How to Make your Neural Network Robust Enough to Outliers: Applications in Pattern Classification and Regression

Guilherme Barreto

gbarreto@ufc.br

www.researchgate.net/profile/Guilherme_Barreto2/

Center of Reference in Automation and Robotics (CENTAURO)
Department of Teleinformatics Engineering (DETI)
Federal University of Ceará (UFC)
Fortaleza, Ceará, Brazil

Fortaleza-CE, Brazil

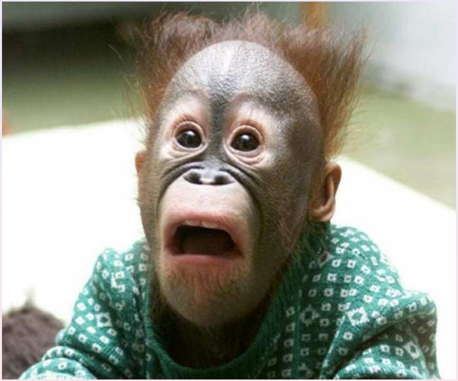
November 20, 2015

Outline of the Talk

Parte I

What is an Outlier?

Nobody knows!



What?

What is an Outlier?
Definition

- Yes, this is true! It is extremely difficult to define an outlier.

What is an Outlier?
Definition

- Yes, this is true! It is extremely difficult to define an outlier.
- It is a very subjective and unpleasant topic.

- Yes, this is true! It is extremely difficult to define an outlier.
- It is a very subjective and unpleasant topic.
- Worse, outliers are difficult to detect, especially in high-dimensional data.

What is an Outlier?
Definition

- Yes, this is true! It is extremely difficult to define an outlier.
- It is a very subjective and unpleasant topic.
- Worse, outliers are difficult to detect, especially in high-dimensional data.
- It is the type of sample you do not want in your dataset ...

- Yes, this is true! It is extremely difficult to define an outlier.
- It is a very subjective and unpleasant topic.
- Worse, outliers are difficult to detect, especially in high-dimensional data.
- It is the type of sample you do not want in your dataset ...
- ... because it can spoil your <u>standard</u> data modeling procedure!

- By standard, I mean the one based on the following assumptions:

  Assumption 1 - **Gaussianity** of errors.

- By standard, I mean the one based on the following assumptions:

  Assumption 1 - **Gaussianity** of errors.

  Assumption 2 - **Linearity** of the model.

- By standard, I mean the one based on the following assumptions:

  Assumption 1 - **Gaussianity** of errors.

  Assumption 2 - **Linearity** of the model.

  Assumption 3 - **Stationarity** of data distribution.

- By standard, I mean the one based on the following assumptions:

Assumption 1 - **Gaussianity** of errors.

Assumption 2 - **Linearity** of the model.

Assumption 3 - **Stationarity** of data distribution.

Assumption 4 - **Sufficiency of information** in the samples.

From the exposed, outliers are...

... data samples that do not fit to our *current assumptions* about the data generating process.

I got it! Outliers suck!
But...

Well, the one million dollar question is then...

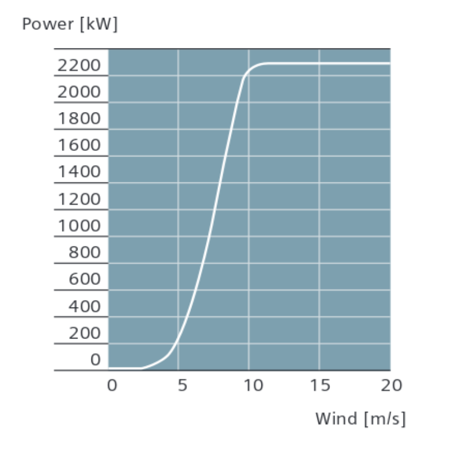
To remove or not to remove
the outliers from my dataset?

I'll try to answer this question along the talk with several examples!

- Let us start with a real-world problem I faced some time ago.
- The determination of the power curve of a wind turbine[1].

[1] M. Lydia, S. S. Kumar, A. I. Selvakumar & G. E. P. Kumar (2014). "A comprehensive review on wind turbine power curve modeling techniques", Renewable and Sustainable Energy Reviews, 30:452-460.

Example: Siemens Wind Turbine SWT-2.3-108



Initial Data Acquisition

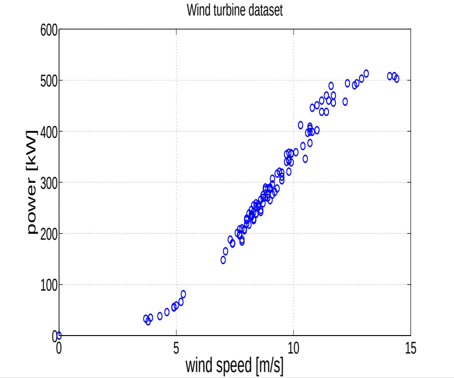


Figure : Initial data samples collected by our acquisition system *in situ*.

Octave/Matlab Code (polynomial curve fitting)

```
>> load aerogerador_reduced.dat; % load data samples
>> v=aerogerador(:,1); % speed measurements
>> p=aerogerador(:,2); % power measurements
>> figure; plot(v,p,'bo'); grid; hold on;
>> xlabel('wind speed [m/s]'); ylabel('power [kW]');
>> k=5; % order of polynomial
>> B=polyfit(v,p,k); % fit a polynomial to data
>> vv=min(v):0.1:max(v); vv=vv'; % grid over range of wind data
>> ypred=polyval(B,vv); % predictions for grid values
>> plot(vv,ypred,'k-'); hold off; % overlap curve to data
```

Initial Curve Fitting

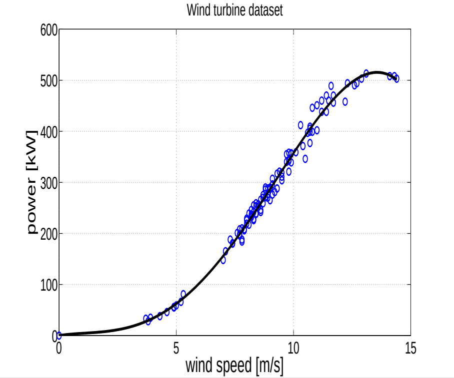


Figure : Initial attempt to fit a 5th order polynomial to the data.

Initial Data Acquisition

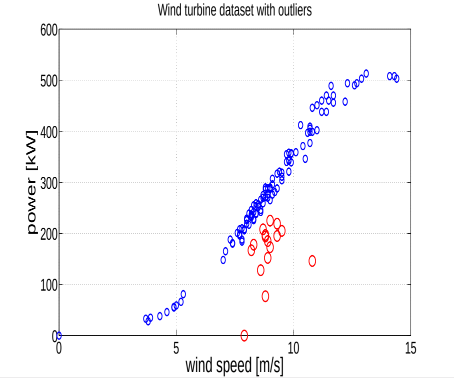


Figure : Initial data samples collected by our acquisition system *in situ*.

