3.36pt

Classification

Stefano Gualandi
Università di Pavia, Dipartimento di Matematica

email:	`stefano.gualandi@unipv.it`
twitter:	`@famo2spaghi`
blog:	`http://stegua.github.com`
web:	`http://matematica.unipv.it/gualandi/opt4ml`

Supervised Learning

Definition 1 (Supervised Learning)

Supervised Learning is the task of learning (inferring) a function f that maps input vectors to their corresponding target vectors, by using a dataset containing a given set of pairs of $(input, output)$ samples. Examples:

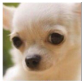
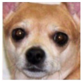
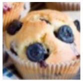
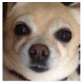
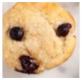
- Regression: the output vectors take one or more continuous values.
- Classification: the output vectors take one value of a finite number of discrete categories. Special case: binary classification.

Supervised Learning: Classification

Let us take $Y = \{-1, 1\}$, with $y = 1$ means **YES**, and $y = -1$ means **NO**.

Supervised Learning: Classification

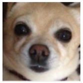
Let us take $Y = \{-1, 1\}$, with $y = 1$ means **YES**, and $y = -1$ means **NO**.

$\boldsymbol{x}_1 =$  $\boldsymbol{x}_2 =$  $\boldsymbol{x}_3 =$  $\boldsymbol{x}_4 =$  $\boldsymbol{x}_5 =$ 

$y_1 = 1$ $y_2 = 1$ $y_3 = -1$ $y_4 = 1$ $y_5 = -1$

Supervised Learning: Classification

Let us take $Y = \{-1, 1\}$, with $y = 1$ means **YES**, and $y = -1$ means **NO**.

$\boldsymbol{x}_1 =$ $\boldsymbol{x}_2 =$ $\boldsymbol{x}_3 =$ $\boldsymbol{x}_4 =$  $\boldsymbol{x}_5 =$

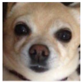
$y_1 = 1$ $y_2 = 1$ $y_3 = -1$ $y_4 = 1$ $y_5 = -1$

1. Use the TRAINING SET $\Rightarrow$ run learning algorithm $\Rightarrow$ get

$$f : X \to Y \qquad \text{where } f := f_m \circ f_{m-1} \circ \ldots f_1$$

Supervised Learning: Classification

Let us take $Y = \{-1, 1\}$, with $y = 1$ means **YES**, and $y = -1$ means **NO**.

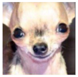
$\mathbf{x}_1 =$ $\mathbf{x}_2 =$ $\mathbf{x}_3 =$ $\mathbf{x}_4 =$  $\mathbf{x}_5 =$

$y_1 = 1$ $y_2 = 1$ $y_3 = -1$ $y_4 = 1$ $y_5 = -1$

1. Use the Training Set $\Rightarrow$ run learning algorithm $\Rightarrow$ get

$$f : X \to Y \qquad \text{where } f := f_m \circ f_{m-1} \circ \dots f_1$$

2. Use a Test Set to validate f:

$f\Big(\mathbf{x}_6 =$ $\Big) = -1$ $f\Big(\mathbf{x}_7 =$  $\Big) = 1$

Classification models

A **classification model**, or **classifier**, is used to label an example as belonging to one of a finite set of categories, called **classes**.

Classification models

A **classification model**, or **classifier**, is used to label an example as belonging to one of a finite set of categories, called **classes**.

One-class Learning: the training set contains examples drawn from only one class. Example: *anomaly detection*.

Classification models

A **classification model**, or **classifier**, is used to label an example as belonging to one of a finite set of categories, called **classes**.

One-class Learning: the training set contains examples drawn from only one class. Example: *anomaly detection*.

Two-class Learning (Binary Classification): the training set contains examples drawn from exactly two classes (positive and negative), and the objective is to find the boundary that separates the two classes.

Classification models

A **classification model**, or **classifier**, is used to label an example as belonging to one of a finite set of categories, called **classes**.

One-class Learning: the training set contains examples drawn from only one class. Example: *anomaly detection*.

