~~~
---
output:
  word_document: default
  html_document: default
---
--
title: "Final_DKE_R_Script"
author: "dnsanc"
date: "7/20/2020"
output: word_document
---

```{r, libraries}
library(readr)
library(psych)
library(janitor)
library(lme4)
library(ggplot2)
library(dplyr)
library(Rmisc)
library(lm.beta)
library(car)
library(matrixStats)
library(Hmisc)
library(jmv)
library(xtable)
library(htmltools)
library(psycho)
library(tidyverse)
library(broom)
library(tidyr)
library(psycho)
library(tidyverse)
library(Hmisc)
library(ppcor)
library(corpcor)
library(rsq)
library(lmerTest)
library(jtools)
library(ggstance)
library(broom.mixed)
library(huxtable)
library(flextable)
library(sjPlot)
library(sjmisc)
library(sjlabelled)
library(stargazer)
library(multilevel)
library(bda)
~~~

~~~
library(gvlma)
library(QuantPsyc)
library(interactions)
library(gridExtra)
library(gtable)
library(reshape2)
library(ggpubr)
```

```{r data call}

#changes from raw data to files
#Removed the first line in the CSV before loading
#also removed spaces in column headers and substitute periods in

df.summary<- read_csv("~/Desktop/DKE/data_summary.csv")
View(df.summary)
#removing subjects for non-compliance
  #row 1 was removed because it was a pilot subject that was not blind to the
experiment, but was checking overall question quality and grammar
  #rows 2-5 were removed as there was a concern about two questions in the
logic portion. The two questions were reworded slightly. We can remove the
questions, but it might be simpler to remove the participants.
  #row 9 was my attempt at the task to supply Michael with a test sheet
  #row 48 was also removed for not providing estimates of how many questions
they answered correctly
df.summary<-df.summary[-c(1:5,9,48),]

df.master<- read_csv("~/Desktop/DKE/data_scores_master.csv")
df.master<-remove_empty(df.master, which = c("rows", "cols"), quiet = TRUE)
df.master<-df.master[-c(1:5,9,48),]

#df.conf<-read_csv("~/Desktop/DKE/DKE/data_confidence.csv")

#combine the datasummary script with the data master. Make sure you do a TRUE/
False check to make sure the dataIDs match and sort each file to oldest first.

df.TotalDKE<-df.master[c(1,3,8:102, 201,406, 312, 314, 316, 318, 320, 322, 324, 326, 328,
376,378,380,382,384,386,388,390,392,394,396,398,400,402,404,407,409,411,413,415,417,419,42





#write.csv(df.TotalDKE, "renaming.csv")
```


```{r, renaming columns}
~~~

```
df.TotalDKE<-df.TotalDKE %>%
  dplyr::rename(
    Subject = Respondent.ID,
    OCQ.Accuracy = "Accuracy",
    OCQ.Bias = "Bias",
    OCQ.Accuracy.NW = "Accuracy.Nuclear",
    OCQ.TotalAccuracy = "Accuracy.with.Nuclear",
    Pre.Log.Est = "Pre-task.Logic.Correct.Answers.Estimate",
    Pre.Log.EstPerc = "Pre-task.Logic.Accuracy.Esimate",
    Pre.LogSandian = "Pre-task.Logic.Sandian.Comparison.Estimate",
    Pre.Log.Dif  = "Pre-task.Logic.Difficulty.Estimate",
   # Overall.Correct.Answers = "Correct.Overall.Answers",
    #Overall.Correct.AnswersPerc = "Overall.Correct.Answersuracy",
    Post.Log.Est= "Post-task.Logic.Correct.Answers.Esimate",
    Post.Log.EstPerc= "Post-task.Logic.Accuracy.Esimate",
    Post.LogSandian = "Post-task.Logic.Sandian.Comparison.Estimate",
    Post.Log.Dif = "Post-task.Logic.Difficulty.Estimate",
    Logic.Acc = "Correct.Logic.Answers",
    Logic.AccPerc = "Logic.Accuracy",
    Logic.Overest = "Post-task.Logic.Overestimation.Score",
    Pre.Gram.Est  =  "Pre-task.Correct.Grammar.Answers.Estimate",
    Pre.Gram.EstPerc = "Pre-task.Grammar.Accuracy.Esimate",
    Pre.GramSandian  = "Pre-task.Grammar.Sandian.Comparison.Estimate",
    Pre.Gram.Dif = "Pre-task.Grammar.Difficulty.Estimate",
    Gram.Acc = "Correct.Grammar.Answers",
    Gram.AccPerc = "Grammar.Accuracy",
    Post.Gram.Est = "Post-task.Correct.Grammar.Answers.Esimate",
    Post.Gram.EstPerc = "Post-task.Grammar.Accuracy.Esimate",
    Post.GramSandian= "Post-task.Grammar.Sandian.Comparison.Estimate",
    Post.Gram.Dif = "Post-task.Grammar.Difficulty.Estimate",
    Pre.NW.Est = "Pre-task.NW.Correct.Answers.Esimate",
    Pre.NW.EstPerc = "Pre-task.NW.Accuracy.Esimate",
    Pre.NWSandian = "Pre-task.NW.Sandian.Comparison.Estimate",
    Pre.NW.Dif = "Pre-task.NW.Difficulty.Estimate",
    NW.Acc = "NW.Correct.Answers",
    NW.AccPerc = "NW.Accuracy",
    Post.NW.Est = "Post-task.NW.Correct.Answers.Esimate",
    Post.NW.EstPerc = "Post-task.NW.Accuracy.Esimate",
    Post.NWSandian = "Post-task.NW.Sandian.Comparison.Estimate",
    Post.NW.Dif = "Post-task.NW.Difficulty.Estimate",
    #Overall.Correct.Answers = "Overall.Correct.Answers",
    #Overall.Est = "Overall.Correct.Answers.Estimate",
    #Overall.Overest = "Overall.Overestimation.Score",
    #Overall.Overest = "Post-task.Overall.Overestimation.Score",
    Grammar.Overest = "Post-task.Grammar.Overestimation.Score",
    NW.Overest = "Post-task.NW.Overestimation.Score",
   Logic_1 = "In.her.introductory.psychology.class,.Anna.received.both.the.
14th.highest.and.the.
14th.lowest.test.score.in.the.class..How.many.students.are.in.the.class?",
Logic_2 = "A.pet.store.owner.buys.a.hamster.for.$50,.sells.it.for.
$60,.buys.it.back.for.$70,.and.sells.it.finally.for.
$80..How.much.has.he.made?",
Logic_3 = "A.package.of.golf.balls.and.a.set.of.tees.cost.
$1.10.in.total..The.package.of.golf.balls.costs.
$1.00.more.than.the.set.of.tees..How.much.does.the.set.of.tees.cost?",
Logic_4 = "Eric.drinks.a.gallon.of.juice.in.
6.days.and.Harry.drinks.a.gallon.of.juice.in.
12.days..How.long.would.it.take.them.to.drink.a.gallon.of.juice.together?",
Logic_5 = "If.it.takes.5.machines.5.minutes.to.make.
5.components,.how.long.would.it.take.100.machines.to.make.100.components?",
Logic_6 = "In.a.pond,.there.is.a.growing.bloom.of.algae..Every.day,.the.algae.doubles.in.s
48.days.for.the.bloom.to.cover.the.entire.pond,.how.long.would.it.take.for.only.half.the.p

Logic_7 = "Efraim.decided.to.invest.
$3,000.in.the.stock.market..Seven.months.after.his.original.investment.date.of.May.
17,.his.stocks.were.down.50%..From.May.17.to.August.
17,.the.stocks.he.had.purchased.went.up.75%..At.this.point,.Efraim:",
Logic_8 =
"If.youre.in.a.biking.competition.and.you.pass.the.biker.in.third.place,.what.place.are.yo
Logic_9 = "A.dentist.had.
15.exotic.fish.in.his.aquarium.tank.to.keep.patients.calm..All.but.
8.died..How.many.are.left?",
Logic_10 = "Bellas.owner.has.three.dogs.",
Logic_11 = "How.many.cubic.feet.",
Logic_12 =
"The.Jacobsons.had.decided.that.if.they.replaced.their.car.they.would.get.what.they.called
Logic_13 = "The.town.of.Recida.has.two.hospitals,.a.big.and.a.small.one..In.the.big.hospit
50.children.are.born.each.day..In.the.small.hospital.about.20.children.are.born..Around.
50.percent.of.all.children.are.girls,.though.the.percentage.changes.every.day..Sometimes.t
50.percent,.sometimes.lower..For.a.period.of.1.year,.each.hospital.recorded.the.days.on.wh
60.percent.of.the.children.born.were.girls..Which.hospital.do.you.think.recorded.more.such



Logic_14 = "Imagine.a.container.filled.with.gum.balls..
2/3.of.the.gumballs.in.the.container.are.cherry.flavored.and.
1/3.are.mango.flavored..Kara.has.drawn.5.pieces.of.gum.from.the.container.and.found.that.
4.were.cherry.and.1.was.mango..Tyler.has.drawn.20.pieces.of.gum.and.found.that.12.were.che
8.were.mango..Which.of.the.two.individuals.should.feel.more.confident.that.the.container.c
2/3.cherry.and.1/3.mango.balls,.rather.than.the.opposite?",
Logic_15 = "Given.the.hypothesis.that.the.container.contains.2/3.cherry.and.
1/3.mango.balls,.rather.than.the.opposite,.what.odds.should.Kara.give.that.the.container.c
Logic_16 = "Given.the.hypothesis.that.the.container.contains.2/3.cherry.and.
1/3.mango.balls,.rather.than.the.opposite,.what.odds.should.Tyler.give.that.the.container.
Logic_17 = "A.game.of.racquetball.can.be.played.to.either.8.or.
16.points..Keeping.all.other.rules.for.the.game.the.same,.if.Adam.is.a.better.player.than.
Logic_18 = "After.the.first.
3.weeks.of.the.season,.newspapers.print.the.batting.averages.for.the.top.
10.players.in.major.league.baseball..The.leading.batter.often.has.an.average.of.about..
430.during.the.first.
3.weeks.of.the.season..However,.no.batter.has.managed.to.maintain.an.average.of..
430.at.the.end.of.the.season..Why.do.you.think.this.is?.Select.one:",
Logic_19 = "When.playing.slot.machines,.people.win.something.about.1.in.every.
10.times..Rebecca,.however,.has.just.won.on.her.first.three.plays..What.are.her.chances.of
Logic_20 = "Imagine.that.we.are.tossing.a.fair.coin.
(a.coin.that.has.an.equal.chance.
(50%.likelihood).of.landing.on.heads.or.tails).and.it.has.just.come.up.heads.
5.times.in.a.row..For.the.6th.toss.do.you.think.that:",
Logic_21 = "A.doctor.had.been.working.on.a.cure.for.a.mysterious.disease.",
Logic_22 = "Amy.is.
28.years.old,.single,.outspoken,.and.very.bright..She.majored.in.History..As.a.student,.sh
nuclear.demonstrations.Is.Amy.more.likely.to.be:",
Logic_23 =
"The.city.of.Middleton.has.had.an.unpopular.police.chief.for.a.year.and.a.half..The.police
12%..Which.of.the.following.pieces.of.evidence.would.most.damage.the.mayors.claim.that.the
Logic_24 =
"Imagine.you.are.a.doctor.attempting.to.diagnose.a.patient.with.boils.on.his.fingers..What
1..percentage.of.people.without.Digititis.who.have.boils.on.their.fingers.2..percentage.of


Logic_25 = "You.are.playing.a.game.with.two.stages..In.the.first.stage,.there.is.a.
60%.chance.to.end.the.game.without.winning.anything.and.a.
40%.chance.to.move.into.the.second.stage..If.you.reach.the.second.stage.you.have.a.choice.
1..Sure.win.of.$402..80%.chance.to.win.$60.and.
20%.chance.to.win.nothingWhich.of.the.following.options.do.you.prefer?",

Logic_26 =
"Assume.that.you.are.presented.with.two.containers.of.red.and.green.marbles:.a.large.conta
100.marbles.and.a.small.container.that.contains.
10.marbles..The.marbles.are.spread.in.a.single.layer.on.each.container..You.must.draw.out.
(without.peeking,.of.course).from.either.container..If.you.draw.a.red.marble,.you.win.
$2..Consider.a.condition.in.which.the.small.container.contains.1.red.marble.and.
9.green.marbles,.and.the.large.container.contains.8.red.marbles.and.
92.green.marbles..From.which.container.would.you.prefer.to.select.a.marble.in.a.real.situa
"A.die.with.4.red.faces.and.2.green.faces.",
Logic_28 = "Situation.
1:.The.weather.outside.is.awful.and.you.decide.to.stay.in.for.the.day..To.keep.yourself.oc
$7.99..After.10.minutes.you.are.bored.and.the.movie.seems.pretty.bad..Would.you.continue.t
2:.The.weather.outside.is.awful.and.you.decide.to.stay.in.for.the.day..To.keep.yourself.oc
10.minutes.you.are.bored.and.the.movie.seems.pretty.bad..Would.you.continue.to.watch.the.m
In.which.of.these.two.situations.would.you.continue.to.watch.the.movie?",

Logic_29 = "Patient.Harold:.A.60-year-
old.man.named.Harold.had.a.heart.condition..He.had.to.stop.working.because.of.chest.pain..
65.to.age.70..However,.8%.of.the.people.who.have.this.operation.die.from.the.operation.its
old.man.named.Roger.had.a.heart.condition..He.had.to.stop.working.because.of.chest.pain..H
65.to.age.70..However,.2%.of.the.people.who.have.this.operation.die.from.the.operation.its



Logic_Conf_1 =
"How.confident.are.you.that.you.answered.this.question.correctly?",
Logic_Conf_2 =
"How.confident.are.you.that.you.answered.this.question.correctly?_1",
Logic_Conf_3 =
"How.confident.are.you.that.you.answered.this.question.correctly?_2",
Logic_Conf_4 =
"How.confident.are.you.that.you.answered.this.question.correctly?_3",
Logic_Conf_5 =
"How.confident.are.you.that.you.answered.this.question.correctly?_4",
Logic_Conf_6 =
"How.confident.are.you.that.you.answered.this.question.correctly?_5",
Logic_Conf_7 =
"How.confident.are.you.that.you.answered.this.question.correctly?_6",
Logic_Conf_8 =
"How.confident.are.you.that.you.answered.this.question.correctly?_7",
Logic_Conf_9 =
"How.confident.are.you.that.you.answered.this.question.correctly?_8",
Logic_Conf_10 =
"How.confident.are.you.that.you.answered.this.question.correctly?_9",
Logic_Conf_11 =
"How.confident.are.you.that.you.answered.this.question.correctly?_10",
Logic_Conf_12 =
"How.confident.are.you.that.you.answered.this.question.correctly?_11",
Logic_Conf_13 =
"How.confident.are.you.that.you.answered.this.question.correctly?_12",
Logic_Conf_14 =
"How.confident.are.you.that.you.answered.this.question.correctly?_13",
Logic_Conf_15 =
"How.confident.are.you.that.you.answered.this.question.correctly?_14",
Logic_Conf_16 =
"How.confident.are.you.that.you.answered.this.question.correctly?_15",
Logic_Conf_17 =
"How.confident.are.you.that.you.answered.this.question.correctly?_16",
Logic_Conf_18 =
"How.confident.are.you.that.you.answered.this.question.correctly?_17",
Logic_Conf_19 =
"How.confident.are.you.that.you.answered.this.question.correctly?_18",
Logic_Conf_20 =
"How.confident.are.you.that.you.answered.this.question.correctly?_19",
Logic_Conf_21 =
"How.confident.are.you.that.you.answered.this.question.correctly?_20",
Logic_Conf_22 =
"How.confident.are.you.that.you.answered.this.question.correctly?_21",
Logic_Conf_23 =
"How.confident.are.you.that.you.answered.this.question.correctly?_22",
Logic_Conf_24 =
"How.confident.are.you.that.you.answered.this.question.correctly?_23",
Logic_Conf_25 =
"How.confident.are.you.that.you.answered.this.question.correctly?_24",
Logic_Conf_26 =
"How.confident.are.you.that.you.answered.this.question.correctly?_25",
Logic_Conf_27 =
"How.confident.are.you.that.you.answered.this.question.correctly?_26",
Logic_Conf_28 =
"How.confident.are.you.that.you.answered.this.question.correctly?_27",
Logic_Conf_29 =
"How.confident.are.you.that.you.answered.this.question.correctly?_28",
Grammar_1 =
"The.Round.Island.Channel.is.a.Lake.Huron.waterway.between.Mackinac.Island.and.Round.Islan
,
Grammar_2 =
"Settlers.in.the.eighteenth,.and.nineteenth,.centuries.crossed.the.Round.Island.Channel.by
,
Grammar_3 = "By.the.1880s,.the.Michigan.Legislature.had.begun.discussing.building.a.bridge
built.Williamsburg.Bridge..However,.the.path.forward.was.not.easy..During.the.late.ninetee
,

Grammar_4 =
"Statistics.consistently.indicate.that.car.accidents.occur.more.often.during.the.night.tha
,
Grammar_5 =
"These.statistics.takes.into.account.that.there.are.fewer.drivers.on.the.road.at.night."
,
Grammar_6 = "One.such.design.employs.headlights.that.swivel.back.and.forth,.allowing.the.d
beam.headlights.are.another.innovative.design.that.could.improve.the.drivers.reaction.time
,
Grammar_7 = "All.cars.have.been.equipped.with.high-
beam.headlight.switches.for.many.years,.most.drivers.do.not.use.their.high-
beams.even.when.they.would.provide.a.great.deal.more.light.on.the.road." ,
Grammar_8 =
"Even.as.car.makers.work.diligently.to.improve.safety.on.the.road.particularly.during.dang
,
Grammar_9 =
"More.advanced.technologies.might.improve.safety,.but.the.person.in.the.drivers.seat.plays
driving.incidents." ,
Grammar_10 = "Two.other.systems.that.are.being.developed.to.potentially.increase.safety.on
Infrared).system..In.the.NIR.system,.an.infrared.light.is.emitted.from.the.front.of.the.ve
,


