Omnipredictors[1]: One Predictor to Rule Them All

Heavily adapted from P. Gopalan's Talk at IAS

J. Setpal

April 18, 2024

[1]Gopalan, Kalai, Reingold, Sharan, Wieder

ERM Synopsis

We'll start with an *overview* of supervised learning paradigm:

1. Dataset $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^{N}$; $1 \ll N \ll \infty$; $\mathcal{D} \sim$ "Real World"

ERM Synopsis

We'll start with an *overview* of supervised learning paradigm:

1. Dataset $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^N$; $1 \ll N \ll \infty$; $\mathcal{D} \sim$ "Real World"
2. Parameterized model $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$

ERM Synopsis

We'll start with an *overview* of supervised learning paradigm:

1. Dataset $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^{N}$; $1 \ll N \ll \infty$; $\mathcal{D} \sim$ "Real World"
2. Parameterized model $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$
3. Objective: Train $\boldsymbol{\theta}$ s.t. $f_{\boldsymbol{\theta}}(x) = \hat{y} \approx y$

ERM Synopsis

We'll start with an *overview* of supervised learning paradigm:

1. Dataset $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^{N}$; $1 \ll N \ll \infty$; $\mathcal{D} \sim$ "Real World"
2. Parameterized model $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$
3. Objective: Train $\boldsymbol{\theta}$ s.t. $f_{\boldsymbol{\theta}}(x) = \hat{y} \approx y$

How do we mathematically encode $\hat{y} \approx y$?

ERM Synopsis

We'll start with an *overview* of supervised learning paradigm:

1. Dataset $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^{N}$; $1 \ll N \ll \infty$; $\mathcal{D} \sim$ "Real World"
2. Parameterized model $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$
3. Objective: Train $\boldsymbol{\theta}$ s.t. $f_{\boldsymbol{\theta}}(x) = \hat{y} \approx y$

How do we mathematically encode $\hat{y} \approx y$? A loss (distance) function!

4. Loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$; $L(\hat{y}, y) \approx 0$ iff $\hat{y} \approx y$; L is continuous.

ERM Synopsis

We'll start with an *overview* of supervised learning paradigm:

1. Dataset $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^N$; $1 \ll N \ll \infty$; $\mathcal{D} \sim$ "Real World"
2. Parameterized model $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$
3. Objective: Train $\boldsymbol{\theta}$ s.t. $f_{\boldsymbol{\theta}}(x) = \hat{y} \approx y$

How do we mathematically encode $\hat{y} \approx y$? A loss (distance) function!

4. Loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$; $L(\hat{y}, y) \approx 0$ iff $\hat{y} \approx y$; L is continuous.

How can we update our weights to optimize against this loss function?

ERM Synopsis

We'll start with an *overview* of supervised learning paradigm:

1. Dataset $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^{N}$; $1 \ll N \ll \infty$; $\mathcal{D} \sim$ "Real World"
2. Parameterized model $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$
3. Objective: Train $\boldsymbol{\theta}$ s.t. $f_{\boldsymbol{\theta}}(x) = \hat{y} \approx y$

How do we mathematically encode $\hat{y} \approx y$? A loss (distance) function!

4. Loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$; $L(\hat{y}, y) \approx 0$ iff $\hat{y} \approx y$; L is continuous.

How can we update our weights to optimize against this loss function?

5. Gradient Descent! $\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \cdot \frac{\partial L}{\partial \boldsymbol{\theta}}$

Iterate (5) until convergence.

ERM Synopsis

We'll start with an *overview* of supervised learning paradigm:

1. Dataset $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^{N}$; $1 \ll N \ll \infty$; $\mathcal{D} \sim$ "Real World"
2. Parameterized model $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$
3. Objective: Train $\boldsymbol{\theta}$ s.t. $f_{\boldsymbol{\theta}}(x) = \hat{y} \approx y$

How do we mathematically encode $\hat{y} \approx y$? A loss (distance) function!

4. Loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$; $L(\hat{y}, y) \approx 0$ iff $\hat{y} \approx y$; L is continuous.

How can we update our weights to optimize against this loss function?