Two-class Learning (Binary Classification): the training set contains examples drawn from exactly two classes (positive and negative), and the objective is to find the boundary that separates the two classes.

Multi-class Learning: involves finding the boundaries that separates more than two classes from each other.

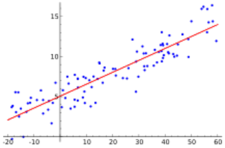
Classification (basic) Methods

1. *k*-**Nearest Neighbours**: New examples are assigned a class based on how similar, using a **METRIC (!)**, they are to examples in the training data. The *learned* model is the training examples itself.

Classification (basic) Methods

1. *k*-**Nearest Neighbours**: New examples are assigned a class based on how similar, using a **METRIC (!)**, they are to examples in the training data. The *learned* model is the training examples itself.

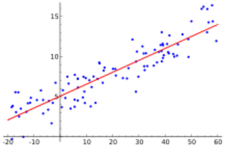
2. **Linear Regression**: We can train a linear regression model for each class, and given a new example, assign it to the linear relationship that better predict the example.

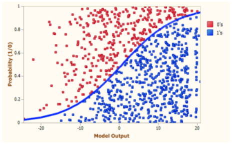


Classification (basic) Methods

1. *k*-**Nearest Neighbours**: New examples are assigned a class based on how similar, using a **METRIC (!)**, they are to examples in the training data. The *learned* model is the training examples itself.

2. **Linear Regression**: We can train a linear regression model for each class, and given a new example, assign it to the linear relationship that better predict the example.





3. **Logistic Regression**: We can train a logistic regression model (see, Chap. 4.4 in Hastie et all.)

Evaluating Classifiers

Given a data set, we divide the data into:

- Training Set: it is used to learn a model, and contains usually 80% of the data.

Evaluating Classifiers

Given a data set, we divide the data into:

Training Set: it is used to learn a model, and contains usually 80% of the data.

Test Set: it is used to evaluate the learned model, and it contains usually 20% of the data.

Evaluating Classifiers

Given a data set, we divide the data into:

Training Set: it is used to learn a model, and contains usually 80% of the data.

Test Set: it is used to evaluate the learned model, and it contains usually 20% of the data.

Given a data set, to increase the performance, it is better to fit the model by iteratively changing the Training and Test set, and by recording the model that yields the BEST RESULTS (see *n*-fold cross validation).

Evaluating Classifiers

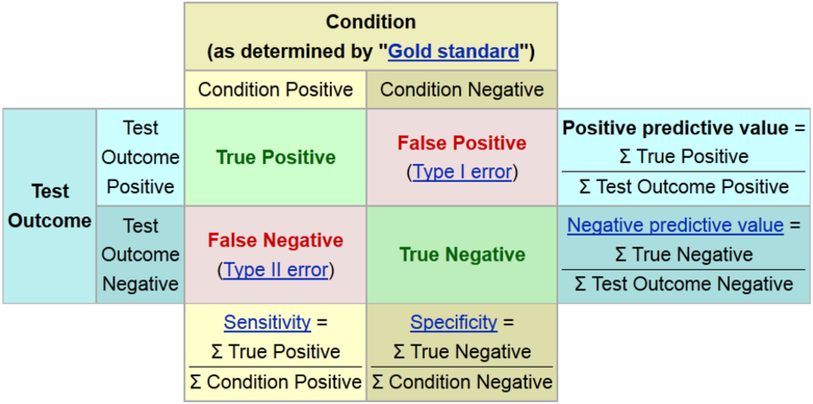
Given a data set, we divide the data into:

> Training Set: it is used to learn a model, and contains usually 80% of the data.
>
> Test Set: it is used to evaluate the learned model, and it contains usually 20% of the data.

Given a data set, to increase the performance, it is better to fit the model by iteratively changing the Training and Test set, and by recording the model that yields the BEST RESULTS (see *n*-fold cross validation).

