FA17 10-701 Homework 5 Recitation 1

Easwaran Ramamurthy Guoquan (GQ) Zhao Logan Brooks

Note

Remember that there is no problem set covering some of the lecture material; you may need to study these topics more.

ICA: why whiten?

ICA is simpler for centered, white $\mathbf{x}^*$'s:

- $\frac{1}{n}\sum_i \mathbf{x}_i^* = \mathbf{0}_N$
- $\frac{1}{n}\mathbf{X}^*(\mathbf{X}^*)^T = \mathbf{I}_{N\times N}$ (dimensions of $\mathbf{X}$ are flipped from what we are used to)

We want centered, white $\mathbf{y}^*$'s: only orthogonal $\mathbf{W}^*$ always work

- Can't tell exact scale and ordering $\rightsquigarrow$ considering only rotation matrices $\mathbf{W}^*$ is just as good

We get simplifications in kurtosis calculations, too.
Transformation: we found $\mathbf{Q}$ to get $\mathbf{X}^* = \mathbf{QX} = \mathbf{QAS} = \mathbf{A}^*\mathbf{S}$

- Want $\mathbf{W}$ to act like $\mathbf{A}^{-1}$; $\mathbf{A}^* = \mathbf{QA}$ so choose $\mathbf{W} = \mathbf{W}^*\mathbf{Q}^{-1} = \mathbf{W}^*\mathbf{UD}^{1/2}$
- Choose $\mathbf{Y} = \mathbf{Y}^*$
- We considered enough $\mathbf{W}$'s; considering all orthogonal $\mathbf{W}^*$'s would consider all working $\mathbf{W}$'s

ICA: different measures of non-normality

(From the reading material.)
Absolute value of kurtosis, $\left|\mathbb{E}[y^4] - 3(\mathbb{E}[y^2])^2\right|$:

- Maximized to choose the first $\mathbf{w}$
- Maximized subject to orthogonality constraints to choose later $\mathbf{w}$'s

ICA: different measures of non-normality

(From the reading material.)

Negentropy $H(\mathbf{y}_{\text{Gaussian}}) - H(\mathbf{y})$:

- $\mathbf{y}_{\text{Gaussian}}$: Gaussian RV with same mean, covariance as $\mathbf{y}$
- Maximized to choose $\mathbf{W}$
- Exact form: appealing theoretically, problematic computationally
- Approximations for a single y (single $\mathbf{w}$): of form $\sum_{i=1}^{p} k_i(\mathbb{E}[G_i(y)] - \mathbb{E}[G_i(\nu)])^2$
  - G_i's non-quadratic
  - y, ν: mean 0, variance 1
  - ν Gaussian
  - First expectation: actually the sample mean

ICA: different measures of non-normality

KL divergence of joint from product of marginals ("mutual information", at least for two y's): $\int p(y_1, \ldots, y_M) \frac{p(y_1,\ldots,y_M)}{p(y_1)\ldots,p(y_M)}$:

- Minimizing this is roughly equivalent or equivalent under some constraints to maximizing negentropy

Learning theory: review of notation

1st Slides	Reading	Meaning
f	g	Some model (1 input $\mapsto$ 1 prediction)
$L(x, y, f(x))$	$f(x, y)$	Loss of a model on 1 example (x, y)
$R_{L,P}(f)$, $R(f)$	$R(g)$, Pf	Risk (expected loss) of a model
$\hat{R}_n(f)$	$P_n f$	Empirical (training) risk of a model
f^*	g^*	Minimal-risk model
f_D	g_n	Model learned on n training points
$\vdots$	$\vdots$	$\vdots$

- Based on true distribution
- Based on training/empirical data (Random!)
- What are the following? Which are random?
  $\hat{R}_n(f_D)$, $R(f_D)$, $\hat{R}_n(f)$, $R(f)$
- What's the probability that we get a training set that makes our algorithm's fit model perform poorly (for some definition of poorly)?

Learning theory: review of notation

$$\left|R(f^*_{n,\mathcal{F}}) - R^*_{\mathcal{F}}\right|$$

- Meaning?
- Why is there an absolute value? Can we get rid of it?

$$\sup_{f\in\mathcal{F}}\left|\hat{R}_n(f) - R(f)\right|$$

- Meaning?
- Why is there an absolute value? Can we get rid of it?

Learning theory: review of notation

$\left| R(f^*_{n,\mathcal{F}}) - R^*_{\mathcal{F}} \right|$

- Meaning? Absolute difference in risk of fit and best model
- Why is there an absolute value? Can we get rid of it? Easier to apply common inequalities. Yes; $R(f^*_{n,\mathcal{F}}) \geq R^*_{\mathcal{F}}$.