5. Gradient Descent! $\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \cdot \frac{\partial L}{\partial \boldsymbol{\theta}}$

Iterate (5) until convergence.

L is minimized over $\mathcal{D}$, not over the real world.

ERM Synopsis

We'll start with an *overview* of supervised learning paradigm:

1. Dataset $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^{N}$; $1 \ll N \ll \infty$; $\mathcal{D} \sim$ "Real World"
2. Parameterized model $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$
3. Objective: Train $\boldsymbol{\theta}$ s.t. $f_{\boldsymbol{\theta}}(x) = \hat{y} \approx y$

How do we mathematically encode $\hat{y} \approx y$? A loss (distance) function!

4. Loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$; $L(\hat{y}, y) \approx 0$ iff $\hat{y} \approx y$; L is continuous.

How can we update our weights to optimize against this loss function?

5. Gradient Descent! $\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \cdot \frac{\partial L}{\partial \boldsymbol{\theta}}$

Iterate (5) until convergence.

L is minimized over $\mathcal{D}$, not over the real world. This is **empirical risk**:

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} L(f_{\boldsymbol{\theta}}(x_i), y_i) \tag{1}$$

Agnostic Boosting

Probably Approximately Correct (PAC) Learning is a *generalization* of ERM.

Agnostic Boosting

Probably Approximately Correct (PAC) Learning is a *generalization* of ERM.

We define a PAC-learnable concept C if a learner L can with $\Pr = 1 - \delta$ output hypothesis $h \in \mathcal{H}$ s.t. $\text{error}_{\mathcal{D}}(h) \leq \epsilon$ with required samples $|\mathcal{D}| \in f(\epsilon, \delta, n) = \frac{1}{\epsilon^a} + \frac{1}{\delta^b} + |\mathcal{H}|^c$.

Agnostic Boosting

Probably Approximately Correct (PAC) Learning is a *generalization* of ERM.

We define a PAC-learnable concept C if a learner L can with $\Pr = 1 - \delta$ output hypothesis $h \in \mathcal{H}$ s.t. $\text{error}_{\mathcal{D}}(h) \leq \epsilon$ with required samples $|\mathcal{D}| \in f(\epsilon, \delta, n) = \frac{1}{\epsilon^a} + \frac{1}{\delta^b} + |\mathcal{H}|^c$.

One interpretation of PAC allows an arbitrary concept function, that learns by comparing error in the learner's hypothesis against the best predictor in a pre-specified comparison class of predictors.

Agnostic Boosting

Probably Approximately Correct (PAC) Learning is a *generalization* of ERM.

We define a PAC-learnable concept C if a learner L can with $\Pr = 1 - \delta$ output hypothesis $h \in \mathcal{H}$ s.t. $\text{error}_{\mathcal{D}}(h) \leq \epsilon$ with required samples $|\mathcal{D}| \in f(\epsilon, \delta, n) = \frac{1}{\epsilon^a} + \frac{1}{\delta^b} + |\mathcal{H}|^c$.

One interpretation of PAC allows an arbitrary concept function, that learns by comparing error in the learner's hypothesis against the best predictor in a pre-specified comparison class of predictors. This is **Agnostic Learning**.

Agnostic Boosting

Probably Approximately Correct (PAC) Learning is a *generalization* of ERM.

We define a PAC-learnable concept C if a learner L can with $\Pr = 1 - \delta$ output hypothesis $h \in \mathcal{H}$ s.t. $\text{error}_{\mathcal{D}}(h) \leq \epsilon$ with required samples $|\mathcal{D}| \in f(\epsilon, \delta, n) = \frac{1}{\epsilon^a} + \frac{1}{\delta^b} + |\mathcal{H}|^c$.

One interpretation of PAC allows an arbitrary concept function, that learns by comparing error in the learner's hypothesis against the best predictor in a pre-specified comparison class of predictors. This is **Agnostic Learning**.

Boosting a weak agnostic learner is a critical aspect of the Omnipredictors approach to learning "multicalibrated partitions" (we'll get to this soon).

Challenge Statement

Problem: Different loss functions typically have divergent geometries.

[2] usually, local

Challenge Statement

Problem: Different loss functions typically have divergent geometries.