Grammar_11 =
"Dog.obedience.training.is.an.important.undertaking.when.one.acquires.a.new.dog..This.is.p
,
Grammar_12 =
"Many.owners.dont.understand.that.the.animals.communicate.back..Carefully.watching.a.dog.m
,
Grammar_13 =
"Other.forms.of.body.language.can.also.indicate.which.emotion.a.dog.is.experiencing..For.e
,
Grammar_14 =
"Traveling.across.time.zones.particularly.via.airplane,.can.be.very.disruptive.to.the.huma
,
Grammar_15 = "When.you.gain.or.lose.time.while.travelling,.a.condition.
(desynchronosis).is.likely.to.affect.you..Jet.lag.is.medically.considered.a.sleeping.disor
,
Grammar_16 = "My.best.friend.Tony.is.known.for.not.paying.attention.to.many.of.the.modern.
sooner.mail.something.than.sit.at.a.computer.to.type..Therefore,.I.was.unsurprised.to.lear
off.game.of.the.season,.he.was.sad.to.learn.that.he.did.not.have.the.proper.cable.channels


Grammar_17 = "After.being.unable.to.watch.the.first.play-
off.game.of.the.season,.Tony.decided.to.purchase.cable.and.asked.me.to.help..I.spoke.to.a.
$15.per.month,.would.more.than.suffice,.I.arranged.to.have.it.installed." ,
Grammar_18 = "John.Bronson.is.an.old-fashioned,.modern-
day.blacksmith.who.still.practices.the.fine.art.of.manipulating.metal.over.a.hot.fire..In.
,
Grammar_19 = "John.Bronson.had.began.his.career.in.hand-forged.ironwork.at.the.age.of.
30..The.idea.of.creating.an.object.out.of.iron.greatly.appealed.to.him..He.started.on.this
,

Grammar_20 =
"Once.Bronson.obtained.his.first.portable.forge,.he.was.ready.to.build.his.blacksmith.shop
,
Grammar_21 =
"Bronsons.shop.was.a.crude.building.but.stood.for.only.eight.years..Bronson,.who.by.then.w
,
Grammar_22 =
"Scorpions.will.sting.anyone.they.accidentally.encounter..Of.the.ninety-
five.scorpions.native.to.the.United.States,.25.percent.live.in.Nevada." ,
Grammar_23 =
"Unfortunately,.one.of.those.species.native.to.Nevada.is.the.Deathstalker.Scorpion,.just.a
,
Grammar_24 = "Over.a.period.of.fifteen.years,.a.Korean.man.by.the.name.of.Kalalyani.Chaluk
four.characters..Unlike.an.alphabet,.where.each.letter.represents.a.basic.sound.of.speech,
,

Grammar_25 =
"Few.facts.exist.regarding.the.life.of.Kalalyani.Chalukyas,.the.information.that.is.availa
,
Gram_Conf_1 =
"How.confident.are.you.that.you.answered.this.question.correctly?_29",
Gram_Conf_2 =
"How.confident.are.you.that.you.answered.this.question.correctly?_30",
Gram_Conf_3 =
"How.confident.are.you.that.you.answered.this.question.correctly?_31",
Gram_Conf_4 =
"How.confident.are.you.that.you.answered.this.question.correctly?_32",
Gram_Conf_5 =
"How.confident.are.you.that.you.answered.this.question.correctly?_33",
Gram_Conf_6 =
"How.confident.are.you.that.you.answered.this.question.correctly?_34",
Gram_Conf_7 =
"How.confident.are.you.that.you.answered.this.question.correctly?_35",
Gram_Conf_8 =
"How.confident.are.you.that.you.answered.this.question.correctly?_36",
Gram_Conf_9 =
"How.confident.are.you.that.you.answered.this.question.correctly?_37",
Gram_Conf_10 =
"How.confident.are.you.that.you.answered.this.question.correctly?_38",
Gram_Conf_11 =
"How.confident.are.you.that.you.answered.this.question.correctly?_39",
Gram_Conf_12 =
"How.confident.are.you.that.you.answered.this.question.correctly?_40",
Gram_Conf_13 =
"How.confident.are.you.that.you.answered.this.question.correctly?_41",
Gram_Conf_14 =
"How.confident.are.you.that.you.answered.this.question.correctly?_42",
Gram_Conf_15 =
"How.confident.are.you.that.you.answered.this.question.correctly?_43",
Gram_Conf_16 =
"How.confident.are.you.that.you.answered.this.question.correctly?_44",
Gram_Conf_17 =
"How.confident.are.you.that.you.answered.this.question.correctly?_45",
Gram_Conf_18 =
"How.confident.are.you.that.you.answered.this.question.correctly?_46",
Gram_Conf_19 =
"How.confident.are.you.that.you.answered.this.question.correctly?_47",
Gram_Conf_20 =
"How.confident.are.you.that.you.answered.this.question.correctly?_48",
Gram_Conf_21 =
"How.confident.are.you.that.you.answered.this.question.correctly?_49",
Gram_Conf_22 =
"How.confident.are.you.that.you.answered.this.question.correctly?_50",
Gram_Conf_23 =
"How.confident.are.you.that.you.answered.this.question.correctly?_51",
Gram_Conf_24 =
"How.confident.are.you.that.you.answered.this.question.correctly?_52",
Gram_Conf_25 =
"How.confident.are.you.that.you.answered.this.question.correctly?_53",
NW_1 = "How.does.a.nuclear.weapon.(NW).differ.from.a.conventional.bomb?",
NW_2 = "What.is.the.most.important.degree.NW.",
NW_3 = "Who.was.the.military.lead.of.the.Manhattan.Project?",
NW_4 = "What.did.Albert.Einstein.do.for.the.Manhattan.project?",
NW_5 =
"What.was.the.single.most.important.discovery.in.the.development.of.nuclear.weapons?",
NW_6 = "How.many.neutrons.(on.average).are.given.off.by.fission?",
NW_7 = "What.material.is.the.best.for.nuclear.weapon.(fission.only)?",
NW_8 = "How.do.explosives.
(main.charge).in.nuclear.weapon.differ.from.those.in.conventional.bombs?",
NW_9 = "What.are.the.main.ingredients.in.a.modern.explosive?",
NW_10 = "Which.tail.numbers.are.still.in.the.stockpile?",
NW_11 = "What.were.the.secret.cities.formed.for.the.Manhattan.Project?",
NW_12 = "What.reaction.releases.the.most.energy.per.molecule?",
NW_13 = "What.are.the.declared.5.nuclear.weapon.states?",
NW_14 = "Which.answer.is.not.a.nuclear.weapon.free-zone?",
NW_15 = "What.is.the.difference.between.fast.and.slow.neutrons?",
NW_16 = "Who.explained.the.energy.balance.of.fission?",
NW_17 = "What.is.process.the.the.DOE.follows.to.update.nuclear.weapons?",
NW_18 = "What.document.regulates.classification.in.nuclear.weapons?",
NW_19 = "What.parameter(s).define.critical.mass?",
NW_20 = "Who.owns.US.nuclear.weapons?",
NW_21 =
"Amongst.all.competing.priorities.in.building.and.maintaining.Nuclear.Weapons,.which.one.t
NW_22 =
"Who.is.the.greatest.proliferator.of.nuclear.weapon.in.modern.history?",
NW_23 = "What.makes.nuclear.weapon.safe?",
NW_24 = "What.makes.nuclear.weapons.secure?",
NW_25 = "What.factor.is.key.in.determining.the.yield.of.a.nuclear.weapon.?",
NW_26 = "How.many.full.scale.nuclear.weapon.
1pt.safety.tests.have.been.conducted?",
NW_27 = "What.is.the.National.Ignition.Facility.trying.to.accomplish?",
NW_Conf_1 = "How.confident.are.you.that.you.answered.this.question.correctly?
_54",
NW_Conf_2 = "How.confident.are.you.that.you.answered.this.question.correctly?
_55",
NW_Conf_3 = "How.confident.are.you.that.you.answered.this.question.correctly?
_56",
NW_Conf_4 = "How.confident.are.you.that.you.answered.this.question.correctly?
_57",
NW_Conf_5 = "How.confident.are.you.that.you.answered.this.question.correctly?
_58",
NW_Conf_6 = "How.confident.are.you.that.you.answered.this.question.correctly?
_59",
NW_Conf_7 = "How.confident.are.you.that.you.answered.this.question.correctly?
_60",
NW_Conf_8 = "How.confident.are.you.that.you.answered.this.question.correctly?
_61",
NW_Conf_9 = "How.confident.are.you.that.you.answered.this.question.correctly?
_62",
NW_Conf_10 = "How.confident.are.you.that.you.answered.this.question.correctly?
_63",
NW_Conf_11 = "How.confident.are.you.that.you.answered.this.question.correctly?
_64",
NW_Conf_12 = "How.confident.are.you.that.you.answered.this.question.correctly?
_65",
NW_Conf_13 = "How.confident.are.you.that.you.answered.this.question.correctly?
_66",
NW_Conf_14 = "How.confident.are.you.that.you.answered.this.question.correctly?
_67",
NW_Conf_15 = "How.confident.are.you.that.you.answered.this.question.correctly?
_68",
NW_Conf_16 = "How.confident.are.you.that.you.answered.this.question.correctly?
_69",
NW_Conf_17 = "How.confident.are.you.that.you.answered.this.question.correctly?
_70",
NW_Conf_18 = "How.confident.are.you.that.you.answered.this.question.correctly?
_71",
NW_Conf_19 = "How.confident.are.you.that.you.answered.this.question.correctly?
_72",
NW_Conf_20 = "How.confident.are.you.that.you.answered.this.question.correctly?
_73",
NW_Conf_21 = "How.confident.are.you.that.you.answered.this.question.correctly?
_74",
NW_Conf_22 = "How.confident.are.you.that.you.answered.this.question.correctly?
_75",
```

~~~
NW_Conf_23 = "How.confident.are.you.that.you.answered.this.question.correctly?
_76",
NW_Conf_24 = "How.confident.are.you.that.you.answered.this.question.correctly?
_77",
NW_Conf_25 = "How.confident.are.you.that.you.answered.this.question.correctly?
_78",
NW_Conf_26 = "How.confident.are.you.that.you.answered.this.question.correctly?
_79",
NW_Conf_27 = "How.confident.are.you.that.you.answered.this.question.correctly?
_80",
)

#write.csv(df.TotalDKE, "subjectnoncomply.csv")
```

```{r, descriptives}

#For later regression analysis, recoding fields of experience

df.TotalDKE$Mathematics[df.TotalDKE$Mathematics=="Mathematics"]<-1
df.TotalDKE$Computer.and.information.sciences[df.TotalDKE$Computer.and.information.science
and information sciences"]<-1
df.TotalDKE$Physical.sciences[df.TotalDKE$Physical.sciences=="Physical
sciences"]<-1
df.TotalDKE$Chemical.sciences[df.TotalDKE$Chemical.sciences=="Chemical
sciences"]<-1
df.TotalDKE$Earth.and.related.environmental.science[df.TotalDKE$Earth.and.related.environm
and related environmental science"]<-1
df.TotalDKE$Biological.sciences[df.TotalDKE$Biological.sciences=="Biological
sciences"]<-1
df.TotalDKE$Other.natural.sciences[df.TotalDKE$Other.natural.sciences=="Other
natural sciences"]<-1
df.TotalDKE$Civil.engineering[df.TotalDKE$Civil.engineering=="Civil
engineering"]<-1
df.TotalDKE$"Electrical.engineering,.electronic.engineering,.information.engineering"[df.T
engineering, electronic engineering, information engineering"]<-1
df.TotalDKE$Mechanical.engineering[df.TotalDKE$Mechanical.engineering=="Mechanical
engineering"]<-1
df.TotalDKE$Chemical.engineering[df.TotalDKE$Chemical.engineering=="Chemical
engineering"]<-1
df.TotalDKE$Materials.engineering[df.TotalDKE$Materials.engineering=="Materials
engineering"]<-1
df.TotalDKE$Environmental.engineering[df.TotalDKE$Environmental.engineering=="Environmenta
engineering"]<-1
df.TotalDKE$Industrial.Engineering[df.TotalDKE$Industrial.Engineering=="Industrial
Engineering"]<-1
df.TotalDKE$Nano.technology[df.TotalDKE$Nano.technology=="Nano technology"]<-1
df.TotalDKE$Robotics[df.TotalDKE$Robotics=="Robotics"]<-1
~~~

```
df.TotalDKE$Physics[df.TotalDKE$Physics=="Physics"]<-1
df.TotalDKE$Other[df.TotalDKE$Other=="Other"]<-1

df.TotalDKE$Mathematics[df.TotalDKE$Mathematics== ' ']<-0
df.TotalDKE$Computer.and.information.sciences[df.TotalDKE$Computer.and.information.science
' ']<-0
df.TotalDKE$Physical.sciences[df.TotalDKE$Physical.sciences== ' ']<-0
df.TotalDKE$Chemical.sciences[df.TotalDKE$Chemical.sciences== ' ']<-0
df.TotalDKE$Earth.and.related.environmental.science[df.TotalDKE$Earth.and.related.environm
' ']<-0
df.TotalDKE$Biological.sciences[df.TotalDKE$Biological.sciences== ' ']<-0
df.TotalDKE$Other.natural.sciences[df.TotalDKE$Other.natural.sciences== '
']<-0
df.TotalDKE$Civil.engineering[df.TotalDKE$Civil.engineering== ' ']<-0
df.TotalDKE$"Electrical.engineering,.electronic.engineering,.information.engineering"[df.T
' ']<-0
df.TotalDKE$Mechanical.engineering[df.TotalDKE$Mechanical.engineering== '
']<-0
df.TotalDKE$Chemical.engineering[df.TotalDKE$Chemical.engineering== ' ']<-0
df.TotalDKE$Materials.engineering[df.TotalDKE$Materials.engineering== ' ']<-0
df.TotalDKE$Environmental.engineering[df.TotalDKE$Environmental.engineering==
' ']<-0
df.TotalDKE$Industrial.Engineering[df.TotalDKE$Industrial.Engineering== '
']<-0
df.TotalDKE$Nano.technology[df.TotalDKE$Nano.technology== ' ']<-0
df.TotalDKE$Robotics[df.TotalDKE$Robotics== ' ']<-0
df.TotalDKE$Physics[df.TotalDKE$Physics== ' ']<-0
df.TotalDKE$Other[df.TotalDKE$Other== ' ']<-0

#Breakdown of where the sample currently works
table1 <- table(df.TotalDKE$Current.Field)
table1.prop<-prop.table(table1)
table1<-cbind(table1, table1.prop)
view(table1)

#Breakdown of years of experience
table2<-table(df.TotalDKE$Years.Experience)
table2.prop<-prop.table(table2)
table1<-cbind(table2, table2.prop)
view(table2)

#Best Subject
bestsub.tb<- table(df.summary$Actual.Best.Subject,
df.summary$Estimated.Best.Subject)
view(bestsub.tb)
#write.csv(table1, "currentfield.csv")
#write.csv(table2, "yearsexperience.csv")

#Highest Degree
```

~~~
degree.tb<- table(df.TotalDKE$Degree)
view(degree.tb)


#Descriptives
df.descriptives<-df.TotalDKE[c(3:75)]
describe(df.descriptives)
df.TotalDKE%>%
  summarise_at(vars(Overall.Correct.Answers,
Overall.Correct.Answers.Estimate), funs(mean(., na.rm=T), sd(.,na.rm = T),
n=n(), se=sd(.,na.rm=T)/sqrt(n())))
#write.csv(df.descrpitives, "descriptives.csv")
```



```{r, percentiles}

#Overall
#Overall ACC
df.TotalDKE$OverallQuantile<- with(df.TotalDKE,
cut(df.TotalDKE$Overall.Correct.Answers,
breaks=quantile(df.TotalDKE$Overall.Correct.Answers, probs = seq(0,1,
by=0.25), na.rm=TRUE), include.lowest=TRUE))

df.TotalDKE$Overall.AccPercent= df.summary$Overall.AccPercent<-
df.TotalDKE$Overall.Correct.Answers/81

#by percent
df.TotalDKE$OverallQuartilePerc<- with(df.TotalDKE,
cut(df.TotalDKE$Overall.AccPercent,
breaks=quantile(df.TotalDKE$Overall.AccPercent, probs = seq(0,1, by=0.25),
na.rm=TRUE), include.lowest=TRUE))

#Percentile Rank
df.TotalDKE$OverallPercentileRank<-
percent_rank(df.TotalDKE$Overall.Correct.Answers)
df.TotalDKE$OverallPercentileRank<-round(df.TotalDKE$OverallPercentileRank,
digits=3)

#Quartile of Percentile Rank
df.TotalDKE$OverallQuartileRank<- with(df.TotalDKE,
cut(df.TotalDKE$OverallPercentileRank,
breaks=quantile(df.TotalDKE$OverallPercentileRank, probs = seq(0,1, by=0.25),
na.rm=TRUE), include.lowest=TRUE))


#Grammar
~~~

```
# Grammar Accuracy Quantile
df.TotalDKE$GrammarQuantile<- with(df.TotalDKE, cut(df.TotalDKE$Gram.Acc,
breaks=quantile(df.TotalDKE$Gram.Acc, probs = seq(0,1, by=0.25), na.rm=TRUE),
include.lowest=TRUE))

#by percent
df.TotalDKE$GrammarQuartilePerc<- with(df.TotalDKE,
cut(df.TotalDKE$Gram.AccPerc, breaks=quantile(df.TotalDKE$Gram.AccPerc, probs
= seq(0,1, by=0.25), na.rm=TRUE), include.lowest=TRUE))

#Percentile Rank
df.TotalDKE$GrammarPercentileRank<-percent_rank(df.TotalDKE$Gram.Acc)
df.TotalDKE$GrammarPercentileRank<-round(df.TotalDKE$GrammarPercentileRank,
digits=3)

#Quartile of GrammarPercentile Rank
df.TotalDKE$GrammarQuartileRank<- with(df.TotalDKE,
cut(df.TotalDKE$GrammarPercentileRank,
breaks=quantile(df.TotalDKE$GrammarPercentileRank, probs = seq(0,1, by=0.25),
na.rm=TRUE), include.lowest=TRUE))

#NW
#NW Accuracy Quantile

df.TotalDKE$NWQuantile<- with(df.TotalDKE, cut(df.TotalDKE$NW.Acc,
breaks=quantile(df.TotalDKE$NW.Acc, probs = seq(0,1, by=0.25), na.rm=TRUE),
include.lowest=TRUE))

