INF283 - Exercise - 1

(**Exercise 1 deadline: 31st of August, 23.59**).
Submission details: Computer written or scanned pages. Give answers only, unless you want to show calculation.
Deliver here: https://mitt.uib.no/courses/12791/assignments

Weekly exercises are a compulsory part of the course. You will need to complete at least half of them. Weekly exercises give a total of 16 points to the final grade, of the total 8 exercises each then gives 2 points to you final grade, as long as you upload your answer to MittUIB.no/assignments before 23.59 on Fridays. They will then be reviewed, and points are added to your total grade score (If we see you have made an effort. So no score loss if your answer had some error in the calculation etc). If you have completed only a fraction of the tasks then you will get a fraction of 2 points.

If you follow these exercises and ask for help when you need it during the group sessions, it will help you a lot, especially through the more difficult parts of the course.

In this exercise we will refresh our memory in relevant math topics, practice formulating machine learning problems and learn about the KNN algorithm. You can also get help for installing Python and packages that are relevant for machine learning. Since this is background topics, this first exercise will take some time for many of you. Remember that spending more time on this exercise will make the rest of the course go much more smooth.

1. Background topics

This will be a **recap** from a small part of the courses **MAT121** (linear algebra) and **MAT101/MAT111** (Introduction to mathematics). It is important to understand these parts, since later topics will rely on them.

Linear algebra

Linear algebra is the branch of mathematics concerning linear equations, linear functions and and their representations through matrices and vector spaces.
In this section, we will practice matrix multiplication, matrix inversion and finding eigenvalues and eigenvectors of a matrix. Linear algebra is used in machine learning for among others high dimensional data and face recognition.

Matrix multiplication

A: **B:**

$$\begin{pmatrix} 2 & -1 \\ 1 & 3 \end{pmatrix} \qquad \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

1.a: Solve this matrix multiplication A*B:

$$\begin{pmatrix} 2 & -1 \\ 1 & 3 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Matrix inversion

For real numbers, a multiplicative inverse of number x is a number which when multiplied with x yields identity 1. .For example, the inverse of 2 is ½:

$2 * 1/2 = 1$

Analogously, the can define an inverse for a matrix. The inverse of matrix A, denoted by A^{-1}, is a matrix for which it holds that

$A \times A^{-1} = A^{-1} \times A = \mathbf{I}$**,**

where I is the identity matrix. I.g for a 2x2 matrix:

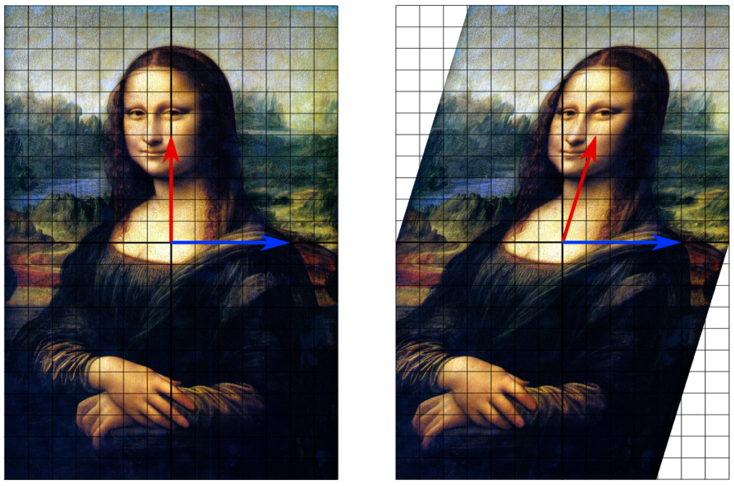
$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Check this link if you forgot the steps for matrix inversion:
https://www.mathsisfun.com/algebra/matrix-inverse.html

1.b: Find the inverse of the matrix found in 1.a

Eigenvectors and Eigenvalues

Vectors can be transformed linearly to a new space by multiplying them with a matrix (like Mona Lisa below). Vectors whose direction does not change under the transform are called eigenvectors. Below you can see two pictures. The left picture is original and the right one is the result of a shear mapping transform (shear mapping is a specific matrix). The left vectors can be represented as blue: {0,1} x direction and red: {0,1} y direction. In the right picture the blue vector did not change direction, while the red one did, that means the blue vector was transformed by an eigenvector, while the red one was not. Eigenvalues tell how much the eigenvector scaled the original. In this example, the length of the blue vector stays the same, therefore the eigenvalue is 1, double the length means the eigenvalue was 2 etc.