$\sup_{f \in \mathcal{F}} \left| \hat{R}_n(f) - R(f) \right|$

- Meaning?
- Why is there an absolute value? Can we get rid of it?

Learning theory: review of notation

$\left| R(f^*_{n,\mathcal{F}}) - R^*_{\mathcal{F}} \right|$

- Meaning? Absolute difference in risk of fit and best model
- Why is there an absolute value? Can we get rid of it? Easier to apply common inequalities. Yes; $R(f^*_{n,\mathcal{F}}) \geq R^*_{\mathcal{F}}$.

$\sup_{f \in \mathcal{F}} \left| \hat{R}_n(f) - R(f) \right|$

- Meaning? Max absolute difference in true and empirical risk among all models
- Why is there an absolute value? Can we get rid of it? Easier/quicker to prove than two directions. No; either term can be larger.

Learning theory: VC dimension

- $S_{\mathcal{F}}(n)$: nth shatter coefficient; maximum number of "behaviors" we can obtain from f's in $\mathcal{F}$ on datasets of size n
  - "Behavior" of f: subset of x's selected by f
  - Number of behaviors: number of unique subsets (consider all possible f's in $\mathcal{F}$)
  - Maximum number of behaviors: take max over all possible datasets of size n
  - What's the lowest possible $S_{\mathcal{F}}(n)$ (as a function of n)? What's the highest possible $S_{\mathcal{F}}(n)$?
- VC dimension: maximum n such that f's in $\mathcal{F}$ display all possible behaviors (try to express this using $S_{\mathcal{F}}(n)$).
- True or false: "we should always favor a $\mathcal{F}$ with a higher VC dimension".

HW5 FAQ

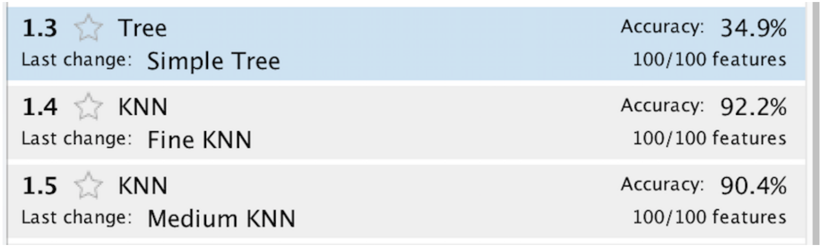
I can't read in the data.

- Look on the Piazza tool list or a search engine for a library that will help. For example, `pandas.read_table` seems to work a lot better than `numpy.loadtxt`.
- Be somewhat patient when loading the training covariates — this is around 4.3 GiB; hard drives will take a while to load this (check whether your disk is at full utilization)
- Consider saving the data in a format that is quicker or easier to load for your platform, for later use

HW5 FAQ

I divided the data set randomly, with 3/4 into training and 1/4 into validation. Almost every time I obtained a test accuracy of around 92%. Why?

- There are experimental biases in the given dataset and your classifier is almost guaranteed to be affected by these biases. You shouldn't ignore the experimental biases in the training data, and your classifier should learn the underlining pattern instead of the biases. In order to infer the true performance of your classifier, you need to create your own test sets NOT by randomly splitting the dataset.
- Your test data shouldn't contain the same accession ID as those in the training data.



1.3 ☆ Tree	Accuracy: 34.9%
Last change: Simple Tree	100/100 features
1.4 ☆ KNN	Accuracy: 92.2%
Last change: Fine KNN	100/100 features
1.5 ☆ KNN	Accuracy: 90.4%
Last change: Medium KNN	100/100 features

- Randomly splitting the data. (Using Matlab's built-in classifier).

HW5 FAQ

How to choose an appropriate library and algorithm?

- scikit-learn, Shogun, Matlab for PCA, SVM, ensembles, KNN, basic neural networks etc.
- Deep neural networks: Keras, Pytorch, TFLeran, Tensorflow
- If you use these classical classifiers (KNN, SVM, etc), it's very likely that your program eventually crash because of Out-Of-Memory error. **Reduce the dimension** before running these algorithm.
- PCA is a good way to reduce the dimension, but we expect more and well-justified and novel ideas will generally receive higher scores. (25pts for ideas)
- For Deep Neural Networks, for example, Keras, the input data is a numpy array. Use a data generator rather than load everything into memory. The data generator should **not** randomly split the data.