Bayesian_Pricing_Exercise

Introduction

In this last scenario, I want to emphasize that the goal of analytics, is **not** building models, **not** prediction, and **not** data and technology. These are all tools - some of which are useful. I began this course describing how the term Analytics was first defined by Aristotle as a system of logic and inference, a way of understanding.

Pricing Scenario

RT Inc. sells engineered industrial equipment. They track opportunties to orders in the CRM system, which integrates with the ERP to create orders and invoices *(typical archiecture)*. Sales engineers are expected to track opportunities in the CRM and record detials on competitive position *(e.g., competitor prices)*. There's typically a lot of data that goes into CRMs *(contacts, calls, issues, feedback, meetings, actions, etc.)*, and much of that is summarized in forecasting indices. This exercise takes a small portion of that data *(most of these analyses involve 20+ dimensions)*.

Pricing is a frequent project in consulting *(interal and external)* - a favorite of CEO's. Why? Because every gain in pricing fall directly to the bottom line *(very little cost)*. These projects are also high visibility because poor pricing policies can lower margins and/or reduce sales volume. So it's an opporutnity to be a hero *(or a bozo)*. In this particular case, RT *(and many ETO companies)* assigns engineering resources to opportunities based on their preception *(intuition)* of Risk - probability of winning a deal.

We'll use the following data for our first analysis:

- SampleID
- RSF *(Relationship Factor - divided into 4 tiers: Green, Silver, Gold and Platinum - based on potential, purchase history, brand, etc. - with service levels allocated baed on tier)*
- QuoteDiff *(diffence between our quote and primary competitor quote)*
- RFPDiff *(difference between the dates the RFP response was requested, and when it was returned)*
- ATPDifference *(diffeence between the available to promise - ATP - date and the date required)*
- Result *(whether the opportunity was won or lost)*

Our goal in this case is to use Bayesian analysis for exploration and understanding. So our emphasis is not on prediction *(although we use prediction parameters for analysis)*, but on probabilistic inference.

Development of Analysis Model

First let's try to understand what causes wins and losses. The data are not nested - each observation is exchangable, so a crossed-effects, multiple regression model, transformed for classification using a logit function is where we start:

A few final transformations were implemented:

- The ATPDifference as scaled down *(divided by 1000)* to bring it into scale with the other dimensions and help the sampler.

- Data was divided into using a holdout validation set of 100 observations, with the rest for training *(dividing data into training and test is not necessary for exploratory analysis - just a levelset).*

- Result is transformed between logisitic and regression models to adapt to a *c(0,1)* outcome in logreg, and a *(1,2)* factor as an independent variable.

We use a typical logistic regression equation equation:

$P(\hat{Y}) = exp(\hat{\beta}X)/exp(1 + exp(\hat{\beta}X)$

Results are "levelset" *(not a predictive model)* with a small holdout set. After pulling the parameters from the sampler, we used the above equation to compute probability and then generated a binomial result using the following:

$Class = ifelse(Prob < .5, 0, 1))$

These results were run through a confusion matrix to begin analysis:

```
library(lme4)
library(tidyverse)
library(rstan)
library(shinystan)
library(gridExtra)
library(caret)
library(cowplot)
library(lubridate)
library(stringr)
library(ggridges)

set.seed(102)

setwd("C:/Users/ellen/Documents/UH/Spring 2020/DA2/tmpGitHub/EllenwTerry/Data_Files")
quoteData <- read.csv("QuoteData2.csv")



RSF = data.frame(RSF = c(1, 2, 3, 4))
Desc <- data.frame(RSFDesc = factor(c("Green", "Silver", "Gold", "Platinum"),
                                    levels = c("Green", "Silver", "Gold", "Platinum")))
RSFDesc = cbind(RSF, Desc)
quoteData = quoteData %>% inner_join(RSFDesc, by = "RSF")

quoteData <- quoteData %>% rownames_to_column("SampleID")
quoteData$SampleID  <- as.numeric(quoteData$SampleID)
quoteData$QuoteDiff <- quoteData$QuoteDiff/1000
train <- sample_n(quoteData, nrow(quoteData)-100)
test <- quoteData %>% anti_join(train, by = "SampleID")

xTrain <- select(train, RSF, QuoteDiff, RFPDiff, ATPDiff)
xTest <-  select(test, RSF, QuoteDiff, RFPDiff, ATPDiff)
x_train <- as.numeric(train$QuoteDiff)
y_train <- as.integer(train$Result)
N_train <- length(x_train)

# Quick Priors:
```

```
glm.priors <- glm(Result ~ RSF + QuoteDiff + RFPDiff + ATPDiff, data = quoteData, family = binomial)
mQuotes = model.matrix(Result ~ RSF + QuoteDiff + RFPDiff + ATPDiff, data = quoteData)
mTrain = model.matrix(Result ~ RSF + QuoteDiff + RFPDiff + ATPDiff, data = train)
mTest = model.matrix(Result ~ RSF + QuoteDiff + RFPDiff + ATPDiff, data = test)

beta = as.numeric(glm.priors$coefficients)

stanMod <- '
data {
  int N_train;
  int K;
  int y_train[N_train];
  matrix[N_train, K] x_train;
  real p_b[K];
  real<lower = 0> p_sb[K];
}
parameters {
  vector[K] beta;
}
transformed parameters {
  vector[N_train] y_hat;
  for(n in 1:N_train)
    y_hat[n] = x_train[n]*beta;
}
model {
  target += normal_lpdf(beta | p_b, p_sb);
  target += bernoulli_lpmf(y_train | inv_logit(y_hat));
}
'

beta = as.numeric(glm.priors$coefficients)
betaSigma  = rep(.2, 5)

fit <- stan(model_code=stanMod,
            data = list(
              N_train=nrow(mTrain),
              K=ncol(mTrain),
              y_train=train$Result,
              x_train=mTrain,
              p_b = beta,
              p_sb = betaSigma
            ), refresh = 0)


Coef <- summary(fit, pars = c('beta'), probs = c(0.1, 0.9))$summary[,1]
Stanbeta = summary(fit, pars = c('beta'), probs = c(0.1, 0.9))$summary[,1]

test$Prob <- exp(t(Stanbeta%*%t(mTest)))/(1+exp(t(Stanbeta%*%t(mTest))))
test = test %>% mutate (Class = if_else(Prob < .5, 0, 1))
```