More formally we say that an **eigenvector v** of a matrix C is a non-zero vector that changes by only a scalar factor (the eigenvalue) when that linear transformation (that is, matrix multiplication) is applied to it. Formally, we can find eigenvectors v and eigenvalues λ by solving

$Cv = \lambda v$.

Specifically, eigenvalues (λ) are found by solving
$\det(C-\lambda I) = 0$, where “det” is the determinant.

1.c: Find the two eigenvalues of C (λ1 and λ2)

C: $\begin{pmatrix} 2 & -2 \\ 1 & -1 \end{pmatrix}$

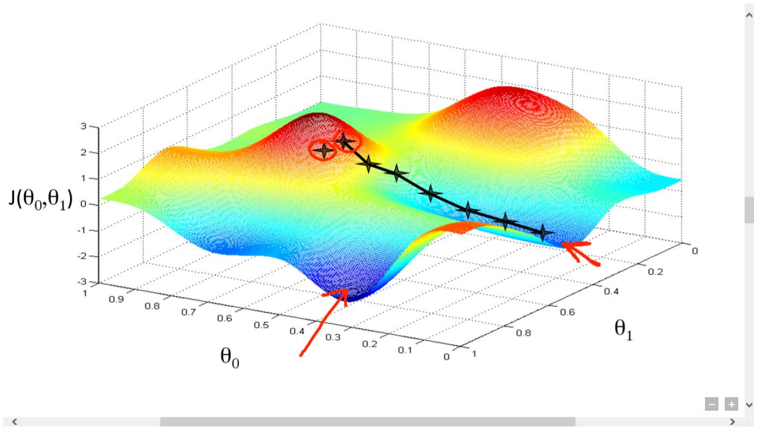
1.d: Find the eigenvectors of C
Hint: Use row reduction.
If you have the eigenvalues, it is easy to get the eigenvectors by solving $(C-\lambda I)*v = 0$

Calculus

Calculus is the mathematical study of continuous change (speed, time, momentum etc). In this section a we calculate a gradient, then use Lagrange multiplier to find the local minimum.

Gradient

The **gradient** is a generalization of derivative in multivariate functions. By definition, gradient is a vector which contains the partial derivatives of a multivariate function. Gradient is usually denoted by the **nabla symbol**(∇). When evaluated at a given point, the gradient gives the direction of largest increase in the value of the function. The extreme values (minima and maxima) of a function can be found among the points for which the gradient is zero. From the figure below the gradient represent the the current “fall” or “climb” per point. Gradients can be used in machine learning algorithms like neural networks.



Example:

$$\text{If } f(x,y) = x\hat{}3 + y\hat{}2$$

Then the gradient of f is (in vector coordinates):

$$\nabla f(x,y) = [\ d(x\hat{}3 + y\hat{}2)/dx,\ d(x\hat{}3 + y\hat{}2)/dy\ ]$$
$$= [\ d(x\hat{}3)/dx + d(y\hat{}2)/dx,\ d(x\hat{}3)/dy + d(y\hat{}2)/dy\ ]$$
$$[\ 3x\hat{}2 + 0,\ 0 + 2y\ ]$$
$$\nabla f(x,y) = [\ 3x\hat{}2,\ \ 2y\ ] \text{ (note the comma, this is a vector)}$$

Given

$$f1(x,y) = x\hat{}2 + 2y\hat{}2 - xy$$

1.e: What is ∇f1:

Minima/maxima of a function

We want to find the minima of our function f (the blue area in figure above, red is maxima)

We know that in the minima, the derivative should be 0 (the top red can not go more up than it is)

We can then set ∇f = 0 (no fall or climb, only possible in minima/maxima/saddle point)

1.f: Find the single point (x*,y*) that minimizes f1(x, y):

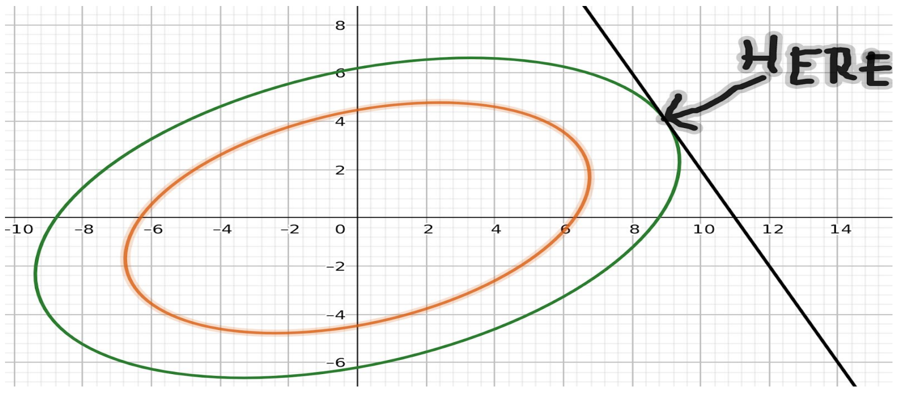
Hint: The minimum is $\nabla f1(x, y) = 0$

Remember that $\nabla f1$ is the answer from the previous question.

Lagrange multipliers

The **method of Lagrange multipliers** (named after Joseph-Louis Lagrange[1]) is a strategy for finding the local maxima and minima of a function subject to equality constraints
It is widely used to solve challenging constrained optimization problems. For those of you who have not had this in a math course and think it is confusing, come ask us or look at the tutorial we added.

In the figure below you see the circle function f, constrained by the line function g.



f: the green, red, or any elliptic function of $x\hat{}2 + 2y\hat{}2 - xy$ *, g: the black line equation, we want to find the point of the arrow, where they meet.*

This question will really ask: Image a super small version of the red/green ellipse in the point (0,0), how big can we make the ellipse before it hits the black line? The answer is the green ellipse. But how do we compute this on a computer?
This function L is called the "Lagrangian", and the new variable λ is referred to as a "Lagrange multiplier"

$$L(x,y,\ \lambda)\ =\ f(x,y)\ -\ \lambda g(x,y)$$

We can find the answer by solving the three partial derivatives:

$$dL(x,y,\lambda)/dx\ =\ d(f(x,y)\ -\ \lambda g(x,y))/dx\ =\ 0$$
$$dL(x,y,\lambda)/dy\ =\ d(f(x,y)\ -\ \lambda g(x,y))/dy\ =\ 0$$
$$dL(x,y,\lambda)/d\lambda\ =\ d(f(x,y)\ -\ \lambda g(x,y))/d\lambda\ =\ 0$$

Good tutorial here:
https://nb.khanacademy.org/math/multivariable-calculus/applications-of-multivariable-derivatives/constrained-optimization/a/lagrange-multipliers-single-constraint

1.h: Find the value of x , y and λ for to maximize:

function $f1(x,y)\ =\ x\hat{}2 + 2y\hat{}2\ - xy$

constrained by g(x,y) = 2x + y - 22

$$L(x,y,\ \lambda)\ =\ x\hat{}2 + 2y\hat{}2\ - xy\ -\ \lambda(2x\ +\ y\ - 22)$$

Step one: set up the three function $dL(x,y,\lambda)/d(x,y,\lambda)\ =\ 0$

Hint: First of the three partial derivatives should be: $dL(x,y,\lambda)/dx$ = 2x - y - λ = 0.