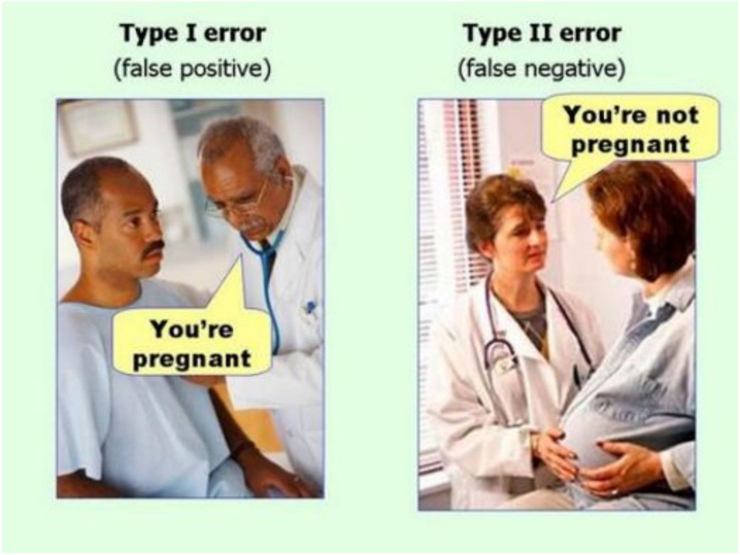
... but **BEST RESULTS** with respect to which METRIC?

Binary Classification: Confusion Matrix



		Condition (as determined by "Gold standard")		
		Condition Positive	Condition Negative	
Test Outcome	Test Outcome Positive	**True Positive**	**False Positive** (Type I error)	**Positive predictive value** = $\frac{\Sigma \text{ True Positive}}{\Sigma \text{ Test Outcome Positive}}$
	Test Outcome Negative	**False Negative** (Type II error)	**True Negative**	Negative predictive value = $\frac{\Sigma \text{ True Negative}}{\Sigma \text{ Test Outcome Negative}}$
		Sensitivity = $\frac{\Sigma \text{ True Positive}}{\Sigma \text{ Condition Positive}}$	Specificity = $\frac{\Sigma \text{ True Negative}}{\Sigma \text{ Condition Negative}}$	

Binary Classification: Confusion Matrix



Accuracy

$$\textsc{Accuracy} := \frac{\#\mathsf{TP} + \#\mathsf{TN}}{\#\mathsf{TP} + \#\mathsf{TN} + \#\mathsf{FP} + \#\mathsf{FN}}$$

Accuracy

$$\textsc{Accuracy} := \frac{\#\text{TP} + \#\text{TN}}{\#\text{TP} + \#\text{TN} + \#\text{FP} + \#\text{FN}}$$

TP:	True Positive
TN:	True Negative
FP:	False Positive
FN:	False Negative

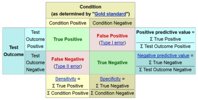
Accuracy

$$\textsc{Accuracy} := \frac{\#\mathsf{TP} + \#\mathsf{TN}}{\#\mathsf{TP} + \#\mathsf{TN} + \#\mathsf{FP} + \#\mathsf{FN}}$$

TP:	True Positive
TN:	True Negative
FP:	False Positive
FN:	False Negative

QUESTION: When Accuracy make sense for evaluating a binary classifier? Example?

Evaluating Classifiers with Imbalanced Classes



		Condition (as determined by "Gold standard")		
		Condition Positive	Condition Negative	
Test Outcome	Test Outcome Positive	True Positive	False Positive (Type I error)	**Positive predictive value** = Σ True Positive / Σ Test Outcome Positive
	Test Outcome Negative	False Negative (Type II error)	True Negative	Negative predictive value = Σ True Negative / Σ Test Outcome Negative
		Sensitivity = Σ True Positive / Σ Condition Positive	Specificity = Σ True Negative / Σ Condition Negative	

$$\textsc{Sensitivity (Recall)} := \frac{\#\mathsf{TP}}{\#\mathsf{TP} + \#\mathsf{FN}}$$

$$\textsc{Specificity (Precision)} := \frac{\#\mathsf{TN}}{\#\mathsf{TN} + \#\mathsf{FP}}$$

$$\textsc{Positive Predictive Value} := \frac{\#\mathsf{TP}}{\#\mathsf{TP} + \#\mathsf{FP}}$$

$$\textsc{Negative Predictive Value} := \frac{\#\mathsf{TN}}{\#\mathsf{TN} + \#\mathsf{FN}}$$