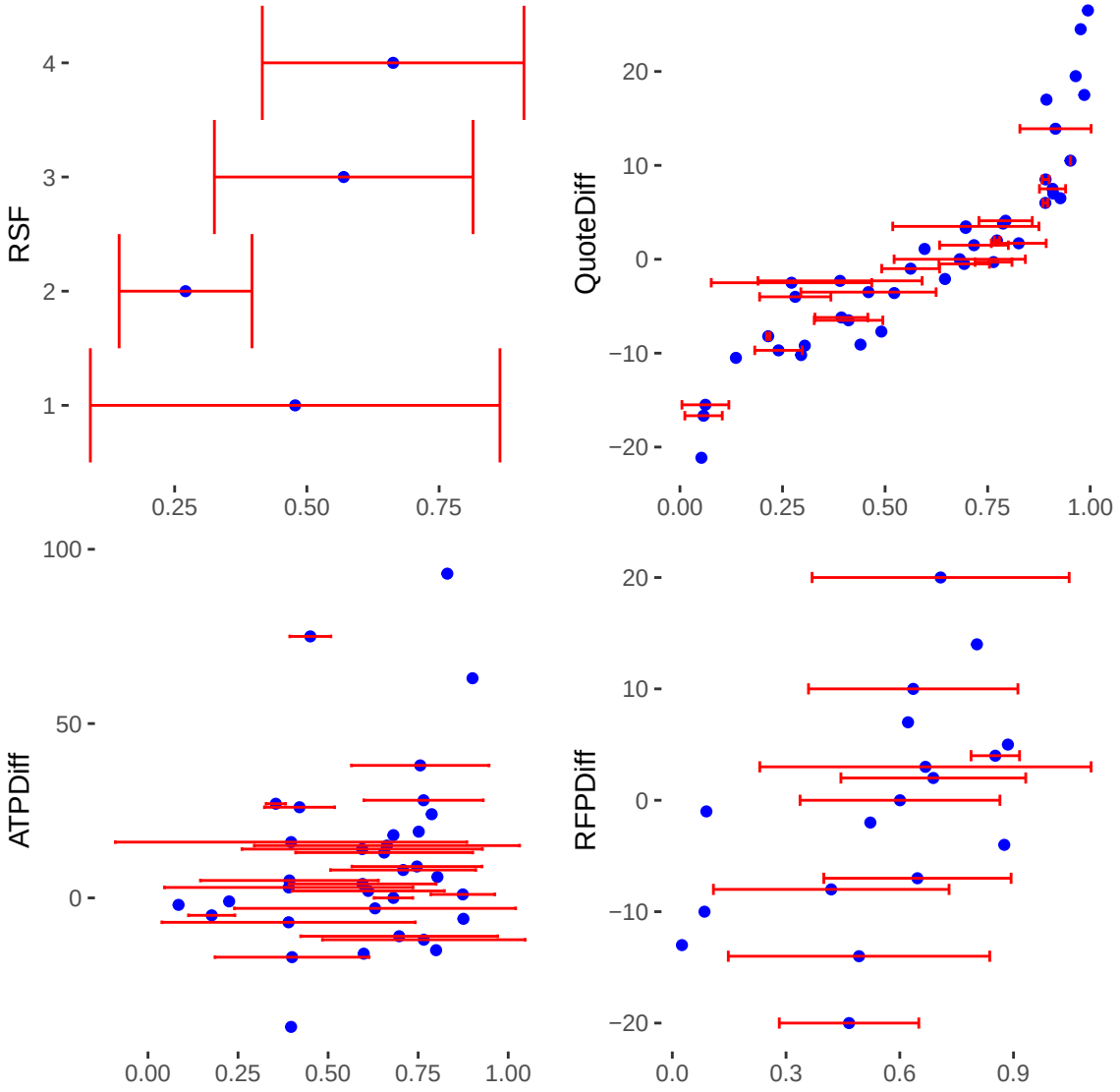
Confusion Matrix:

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction  0  1
##          0 25 10
##          1 18 47
##
##                Accuracy : 0.72
##                  95% CI : (0.6213, 0.8052)
##     No Information Rate : 0.57
##     P-Value [Acc > NIR] : 0.001409
##
##                   Kappa : 0.4154
##
##  Mcnemar's Test P-Value : 0.185877
##
##             Sensitivity : 0.8246
##             Specificity : 0.5814
##          Pos Pred Value : 0.7231
##          Neg Pred Value : 0.7143
##              Prevalence : 0.5700
##          Detection Rate : 0.4700
##    Detection Prevalence : 0.6500
##       Balanced Accuracy : 0.7030
##
##        'Positive' Class : 1
##
```

70% is a good start for exploratory analysis *(80+% accuracy in operational sales models is good - trust me)*. But for analysis, we're really interested in the parameters - that's the story. So, let's take a look at those:

```
dfPlotRSF <- test %>% group_by(RSF) %>% summarise(meanP = mean(Prob), sdP = sd(Prob))
p1 <- ggplot(dfPlotRSF, aes(x=meanP)) + geom_point(aes(y = RSF), color = 'blue') +
  geom_errorbarh(aes(xmin= (meanP-sdP) , xmax = (meanP + sdP), y = RSF), height = 1, color = "red") +
  theme(panel.background = element_rect(fill = "white")) +
  theme(axis.title.x=element_blank())

dfPlotQuote <- test %>% group_by(QuoteDiff) %>% summarise(meanP = mean(Prob), sdP = sd(Prob))
p2 <- ggplot(dfPlotQuote, aes(x=meanP)) + geom_point(aes(y = QuoteDiff), color = 'blue') +
  geom_errorbarh(aes(xmin= (meanP-sdP) , xmax = (meanP + sdP), y = QuoteDiff), height = 1, color = "red
  theme(panel.background = element_rect(fill = "white")) +
  theme(axis.title.x=element_blank())