Curve Fitting with Outliers

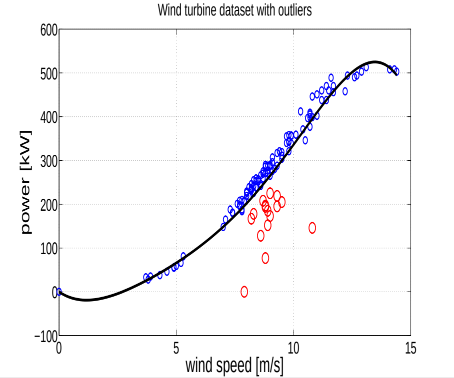


Figure : Fitting a 5th order polynomial to the data with outliers.

Curve Fitting with Outliers

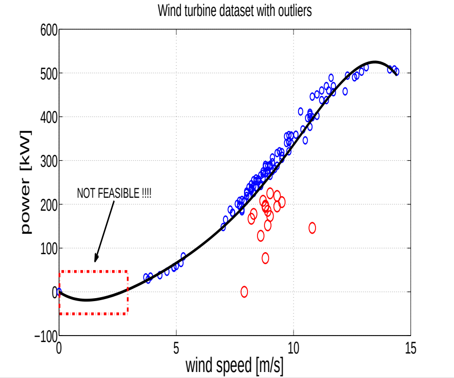


Figure : According to this model, the power generator acts as a “fan” in a certain range, demanding energy instead of generating it!

Curve Fitting with Outliers

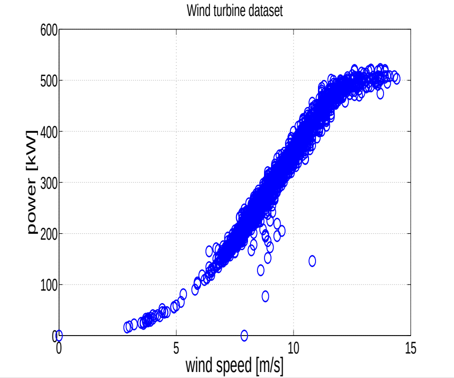


Figure : After acquiring enough sample, it seems that the number of outliers is very small compared to the whole set. Do they still affect the curve fitting process?

However, after getting 2250 samples...

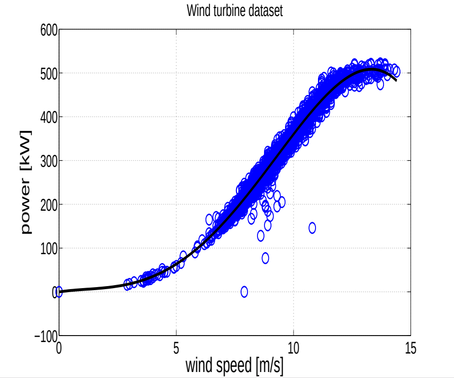
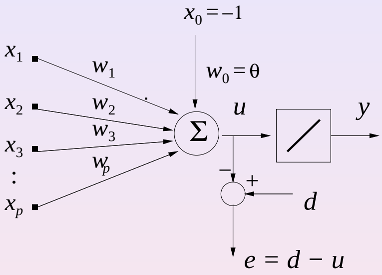


Figure : After acquiring enough samples, the number of outliers is so small that they DO NOT affect the curve fitting process AT ALL.

Parte II

OLS and LMS Estimation Methods

- Consider the linear combiner model as shown below.

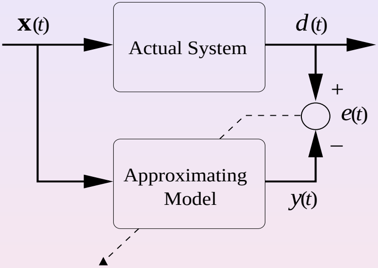


- Model output at time t is given by

$$y(t) = \sum_{j=0}^{p} w_j(t)x_j(t), \quad (1)$$

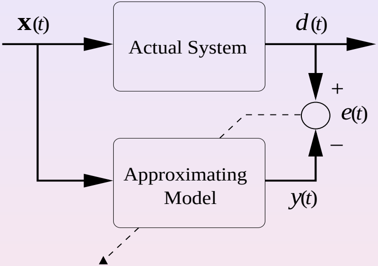
$$= \mathbf{w}^T(t)\mathbf{x}(t). \quad (2)$$

- It can be used for function approximation.



- For that, we need to estimate the parameter vector $\mathbf{w} \in \mathbb{R}^p$.
- We'll use observed data $\{\mathbf{x}(t), d(t)\}_{t=1}^{N}$ to estimate $\mathbf{w}$.
- I'll discuss in this talk two parameter estimation methods:
  1. OLS - Ordinary Least Squares (a batch method).

- It can be used for function approximation.



- For that, we need to estimate the parameter vector $\mathbf{w} \in \mathbb{R}^p$.
- We'll use observed data $\{\mathbf{x}(t), d(t)\}_{t=1}^{N}$ to estimate $\mathbf{w}$.
- I'll discuss in this talk two parameter estimation methods:
  1. OLS - Ordinary Least Squares (a batch method).
  2. LMS - Least Mean Squares (an online method).

- Initially proposed in 1795 by **Carl Friedrich Gauss** (1777 - 1855), but published only in 1809[2].
- However, **Adrien Marie Legendre** (1752-1833) developed the same method independently and published it first in 1806[3].

- Both applied the method to compute orbits of celestial bodies using measurements from telescopes.

[2] C. F. Gauss (1809). "Theoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientium", Perthes et I. H. Besser, Hamburgi.

[3] A. M. Legendre (1805). "Nouvelles Méthodes pour la Détermination des Orbites des Comètes", Courcier, Paris.

- OLS Optimality criterion: Sum of Squared Errors (SSE)

$$
\begin{aligned}
J_{OLS}(\mathbf{w}) &= \sum_{t=1}^{N} e^2(t) = \sum_{t=1}^{N} (d(t) - y(t))^2, && (3) \\
&= \sum_{t=1}^{N} (d(t) - \mathbf{w}^T \mathbf{x}(t))^2 = \|\mathbf{e}\|^2, && (4)
\end{aligned}
$$

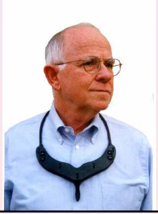
where $\|\mathbf{e}\|$ is the Euclidean norm of the error vector $\mathbf{e}$.

- The optimal estimate of the parameter vector $\mathbf{w}$ is given by

$$\boxed{\mathbf{w} = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{d}} \qquad (5)$$

where $\mathbf{X} = [\mathbf{x}(1) \mid \mathbf{x}(2) \mid \cdots \mid \mathbf{x}(N)] \in \mathbb{R}^{(p+1)\times N}$ and $\mathbf{d} = [d(1) \mid d(2) \mid \cdots \mid d(N)]^T \in \mathbb{R}^N$.