#by percent
df.TotalDKE$NWQuartilePerc<- with(df.TotalDKE, cut(df.TotalDKE$NW.AccPerc,
breaks=quantile(df.TotalDKE$NW.AccPerc, probs = seq(0,1, by=0.25),
na.rm=TRUE), include.lowest=TRUE))

#Percentile Rank
df.TotalDKE$NWPercentileRank<-percent_rank(df.TotalDKE$NW.Acc)
df.TotalDKE$NWQuartileRank<- with(df.TotalDKE,
cut(df.TotalDKE$NWPercentileRank,
breaks=quantile(df.TotalDKE$NWPercentileRank, probs = seq(0,1, by=0.25),
na.rm=TRUE), include.lowest=TRUE))

#Logic
#Question
df.TotalDKE$LogicQuantile<- with(df.TotalDKE, cut(df.TotalDKE$Logic.Acc,
breaks=quantile(df.TotalDKE$Logic.Acc, probs = seq(0,1, by=0.25), na.rm=TRUE),
include.lowest=TRUE))

#by percent
```

~~~
df.TotalDKE$LogicQuartilePerc<- with(df.TotalDKE,
cut(df.TotalDKE$Logic.AccPerc, breaks=quantile(df.TotalDKE$Logic.AccPerc,
probs = seq(0,1, by=0.25), na.rm=TRUE), include.lowest=TRUE))

#Logic Quartile of Percentile Rank
#by percentile
df.TotalDKE$LogicPercentileRank<-percent_rank(df.TotalDKE$Logic.Acc)
df.TotalDKE$LogicQuartileRank<- with(df.TotalDKE,
cut(df.TotalDKE$LogicPercentileRank,
breaks=quantile(df.TotalDKE$LogicPercentileRank, probs = seq(0,1, by=0.25),
na.rm=TRUE), include.lowest=TRUE))



```



```{r, graphing - global}

#pulling relevant data
df.overallquartilestats<- df.TotalDKE%>%
  group_by(df.TotalDKE$OverallQuantile)%>%
  summarise_at(vars(Overall.Correct.Answers,
Overall.Correct.Answers.Estimate), funs(mean(., na.rm=T), sd(.,na.rm = T),
n=n(), se=sd(.,na.rm=T)/sqrt(n())))
df.overallquartilestats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile
3", "Quartile 4")

df.overallquartilestats$Overall.Correct.Answers.Estimate_mean<-
round(df.overallquartilestats$Overall.Correct.Answers.Estimate_mean, digits=2)
df.overallquartilestats$Overall.Correct.Answers_mean<-
round(df.overallquartilestats$Overall.Correct.Answers_mean, digits=2)

df.logicquartilestats<- df.TotalDKE%>%
  group_by(df.TotalDKE$LogicQuantile)%>%
  summarise_at(vars(Logic.Acc, Post.Log.Est), funs(mean(., na.rm=T),
sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.logicquartilestats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile
3", "Quartile 4")

df.logicquartilestats$Logic.Acc_mean<-
round(df.logicquartilestats$Logic.Acc_mean, digits=2)
df.logicquartilestats$Post.Log.Est_mean<-
round(df.logicquartilestats$Post.Log.Est_mean, digits=2)


df.NWquartilestats<- df.TotalDKE%>%
  group_by(df.TotalDKE$NWQuantile)%>%
~~~

```
  summarise_at(vars(NW.Acc,Post.NW.Est), funs(mean(., na.rm=T), sd(.,na.rm =
T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.NWquartilestats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile 3",
"Quartile 4")


df.NWquartilestats$NW.Acc_mean<-round(df.NWquartilestats$NW.Acc_mean,
digits=2)
df.NWquartilestats$Post.NW.Est_mean<-
round(df.NWquartilestats$Post.NW.Est_mean, digits=2)


df.grammarquartilestats<- df.TotalDKE%>%
  group_by(df.TotalDKE$GrammarQuantile)%>%
  summarise_at(vars(Gram.Acc, Post.Gram.Est), funs(mean(., na.rm=T),
sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.grammarquartilestats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile
3", "Quartile 4")

df.grammarquartilestats$Gram.Acc_mean<-
round(df.grammarquartilestats$Gram.Acc_mean, digits=2)
df.grammarquartilestats$Post.Gram.Est_mean<-
round(df.grammarquartilestats$Post.Gram.Est_mean, digits=2)

   #Overall
  ggplot(df.overallquartilestats, aes(quartiles)) +
  # geom_line(aes(y= Overall.Correct.Answers_mean, group=1,
color="dodgerblue2")) +
  # geom_line(aes(y= Overall.Correct.Answers.Estimate_mean, group=1, color =
"coral1")) +
  geom_errorbar(aes(ymin= df.overallquartilestats$Overall.Correct.Answers_mean
- Overall.Correct.Answers_sd, ymax=
df.overallquartilestats$Overall.Correct.Answers_mean +
Overall.Correct.Answers_sd),   width=0.1) +
  geom_errorbar(aes(ymin=
df.overallquartilestats$Overall.Correct.Answers.Estimate_mean -
Overall.Correct.Answers.Estimate_sd, ymax=
df.overallquartilestats$Overall.Correct.Answers.Estimate_mean +
Overall.Correct.Answers_sd), width=0.1)+
  geom_point(y= df.overallquartilestats$Overall.Correct.Answers_mean, color =
"dodgerblue2")+
  geom_point(y=
df.overallquartilestats$Overall.Correct.Answers.Estimate_mean,color = "coral1"
)+
  # geom_label(aes(quartiles, Overall.Correct.Answers_mean, label =
Overall.Correct.Answers_mean), vjust = -1) +
  # geom_label(aes(quartiles, Overall.Correct.Answers.Estimate_mean, label =
Overall.Correct.Answers.Estimate_mean), vjust = 2) +
  expand_limits(y=c(0,81)) +
  scale_y_continuous(breaks= seq(0,81, by=10)) +
  xlab("Performance Quartile") +
  ylab("Estimated/Achieved score") +
  ggtitle("Overall Estimated Vs. Achieved Accuracy (Out of 81 Task 
Questions)") +
    theme_bw() +
    scale_color_identity(name = "Legend",
                         labels = c("Estimated Accuracy", "Achieved 
Accuracy"),
                         guide = "legend")
  ggsave("Overall_Questions.png")

    #Logic
  ggplot(df.logicquartilestats, aes(quartiles)) +
  # geom_line(aes(y= Logic.Acc_mean, group=1, color="dodgerblue2")) +
  # geom_line(aes(y= Post.Log.Est_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin= df.logicquartilestats$Logic.Acc_mean - Logic.Acc_sd, 
ymax= df.logicquartilestats$Logic.Acc_mean + Logic.Acc_sd), width=0.1) +
  geom_errorbar(aes(ymin= df.logicquartilestats$Post.Log.Est_mean - 
Logic.Acc_sd, ymax= df.logicquartilestats$Post.Log.Est_mean + Logic.Acc_sd), 
width=0.1)+
  #geom_label(aes(quartiles, Logic.Acc_mean, label = Logic.Acc_mean), vjust = 
1.5, hjust=0.5) +
  #geom_label(aes(quartiles, Post.Log.Est_mean, label = Post.Log.Est_mean), 
vjust = -1, hjust= 0.5) +
  geom_point(y= df.logicquartilestats$Post.Log.Est_mean, color = "coral1")+
  geom_point(y= df.logicquartilestats$Logic.Acc_mean, color = "dodgerblue2")+
  expand_limits(y=c(0,29)) +
  scale_y_continuous(breaks= seq(0,29, by=5)) +
  xlab("Performance Quartile") +
  ylab("Estimated/Achieved score") +
  ggtitle("Logic Estimated Vs. Achieved Accuracy (Out of 29 Task Questions)") 
+
    theme_bw() +
    scale_color_identity(name = "Legend",
                         labels = c("Estimated Accuracy", "Achieved 
Accuracy"),
                         guide = "legend")

  ggsave("Logic_Questions.png")

  #NW
  ggplot(df.NWquartilestats, aes(quartiles)) +
  # geom_line(aes(y= NW.Acc_mean, group=1, color="dodgerblue2")) +
  # geom_line(aes(y= Post.NW.Est_mean, group=1, color = "coral1")) +
  geom_text(aes(quartiles, NW.Acc_mean, label = NW.Acc_mean), vjust = -1.2, 
hjust= -0.2) +
```

~~~
  geom_text(aes(quartiles, Post.NW.Est_mean, label = Post.NW.Est_mean), vjust
= 1.5, hjust= - 0.4) +
  geom_errorbar(aes(ymin= df.NWquartilestats$NW.Acc_mean - NW.Acc_sd, ymax=
df.NWquartilestats$NW.Acc_mean + NW.Acc_sd), width=0.1) +
  geom_errorbar(aes(ymin= df.NWquartilestats$Post.NW.Est_mean - NW.Acc_sd,
ymax= df.NWquartilestats$Post.NW.Est_mean + NW.Acc_sd), width=0.1)+
  geom_point(y= df.NWquartilestats$NW.Acc_mean, color = "dodgerblue2")+
  geom_point(y= df.NWquartilestats$Post.NW.Est_mean, color = "coral1")+
  expand_limits(y=c(0,27)) +
  scale_y_continuous(breaks= seq(0,27, by=5)) +
  xlab("Performance Quartile") +
  ylab("Estimated/Achieved score") +
  ggtitle("NW Estimated Vs. Achieved Accuracy (Out of 27 Questions)") +
    theme_bw() +
    scale_color_identity(name = "Legend",
                          labels = c("Estimated Accuracy", "Achieved
Accuracy"),
                          guide = "legend")
   ggsave("NW_Questions.png")

  #Grammar
  ggplot(df.grammarquartilestats, aes(quartiles)) +
  # geom_line(aes(y= Gram.Acc_mean, group=1, color="dodgerblue2")) +
  # geom_line(aes(y= Post.Gram.Est_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin= df.grammarquartilestats$Gram.Acc_mean - Gram.Acc_sd,
ymax= df.grammarquartilestats$Gram.Acc_mean + Gram.Acc_sd), width=0.1) +
  geom_errorbar(aes(ymin= df.grammarquartilestats$Post.Gram.Est_mean -
Gram.Acc_sd, ymax= df.grammarquartilestats$Post.Gram.Est_mean + Gram.Acc_sd),
width=0.1)+
  geom_point(y= df.grammarquartilestats$Gram.Acc_mean, color = "dodgerblue2")+
  geom_point(y= df.grammarquartilestats$Post.Gram.Est_mean, color = "coral1")+
  # geom_text(aes(quartiles, Gram.Acc_mean, label = Gram.Acc_mean), vjust =
-0.6, hjust=1.2) +
  # geom_text(aes(quartiles, Post.Gram.Est_mean, label = Post.Gram.Est_mean),
vjust = 1.5, hjust=-0.3) +
  expand_limits(y=c(0,25)) +
  scale_y_continuous(breaks= seq(0,25, by=5)) +
  xlab("Performance Quartile") +
  ylab("Estimated/Achieved score") +
  ggtitle("Grammar Estimated Vs. Achieved Accuracy (Out of 25 Task
Questions)") +
    theme_bw()+
    scale_color_identity(name = "Legend",
                          labels = c("Estimated Accuracy", "Achieved
Accuracy"),
                          guide = "legend")
ggsave("Grammar_Questions.png")
```
~~~

~~~
```{r, graphing percentiles (overplacing)}

df.TotalDKE$OverallEstimatedPercentile<- ((df.TotalDKE$Post.LogSandian +
df.TotalDKE$Post.NWSandian + df.TotalDKE$Post.GramSandian) / 3)/100
df.TotalDKE$GrammarEstimatedPercentile<- (df.TotalDKE$Post.GramSandian) /100
df.TotalDKE$LogicEstimatedPercentile<- (df.TotalDKE$Post.LogSandian) /100
df.TotalDKE$NWEstimatedPercentile<- (df.TotalDKE$Post.NWSandian) /100

# OverallPercentileRank is the achieved percentile
# OverallQuartileRank is the quartile breakdowns

  df.overallpercentilestats<- df.TotalDKE%>%
  group_by(OverallQuartileRank)%>%
  summarise_at(vars(OverallPercentileRank, OverallEstimatedPercentile),
funs(mean(., na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.overallpercentilestats$quartiles<- c("Quartile 1", "Quartile 2",
"Quartile 3", "Quartile 4")

  df.overallpercentilestats$OverallPercentileRank_mean<-
round(df.overallpercentilestats$OverallPercentileRank_mean, digits=2)
  df.overallpercentilestats$OverallEstimatedPercentile_mean<-
round(df.overallpercentilestats$OverallEstimatedPercentile_mean, digits=2)

  df.logicpercentilestats<- df.TotalDKE%>%
  group_by(LogicQuartileRank)%>%
  summarise_at(vars(LogicPercentileRank, LogicEstimatedPercentile),
funs(mean(., na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.logicpercentilestats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile
3", "Quartile 4")

  df.logicpercentilestats$LogicPercentileRank_mean<-
round(df.logicpercentilestats$LogicPercentileRank_mean, digits=2)
  df.logicpercentilestats$LogicEstimatedPercentile_mean<-
round(df.logicpercentilestats$LogicEstimatedPercentile_mean, digits=2)

  df.grammarpercentilestats<- df.TotalDKE%>%
  group_by(GrammarQuartileRank)%>%
  summarise_at(vars(GrammarPercentileRank, GrammarEstimatedPercentile),
funs(mean(., na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.grammarpercentilestats$quartiles<- c("Quartile 1", "Quartile 2",
"Quartile 3", "Quartile 4")

  df.grammarpercentilestats$GrammarPercentileRank_mean<-
round(df.grammarpercentilestats$GrammarPercentileRank_mean, digits=2)
  df.grammarpercentilestats$GrammarEstimatedPercentile_mean<-
round(df.grammarpercentilestats$GrammarEstimatedPercentile_mean, digits=2)
~~~

```
  df.NWpercentilestats<- df.TotalDKE%>%
  group_by(NWQuartileRank)%>%
  summarise_at(vars(NWPercentileRank, NWEstimatedPercentile), funs(mean(.,
na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.NWpercentilestats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile 3",
"Quartile 4")

  df.NWpercentilestats$NWPercentileRank_mean<-
round(df.NWpercentilestats$NWPercentileRank_mean, digits=2)
  df.NWpercentilestats$NWEstimatedPercentile_mean<-
round(df.NWpercentilestats$NWEstimatedPercentile_mean, digits=2)

  #Overall
  ggplot(df.overallpercentilestats, aes(quartiles)) +
  # geom_line(aes(y= OverallPercentileRank_mean, group=1,
color="dodgerblue2")) +
  # geom_line(aes(y= OverallEstimatedPercentile_mean, group=1, color =
"coral1")) +
  geom_errorbar(aes(ymin= df.overallpercentilestats$OverallPercentileRank_mean
- OverallPercentileRank_sd, ymax=
df.overallpercentilestats$OverallPercentileRank_mean +
OverallPercentileRank_sd), width=0.1) +
  geom_errorbar(aes(ymin=
df.overallpercentilestats$OverallEstimatedPercentile_mean -
OverallPercentileRank_sd, ymax=
df.overallpercentilestats$OverallEstimatedPercentile_mean +
OverallPercentileRank_sd), width=0.1)+
  geom_point(y= df.overallpercentilestats$OverallPercentileRank_mean, color =
"dodgerblue2")+
  geom_point(y= df.overallpercentilestats$OverallEstimatedPercentile_mean,
color = "coral1")+
  # geom_label(aes(quartiles, OverallPercentileRank_mean, label =
OverallPercentileRank_mean), vjust = -0.5) +
  # geom_label(aes(quartiles, OverallEstimatedPercentile_mean, label =
OverallEstimatedPercentile_mean), vjust = 1, hjust= -0.2) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits= c(0,1)) +
  xlab("Performance Quartile") +
  ylab("Estimated/Percentile") +
  ggtitle("Comparison Between Estimated Score Percentile and Achieved
Percentile (Overall)") +
    theme_bw()+
    scale_color_identity(name = "Legend",
                         labels = c("Estimated Accuracy", "Achieved
Accuracy"),
                         guide = "legend")
 ggsave("Overall_Percentile.png")
```

```
  #Logic
  ggplot(df.logicpercentilestats, aes(quartiles)) +
  geom_line(aes(y= LogicPercentileRank_mean, group=1, color="dodgerblue2")) +
  geom_line(aes(y= LogicEstimatedPercentile_mean, group=1, color = "coral1"))
+
  geom_errorbar(aes(ymin= df.logicpercentilestats$LogicPercentileRank_mean -
LogicPercentileRank_sd, ymax= df.logicpercentilestats$LogicPercentileRank_mean
+ LogicPercentileRank_sd), width=0.1) +
geom_errorbar(aes(ymin= df.logicpercentilestats$LogicEstimatedPercentile_mean
- LogicPercentileRank_sd, ymax=
df.logicpercentilestats$LogicEstimatedPercentile_mean +
LogicPercentileRank_sd), width=0.1)+
  geom_point(y= df.logicpercentilestats$LogicPercentileRank_mean)+
  geom_point(y= df.logicpercentilestats$LogicEstimatedPercentile_mean)+
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits= c(0,1))+
  geom_text(aes(quartiles,  LogicPercentileRank_mean, label =
LogicPercentileRank_mean), vjust = 1, hjust= -0.7) +
  geom_text(aes(quartiles, LogicEstimatedPercentile_mean, label =
LogicEstimatedPercentile_mean), vjust = -0.5, hjust= 1.2) +
  xlab("Performance Quartile") +
  ylab("Estimated/Achieved Percentile") +
  ggtitle("Comparison Between Estimated Score Percentile and Achieved
Percentile (Logic)") +
    theme_bw()+
    scale_color_identity(name = "Legend",
                          labels = c("Estimated Accuracy", "Achieved
Accuracy"),
                          guide = "legend")

  ggsave("Logic_Percentile.png")

  #Grammar
  ggplot(df.grammarpercentilestats, aes(quartiles)) +
  geom_line(aes(y= GrammarPercentileRank_mean, group=1, color="dodgerblue2"))
+
  geom_line(aes(y= GrammarEstimatedPercentile_mean, group=1, color =
"coral1")) +
  geom_errorbar(aes(ymin= df.grammarpercentilestats$GrammarPercentileRank_mean
- GrammarPercentileRank_sd, ymax=
df.grammarpercentilestats$GrammarPercentileRank_mean +
GrammarPercentileRank_sd), width=0.1) +
 geom_errorbar(aes(ymin=
df.grammarpercentilestats$GrammarEstimatedPercentile_mean -
GrammarPercentileRank_sd, ymax=
df.grammarpercentilestats$GrammarEstimatedPercentile_mean +
GrammarPercentileRank_sd), width=0.1)+
  geom_point(y= df.grammarpercentilestats$GrammarPercentileRank_mean)+
  geom_point(y= df.grammarpercentilestats$GrammarEstimatedPercentile_mean)+
```

