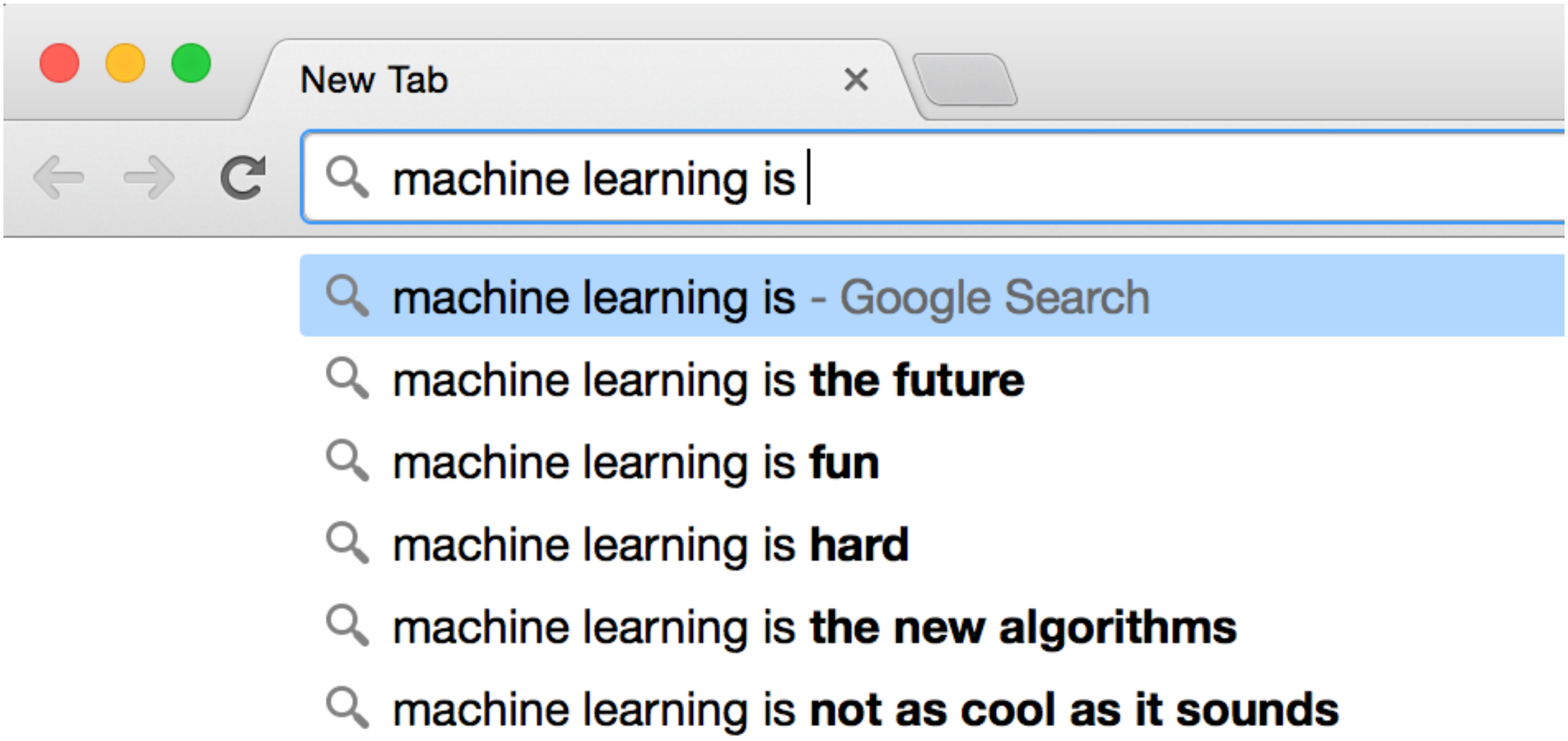
# Data science the functional way

**Evelina Gabasova**

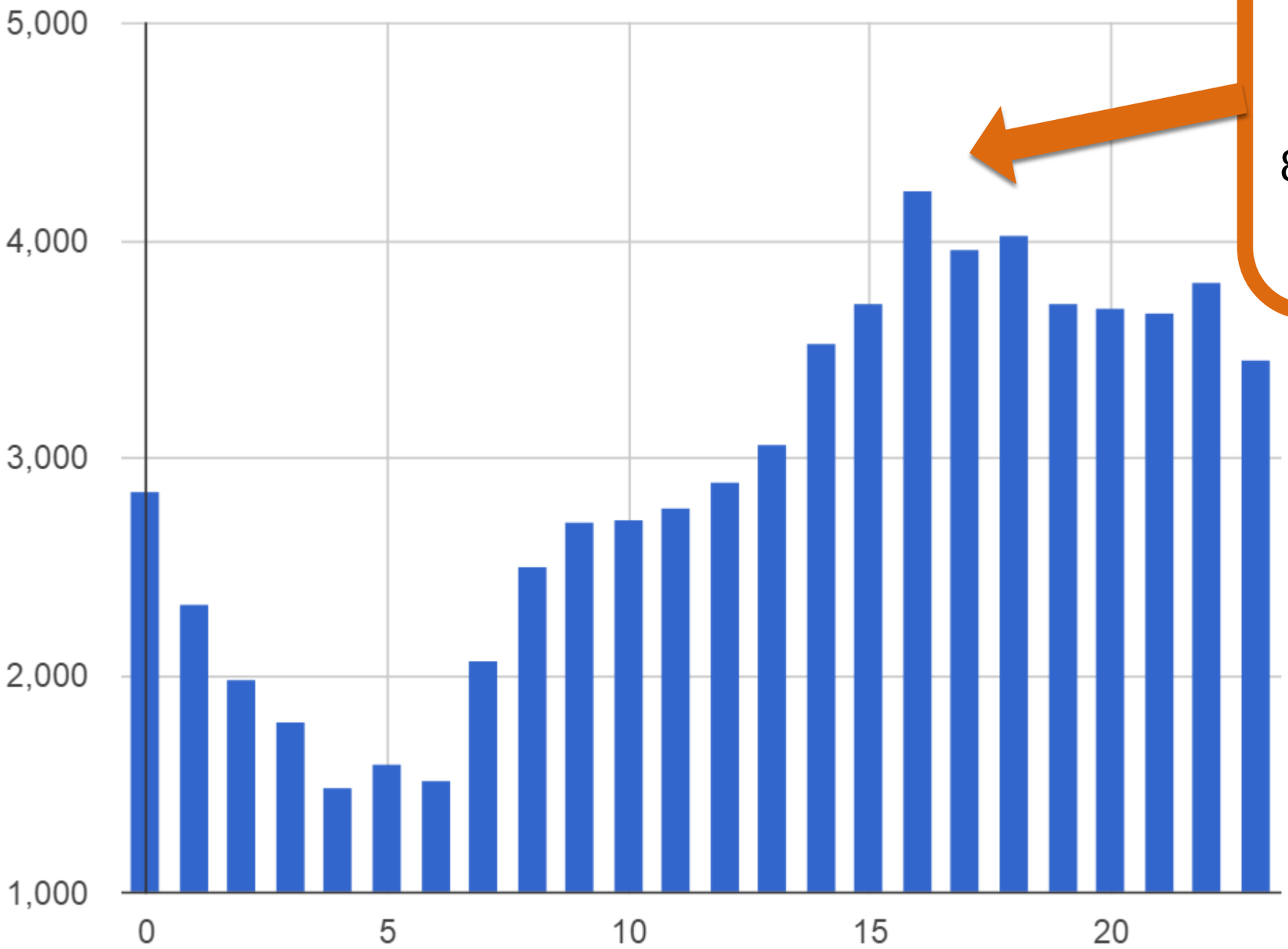
# Machine learning



# With F# about F#

# What is the community's dynamics?

# When do people tweet?




16:00 in London

11:00 in New York

8:00 in San Francisco

0:00 in Tokyo

# Programming in data science

## Scripting languages

fast prototyping, easy to use

R, Python, Matlab

# Why F#?

```
    Lp = lapply(logL, exp)
for(m in 1:M){
    for(i in 1:N) L[[m]][i,] = rmultinom(1,1,Lp[[m]][i,]) #Generate L from Lp
    if(w>1) L[[m]] = AlignClusters(C,L[[m]], type = 'mat') #Helps to align indices
    n[m,] = colSums(L[[m]])
    for(k in 1:K){ ###Update cluster parameters based on normal-gamma distribution
        if(d[m]==1&n[m,k]>1){
            S[[m]][,k] = sd(X[[m]][,L[[m]][,k]==1])^2
            PostMean = sum(X[[m]][,L[[m]][,k]==1])/(n[m,k]+1)
            B[[m]][,k] = b0[[m]]+0.5*(n[m,k]*S[[m]][,k]+n[m,k]*(mean(X[[m]][,L[[m]][,k]==1])-mu0
        if(d[m]>1&n[m,k]>1){
