Extrapolating prediction error for ‘extreme’ multi-class classification

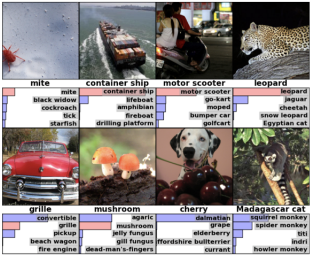
Charles Zheng

Stanford University

February 20, 2017

(Joint work with Rakesh Achanta and Yuval Benjamini.)

Multi-class classification



from Krizhevsky et al. 2012

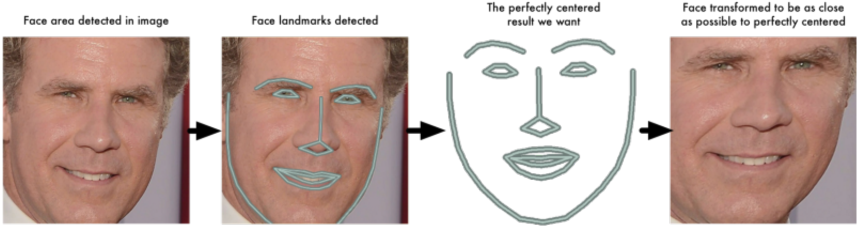
- MNIST digit recognition: 10 categories
- Human motion database: 51 categories
- ImageNet: 22,000 categories
- Wikipedia: 325,000 categories

Facial recognition

- Used to tag images in software, security

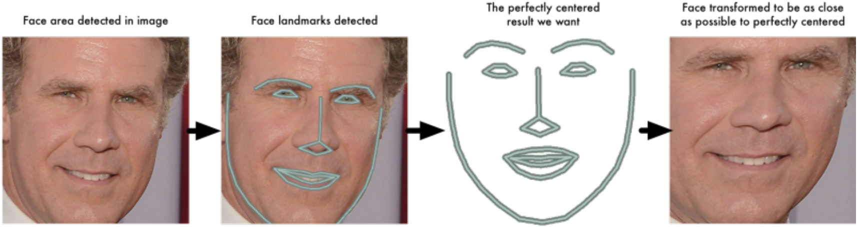
Facial recognition

- Used to tag images in software, security
- Preprocessing

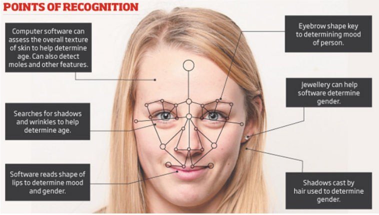


Facial recognition

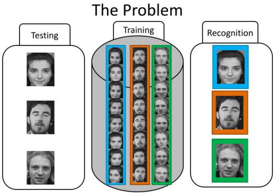
- Used to tag images in software, security
- Preprocessing



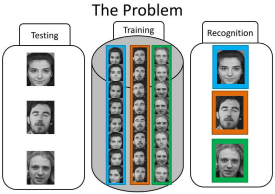
- Feature extraction

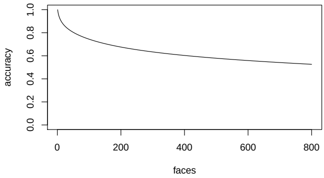


Accuracy vs. number of classes

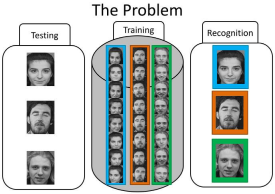


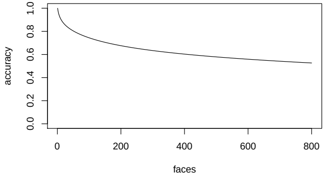
Accuracy vs. number of classes





Accuracy vs. number of classes

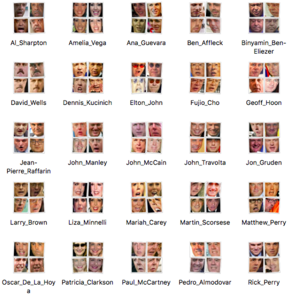




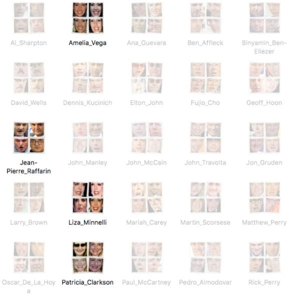
How does the accuracy scale with the number of classes (faces)?

