Design and analysis of machine learning experiments

Based on chapter 19 in Alpaydin ML

Najmeh Abiri

Department of Computer Science
IT University of Copenhagen

Table of contents

Introduction

Managing machine learning

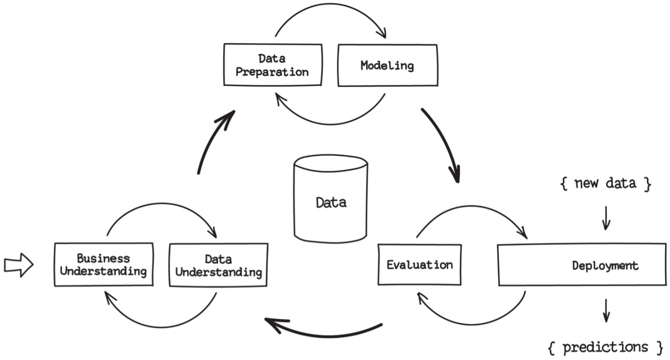
- How confident we are on the error of a model on a dataset?
- How can we compare several methods output on a dataset?
- **Memorizing vs learning**: training - validation - test

Randomness

- Training - validation - test: sampling from data .ipynb
- Model initialization: e.g., with different initial weights, gradients in MLP converge to different local minima
- **Solution: Generalization**
- Average over randomness: use the same algorithms and generate multiple learners, test learners on several validations $\longrightarrow$ distribution over errors (average and scale)

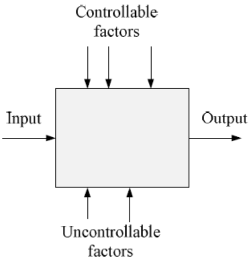
ML process

- ~~**Best algorithm**~~ : Learner is conditioned on dataset
- Training set: optimize parameters
- Validation set: optimize hyperparameters
- Test set: evaluation
- More on model parameter/hyperparameters



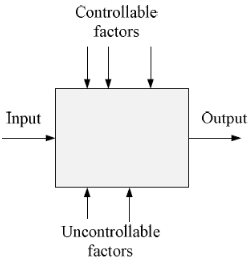
Pictures from towardsdatascience.com

Factors and responses



Factors : algorithms, training set, selected features, etc
Observe the change in response to extract information
Aim : identify important factors and optimize the response

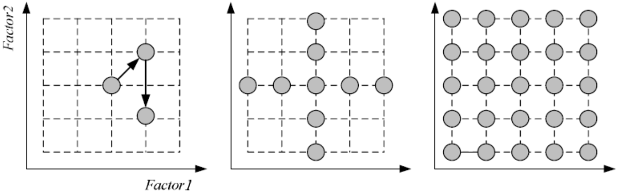
Factors and responses



- Best response based on output
- Source of the randomness: uncontrollable factors (noise in the data, randomly sampling training/validation/test sets and randomness in the optimization process) .ipynb
- Find the configuration of controllable factors that maximizes response and minimally affected by uncontrollable factors

Factor space

How to search the factor space?



(a) Best guess (b) One factor at a time (c) Factorial design

(a) No systematic search and criteria to stop
(b) Assumption: no correlation between factors (often not true)
(c) Grid search: checking all parameter combinations based on a given model - computationally expensive

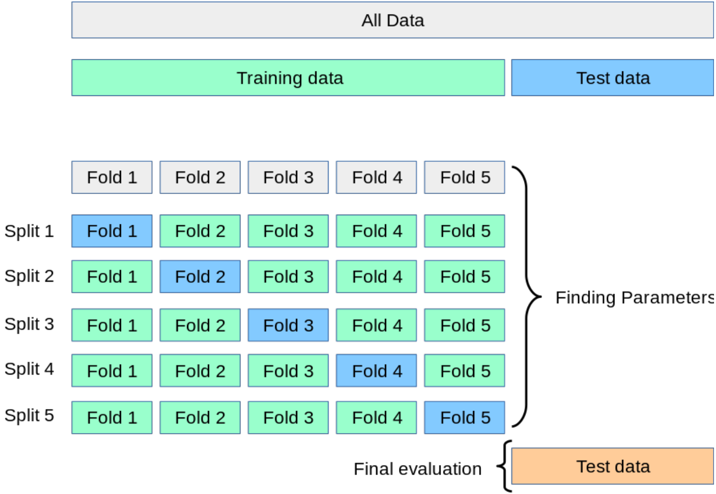
Use knowledge gathered from previous runs that shown a better response. Define a range for hyperparameters and generate random sets of their combinations for random search

