Inductive Programming

Amanda Lee and Alec Radford

The Problem

- Explore inductive programming
  - Machine Learning
  - Given examples and background knowledge
  - Generate a program or hypothesis satisfying all initial conditions

- Focus on learning functions
  - Given a set of data
  - Determine a functional model of the data

Linear Regression

- Exact solution for m known via linalg - OLS - old news
  - $[a] = [X^T X]^{-1} X^T y$
  - For input learns a multiplicative “weight”.
  - b can be found via adding a “dummy” variable

$$X = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix} \quad Y = \begin{bmatrix} 3 \\ 5 \\ 7 \\ 9 \\ 11 \end{bmatrix} \quad \Rightarrow \quad X = \begin{bmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \\ 4 & 1 \\ 5 & 1 \end{bmatrix} \quad Y = \begin{bmatrix} 3 \\ 5 \\ 7 \\ 9 \\ 11 \end{bmatrix}$$

Generalizing

- $[a,b] = [X^T X]^{-1} X^T y$
  - The ones allow us to learn a constant offset in a multiplicative framework.
  - Equivalent to
    - $f(x) = a*x + b*1$
- Modify input x via some function and OLS will learn the multiplicative “contribution” of that function to the overall function.

Insight

- We can now learn parameters using this tool.
  - But how can we learn the functions themselves?

- A weight of 0 corresponds to a term not being present!
- Given inputs x and outputs y
  - Expand x into a vector of functional transformations of x

$f(x) = (x > 0) + (x < 0)$

$x \rightarrow [\ x\ ,\ x^2\ ,\ x^3\ , x > 0, x < 0, \exp(x)]$

$\text{model} \rightarrow [\ 0\ ,\ 0\ ,\ 0\ ,\ 1\ ,\ 1\ ,\ 0\ ]$

A New Approach

- Stochastic Superoptimization
  - Took assembly for basic programs compiled via gcc and tried to make it faster via MCMC.
  - The idea is take the current program and evaluate proposed new programs which are slight modifications of the current program.
  - Then probabilistically decide whether to make the proposal program the current program based on relative improvement (or lack thereof).

Learning Python Functions

- Internal representation of a function:
  - {inputs:[a,b...],vars:[a,b...],vinits:[a,b...],code:[[line 1],[line 2],[...]]
    - Code
      - [a,+,b,c]
      - [d,sqrt,e]
      - [NOP]
- Render internal representation to a string
- Exec it
- Initialize variables at beginning of function
- Followed by n lines
- Return all variables

Learning Python Functions

- All variables are then passed to an evaluator
  - Evaluator checks mean error between each variable and desired output
  - The “goodness” of a program is that of the current best variable

- MCMC didn’t work well (because of evaluator)
  - Mean error is a poor metric (want functional distance)

Quadratic Function Output

```
def f(a,b,c,d):
    e,f,g,h,i,j,k,l,m,n,o,p,q = d,b,a,d,d,b,a,c,b,d,c,c,c
    o = g * p
    p = np.square(k)

    k = np.square(f)




    h = o + i
    l = p * j
    q = l + h

    return q
```

Distance Function Output

```
def f(a,b,c,d):
    e,f,g,h,i,j,k,l,m,n,o,p,q = d,b,a,d,d,b,a,c,b,d,c,c,c
    o = e - b
    o = np.square(o)
    j = e * m
    p = a - q
    f = np.square(b)

    j = np.square(p)
    e = o - o
    q = l - j
    g = j + o

    q = np.sqrt(g)
    h = np.square(c)
    return q
```

Euclidean Distance

```
def f(a,b,c,d):
    o = d - b
    p = a - c
    o = np.square(o)
    j = np.square(p)
    q = np.sqrt(j + o)
    return q
```

- Odds of randomly generating from combinations of variables and operators is at best 1/1,400,000,000
  - If you were to generate random strings, it would be on the order of $1/10^{131}$
- Took 162,000 evaluations to generate.

Functional Distance

- The “edit” distance between two functions.
  - $f(X)=X$
  - $f'(X)=X\cdot(-1)$
    - Functional Distance = 1
  - $f(X)= \exp(X)$
  - $f'(X)= \log(X+10)\cdot 20$
    - Functional Distance = 4

Functional Distance

- Given input *X, Y*, and function *f'* what is the distance between *f'* and *f(X)=Y* ?
  - Not aware of any solutions that don't involve learning *f* which is exactly the problem we're trying to solve...

- May be possible to approximate with an ML model trained on known examples and relevant features
  - What features are relevant?

QUESTIONS?