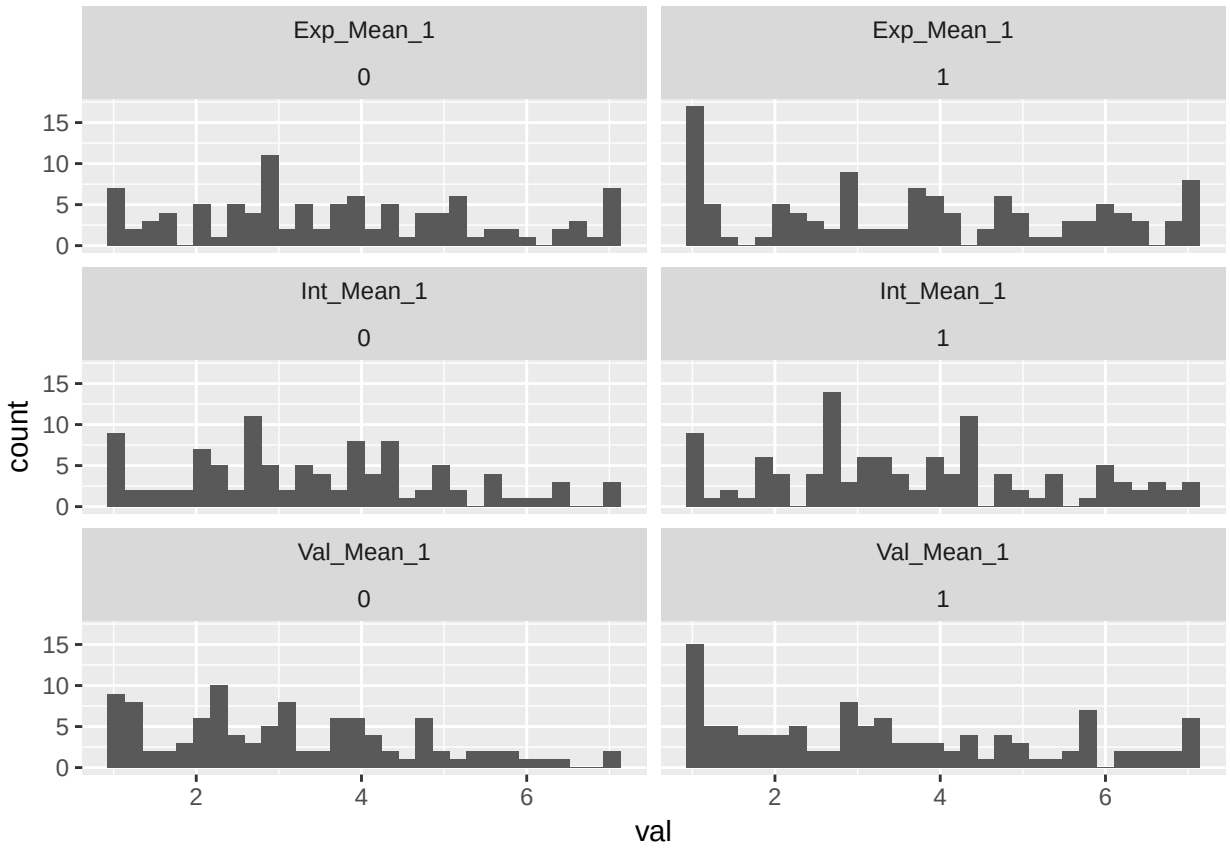
# CT_Survey

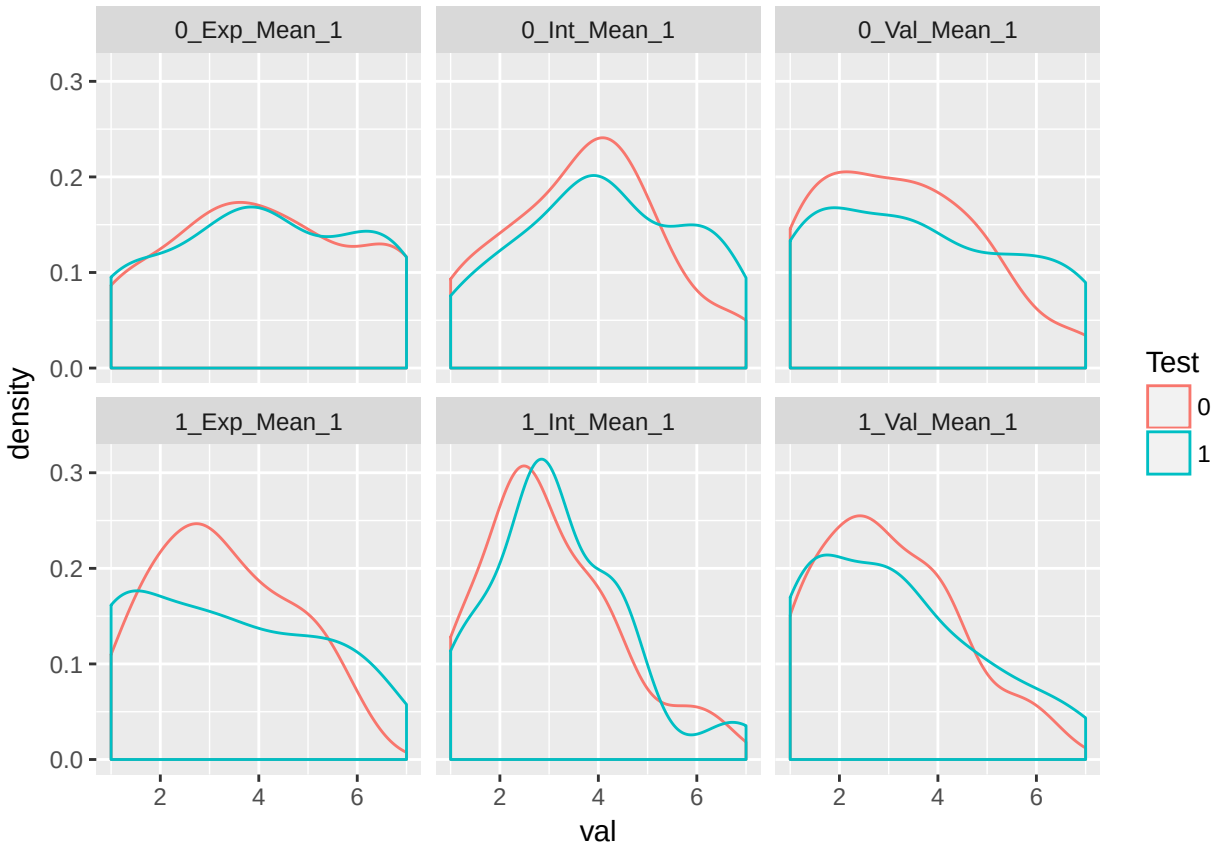
*Mete and Josh*

*7/26/2017*

```
library(tidyverse)
library(haven)
library(lsmeans)

df <- read_sav("FullData.sav")

# df %>%
#   mutate(performance = ifelse(df$DBN_POMP == 0, 0, 1)) %>%
#   mutate(test_results = ifelse(DBN_POMP == 0, 0, 1)) %>%
#   group_by(ProgramingExtent, test_results) %>%
#   select(Exp_Mean_1, Int_Mean_1, Val_Mean_1) %>%
#   summarize_all(funs(mean)) %>%
#   mutate(test_results = as.factor(test_results))

df %>%
  select(contains("Mean_1"), Test) %>%
  gather(key, val, -Test) %>%
  mutate(Test = as.factor(Test)) %>%
  ggplot(aes(x = val)) +
  geom_histogram() +
  facet_wrap( ~ key + Test, ncol = 2)
```




```
df %>%
  select(contains("Mean_1"), Test, Gender) %>%
  gather(key, val, -Test, -Gender) %>%
  mutate(Test = as.factor(Test)) %>%
  filter(!is.nan(Gender)) %>%
  unite(facet_var, Gender, key) %>%
  ggplot(aes(x = val, colour = Test)) +
  geom_density() +
  facet_wrap( ~ facet_var)
```




```
# girls possibly calibrate their expectancies
# boys possibly get more interested
# girls and boys possibly both value it more

df %>%
  select(contains("Mean_1"), Test, Gender, DBN_POMP) %>%
  mutate(DBN_POMP_0 = ifelse(DBN_POMP == 0, 0, 1),
         DBN_POMP_17 = ifelse(DBN_POMP <= 17, 0, 1)) %>%
  filter(!is.nan(Gender)) %>%
  group_by(Test, Gender, DBN_POMP_0)  %>%
  summarize_if(is.numeric, funs(mean)) %>%
  select(Test, Gender, DBN_POMP_0, everything(), -DBN_POMP, -DBN_POMP_17)
```

```
## # A tibble: 6 x 6
## # Groups:   Test, Gender [4]
##        Test    Gender DBN_POMP_0 Int_Mean_1 Val_Mean_1 Exp_Mean_1
##   <dbl+lbl> <dbl+lbl>      <dbl>      <dbl>      <dbl>      <dbl>
## 1         0         0         NA   3.710345   3.234553   4.128211
## 2         0         1         NA   3.006977   2.983721   3.255814
## 3         1         0          0   3.537668   2.946802   3.748624
## 4         1         0          1   4.603030   4.333333   4.451098
## 5         1         1          0   3.156522   3.430435   3.608696
## 6         1         1          1   3.176923   2.857692   3.087932
```

```
df <- df %>%
  mutate(DBN_POMP_0 = ifelse(DBN_POMP == 0, 0, 1),
         DBN_POMP_17 = ifelse(DBN_POMP <= 17, 0, 1),
         Test = as.factor(Test),
```

```
         DBN_POMP_0 = as.factor(DBN_POMP_0))

# df_ss <- filter(df, !is.na(DBN_POMP_0))
df_ss <- filter(df, !is.nan(Gender))
df_ss$Gender <- as.factor(df_ss$Gender)
df_ss<-mutate(df_ss,
              DBNandNoTest = case_when(
                Test == 0 ~ "notest",