Setup

1. Population of categories $\pi(y)$

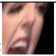
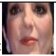


2. Subsample k labels, $y_1, \ldots, y_k$



Setup

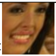
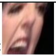
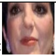
3. Collect training and test data $x_j^{(i)}$ (faces) for labels (people) $\{y_1, \ldots, y_k\}$.

Label	Training			Test
y_1=Amelia	$x_1^{(1)} =$ 	$x_1^{(2)} =$ 	$x_1^{(3)} =$ 	$x_1^* =$ 
y_2=Jean-Pierre	$x_2^{(1)} =$ 	$x_2^{(2)} =$ 	$x_2^{(3)} =$ 	$x_2^* =$ 
y_3=Liza	$x_3^{(1)} =$ 	$x_3^{(2)} =$ 	$x_3^{(3)} =$ 	$x_3^* =$ 
y_4=Patricia	$x_4^{(1)} =$ 	$x_4^{(2)} =$ 	$x_4^{(3)} =$ 	$x_4^* =$ 

4. Train a classifier and compute test error.

Setup

3. Collect training and test data $x_j^{(i)}$ (faces) for labels (people) $\{y_1, \ldots, y_k\}$.

Label	Training			Test
y_1=Amelia	$x_1^{(1)} =$ 	$x_1^{(2)} =$ 	$x_1^{(3)} =$ 	$x_1^* =$ 
y_2=Jean-Pierre	$x_2^{(1)} =$ 	$x_2^{(2)} =$ 	$x_2^{(3)} =$ 	$x_2^* =$ 
y_3=Liza	$x_3^{(1)} =$ 	$x_3^{(2)} =$ 	$x_3^{(3)} =$ 	$x_3^* =$ 
y_4=Patricia	$x_4^{(1)} =$ 	$x_4^{(2)} =$ 	$x_4^{(3)} =$ 	$x_4^* =$ 

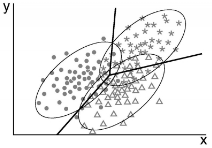
4. Train a classifier and compute test error.

Can we analyze how error depends on k?

Key assumption: marginal classifier

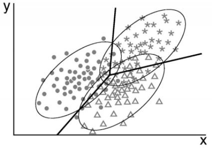
- The classifier is *marginal* if it learns a model *independently* for each class.

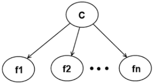
Key assumption: marginal classifier

- The classifier is *marginal* if it learns a model *independently* for each class.
- Examples: LDA/QDA 

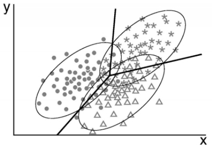
,

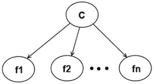
Key assumption: marginal classifier

- The classifier is *marginal* if it learns a model *independently* for each class.
- Examples: LDA/QDA 

, naïve Bayes 

Key assumption: marginal classifier

- The classifier is *marginal* if it learns a model *independently* for each class.
- Examples: LDA/QDA 

, naïve Bayes 

- Non-marginal classifiers: Multinomial logistic, multilayer neural networks, k-nearest neighbors

Definitions

$\hat{F}_{y^{(i)}}$ is the empirical distribution obtained from the training data for label $y^{(i)}$.

