Lab 03: FNN

Fabian Haas, Markus Reichl, Florian Weingartshofer

Loading and Preprocessing the Dataset

The dataset that was used, turned out to be very different in structure, than the datasets used in the exercises before. First the format had to be understood since no member of our team had knowledge with ECG data. Loading the dataset was solved by loading from both directories `normal` and `abnormal`. Each entry ending in a `0` or `1` is loaded and turned into an entry in a pandas DataFrame. Depending on in which directory the entry resided in it was labeled with a 0 or 1, with 0 being the normal label and 1 being the abnormal label. The data was concatenated if it had more than 75 entries and padded with zeroes if it was too short. We also included an option to use the average of the values present in the dataset, but decided to stay with the zero padding approach.

The result of the process was a pandas DataFrame with 76 columns, 75 of them being the ECG measurements and one being the label for normal or abnormal. This DataFrame is then split into training and test data and into the features `X` and the target variable `y` The training dataset is then oversampled using `SMOTE` from the `imblearn`-library.

Lasty, the features `X` are resampled using the sklearn standardscaler.

This process results in our loaded and preprocessed training and test data.

Building the FNN

Keras with the tensorflow backend is used to build the FNN. We decided to start with a small test neural network consisting of 3 densely connected neural network layers. This small network already shows good results with an accuracy of 80%, precision of 77.5% and a recall of 91.18%.

The keras hyperparameter tuner is used to find an even better fnn, we decided to alternate between density layers and dropout layers. The tuner will generate models with 1 to 7 density dropout layer combination and one input density layer. Each hidden density layer can have between 32 and 1024 units with a step size of 32. The dropout layers can have a rate between `.0` and `.5` with a step size of `.1`. Last an output layer is added. The resulting models are trained in 150 epochs. The best model has an accuracy of 93.33%, a precision of 94.12% and a recall of 94.12. Its hyperparameters are three layers, with a learning rate of `0.01`.