Evaluating a Classifiers: Loss Functions

1. **Mean Square Error/Quadratic Loss/L2 Loss**:

$$MSE := \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i; w))^2$$

Evaluating a Classifiers: Loss Functions

1. **Mean Square Error/Quadratic Loss/L2 Loss**:

$$MSE := \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i; w))^2$$

2. **Mean Absolute Error/L1 Loss**:

$$MAE := \frac{1}{n}\sum_{i=1}^{n}|y_i - f(x_i; w))|$$

Evaluating a Classifiers: Loss Functions

1. **Mean Square Error/Quadratic Loss/L2 Loss**:

$$MSE := \frac{1}{n}\sum_{i=1}^{n}(y_i - f(x_i; w))^2$$

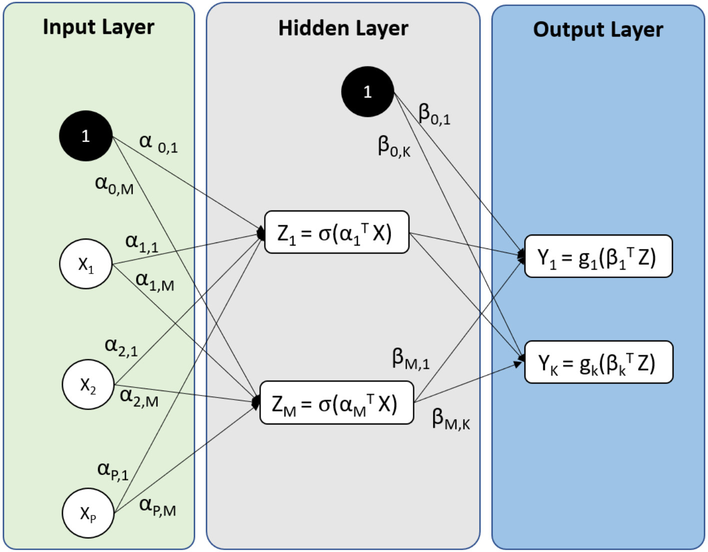
2. **Mean Absolute Error/L1 Loss**:

$$MAE := \frac{1}{n}\sum_{i=1}^{n}|y_i - f(x_i; w))\,|$$

3. **Cross Entropy Loss/Negative Log Likelihood**:

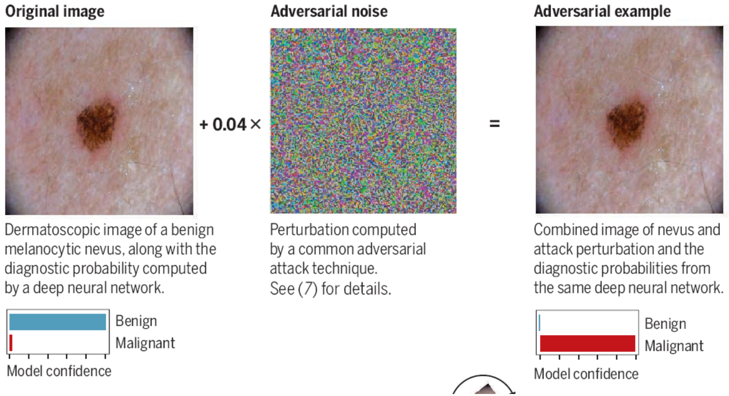
$$CrossEntropy := \sum_{i=1}^{n} -(y_i \log(f(x_i; w)) + (1 - y_i)\log(1 - f(x_i; w)))$$