            PostMean = (mu0[[m]]+rowSums(X[[m]][,L[[m]][,k]==1]))/(n[m,k]+1)
            S[[m]][,k] = apply(X[[m]][,L[[m]][,k]==1],MARGIN=1,FUN='sd')^2
        B[[m]][,k] = b0[[m]]+0.5*(n[m,k]*S[[m]][,k]+n[m,k]*(rowMeans(X[[m]][,L[[m]][,k]==1])-mu0
        if(n[m,k]==1){
            PostMean = (mu0[[m]]+X[[m]][,L[[m]][,k]==1])/2
        B[[m]][,k] = b0[[m]]+0.5*(X[[m]][,L[[m]][,k]==1]-mu0[[m]])^2/2)
        if(n[m,k]==0){
            PostMean = mu0[[m]]
            B[[m]][,k] = b0[[m]])
        Lambda = 1+n[m,k]
        A[[m]][,k] = a0[[m]]+n[m,k]/2
        Tau[[m]][,k] = rgamma(d[m],shape=A[[m]][,k],rate=B[[m]][,k])
        mu[[m]][,k] = rnorm(d[m],PostMean,sqrt(1/(Tau[[m]][,k]*Lambda)))
        Sigma[[m]][,k] = sqrt(1/Tau[[m]][,k]))
}
```

# Why F#?

Vector? Matrix?

List? Array?

Data frame?

# Why F#?

```
25          let newas =
26              data.Contexts
27              |> Array.map (fun context ->
28                  let pis = state.ContextWeights.[context]
29                  let rawPriorPis = Array.create pis.Length 1.0
30                  let value = sampleDirichletConcentration (hyperprior.AlphasPrior.[context])
31                  (
32              |> to
33          { state w                                                    as = newas}}
34
35      | FixedValue(
36          // do not
37          state
38
39  // Context-specif
40  // ***************
41
42  // Sample from conditional Dirichlet distribution with random walk Metropolis-Hastings
43  let sampleDirichlet_MetropolisHastings (currentValues: float[])
44      priorConcentration loglikFunction =
45      // 1. Add random walk proposal to current values
46      let randomWalkProposal = Normal(0.0, 1.0, rnd)
47      let proposal_unnorm =
48          currentValues
49          |> Array.map (fun x -> x + 0.1 * randomWalkProposal.Sample())
50
```

```
val sampleDirichletConcentration :
    gammaDist  : Gamma ->
    rawPriorPis: seq<float> ->
    pis        : seq<float> ->
    sampleType : SampleConcentrationParams
                 -> float

Summary
Adaptive rejection sampling (derivative-free)
```