                DBN_POMP_0 == 0 ~ "lowperformance",
                DBN_POMP_0 == 1 ~ "moderateperformance",
                TRUE ~ "NA"))
```

```
m5 <- aov(Val_Mean_1 ~ as.factor(Gender) * DBNandNoTest, data = df_ss)
summary(m5)
```

```
##                                Df Sum Sq Mean Sq F value  Pr(>F)
## as.factor(Gender)               1    8.5   8.492   2.996 0.08498 .
## DBNandNoTest                    2   13.0   6.476   2.284 0.10440
## as.factor(Gender):DBNandNoTest  2   27.5  13.768   4.857 0.00868 **
## Residuals                     207  586.8   2.835
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
TukeyHSD(m5)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Val_Mean_1 ~ as.factor(Gender) * DBNandNoTest, data = df_ss)
##
## $`as.factor(Gender)`
##           diff        lwr        upr    p adj
## 1-0 -0.4030943 -0.8622522 0.05606356 0.084983
##
## $DBNandNoTest
##                                          diff        lwr        upr
## moderateperformance-lowperformance  0.52907764 -0.2231503 1.28130560
## notest-lowperformance              -0.03223066 -0.7063956 0.64193424
## notest-moderateperformance         -0.56130831 -1.2126032 0.08998662
##                                        p adj
## moderateperformance-lowperformance 0.2230137
## notest-lowperformance              0.9930024
## notest-moderateperformance         0.1065609
##
## $`as.factor(Gender):DBNandNoTest`
##                                                    diff        lwr
## 1:lowperformance-0:lowperformance             0.48363298 -0.8586934
## 0:moderateperformance-0:lowperformance        1.38653153  0.1647391
## 1:moderateperformance-0:lowperformance       -0.08910949 -1.3868607
## 0:notest-0:lowperformance                     0.28775085 -0.8014587
## 1:notest-0:lowperformance                     0.03691913 -1.1152369
## 0:moderateperformance-1:lowperformance        0.90289855 -0.4126833
## 1:moderateperformance-1:lowperformance       -0.57274247 -1.9591541
## 0:notest-1:lowperformance                    -0.19588213 -1.3893461
## 1:notest-1:lowperformance                    -0.44671385 -1.6978903
```

```
## 1:moderateperformance-0:moderateperformance -1.47564103 -2.7457094
## 0:notest-0:moderateperformance              -1.09878068 -2.1548550
## 1:notest-0:moderateperformance              -1.34961240 -2.4704955
## 0:notest-1:moderateperformance               0.37686035 -0.7662383
## 1:notest-1:moderateperformance               0.12602862 -1.0772006
## 1:notest-0:notest                           -0.25083173 -1.2255008
##                                                    