

**TERM: M1-RAIA**

**SEMESTER: 2**

**AY: 2022-2023**

# Abdelbacet Mhamdi

Dr.-Ing. in Electrical Engineering

Senior Lecturer at ISET Bizerte

abdelbacet.mhamdi@bizerte.r-iset.tn

## ARTIFICIAL INTELLIGENCE - PART 2

LAB MANUAL



**Higher Institute of Technological Studies of Bizerte**

---

Available at <https://github.com/a-mhamdi/isetbz/>



## --- HONOR CODE ---

THE UNIVERSITY OF NORTH CAROLINA AT CHAPEL HILL

Department of Physics and Astronomy

<http://physics.unc.edu/undergraduate-program/labs/general-info/>

“During this course, you will be working with one or more partners with whom you may discuss any points concerning laboratory work. However, you must write your lab report, in your own words.

Lab reports that contain identical language are not acceptable, so do not copy your lab partner’s writing.

If there is a problem with your data, include an explanation in your report. Recognition of a mistake and a well-reasoned explanation is more important than having high-quality data, and will be rewarded accordingly by your instructor. A lab report containing data that is inconsistent with the original data sheet will be considered a violation of the Honor Code.

Falsification of data or plagiarism of a report will result in prosecution of the offender(s) under the University Honor Code.

On your first lab report you must write out the entire honor pledge:

---

**The work presented in this report is my own, and the data was obtained by my lab partner and me during the lab period.**

---




On future reports, you may simply write “Laboratory Honor Pledge” and sign your name.”

# Contents

<b>1</b>	<b>Linear Regression</b>	<b>1</b>
<b>2</b>	<b>Logistic Regression</b>	<b>2</b>
<b>3</b>	<b><math>k</math>-Nearest Neighbors</b>	<b>3</b>
<b>4</b>	<b>Support Vector Machine</b>	<b>5</b>
<b>5</b>	<b><math>K</math>-Means for Clustering</b>	<b>7</b>

---

In order to activate the virtual environment and launch **Jupyter Notebook**, we recommend you to proceed as follow

- ① Press simultaneously the keys  &  on the keyboard. This will open the dialog box **Run**;
- ② Then enter `cmd` in the command line and confirm with  key on the keyboard;
- ③ Type the instruction `jlai.bat` in the console prompt line;



- ④ Finally press the  key.

---

**LEAVE THE SYSTEM CONSOLE ACTIVE.**

# 1 | Linear Regression

<b>Student's name</b>	.....	.....	.....
	.....	.....	.....
	.....	.....	.....
<b>Score</b> /20	.....	.....	.....

## Detailed Credits

<b>Anticipation (4 points)</b>	.....	.....	.....
<b>Management (2 points)</b>	.....	.....	.....
<b>Testing (7 points)</b>	.....	.....	.....
<b>Data Logging (3 points)</b>	.....	.....	.....
<b>Interpretation (4 points)</b>	.....	.....	.....



The notebook is available at <https://github.com/a-mhamdi/cosnip/> → Julia → ml → linear-regression.jl

Linear regression is a type of machine learning algorithm that is used to predict a continuous outcome variable based on one or more predictor variables. It is a type of regression analysis that models the relationship between the dependent variable and the independent variables by fitting a straight line to the data. This line can then be used to make predictions about the value of the dependent variable based on the values of the independent variables. Linear regression is a simple and popular method for modeling relationships in data and is often used as a starting point for more complex machine learning algorithms.

This kind of supervised learning deals with labelled data. A subset of this data is used later to predict in continuous form. Regression problems involve tasks where the outputs form generally a set of real numbers. They often follow linear formats.

## 2 | Logistic Regression

<b>Student's name</b>	.....	.....	.....
	.....	.....	.....
	.....	.....	.....
<b>Score</b> /20	.....	.....	.....

### Detailed Credits

<b>Anticipation (4 points)</b>	.....	.....	.....
<b>Management (2 points)</b>	.....	.....	.....
<b>Testing (7 points)</b>	.....	.....	.....
<b>Data Logging (3 points)</b>	.....	.....	.....
<b>Interpretation (4 points)</b>	.....	.....	.....



The notebook is available at <https://github.com/a-mhamdi/cosnip/> → Julia → ml → logistic-regression.jl

Logistic regression is a type of statistical model that is used to predict the likelihood of an event occurring. It is a type of regression analysis that is used when the dependent variable is binary, meaning it can only take on one of two values, such as 0 or 1. Logistic regression is used to model the relationship between a dependent variable and one or more independent variables by fitting a logistic curve to the data. This curve can then be used to make predictions about the likelihood of an event occurring.

1  
2  
3

\*\*\*

### 3 | $k$ -Nearest Neighbors

<b>Student's name</b>	..... ..... .....	..... ..... .....	..... ..... .....
<b>Score</b> /20	.....	.....	.....

#### Detailed Credits

<b>Anticipation (4 points)</b>	.....	.....	.....
<b>Management (2 points)</b>	.....	.....	.....
<b>Testing (7 points)</b>	.....	.....	.....
<b>Data Logging (3 points)</b>	.....	.....	.....
<b>Interpretation (4 points)</b>	.....	.....	.....