- Proposed in 1960 by **Dr. Bernard Widrow** (1929 -) and his first PhD. student Marcian "Ted" Hoff, Jr. (1937 -)[4].
- Ted Hoff is considered the "inventor" of the microprocessor (1st patent), entering the Intel Corporation in 1967 as the employee number 12.
- There, he designed the 1st computer-on-a-chip microprocessor (1968), which came on the market as the Intel 4004 (1971), starting the microcomputer industry[5].



[4] B. Widrow and M.E. Hoff, Jr., "Adaptive Switching Circuits," IRE WESCON Convention Record, 4:96-104, August 1960.

[5] More at www.thocp.net/biographies/hoff_ted.html

- LMS Optimality criterion: Instantaneous Squared Error (ISE)

$$
\begin{aligned}
J_{LMS}(t) &= e^2(t) = (d(t) - y(t))^2, && (6)\\
&= (d(t) - \mathbf{w}^T(t)\mathbf{x}(t))^2, && (7)
\end{aligned}
$$

where $\mathbf{w}(t)$ is the current estimation of the parameter vector.

- The recursive updating rule for $\mathbf{w}(t)$ is given by

$$
\begin{aligned}
\mathbf{w}(t+1) &= \mathbf{w}(t) - \alpha \frac{\partial J(t)}{\partial \mathbf{w}(t)}, && (8)\\
&= \mathbf{w}(t) + \alpha e(t)\mathbf{x}(t), && (9)\\
&= \mathbf{w}(t) + \alpha(d(t) - y(t))\mathbf{x}(t), && (10)\\
&= \mathbf{w}(t) + \alpha(d(t) - \mathbf{w}^T(t)\mathbf{x}(t))\mathbf{x}(t), && (11)
\end{aligned}
$$

where $0 < \alpha < 1$ is the learning step.

- In the context of neural networks, the linear combiner together with the OLS rule give us the **Optimal Linear Associative Memory** (OLAM) model by Kohonen & Ruohonen[6]

[6] T. Kohonen and M. Ruohonen (1973). "Representation of Associated Data by Matrix Operators", IEEE Trans. on Computers, vol 22, no. 7, p. 701–702.

[7] B. Widrow and M. E. Hoff, Jr. (1960). "Adaptive switching circuits". In: Proc. IRE WESCON Conf. Rec. Part 4, p. 96–104.

- In the context of neural networks, the linear combiner together with the OLS rule give us the **Optimal Linear Associative Memory** (OLAM) model by Kohonen & Ruohonen[6]
- In the context of neural networks, the linear combiner together with the LMS rule give us the **ADAptive LINear Element** (ADALINE) model by Widrow & Hoff[7]

[6] T. Kohonen and M. Ruohonen (1973). "Representation of Associated Data by Matrix Operators", IEEE Trans. on Computers, vol 22, no. 7, p. 701–702.

[7] B. Widrow and M. E. Hoff, Jr. (1960). "Adaptive switching circuits". In: Proc. IRE WESCON Conf. Rec. Part 4, p. 96–104.

- OLS/LMS methods are widely used in multilayer feedforward/recurrent neural network architectures.

- OLS/LMS methods are widely used in multilayer feedforward/recurrent neural network architectures.
- To name a few:

- OLS/LMS methods are widely used in multilayer feedforward/recurrent neural network architectures.
- To name a few:
  1. MLP - Multilayer Perceptron (LMS, output layer)

- OLS/LMS methods are widely used in multilayer feedforward/recurrent neural network architectures.
- To name a few:
  1. MLP - Multilayer Perceptron (LMS, output layer)
  2. RBF - Radial Basis Functions Network (OLS, output layer)

- OLS/LMS methods are widely used in multilayer feedforward/recurrent neural network architectures.
- To name a few:
  1. MLP - Multilayer Perceptron (LMS, output layer)
  2. RBF - Radial Basis Functions Network (OLS, output layer)
  3. ELM - Extreme Learning Machine (OLS, output layer)

- OLS/LMS methods are widely used in multilayer feedforward/recurrent neural network architectures.
- To name a few:
  1. MLP - Multilayer Perceptron (LMS, output layer)
  2. RBF - Radial Basis Functions Network (OLS, output layer)
  3. ELM - Extreme Learning Machine (OLS, output layer)
  4. NoProp - No-Propagation Network (LMS, output layer)

- OLS/LMS methods are widely used in multilayer feedforward/recurrent neural network architectures.
- To name a few:
  1. MLP - Multilayer Perceptron (LMS, output layer)
  2. RBF - Radial Basis Functions Network (OLS, output layer)
  3. ELM - Extreme Learning Machine (OLS, output layer)
  4. NoProp - No-Propagation Network (LMS, output layer)
  5. ESN - Echo-State Network (OLS, output layer)

Assumptions Behind OLS/LMS Estimation Methods

1. Cost functions that assign same importance to all errors.

Assumptions Behind OLS/LMS Estimation Methods

1. Cost functions that assign same importance to all errors.
2. All errors contribute the same way to the final solution.

Assumptions Behind OLS/LMS Estimation Methods

1. Cost functions that assign same importance to all errors.
2. All errors contribute the same way to the final solution.
3. Solution is optimal only under Gaussian errors!

Assumptions Behind OLS/LMS Estimation Methods

1. Cost functions that assign same importance to all errors.
2. All errors contribute the same way to the final solution.
3. Solution is optimal only under Gaussian errors!
4. Outliers produce larger errors ...

Assumptions Behind OLS/LMS Estimation Methods

1. Cost functions that assign same importance to all errors.
2. All errors contribute the same way to the final solution.
3. Solution is optimal only under Gaussian errors!
4. Outliers produce larger errors ...
5. ... then biasing the solution towards outliers locations.

Parte III

OLS/LMS Performance and Outliers

Lung Cancer Dataset w/o USA sample

Sample	Country	Cigarette per capita	Death per million
1	Australia	480	180
2	Canada	500	150
3	Denmark	380	170
4	Finland	1100	350
5	Great Britain	1100	460
6	Iceland	230	60
7	Netherlands	490	240
8	Norway	250	90
9	Sweden	300	110
10	Switzerland	510	250

Table : Consumption per capita of cigarettes in several countries in 1930 and death rates due to lung cancer in 1950[b].

[a] Source: D. Freedman, R. Pisani and R. Purves (2007), "Statistics", 4th edition, W. W. Norton & Company.
[b] Source: D. Freedman, R. Pisani and R. Purves (2007), "Statistics", 4th edition, W. W. Norton & Company.

Lung Cancer Dataset w/o USA sample (scatterplot)

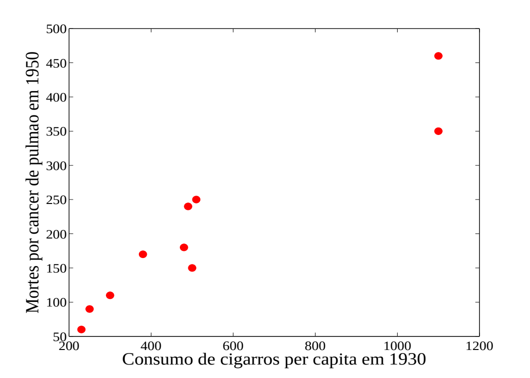


Figure : Scatterplot of the Lung Cancer dataset w/o USA sample.