upr     p adj
## 1:lowperformance-0:lowperformance            1.8259594 0.9051271
## 0:moderateperformance-0:lowperformance       2.6083239 0.0159952
## 1:moderateperformance-0:lowperformance       1.2086417 0.9999581
## 0:notest-0:lowperformance                    1.3769604 0.9737942
## 1:notest-0:lowperformance                    1.1890752 0.9999991
## 0:moderateperformance-1:lowperformance       2.2184804 0.3606981
## 1:moderateperformance-1:lowperformance       0.8136691 0.8420565
## 0:notest-1:lowperformance                    0.9975819 0.9970506
## 1:notest-1:lowperformance                    0.8044626 0.9084195
## 1:moderateperformance-0:moderateperformance -0.2055727 0.0124917
## 0:notest-0:moderateperformance              -0.0427064 0.0361275
## 1:notest-0:moderateperformance              -0.2287293 0.0083901
## 0:notest-1:moderateperformance               1.5199590 0.9332744
## 1:notest-1:moderateperformance               1.3292579 0.9996637
## 1:notest-0:notest                            0.7238374 0.9766401
```

```
m5a <- aov(Exp_Mean_1 ~ as.factor(Gender) * DBNandNoTest, data = df_ss)
summary(m5a)
```

```
##                                Df Sum Sq Mean Sq F value  Pr(>F)
## as.factor(Gender)               1   35.6   35.62  10.582 0.00133 **
## DBNandNoTest                    2    0.8    0.41   0.123 0.88469
## as.factor(Gender):DBNandNoTest  2   10.4    5.19   1.541 0.21651
## Residuals                     207  696.8    3.37
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
m5b <- aov(Int_Mean_1 ~ as.factor(Gender) * DBNandNoTest, data = df_ss)
summary(m5b)
```

```
##                                Df Sum Sq Mean Sq F value   Pr(>F)
## as.factor(Gender)               1   35.0   35.02  14.629 0.000173 ***
## DBNandNoTest                    2   14.7    7.36   3.077 0.048235 *
## as.factor(Gender):DBNandNoTest  2    8.2    4.09   1.709 0.183507
## Residuals                     207  495.5    2.39
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
m6 <- aov(Int_Mean_1 ~ Gender * DBNandNoTest + ProgramingExtent, data = df_ss)
summary(m6)
```

```
##                     Df Sum Sq Mean Sq F value   Pr(>F)
## Gender               1   35.0   35.02  15.261 0.000127 ***
## DBNandNoTest         2   14.7    7.36   3.209 0.042416 *
## ProgramingExtent     1   24.2   24.18  10.537 0.001365 **
## Gender:DBNandNoTest  2    6.8    3.39   1.479 0.230219
## Residuals          206  472.8    2.29
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
lsmeans(ref.grid(m6), pairwise ~ Gender * DBNandNoTest, adjust = "tukey")
```