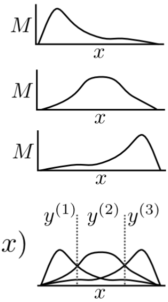
Classification Rule

$$M_{y^{(1)}}(x) = \mathcal{M}(\hat{F}_{y^{(1)}})(x)$$

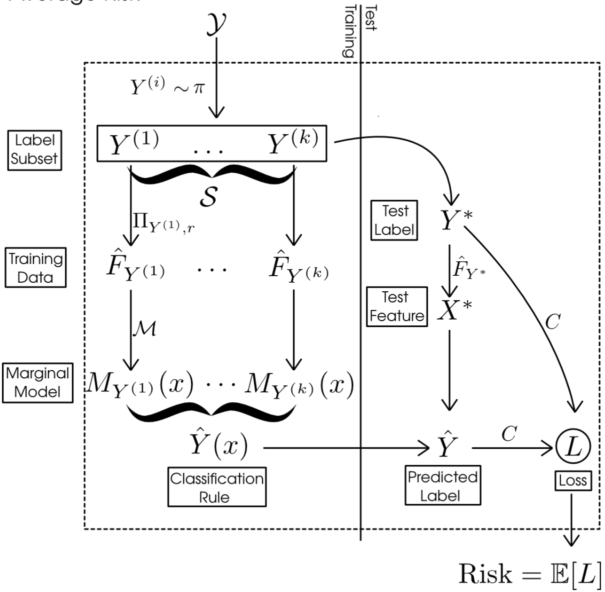
$$M_{y^{(2)}}(x) = \mathcal{M}(\hat{F}_{y^{(2)}})(x)$$

$$M_{y^{(3)}}(x) = \mathcal{M}(\hat{F}_{y^{(3)}})(x)$$

$$\hat{Y}(x) = \text{argmax}_{y \in \mathcal{S}} M_y(x)$$



Average Risk



Theoretical Result

Theorem. **(Z.**, Achanta, Benjamini.) Suppose π, $\{F_y\}_{y\in\mathcal{Y}}$ and marginal classifier $\mathcal{F}$ satisfy *(some regularity condition)*. Then, there exists some function $\bar{D}(u)$ on $[0,1] \to [0,1]$ such that the k-class average risk is given by

$$\text{AvRisk}_k = (k-1)\int \bar{D}(u)u^{k-2}du.$$

Theoretical Result

Theorem. (Z., Achanta, Benjamini.) Suppose π, $\{F_y\}_{y\in\mathcal{Y}}$ and marginal classifier $\mathcal{F}$ satisfy *(some regularity condition)*. Then, there exists some function $\bar{D}(u)$ on $[0,1] \to [0,1]$ such that the k-class average risk is given by

$$\text{AvRisk}_k = (k-1)\int \bar{D}(u)u^{k-2}du.$$

What is this $\bar{D}(u)$ function? We will explain in the following toy example...

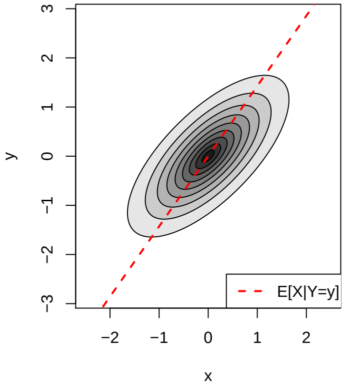
Toy example

$Y_1, \ldots, Y_k \overset{iid}{\sim} N(0, 1)$;

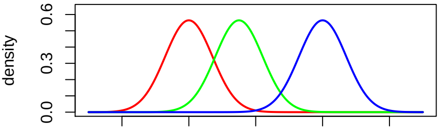
Toy example

$Y_1, \ldots, Y_k \overset{iid}{\sim} N(0,1)$;

$X|Y \sim N(\rho Y, 1-\rho^2)$ i.e. $(Y,X) \sim N(0, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix})$.

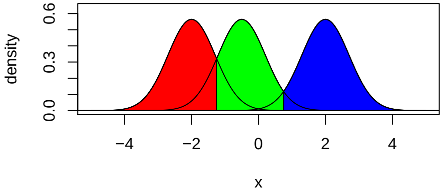


Toy example



- Suppose $k = 3$, and we draw Y_1, Y_2, Y_3.
- The *Bayes rule* is the optimal classifier and depends on knowing the true densities:
$$\hat{y}(x) = \text{argmax}_{y_i} p(x|y_i)$$
- The *Bayes Risk*, which is the misclassification rate of the optimal classifier.