Octave/Matlab Code (linear regression and OLS estimation)

```
>> x=[480; 500; 380; 1100; 1100; 230; 490; 250; 300; 510];
>> y=[180; 150; 170; 350; 460; 60; 240; 90; 110; 250];
>> n=length(x);
>> X=[ones(n,1) x];
>> B=X\y    % Uses QR decomposition
B=
9.1393
0.3687
>> B=regress(y,X)
B=
9.1393
0.3687
>> B=pinv(X)*y     % Uses SVD method (recommended)
B=
9.1393
0.3687
```

Lung Cancer Dataset w/o USA sample (regression line)

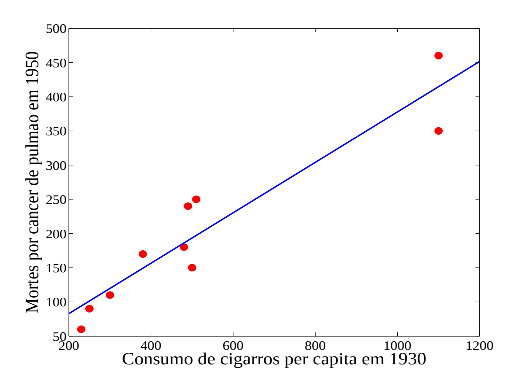


Figure : Regression line ($\hat{y} = 0.37x + 9.14$) for Lung Cancer data w/o USA sample.

Lung Cancer Dataset with USA sample

Sample	Country	Cigarette per capita	Death per million
1	Australia	480	180
2	Canada	500	150
3	Denmark	380	170
4	Finland	1100	350
5	Great Britain	1100	460
6	Iceland	230	60
7	Netherlands	490	240
8	Norway	250	90
9	Sweden	300	110
10	Switzerland	510	250
11	**United States**	**1300**	**200**

Table : Consumption per capita of cigarettes in several countries in 1930 and death rates due to lung cancer in 1950[b].

[a] Source: D. Freedman, R. Pisani and R. Purves (2007), "Statistics", 4th edition, W. W. Norton & Company.

[b] Source: D. Freedman, R. Pisani and R. Purves (2007), "Statistics", 4th edition, W. W. Norton & Company.

Lung Cancer Dataset with USA sample (scatterplot)

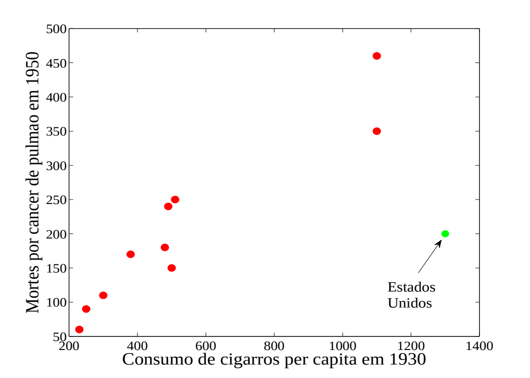


Figure : Scatterplot of the Lung Cancer data with USA sample.

Octave/Matlab Code (linear regression and OLS estimation)

```
>> x=[480; 500; 380; 1100; 1100; 230; 490; 250; 300; 510; 1300];
>> y=[180; 150; 170; 350; 460; 60; 240; 90; 110; 250; 200];
>> n=length(x);
>> X=[ones(n,1) x];
>> B=X\y    % Uses QR decomposition
B=
67.5609
0.2284
>> B=pinv(X)*y     % Uses SVD method (recommended)
B=
67.5609
0.2284
```

Lung Cancer Dataset with USA sample (regression lines)

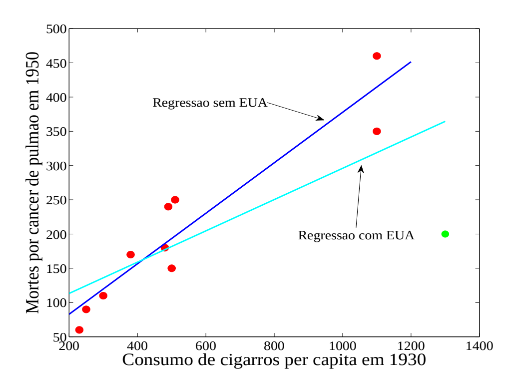


Figure : Blue: $\hat{y} = 0.23x + 67.56$ (with USA sample).
Cyan: $\hat{y} = 0.37x + 9.14$ (w/o USA sample).

Summary table with regression lines (OLS estimation)

Dataset	Slope	Bias	Regression line
Lung cancer w/o USA	9.14	0.37	$\hat{y}_i = 9.14 + 0.37x_i$
Lung cancer with USA	67.56	0.23	$\hat{y}_i = 67.56 + 0.23x_i$

Table : Regression lines whose parameters were estimated using the OLS method for the lung cancer dataset with and without the USA sample.

- So far, we have dealt with outliers in regression problems.

[8] B. Frenay and M. Verleysen (2014). "Classification in the presence of label noise: a survey", IEEE Trans. on Neural Networks and Learning Systems, 25(5):845-869

- So far, we have dealt with outliers in regression problems.
- However, another type of outlier is drawing much attention from Machine Learning community.

[8] B. Frenay and M. Verleysen (2014). "Classification in the presence of label noise: a survey", IEEE Trans. on Neural Networks and Learning Systems, 25(5):845-869

- So far, we have dealt with outliers in regression problems.
- However, another type of outlier is drawing much attention from Machine Learning community.
- It is called **label noise**[8] .

[8] B. Frenay and M. Verleysen (2014). "Classification in the presence of label noise: a survey", IEEE Trans. on Neural Networks and Learning Systems, 25(5):845-869

- So far, we have dealt with outliers in regression problems.
- However, another type of outlier is drawing much attention from Machine Learning community.
- It is called **label noise**[8] .
- Label noise may result of striking an incorrect key on a keyboard errors, misplaced decimal points, misjudgment of a specialist, recording or transmission errors.

[8] B. Frenay and M. Verleysen (2014). "Classification in the presence of label noise: a survey", IEEE Trans. on Neural Networks and Learning Systems, 25(5):845-869

- So far, we have dealt with outliers in regression problems.
- However, another type of outlier is drawing much attention from Machine Learning community.
- It is called **label noise**[8] .
- Label noise may result of striking an incorrect key on a keyboard errors, misplaced decimal points, misjudgment of a specialist, recording or transmission errors.
- Such outliers often go unnoticed because pattern classification is being more and more automatically executed by computers, without careful inspection or screening.

