Statistics, Big Data ...

... and small data

Pierre Lebrun, Arlenda SA / Pharmalex
pierre.lebrun@arlen da.com

QPRC2017, UCONN, Storrs
I am not a “big data analyst”, merely a statistician working as a consultant for companies with – sometimes – large datasets
   – More often, very small datasets

My “big data” problems are more a collection of an awful lot of small data problems

As a consultant, the customer big data hardware
   – is central to define a working solution
   – is often fixed and sometimes not adapted
   – was probably the best when they installed it (10 years ago)

As a consultant, the customer software...
   – is often fixed (protected environment... e.g. inability to install an R package...)
Why Bayesian statistics?

- Focus on prediction instead of model parameters
  (not saying that parameters aren’t valuable!)
- Integrate parameter uncertainty and measurement error into the predictive distribution
  (works for all types of models → unified framework)
- Don’t stop to binary answers (go/no go or pass/fail)
- Allow combination of knowledge through the prior distribution, in a natural scheme of updating prior knowledge using Bayes theorem
- **Probability** is a coherent measure of plausibility of an event occurring, given the model hypothesis and data, instead of the frequency of the event

All points above are independent of the size of the dataset
Bayesian statistics

Simulations
where the “new observations” are drawn from distribution “centered” on estimated location and dispersion parameters (treated wrongly as “true values”).

Posterior Predictions
First, by drawing a mean and a variance from the posteriors and, second, drawing an observation from resulting distribution

\[ p(\tilde{y} \mid y) = \int_\theta p(\tilde{y} \mid \theta) \ p(\theta \mid y) \ d\theta \]
Case 1: Time series analysis

- Identify outlier on time series made of count data
  - Compliance: authorities said to the banks “if you have the ability to detect weird patterns in your customer data and raise alert, please do it”
  - One pattern that can be found easily is when a number of aggregated transactions between two entities is not “Normal”
  - Then a customer can identify a root cause or a non compliance issue
    - Strong link with statistical process control (SPC)
Time series analysis

- Identify outlier on time series made of count data
  - Poisson or negative binomial regression
  - Derive a prediction interval for the next number of aggregated transactions with large coverage (e.g. 99%)
  - If a point is outside, it means it does not belong to the same population (it would occur only in 1-99% of the case if the time series was behaving normally)
Some example of R code (Normal approximation, no AR structure)

```r
library(MASS)
mod <- glm.nb(formula = y ~ month, data = datas) # Log-link is implicit
pred <- predict(mod, data.frame(month = 1:(nrow(datas) + 1)), type = "link", se.fit = TRUE)
X <- model.matrix(mod)
xprimex_1 <- solve(t(X) %*% X)
Xpred <- rbind(X, t(c(1, 24)));
S = sqrt(sum(mod$residuals^2)/(ndata - 2))
Root = sqrt(1 + diag(Xpred %*% xprimex_1 %*% t(Xpred)));
meanpred = exp(pred$fit)
PIpredLL = exp(pred$fit - qt(0.975, df = ndata - 2) * S * Root)
PIpredUU = exp(pred$fit + qt(0.975, df = ndata - 2) * S * Root)
```