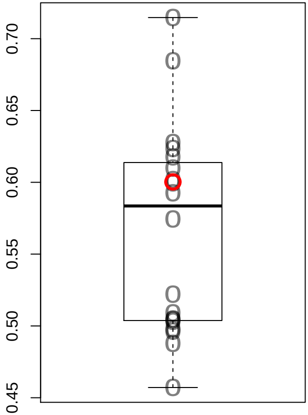
Toy example

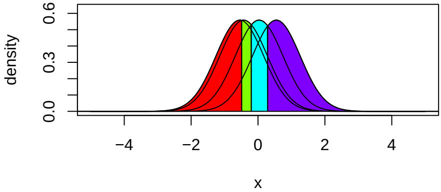


- The *Bayes Risk* is the expected test error of the Bayes rule,

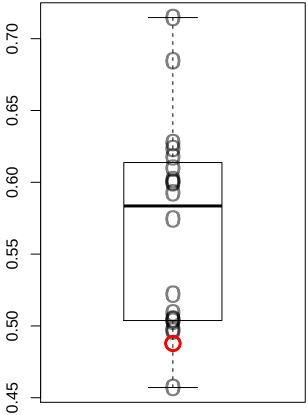
$$\frac{1}{k}\sum_{i=1}^{k}\Pr[\hat{y}(x) \neq Y | Y = y_i]$$

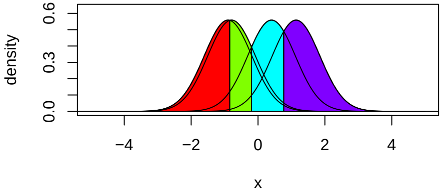
Toy example



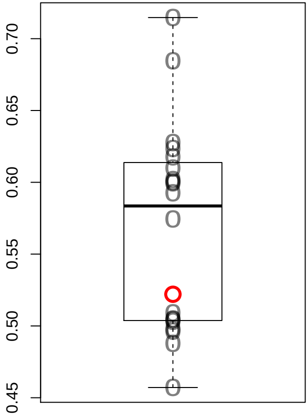


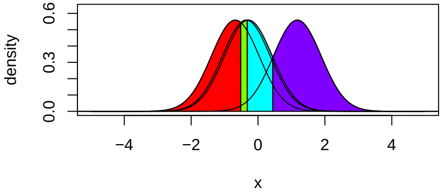
Toy example



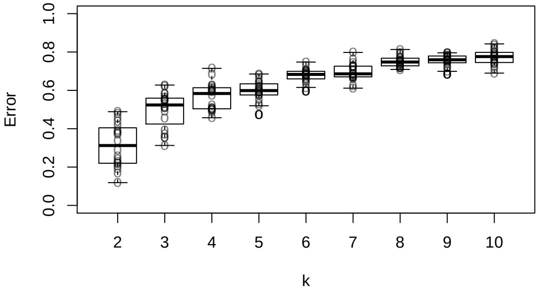


Toy example

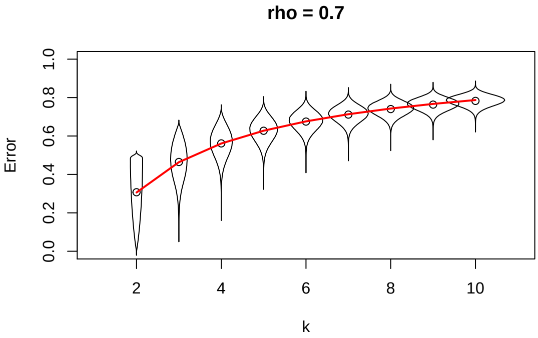




Toy example



Toy example

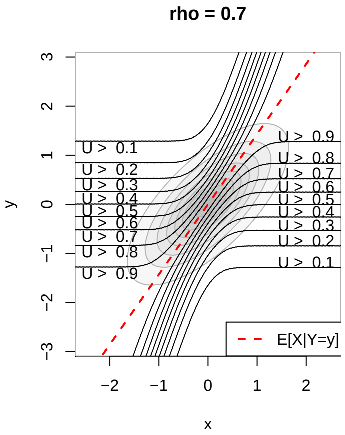


Defining the U-function

Define $U_x(y)$ as follows:

- Suppose we have test instance (face) x whose true label (person) is y.
- Let Y' be a random *incorrect* label (person).
- Use the classifier to guess whether x belongs to y or Y'.
- Define $U_x(y)$ as the probabilility of success (randomizing over training data).

