Research Appendix

Nicholas K. Neuteufel

November 9, 2015

R code for the data collection, management, and analysis (including the traditional ordered probit):

```
library(ggplot2)
library(dplyr)

#Import csv file from Qualtrics here
data <- read.csv(file="10-26data.csv", h=T, na.strings = "")

#Summary statistics

#Ranking the comms strategies
#n of 212 is different than typical n because drop-out/non-response/not changing the order
PollResult <- data$PollResult[!is.na(data$PollResult)]
LeadOverMoE <- data$LeadOverMoE[!is.na(data$LeadOverMoE)]
BattleForMoE <- data$BattleForMoE[!is.na(data$BattleForMoE)]
Trend <- data$Trend[!is.na(data$Trend)]

resultUpper <- round(mean(PollResult) + 1.96*(sd(PollResult)/sqrt(212)), 2)
resultLower <- round(mean(PollResult) - 1.96*(sd(PollResult)/sqrt(212)), 2)
resultMean <- round(mean(PollResult), 2)
result <- c(resultMean, resultLower, resultUpper)

leadUpper <- round(mean(LeadOverMoE) + 1.96*(sd(LeadOverMoE)/sqrt(212)), 2)
leadLower <- round(mean(LeadOverMoE) - 1.96*(sd(LeadOverMoE)/sqrt(212)), 2)
leadMean <- round(mean(LeadOverMoE), 2)
lead <- c(leadMean, leadLower, leadUpper)

battleUpper <- round(mean(BattleForMoE) + 1.96*(sd(BattleForMoE)/sqrt(212)), 2)
battleLower <- round(mean(BattleForMoE) - 1.96*(sd(BattleForMoE)/sqrt(212)), 2)
battleMean <- round(mean(BattleForMoE), 2)
battle <- c(battleMean, battleLower, battleUpper)

trendUpper <- round(mean(Trend) + 1.96*(sd(Trend)/sqrt(212)), 2)
trendLower <- round(mean(Trend) - 1.96*(sd(Trend)/sqrt(212)), 2)
trendMean <- round(mean(Trend), 2)
trend <- c(trendMean, trendLower, trendUpper)

comms <- rbind(result, lead, battle, trend)
strat <- c("Topline", "Lead>MoE", "MoE Gain", "Trend")
comms <- cbind(comms, strat)
colnames(comms) <- c("Mean", "Upper", "Lower", "Strategy")
comms <- data.frame(comms)

strategies <- ggplot(comms, aes(y=Mean, x=Strategy)) + guides(colour=F)
strategies <- strategies +
  geom_point(shape=12, size=5) + geom_errorbar(wigth=0.1, aes(ymax=comms$Upper, ymin=comms$Lower))
strategies + ylab("Rank") + theme_bw()
```

```r
#Subset by assignment group
controlGroup <- filter(data, !is.na(Control))
barGroup <- filter(data, !is.na(Bar))
dotGroup <- filter(data, !is.na(Dot))

#Make a result vector/column (called Likelihood)for the probit model
#and tell R the treatment group
controlGroup$Treatment <- 1
controlGroup$Likelihood <- controlGroup$Control
barGroup$Treatment <- 2
barGroup$Likelihood <- barGroup$Bar
dotGroup$Treatment <- 3
dotGroup$Likelihood <- dotGroup$Dot

final <- rbind(controlGroup, barGroup, dotGroup)
final <- filter(final, Likelihood != "I Do Not Know",
                Likelihood != "Very Unlikely", Likelihood != "Unlikely",
                Likelihood != "Somewhat Unlikely")
final <- select(final, -c(Control, Dot, Bar))
final$Likelihood <- as.character(final$Likelihood)
final$Likelihood[final$Likelihood=="Neither Likely nor Unlikely"] <- "Neither Likely Nor Unlikely"
#Stupid me
final$Likelihood[final$Likelihood=="Very Likely"] <- "Likely" #Group these
#Ordering the Likelihoods
final$Likelihood[final$Likelihood=="Neither Likely Nor Unlikely"] <- 1
final$Likelihood[final$Likelihood=="Somewhat Likely"] <- 2
final$Likelihood[final$Likelihood=="Likely"] <- 3
#Factor the important things
final$Likelihood <- as.factor(final$Likelihood)
final$Treatment <- as.factor(final$Treatment)

#Treatment summary statistics
library(ggplot2)
ggplot(final, aes(x=Treatment, y=Likelihood)) + geom_point(size=0.75) +
  geom_jitter(alpha = 0.75)

#Marginal effects: Simple
library(mfx)
probeME <- probitmfx(as.ordered(Likelihood) ~
                       PresentationPreference+StatsClass+Treatment,
                     data=final, atmean=F, robust=T)
probeME
#Marginal effects: Pretreatment too
#Make poll trust into "Agree", "Disagree", and "Neither"
final$PollTrust <- as.character(final$PollTrust)
final$PollTrust[final$PollTrust=="Strongly Agree"] <- "Agree"