dfPlotATP <- test %>% group_by(ATPDiff) %>% summarise(meanP = mean(Prob), sdP = sd(Prob))
p3 <- ggplot(dfPlotATP, aes(x=meanP)) + geom_point(aes(y = ATPDiff), color = 'blue') +
  geom_errorbarh(aes(xmin= (meanP - sdP) , xmax = (meanP + sdP), y = ATPDiff), height = 1, color = "red
  theme(panel.background = element_rect(fill = "white")) +
  theme(axis.title.x=element_blank())

dfPlotRFP <- test %>% group_by(RFPDiff) %>% summarise(meanP = mean(Prob), sdP = sd(Prob))
p4 <- ggplot(dfPlotRFP, aes(x=meanP)) + geom_point(aes(y = RFPDiff), color = 'blue') +
  geom_errorbarh(aes(xmin= (meanP - sdP) , xmax = (meanP + sdP), y = RFPDiff), height = 1, color = "red
  theme(panel.background = element_rect(fill = "white")) +
  theme(axis.title.x=element_blank())

plot_grid(p1, p2, p3, p4, align = 'h')
```



Lots of stories - when you start to dig into data like this, keep an holistic, business analytics mindset. Here are some initial questions:

- **RSF** is the customer tier. The "green" tier has a wider standard deviation than the others. Could it be that reponse to new customers is less consistent? Do new customers ask engineering for unrealisitic function? Why is the variance in "silver" customers less than than the others - is that just the sample or some characteristic of transition from "green" to "gold"? And why is success rate with "green" customers higher than "silver"?
- **QuoteDiff** is follows an expected pattern *(remeber, a positive number means that competitors' prices are higher, so probabity increases as QuoteDiff increases)*. Why does the standard deviation increase around 0?
- **ATPDiff** Difference between Available to Promise and Date Required is all over the board. Why? is this reliable? Is it too complex? Is there a consistent policy?
- **RFPDiff** Difference between RFP Due Date and Date Submitted is all over the place too. Why? Policy? What if we look at ATP and RFP by Customer Tier - do you think that might provide some insight?

Maybe we should dig a little deeper...

Extending Analysis of QuoteDiff with Regression

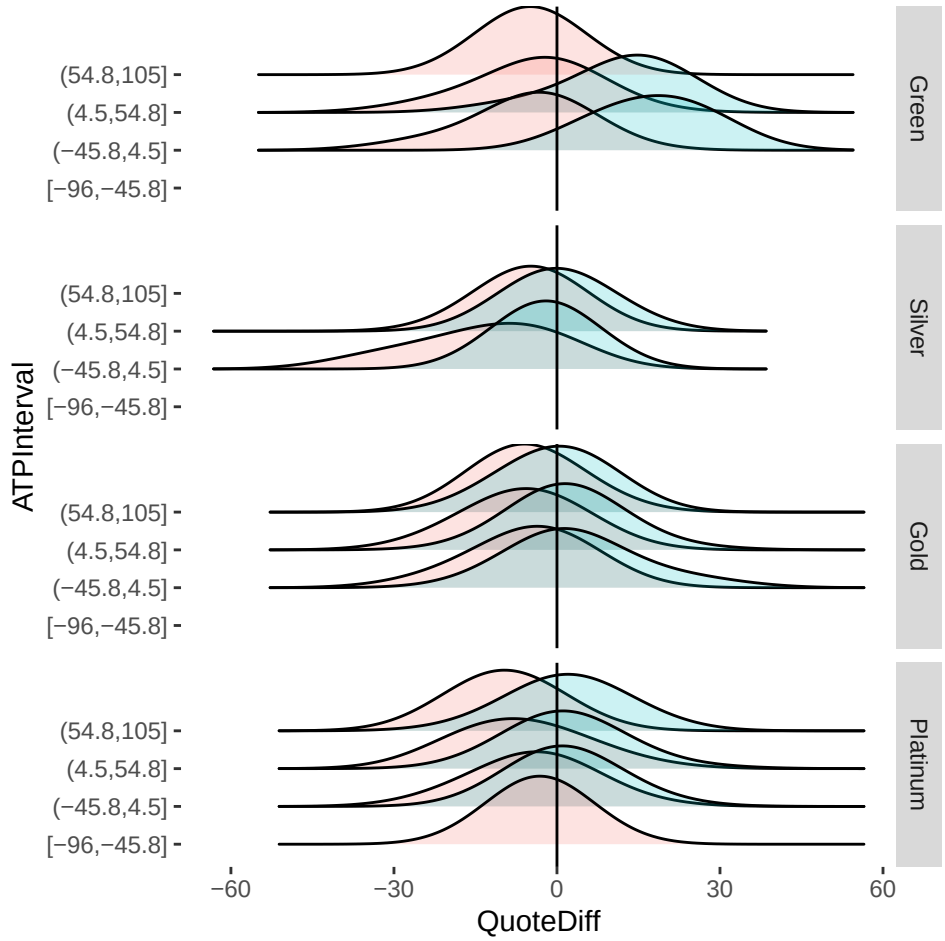
QuoteDiff appears to be the best predictor of Win %. Let's turn the analysis around and model QuoteDiff as a reponse variable - maybe there's a causal chain here *(maybe the customer tier affects price and engineering, which affects Win %)*. Let's create a regression model with QuoteDiff as the response.

Quick Priors:

```
quoteData$Result = quoteData$Result + 1
quoteData$Result = factor(quoteData$Result)

tstMod1 = lmer(QuoteDiff ~ 1 + ATPDiff +
                 (1 | RSF) +
                 (1 | Result),
               data = quoteData)
```

Before modeling, let's look at WvL% vs. RSF and ATP:

```
quoteData$ATPInterval = as.factor(cut_interval(quoteData$ATPDiff, 4))

p <- ggplot(quoteData, aes(y = ATPInterval, x = QuoteDiff, fill = Result)) +
  guides(fill=FALSE) +
  geom_density_ridges(bandwidth = 10, alpha = .2) +
  geom_vline(xintercept = 0) +
  facet_grid(rows = vars(RSFDesc)) +
  theme(panel.background = element_rect(fill = "white"))
p
```



What do you see? Any patterns? Is there a relationship between ATP, Tier and WvL? What could be causes? Why does the spread between W and L change so much for Green? Let's so this:

Build a Stan Regression model and determine the Win probability for green vs platinum tier for different QuoteDiffs. First we build a model that predicts the QuoteDiff, given the other dimensions *(including WL result)*

```
stanModel1 <- '
data {

int<lower=0> N;
int<lower=0> K;
int<lower=0> L;

vector[N] y;
real ATP[N];
int Result[N];
int RSF[N];

real p_ATP;
real p_Result[K];
real p_RSF[L];

}
```

```
parameters {

real<lower = 0> sigma[L];

real aATP;
vector[K] aResult;
vector[L] aRSF;

}
transformed parameters {

vector[N] y_hat;

for (i in 1:N)
y_hat[i] = aResult[Result[i]] + aRSF[RSF[i]] + aATP *ATP[i];

}
model {

target += normal_lpdf(y | y_hat,  sigma[RSF]);
target += normal_lpdf(aATP | p_ATP, .5);
target += normal_lpdf(aResult | p_Result, 2);
target += normal_lpdf(aRSF | p_RSF, .5);

}
'


stanData <- list(

  N=nrow(quoteData),

  K=length(unique(quoteData$Result)),
  L=length(unique(quoteData$RSF)),

  y=quoteData$QuoteDiff,
  ATP=as.numeric(quoteData$ATPDiff),
  Result = as.numeric(quoteData$Result),
  RSF = as.numeric(quoteData$RSF),

  p_ATP = fixef(tstMod1)[2],
  p_Result = coef(tstMod1)$`Result`[,1],
  p_RSF = coef(tstMod1)$`RSF`[,1]

)


fit1 <- stan(model_code = stanModel1, data = stanData, refresh = 0)
```

Now we can get the effects of each dimension.