Toy example



$$U_y(x) = \Pr[d(x, \rho Y') > d(x, \rho y)], \text{ for } Y' \sim N(0, 1).$$

Defining $\bar{D}(u)$

- Define random variable as $U_Y(X)$ for (Y, X) drawn from the joint distribution.

Defining $\bar{D}(u)$

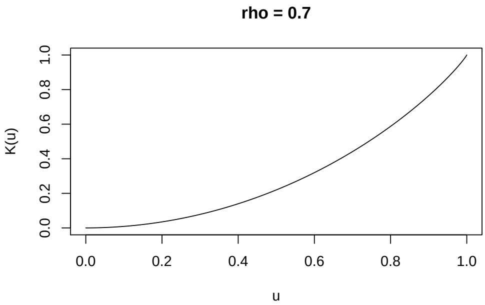
- Define random variable as $U_Y(X)$ for (Y, X) drawn from the joint distribution.
- $\bar{D}(u)$ is the cumulative distribution function of U,

$$\bar{D}(u) = \Pr[U_Y(X) \leq u].$$

Defining $\bar{D}(u)$

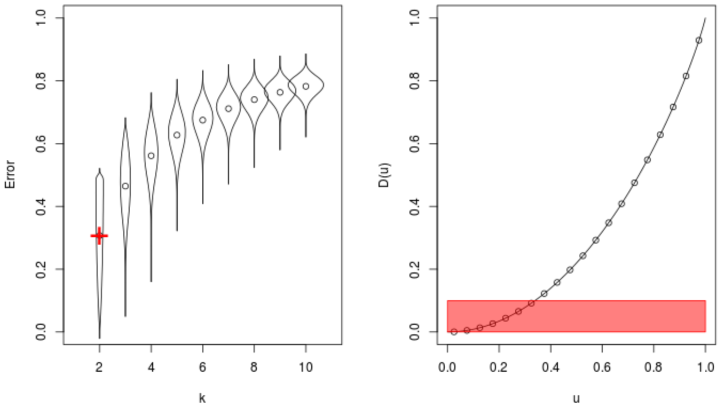
- Define random variable as $U_Y(X)$ for (Y, X) drawn from the joint distribution.
- $\bar{D}(u)$ is the cumulative distribution function of U,

$$\bar{D}(u) = \Pr[U_Y(X) \leq u].$$



Computing average risk

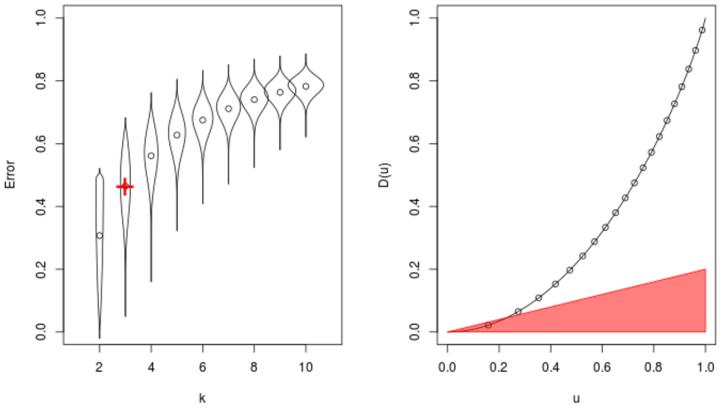
$$\text{AvRisk}_k = (k-1) \int \bar{D}(u) u^{k-2} du.$$



