# Project 2 on Machine Learning

**Data Analysis and Machine Learning**

Jan 28, 2019

## Paths for project 2

### Defining the data sets to analyze yourself

For project 2, you can propose own data sets that relate to your research interests or just use existing data sets from say

1. Kaggle
2. The University of California at Irvine (UCI) with its machine learning repository
3. The credit card data set from UCI is also interesting and links to a recent scientific article. See however below for possible project example
4. The pulsar classification data set (PulsarStars.csv) is obtained from Kaggle, where it was posted by Pavan Raj. The data file is available in the **DataFiles** folder of this project.

The approach to the analysis of these new data sets should follow to a large extent what you did in project 1. That is:

1. Whether you end up with a regression or a classification problem, you should employ at least two of the methods we have discussed among **linear regression (including Ridge and Lasso)**, **Logistic Regression**, **Neural Networks**, **Support Vector Machines** and **Decision Trees and Random Forests**. If you wish to venture into **convolutional neural networks** or **recurrent neural networks**, or extensions of neural networks, feel free to do so.
2. For project 2, you should feel free to write your own code or use the available functionality of **scikit-learn**, **tensorflow**, etc.
3. The estimates you used and tested in project 1 should also be included, that is the $R2$-score, **MSE**, cross-validation and/or bootstrap if these are relevant.

4. If possible, you should link the data sets with exisiting research and analyses thereof. Scientific articles which have used Machine Learning algorithms to analyze the data are highly welcome. Perhaps you can improve previous analyses and even publish a new article?

5. A critical assessment of the methods with ditto perspectives and recommendations is also something you need to include.

All in all, the report should follow the same pattern with abstract, introduction, methods, code, results, conclusions etc as the two previous projects.

We propose also an alternative to the above. This is a project on using machine learning methods (neural networks mainly) to the solution of ordinary differential equations and partial differential equations.

This is a field with a large interest recently, spanning from studies of turbulence in meteorology to the solution of quantum mechanical systems.

## Studying the credit card data set as possible project

We include this data set as an example on how one could study new data sets with the algorithms we have discussed during the lectures, using either your own codes or the functionality of **scikit-learn**, **tensorflow** or other Python packages.

The data set is presented at the site of UCI. It is particularly interesting since it is also analyzed using ML methods in a recent scientific article.

The authors apply several ML methods, from nearest neighbors via logistic regression to neural networks and Bayesian analysis (not covered much in our course). Here follows a set up on how to analyze these data.

**Part a).** The first part deals with structuring and reading the data, much along the same lines as done in projects 1 and 2.

**Part b).** Perform a logistic regression analysis and see if you can reproduce the results of figure 3 of the above article.

**Part c).** The next step is to use neural networks and the functionality provided by tensorflow/keras or scikit-learn's MLP method (or you could write your own code). Compare and discuss again your results with those from the above article.

**Part d).** The above article does not study random forests or support vector machine algorithms. Try to apply one of these methods or both to the credit card data and see if these methods provide a better description of the data. Can you outperform the authors of the article?

**Part e).** Finally, here you should present a critical assessment of the methods you have studied and link your results with the existing literature.

## The Pulsar data

The pulsar classification data set (PulsarStars.csv) is obtained from Kaggle, where it was posted by Pavan Raj. It offers an interesting possible classification problem. In the field of radio astronomy, pulsars are among the most studied phenomena in nature. But despite astronomers' long history with pulsars, little is actually known with certainty. However, much of the uncertainty likely boils down to the difficulty of confirming pulsar observations. While pulsars radiate unmistakable radio signals, they are often lost in the sheer number of radio signals observed by radio telescopes every day. Furthermore, due to the uniqueness of pulsar radio signals, classifying pulsars in large data sets of radio observations have historically been very difficult as human supervision has been a necessity. However, recent advancements in machine learning and data mining has made this task much simpler by introducing incredibly fast, in comparison to humans that is, classification methods.

You could repeat many of the steps discussed or the credit card data problem. The article of Bathes et. al. can serve as a reference for your discussions.

## Solving partial differential equations with neural networks

For this variant of project 2, we will assume that you have some background in the solution of partial differential equations using finite difference schemes. We will study the solution of the diffusion equation in one dimension using a standard explicit scheme and neural networks to solve the same equations.

For the explicit scheme, you can study for example chapter 10 of the lecture notes in Computational Physics or alternative sources. For the solution of ordinary and partial differential equations using neural networks, the lectures by Kristine Baluka Hein at this course are highly recommended.

**Part a), setting up the problem.** The physical problem can be that of the temperature gradient in a rod of length $L = 1$ at $x = 0$ and $x = 1$. We are looking at a one-dimensional problem

$$\frac{\partial^2 u(x,t)}{\partial x^2} = \frac{\partial u(x,t)}{\partial t}, t > 0, x \in [0, L]$$

or

$$u_{xx} = u_t,$$

with initial conditions, i.e., the conditions at $t = 0$,

$$u(x, 0) = \sin(\pi x) \qquad 0 < x < L,$$

with $L = 1$ the length of the $x$-region of interest. The boundary conditions are

$$u(0, t) = 0 \qquad t \geq 0,$$

and

$$u(L,t) = 0 \qquad t \geq 0.$$

The function $u(x,t)$ can be the temperature gradient of a rod. As time increases, the velocity approaches a linear variation with $x$.