~~~
  geom_text(aes(quartiles,  GrammarPercentileRank_mean, label =
GrammarPercentileRank_mean), vjust = -.5, hjust = 1.2) +
  geom_text(aes(quartiles, GrammarEstimatedPercentile_mean, label =
GrammarEstimatedPercentile_mean), vjust = 1.4, hjust= -0.2) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits= c(0,1))+
  xlab("Performance Quartile") +
  ylab("Estimated/Achieved Percentile") +
  ggtitle("Comparison Between Estimated Score Percentile & Achieved Percentile
(Grammar)") +
    theme_bw()+
    scale_color_identity(name = "Legend",
                         labels = c("Estimated Accuracy", "Achieved
Accuracy"),
                         guide = "legend")

  ggsave("Grammar_Percentile.png")


  #NW
  ggplot(df.NWpercentilestats, aes(quartiles)) +
  geom_line(aes(y= NWPercentileRank_mean, group=1, color="dodgerblue2")) +
  geom_line(aes(y= NWEstimatedPercentile_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin= df.NWpercentilestats$NWPercentileRank_mean -
NWPercentileRank_sd, ymax= df.NWpercentilestats$NWPercentileRank_mean +
NWPercentileRank_sd), width=0.1) +
  geom_errorbar(aes(ymin= df.NWpercentilestats$NWEstimatedPercentile_mean -
NWPercentileRank_sd, ymax= df.NWpercentilestats$NWEstimatedPercentile_mean +
NWPercentileRank_sd), width=0.1)+
  geom_point(y= df.NWpercentilestats$NWPercentileRank_mean)+
  geom_point(y= df.NWpercentilestats$NWEstimatedPercentile_mean)+
  geom_text(aes(quartiles,  NWPercentileRank_mean, label =
NWPercentileRank_mean), vjust= 1.5, hjust = -0.2) +
  geom_text(aes(quartiles, NWEstimatedPercentile_mean, label =
NWEstimatedPercentile_mean), vjust = -0.5, hjust= 1.2) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits= c(0,1))+
  xlab("Performance Quartile") +
  ylab("Estimated/Achieved Percentile") +
  ggtitle("Comparison Between Estimated Score Percentile and Achieved
Percentile (NW)") +
    theme_bw()+
    scale_color_identity(name = "Legend",
                         labels = c("Estimated Accuracy", "Achieved
Accuracy"),
                         guide = "legend")
  ggsave("NW_Percentile.png")
```

```{r, graphing - overprecision}
~~~

```
df.TotalDKE$OverallOverPrecision <-df.TotalDKE$Overall.Confidence.Average -
df.TotalDKE$Overall.AccPercent
df.TotalDKE$NWOverPrecision <-df.TotalDKE$NW.Confidence.Avg -
df.TotalDKE$NW.AccPerc
df.TotalDKE$GrammarOverPrecision <-df.TotalDKE$Grammar.Confidence.Avg -
df.TotalDKE$Gram.AccPerc
df.TotalDKE$LogicOverPrecision <-df.TotalDKE$Logic.Confidence.Avg -
df.TotalDKE$Logic.AccPerc

# OverallPercentileRank is the achieved percentile
# OverallQuartileRank is the quartile breakdowns

  df.overalloverprecisionstats<- df.TotalDKE%>%
  group_by(OverallQuartilePerc)%>%
  summarise_at(vars(Overall.Confidence.Average, OverallAccPerc), funs(mean(.,
na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.overalloverprecisionstats$quartiles<- c("Quartile 1", "Quartile 2",
"Quartile 3", "Quartile 4")

df.overalloverprecisionstats$Overall.Confidence.Average_mean<-
round(df.overalloverprecisionstats$Overall.Confidence.Average_mean, digits=2)
df.overalloverprecisionstats$OverallAccPerc_mean<-
round(df.overalloverprecisionstats$OverallAccPerc_mean, digits=2)

  df.logicoverprecisionstats<- df.TotalDKE%>%
  group_by(LogicQuartilePerc)%>%
  summarise_at(vars(Logic.Confidence.Avg, Logic.AccPerc), funs(mean(.,
na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.logicoverprecisionstats$quartiles<- c("Quartile 1", "Quartile 2",
"Quartile 3", "Quartile 4")

  df.logicoverprecisionstats$Logic.Confidence.Avg_mean<-
round(df.logicoverprecisionstats$Logic.Confidence.Avg_mean, digits=2)
  df.logicoverprecisionstats$Logic.AccPerc_mean<-
round(df.logicoverprecisionstats$Logic.AccPerc_mean, digits=2)

  df.grammaroverprecisionstats<- df.TotalDKE%>%
  group_by(GrammarQuartilePerc)%>%
  summarise_at(vars(Grammar.Confidence.Avg, Gram.AccPerc), funs(mean(.,
na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.grammaroverprecisionstats$quartiles<- c("Quartile 1", "Quartile 2",
"Quartile 3", "Quartile 4")

  df.grammaroverprecisionstats$Grammar.Confidence.Avg_mean<-
round(df.grammaroverprecisionstats$Grammar.Confidence.Avg_mean, digits=2)
  df.grammaroverprecisionstats$Gram.AccPerc_mean<-
round(df.grammaroverprecisionstats$Gram.AccPerc_mean, digits=2)
```

```
  df.NWoverprecisionstats<- df.TotalDKE%>%
  group_by(NWQuartileRank)%>%
  summarise_at(vars(NW.Confidence.Avg, NW.AccPerc), funs(mean(., na.rm=T),
sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
  df.NWoverprecisionstats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile
3", "Quartile 4")

  df.NWoverprecisionstats$NW.Confidence.Avg_mean<-
round(df.NWoverprecisionstats$NW.Confidence.Avg_mean, digits=2)
  df.NWoverprecisionstats$NW.AccPerc_mean<-
round(df.NWoverprecisionstats$NW.AccPerc_mean, digits=2)

  #Overall
  ggplot(df.overalloverprecisionstats, aes(quartiles)) +
  geom_line(aes(y= OverallAccPerc_mean, group=1, color="dodgerblue2")) +
  geom_line(aes(y= Overall.Confidence.Average_mean, group=1, color =
"coral1")) +
  geom_errorbar(aes(ymin= df.overalloverprecisionstats$OverallAccPerc_mean -
OverallAccPerc_sd, ymax= df.overalloverprecisionstats$OverallAccPerc_mean +
OverallAccPerc_sd), width=0.1) +
  geom_errorbar(aes(ymin=
df.overalloverprecisionstats$Overall.Confidence.Average_mean -
OverallAccPerc_sd, ymax=
df.overalloverprecisionstats$Overall.Confidence.Average_mean +
OverallAccPerc_sd), width=0.1)+
  geom_point(y= df.overalloverprecisionstats$OverallAccPerc_mean)+
  geom_point(y= df.overalloverprecisionstats$Overall.Confidence.Average_mean)+
  geom_text(aes(quartiles, OverallAccPerc_mean, label = OverallAccPerc_mean),
vjust = 2, hjust= -0.2) +
  geom_text (aes(quartiles, Overall.Confidence.Average_mean, label =
Overall.Confidence.Average_mean), vjust = -1, hjust= -0.2) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits= c(0,1)) +
  xlab("Performance Quartile") +
  ylab("Average Confidence / Accuracy (in %)") +
  ggtitle("Item-by-Item Comparison Between Estimated and Achieved Accuracy
(Overall)") +
    theme_bw()+
    scale_color_identity(name = "Legend",
                         labels = c("Estimated Accuracy", "Achieved
Accuracy"),
                         guide = "legend")
 ggsave("Overall_Overprecsion.png")


  #Logic
  ggplot(df.logicoverprecisionstats, aes(quartiles)) +
  geom_line(aes(y= Logic.AccPerc_mean, group=1, color="dodgerblue2")) +
  geom_line(aes(y= Logic.Confidence.Avg_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin= df.logicoverprecisionstats$Logic.AccPerc_mean - 
Logic.AccPerc_sd, ymax= df.logicoverprecisionstats$Logic.AccPerc_mean + 
Logic.AccPerc_sd), width=0.1) +
geom_errorbar(aes(ymin= df.logicoverprecisionstats$Logic.Confidence.Avg_mean - 
Logic.AccPerc_sd, ymax= df.logicoverprecisionstats$Logic.Confidence.Avg_mean + 
Logic.AccPerc_sd), width=0.1)+
  geom_point(y= df.logicoverprecisionstats$Logic.AccPerc_mean)+
  geom_point(y= df.logicoverprecisionstats$Logic.Confidence.Avg_mean)+
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits= c(0,1))+
  geom_text(aes(quartiles,  Logic.AccPerc_mean, label =  Logic.AccPerc_mean), 
vjust = 0.9, hjust= -0.3) +
  geom_text(aes(quartiles, Logic.Confidence.Avg_mean, label = 
Logic.Confidence.Avg_mean), vjust = -0.7, hjust= -0.2) +
  xlab("Performance Quartile") +
  ylab("Average Confidence / Accuracy") +
  ggtitle("Item-by-Item Comparison Between Estimated and Achieved Accuracy 
(Logic)") +
    theme_bw()+
    scale_color_identity(name = "Legend",
                          labels = c("Estimated Accuracy", "Achieved 
Accuracy"),
                          guide = "legend")

  ggsave("Logic_Overprecision.png")

  #Grammar
  ggplot(df.grammaroverprecisionstats, aes(quartiles)) +
  geom_line(aes(y= Gram.AccPerc_mean, group=1, color="dodgerblue2")) +
  geom_line(aes(y= Grammar.Confidence.Avg_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin= df.grammaroverprecisionstats$Gram.AccPerc_mean - 
Gram.AccPerc_sd, ymax= df.grammaroverprecisionstats$Gram.AccPerc_mean + 
Gram.AccPerc_sd), width=0.1) +
 geom_errorbar(aes(ymin= 
df.grammaroverprecisionstats$Grammar.Confidence.Avg_mean - Gram.AccPerc_sd, 
ymax= df.grammaroverprecisionstats$Grammar.Confidence.Avg_mean + 
Gram.AccPerc_sd), width=0.1)+
  geom_point(y= df.grammaroverprecisionstats$Gram.AccPerc_mean)+
  geom_point(y= df.grammaroverprecisionstats$Grammar.Confidence.Avg_mean)+
  geom_text(aes(quartiles,  Gram.AccPerc_mean, label =  Gram.AccPerc_mean), 
vjust = 0.9, hjust= -0.3) +
  geom_text(aes(quartiles, Grammar.Confidence.Avg_mean, label = 
Grammar.Confidence.Avg_mean), vjust = -0.7, hjust= -0.2) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits= c(0,1))+
  xlab("Performance Quartile") +
  ylab("Average Confidence / Accuracy") +
  ggtitle("Item-by-Item Comparison Between Estimated and Achieved Accuracy 
(Grammar)") +
    theme_bw()+
```

~~~~
    scale_color_identity(name = "Legend",
                          labels = c("Estimated Accuracy", "Achieved
Accuracy"),
                          guide = "legend")

  ggsave("Grammar_Overprecision.png")


  #NW
  ggplot(df.NWoverprecisionstats, aes(quartiles)) +
  geom_line(aes(y= NW.AccPerc_mean, group=1, color="dodgerblue2")) +
  geom_line(aes(y= NW.Confidence.Avg_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin= df.NWoverprecisionstats$NW.AccPerc_mean -
NW.AccPerc_sd, ymax= df.NWoverprecisionstats$NW.AccPerc_mean + NW.AccPerc_sd),
width=0.1) +
geom_errorbar(aes(ymin= df.NWoverprecisionstats$NW.Confidence.Avg_mean -
NW.AccPerc_sd, ymax= df.NWoverprecisionstats$NW.Confidence.Avg_mean +
NW.AccPerc_sd), width=0.1)+
  geom_point(y= df.NWoverprecisionstats$NW.AccPerc_mean)+
  geom_point(y= df.NWoverprecisionstats$NW.Confidence.Avg_mean)+
  geom_text(aes(quartiles,  NW.AccPerc_mean, label =  NW.AccPerc_mean), vjust
= 0.9, hjust= -0.3) +
  geom_text(aes(quartiles, NW.Confidence.Avg_mean, label =
NW.Confidence.Avg_mean), vjust = -1.4, hjust= -0.2) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits= c(0,1))+
  xlab("Performance Quartile") +
  ylab("Average Confidence / Accuracy") +
  ggtitle("Item-by-Item Comparison Between Estimated and Achieved Accuracy
(NW)") +
    theme_bw()+
    scale_color_identity(name = "Legend",
                          labels = c("Estimated Accuracy", "Achieved
Accuracy"),
                          guide = "legend")
  ggsave("NW_Overprecision.png")
```

```{r, outcome measure 1 - overprecision}
#Overprecision calculation
#Overprecision – excessive confidence in the accuracy of our beliefs
#It is the difference between the mean subjective probability and mean
accuracy (proportion of correct answers) of judgments in a task
#Positive = overprecise
# Negative = Underconfident
#0 = the person is considered to be well-calibrated.

df.TotalDKE$Overall.AccPercent<- df.TotalDKE$Overall.Correct.Answers / 81
df.TotalDKE$Overall.Confidence.Average<- (df.TotalDKE$Logic.Confidence.Avg +
df.TotalDKE$Grammar.Confidence.Avg + df.TotalDKE$NW.Confidence.Avg) / 3
~~~~

~~~
df.TotalDKE$Overall.Confidence.Average =
df.TotalDKE$Overall.Confidence.Average

df.TotalDKE$OverallOverPrecision <-df.TotalDKE$Overall.Confidence.Average -
df.TotalDKE$Overall.AccPercent
df.TotalDKE$NWOverPrecision <-df.TotalDKE$NW.Confidence.Avg -
df.TotalDKE$NW.AccPerc
df.TotalDKE$GrammarOverPrecision <-df.TotalDKE$Grammar.Confidence.Avg -
df.TotalDKE$Gram.AccPerc
df.TotalDKE$LogicOverPrecision <-df.TotalDKE$Logic.Confidence.Avg -
df.TotalDKE$Logic.AccPerc

df.TotalDKE$OverallOverestPerc<-df.TotalDKE$Overall.Overestimation.Score/100

cor(df.TotalDKE$OverallOverestPerc, df.TotalDKE$OverallOverPrecision)
#Corr is .84, meaning that these are almost the same thing. Overprecision is
item-by-item, while overest is global.

```


```{r, individual confidence means and medians}

df.conf<- df.TotalDKE[c(129:157, 183:207,235:261)]

# df.TotalDKE$MeanHighConfAcc0<- rowsums(df.TotalDKE[c(100:261)] >=
df.TotalDKE$OverallAvgConf && df.TotalDKE[c(100:261)])
#
# df.confcounts0$MeanHighConfAcc0<-rowSums(df.confcounts[c(1:35)] >=
df.confcounts0$OverallAvgConf)

#
#
# df.confcounts$MeanHighConfAcc0<-rowSums(df.confcounts[c(2:12, 24:34, 46:56)]
>= df.confcounts$OverallAvgConf)
#
# df.confcounts$MeanHighConfAcc0<- sum(df.confcounts[, c(2:12, 24:34, 46:56)]
>= "OverallAvgConf")

#Overall
df.TotalDKE$OverallConfMedian<- apply(df.conf[c(1:81)], 1, median, na.rm =T)
df.TotalDKE$OverallConfMedian<-round(df.TotalDKE$OverallConfMedian, digits =
1)

#Logic
df.TotalDKE$LogicConfMedian<- apply(df.conf[1:29], 1, median, na.rm =T)
df.confcounts$LogicAvgConf <- round(df.TotalDKE$Logic.Confidence.Avg, digits =
1)
~~~

~~~~
#Grammar
df.TotalDKE$GrammarConfMedian<- apply(df.conf[30:54], 1, median, na.rm =T)
df.confcounts$GrammarAvgConf <- round(df.TotalDKE$Grammar.Confidence.Avg,
digits = 1)


#NW
df.TotalDKE$NWConfMedian<- apply(df.conf[55:81], 1, median, na.rm =T)
df.confcounts$NWAvgConf <- round(df.TotalDKE$NW.Confidence.Avg, digits = 1)


#=COUNTIF($IT2:$JT2,"<"&$KA2)
# df.conf$subject<- df.TotalDKE$Subject


```

```{r, subjective probability overconfidences}

#Turn the task scores and confs into matrices

logic.matrix <- as.matrix(df.TotalDKE[c(100:128)])
logicConf.matrix <- as.matrix(df.TotalDKE[c(129:157)])

grammar.matrix <- as.matrix(df.TotalDKE[c(158:182)])
grammarConf.matrix <- as.matrix(df.TotalDKE[c(183:207)])

NW.matrix <- as.matrix(df.TotalDKE[c(208:234)])
NWConf.matrix <- as.matrix(df.TotalDKE[c(235:261)])

Overall.matrix <-as.matrix(df.TotalDKE[c(100:128, 158:182,208:234 )])
OverallConf.matrix <- as.matrix(df.TotalDKE[c(129:157, 183:207, 235:261)])

# Subtract (stated confidence - score / task score)

df.LogicOverConf<-as.data.frame(logicConf.matrix - logic.matrix)
df.GrammarOverConf<-as.data.frame(grammarConf.matrix - grammar.matrix)
df.NWOverConf<-as.data.frame(NWConf.matrix - NW.matrix)
df.OverallOverConf<-as.data.frame(OverallConf.matrix - Overall.matrix)

df.TotalDKE$Logicsubprobover<- round(apply(df.LogicOverConf, 1, sum,
na.rm=TRUE) / 29, digits = 2)
df.TotalDKE$Grammarsubprobover<- round((apply(df.GrammarOverConf, 1, sum,
na.rm= TRUE)) / 25, digits = 2)
df.TotalDKE$NWsubprobover<- round(apply(df.NWOverConf, 1, sum, na.rm= TRUE) /
27, digits = 2)
df.TotalDKE$Overallsubprobover<- round(apply(df.OverallOverConf, 1, sum,
na.rm= TRUE) / 81, digits = 2)
~~~~