This means gradient descent obtains a **different**[2] **optima** for two such functions, despite sharing minima for $\boldsymbol{\theta}$ s.t. $\hat{y} \approx y$.

[2] usually, local

Challenge Statement

Problem: Different loss functions typically have divergent geometries.

This means gradient descent obtains a **different**[2] **optima** for two such functions, despite sharing minima for $\boldsymbol{\theta}$ s.t. $\hat{y} \approx y$.

Let's evaluate this empirically on ℓ_1 and ℓ_2 losses, which optimize for median and mean respectively:

$$\ell_1 = |y - \hat{y}|,\ \ell_2 = (y - \hat{y})^2 \tag{2}$$

$$x \sim f(\epsilon \sim \mathcal{U}[0,1]) := \begin{cases} 0 & \epsilon \leq 0.4 \\ \mathcal{U}[0.8, 1] & \text{otherwise} \end{cases} \tag{3}$$

[2] usually, local

Challenge Statement

Problem: Different loss functions typically have divergent geometries.

This means gradient descent obtains a **different**[2] **optima** for two such functions, despite sharing minima for $\boldsymbol{\theta}$ s.t. $\hat{y} \approx y$.

Let's evaluate this empirically on ℓ_1 and ℓ_2 losses, which optimize for median and mean respectively:

$$\ell_1 = |y - \hat{y}|,\ \ell_2 = (y - \hat{y})^2 \tag{2}$$

$$x \sim f(\epsilon \sim \mathcal{U}[0,1]) := \begin{cases} 0 & \epsilon \leq 0.4 \\ \mathcal{U}[0.8, 1] & \text{otherwise} \end{cases} \tag{3}$$

Omnipredictors provides a framework for rigorous guarantees, deriving $\tilde{p} \approx p^*$: a predictor that is able to *simultaneously minimize* a family of convex loss functions.

[2] usually, local

Accuracy in Expectation & Calibration

We subject the model's predicted probabilities to 'sanity checks'

Accuracy in Expectation & Calibration

We subject the model's predicted probabilities to 'sanity checks' – these are classic **interpretability notions**:

a. $\tilde{p}$ is accurate in expectation if:

$$\mathbb{E}[\tilde{p}(x)] = \mathbb{E}[y] \tag{4}$$

Accuracy in Expectation & Calibration

We subject the model's predicted probabilities to 'sanity checks' – these are classic **interpretability notions**:

a. $\tilde{p}$ is <u>accurate in expectation</u> if:

$$\mathbb{E}[\tilde{p}(x)] = \mathbb{E}[y] \tag{4}$$

b. $\tilde{p}$ is <u>calibrated</u> if:

$$\mathbb{E}[y|\tilde{p}(x)] = \tilde{p}(x) \tag{5}$$

Accuracy in Expectation & Calibration

We subject the model's predicted probabilities to 'sanity checks' – these are classic **interpretability notions**:

a. $\tilde{p}$ is accurate in expectation if:

$$\mathbb{E}[\tilde{p}(x)] = \mathbb{E}[y] \tag{4}$$

b. $\tilde{p}$ is calibrated if:

$$\mathbb{E}[y|\tilde{p}(x)] = \tilde{p}(x) \tag{5}$$

These may be **orthogonal to model performance**.

Accuracy in Expectation & Calibration

We subject the model's predicted probabilities to 'sanity checks' – these are classic **interpretability notions**:

a. $\tilde{p}$ is <u>accurate in expectation</u> if:

$$\mathbb{E}[\tilde{p}(x)] = \mathbb{E}[y] \tag{4}$$

b. $\tilde{p}$ is <u>calibrated</u> if:

$$\mathbb{E}[y|\tilde{p}(x)] = \tilde{p}(x) \tag{5}$$

These may be **orthogonal to model performance**.

"If you posit a more complex view of the world, I will subject you to a more rigorous test." – P. Gopalan.