[8] B. Frenay and M. Verleysen (2014). "Classification in the presence of label noise: a survey", IEEE Trans. on Neural Networks and Learning Systems, 25(5):845-869

Ordinary Least Squares (OLS) Estimation

Pattern Classification Example (linear separable case)

Label noise scenario: labels are changed from Class -1 to +1

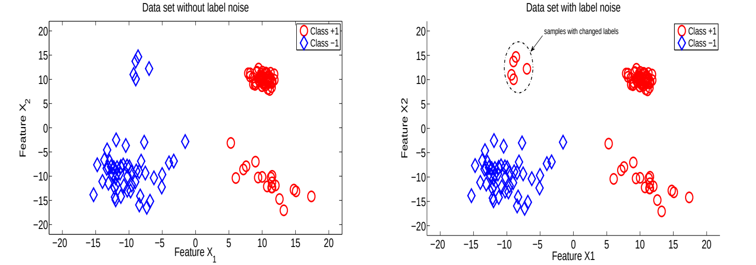


Figure : Simulating noise label by changing the labels of a few samples.

Ordinary Least Squares (OLS) Estimation

Pattern Classification Example (linear separable case)

Label noise scenario: labels are changed from Class -1 to +1

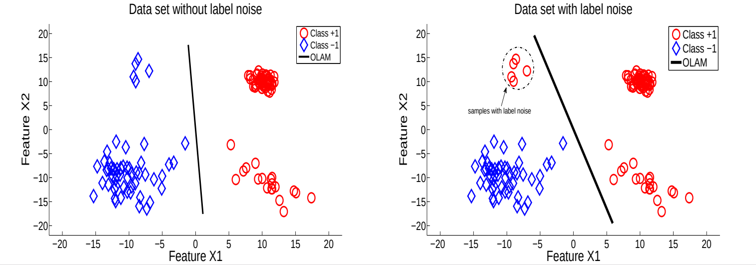


Figure : Decision line moved towards the sample with label noise.

- Another type of scenario where a sample can be wrongly considered an outlier involves **nonstationary**[9] problems.

[9] C.Alippi & R. Polikar (2014). "Special Issue on Learning In Nonstationary and Evolving Environments", IEEE Trans. on Neural Networks and Learning Systems, vol. 25, no. 1.

[10] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy & A. Bouchachia (2014). "A survey on concept drift adaptation", ACM Computing Surveys, 46(4), article no. 44.

- Another type of scenario where a sample can be wrongly considered an outlier involves **nonstationary**[9] problems.
- In classification, this is usually called **Concept Drift Problem**[10].

[9] C.Alippi & R. Polikar (2014). "Special Issue on Learning In Nonstationary and Evolving Environments", IEEE Trans. on Neural Networks and Learning Systems, vol. 25, no. 1.

[10] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy & A. Bouchachia (2014). "A survey on concept drift adaptation", ACM Computing Surveys, 46(4), article no. 44.

- Another type of scenario where a sample can be wrongly considered an outlier involves **nonstationary**[9] problems.
- In classification, this is usually called **Concept Drift Problem**[10].
- A neural network classifier must be capable of handling this type of situation, specially if it is designed for lifelong learning.

[9] C.Alippi & R. Polikar (2014). "Special Issue on Learning In Nonstationary and Evolving Environments", IEEE Trans. on Neural Networks and Learning Systems, vol. 25, no. 1.

[10] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy & A. Bouchachia (2014). "A survey on concept drift adaptation", ACM Computing Surveys, 46(4), article no. 44.

Nonstationary scenario: incoming of new samples of Class +1

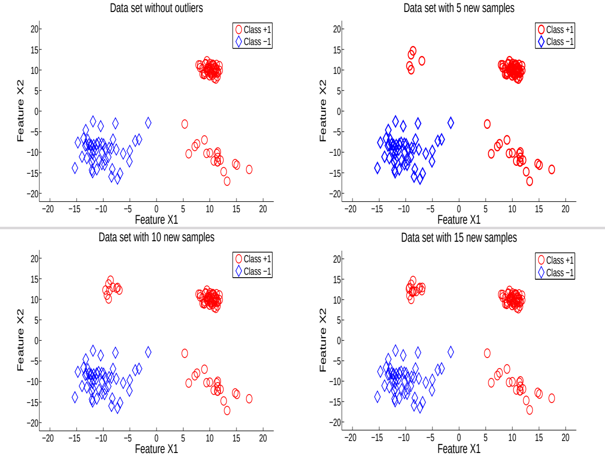


Figure : Simulating nonstationarity by adding new samples.

Ordinary Least Squares (OLS) Estimation
Pattern Classification Example (linear separable case)

Nonstationary scenario: 0, 5, 10 and 15 new samples to Class +1

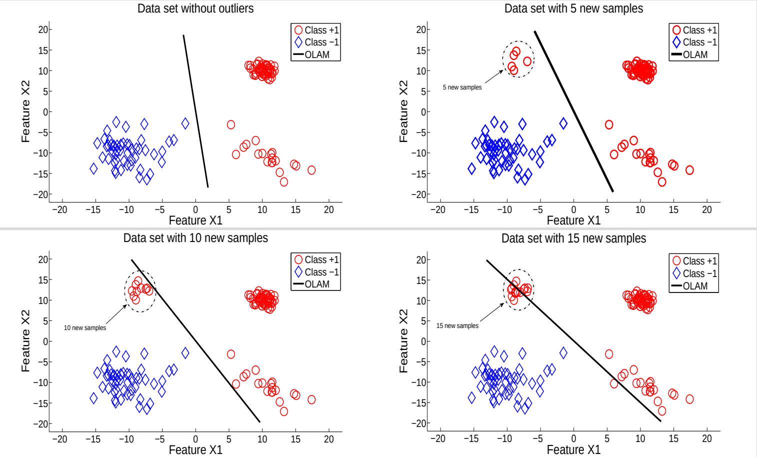


Figure : Decision lines for OLAM classifier trained with OLS rule.

Nonstationary scenario: 0, 5, 10 and 15 new samples to Class +1

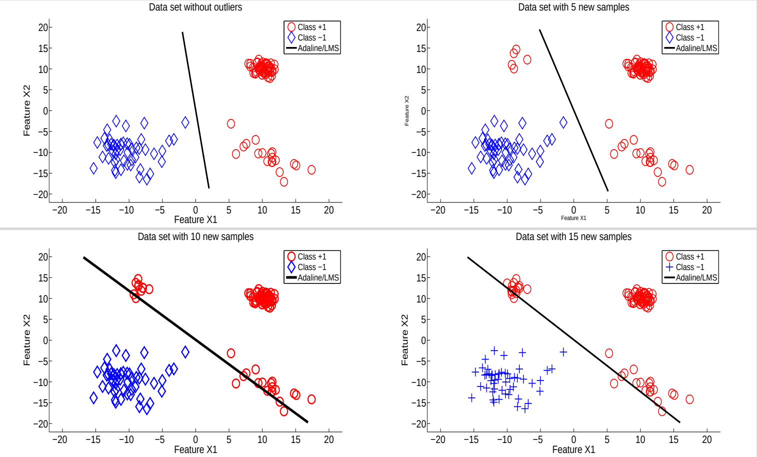


Figure : Decision lines for Adaline classifier trained with LMS rule.