```
# #fix for e in grammar outupt
# write.csv(df.TotalDKE, "grammar_fix.csv")
# #go into excel and change grammar to number, which should change to 2
digits.
# #re-upload
# df.TotalDKE<-read.csv("grammar_fix.csv")

#change everything to 2 digits for readability
df.TotalDKE$Logicsubprobover<-round(df.TotalDKE$Logicsubprobover, digits = 2)
df.TotalDKE$Grammarsubprobover<-round(df.TotalDKE$Grammarsubprobover, digits =
2)
df.TotalDKE$NWsubprobover<-round(df.TotalDKE$NWsubprobover, digits = 2)
df.TotalDKE$Overallsubprobover<-round(df.TotalDKE$Overallsubprobover, digits =
2)

df.TotalDKE$OverallAccPerc <- df.TotalDKE$Overall.Correct.Answers/81

#Correlation
cor.test(df.TotalDKE$OverallOverestPerc, df.TotalDKE$Overallsubprobover,
method = c("pearson"))
cor.test(df.TotalDKE$OverallAccPerc, df.TotalDKE$Overallsubprobover, method =
c("pearson"))
cor.test(df.TotalDKE$OverallAccPerc, df.TotalDKE$Overall.Overestimation.Score,
method = c("pearson"))

cor.test(df.TotalDKE$OverallAccPerc, df.TotalDKE$Overall.Overestimation.Score,
method = c("pearson"))

cor.test(df.TotalDKE$Intuitive, df.TotalDKE$Debugging.Strategies, method
=c("pearson"))

cor.test(df.TotalDKE$Logic.Acc, df.TotalDKE$Debugging.Strategies, method
=c("pearson"))

cor.test(df.TotalDKE$Spontaneous, df.TotalDKE$Openness, method =c("pearson"))

df.TotalDKE$LogicOverest<- df.TotalDKE$Post.Log.Est - df.TotalDKE$Logic.Acc
df.TotalDKE$GrammarOverest<-df.TotalDKE$Post.Gram.Est - df.TotalDKE$Gram.Acc
df.TotalDKE$NWOverest<-df.TotalDKE$Post.NW.Est - df.TotalDKE$NW.Acc
#Logic

#Correlations between all of the tasks

cor.test(df.TotalDKE$LogicOverest, df.TotalDKE$Logicsubprobover)
cor.test(df.TotalDKE$GrammarOverest, df.TotalDKE$Grammarsubprobover)
cor.test(df.TotalDKE$NWOverest, df.TotalDKE$NWsubprobover)

cor.test(df.TotalDKE$OverallOverplacing,
df.TotalDKE$Overall.Overestimation.Score, method=c("pearson"))
```

~~~
cor.test(df.TotalDKE$OverallOverplacing, df.TotalDKE$OverallOverPrecision,
method=c("pearson"))
cor.test(df.TotalDKE$Overall.Overestimation.Score,
df.TotalDKE$OverallOverPrecision, method=c("pearson"))

cor.test(df.TotalDKE$Overall.Overestimation.Score,
df.TotalDKE$OverallOverPrecision)

# Correlations between difficulty and the tasks

cor.test(df.TotalDKE$Post.Gram.Dif, df.TotalDKE$Grammar.Overest)
cor.test(df.TotalDKE$Post.Log.Dif, df.TotalDKE$Logic.Overest)
cor.test(df.TotalDKE$Post.NW.Dif, df.TotalDKE$NW.Overest)
cor.test(df.TotalDKE$OverallPostDif, df.TotalDKE$Overall.Overestimation.Score)
```


```{r, over/understimation in more than 2 domains}
  df.calibration<-read.csv("overestimation.across.domains.csv")
  df.calibration<-df.calibration[-c(1)]


calibrationmetrics$TwoDomainsUnderest<- sum(df.calibration$TwoDomainsUnderest)
calibrationmetrics$ThreeDomainsUnderest<-
sum(df.calibration$ThreeDomainsUnderest)
calibrationmetrics$TwoDomainsCalibrated<-
sum(df.calibration$TwoDomainsCalibrated)
calibrationmetrics$TwoDomainsOver<-sum(df.calibration$TwoDomainsOver)
calibrationmetrics$ThreeDomainsOver<-sum(df.calibration$ThreeDomainsOver)
calibrationmetrics$AllDomainsDifferent<-
sum(df.calibration$AllDomainsDifferent)

calibrationmetrics$TwoDomainsPerc<-
round(sum(df.calibration$TwoDomainsUnderest)/93, digits=3)
calibrationmetrics$TwoDomainsUnderestPerc<-
round(sum(df.calibration$TwoDomainsUnderest)/93, digits = 3)
calibrationmetrics$ThreeDomainsUnderestPerc<-
round(sum(df.calibration$ThreeDomainsUnderest)/93, digits = 3)
calibrationmetrics$TwoDomainsCalibratedPerc<-
round(sum(df.calibration$TwoDomainsCalibrated)/93, digits = 3)
calibrationmetrics$TwoDomainsOverPerc<-
round(sum(df.calibration$TwoDomainsOver)/93, digits = 3)
calibrationmetrics$ThreeDomainsOverPerc<-
round(sum(df.calibration$ThreeDomainsOver)/93, digits = 3)
calibrationmetrics$AllDomainsDifferentPerc<-
round(sum(df.calibration$AllDomainsDifferent)/93, digits = 3)

calibrationmetrics<-table(calibrationmetrics)
~~~

~~~
View(calibrationmetrics)

```

```{r, correlations}
#http://www.sthda.com/english/wiki/correlation-matrix-a-quick-start-guide-to-analyze-format-and-visualize-a-correlation-matrix-using-r-software
#http://www.sthda.com/english/wiki/elegant-correlation-table-using-xtable-r-package
#

df.correlation<-df.TotalDKE[c(3:10,
17:34,39,43:46,51,54,56:58,63,66,68:70,73:75, 96, 265,269, 273, 277, 279:282,
284:287,291:294, 296:299)]

res <- cor(df.correlation, use = "complete.obs")
round(res, 2)

res2 <- rcorr(as.matrix(df.correlation))
#coeffs
res2$r
# Extract p-values
res2$P

#function to organize the output
flattenCorrMatrix <- function(cormat, pmat) {
  ut <- upper.tri(cormat)
  data.frame(
    row = rownames(cormat)[row(cormat)[ut]],
    column = rownames(cormat)[col(cormat)[ut]],
    cor  =(cormat)[ut],
    p = pmat[ut]
    )
}

df.corr<-flattenCorrMatrix(res2$r, res2$P)

write.csv(df.corr, "correlations.august2020.csv")

#Partial correlation matrix
partialcor<-cor2pcor(res)
write.csv(res, "res.csv")
write.csv(partialcor, "partialcor.csv")

pcor<-pcor(df.correlation)
lapply(pcor, function(x) write.table( data.frame(pcor), 'partialcorrel.csv'  ,
append= T, sep=',' ))

#partial correlations
~~~

~~~
#Accuracy and estimation
pcor.test(df.TotalDKE$NW.Acc, df.TotalDKE$Post.NW.Est,
df.TotalDKE$Post.NW.Dif, method = c("pearson"))
pcor.test(df.TotalDKE$Gram.Acc, df.TotalDKE$Post.Gram.Est,
df.TotalDKE$Post.Gram.Dif, method = c("pearson"))
pcor.test(df.TotalDKE$Logic.Acc, df.TotalDKE$Post.Log.Est,
df.TotalDKE$Post.Log.Dif, method = c("pearson"))

#Accuracy and Overest
pcor.test(df.TotalDKE$NW.Acc, df.TotalDKE$NW.Overest, df.TotalDKE$Post.NW.Dif,
method = c("pearson"))
pcor.test(df.TotalDKE$Gram.Acc, df.TotalDKE$Grammar.Overest,
df.TotalDKE$Post.Gram.Dif, method = c("pearson"))
pcor.test(df.TotalDKE$Logic.Acc, df.TotalDKE$Logic.Overest,
df.TotalDKE$Post.Log.Dif, method = c("pearson"))

#Overest and Difficulty
pcor.test(df.TotalDKE$NW.Overest, df.TotalDKE$Post.NW.Dif, df.TotalDKE$NW.Acc,
method = c("pearson"))
pcor.test(df.TotalDKE$Grammar.Overest, df.TotalDKE$Post.Gram.Dif,
df.TotalDKE$Gram.Acc, method = c("pearson"))
pcor.test(df.TotalDKE$Logic.Overest, df.TotalDKE$Post.Log.Dif,
df.TotalDKE$Logic.Acc, method = c("pearson"))

#Individual traits

pcor.test(df.TotalDKE$Logic.Overest, df.TotalDKE$Openness,
df.TotalDKE$Logic.Acc,  method = c("pearson"))
pcor.test(df.TotalDKE$Logic.Overest, df.TotalDKE$Powerful.Others,
df.TotalDKE$Logic.Acc,  method = c("pearson"))
pcor.test(df.TotalDKE$Logic.Overest, df.TotalDKE$Declarative,
df.TotalDKE$Logic.Acc,  method = c("pearson"))


cor.test(df.TotalDKE$Overall.Overestimation.Score,
df.TotalDKE$Overallsubprobover)
cor.test(df.TotalDKE$Overall.Overestimation.Score,
df.TotalDKE$OverallOverplacing)
cor.test(df.TotalDKE$Overallsubprobover, df.TotalDKE$OverallOverplacing)

```





```{r, brier score - metacognition calibration}
~~~

```
#https://link.springer.com/content/pdf/10.1007/
s11238-015-9509-9.pdf#cite.Murphy

#f = mean success rate (the sum of correct questions divided by total number
of questions) - same as "AccPerc" columns in TotalDKE sheet
#UNC = variance of the outcome variable which is independent of the judgment.
Equal to f(1-f)

#DI = the discrimination index measures the variability of the success rate
around the overall base rate (f)
  ###Np = number of times a confidence category is used
  ###fp = mean success rate for each confidence rating
#fp is equal to the stated probability category + accuracy (1 or 0) /Np or the
confidence category sum
# DI = [the sum of (Np(fp - f)^2)] / number of judgments

#CI = which measuresthe difference between the observed success rate (fp) and
the stated confidence
# CI = [the sum of (Np(p - fp)^2)] / number of judgments
    ### rather than calculating measures for each confidence, authors used a
binary (high/low) to calculate CI & DI. higher equal to mean = high
confidence. Low confidence = lower then the median.

#data wrangling

df.TotalDKE$HighConfMedian<-

df.conf<-as.data.frame(df.conf)
df.conf$HighConfMedian<- apply(df.conf[c(2:82)],1,FUN=function(x)
length(which(. >= df.conf$OverallConfMedian)))

df.conf %>%
  group_by(Subject) %>%
  filter(. >= OverallConfMedian) %>%
  nrow()

df.conf$HighConfMedian<-df.conf%>% group_by(Subject) %>% . >=
df.conf$OverallConfMedian

df.TotalDKE %>%
  mutate(LogicHighConfMedian = length(df.TotalDKE[c(129:157)] >=
df.TotalDKE$OverallConfMedian))



df.TotalDKE$OverallUNC <- df.TotalDKE$OverallAccPerc*(1-
df.TotalDKE$OverallAccPerc)
df.TotalDKE$GrammarUNC <- df.TotalDKE$GramAccPerc*(1-df.TotalDKE$GramAccPerc)
df.TotalDKE$NWUNC <- df.TotalDKE$NWAccPerc*(1-df.TotalDKE$NWAccPerc)
df.TotalDKE$OverallUNC <- df.TotalDKE$OverallAccPerc*(1-
df.TotalDKE$OverallAccPerc)

#Count the columns in which the confidence is greater than or equal to the
mean
#df.TotalDKE$OverallHighConfMean = apply(df.TotalDKE[c(148:176)], 2,
function(x) sum(x >= df.TotalDKE$OverallConfMean))
### couldn't get this to work in R. It keeps saying (Error in
`$<-.data.frame`(`*tmp*`, OverallHighConfMean, value = c(Overall_Conf_1 = 64L,
: replacement has 29 rows, data has 69)
###In excel code: =COUNTIF($ER2:$FT2,">="&$JV2), the first is the range of the
confidence questions, the ">=" means greater or equal, and the &$JV2 is the
mean column in which we are comparing. For overall confidence, but it into
it's own sheet and calculate across confidence only,

#df.TotalDKE<- read.csv("TotalDKE2.csv")

#DI = [the sum of (Np(fp - f)^2)] / number of judgments
#fp = sum of stated confidence (high/low) when accuracy was 1 or zero / count
of how many times that confidence category was used
#fp is equal to the sum of all (stated probability categoris + accuracy (1 or
0)) /Np (number of times the confidence category was used)
#count frequency for each probability category (high/low) when accuracy is 1
or 0. The excel code for this is =COUNTIFS($ER2:$FT2,">="&$JV2, $DO2:$EQ2, "=
1") / =COUNTIFS($ER2:$FT2,">="&$JV2, $DO2:$EQ2, "= 0")

#fp

df.TotalDKE$fp.highacc<- df.TotalDKE$OverallHighMean_Acc1 /
(df.TotalDKE$OverallHighMean_Acc1 + df.TotalDKE$OverallHighMean_Acc0)

df.TotalDKE$fp.Lowacc<- df.TotalDKE$OverallLowMean_Acc1 /
(df.TotalDKE$OverallLowMean_Acc1 + df.TotalDKE$OverallLowMean_Acc0)

df.TotalDKE$fp.highaccmedian<- df.TotalDKE$OverallHighMedian_Acc1 /
(df.TotalDKE$OverallHighMedian_Acc1 + df.TotalDKE$OverallHighMedian_Acc0)

df.TotalDKE$fp.Lowaccmedian<- df.TotalDKE$OverallLowMedian_Acc1 /
(df.TotalDKE$OverallLowMedian_Acc1 + df.TotalDKE$OverallLowMedian_Acc0)

#DI index
# DI = [the sum of (Np(fp - f)^2)] / number of judgments
#Np = (df.TotalDKE$OverallHighMean_Acc1 + df.TotalDKE$OverallHighMean_Acc0) =
the number of times a confidence category (high/low) is used
# (fp - f)^2= (df.TotalDKE$fp.highacc - df.TotalDKE$OverallAccPerc)^2)

df.TotalDKE$highNP<- df.TotalDKE$OverallHighMean_Acc1 +
df.TotalDKE$OverallHighMean_Acc0
df.TotalDKE$lowNP<- df.TotalDKE$OverallLowMean_Acc1 +
df.TotalDKE$OverallLowMean_Acc0
df.TotalDKE$highNPMedian<- df.TotalDKE$OverallHighMedian_Acc1 +
df.TotalDKE$OverallHighMedian_Acc0
df.TotalDKE$lowNPMedian<- df.TotalDKE$OverallLowMedian_Acc1 +
df.TotalDKE$OverallLowMedian_Acc0

df.TotalDKE$DIMean<- ((df.TotalDKE$highNP * (df.TotalDKE$fp.highacc -
df.TotalDKE$OverallAccPerc)^2) + (df.TotalDKE$lowNP *
(df.TotalDKE$fp.Lowaccmedian - df.TotalDKE$OverallAccPerc)^2)) / 81

df.TotalDKE$DIMedian<- ((df.TotalDKE$highNPMedian *
(df.TotalDKE$fp.highaccmedian - df.TotalDKE$OverallAccPerc)^2) +
(df.TotalDKE$lowNPMedian * (df.TotalDKE$fp.Lowaccmedian -
df.TotalDKE$OverallAccPerc)^2)) / 81

#write.csv(df.TotalDKE, "fp_check.csv")

df.TotalDKE$DIMean<-round(df.TotalDKE$DIMean, digits = 2)
df.TotalDKE$DIMedian<-round(df.TotalDKE$DIMedian, digits = 2)

#CI index
#CI = which measuresthe difference between the observed success rate (fp) and
the stated confidence
# CI = [the sum of (Np(p - fp)^2)] / number of judgments
#Np = (df.TotalDKE$OverallHighMean_Acc1 + df.TotalDKE$OverallHighMean_Acc0) =
the number of times a confidence category (high/low) is used

(df.TotalDKE$highNP * (df.TotalDKE$OverallHighMean_Acc1 -
df.TotalDKE$fp.highacc)^2)

df.TotalDKE$CIMean<- ((df.TotalDKE$highNP * (df.TotalDKE$OverallHighMean_Acc1
- df.TotalDKE$fp.highacc)^2) + (df.TotalDKE$highNP
*(df.TotalDKE$OverallHighMean_Acc0 - df.TotalDKE$fp.highacc)^2) +
(df.TotalDKE$lowNP *(df.TotalDKE$OverallLowMean_Acc1 -
df.TotalDKE$fp.Lowacc)^2) + (df.TotalDKE$lowNP*
(df.TotalDKE$OverallLowMean_Acc0 - df.TotalDKE$fp.Lowacc)^2))/ 81

df.TotalDKE$CIMedian<- (((df.TotalDKE$OverallHighMedian_Acc1 +
df.TotalDKE$OverallHighMedian_Acc0) * (df.TotalDKE$fp.highaccmedian -
df.TotalDKE$OverallAccPerc)^2) + ((df.TotalDKE$OverallLowMedian_Acc1 +
df.TotalDKE$OverallLowMedian_Acc0) * (df.TotalDKE$fp.Lowaccmedian -
df.TotalDKE$OverallAccPerc)^2)) / 81

#write.csv(df.TotalDKE, "fp_check.csv")

df.TotalDKE$DIMean<-round(df.TotalDKE$DIMean, digits = 2)
df.TotalDKE$DIMedian<-round(df.TotalDKE$DIMedian, digits = 2)
```

~~~
```

```{r, Linear Regression - Global}

df.TotalDKE$OverallPostDif<- (df.TotalDKE$Post.Gram.Dif +
df.TotalDKE$Post.NW.Dif + df.TotalDKE$Post.Log.Dif) / 3

#Overall Overestimation
OverallOverest.model<- lm(Overall.Overestimation.Score~
Overall.Correct.Answers + Years.Experience + Internal + Chance +
Powerful.Others + Extraversion + Agreeableness + Conscientiousness + Openness
+ Neuroticism + OCQ.Bias + Avoidant + Dependent + Vigilant + Spontaneous +
Intuitive + Respected + Brooding + Social.Desiribility + Declarative +
Procedural + Conditional + Planning + Information.Management +
Comprehension.Monitoring + Debugging.Strategies + Evaluation + OverallPostDif,
data=df.TotalDKE)
summ(OverallOverest.model)

vif(OverallOverest.model)
Overall.bc<-lm.beta(OverallOverest.model)
summary(Overall.bc)