# Answering real questions:

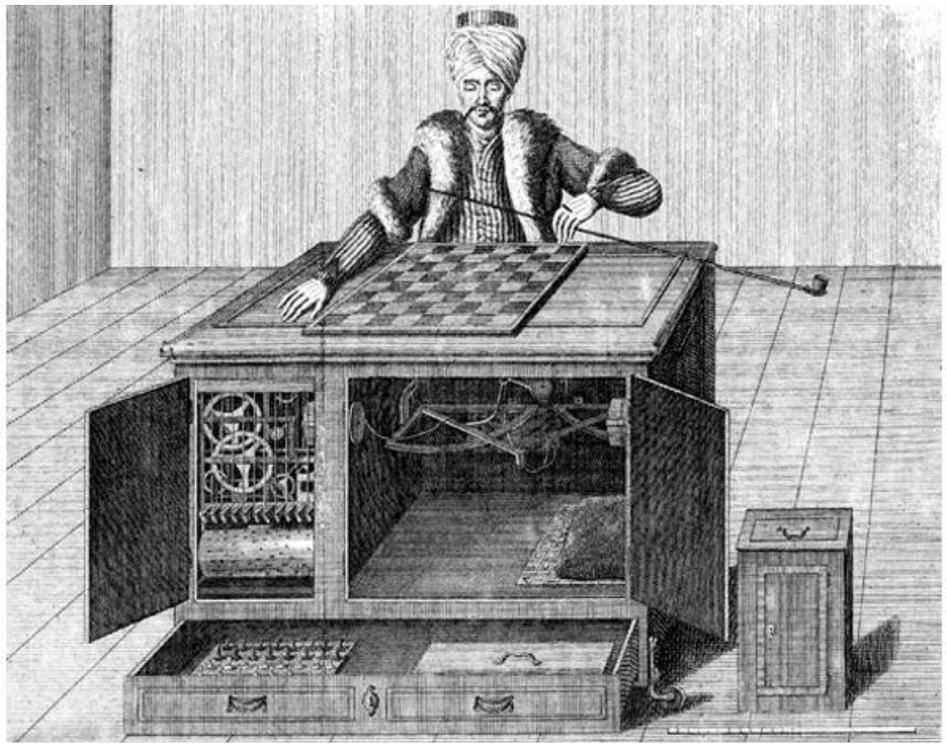
# Are people happy with F#?

# Sentiment analysis

## Stanford NLP library

How to find sentiment from text?

Humans are good at this but computers are not!