Label noise scenario: labels are changed from Class -1 to +1

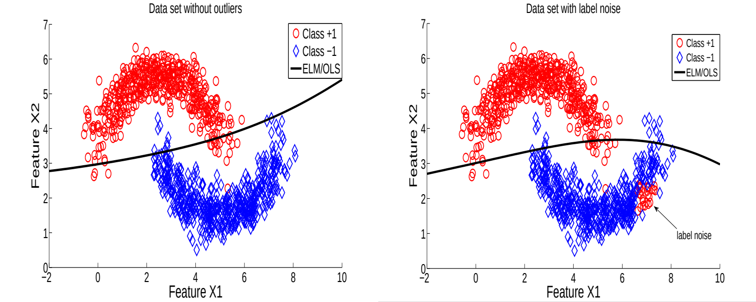


Figure : Decision curves for ELM classifier trained with OLS rule.

Nonstationary scenario: adding new samples to Class +1

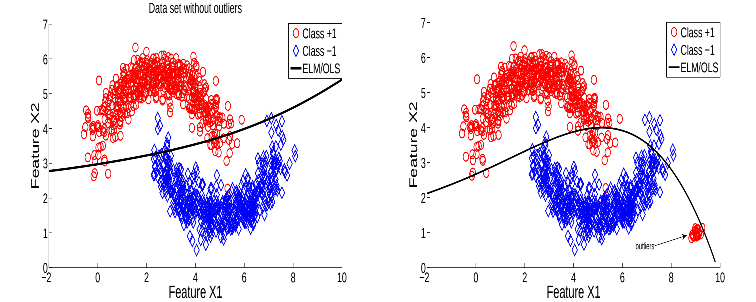
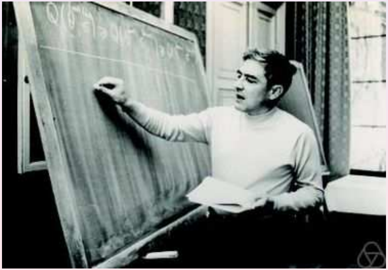


Figure : Decision lines for ELM classifier trained with OLS rule.

Parte IV

The M-Estimation Framework

- **Dr. Peter J. Huber**[11] (1934 -) introduced the concept of M-estimation.
- The M stands for "Maximum likelihood'" type estimation.
- Robustness is achieved by minimizing another function than the sum of the squared errors.



[11] P. J. Huber (1964). "Robust Estimation of a Location Parameter", Annals of Mathematical Statistics, 35(1):73–101.

- Based on Huber theory, a general M-estimator minimizes the following cost function:

$$J_M(\mathbf{w}) = \sum_{t=1}^{N} \rho(e(t)) = \sum_{t=1}^{N} \rho(d(t) - y(t)), \quad (12)$$

$$= \sum_{t=1}^{N} \rho(d(t) - \mathbf{w}^T \mathbf{x}(t)), \quad (13)$$

where the function $\rho(\cdot)$ computes the contribution of each error sample $e_{in} = d_{in} - y_{in}$ to the cost function.

- The OLS rule is a particular type of M-estimator, achieved when $\rho(e(t)) = e^2(t)$.

- The function ρ has the following properties:

Property 1 : $\rho(e(t)) \geq 0$.

Property 2 : $\rho(0) = 0$.

Property 3 : $\rho(e(t)) = \rho(-e(t))$.

Property 4 : $\rho(e(t)) \geq \rho(e(t')$, for $|e(t)| > |e(t')|$.

- For the sake of example, let us consider the Huber's cost function:

$$\rho(e(t)) = \begin{cases} \frac{1}{2}e^2, & |e(t)| \leq k \\ k|e(t)| - \frac{1}{2}k^2, & |e(t)| > k \end{cases} \qquad (14)$$

where the $k > 0$ is the error (or outlier) threshold.

- The corresponding weight function is given by

$$w(e(t)) = \begin{cases} 1, & |e(t)| \leq k \\ \frac{k}{|e(t)|}, & |e(t)| > k \end{cases} \qquad (15)$$

- The value $k = 1.345\hat{\sigma}$ is commonly used, where $\hat{\sigma}$ is itself a robust estimate of the dispersion of the errors.
- A usual approach is to take $\hat{\sigma} = \text{MAR}/0.6745$, where MAR is the median absolute residual.
- The constant value 0.6745 makes $\hat{\sigma}$ an unbiased estimate for Gaussian errors.

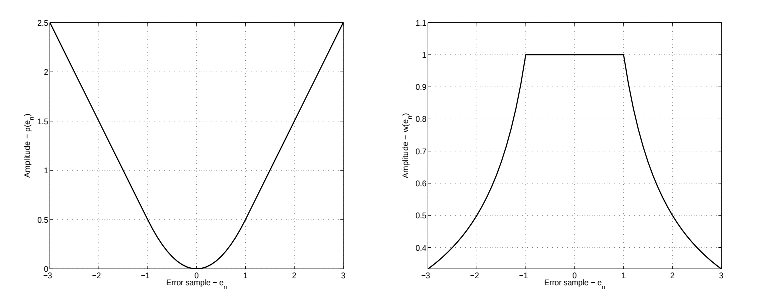


Figure : Huber's cost function (left) and weight function (right).

- In order to derive a learning rule based on M-estimators, let us define the *score function* $\psi(e(t)) = \partial\rho(e(t))/\partial e(t)$.
- Differentiating ρ w.r.t. the parameter vector $\mathbf{w}(t)$, we get

$$\sum_{t=1}^{N} \psi(y(t) - \mathbf{w}^T\mathbf{x}(t))\mathbf{x}(t)^T = \mathbf{0}, \qquad (16)$$

  where $\mathbf{0}$ is a $(p+1)$-dimensional row vector of zeros.
- Defining the weight function $w(e(t)) = \psi(e(t))/e(t)$, and let $w(t) = w(e(t))$, we arrive at

$$\sum_{t=1}^{N} w(t)(y(t) - \mathbf{w}^T\mathbf{x}(t))\mathbf{x}^T(t) = \mathbf{0}. \qquad (17)$$

- Thus, solving the previous equations corresponds to solving a weighted least-squares problem, since we want to minimize

$$J_M(\mathbf{w}) = \sum_{t=1}^{N} w^2(t)e^2(t) = \sum_{t=1}^{N} w^2(e(t))e^2(t) \qquad (18)$$

- However, the weights depend on the residuals, the residuals depend upon the estimated coefficients, and the estimated coefficients depend upon the weights.

- Hence, an iterative estimation method is required. The *iteratively reweighted least-squares*[12] (IRLS) is commonly used of this purpose.

[12] J. Fox (2002). "An R and S-PLUS Companion to Applied Regression", SAGE Publications.

Iteratively Reweighted Least Squares (IRLS) Algorithm

- **Step 1** - Provide an initial estimate $\mathbf{w}(0)$ using the standard OLS rule. Let $n = 1$.
- **Step 2** - At each iteration n, compute the residuals $e^{(n-1)}(t)$ and their corresponding weights $w^{(n-1)}(t) = w[e^{(n-1)}(t)]$ for all input patterns $\mathbf{x}(t)$, $t = 1, \ldots, N$, using the current estimate of the parameter vector.