```
## $lsmeans
##  Gender DBNandNoTest          lsmean        SE  df lower.CL upper.CL
##  0      lowperformance      3.542077 0.2765855 206 2.996776 4.087379
##  1      lowperformance      3.238454 0.3169472 206 2.613578 3.863330
##  0      moderateperformance 4.541860 0.2644241 206 4.020535 5.063184
##  1      moderateperformance 3.292313 0.2993450 206 2.702141 3.882486
##  0      notest              3.626987 0.2006677 206 3.231362 4.022613
##  1      notest              3.049686 0.2314176 206 2.593436 3.505937
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast                                          estimate        SE  df
##  0,lowperformance - 1,lowperformance            0.30362348 0.4205738 206
##  0,lowperformance - 0,moderateperformance      -0.99978236 0.3827192 206
##  0,lowperformance - 1,moderateperformance       0.24976392 0.4074365 206
##  0,lowperformance - 0,notest                   -0.08490984 0.3418203 206
##  0,lowperformance - 1,notest                    0.49239098 0.3605770 206
##  1,lowperformance - 0,moderateperformance      -1.30340585 0.4139869 206
##  1,lowperformance - 1,moderateperformance      -0.05385956 0.4337726 206
##  1,lowperformance - 0,notest                   -0.38853333 0.3769598 206
##  1,lowperformance - 1,notest                    0.18876750 0.3915416 206
##  0,moderateperformance - 1,moderateperformance  1.24954628 0.4011848 206
##  0,moderateperformance - 0,notest               0.91487252 0.3303946 206
##  0,moderateperformance - 1,notest               1.49217335 0.3521370 206
##  1,moderateperformance - 0,notest              -0.33467376 0.3630597 206
##  1,moderateperformance - 1,notest               0.24262707 0.3770529 206
##  0,notest - 1,notest                            0.57730083 0.3074715 206
##  t.ratio p.value
##    0.722  0.9791
##   -2.612  0.0989
##    0.613  0.9900
##   -0.248  0.9999
##    1.366  0.7475
##   -3.148  0.0229
##   -0.124  1.0000
##   -1.031  0.9071
##    0.482  0.9967
##    3.115  0.0253
##    2.769  0.0667
##    4.237  0.0005
##   -0.922  0.9406
##    0.643  0.9875
##    1.878  0.4189
##
## P value adjustment: tukey method for comparing a family of 6 estimates
```

```r
m6a <- aov(Val_Mean_1 ~ Gender * DBNandNoTest + ProgramingExtent, data = df_ss)
summary(m6a)
```

```
##                      Df Sum Sq Mean Sq F value  Pr(>F)
## Gender                1    8.5    8.49   3.142 0.07778 .
```

```
## DBNandNoTest          2   13.0    6.48   2.396 0.09362 .
## ProgramingExtent      1   33.1   33.11  12.250 0.00057 ***
## Gender:DBNandNoTest   2   24.5   12.23   4.525 0.01193 *
## Residuals           206  556.8    2.70
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