final$PollTrust[final$PollTrust=="Somewhat Agree"] <- "Agree"
final$PollTrust[final$PollTrust=="Strongly Disagree"] <- "Disagree"
final$PollTrust[final$PollTrust=="Somewhat Disagree"] <- "Disagree"
final$PollTrust <- as.factor(final$PollTrust)
#Group "Less than once a month" for horse race exposure into Never and
#"Once a month/weeek" into "Occasionally" and "2-3x a week" and "Daily" into "Often"
final$HorseRaceExposure <- as.character(final$HorseRaceExposure)
```

```
final$HorseRaceExposure[final$HorseRaceExposure=="Less than Once a Month"] <- "Never"
final$HorseRaceExposure[final$HorseRaceExposure=="Once a Month"] <- "Occasionally"
final$HorseRaceExposure[final$HorseRaceExposure=="Once a Week"] <- "Occasionally"
final$HorseRaceExposure[final$HorseRaceExposure=="2-3 Times a Week"] <- "Often"
final$HorseRaceExposure[final$HorseRaceExposure=="Daily"] <- "Often"
final$HorseRaceExposure <- as.factor(final$HorseRaceExposure)
#Find marginal effects
probeMEpretreatment <- probitmfx(as.ordered(Likelihood) ~
                                   PollTrust+HorseRaceExposure+
                                   PresentationPreference+StatsClass+Treatment,
                                 data=final, atmean=F, robust=T)
probeMEpretreatment
#Test the standard errors for heterosckedasicity
probeMEpretreatmentNotRobust <- probitmfx(as.ordered(Likelihood) ~
                                            PollTrust+HorseRaceExposure+PresentationPreference+
                                            StatsClass+Treatment,
                                          data=final, atmean=F, robust=F)
probeMEpretreatmentNotRobust
#Compare them by converting them to dataframes and then subtracting the Std..Err. vectors
robust <- data.frame(probeMEpretreatment$mfxest)
notRobust <- data.frame(probeMEpretreatmentNotRobust$mfxest)
difference <- robust$Std..Err.-notRobust$Std..Err.
#Average differences
mean(difference)
median(difference)

#Add Education analysis
finalEdu <- final
finalEdu$EduLevel <- as.character(finalEdu$EduLevel)
finalEdu$EduLevel[finalEdu$EduLevel=="Completed trade/vocational school"] <- "Before college"
finalEdu$EduLevel[finalEdu$EduLevel=="Some HS or less"] <- "Before college"
finalEdu$EduLevel[finalEdu$EduLevel=="HS Graduate/GED completed"] <- "Before college"
finalEdu$EduLevel[finalEdu$EduLevel=="Graduated college"] <- "Post-college"
finalEdu$EduLevel[finalEdu$EduLevel=="Post-graduate or more"] <- "Post-college"
finalEdu$EduLevel <- as.factor(finalEdu$EduLevel)
summary(finalEdu$EduLevel)
#Marginal effects
probeMEpretreatmentEdu <- probitmfx(as.ordered(Likelihood) ~ EduLevel+PollTrust+HorseRaceExposure+
                                      PresentationPreference+StatsClass+Treatment,
                                    data=finalEdu, atmean=F, robust=T)
probeMEpretreatmentEdu

#Add Gender
final$Gender <- as.character(final$Gender)
summary(final$Gender)
final$Gender[final$Gender=="Other/Prefer not to respond"] <- NA
probeMEpretreatmentEduG <- probitmfx(as.ordered(Likelihood) ~
                                       Gender+EduLevel+PollTrust+HorseRaceExposure+
                                       PresentationPreference+StatsClass+Treatment,
                                     data=finalEdu, atmean=F, robust=T)
probeMEpretreatmentEduG

#From general
```

```
#Drop trust
probeMEpretreatmentMinusTrust <- probitmfx(as.ordered(Likelihood) ~ HorseRaceExposure+
                                             PresentationPreference+StatsClass+Treatment,
                                           data=final, atmean=F, robust=T)
probeMEpretreatmentMinusTrust
#Trust instead of exposure
probeMEminusExposure <- probitmfx(as.ordered(Likelihood) ~
                                    PollTrust+PresentationPreference+
                                    StatsClass+Treatment, data=final, atmean=F, robust=T)
probeMEminusExposure
#Neither
probeMEneither <- probitmfx(as.ordered(Likelihood) ~
