Case 1. Heart Disease Classification

Neural Networks for Machine Learning Applications, Spring 2022

Type

Team work, 15-20 hours, 15 p

Aims

The aims of this assignment are to learn to

- use data from external sources
    - Pandas read_csv function or Cloud services data import functions
- use neural networks to make an expert system to support in diagnostic decision making
- try and test model architectures
    - number of layers, number of units, activation functions
    - solver optimizers and training settings
- use visualization tools to make graphical presentations of the training results
- use metrics to calculate and validate training results
- learn to document the results clearly and in an easily readable format

Task

Your task is to read and preprocess the data Heart Disease Health Indicators Dataset (Kaggle) and create and train a dense neural network to predict to classify the presence of heart disease. **NOTE: This is a binary classification problem, e.g. the aim is to predict the presence of disease or not.**

The dataset page (see the link below) contains information about the features and the number of instances in the data files. You can use (for example Panda's) summary statistics and graphical presentation tools to study the dataset. However, **we are not interested** in *extensive* statistical and descriptive analysis, **but** *short overview of the data and pre-processing methods*.

In the final report you should show:

- how to preprocess the data (normalize and encode categories)
- how to make classification on imbalanced data
- how to split a dataset into train and test sets
- how to use the sequential model
- how to train and evaluate the models
- how to analyse results using numerical metrics and graphical plots

Try and test different settings and models and compare their results. Try to find the simplest, fastest and smallest possible settings and model that finds most of the disease cases without misclassifying too many healthy cases (aiming to high sensitivity without losing specificity).

Use Jupyter Notebook to solve the problem. Use the Markdown cells to document your report. The recommended documentation structure is the following:

- Main heading (Titles) including:
  - The assignment name as the title/heading
  - Your team name and your team members names
  - The last date, when you edited the document
  - Organization name (Metropolia University of Applied Sciences)
- 1. Introduction/Background
  - Why this document is created, and what were the main objectives
- 2. Setup
  - Setup of the necessary libraries
- 3. Dataset
  - Introduction to the data
  - Code for reading the dataset and showing the main characteristics
- 3. Preprocessing
  - Handling the missing values
  - Conversions of textual and categorical data into numerical values (if needed)
  - Splitting the data into train and test sets
  - Separate the features (=input) and labels (=output)
  - Feature normalization
- 4. Modeling
  - Short description of the model
  - Selected loss, optimizer and metrics settings
  - The summary of the selected model architecture
- 5. Training
  - Short description of the training process
  - The code for training and the total time spend on it
    - Note! **Use verbose = 0 for clarity. Don't display the results epoch-by-epoch in the final version.**
    - See: how to measure elapsed time in Python
- 6. Evaluation and performance
  - The training and validation loss plot
  - The training and validation accuracy plot
  - Explanations for the loss and accuracy plot (e.g. how do you interpret the results)
  - Final performance of the model with test set
    - Note: **Use the test set only after you have decided what is your best model, not before!**
- 7. Discussion and conclusions
  - What settings and models were tested before the best model was found
    - What where the results of these experiments
  - Summary of
    - What was your best model and its settings
    - What was the final achieved performance

        - Note: **Remember to evaluate the final metrics using the test set.**
    - What are your main observations and learning points
    - Discussion how the model could be improved in future

Return

Save your final Notebook to OMA.

Evaluation

The following categories are used for evaluation:

- Organization
    - The code proceeds logically and the code cells are in proper order
    - The document has a clear structure
- Clarity
    - The document is clear, polished, and easy to understand
    - The code follows good coding practices and is adequately commented
    - The document parts supports the code
- Contents
    - The introduction, dataset and data preprocessing are well explained
    - The model is validated and tested
    - The results are reasonable
    - The conclusions are clearly stated and in line with the results

Max. 15 points. Late submission reduces the maximum achievable points.

Tips and links

- Reading data from a csv file
- Panda's cookbook
- Panda's tutorials
- Guide to the sequential model
- Preprocessing data (scikit-learn)
- Keras Getting Started