lsmeans(ref.grid(m6a), pairwise ~ Gender * DBNandNoTest, adjust = "tukey")
```

```
## $lsmeans
##  Gender DBNandNoTest          lsmean        SE  df lower.CL upper.CL
##  0      lowperformance      2.951864 0.3001593 206 2.360086 3.543642
##  1      lowperformance      3.524498 0.3439611 206 2.846363 4.202634
##  0      moderateperformance 4.263105 0.2869613 206 3.697348 4.828863
##  1      moderateperformance 2.990168 0.3248586 206 2.349694 3.630642
##  0      notest              3.138852 0.2177708 206 2.709507 3.568198
##  1      notest              3.032754 0.2511417 206 2.537617 3.527892
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast                                          estimate        SE  df
##  0,lowperformance - 1,lowperformance           -0.57263460 0.4564199 206
##  0,lowperformance - 0,moderateperformance      -1.31124147 0.4153389 206
##  0,lowperformance - 1,moderateperformance      -0.03830445 0.4421629 206
##  0,lowperformance - 0,notest                   -0.18698856 0.3709541 206
##  0,lowperformance - 1,notest                   -0.08089064 0.3913094 206
##  1,lowperformance - 0,moderateperformance      -0.73860687 0.4492716 206
##  1,lowperformance - 1,moderateperformance       0.53433015 0.4707436 206
##  1,lowperformance - 0,notest                    0.38564604 0.4090886 206
##  1,lowperformance - 1,notest                    0.49174396 0.4249132 206
##  0,moderateperformance - 1,moderateperformance  1.27293702 0.4353784 206
##  0,moderateperformance - 0,notest               1.12425292 0.3585545 206
##  0,moderateperformance - 1,notest               1.23035084 0.3821501 206
##  1,moderateperformance - 0,notest              -0.14868411 0.3940038 206
##  1,moderateperformance - 1,notest              -0.04258619 0.4091896 206
##  0,notest - 1,notest                            0.10609792 0.3336777 206
##  t.ratio p.value
##   -1.255  0.8091
##   -3.157  0.0223
##   -0.087  1.0000
##   -0.504  0.9960
##   -0.207  0.9999
##   -1.644  0.5704
##    1.135  0.8662
##    0.943  0.9349
##    1.157  0.8564
##    2.924  0.0439
##    3.136  0.0238
##    3.220  0.0184
##   -0.377  0.9990
##   -0.104  1.0000
##    0.318  0.9996
##
## P value adjustment: tukey method for comparing a family of 6 estimates
```

```r
m6b <- aov(Exp_Mean_1 ~ Gender * DBNandNoTest + ProgramingExtent, data = df_ss)
summary(m6b)
```

```