  Step 3 - Solve for new weighted-least-squares estimate of $\mathbf{w}(t)$:

$$\mathbf{w}^{(n)} = (\mathbf{X}\mathbf{W}^{(n-1)}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{W}^{(n-1)}\mathbf{d}, \qquad (19)$$

  where $\mathbf{W}^{(n-1)} = \text{diag}\{w^{(n-1)}(t)\}$ is an $N \times N$ weight matrix for the residuals of all N input patterns. Let $n = n + 1$ and repeat Steps 2 and 3 until the convergence of the parameter vector $\mathbf{w}^{(n)}$.

- LMM cost function[13]

$$\begin{aligned} J_{LMM}(t) &= \rho(e(t)) = \rho(d(t) - y(t)), && (20) \\ &= \rho(d(t) - \mathbf{w}^T(t)\mathbf{x}(t)), && (21) \end{aligned}$$

where $\mathbf{w}(t)$ is the current estimation of the parameter vector.

- The recursive updating rule for $\mathbf{w}(t)$ is given by

$$\begin{aligned} \mathbf{w}(t+1) &= \mathbf{w}(t) - \alpha \frac{\partial J_{LMM}(t)}{\partial \mathbf{w}(t)}, && (22) \\ &= \mathbf{w}(t) + \alpha w(e(t)) e(t) \mathbf{x}(t), && (23) \\ &= \mathbf{w}(t) + \alpha \psi(e(t)) \mathbf{x}(t), && (24) \end{aligned}$$

where we used the fact that $w(e(t)) = \psi(e(t))/e(t)$.

[13] Y. Zou, S. C. Chan & T. S. Ng (2000). "Least mean M-estimate algorithms for robust adaptive filtering in impulsive noise", IEEE Transactions on Circuits and Systems II, 47(12):1564–1569.

LMS estimation

- **Cost function**: $J_{LMS}(t) = e^2(t) = (d(t) - y(t))^2$
- **Learning rule**:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \alpha e(t)\mathbf{x}(t) \tag{25}$$

LMM estimation

- **Cost function**: $J_{LMM}(t) = \rho(e(t)) = \rho(d(t) - y(t))$
- **Learning rule**:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \alpha w(e(t))e(t)\mathbf{x}(t) \tag{26}$$

Parte V

Robust Linear Neural Network Models

Nonstarionary scenario: 0, 5, 10, 15 new samples to Class $+1$

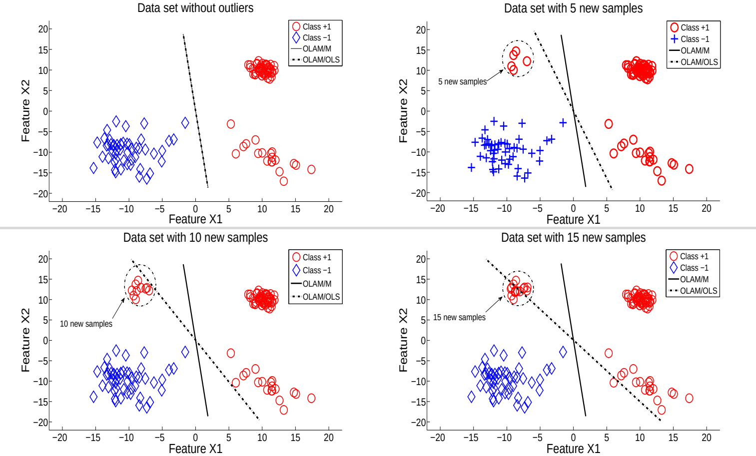


Figure : Decision lines for OLAM classifier trained with OLS and M-estimation (Andrews weight function).

M-Estimation for Robust Classification

Pattern Classification Example (linear separable case)

Nonstarionary scenario: 0, 5, 10, 15 new samples to Class +1

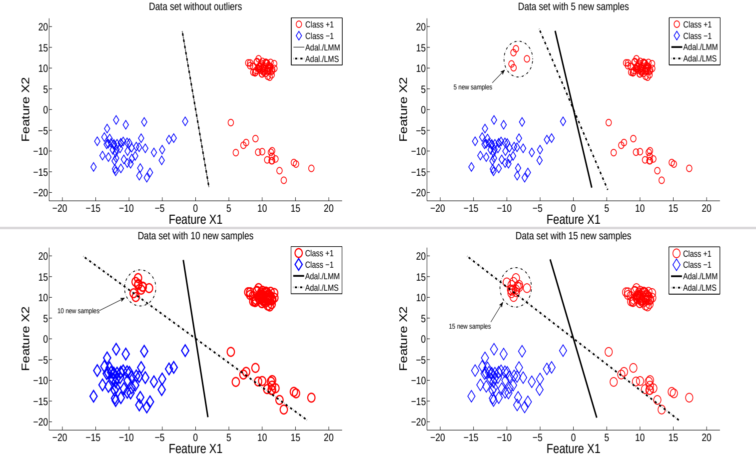


Figure : Decision lines for ADALINE classifier trained with LMS and LMM learning rules (Talwar weight function).

Parte VI

Robust Nonlinear Neural Network Models

Label noise scenario: labels are changed from Class -1 to +1

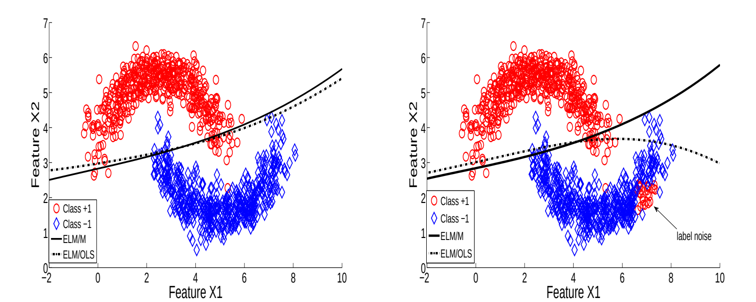


Figure : Decision curves for the ELM classifier ($U(-0.01, 0.01)$, 4 tanh hidden neurons) trained with OLS rule and M-estimation (Andrews weight function).

Ordinary Least Squares (OLS) Estimation
Pattern Classification Example (nonlinear separable case)

Nonstationary scenario: adding new samples to Class $+1$

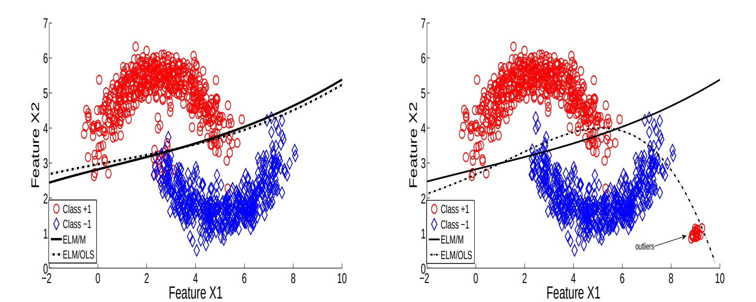


Figure : Decision curves for the ELM classifier ($U(-0.01, 0.01)$, 4 tanh hidden neurons) trained with OLS rule and M-estimation (Andrews weight function).