(Here, 95% bilateral quantiles)
Example with Stan code (same model)

```r
model <- "
data {
  int<lower=1> N; // rows of data
  vector[N] x;   // predictor
  int<lower=0> y[N]; // response
}
parameters {
  real<lower=0> phi; // neg. binomial dispersion parameter
  real b0;     // intercept
  real b1;     // slope
}
model {
  // priors:
  phi ~ cauchy(0, 20);
  b0 ~ normal(0, 20);
  b1 ~ normal(0,20);
  // data model:
  y ~ neg_binomial_2_log(b0 + b1 * x, phi);
}
"

stanmod <- stan(model_code = model,
               data=list(N=nrow(datas),x=datas$month, y=datas$y),
               iter=10000,chains=4)

chains <- extract(stanmod, pars = c("b0", "b1", "phi"))
x = 1:24
N = length(x)
simul = matrix(0,ncol=length(x),nrow=length(chains [[1]]))

for(i in seq_along(x)) {
  #draw from the predictive at every months
  predictive[i,] = rnegbin(length(chains$b0),
                            mu=exp(chains$b0 + chains$b1 * (x[i])),
                            theta = chains$phi)
}

PIPpred = apply(predictive[i, 2, function(l) {
  quantile(l,probs = c((1-0.95)/2,(1+0.95)/2))
})
- Why a “non-Bayesian” version?
  - On millions of subsets of data, running an MCMC sampler can be hopeless (depending on the customer architecture)
  - Generally, it is interesting to verify if an analytical solution can be identified
  - Unfortunately, this is not the case for negative binomial model
  - As shown, A “Normal” approximation can be developed, and coverages verified in various simulations
    - But it is noted that a real Bayesian prediction is more powerful, especially with small counts

Green: Bayesian interval (in Stan)
Red: Normal approx. (R code)
Time series analysis

- So far, the proposed solution answers yes/no
- Customers may have thousands of alerts, all needs to be verified

- How to improve the solution, by ranking these alerts
  - Provide a posterior probability that the data point is out-of-trend (OOT)

- Example following Stan implementation
  > ecdf(predictive[,24])(35000) #24 is the last time point (not included in the model)

[1] 0.98915 #probability to be OOT

- Instead of reporting yes/no, it is better to report P(OOT)
Time series analysis

■ Difficulties specific to big data
  – Customer hardware and software poorly adapted
  – Structure of the data
    Data are structured... but still, plenty of problems, leading to additional data consolidations
  – Finding robust simple models and handling error case
    E.g. in some cases, one model will not converge and R would crash if the error is not handled (try....catch mechanism)

■ Take time to build appropriate models, on which you make predictive inference!!
  – If the models are not good, inference is questionable... (sensitivity ↓↓)
  – Easy in small data... Close to impossible in big data
  – How to develop robust models without having seen (all) the data?
Case 2: Accelerometer data

- Pre-clinical data are gathered on animal to obtain an early idea of drug efficacy
  - 32 or 64 mice
  - Followed online during 3 to 6 weeks
  - Videos (like parking security videos)
  - 3-dimensional accelerometer data (a device is attached on their back)
  - Sampling frequency ~100hz

- Mice are epileptic-induced, and the different treatment groups (control, placebo, dose 1, dose 2, etc.) should see different (significant?) numbers of crisis

- Problem is that it is not possible to analyze every video
  - Instead, use the signal from accelerometers to automatically detect seizures
The goal is to detect epileptic seizures from accelerometer signals.

But sometimes, mice are scratching, running, dancing...
A first algorithm has been developed to be very sensitive (HMM model on simple features)

- Find signal patterns (“no movement – movement – no movement”)
- Can be run “online” during experiment
- ~100% detection

But the false positive rate was ~97%

- Highly skilled scientists need to confirm seizures visually (takes about 20 sec. / suspicion)
- Total number of found seizure is about 15k !!
- About 100h+ of video analysis (don’t do that), for a small experiment
- ...this is not accounting for coffee break... such rate is not acceptable

Let’s call ‘suspicions’ the detected seizure so far
Accelerometer data

To improve:

- Add a filter on top of the suspicion, based on more specific features
- As nobody knows what is a good feature

1) Extract many features, as orthogonal as possible (frequencies, amplitude, rolling SD)

2) Train a support vector machine (the skilled scientists already did the 100h+ hours analysis on some experiments, so we have training data...)

3) Tune the SVM (mainly, weighted SVM due to class highly unbalanced)

4) Verify/optimize using stratified cross-validation

<table>
<thead>
<tr>
<th></th>
<th>predicted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>actual</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>178998</td>
</tr>
<tr>
<td>1</td>
<td>103</td>
</tr>
<tr>
<td>total</td>
<td>179101</td>
</tr>
</tbody>
</table>

missed seizure = 2%
reduction in working time = 86%
Accelerometer data

- Implementation
  - No SVM available in “big data” R packages such as sparklyr or rsparkling
  - Sad, but not such an issue as we work only on the predefined suspicions
    Only about 15k chunks accelerometer data...
  - Suspicion accelerometer data can be accessed one by one
    • Features can be extracted
    • This summary is easily handled by one machine
    • Still embarrassingly parallel...

- "Bayesian" SVM not very available
  - Answer will remains simple “yes/no” decision
  - No ranking of the suspicions given how likely they would be a real seizure
  - Please implement it in BoomSpikeSlab 😊
The statistical question
- There is, and will always be, a trade-off between sensitivity and false positive rate

What is the impact of missing a few seizures
- Clinical relevance

Simulation study
Accelerometer data

- **Notations**
  
  \[ Y_i = \text{number of seizures in mouse } i, \text{ with } i = 1, \ldots, 30 \]
  
  \[ t = 30 \text{ days of follow-up (assumed constant across mice)} \]

- **Model** [mixed effects Poisson model for rate data]

  \[
  \log \left( \frac{\text{E}(Y_i)}{t} \right) = \alpha + \beta x_i + z_i
  \]

  with

  \[ x_i = \begin{cases} 
  0 & \text{if placebo} \quad (15 \text{ mice}) \\
  1 & \text{if treatment} \quad (15 \text{ mice}) 
  \end{cases} \]

  \[ z_i \overset{iid}{\sim} \mathcal{N}(0, \sigma^2_M) = \text{mouse random effect} \]

- **Parameter setting**

  \[ \alpha = 0 \quad \text{[a mouse from the control group with } z_i = 0 \text{ has, on average, 1 seizure per day]} \]

  \[ \beta = \log(0.65) \quad \text{[seizure rate reduction of 35% in the treatment group]} \]
See the impact with some seizure decrease assumptions and subject-to-subject variability.
Conclusions

- Bayesian statistics allow providing better answer through the use of the complete predictive distribution
  - Use it when feasible
  - Approximation is not a crime
- Making millions of models on small data is hard
  - Take time to adjust your model
  - Inference otherwise is questionable
- Be sure to answer the very question of the customer/scientist
  - Optimizing specificity and sensitivity or RMSECV is not sufficient
  - The results are generally used by others (we are just a small piece in a big process)

  Look at the impact of errors in the final outcome, instead of taking extra time to continue trying to optimize your classifier
Thanks

- Acknowledgement
  - Marco Munda (Arlenda)