##                       Df Sum Sq Mean Sq F value  Pr(>F)
## Gender                 1   35.6   35.62  11.016 0.00107 **
## DBNandNoTest           2    0.8    0.41   0.128 0.88026
## ProgramingExtent       1   32.5   32.50  10.050 0.00176 **
## Gender:DBNandNoTest    2    8.5    4.26   1.318 0.26994
## Residuals            206  666.1    3.23
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
lsmeans(ref.grid(m6b), pairwise ~ Gender * DBNandNoTest, adjust = "tukey")
```

```
## $lsmeans
##  Gender DBNandNoTest          lsmean        SE  df lower.CL upper.CL
##  0      lowperformance      3.753737 0.3283130 206 3.106452 4.401021
##  1      lowperformance      3.703707 0.3762232 206 2.961965 4.445448
##  0      moderateperformance 4.380163 0.3138771 206 3.761339 4.998986
##  1      moderateperformance 3.221742 0.3553290 206 2.521195 3.922290
##  0      notest              4.031547 0.2381969 206 3.561931 4.501163
##  1      notest              3.305341 0.2746977 206 2.763762 3.846921
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast                                          estimate        SE  df
##  0,lowperformance - 1,lowperformance             0.05003022 0.4992302 206
##  0,lowperformance - 0,moderateperformance       -0.62642595 0.4542960 206
##  0,lowperformance - 1,moderateperformance        0.53199443 0.4836359 206
##  0,lowperformance - 0,notest                    -0.27781005 0.4057480 206
##  0,lowperformance - 1,notest                     0.44839554 0.4280126 206
##  1,lowperformance - 0,moderateperformance       -0.67645617 0.4914114 206
##  1,lowperformance - 1,moderateperformance        0.48196421 0.5148975 206
##  1,lowperformance - 0,notest                    -0.32784026 0.4474594 206
##  1,lowperformance - 1,notest                     0.39836532 0.4647683 206
##  0,moderateperformance - 1,moderateperformance  1.15842038 0.4762151 206
##  0,moderateperformance - 0,notest                0.34861591 0.3921855 206
##  0,moderateperformance - 1,notest                1.07482149 0.4179942 206
##  1,moderateperformance - 0,notest               -0.80980447 0.4309597 206
##  1,moderateperformance - 1,notest               -0.08359889 0.4475699 206
##  0,notest - 1,notest                             0.72620559 0.3649753 206
##  t.ratio p.value
##    0.100  1.0000
##   -1.379  0.7397
##    1.100  0.8809
##   -0.685  0.9835
##    1.048  0.9011
##   -1.377  0.7410
##    0.936  0.9367
##   -0.733  0.9777
##    0.857  0.9561
##    2.433  0.1500
##    0.889  0.9488
```

```
##    2.571  0.1091
##   -1.879  0.4180
##   -0.187  1.0000
##    1.990  0.3518
##
## P value adjustment: tukey method for comparing a family of 6 estimates
```