                              PresentationPreference+StatsClass+Treatment,
                            data=final, atmean=F, robust=T)
probeMEneither

#Analyze surprising result of stats class
finalStats <- filter(final, !is.na(final$StatsClassAgo))
nrow(finalStats)
finalStats$StatsClassAgo <- as.character(finalStats$StatsClassAgo)
finalStats$StatsClassAgo[finalStats$StatsClassAgo=="6+ years ago"] <- "3-5 years ago"
#Group not-recent
finalStats$StatsClassAgo[finalStats$StatsClassAgo=="Less than a year ago"] <- 1
finalStats$StatsClassAgo[finalStats$StatsClassAgo=="1-3 years ago"] <- 2
finalStats$StatsClassAgo[finalStats$StatsClassAgo=="3-5 years ago"] <- 3
finalStats$StatsClassAgo <- as.factor(finalStats$StatsClassAgo)
summary(finalStats$StatsClassAgo)
#New model replacing stats class with this new factor
probeMEduration <- probitmfx(as.ordered(Likelihood) ~
                               HorseRaceExposure+PollTrust+PresentationPreference
                             +StatsClassAgo+Treatment, data=finalStats, atmean=F, robust=T)
probeMEduration
#Take out both other factors
probeMEdurationNo <- probitmfx(as.ordered(Likelihood) ~
                                 PresentationPreference+StatsClassAgo+Treatment,
                               data=finalStats, atmean=F, robust=T)
probeMEdurationNo
#Just treatment
probeMEdurationSimple <- probitmfx(as.ordered(Likelihood) ~
                                     Treatment, data=finalStats, atmean=F, robust=T)
probeMEdurationSimple

#Analyzing the trend question via ordered probit
finalTrend <- subset(final, !is.na(final$Trend))
finalTrend$Trend <- as.factor(finalTrend$Trend)
probeTrend <- probitmfx(Trend~
                          PollTrust+HorseRaceExposure,
                        data=finalTrend, atmean=F, robust=T)
probeTrend

#Export to CSV
finalStata <- final
#Convert poll trust to numbers
```

```
finalStata$PollTrust <- as.character(finalStata$PollTrust)
finalStata$PollTrust[finalStata$PollTrust=="Agree"] <- 1
finalStata$PollTrust[finalStata$PollTrust=="Disagree"] <- 2
finalStata$PollTrust[finalStata$PollTrust=="Neither Agree nor Disagree"] <- 3
finalStata$PollTrust <- as.numeric(finalStata$PollTrust)
#Presentation preference
finalStata$PresentationPreference <- as.character(finalStata$PresentationPreference)
finalStata$PresentationPreference[finalStata$PresentationPreference=="Graphics"] <- 1
finalStata$PresentationPreference[finalStata$PresentationPreference=="Text"] <- 2
finalStata$PresentationPreference <- as.numeric(finalStata$PresentationPreference)
#Horse race exposure
finalStata$HorseRaceExposure <- as.character(finalStata$HorseRaceExposure)
finalStata$HorseRaceExposure[finalStata$HorseRaceExposure=="Never"] <- 1
finalStata$HorseRaceExposure[finalStata$HorseRaceExposure=="Occasionally"] <- 2
finalStata$HorseRaceExposure[finalStata$HorseRaceExposure=="Often"] <- 3
finalStata$HorseRaceExposure <- as.numeric(finalStata$HorseRaceExposure)
#Stats class
finalStata$StatsClass <- as.character(finalStata$StatsClass)
finalStata$StatsClass[finalStata$StatsClass=="No"] <- 1
finalStata$StatsClass[finalStata$StatsClass=="Yes"] <- 2
finalStata$StatsClass <- as.numeric(finalStata$StatsClass)
#Write the csv--saved time bc Treatment and Likelihood already numeric
write.csv(file="stataExport.csv", x=finalStata)
```

Bayesian ordered probit:

```
bayesData <- read.csv(file="gistStuff.csv", h=T) #Formerly known as `final'
set.seed(12) #reproducibility
bayesData$Treatment <- as.factor(bayesData$Treatment)

library(Zelig)
bayesianOProbit <- zelig(Likelihood~
                           Treatment+HorseRaceExposure+PollTrust+
                           PresentationPreference+StatsClass,
                         model="oprobit.bayes", data=bayesData)
#Check for convergence
heidel.diag(bayesianOProbit$result$coefficients) #Yay
raftery.diag(bayesianOProbit$result$coefficients) #Yay
#Summarize results
summary(bayesianOProbit)
```