Multigroup Fairness

We can split $\mathcal{D}$ into various *subgroups* based on **shared characteristics**. These can be explicit or implicit (i.e. subgroups we don't know of):

	Group-1	**Group-2**	**Group-3**	**Group-4**
Accuracy	0.9593	0.6249	0.3157	0.2664
Loss	0.0021	0.4102	1.3457	1.7664
Proportion	0.9	0.08	0.0075	0.0025

Multigroup Fairness

We can split $\mathcal{D}$ into various *subgroups* based on **shared characteristics**. These can be explicit or implicit (i.e. subgroups we don't know of):

	Group-1	**Group-2**	**Group-3**	**Group-4**
Accuracy	0.9593	0.6249	0.3157	0.2664
Loss	0.0021	0.4102	1.3457	1.7664
Proportion	0.9	0.08	0.0075	0.0025

Empirical Risk is *only* 0.0492, but inference is unreliable for subgroups 2-4.

Multigroup Fairness

We can split $\mathcal{D}$ into various *subgroups* based on **shared characteristics**. These can be explicit or implicit (i.e. subgroups we don't know of):

	Group-1	**Group-2**	**Group-3**	**Group-4**
Accuracy	0.9593	0.6249	0.3157	0.2664
Loss	0.0021	0.4102	1.3457	1.7664
Proportion	0.9	0.08	0.0075	0.0025

Empirical Risk is *only* 0.0492, but inference is unreliable for subgroups 2-4.

One notion of fairness stipulates equal risk for every subgroup. However, finding subgroups is hard for <u>high-dimensional data</u>.

Multiaccuracy & Multicalibration

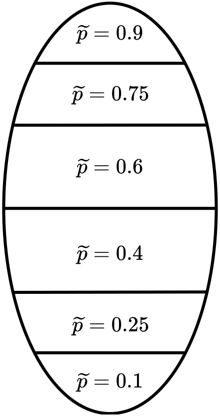
Can we elicit more information from our model while *retaining* calibration?

Multiaccuracy & Multicalibration

Can we elicit more information from our model while *retaining* calibration? Here, we introduce **multiaccuracy** and **multicalibration**.

Multiaccuracy & Multicalibration

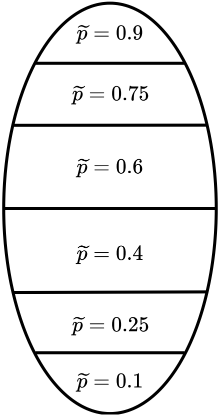
Can we elicit more information from our model while *retaining* calibration? Here, we introduce **multiaccuracy** and **multicalibration**.



Let $C = \{c : \mathcal{X} \to [-1, 1]\}$ be a collection of subsets, generalized as **real-valued functions**.

Multiaccuracy & Multicalibration

Can we elicit more information from our model while *retaining* calibration? Here, we introduce **multiaccuracy** and **multicalibration**.



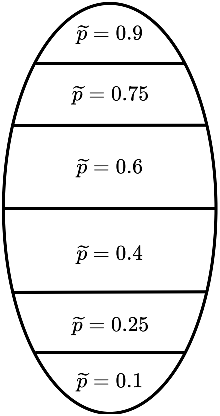
Let $C = \{c : \mathcal{X} \to [-1, 1]\}$ be a collection of subsets, generalized as **real-valued functions**.

$\tilde{p}$ is (C, α)-multiaccurate if:

$$\max_{c \in C} |\mathbb{E}[c(x)(y - \tilde{p}(x))]| \leq \alpha \tag{6}$$

Multiaccuracy & Multicalibration

Can we elicit more information from our model while *retaining* calibration? Here, we introduce **multiaccuracy** and **multicalibration**.



Let $C = \{c : \mathcal{X} \to [-1, 1]\}$ be a collection of subsets, generalized as **real-valued functions**.

$\tilde{p}$ is (C, α)-multiaccurate if:

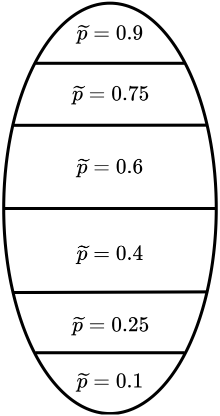
$$\max_{c \in C} |\mathbb{E}[c(x)(y - \tilde{p}(x))]| \leq \alpha \tag{6}$$

$\tilde{p}$ is (C, α)-multicalibrated if:

$$\max_{c \in C} \mathbb{E}[|\mathbb{E}[c(x)(y - \tilde{p}(x))]|] \leq \alpha \tag{7}$$

Multiaccuracy & Multicalibration

Can we elicit more information from our model while *retaining* calibration? Here, we introduce **multiaccuracy** and **multicalibration**.



