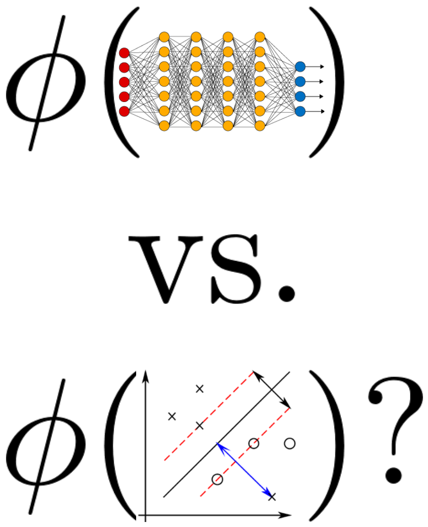
QUERY AN EXPERT
(OR USER POPULATION)
TO QUANTIFY UTILITY

Querying the oracle

Humans (even domain experts) are often inaccurate when asked to quantify value



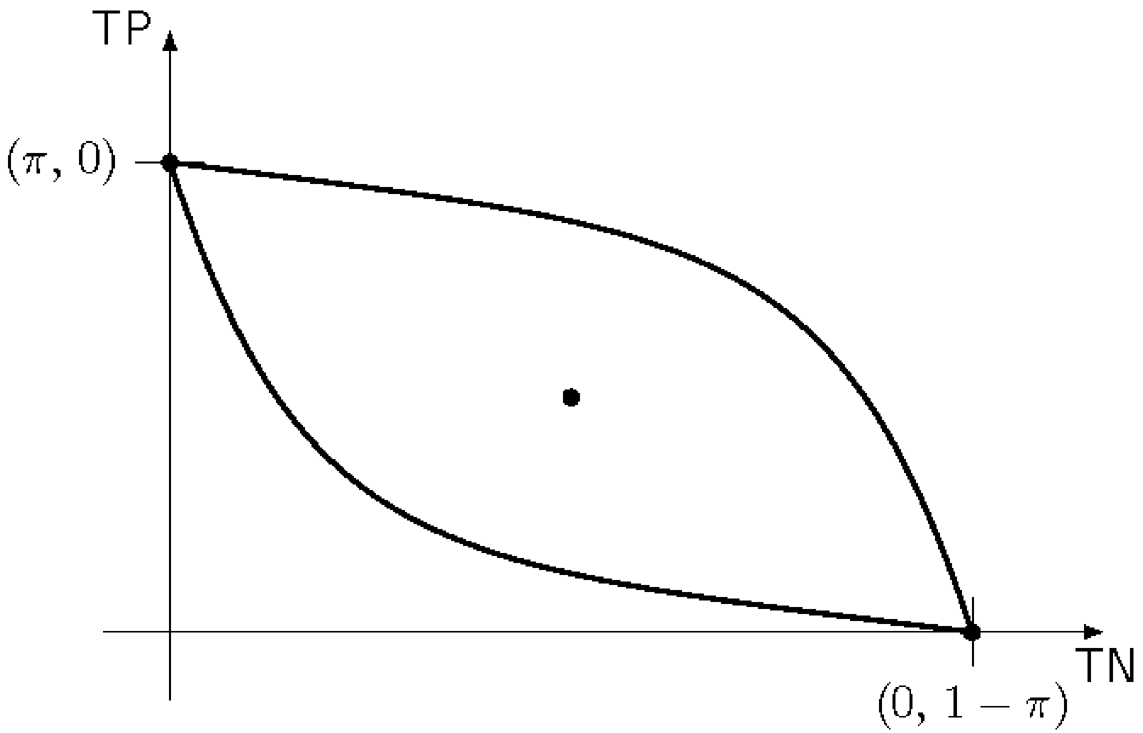
We're much more accurate when asked for pairwise preferences