OverallPartial2<-rsq.partial(OverallOverest.model)
lapply(OverallPartial2, function(x) write.table(data.frame(OverallPartial2),
'Overallpartial.csv'  , append= T, sep=','))




##Individual Task LMs

#Logic

df.TotalDKE$Logic.Over

logic.model<- lm(Logic.Overest ~ Logic.Acc + Years.Experience + Internal +
Chance + Powerful.Others + Extraversion + Agreeableness + Conscientiousness +
Openness + Neuroticism + OCQ.Bias + Avoidant + Dependent + Vigilant +
Spontaneous + Intuitive + Respected + Brooding + Social.Desiribility +
Declarative + Procedural + Conditional + Planning + Information.Management +
Comprehension.Monitoring + Debugging.Strategies + Evaluation + Post.Log.Dif,
data=df.TotalDKE)
summary(logic.model)
vif(logic.model)
l.bc<-lm.beta(logic.model)
summary(l.bc)
~~~

```
logicshort.model<- lm(Logic.Overest ~ Debugging.Strategies, data=df.TotalDKE)
lshort.bc<-lm.beta(logicshort.model)
summary(lshort.bc)

DebuggingXLogicAcc.model <- lm(Debugging.Strategies ~ Logic.Acc, data=
df.TotalDKE)
debug.acc.bc<- lm.beta(DebuggingXLogicAcc.model)
summary(debug.acc.bc)


#logic - difficulty
logicshort2.model<- lm(Logic.Overest ~ Debugging.Strategies + Logic.Acc,
data=df.TotalDKE)
lshort3.bc<-lm.beta(logicshort3.model)
summary(lshort3.bc)


logicshort3.model<- lm(Logic.Overest ~ Debugging.Strategies, data=df.TotalDKE)
lshort.bc<-lm.beta(logicshort.model)
summary(lshort.bc)

DebuggingXLogicDif.model <- lm(Debugging.Strategies ~ Post.Log.Dif, data=
df.TotalDKE)
debug.dif.bc<- lm.beta(DebuggingXLogicDif.model)
summary(debug.dif.bc)


#SEM
sem.model.measurement <- "Logic.Overest =~ Logic.Acc + Years.Experience +
Internal + Chance + Powerful.Others + Extraversion + Agreeableness +
Conscientiousness + Openness + Neuroticism + OCQ.Bias + Avoidant + Dependent +
Vigilant + Spontaneous + Intuitive + Respected + Brooding +
Social.Desiribility + Declarative + Procedural + Conditional + Planning +
Information.Management + Comprehension.Monitoring + Debugging.Strategies +
Evaluation + Post.Log.Dif"

sem.fit.measurement <- sem(sem.model.measurement, data = df.TotalDKE)
summary(sem.fit.measurement, fit.measures = TRUE)


#
# write.csv(df.TotalDKE, "FinalDKEData_Aug2020.csv")




logicPartial2<-rsq.partial(logic.model)
```

~~~
lapply(logicPartial2, function(x) write.table( data.frame(logicPartial),
'r2partial.csv'  , append= T, sep=',' ))


#Grammar Overestimation

grammar.model<- lm(Grammar.Overest ~ Gram.Acc + Years.Experience + Internal +
Chance + Powerful.Others + Extraversion + Agreeableness + Conscientiousness +
Openness + Neuroticism + OCQ.Bias + Avoidant + Dependent + Vigilant +
Spontaneous + Intuitive + Respected + Brooding + Social.Desiribility +
Declarative + Procedural + Conditional + Planning + Information.Management +
Comprehension.Monitoring + Debugging.Strategies + Evaluation + Post.Gram.Dif,
data=df.TotalDKE)
summary(grammar.model)
vif(grammar.model)
g.bc<-lm.beta(grammar.model)
summary(g.bc)

gramPartial2<-rsq.partial(grammar.model)
lapply(gramPartial2, function(x) write.table( data.frame(gramPartial2),
'grampartial.csv'  , append= T, sep=',' ))


#NW Overestimation

NW.model<- lm(NW.Overest ~ NW.Acc + Years.Experience + Internal + Chance +
Powerful.Others + Extraversion + Agreeableness + Conscientiousness + Openness
+ Neuroticism + OCQ.Bias + Avoidant + Dependent + Vigilant + Spontaneous +
Intuitive + Respected + Brooding + Social.Desiribility + Declarative +
Procedural + Conditional + Planning + Information.Management +
Comprehension.Monitoring + Debugging.Strategies + Evaluation + Post.NW.Dif,
data=df.TotalDKE)
summary(NW.model)
vif(NW.model)
n.bc<-lm.beta(NW.model)
summary(n.bc)

NWPartial2<-rsq.partial(NW.model)
lapply(NWPartial2, function(x) write.table( data.frame(NWPartial2),
'NWpartial.csv'  , append= T, sep=',' ))
```

```{r, overprecision linear regressions }
#Overprecision
~~~

```
overprecise.model <- lm(OverallOverPrecision ~ Overall.AccPercent +
Years.Experience + Internal + Chance + Powerful.Others + Extraversion +
Agreeableness + Conscientiousness + Openness + Neuroticism + OCQ.Bias +
Avoidant + Dependent + Vigilant + Spontaneous + Intuitive + Respected +
Brooding + Social.Desiribility + Declarative + Procedural + Conditional +
Planning + Information.Management + Comprehension.Monitoring +
Debugging.Strategies + Evaluation + OverallPostDif, data=df.TotalDKE)
summ(overprecise.model)

# summary(overprecise.model)
# vif(overprecise.model)
#
# overprecise.bc<-lm.beta(overprecise.model)
# summary(overprecise.bc)

OverallOverprecisionPartial2<-rsq.partial(overprecise.model)
lapply(OverallOverprecisionPartial2, function(x)
write.table(data.frame(OverallOverprecisionPartial2),
'OverallOverprecisionpartial.csv'  , append= T, sep=','))

tidy_overprecise<- tidy(overprecise.model)
write.csv(tidy_overprecise, "overprecise_reg_table.csv")

#Logic

log.op.model<- lm(LogicOverPrecision ~ Logic.AccPerc + Years.Experience +
Internal + Chance + Powerful.Others + Extraversion + Agreeableness +
Conscientiousness + Openness + Neuroticism + OCQ.Bias + Avoidant + Dependent +
Vigilant + Spontaneous + Intuitive + Respected + Brooding +
Social.Desiribility + Declarative + Procedural + Conditional + Planning +
Information.Management + Comprehension.Monitoring + Debugging.Strategies +
Evaluation + Post.Log.Dif, data=df.TotalDKE)
summary(log.op.model)
vif(log.op.model)

log.op.bc<-lm.beta(log.op.model)
summary(log.op.bc)

LogicOverprecisionPartial2<-rsq.partial(log.op.model)
lapply(LogicOverprecisionPartial2, function(x)
write.table(data.frame(LogicOverprecisionPartial2),
'LogicOverprecisionpartial.csv'  , append= T, sep=','))

tidy_logoverprecise<- tidy(log.op.model)
write.csv(tidy_logoverprecise, "logoverprecise_reg_table.csv")

#Grammar OverPrecision
```

~~~
gram.op.model<- lm(GrammarOverPrecision ~ Gram.AccPerc + Years.Experience +
Internal + Chance + Powerful.Others + Extraversion + Agreeableness +
Conscientiousness + Openness + Neuroticism + OCQ.Bias + Avoidant + Dependent +
Vigilant + Spontaneous + Intuitive + Respected + Brooding +
Social.Desiribility + Declarative + Procedural + Conditional + Planning +
Information.Management + Comprehension.Monitoring + Debugging.Strategies +
Evaluation + Post.Gram.Dif, data=df.TotalDKE)
summary(gram.op.model)
vif(gram.op.model)

gram.op.bc<-lm.beta(gram.op.model)
summary(gram.op.bc)

tidy_gramoverprecise<- tidy(gram.op.model)
write.csv(tidy_gramoverprecise, "gramoverprecise_reg_table.csv")

GrammarOverprecisionPartial2<-rsq.partial(gram.op.model)
lapply(GrammarOverprecisionPartial2, function(x)
write.table(data.frame(GrammarOverprecisionPartial2),
'GrammarOverprecisionpartial.csv'  , append= T, sep=','))

#NW Overprecision

NW.op.model<- lm(NWOverPrecision ~ NW.AccPerc + Years.Experience + Internal +
Chance + Powerful.Others + Extraversion + Agreeableness + Conscientiousness +
Openness + Neuroticism + OCQ.Bias + Avoidant + Dependent + Vigilant +
Spontaneous + Intuitive + Respected + Brooding + Social.Desiribility +
Declarative + Procedural + Conditional + Planning + Information.Management +
Comprehension.Monitoring + Debugging.Strategies + Evaluation + Post.NW.Dif,
data=df.TotalDKE)
summary(NW.op.model)
vif(NW.op.model)

NW.op.bc<-lm.beta(NW.op.model)
summary(NW.op.bc)

NWOverprecisionPartial2<-rsq.partial(NW.op.model)
lapply(NWOverprecisionPartial2, function(x)
write.table(data.frame(NWOverprecisionPartial2), 'NWOverprecisionpartial.csv'
, append= T, sep=','))

tidy_nwoverprecise<- tidy(NW.op.model)
write.csv(tidy_nwoverprecise, "nwoverprecise_reg_table.csv")

```

```{r,linear regression - overplacing model}
~~~

```
#Overplacing
df.TotalDKE$OverallOverplacing <- df.TotalDKE$OverallEstimatedPercentile -
df.TotalDKE$OverallPercentileRank
df.TotalDKE$LogicOverplacing <- df.TotalDKE$LogicEstimatedPercentile -
df.TotalDKE$LogicPercentileRank
df.TotalDKE$GrammarOverplacing <- df.TotalDKE$GrammarEstimatedPercentile -
df.TotalDKE$GrammarPercentileRank
df.TotalDKE$NWOverplacing <- df.TotalDKE$NWEstimatedPercentile -
df.TotalDKE$NWPercentileRank


#Overall Overplacing

df.TotalDKE$Evaluation_mc<- scale(df.TotalDKE$Evaluation)
df.TotalDKE$OverallPercentileRank_mc<-scale(df.TotalDKE$OverallPercentileRank)
#moderate.lm(IV, mod, DV, data, mc = FALSE)


OverallOverplacing.model<- lm(OverallOverplacing~ OverallPercentileRank +
Years.Experience + Internal + Chance + Powerful.Others + Extraversion +
Agreeableness + Conscientiousness + Openness + Neuroticism + OCQ.Bias +
Avoidant + Dependent + Vigilant + Spontaneous + Intuitive + Respected +
Brooding + Social.Desiribility +  Declarative + Procedural + Conditional +
Planning + Information.Management + Comprehension.Monitoring +
Debugging.Strategies + Evaluation + OverallPostDif,  data=df.TotalDKE)
summ(OverallOverplacing.model)

# summary(OverallOverplacing.model)
#
# vif(OverallOverplacing.model)
#
# overalloverplacing.bc<-lm.beta(OverallOverplacing.model)
# summary(overalloverplacing.bc)
# gvlma(OverallOverplacing.model)


OverallOverplacingPartial2<-rsq.partial(OverallOverplacing.model)
lapply(OverallOverplacingPartial2, function(x)
write.table(data.frame(OverallOverplacingPartial2),
'OverallOverplacingpartial.csv'  , append= T, sep=','))

tidy_OverallOverplacing<- tidy(OverallOverplacing.model)
write.csv(tidy_OverallOverplacing, "OverallOverplacing_reg_table.csv")

#Logic overplacing
```

```
LogicOverplacing.model<- lm(LogicOverplacing~ LogicPercentileRank +
Years.Experience + Internal + Chance + Powerful.Others + Extraversion +
Agreeableness + Conscientiousness + Openness + Neuroticism + OCQ.Bias +
Avoidant + Dependent+ Vigilant + Spontaneous + Intuitive + Respected +
Brooding + Social.Desiribility + Declarative + Procedural + Conditional +
Planning + Information.Management + Comprehension.Monitoring +
Debugging.Strategies + Evaluation + Post.Log.Dif,  data=df.TotalDKE)
summary(LogicOverplacing.model)
vif(LogicOverplacing.model)
gvlma(LogicOverplacing.model)

Logicoverplacing.bc<-lm.beta(LogicOverplacing.model)
summary(Logicoverplacing.bc)

LogicOverplacingPartial2<-rsq.partial(LogicOverplacing.model)
lapply(LogicOverplacingPartial2, function(x)
write.table(data.frame(LogicOverplacingPartial2),
'LogicOverplacingpartial.csv'  , append= T, sep=','))

tidy_LogicOverplacing<- tidy(LogicOverplacing.model)
write.csv(tidy_LogicOverplacing, "LogicOverplacing_reg_table.csv")

#Grammar overplacing
GrammarOverplacing.model<- lm(GrammarOverplacing~ GrammarPercentileRank +
Years.Experience + Internal + Chance + Powerful.Others + Extraversion +
Agreeableness + Conscientiousness + Openness + Neuroticism + OCQ.Bias +
Avoidant + Dependent + Vigilant + Spontaneous + Intuitive + Respected +
Brooding + Social.Desiribility + Declarative + Procedural + Conditional +
Planning + Information.Management + Comprehension.Monitoring +
Debugging.Strategies + Evaluation + Post.Gram.Dif,  data=df.TotalDKE)
summary(GrammarOverplacing.model)
vif(GrammarOverplacing.model)

Grammaroverplacing.bc<-lm.beta(GrammarOverplacing.model)
summary(Grammaroverplacing.bc)
gvlma(GrammarOverplacing.model)

GrammarOverplacingPartial2<-rsq.partial(GrammarOverplacing.model)
lapply(GrammarOverplacingPartial2, function(x)
write.table(data.frame(GrammarOverplacingPartial2),
'GrammarOverplacingpartial.csv', append= T, sep=','))

tidy_GrammarOverplacing<- tidy(GrammarOverplacing.model)
write.csv(tidy_GrammarOverplacing, "GrammarOverplacing_reg_table.csv")

#NW overplacing
```

~~~
NWOverplacing.model<- lm(NWOverplacing~ NWPercentileRank + Years.Experience +
Internal + Chance + Powerful.Others + Extraversion + Agreeableness +
Conscientiousness + Openness + Neuroticism + OCQ.Bias + Avoidant + Dependent +
Vigilant + Spontaneous + Intuitive + Respected + Brooding +
Social.Desiribility + Declarative + Procedural + Conditional + Planning +
Information.Management + Comprehension.Monitoring + Debugging.Strategies +
Evaluation + Post.NW.Dif,  data=df.TotalDKE)
summary(NWOverplacing.model)
vif(NWOverplacing.model)

NWoverplacing.bc<-lm.beta(NWOverplacing.model)
summary(NWoverplacing.bc)
gvlma(NWOverplacing.model)

NWOverplacingPartial2<-rsq.partial(NWOverplacing.model)
lapply(NWOverplacingPartial2, function(x)
write.table(data.frame(NWOverplacingPartial2), 'NWOverplacingpartial.csv'  ,
append= T, sep=','))


tidy_NWOverplacing<- tidy(NWOverplacing.model)
write.csv(tidy_NWOverplacing, "NWOverplacing_reg_table.csv")

```
~~~

~~~
```{r, confidence on error & correct trials}
#write.csv(df.TotalDKE, "TotalDKE_Final.csv")
# df.TotalDKE<-read_csv("TotalDKE_Final.csv")
# df.TotalDKE<-df.TotalDKE[-c(1,2)]

#Overall

df.overallcorrectnessstats<- df.TotalDKE%>%
  group_by(OverallQuantile)%>%
  summarise_at(vars(OverallAvgConfIncorrect, OverallAvgConfCorrect),
funs(mean(., na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
df.overallcorrectnessstats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile
3", "Quartile 4")

df.overallcorrectnessstats$OverallAvgConfIncorrect_mean<-
round(df.overallcorrectnessstats$OverallAvgConfIncorrect_mean, digits=2)
df.overallcorrectnessstats$OverallAvgConfCorrect_mean<-
round(df.overallcorrectnessstats$OverallAvgConfCorrect_mean, digits=2)
~~~

```
 ggplot(df.overallcorrectnessstats, aes(quartiles)) +
  geom_line(aes(y= OverallAvgConfCorrect_mean, group=1, color="dodgerblue2"))
+
  geom_line(aes(y= OverallAvgConfIncorrect_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin=
df.overallcorrectnessstats$OverallAvgConfCorrect_mean -
OverallAvgConfCorrect_se, ymax=
df.overallcorrectnessstats$OverallAvgConfCorrect_mean +
OverallAvgConfCorrect_se),   width=0.1) +
  geom_errorbar(aes(ymin=
df.overallcorrectnessstats$OverallAvgConfIncorrect_mean -
OverallAvgConfIncorrect_se, ymax=
df.overallcorrectnessstats$OverallAvgConfIncorrect_mean +
OverallAvgConfIncorrect_se), width=0.1)+
  geom_point(y= df.overallcorrectnessstats$OverallAvgConfCorrect_mean)+
  geom_label(aes(quartiles, OverallAvgConfCorrect_mean, label =
OverallAvgConfCorrect_mean), vjust = -1) +
  geom_point(y= df.overallcorrectnessstats$OverallAvgConfIncorrect_mean)+
  geom_label(aes(quartiles, OverallAvgConfIncorrect_mean, label =
OverallAvgConfIncorrect_mean), vjust = 2) +
  expand_limits(y=c(0,1)) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits = c(0,1)) +
  xlab("Performance Quartile") +
  ylab("Confidence Level") +
  ggtitle("Avg Confidence When Correct/Incorrect (Overall)") +
    theme_bw() +
    scale_color_identity(name = "Legend",
                          labels = c("Confidence When Incorrect", "Confidence
when Correct"),
                          guide = "legend")
 ggsave("OverallConfWhenCorrect.png")