$(k = 2)$

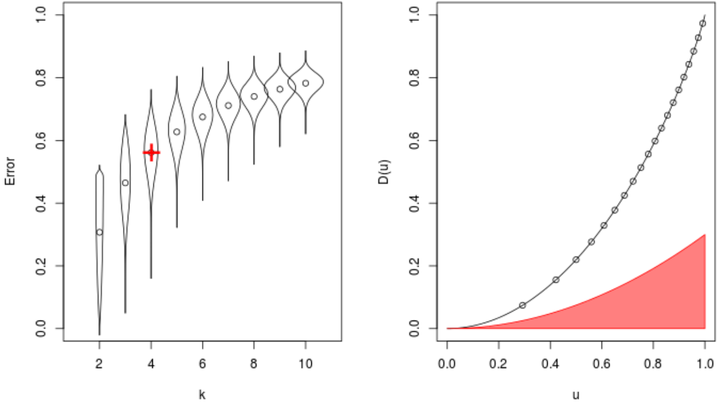
Computing average risk

$$\text{AvRisk}_k = (k-1)\int \bar{D}(u)u^{k-2}du.$$



$(k = 3)$

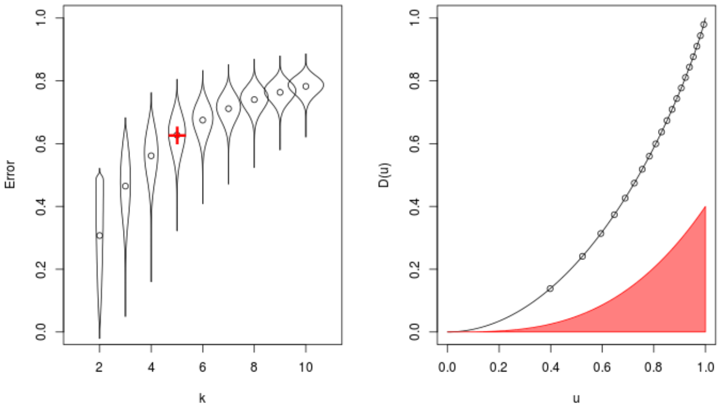
$$\text{AvRisk}_k = (k-1) \int \bar{D}(u) u^{k-2} du.$$



$(k = 4)$

Computing average risk

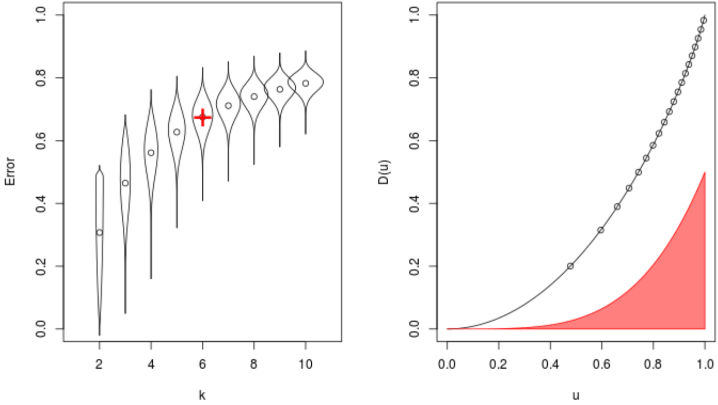
$$\text{AvRisk}_k = (k-1)\int \bar{D}(u)u^{k-2}du.$$



$(k = 5)$

Computing average risk

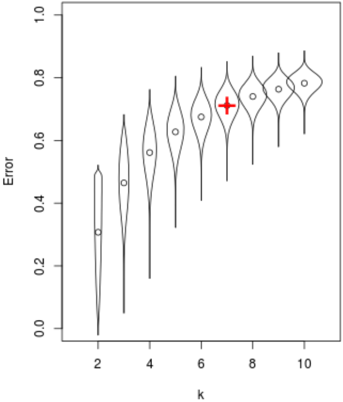
$$\text{AvRisk}_k = (k-1)\int \bar{D}(u)u^{k-2}du.$$