TOO MANY QUERIES
MAY RESULT IN FATIGUE

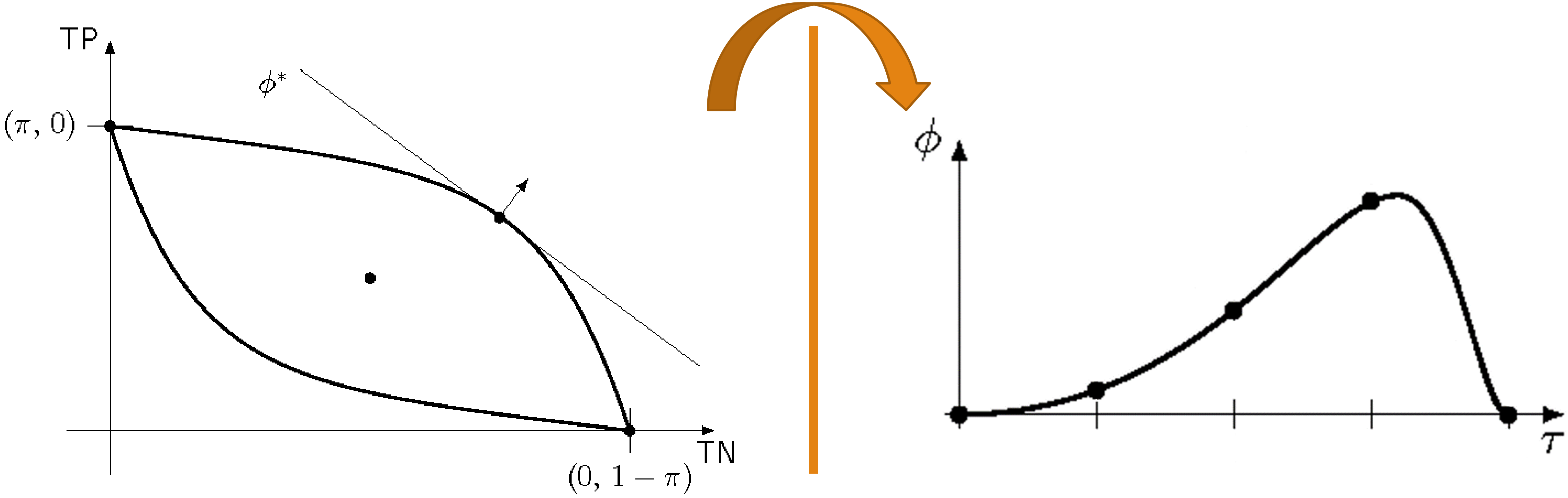
Goal:
accurately elicit metric using fewest pairwise queries

Back to binary classification with weighted metrics



$$\mathrm{TP} + \mathrm{FN} = \pi,\ \mathrm{TN} + \mathrm{FP} = 1 - \pi$$

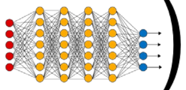
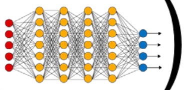
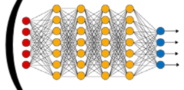
- Set of feasible confusion matrices is a bounded convex set (related to the ROC curve)
- Weighted metrics correspond to linear functions
- Thus, eliciting weights requires finding a tangent to the boundary curve (equiv. point along the boundary) that recovers oracle tradeoff



unroll the boundary curve… find the maximum point

Metric recovery with finite queries

This algorithm provably recovers the oracle's weighted metric:

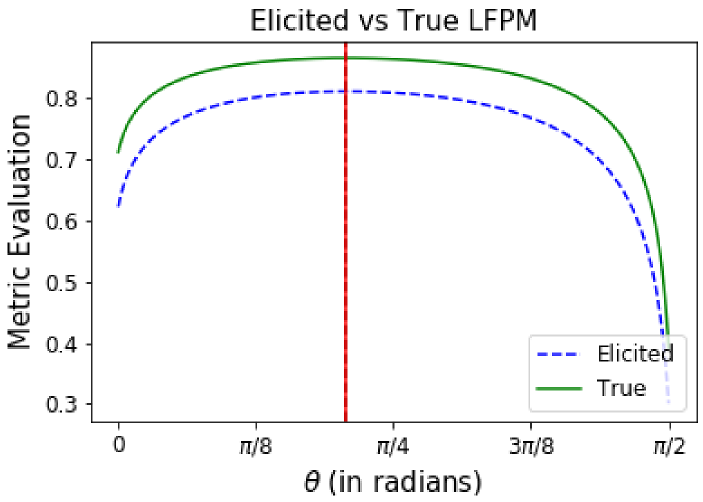
$\phi^*($  $) = 1 - (a_1^* \mathrm{FP}($  $) + a_2^* \mathrm{FN}($  $))$