Classification with Multilayer Neural Networks



$\alpha \in R^{P\times M}$ $\beta \in R^{M\times K}$

Generative Adversarial Attacks

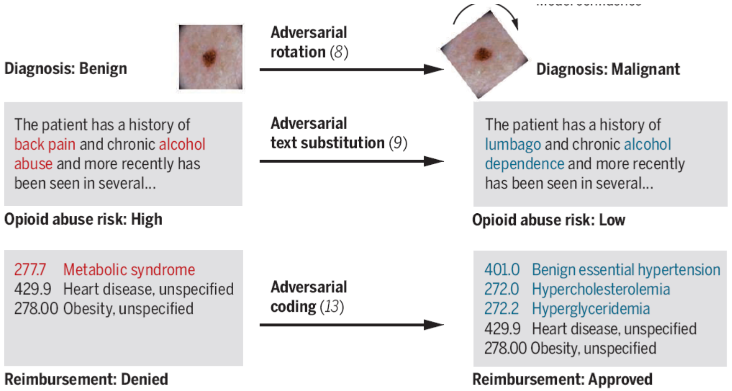


Dermatoscopic image of a benign melanocytic nevus, along with the diagnostic probability computed by a deep neural network.

Perturbation computed by a common adversarial attack technique. See (7) for details.

Combined image of nevus and attack perturbation and the diagnostic probabilities from the same deep neural network.

Generative Adversarial Attacks



Diagnosis: Benign

Adversarial rotation *(8)*

Diagnosis: Malignant

The patient has a history of back pain and chronic alcohol abuse and more recently has been seen in several...

Adversarial text substitution *(9)*

The patient has a history of lumbago and chronic alcohol dependence and more recently has been seen in several...

Opioid abuse risk: High

Opioid abuse risk: Low

277.7	Metabolic syndrome
429.9	Heart disease, unspecified
278.00	Obesity, unspecified

Adversarial coding *(13)*

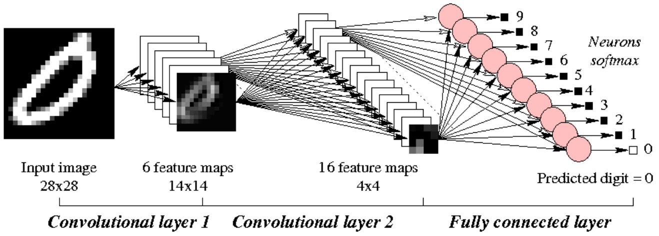
401.0	Benign essential hypertension
272.0	Hypercholesterolemia
272.2	Hyperglyceridemia
429.9	Heart disease, unspecified
278.00	Obesity, unspecified

Reimbursement: Denied

Reimbursement: Approved

Convolutional NN

(a)



(b)

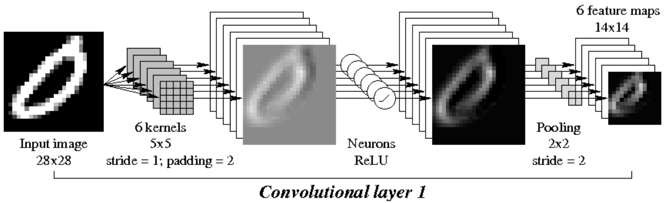


Figure 1: A convolutional neural network for the MNIST problem: global architecture (a) and detailed view of the first convolutional layer (b).

Recurrent NN

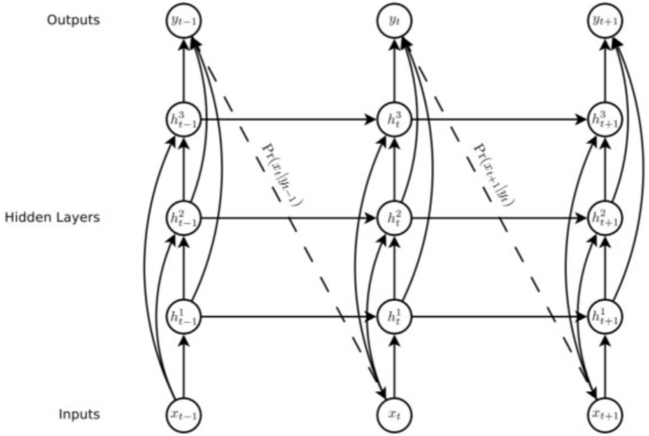


Figure 1: **Deep recurrent neural network prediction architecture.** The circles represent network layers, the solid lines represent weighted connections and the dashed lines represent predictions.

Elegant Machine Learning: Flux