# Supervised learning

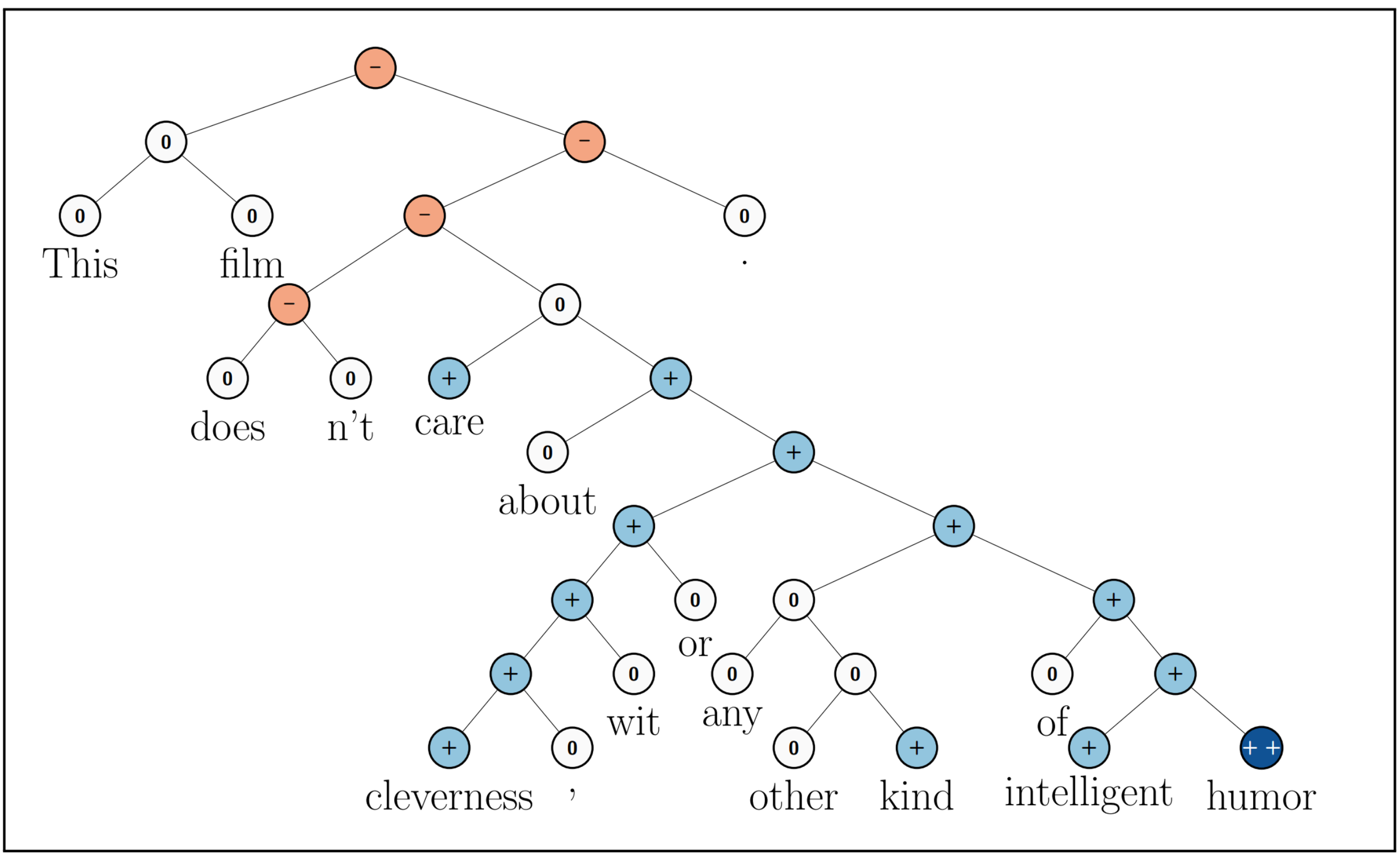
**TEXT**

**LABEL**

# Supervised learning

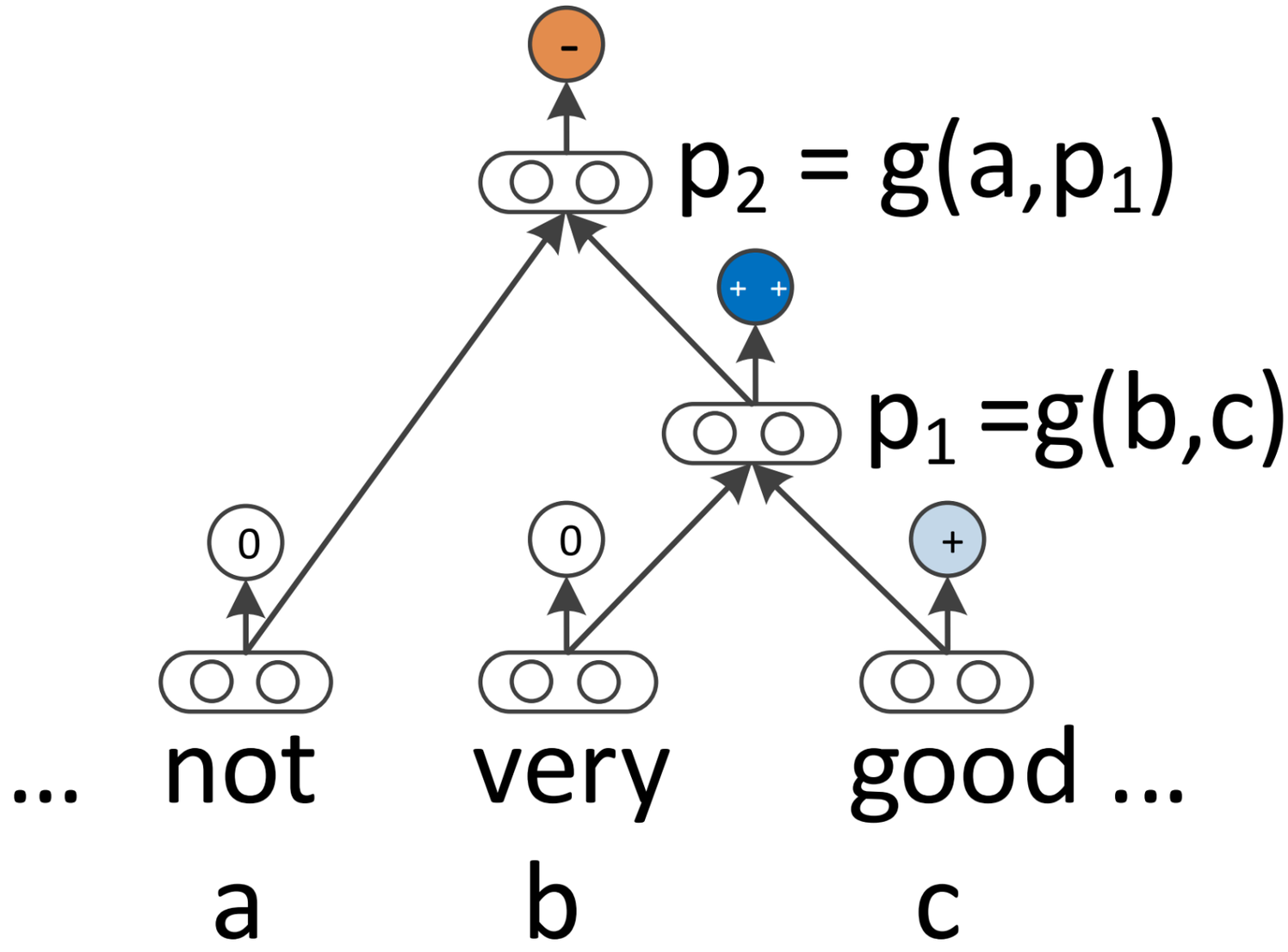
The movie was funny.

The movie was trying to be funny.




Richard Socher, Alex Perelygin, Jean Wu,Jason Chuang, Christopher Manning, Andrew Ng and Christopher Potts: Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank

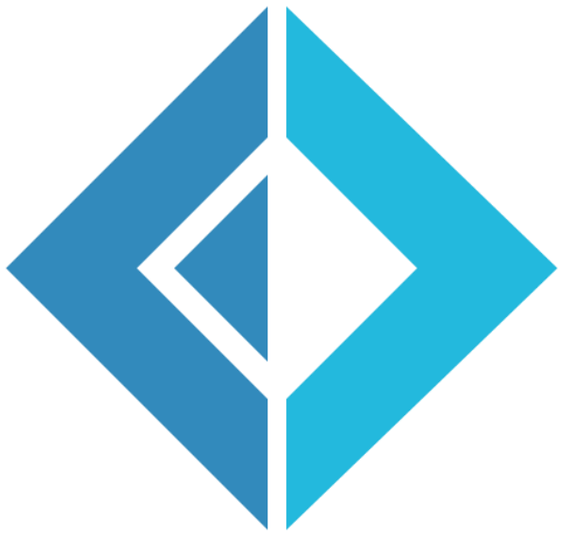
# Deep recurrent neural networks



# Turning analysis into product

SUAVE

# #QConRio



# Why F#?

- Interactive exploration of data & incorporation into larger applications
- Static typing helps
- Type providers for data access
- RProvider allows calling R functions

# Thank you!

@evelgab

evelinag.com

The F# Software Foundation

@fsharporg

fsharp.org