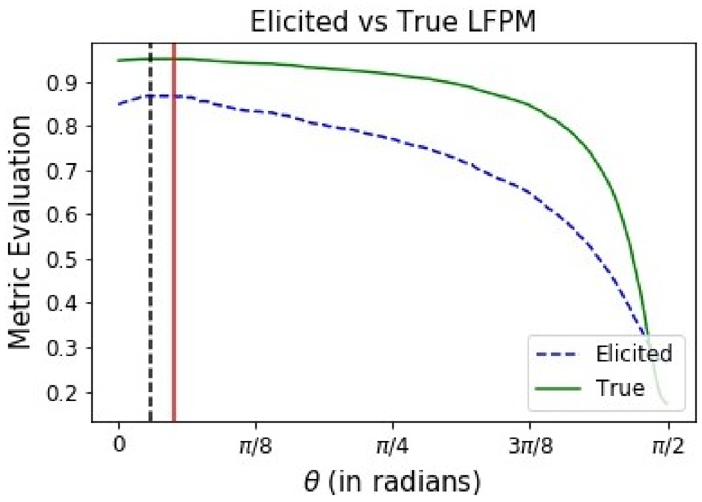
Guaranteed to be ϵ accurate after $\mathcal{O}\left(\log(\frac{1}{\epsilon})\right)$ queries

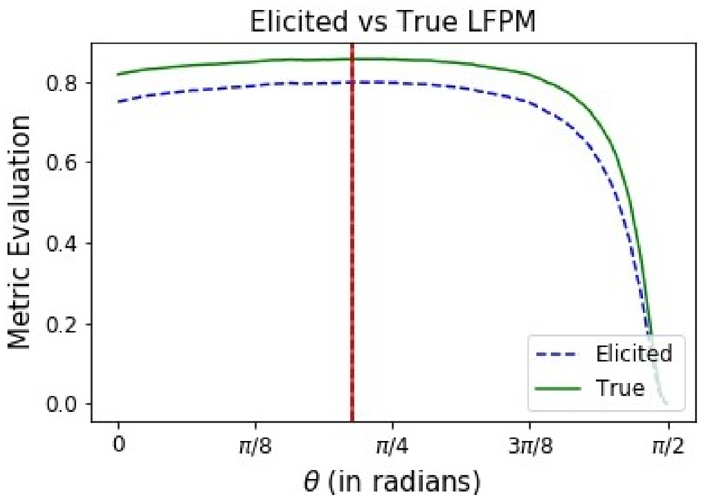
Achieves the theoretical optimal elicitation rate

Stable to system noise e.g. noisy responses from the oracle

Hiranandani, Boodaghians, Mehta, **K.** (2018)







Empirical evaluation

Metric selection is a crucial step in constructing effective machine learning models

Simple algorithms can be used to elicit an oracle metric (i.e. from an expert or user population) using a few queries

Conclusion

Metric choice has a large impact on real-world machine learning performance.

1

Re-weighted post-processing is efficient for optimizing complex metrics.

2

Can reduce metric elicitation for binary classifiers to binary search with bounded query complexity.

01

Extensions to other machine learning problems e.g. ranking, regression, probabilistic density modeling …

02

Query mechanisms, HCI

03

Noise tolerance, robust elicitation

04

Applications to addressing bias and fairness in machine learning

Lots of work in progress

Thank you

QUESTIONS?

SANMI@ILLINOIS.EDU
@SANMIKOYEJO