Let $C = \{c : \mathcal{X} \to [-1, 1]\}$ be a collection of subsets, generalized as **real-valued functions**.

$\tilde{p}$ is (C, α)-multiaccurate if:

$$\max_{c \in C} |\mathbb{E}[c(x)(y - \tilde{p}(x))]| \leq \alpha \tag{6}$$

$\tilde{p}$ is (C, α)-multicalibrated if:

$$\max_{c \in C} \mathbb{E}[|\mathbb{E}[c(x)(y - \tilde{p}(x))]|] \leq \alpha \tag{7}$$

If we can find correlations with the error, there's some advantage to be gained. We enforce multicalibration to train the **weak agnostic learner**.

Omnipredictors

If we know p^*, it is easy for us take take the optimal action.

Omnipredictors

If we know p^*, it is easy for us take take the optimal action.

For $y \in \{0, 1\}$, $y \sim \text{Bernoulli}(p^*)$. We denote optimal action $t := k^*_\ell \circ p^*$ as post-processing function.

If we know p^*, it is easy for us take take the optimal action.

For $y \in \{0, 1\},\ y \sim \text{Bernoulli}(p^*).$ We denote optimal action $t := k_\ell^* \circ p^*$ as post-processing function.

This paper connects multigroup fairness with the notion of a weak agnostic learner, to formulate (L, C)-omnipredictors.

Omnipredictors

If we know p^*, it is easy for us take take the optimal action.

For $y \in \{0, 1\}$, $y \sim \text{Bernoulli}(p^*)$. We denote optimal action $t := k^*_\ell \circ p^*$ as post-processing function.

This paper connects multigroup fairness with the notion of a weak agnostic learner, to formulate (L, C)-omnipredictors.

Intuitively: the idea is to *extract the predictive power* of the data.

Omnipredictors

If we know p^*, it is easy for us take take the optimal action.

For $y \in \{0, 1\},\ y \sim \text{Bernoulli}(p^*)$. We denote optimal action $t := k^*_\ell \circ p^*$ as post-processing function.

This paper connects multigroup fairness with the notion of a weak agnostic learner, to formulate (L, C)-omnipredictors.

Intuitively: the idea is to *extract the predictive power* of the data.

Let L_{cvx} be a set of Lipschitz, convex, bounded losses.

Omnipredictors

If we know p^*, it is easy for us take take the optimal action.

For $y \in \{0, 1\}$, $y \sim \text{Bernoulli}(p^*)$. We denote optimal action $t := k_\ell^* \circ p^*$ as post-processing function.

This paper connects multigroup fairness with the notion of a weak agnostic learner, to formulate (L, C)-omnipredictors.

Intuitively: the idea is to *extract the predictive power* of the data.

Let L_{cvx} be a set of Lipschitz, convex, bounded losses. If $\tilde{p}$ is C-multicalibrated with some error α, it is an (L_{cvx}, C, α)-omnipredictor.

Omnipredictors

If we know p^*, it is easy for us take take the optimal action.

For $y \in \{0, 1\}$, $y \sim \text{Bernoulli}(p^*)$. We denote optimal action $t := k^*_\ell \circ p^*$ as post-processing function.

This paper connects multigroup fairness with the notion of a weak agnostic learner, to formulate (L, C)-omnipredictors.

Intuitively: the idea is to *extract the predictive power* of the data.

Let L_{cvx} be a set of Lipschitz, convex, bounded losses. If $\tilde{p}$ is C-multicalibrated with some error α, it is an (L_{cvx}, C, α)-omnipredictor.

Multicalibration implies omniprediction for all convex loss functions.