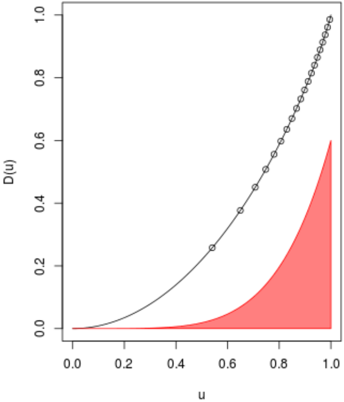


$(k = 6)$

Computing average risk

$$\text{AvRisk}_k = (k-1)\int \bar{D}(u)u^{k-2}du.$$

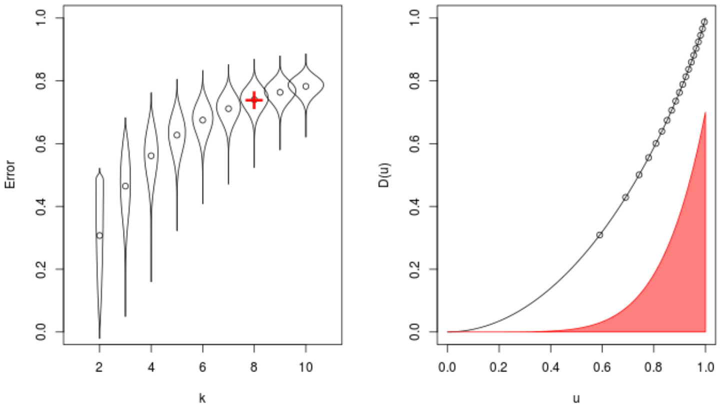




$(k = 7)$

Computing average risk

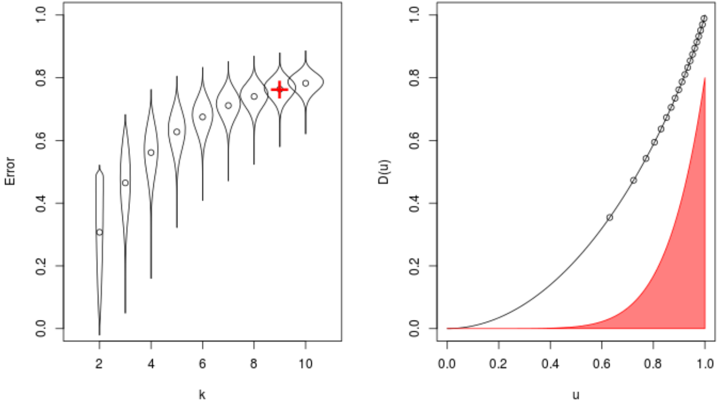
$$\text{AvRisk}_k = (k-1)\int \bar{D}(u)u^{k-2}du.$$



$(k = 8)$

Computing average risk

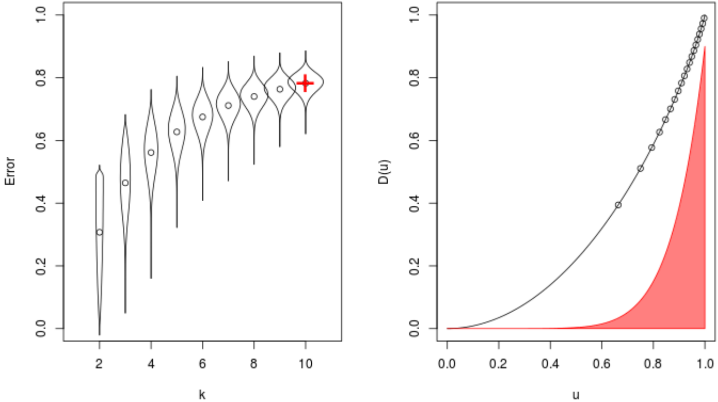
$$\text{AvRisk}_k = (k-1)\int \bar{D}(u)u^{k-2}du.$$