Step 2: Now you can solve for x,y and λ, since you have three functions and three unknowns.

Hint: You can solve by row reduction. The figure above also gives a very strong hint of what x and y should be.

Statistics

Statistics is a branch of mathematics dealing with the collection, organization, analysis, interpretation and presentation of data. At the very heart of machine learning.

Conditional statistics

P(B|A) is read the conditional probability of B given that A has occurred.

1.g: In a normal card deck, what is the probability of:
P(card is a king | card is not an ace)

For a normal dice, the probability of getting any result is ⅙.
The **mean** represents the expected value of a dice
The **variance** describes how much spread there is in the data.
1.h: What is the mean value and variance of a normal dice, also explain why a dice throw have a uniform distribution.

2. Machine learning problems

Recall Mitchell’s definition of a well-defined machine learning problem:

"A computer program is said to learn from experience ***E*** with respect to some class of tasks ***T*** and performance measure ***P*** if its performance at tasks in ***T***, as measured by ***P***, improves with experience ***E***."

In this exercise, we jump into shoes of a machine learning consultant and practice formulating well-defined machine learning problems in real-life scenarios.

Examples

In each of these examples, shortly explain what **T**, **P**, and **E** could be for the specific cases. Note that there is no single correct answer and each problem may be formulated in several reasonable ways.

Task **T** is what the machine learning solution does. It may be useful to think it as a computational problem that has an input and output and specify them. You don't have to think how that program would work.

Performance measure **P** tells how well the program works. The performance measure should reflect your customer's goal and it should be something that can be measured.

Experience **E** specifies the data that is used for learning. Think about what kind of data you would collect.

1. Grocery store problem

The grocery store chain Rema 1000 is considering opening a new store in Marineholmen. They know that popularity of different products varies depending on the location of a store. Now, they want to know what kind of sortiment they should choose for the new store. The managers of Rema 1000 have heard that machine learning can help to solve this kind of problems and asked for your help.
2.a: What would be
the task T:
the performance measure P:
the experience E:

2. Oil drilling problem

Equinor (formerly known as Statoil) produces oil. Each oil platform can use different technologies (for example, drill size and drill density). To maximize the oil production, Equinor would like to maximize the production of each platform. How would you use machine learning to help them?
2.b: What would be
the task T:
the performance measure P:
the experience E:

3. Self-driving car problem

Tesla wants the build reliable self-driving cars that don't crash. Cars use different sensors (cameras, radar, gps) to gather knowledge of their surroundings. Now Tesla is building a steering system to avoid collisions. How would you utilize machine learning in development of such a system?

2.c: What would be
the task T:
the performance measure P:
the experience E:

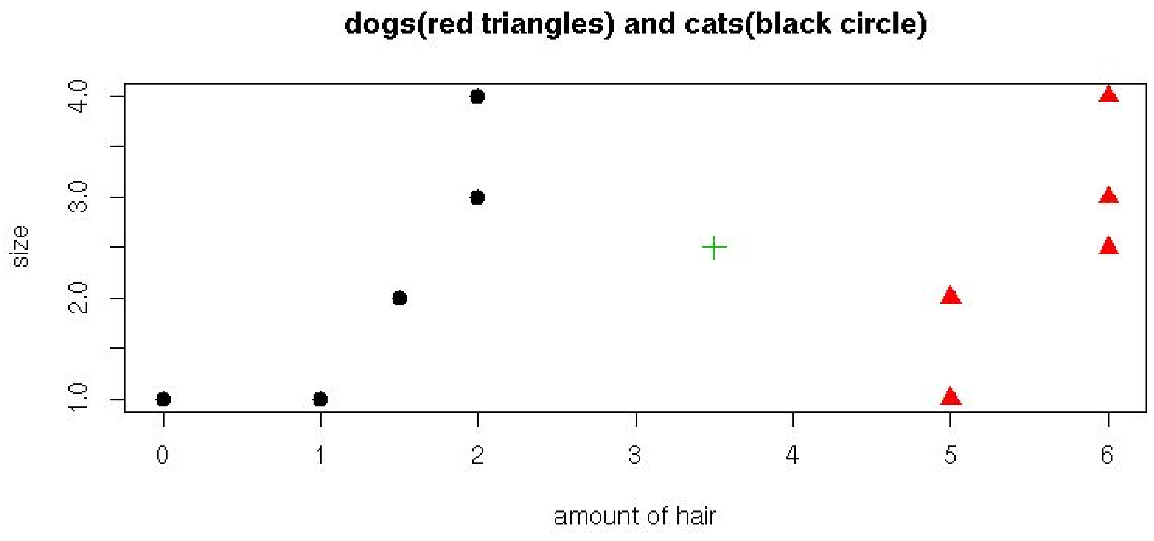
3. K-Nearest-Neighbours

KNN is an classification algorithm. It decides the group of some unknown x, by choosing the group that had the most members in its K closest neighbors.

An example would be to classify an unknown color to its 10 closest neighbors (K is 10), which are 5 red, 3 green and 2 blue. The unknown would then be classified as red.

Cat-dog problem

3.a: Given a the two categories of dogs(red triangles) and cats(black circles), what category does the unknown animal X(green +) belong to if:



K is 3 (are there most cats or dogs in the 3 closest points):
K is 9 (all except 1 point):

KNN using programming

In this task, skip the steps needed if you already have them installed.

If for any reason the instructions do not work for you, ask any of the teaching assistants or group leaders or the other students for help. It can also be noted that you don't need to use python, but this course will use python (version 3) as the main programming language. Installation help for python is in the bottom of this document. Another great programming language for machine learning is the R programming language, but remember we can not help you with the low level details if you don't use Python. Description of problem given for general description, python and R(short description).

KNN problem on IRIS flowers

This problem is similar to the first one, but is too big to calculate by hand. Therefore, we use scikit to help us. In this problem, we want to classify some iris flowers into the three possible types of iris.

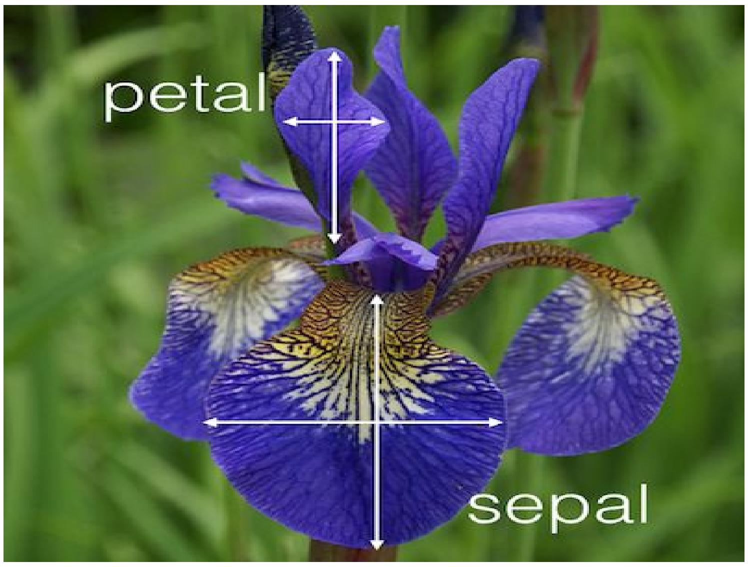
IRIS dataset

Iris Versicolor

Iris Setosa

Iris Virginica

We use the fact that their flowers are of different sizes, so that if we know these parameters, we can try to categorise them. The two parameters we use are Petal length and Sepal length (shown in picture below).