```
sumFit1 <- data.frame(summary(fit1))
Coef <- summary(fit1, pars = c('aATP', 'aResult', 'aRSF', 'sigma'), probs = c(0.1, 0.9))$summary
```

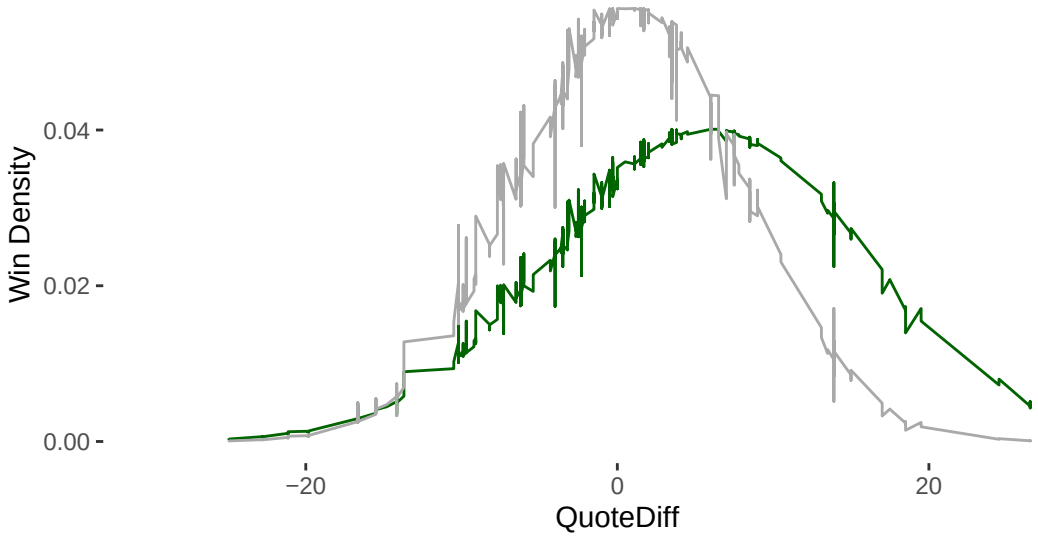
```
sigma = summary(fit1, pars = c("sigma"))$summary[,1]
ATPEffect = summary(fit1, pars = c("aATP"))$summary[,1]
ResultEffect = summary(fit1, pars = c("aResult"))$summary[,1]
RSFEffect = summary(fit1, pars = c("aRSF"))$summary[,1]

fDf = data.frame(QuoteDiff = seq(from = -60, to = 60, by = 1))

dMn = mean(quoteData$QuoteDiff)
dSD = sd(quoteData$QuoteDiff)
dWMn = mean(filter(quoteData, Result == 2)$QuoteDiff)
dWSD = sd(filter(quoteData, Result == 2)$QuoteDiff)
dLMn = mean(filter(quoteData, Result == 1)$QuoteDiff)
dLSD = sd(filter(quoteData, Result == 1)$QuoteDiff)
```

and we can use the effects to model te density of a win *(ResultEffect[2])*, comparing green RSFEffect[1] customers with platinum customers RSFEffect[4]. The value of RSF is evident here as platinum customers

```
1 - pnorm(0, dWMn, dWSD)
```

```
## [1] 0.6271071
```

```
# RSF Effects - quote = competitor, ATP = Required
p10 <- ggplot(quoteData) +
  geom_line(aes(QuoteDiff,
                y= dnorm(QuoteDiff,
                         mean = dMn+RSFEffect[1]+ResultEffect[2]+(ATPEffect*ATPDiff),
                         sd = sigma[1])), color  = 'darkgreen') +
  geom_line(aes(QuoteDiff,y= dnorm(QuoteDiff, mean = dMn+RSFEffect[4]+ResultEffect[2]+(ATPEffect*ATPDif
                                   sd = sigma[4])), color  = 'darkgray') +
  xlim(-30, 30) +
  ylab("Win Density") +
  theme(panel.background = element_rect(fill = "white"))
p10
```



Just looking at these distributions, we can see that the platinum customers are more likely to place orders when RT's prices are higher than competitors. Perhaps that's because new customers are just price shopping?