$(k = 9)$

Computing average risk

$$\text{AvRisk}_k = (k-1)\int \bar{D}(u)u^{k-2}du.$$

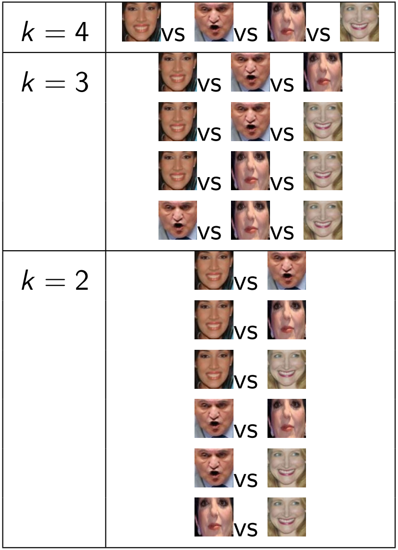


$(k = 10)$

Implication: estimate $\bar{D}(u)$ to predict risk

- Theoretical result links k-class average risk to $\bar{D}(u)$ function
- In real data, we do not know $\bar{D}(u)$ since it depends on the unknown joint distribution
- However, given a model, we can estimate $\bar{D}(u)$

Subsampled risk estimates



- Suppose we have data for k classes (subsampled from π)
- The test error TestErr_k is an unbiased estimate for AvRisk_k
- For any $\ell < k$, we can estimate AvRisk_ℓ by *subsampling* the k classes, and taking the average test risk, AvTestErr_ℓ.

Parametric modelling approach

Assume that for set of basis functions $h_1, \ldots, h_\ell$, we have

$$\bar{D}(u) = \sum_{\ell=1}^{m} \beta_\ell h_\ell(u).$$

Then

$$\text{AvRisk}_k = \sum_{\ell=1}^{m} \beta_\ell H_{\ell,k} = \beta^T \vec{H}_k$$

where

$$H_{\ell,k} = (k-2) \int_0^1 h_\ell(u) u^{k-2} du.$$

and $\vec{H}_k = (H_{1,k}, \ldots, H_{\ell,k})$

Prediction extrapolation as regression

1. Choose basis $h_1, \ldots, h_\ell$

$$\bar{D}(u) = \sum_{\ell=1}^{m} \beta_\ell h_\ell(u).$$

Prediction extrapolation as regression

1. Choose basis $h_1, \ldots, h_\ell$
$$\bar{D}(u) = \sum_{\ell=1}^{m} \beta_\ell h_\ell(u).$$
2. Obtain subsampled test errors $\text{AvTestErr}_2, \ldots, \text{AvTestErr}_k$.

Prediction extrapolation as regression

1. Choose basis $h_1, \ldots, h_\ell$

$$\bar{D}(u) = \sum_{\ell=1}^{m} \beta_\ell h_\ell(u).$$

2. Obtain subsampled test errors $\text{AvTestErr}_2, \ldots, \text{AvTestErr}_k$.
3. Fit regression model

$$\hat{\beta} = \text{argmin}_\beta \sum_{i=2}^{k} \left(\text{AvTestErr}_i - \vec{H}_i^T \beta_\ell\right)^2$$

Prediction extrapolation as regression

1. Choose basis $h_1, \ldots, h_\ell$

$$\bar{D}(u) = \sum_{\ell=1}^{m} \beta_\ell h_\ell(u).$$

2. Obtain subsampled test errors $\text{AvTestErr}_2, \ldots, \text{AvTestErr}_k$.
3. Fit regression model

$$\hat{\beta} = \text{argmin}_\beta \sum_{i=2}^{k} \left(\text{AvTestErr}_i - \vec{H}_i^T \beta_\ell \right)^2$$

4. For $K > k$, predict AvRisk_K as

$$\widehat{\text{AvRisk}}_K = \vec{H}_K^T \beta_\ell.$$

Examples of basis functions

- Polynomials, $1, x, x^2, \ldots$
- Cubic splines
- Linear splines, $[x - t_\ell]_+$

Optional constraint: assume β is *non-negative*. In the case of linear splines, this results in a *convex* fit.