The notebook is available at <https://github.com/a-mhamdi/cosnip/> → Julia → ml → knn.jl

K-nearest neighbors (KNN) is a supervised learning algorithm used for classification and regression. In the classification case, the output is a class membership (e.g. "cat" or "dog"). In the regression case, the output is a continuous value (e.g. temperature).

To make a prediction for a new data point, the algorithm finds the closest data points in the training set (i.e. the "nearest neighbors") and takes the average (for regression) or the majority vote (for classification) of their outputs as the prediction for the new data point. The number of nearest neighbors ( $k$ ) is a hyperparameter that must be specified in advance.

KNN is a simple and effective algorithm, but it can be computationally expensive and is not suitable for large datasets. It is also sensitive to the scale and distribution of the data.

Here is an example of KNN implemented in Julia:

```
1 using Distances
2 using StatsBase
```



```
3
4 function knn(X::Array{T, 2}, y::Array{U, 1}, x::Array{T, 1}, k::Int)␣
   ↳where {T <: Real, U}
5     # Calculate distances between x and each point in X
6     dists = pairwise(Euclidean(), X, x)
7
8     # Sort the distances and indices in ascending order
9     sorted_dists = sortperm(dists)
10
11    # Take the top k distances and their corresponding y values
12    y_neighbors = y[sorted_dists[1:k]]
13
14    # Return the majority vote of the neighbors
15    return mode(y_neighbors)
16 end
```

This implementation uses the Distances and StatsBase packages to calculate distances and perform a majority vote. It takes as input the training data *X* and labels *y*, the test point *x*, and the number of nearest neighbors *k*, and returns the predicted label for *x*.

## 4 | Support Vector Machine

<b>Student's name</b>	.....	.....	.....
	.....	.....	.....
	.....	.....	.....
<b>Score</b> /20	.....	.....	.....

### Detailed Credits

<b>Anticipation (4 points)</b>	.....	.....	.....
<b>Management (2 points)</b>	.....	.....	.....
<b>Testing (7 points)</b>	.....	.....	.....
<b>Data Logging (3 points)</b>	.....	.....	.....
<b>Interpretation (4 points)</b>	.....	.....	.....



The notebook is available at <https://github.com/a-mhamdi/cosnip/> → Julia → ml → svm-clf.jl

Support vector machines (SVMs) are a type of supervised learning algorithm that can be used for classification or regression tasks. SVMs are a powerful and flexible tool for solving a wide range of machine learning problems, and have been widely used in many different fields, including text classification, image classification, and bioinformatics.

Here is an example of how you might implement an SVM in Julia for classification tasks:

```

1 using LIBSVM
2
3 # define the model
4 model = LIBSVM.SVM(SVC(), LinearKernel())
5
6 # train the model on the training data
7 LIBSVM.fit!(model, train_X, train_y)
8

```

```
9  # use the trained model to make predictions on the test data  
10 predictions = LIBSVM.predict(model, test_X)  
11  
12 # evaluate the model's performance  
13 accuracy = mean(test_y .== predictions)
```

Note that this is just one way to implement an SVM in Julia, and there are many other packages and approaches you can use. This example uses the LIBSVM package, which provides a convenient interface for working with SVMs in Julia.

## 5 | K-Means for Clustering

Student's name	.....	.....	.....
	.....	.....	.....
	.....	.....	.....
Score /20	.....	.....	.....

### Detailed Credits

Anticipation (4 points)	.....	.....	.....
Management (2 points)	.....	.....	.....
Testing (7 points)	.....	.....	.....
Data Logging (3 points)	.....	.....	.....
Interpretation (4 points)	.....	.....	.....



The notebook is available at <https://github.com/a-mhamdi/cosnip/> → Julia → ml → kmeans.jl

```

1      # Repeat until convergence
2      converged = false
3      while !converged
4          # Calculate distances between each point and each cluster
5          ↪center
6          dists = pairwise(Euclidean(), X, centers)
7
8          # Assign each point to the closest cluster center
9          clusters = argmin(dists, dims=1)
10
11         # Calculate the new cluster centers as the mean of all points
12         ↪in the cluster
13         new_centers = zeros(k, size(X, 2))
14         for i in 1:k

```

```
13         if sum(clusters .== i) > 0
14             new_centers[i, :] = mean(X[clusters .== i, :], dims=1)
15         else
16             # If a cluster is empty, randomly initialize a new
17             ↪center
18             new_centers[i, :] = X[rand(1:size(X, 1)), :]
19         end
20     end
21     # Check for convergence
22     converged = isapprox(centers, new_centers, rtol=1e-6)
23
24     # Update the cluster centers
25     centers = new_centers
26 end
27
28 return centers, clusters
29 end
```

This implementation uses the Clustering package to calculate distances. It takes as input the dataset  $X$  and the number of clusters  $k$ , and returns the cluster centers and the cluster assignments of each point.



The overall scope of this manual is to introduce **Artificial Intelligence (AI)** , through some numeric simulations, to the students enrolled at the master's program **RAIA**.

The topics discussed in this manuscript are as follow:

① Data Preprocessing

Standardization; Normalization; Encoder.

② Regression

Linear; Polynomial.

③ Classification

Logistic Regression;  $k$ -NN; SVM.

④ Clustering

K-Means.

*Julia; REPL; Pluto; Fuzzy; Flux; CUDA; artificial intelligence; regression; classification; clustering.*