We will limit ourselves to the so-called explicit forward Euler algorithm with discretized versions of time given by a forward formula and a centered difference in space resulting in

$$u_t \approx \frac{u(x,t+\Delta t) - u(x,t)}{\Delta t} = \frac{u(x_i,t_j+\Delta t) - u(x_i,t_j)}{\Delta t}$$

and

$$u_{xx} \approx \frac{u(x+\Delta x,t) - 2u(x,t) + u(x-\Delta x,t)}{\Delta x^2},$$

or

$$u_{xx} \approx \frac{u(x_i+\Delta x,t_j) - 2u(x_i,t_j) + u(x_i-\Delta x,t_j)}{\Delta x^2}.$$

Write down the algorithm and the equations you need to implement. Find also the analytical solution to the problem.

**Part b).** Implement the explicit scheme algorithm and perform tests of the solution for $\Delta x = 1/10$, $\Delta x = 1/100$ using $\Delta t$ as dictated by the stability limit of the explicit scheme. The stability criterion for the explicit scheme requires that $\Delta t/\Delta x^2 \leq 1/2$.

Study the solutions at two time points $t_1$ and $t_2$ where $u(x,t_1)$ is smooth but still significantly curved and $u(x,t_2)$ is almost linear, close to the stationary state.

**Part c) Neural networks.** Study now the lecture notes on solving ODEs and PDEs with neural network and use either your own code or the functionality of tensorflow/keras to solve the same equation as in part b). Discuss your results and compare them with the standard explicit scheme. Include also the analytical solution and compare with that.

**Part d).** Finally, present a critical assessment of the methods you have studied and discuss the potential for the solving differential equations with machine learning methods.

## Introduction to numerical projects

Here follows a brief recipe and recommendation on how to write a report for each project.

- Give a short description of the nature of the problem and the eventual numerical methods you have used.

- Describe the algorithm you have used and/or developed. Here you may find it convenient to use pseudocoding. In many cases you can describe the algorithm in the program itself.

- Include the source code of your program. Comment your program properly.

- If possible, try to find analytic solutions, or known limits in order to test your program when developing the code.

- Include your results either in figure form or in a table. Remember to label your results. All tables and figures should have relevant captions and labels on the axes.

- Try to evaluate the reliabilty and numerical stability/precision of your results. If possible, include a qualitative and/or quantitative discussion of the numerical stability, eventual loss of precision etc.

- Try to give an interpretation of you results in your answers to the problems.

- Critique: if possible include your comments and reflections about the exercise, whether you felt you learnt something, ideas for improvements and other thoughts you've made when solving the exercise. We wish to keep this course at the interactive level and your comments can help us improve it.

- Try to establish a practice where you log your work at the computerlab. You may find such a logbook very handy at later stages in your work, especially when you don't properly remember what a previous test version of your program did. Here you could also record the time spent on solving the exercise, various algorithms you may have tested or other topics which you feel worthy of mentioning.

## Format for electronic delivery of report and programs

The preferred format for the report is a PDF file. You can also use DOC or postscript formats or as an ipython notebook file. As programming language we prefer that you choose between C/C++, Fortran2008 or Python. The following prescription should be followed when preparing the report:

Finally, we encourage you to collaborate. Optimal working groups consist of 2-3 students. You can then hand in a common report.

## Software and needed installations

If you have Python installed (we recommend Python3) and you feel pretty familiar with installing different packages, we recommend that you install the following Python packages via **pip** as

1. pip install numpy scipy matplotlib ipython scikit-learn tensorflow sympy pandas pillow

For Python3, replace **pip** with **pip3**.

See below for a discussion of **tensorflow** and **scikit-learn**.

For OSX users we recommend also, after having installed Xcode, to install **brew**. Brew allows for a seamless installation of additional software via for example

1. brew install python3

For Linux users, with its variety of distributions like for example the widely popular Ubuntu distribution you can use **pip** as well and simply install Python as

1. sudo apt-get install python3 (or python for python2.7)

etc etc.

If you don't want to install various Python packages with their dependencies separately, we recommend two widely used distrubutions which set up all relevant dependencies for Python, namely

1. Anaconda Anaconda is an open source distribution of the Python and R programming languages for large-scale data processing, predictive analytics, and scientific computing, that aims to simplify package management and deployment. Package versions are managed by the package management system **conda**

2. Enthought canopy is a Python distribution for scientific and analytic computing distribution and analysis environment, available for free and under a commercial license.

Popular software packages written in Python for ML are

- Scikit-learn,

- Tensorflow,

- PyTorch and

- Keras.

These are all freely available at their respective GitHub sites. They encompass communities of developers in the thousands or more. And the number of code developers and contributors keeps increasing.