 #Logic

 df.Logiccorrectnessstats<- df.TotalDKE%>%
  group_by(LogicQuantile)%>%
  summarise_at(vars(LogicAvgConfIncorrect, LogicAvgConfCorrect), funs(mean(.,
na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
df.Logiccorrectnessstats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile
3", "Quartile 4")

df.Logiccorrectnessstats$LogicAvgConfIncorrect_mean<-
round(df.Logiccorrectnessstats$LogicAvgConfIncorrect_mean, digits=2)
df.Logiccorrectnessstats$LogicAvgConfCorrect_mean<-
round(df.Logiccorrectnessstats$LogicAvgConfCorrect_mean, digits=2)
```

```
 ggplot(df.Logiccorrectnessstats, aes(quartiles)) +
  geom_line(aes(y= LogicAvgConfCorrect_mean, group=1, color="dodgerblue2")) +
  geom_line(aes(y= LogicAvgConfIncorrect_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin= df.Logiccorrectnessstats$LogicAvgConfCorrect_mean -
LogicAvgConfCorrect_se, ymax=
df.Logiccorrectnessstats$LogicAvgConfCorrect_mean + LogicAvgConfCorrect_se),
width=0.1) +
  geom_errorbar(aes(ymin= df.Logiccorrectnessstats$LogicAvgConfIncorrect_mean
- LogicAvgConfIncorrect_se, ymax=
df.Logiccorrectnessstats$LogicAvgConfIncorrect_mean +
LogicAvgConfIncorrect_se), width=0.1)+
  geom_point(y= df.Logiccorrectnessstats$LogicAvgConfCorrect_mean)+
  geom_label(aes(quartiles, LogicAvgConfCorrect_mean, label =
LogicAvgConfCorrect_mean), vjust = -1) +
  geom_point(y= df.Logiccorrectnessstats$LogicAvgConfIncorrect_mean)+
  geom_label(aes(quartiles, LogicAvgConfIncorrect_mean, label =
LogicAvgConfIncorrect_mean), vjust = 2) +
  expand_limits(y=c(0,1)) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits = c(0,1)) +
  xlab("Performance Quartile") +
  ylab("Confidence Level") +
  ggtitle("Avg Confidence When Correct/Incorrect (Logic)") +
    theme_bw() +
    scale_color_identity(name = "Legend",
                          labels = c("Confidence When Incorrect", "Confidence
when Correct"),
                          guide = "legend")

  ggsave("LogicConfWhenCorrect.png")

  #Grammar
 df.TotalDKE$GrammarAvgConfIncorrect<-
as.numeric(df.TotalDKE$GrammarAvgConfIncorrect)

 df.Grammarcorrectnessstats<- df.TotalDKE%>%
  group_by(GrammarQuantile)%>%
  summarise_at(vars(GrammarAvgConfIncorrect, GrammarAvgConfCorrect),
funs(mean(., na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
df.Grammarcorrectnessstats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile
3", "Quartile 4")

df.Grammarcorrectnessstats$GrammarAvgConfIncorrect_mean<-
round(df.Grammarcorrectnessstats$GrammarAvgConfIncorrect_mean, digits=2)
df.Grammarcorrectnessstats$GrammarAvgConfCorrect_mean<-
round(df.Grammarcorrectnessstats$GrammarAvgConfCorrect_mean, digits=2)


 ggplot(df.Grammarcorrectnessstats, aes(quartiles)) +
  geom_line(aes(y= GrammarAvgConfCorrect_mean, group=1, color="dodgerblue2"))
+
  geom_line(aes(y= GrammarAvgConfIncorrect_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin=
df.Grammarcorrectnessstats$GrammarAvgConfCorrect_mean -
GrammarAvgConfCorrect_se, ymax=
df.Grammarcorrectnessstats$GrammarAvgConfCorrect_mean +
GrammarAvgConfCorrect_se),   width=0.1) +
  geom_errorbar(aes(ymin=
df.Grammarcorrectnessstats$GrammarAvgConfIncorrect_mean -
GrammarAvgConfIncorrect_se, ymax=
df.Grammarcorrectnessstats$GrammarAvgConfIncorrect_mean +
GrammarAvgConfIncorrect_se), width=0.1)+
  geom_point(y= df.Grammarcorrectnessstats$GrammarAvgConfCorrect_mean)+
  geom_label(aes(quartiles, GrammarAvgConfCorrect_mean, label =
GrammarAvgConfCorrect_mean), vjust = -1) +
  geom_point(y= df.Grammarcorrectnessstats$GrammarAvgConfIncorrect_mean)+
  geom_label(aes(quartiles, GrammarAvgConfIncorrect_mean, label =
GrammarAvgConfIncorrect_mean), vjust = 2) +
  expand_limits(y=c(0,1)) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits = c(0,1)) +
  xlab("Performance Quartile") +
  ylab("Confidence Level") +
  ggtitle("Avg Confidence When Correct/Incorrect (Grammar)") +
    theme_bw() +
    scale_color_identity(name = "Legend",
                          labels = c("Confidence When Incorrect", "Confidence
when Correct"),
                          guide = "legend")

  ggsave("GrammarConfWhenCorrect.png")

   #NW

 df.NWcorrectnessstats<- df.TotalDKE%>%
  group_by(NWQuantile)%>%
  summarise_at(vars(NWAvgConfIncorrect, NWAvgConfCorrect), funs(mean(.,
na.rm=T), sd(.,na.rm = T), n=n(), se=sd(.,na.rm=T)/sqrt(n())))
df.NWcorrectnessstats$quartiles<- c("Quartile 1", "Quartile 2", "Quartile 3",
"Quartile 4")

df.NWcorrectnessstats$NWAvgConfIncorrect_mean<-
round(df.NWcorrectnessstats$NWAvgConfIncorrect_mean, digits=2)
df.NWcorrectnessstats$NWAvgConfCorrect_mean<-
round(df.NWcorrectnessstats$NWAvgConfCorrect_mean, digits=2)


 ggplot(df.NWcorrectnessstats, aes(quartiles)) +
  geom_line(aes(y= NWAvgConfCorrect_mean, group=1, color="dodgerblue2")) +
  geom_line(aes(y= NWAvgConfIncorrect_mean, group=1, color = "coral1")) +
  geom_errorbar(aes(ymin= df.NWcorrectnessstats$NWAvgConfCorrect_mean -
NWAvgConfCorrect_se, ymax= df.NWcorrectnessstats$NWAvgConfCorrect_mean +
NWAvgConfCorrect_se),   width=0.1) +
  geom_errorbar(aes(ymin= df.NWcorrectnessstats$NWAvgConfIncorrect_mean -
NWAvgConfIncorrect_se, ymax= df.NWcorrectnessstats$NWAvgConfIncorrect_mean +
NWAvgConfIncorrect_se), width=0.1)+
  geom_point(y= df.NWcorrectnessstats$NWAvgConfCorrect_mean)+
  geom_label(aes(quartiles, NWAvgConfCorrect_mean, label =
NWAvgConfCorrect_mean), vjust = -1) +
  geom_point(y= df.NWcorrectnessstats$NWAvgConfIncorrect_mean)+
  geom_label(aes(quartiles, NWAvgConfIncorrect_mean, label =
NWAvgConfIncorrect_mean), vjust = 2) +
  expand_limits(y=c(0,1)) +
  scale_y_continuous(breaks= seq(0,1, by=0.1), limits = c(0,1)) +
  xlab("Performance Quartile") +
  ylab("Confidence Level") +
  ggtitle("Avg Confidence When Correct/Incorrect (NW)") +
    theme_bw() +
    scale_color_identity(name = "Legend",
                          labels = c("Confidence When Incorrect", "Confidence
when Correct"),
                          guide = "legend")

  ggsave("NWConfWhenCorrect.png")
```

```{r, t-tests between accuracy and estimates on each task}

#t.test to compare means for grammar, logic, and NW means

t.test(df.TotalDKE$Logic.Acc, df.TotalDKE$Post.Log.Est)

t.test(df.TotalDKE$NW.Acc, df.TotalDKE$Post.NW.Est)

t.test(df.TotalDKE$NW.Acc, df.TotalDKE$Post.NW.Est)


#by quartile / questions
#Overall
Overall.AccQ1 <-
df.TotalDKE$Overall.Correct.Answers[df.TotalDKE$OverallQuantile=="[38,50]"]
Overall.AccQ2 <-
df.TotalDKE$Overall.Correct.Answers[df.TotalDKE$OverallQuantile=="(50,54]"]
Overall.AccQ3 <-
df.TotalDKE$Overall.Correct.Answers[df.TotalDKE$OverallQuantile=="(54,58]"]
Overall.AccQ4 <-
df.TotalDKE$Overall.Correct.Answers[df.TotalDKE$OverallQuantile=="(58,69]"]

Overall.EstQ1 <-
df.TotalDKE$Overall.Correct.Answers.Estimate[df.TotalDKE$OverallQuantile=="[38,50]"]
Overall.EstQ2 <-
df.TotalDKE$Overall.Correct.Answers.Estimate[df.TotalDKE$OverallQuantile=="(50,54]"]
Overall.EstQ3 <-
df.TotalDKE$Overall.Correct.Answers.Estimate[df.TotalDKE$OverallQuantile=="(54,58]"]
Overall.EstQ4 <-
df.TotalDKE$Overall.Correct.Answers.Estimate[df.TotalDKE$OverallQuantile=="(58,69]"]

t.test(Overall.AccQ1, Overall.EstQ1, paired = T)
t.test(Overall.AccQ2, Overall.EstQ2, paired = T)
t.test(Overall.AccQ3, Overall.EstQ3, paired = T)
t.test(Overall.AccQ4, Overall.EstQ4, paired = T)

#Logic
Logic.AccQ1 <- df.TotalDKE$Logic.Acc[df.TotalDKE$LogicQuantile=="[10,18]"]
Logic.AccQ2 <- df.TotalDKE$Logic.Acc[df.TotalDKE$LogicQuantile=="(18,21]"]
Logic.AccQ3 <- df.TotalDKE$Logic.Acc[df.TotalDKE$LogicQuantile=="(21,23]"]
Logic.AccQ4 <- df.TotalDKE$Logic.Acc[df.TotalDKE$LogicQuantile=="(23,28]"]

Logic.EstQ1 <- df.TotalDKE$Post.Log.Est[df.TotalDKE$LogicQuantile=="[10,18]"]
Logic.EstQ2 <- df.TotalDKE$Post.Log.Est[df.TotalDKE$LogicQuantile=="(18,21]"]
Logic.EstQ3 <- df.TotalDKE$Post.Log.Est[df.TotalDKE$LogicQuantile=="(21,23]"]
Logic.EstQ4 <- df.TotalDKE$Post.Log.Est[df.TotalDKE$LogicQuantile=="(23,28]"]

t.test(Logic.AccQ1, Logic.EstQ1, paired = T)
t.test(Logic.AccQ2, Logic.EstQ2, paired = T)
t.test(Logic.AccQ3, Logic.EstQ3, paired = T)
t.test(Logic.AccQ4, Logic.EstQ4, paired = T)


#Grammar

Grammar.AccQ1 <- df.TotalDKE$Gram.Acc[df.TotalDKE$GrammarQuantile=="[15,20]"]
Grammar.AccQ2 <- df.TotalDKE$Gram.Acc[df.TotalDKE$GrammarQuantile=="(20,21]"]
Grammar.AccQ3 <- df.TotalDKE$Gram.Acc[df.TotalDKE$GrammarQuantile=="(21,22]"]
Grammar.AccQ4 <- df.TotalDKE$Gram.Acc[df.TotalDKE$GrammarQuantile=="(22,25]"]

Grammar.EstQ1 <-
df.TotalDKE$Post.Gram.Est[df.TotalDKE$GrammarQuantile=="[15,20]"]
Grammar.EstQ2 <-
df.TotalDKE$Post.Gram.Est[df.TotalDKE$GrammarQuantile=="(20,21]"]
Grammar.EstQ3 <-
df.TotalDKE$Post.Gram.Est[df.TotalDKE$GrammarQuantile=="(21,22]"]
Grammar.EstQ4 <-
df.TotalDKE$Post.Gram.Est[df.TotalDKE$GrammarQuantile=="(22,25]"]
t.test(Grammar.AccQ1, Grammar.EstQ1, paired = T)
t.test(Grammar.AccQ2, Grammar.EstQ2, paired = T)
t.test(Grammar.AccQ3, Grammar.EstQ3, paired = T)
t.test(Grammar.AccQ4, Grammar.EstQ4, paired = T)

#NW

NW.AccQ1 <- df.TotalDKE$NW.Acc[df.TotalDKE$NWQuantile=="[5,11]"]
NW.AccQ2 <- df.TotalDKE$NW.Acc[df.TotalDKE$NWQuantile=="(11,12]"]
NW.AccQ3 <- df.TotalDKE$NW.Acc[df.TotalDKE$NWQuantile=="(12,15]"]
NW.AccQ4 <- df.TotalDKE$NW.Acc[df.TotalDKE$NWQuantile=="(15,19]"]

NW.EstQ1 <- df.TotalDKE$Post.NW.Est[df.TotalDKE$NWQuantile=="[5,11]"]
NW.EstQ2 <- df.TotalDKE$Post.NW.Est[df.TotalDKE$NWQuantile=="(11,12]"]
NW.EstQ3 <- df.TotalDKE$Post.NW.Est[df.TotalDKE$NWQuantile=="(12,15]"]
NW.EstQ4 <- df.TotalDKE$Post.NW.Est[df.TotalDKE$NWQuantile=="(15,19]"]

t.test(NW.AccQ1, NW.EstQ1, paired = T)
t.test(NW.AccQ2, NW.EstQ2, paired = T)
t.test(NW.AccQ3, NW.EstQ3, paired = T)
t.test(NW.AccQ4, NW.EstQ4, paired = T)


# Percentile
#Overall
Overall.AccPQ1 <-
df.TotalDKE$OverallPercentileRank[df.TotalDKE$OverallQuartileRank=="[0,0.239]"]
Overall.AccPQ2 <-
df.TotalDKE$OverallPercentileRank[df.TotalDKE$OverallQuartileRank=="(0.239,0.478]"]
Overall.AccPQ3 <-
df.TotalDKE$OverallPercentileRank[df.TotalDKE$OverallQuartileRank=="(0.478,0.717]"]
Overall.AccPQ4 <-
df.TotalDKE$OverallPercentileRank[df.TotalDKE$OverallQuartileRank=="(0.717,1]"]

Overall.EstPQ1 <-
df.TotalDKE$OverallEstimatedPercentile[df.TotalDKE$OverallQuartileRank=="[0,0.239]"]
Overall.EstPQ2 <-
df.TotalDKE$OverallEstimatedPercentile[df.TotalDKE$OverallQuartileRank=="(0.239,0.478]"]
Overall.EstPQ3 <-
df.TotalDKE$OverallEstimatedPercentile[df.TotalDKE$OverallQuartileRank=="(0.478,0.717]"]
Overall.EstPQ4 <-
df.TotalDKE$OverallEstimatedPercentile[df.TotalDKE$OverallQuartileRank=="(0.717,1]"]

t.test(Overall.AccPQ1, Overall.EstPQ1, paired = T)
t.test(Overall.AccPQ2, Overall.EstPQ2, paired = T)
t.test(Overall.AccPQ3, Overall.EstPQ3, paired = T)
t.test(Overall.AccPQ4, Overall.EstPQ4, paired = T)
```

```
#Logic
Logic.AccPQ1 <-
df.TotalDKE$LogicPercentileRank[df.TotalDKE$LogicQuartileRank=="[0,0.239]"]
Logic.AccPQ2 <-
df.TotalDKE$LogicPercentileRank[df.TotalDKE$LogicQuartileRank=="(0.239,0.467]"]
Logic.AccPQ3 <-
df.TotalDKE$LogicPercentileRank[df.TotalDKE$LogicQuartileRank=="(0.467,0.63]"]
Logic.AccPQ4 <-
df.TotalDKE$LogicPercentileRank[df.TotalDKE$LogicQuartileRank=="(0.63,1]"]

Logic.EstPQ1 <-
df.TotalDKE$LogicEstimatedPercentile[df.TotalDKE$LogicQuartileRank=="[0,0.239]"]
Logic.EstPQ2 <-
df.TotalDKE$LogicEstimatedPercentile[df.TotalDKE$LogicQuartileRank=="(0.239,0.467]"]
Logic.EstPQ3 <-
df.TotalDKE$LogicEstimatedPercentile[df.TotalDKE$LogicQuartileRank=="(0.467,0.63]"]
Logic.EstPQ4 <-
df.TotalDKE$LogicEstimatedPercentile[df.TotalDKE$LogicQuartileRank=="(0.63,1]"]

t.test(Logic.AccPQ1, Logic.EstPQ1, paired = T)
t.test(Logic.AccPQ2, Logic.EstPQ2, paired = T)
t.test(Logic.AccPQ3, Logic.EstPQ3, paired = T)
t.test(Logic.AccPQ4, Logic.EstPQ4, paired = T)


#Grammar

Grammar.AccPQ1 <-
df.TotalDKE$GrammarPercentileRank[df.TotalDKE$GrammarQuartileRank=="[0,0.185]"]
Grammar.AccPQ2 <-
df.TotalDKE$GrammarPercentileRank[df.TotalDKE$GrammarQuartileRank=="(0.185,0.413]"]
Grammar.AccPQ3 <-
df.TotalDKE$GrammarPercentileRank[df.TotalDKE$GrammarQuartileRank=="(0.413,0.696]"]
Grammar.AccPQ4 <-
df.TotalDKE$GrammarPercentileRank[df.TotalDKE$GrammarQuartileRank=="(0.696,1]"]

Grammar.EstPQ1 <-
df.TotalDKE$GrammarEstimatedPercentile[df.TotalDKE$GrammarQuartileRank=="[0,0.185]"]
Grammar.EstPQ2 <-
df.TotalDKE$GrammarEstimatedPercentile[df.TotalDKE$GrammarQuartileRank=="(0.185,0.413]"]
Grammar.EstPQ3 <-
df.TotalDKE$GrammarEstimatedPercentile[df.TotalDKE$GrammarQuartileRank=="(0.413,0.696]"]
Grammar.EstPQ4 <-
df.TotalDKE$GrammarEstimatedPercentile[df.TotalDKE$GrammarQuartileRank=="(0.696,1]"]

t.test(Grammar.AccPQ1, Grammar.EstPQ1, paired = T)
t.test(Grammar.AccPQ2, Grammar.EstPQ2, paired = T)
t.test(Grammar.AccPQ3, Grammar.EstPQ3, paired = T)
```

```
t.test(Grammar.AccPQ4, Grammar.EstPQ4, paired = T)