3.b:

General instruction:
Create a uniform weighted KNN algorithm, and use the iris dataset as input (use sepal length (column 1) and petal length (column 3) of iris.data as x, then iris.target is y). Plot the separation boundaries for the classifier, and add your code together with how many flowers that were classified in the three flower groups. Note that not using python, will give you more work to do.
Data download(for other languages than python and R):
https://archive.ics.uci.edu/ml/datasets/iris
NOTE: If you used another programming language than python, include the code

Python Instructions:

1. Copy this script into python (or remake it yourself in another programming language): http://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html#sphx-glr-auto-examples-neighbors-plot-classification-py
2. Change the number of nearest neighbors to 10, use only uniform weights
   On line: for weights in ['uniform', 'distance']: , remove distance.
3. This dataset have several parameters we can use. We want to use petal and sepal length, so switch to those(column dimensions 1 and 3):
   On line: X = iris.data[:, :2] (original uses column dimensions 1 and 2)

4. **Run the script and write down the number of classified flowers in each of the three iris types (points inside each color region)**. The variable y contains the groups, and the default colors are 0- red, 1-green, 2-blue.
   Steps:
   1. Get the plot
   2. You now see a color boundary, some of the flowers are categorized wrongly (a blue dot inside the green region, etc)
   3. From y, see how many flowers are in each group (how many 0, 1 and 2). Use:
   On top of script: *import collections*, in end: *print(collections.Counter(y))*

4. From the total of each group, put the misclassified flowers in their correct group (i.g. if there are originally 20 green flowers, and 1 is classified in the blue region, but two blue is classified as green, green have 20-1+2 = 21 classified flowers in the green region, etc.) Look at the plot to understand what this means.
5. Write down an example of a problem where KNN would not work well as a classifier. (Think about what the x, y axis can be, what data do you have, imbalance in size of the sets, what kind of variable types are the data consisting of).

R instruction:
If you use R, a good KNN function is caret::knn(), install.packages(“caret”)

To get the Iris data in R, you can do:
install.packages(datasets);library(datasets);data(iris)

Help installing Python3 and dependencies

Instead of following this tutorial, if you have nothing preinstalled, a smart way is to just install Anaconda. A kit that includes all we need. Here is a link for windows:
https://www.dummies.com/programming/big-data/data-science/how-to-install-anaconda-on-windows/

1. Install python and PIP

Check if you have python version 3 installed:

For unix (linux and mac):
In the command console (ctrl+alt+t) usually.
python3 --version

For windows:
In command prompt (windows key + x and then find command prompt)
python --version

If the output is “Python is not defined”, it means you need to install python.
Download and install from: https://www.python.org/

PIP
PIP is a recursive acronym for "Pip Installs Packages". It is the most widely used package manager for python.

For linux:
sudo apt-get install python-pip

For mac:
brew install python

For windows:
Download get-pip.py to a folder on your computer. Open a command prompt window and navigate to the folder containing get-pip.py. Then run in command prompt to install:
python get-pip.py

2. Install scikit

Scikit is a popular machine learning package for python.

This kit contains depends on SciPi and numPy.

SciPy (pronounced “Sigh Pie”) is a Python-based ecosystem of open-source software for mathematics, science, and engineering, including among others NumPy.

NumPy (numerical python) is the fundamental package for scientific computing in Python.

For all OS:
Go to the command console
python -m pip install --user numpy scipy
pip install -U scikit-learn

3. Install an IDE

Integrated development environment (**IDE**) is the software application where coding is normally done. It usually includes a source code editor, build automation tools, and a debugger. Choose the one you want from this list, if you already have one like eclipse or python ide, skip this step
https://realpython.com/python-ides-code-editors-guide/#general-editors-and-ides-with-python-support

4. Start Python IDE

If you have never used python before, just go to this page:
https://www.w3schools.com/python/python_syntax.asp

Read about syntax, for loop, while loop, variables and operators.
Just like matlab, python have a simple matrix syntax.
Remember python start on index 0
[a, b], -> a are rows, b are columns
[3:5, :4], -> row 4 to 5 and column 1 to 3
[:, :10], -> all rows, column 1 to 10