Roger

Simple models generator

https://github.com/MartinDepardieu

What is Roger

- **Roger is a simple model generator**

- It takes in a table in a database, in which a result y depends on parameters x1, x2…
  - I used the **European Soccer Database** in which a player overall rating is given along his speed, acceleration, etc
  - https://www.kaggle.com/hugomathien/soccer

- It fits a polynomial to each y, xi combination
- It randomly weights each polynomial and each parameter to create y = f(x1, x2, x3…) functions
- It randomly samples data from the database and tests the functions against the real results
- The function with the lowest root mean square error is saved
- This model is later used to predict the result from given data

Quick run

- **To run Roger**
  - 1 – get the European Soccer Database at: https://www.kaggle.com/hugomathien/soccer
  - 2 – place database.sqlite with the other files
  - 3 – Run RunRogerLearning.py
    - Creates models and saves them in text files
  - 4 – Run RunRogerPredicting.py
    - By defaults it gets Roger to pick random rows in the database and compares the result it predicted to the actual value

- To change model parameters
  - RogerEar.py contains all the parameters the models are built from
  - Re-run RunRogerLearning after changing values to generate new models

- To manually input values for predictions
  - Choose a testModelUserValues function in RunRogerPredicting.py

Organisation for learning

- database.sqlite
- RunRogerLearning.py
  - Runs the learning routine
- RogerEar.py
  - Global parameters
- RogerControlLayer.py
  - Assembles the elements for the learning routine
- RogerEye
  - SQL connections
- RogerNeuronLayerOne
  - Polynomial fit of x, y
- RogerNeuronLayerTwo
  - Weighting of fits
- RogerNeuronLayerThree
  - Weighting of parameters
- RogerNeuronLayerFour
  - Models testing, best models selection
- Models text files
  - One text file is saved for each model, overwritten if the learning routine finds a best one

Learning step : general architecture

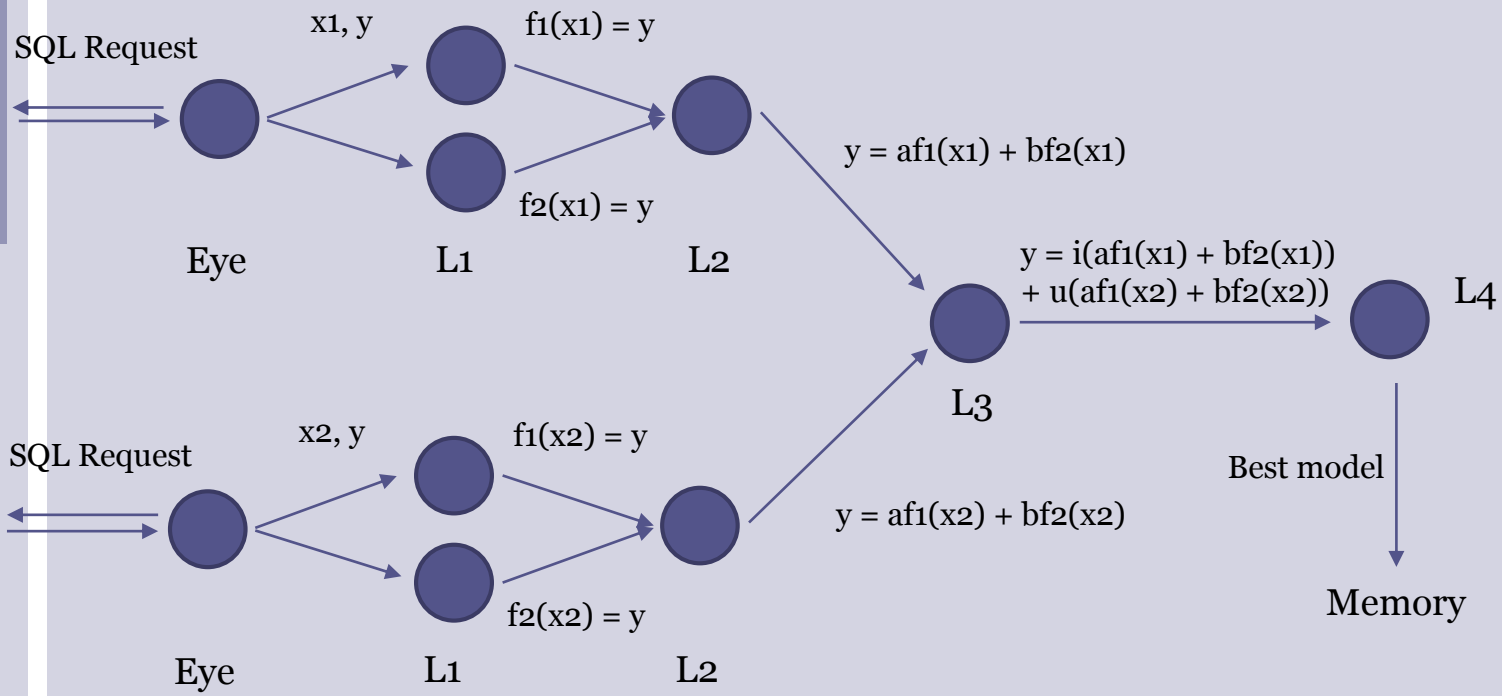
Database

table

y,
x1,
x2,
x3
x4,
…

Global parameters

Control layer



Organisation for learning

- Database connexions
  - **European Soccer Database**
    - https://www.kaggle.com/hugomathien/soccer
    - PlayerAttributes tables
  - ControlLayer takes y to optimise for and parameters from RogerEar
  - Creates SQL requests and send them to RogerEyes
  - RogerEye takes database, table, name x, name y and outputs an x, y table

- Layer 1
  - Each N1 takes in x, y and outputs f(x) = y
  - One N1 per parameter and per fit type
  - Simple polynomial fits
  - Number N1 = number of parameters * number of fits

Organisation for learning

- Layer 2
  - N2 takes in fits, outputs fits and weights
  - For each parameter, weights the fits from L1
  - One L2 cluster per parameter
  - Arbitrary number of N2 per cluster
    - Generation of an arbitrary number of weights combinations

- Layer 3
  - N3 takes in Layer 2
  - Weights parameters
  - Randomly chose one N2 per parameter/cluster and uses its weighted fits to model this parameter
  - Outputs a complete model y = f(x1, x2, x3…)

Organisation for learning

- Layer 4
  - N4 takes in Layer 3
  - Connects to the database and randomly builds a sample of data
  - Feeds this data to each N3
  - Compares the outputs y' to reality y
  - Selects the N3 with the model closest to reality

- Model saved in text file
  - If no model already present
  - If better than existing model

Organisation for predicting

- The prediction part is just loading data either from random database draws or from inputed user values
- It loads models from text files and applies them to the data
- Outputs the predicted result
  - Compares it to the real result if available in database
- Files
  - RunRogerPredictiong.py
    - Runs prediction routines
  - RogerPrediction
    - Functions to load and test the models
  - Models text files
    - Text files containing the saved models