Label noise scenario: labels are changed from Class -1 to $+1$

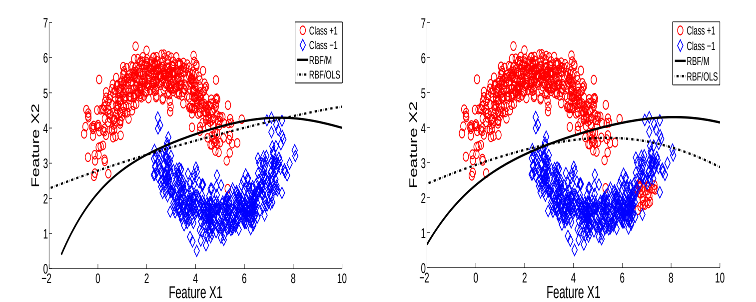


Figure : Decision curves for the RBF classifier (K-means function, 4 Gaussian basis functions) trained with OLS rule and M-estimation (Andrews weight function).

Nonstationary scenario: adding new samples to Class +1

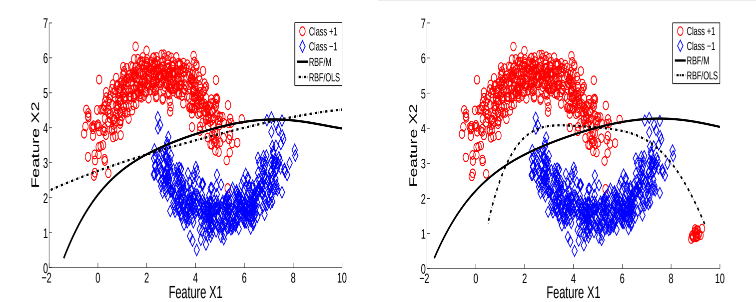
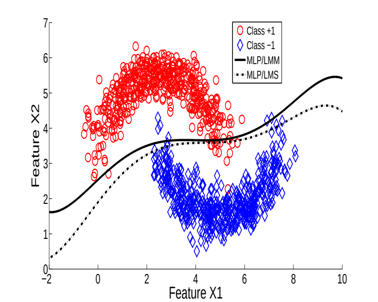


Figure : Decision curves for the RBF classifier (K-means function, 4 Gaussian basis functions) trained with OLS rule and M-estimation (Andrews weight function).

M-Estimation for Robust Classification

Pattern Classification Example (nonlinear separable case)

Label noise scenario: labels are changed from Class -1 to +1



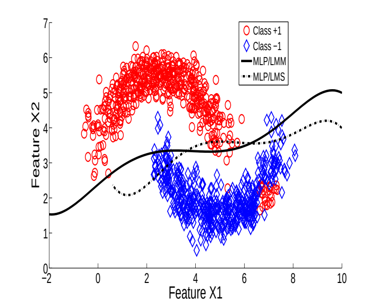


Figure : Decision curves for the MLP classifier (backprop, 4 tanh hidden neurons) trained with LMS rule and LMM (Talwar weight function).

Ordinary Least Squares (OLS) Estimation

Pattern Classification Example (nonlinear separable case)

Nonstationary scenario: adding new samples to Class +1

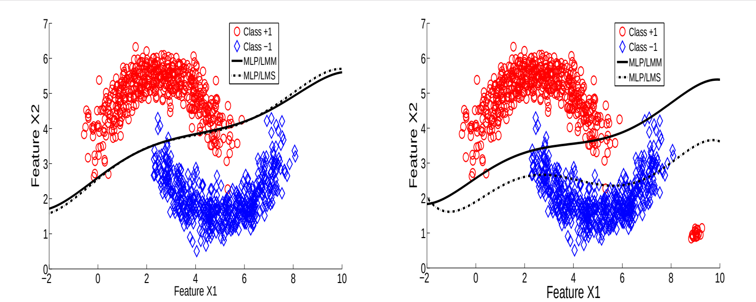


Figure : Decision curves for the MLP classifier (backprop, 4 tanh hidden neurons) trained with LMS rule and LMM (Talwar weight function).

Parte VII

Conclusions

- Since detecting outliers is a very tricky task, ...

- Since detecting outliers is a very tricky task, ...
- ... the removal of suspected data samples are not recommended, because...

- Since detecting outliers is a very tricky task, ...
- ... the removal of suspected data samples are not recommended, because...
- ... they can turn to be relevant to your problem inference.

- Since detecting outliers is a very tricky task, ...
- ... the removal of suspected data samples are not recommended, because...
- ... they can turn to be relevant to your problem inference.
- A better and wiser approach is to use M-estimators!

- They provide resilience to outliers.

[14] When samples suspected to be outliers turn out to be relevant data samples as time goes by.

- They provide resilience to outliers.
- They are simple to incorporate into the neural model.

[14] When samples suspected to be outliers turn out to be relevant data samples as time goes by.

- They provide resilience to outliers.
- They are simple to incorporate into the neural model.
- Only one additional parameter to tune.

[14] When samples suspected to be outliers turn out to be relevant data samples as time goes by.

- They provide resilience to outliers.
- They are simple to incorporate into the neural model.
- Only one additional parameter to tune.
- They can be used in batch and online learning rules.

[14] When samples suspected to be outliers turn out to be relevant data samples as time goes by.

- They provide resilience to outliers.
- They are simple to incorporate into the neural model.
- Only one additional parameter to tune.
- They can be used in batch and online learning rules.
- They fit well to nonstationary scenario[14].

[14] When samples suspected to be outliers turn out to be relevant data samples as time goes by.

- Guilherme A. Barreto & Ana Luiza B. P. Barros (2015). “On the Design of Robust Linear Pattern Classifiers based on M-Estimators”, *Neural Processing Letters*, 42(1):119-137.
- Guilherme A. Barreto & Ana Luiza B. P. Barros (2015). “A Robust Extreme Learning Machine for Pattern Classification with Outliers”, *Neurocomputing*, In Press.
- K. Zhang & M. Luo (2015). “Outlier-robust extreme learning machine for regression problems”, *Neurocomputing*, 151:1519-1527.
- P. Horata, S. Chiewchanwattana & K. Sunat (2013). “Robust Extreme Learning Machine”, *Neurocomputing*, 102:31-44.

CENTAURO
CENTRO DE REFERÊNCIA EM AUTOMAÇÃO E ROBÓTICA

UNIVERSIDADE FEDERAL DO CEARÁ

NUTEC
FUNDAÇÃO NÚCLEO DE TECNOLOGIA INDUSTRIAL DO CEARÁ

CNPq
Conselho Nacional de Desenvolvimento Científico e Tecnológico

Thank you very much!