Proof of Optimality

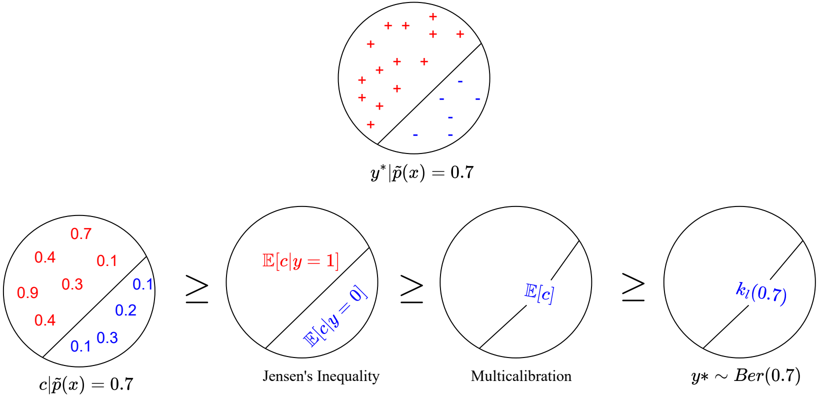
As proof for optimality, we evaluate binary classification. Assumptions:

1. p^* is boolean.
2. Perfect Mutlicalibration:
   $\mathbb{E}[c(x)(y^* - v)|\tilde{p}(x) = v] = 0, \forall v \in [0, 1], c \in C$

Proof of Optimality

As proof for optimality, we evaluate binary classification. Assumptions:

1. p^* is boolean.
2. Perfect Mutlicalibration:
   $\mathbb{E}[c(x)(y^* - v)|\tilde{p}(x) = v] = 0, \forall v \in [0,1], c \in C$



Training Agnostic Learners

We can train an agnostic learner using the following setup:

$$\mathcal{H} : \mathcal{X} \to \{0, 1\} \tag{8}$$

$$\mathcal{D} : \mathcal{X} \times \{0, 1\} \tag{9}$$

$$\ell(h \in \mathcal{H}) = \Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y] = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell_1(y, h(x))] \tag{10}$$

$$OPT(\mathcal{H}) = \min_{h\in\mathcal{H}} \ell(h) \tag{11}$$

Training Agnostic Learners

We can train an agnostic learner using the following setup:

$$\mathcal{H} : \mathcal{X} \to \{0, 1\} \tag{8}$$
$$\mathcal{D} : \mathcal{X} \times \{0, 1\} \tag{9}$$
$$\ell(h \in \mathcal{H}) = \Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y] = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell_1(y, h(x))] \tag{10}$$
$$OPT(\mathcal{H}) = \min_{h\in\mathcal{H}} \ell(h) \tag{11}$$

An agnostic learner for $\mathcal{H}$ produces f s.t. $\ell(f) \leq \mathrm{OPT}(\mathcal{H}) + \epsilon$.

Training Agnostic Learners

We can train an agnostic learner using the following setup:

$$\mathcal{H} : \mathcal{X} \to \{0, 1\} \tag{8}$$

$$\mathcal{D} : \mathcal{X} \times \{0, 1\} \tag{9}$$

$$\ell(h \in \mathcal{H}) = \Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y] = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell_1(y, h(x))] \tag{10}$$

$$OPT(\mathcal{H}) = \min_{h\in\mathcal{H}} \ell(h) \tag{11}$$

An agnostic learner for $\mathcal{H}$ produces f s.t. $\ell(f) \leq \text{OPT}(\mathcal{H}) + \epsilon$.

We (ϵ, W)-approximate $\mathcal{H}$ by $\mathcal{C}$ if $\forall h \in \mathcal{H}, \epsilon > 0$, $g_w \in \text{Lin}_{\mathcal{C}}(W)$:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[|g_w(x) - h(x)|] \leq \epsilon \tag{12}$$

If partition $\mathcal{S}$ is $\frac{\epsilon}{2W}$-approximately multicalibrated for $\mathcal{C}, \mathcal{D}$, $\ell(h^{\mathcal{S}}_{\ell_1}) \leq \text{OPT}(\mathcal{H}) + \epsilon$ and can identify multicalibrated partitions.

Thank you!

Have an awesome rest of your day!

Slides:

https://cs.purdue.edu/homes/jsetpal/slides/omnipredictors.pdf