Guidelines for ML experiments

- **Aim of the study:** ask correct question
- **Selection of the response variable** (MSE, BC)
- **Choice of factors and levels**
- **Choice of experimental design:** Grid search and random search - dividing data into training/testing : small data gives high variance in responses
- **Performing the experiment:** save intermediate results to be able to rerun partially - equal investigation on multiple ML methods
- **Statistical Analysis of the Data:** visual analysis
- **Conclusions and Recommendations:** start small - investigating results for improvement

Cross-Validation and resampling methods

k-fold Cross-Validation



Pictures from scikit-learn.org

5 times 2-fold CV (Dietterich, 1998)

For i in 5:

1. Shuffle X randomly
2. Divide X into $T_{i1} = X_1$ and $V_{i1} = X_2$
3. Replace partitions: $T_{i2} = X_2$ and $V_{i2} = X_1$

We have 10 different sets:

$$
\begin{aligned}
&\text{Set 1}: T_{11} = X_1 \quad V_{11} = X_2 \\
&\text{Set 2}: T_{12} = X_2 \quad V_{12} = X_1 \\
&\text{Set 3}: T_{21} = X_1' \quad V_{21} = X_2' \\
&\vdots \\
&\text{Set 9}: T_{51} = \hat{X}_1 \quad V_{51} = \hat{X}_2 \\
&\text{Set 10}: T_{52} = \hat{X}_2 \quad V_{52} = \hat{X}_1
\end{aligned}
$$

- With more than 5 iterations: sets share many instances and overlap so much that validation error become too dependent and do not add new information

Bootstrapping

Sampling from a data with replacement. Best way to do resampling for very **small** satasets.

Data = $[1, 2, 3, 4, 5]$
3 samples with size 4 with replacement:

- $s_1 = [1, 2, 3, 3]$
- $s_2 = [5, 1, 5, 3]$
- $s_2 = [3, 4, 3, 5]$

The best way to use bootstrapping is to repeat it several times to get a distribution of the responses.

Performance measurments

Binary classification

		Predicted		
		Positive	Negative	Total
True	Positive	TP (# of TPs)	FN (# of FNs)	p
	Negative	FP (# of FPs)	TN (# of TNs)	n

Confusion matrix for binary classification.

- **Error rate** $= \frac{FP+FN}{p+n}$
- **Accuracy** $= \frac{TP+TN}{p+n}$
- **Sensitivity (recall)** $= \frac{TP}{p}$
- **Specificity** $= \frac{TN}{n}$
- **Precision (Positive predictive value)** $= \frac{Tp}{TP+FP}$
- **False positive rate** $= \frac{FP}{n}$

Example

MNIST digits: one or seven

- Fit a logistic regression to two classes of MNIST digits: one and seven
- After training we evaluate the model with 10 samples of test data.

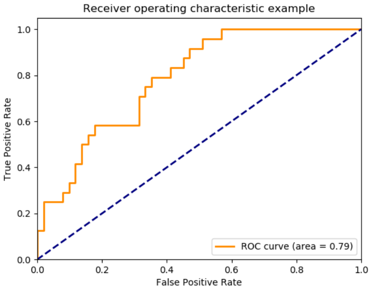
		Predicted		
		one	seven	Total
True	one	4	1	5
	seven	1	4	5

Confusion matrix for binary classification with threshold $= 0.5$

ROC and AUC

- How about different threshold?
- We can calculate sensitivity and specificity for any threshold $\in [0, 1]$.

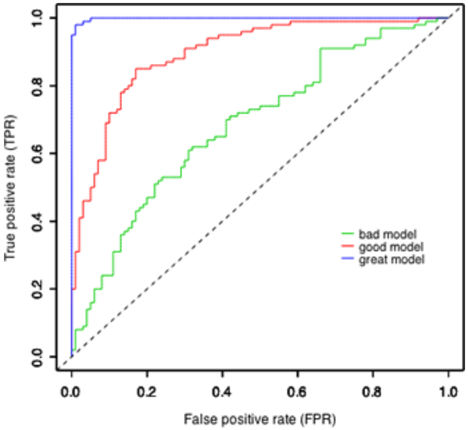
With different classification threshold, instead of using several confusion matrices, we can use ROC (Receiver Operator Characteristic) graphs and AUC (the area under the curve) that show the results in a single easy to interpret graph.



- Diagonal blue line : TP-rate = FP-rate
- X-axis = 1- specificity
- Y-axis : sensitivity
- Compare multiple models with their AUC

Pictures from scikit-learn.org

Model comparison



Pictures from medium.com/usf-msds/