#NW

NW.AccPQ1 <-
df.TotalDKE$NWPercentileRank[df.TotalDKE$NWQuartileRank=="[0,0.228]"]
NW.AccPQ2 <-
df.TotalDKE$NWPercentileRank[df.TotalDKE$NWQuartileRank=="(0.228,0.348]"]
NW.AccPQ3 <-
df.TotalDKE$NWPercentileRank[df.TotalDKE$NWQuartileRank=="(0.348,0.717]"]
NW.AccPQ4 <-
df.TotalDKE$NWPercentileRank[df.TotalDKE$NWQuartileRank=="(0.717,0.989]"]

NW.EstPQ1 <-
df.TotalDKE$NWEstimatedPercentile[df.TotalDKE$NWQuartilePerc=="[0,0.228]"]
NW.EstPQ2 <-
df.TotalDKE$NWEstimatedPercentile[df.TotalDKE$NWQuartileRank=="(0.228,0.348]"]
NW.EstPQ3 <-
df.TotalDKE$NWEstimatedPercentile[df.TotalDKE$NWQuartileRank=="(0.348,0.717]"]
NW.EstPQ4 <-
df.TotalDKE$NWEstimatedPercentile[df.TotalDKE$NWQuartileRank=="(0.717,0.989]"]

t.test(NW.AccPQ1, NW.EstPQ1, paired = T)
t.test(NW.AccPQ2, NW.EstPQ2, paired = T)
t.test(NW.AccPQ3, NW.EstPQ3, paired = T)
t.test(NW.AccPQ4, NW.EstPQ4, paired = T)

#Overprecision

#Overall

Overall.AccOQ1 <-
df.TotalDKE$Overall.AccPercent[df.TotalDKE$OverallQuartilePerc=="[0.469,0.617]"]
Overall.AccOQ2 <-
df.TotalDKE$Overall.AccPercent[df.TotalDKE$OverallQuartilePerc=="(0.617,0.667]"]
Overall.AccOQ3 <-
df.TotalDKE$Overall.AccPercent[df.TotalDKE$OverallQuartilePerc=="(0.667,0.716]"]
Overall.AccOQ4 <-
df.TotalDKE$Overall.AccPercent[df.TotalDKE$OverallQuartilePerc=="(0.716,0.852]"]

Overall.EstOQ1 <-
df.TotalDKE$Overall.Confidence.Average[df.TotalDKE$OverallQuartilePerc==
"[0.469,0.617]"]
Overall.EstOQ2 <-
df.TotalDKE$Overall.Confidence.Average[df.TotalDKE$OverallQuartilePerc==
"(0.617,0.667]"]
Overall.EstOQ3 <-
df.TotalDKE$Overall.Confidence.Average[df.TotalDKE$OverallQuartilePerc==
"(0.667,0.716]"]
Overall.EstOQ4 <-
df.TotalDKE$Overall.Confidence.Average[df.TotalDKE$OverallQuartilePerc==
"(0.716,0.852]"]

t.test(Overall.AccOQ1, Overall.EstOQ1, paired = T)
t.test(Overall.AccOQ2, Overall.EstOQ2, paired = T)
t.test(Overall.AccOQ3, Overall.EstOQ3, paired = T)
t.test(Overall.AccOQ4, Overall.EstOQ4, paired = T)

#Logic

Logic.AccOQ1 <-
df.TotalDKE$Logic.AccPerc[df.TotalDKE$LogicQuartilePerc=="[0.345,0.621]"]
Logic.AccOQ2 <-
df.TotalDKE$Logic.AccPerc[df.TotalDKE$LogicQuartilePerc=="(0.621,0.724]"]
Logic.AccOQ3 <-
df.TotalDKE$Logic.AccPerc[df.TotalDKE$LogicQuartilePerc=="(0.724,0.793]"]
Logic.AccOQ4 <-
df.TotalDKE$Logic.AccPerc[df.TotalDKE$LogicQuartilePerc=="(0.793,0.966]"]

Logic.EstOQ1 <-
df.TotalDKE$Logic.Confidence.Avg[df.TotalDKE$LogicQuartilePerc==
"[0.345,0.621]"]
Logic.EstOQ2 <-
df.TotalDKE$Logic.Confidence.Avg[df.TotalDKE$LogicQuartilePerc==
"(0.621,0.724]"]
Logic.EstOQ3 <-
df.TotalDKE$Logic.Confidence.Avg[df.TotalDKE$LogicQuartilePerc==
"(0.724,0.793]"]
Logic.EstOQ4 <-
df.TotalDKE$Logic.Confidence.Avg[df.TotalDKE$LogicQuartilePerc==
"(0.793,0.966]"]

t.test(Logic.AccOQ1, Logic.EstOQ1, paired = T)
t.test(Logic.AccOQ2, Logic.EstOQ2, paired = T)
t.test(Logic.AccOQ3, Logic.EstOQ3, paired = T)
t.test(Logic.AccOQ4, Logic.EstOQ4, paired = T)

#Grammar

Grammar.AccOQ1 <-
df.TotalDKE$Gram.AccPerc[df.TotalDKE$GrammarQuartilePerc=="[0.6,0.8]"]
Grammar.AccOQ2 <-
df.TotalDKE$Gram.AccPerc[df.TotalDKE$GrammarQuartilePerc=="(0.8,0.84]"]
Grammar.AccOQ3 <-
df.TotalDKE$Gram.AccPerc[df.TotalDKE$GrammarQuartilePerc=="(0.84,0.88]"]
Grammar.AccOQ4 <-
df.TotalDKE$Gram.AccPerc[df.TotalDKE$GrammarQuartilePerc=="(0.88,1]"]

Grammar.EstOQ1 <-
df.TotalDKE$Grammar.Confidence.Avg[df.TotalDKE$GrammarQuartilePerc==
"[0.6,0.8]"]
Grammar.EstOQ2 <-
df.TotalDKE$Grammar.Confidence.Avg[df.TotalDKE$GrammarQuartilePerc==
"(0.8,0.84]"]
Grammar.EstOQ3 <-
df.TotalDKE$Grammar.Confidence.Avg[df.TotalDKE$GrammarQuartilePerc==
"(0.84,0.88]"]
Grammar.EstOQ4 <-
df.TotalDKE$Grammar.Confidence.Avg[df.TotalDKE$GrammarQuartilePerc==
"(0.88,1]"]

t.test(Grammar.AccOQ1, Grammar.EstOQ1, paired = T)
t.test(Grammar.AccOQ2, Grammar.EstOQ2, paired = T)
t.test(Grammar.AccOQ3, Grammar.EstOQ3, paired = T)
t.test(Grammar.AccOQ4, Grammar.EstOQ4, paired = T)


#NW

NW.AccOQ1 <-
df.TotalDKE$NW.AccPerc[df.TotalDKE$NWQuartilePerc=="[0.185,0.407]"]
NW.AccOQ2 <-
df.TotalDKE$NW.AccPerc[df.TotalDKE$NWQuartilePerc=="(0.407,0.444]"]
NW.AccOQ3 <-
df.TotalDKE$NW.AccPerc[df.TotalDKE$NWQuartilePerc=="(0.444,0.556]"]
NW.AccOQ4 <-
df.TotalDKE$NW.AccPerc[df.TotalDKE$NWQuartilePerc=="(0.556,0.704]"]

NW.EstOQ1 <- df.TotalDKE$NW.Confidence.Avg[df.TotalDKE$NWQuartilePerc==
"[0.185,0.407]"]
NW.EstOQ2 <- df.TotalDKE$NW.Confidence.Avg[df.TotalDKE$NWQuartilePerc==
"(0.407,0.444]"]
NW.EstOQ3 <- df.TotalDKE$NW.Confidence.Avg[df.TotalDKE$NWQuartilePerc==
"(0.444,0.556]"]
NW.EstOQ4 <- df.TotalDKE$NW.Confidence.Avg[df.TotalDKE$NWQuartilePerc==
"(0.556,0.704]"]

t.test(NW.AccOQ1, NW.EstOQ1, paired = T)
t.test(NW.AccOQ2, NW.EstOQ2, paired = T)
t.test(NW.AccOQ3, NW.EstOQ3, paired = T)
t.test(NW.AccOQ4, NW.EstOQ4, paired = T)
```

~~~
#min/max calcs

mean(df.TotalDKE$Post.NW.Dif)
sd(df.TotalDKE$Post.NW.Dif)
mean(df.TotalDKE$Post.Gram.Dif)
sd(df.TotalDKE$Post.Gram.Dif)
mean(df.TotalDKE$Post.Log.Dif)
sd(df.TotalDKE$Post.Log.Dif)
min(df.TotalDKE$NW.Acc)
max(df.TotalDKE$NW.Acc)
min(df.TotalDKE$Gram.Acc)
max(df.TotalDKE$Gram.Acc)
min(df.TotalDKE$Logic.Acc)
max(df.TotalDKE$Logic.Acc)
min(df.TotalDKE$Overall.Correct.Answers)
max(df.TotalDKE$Overall.Correct.Answers)
max(df.TotalDKE$Overall.Correct.Answers.Estimate)
mean(df.TotalDKE$Overall.Correct.Answers)
Bestsubtable<-table(df.TotalDKE$Estimated.Best.Subject,
df.TotalDKE$Actual.Best.Subject)
view(Bestsubtable)

#write.csv(df.overallquartilestats, "global_overest.csv")

```

```{r, plotting effect sizes}

#Output table comparing overall models to oneanother
~~~

```
tab_model(OverallOverest.model, OverallOverplacing.model, overprecise.model,
show.std = T, show.fstat = TRUE,  auto.label = FALSE, pred.labels
=c("(Intercept)" ,"Overall.Correct.Answers" = "Skill", "OverallPercentileRank"
= "Skill", "Overall.AccPercent" = "Skill", "Years.Experience"= "Years of
Experience", "Internal" = "Internal LOC", "Chance" ="Chance LOC",
"Powerful.Others" = "Powerful Others LOC","Extraversion" = "Extraversion",
"Agreeableness" = "Agreeableness", "Conscientiousness" = "Conscientiousness",
"Openness" = "Openness to Experience", "Neuroticism"= "Neuroticism",
"OCQ.Bias" = "Overclaiming Bias", "Avoidant" = "Avoidant Decision
Style","Dependent" = "Dependent Decision Style", "Vigilant" = "Vigilant
Decision Style", "Spontaneous" = "Spontaneous Decision Style", "Intuitive" =
"Intuitive Decision Style", "Respected" = "Respected Decision Style",
"Brooding" = "Brooding Decision Style" ,"Social.Desiribility" = "Social
Desirability", "Declarative" = "Metacognitive Declarative Memory",
"Procedural" = "Metacognitive Procedural Memory", "Conditional" =
"Metacognitive Conditional Memory", "Planning" = "Metacognitive Planning",
"Information.Management" = "Metacognitive Information Management",
"Comprehension.Monitoring" = "Metacognitive Comprehension
Monitoring","Debugging.Strategies" = "Metacognitive Debugging Strategies",
"Evaluation" = "Metacognitive Evaluation",  "OverallPostDif" = "Perception of
Difficulty"))

tab_model(logic.model, LogicOverplacing.model, log.op.model, show.std = T,
show.fstat = TRUE,  auto.label = FALSE, pred.labels =c("(Intercept)"
,"Logic.Acc" = "Skill", "LogicPercentileRank" = "Skill", "Logic.AccPerc" =
"Skill", "Years.Experience"= "Years of Experience", "Internal" = "Internal
LOC", "Chance" ="Chance LOC", "Powerful.Others" = "Powerful Others
LOC","Extraversion" = "Extraversion", "Agreeableness" = "Agreeableness",
"Conscientiousness" = "Conscientiousness", "Openness" = "Openness to
Experience", "Neuroticism"= "Neuroticism", "OCQ.Bias" = "Overclaiming Bias",
"Avoidant" = "Avoidant Decision Style","Dependent" = "Dependent Decision
Style", "Vigilant" = "Vigilant Decision Style", "Spontaneous" = "Spontaneous
Decision Style", "Intuitive" = "Intuitive Decision Style", "Respected" =
"Respected Decision Style", "Brooding" = "Brooding Decision Style"
,"Social.Desiribility" = "Social Desirability", "Declarative" = "Metacognitive
Declarative Memory", "Procedural" = "Metacognitive Procedural Memory",
"Conditional" = "Metacognitive Conditional Memory", "Planning" =
"Metacognitive Planning", "Information.Management" = "Metacognitive
Information Management", "Comprehension.Monitoring" = "Metacognitive
Comprehension Monitoring","Debugging.Strategies" = "Metacognitive Debugging
Strategies",  "Evaluation" = "Metacognitive Evaluation",  "Post.Log.Dif" =
"Perception of Difficulty"))

summ(logic.model)
summ(LogicOverplacing.model)
summ(log.op.model)
```

```
#To get tab_model to word, add this: , file = "Overall.Regression.docx"

tab_model(NW.model, NWOverplacing.model, NW.op.model, show.std = T, show.fstat
= TRUE,  auto.label = FALSE, pred.labels =c("(Intercept)" ,"NW.Acc" = "Skill",
"NWPercentileRank" = "Skill", "NW.AccPerc" = "Skill", "Years.Experience"=
"Years of Experience", "Internal" = "Internal LOC", "Chance" ="Chance LOC",
"Powerful.Others" = "Powerful Others LOC","Extraversion" = "Extraversion",
"Agreeableness" = "Agreeableness", "Conscientiousness" = "Conscientiousness",
"Openness" = "Openness to Experience", "Neuroticism"= "Neuroticism",
"OCQ.Bias" = "Overclaiming Bias", "Avoidant" = "Avoidant Decision
Style","Dependent" = "Dependent Decision Style", "Vigilant" = "Vigilant
Decision Style", "Spontaneous" = "Spontaneous Decision Style", "Intuitive" =
"Intuitive Decision Style", "Respected" = "Respected Decision Style",
"Brooding" = "Brooding Decision Style" ,"Social.Desiribility" = "Social
Desirability", "Declarative" = "Metacognitive Declarative Memory",
"Procedural" = "Metacognitive Procedural Memory", "Conditional" =
"Metacognitive Conditional Memory", "Planning" = "Metacognitive Planning",
"Information.Management" = "Metacognitive Information Management",
"Comprehension.Monitoring" = "Metacognitive Comprehension
Monitoring","Debugging.Strategies" = "Metacognitive Debugging Strategies",
"Evaluation" = "Metacognitive Evaluation",  "Post.Log.Dif" = "Perception of
Difficulty"))

summ(NW.model)
summ(NWOverplacing.model)
summ(NW.op.model)

tab_model(grammar.model, GrammarOverplacing.model, gram.op.model, show.std =
T, show.fstat = TRUE,  auto.label = FALSE, pred.labels =c("(Intercept)"
,"Gram.Acc" = "Skill", "GrammarPercentileRank" = "Skill", "Grammar.AccPerc" =
"Skill", "Years.Experience"= "Years of Experience", "Internal" = "Internal
LOC", "Chance" ="Chance LOC", "Powerful.Others" = "Powerful Others
LOC","Extraversion" = "Extraversion", "Agreeableness" = "Agreeableness",
"Conscientiousness" = "Conscientiousness", "Openness" = "Openness to
Experience", "Neuroticism"= "Neuroticism", "OCQ.Bias" = "Overclaiming Bias",
"Avoidant" = "Avoidant Decision Style","Dependent" = "Dependent Decision
Style", "Vigilant" = "Vigilant Decision Style", "Spontaneous" = "Spontaneous
Decision Style", "Intuitive" = "Intuitive Decision Style", "Respected" =
"Respected Decision Style", "Brooding" = "Brooding Decision Style"
,"Social.Desiribility" = "Social Desirability", "Declarative" = "Metacognitive
Declarative Memory", "Procedural" = "Metacognitive Procedural Memory",
"Conditional" = "Metacognitive Conditional Memory", "Planning" =
"Metacognitive Planning", "Information.Management" = "Metacognitive
Information Management", "Comprehension.Monitoring" = "Metacognitive
Comprehension Monitoring","Debugging.Strategies" = "Metacognitive Debugging
Strategies",  "Evaluation" = "Metacognitive Evaluation",  "Post.Log.Dif" =
"Perception of Difficulty"))
```

```
summ(grammar.model)
summ(GrammarOverplacing.model)
summ(gram.op.model)
```

```
plot_summs(OverallOverest.model, scale = T, colors = "orange", coefs =
c("Overall Skill (Questions Correct)" = "Overall.Correct.Answers", "Years of
Experience" ="Years.Experience", "Internal LOC"= "Internal", "Chance LOC" =
"Chance", "Powerful Others LOC"= "Powerful.Others","Extraversion" =
"Extraversion", "Agreeableness" = "Agreeableness", "Conscientiousness" =
"Conscientiousness", "Openness to Experience" = "Openness", "Neuroticism"=
"Neuroticism", "Overclaiming Bias" = "OCQ.Bias", "Avoidant Decision Style" =
"Avoidant","Dependent Decision Style" = "Dependent", "Vigilant Decision Style"
= "Vigilant", "Spontaneous Decision Style"= "Spontaneous", "Intuitive Decision
Style" = "Intuitive", "Respected Decision Style" = "Respected", "Brooding
Decision Style" = "Brooding" , "Social Desirability" = "Social.Desiribility",
"Metacognitive Declarative Memory" = "Declarative", "Metacognitive Procedural
Memory" = "Procedural", "Metacognitive Conditional Memory" = "Conditional",
"Metacognitive Planning" = "Planning", "Metacognitive Information Management"
= "Information.Management", "Metacognitive Comprehension Monitoring" =
"Comprehension.Monitoring","Metacognitive Debugging Strategies" =
"Debugging.Strategies",  "Metacognitive Evaluation" = "Evaluation",
"Perception of Difficulty" = "OverallPostDif"))

plot_summs(OverallOverplacing.model, scale = T, coefs = c("Overall Skill
(Percentile Ranking)" = "OverallPercentileRank", "Years of Experience"
="Years.Experience", "Internal LOC"= "Internal", "Chance LOC" = "Chance",
"Powerful Others LOC"= "Powerful.Others","Extraversion" = "Extraversion",
"Agreeableness" = "Agreeableness", "Conscientiousness" = "Conscientiousness",
"Openness to Experience" = "Openness", "Neuroticism"= "Neuroticism",
"Overclaiming Bias" = "OCQ.Bias", "Avoidant Decision Style" =
"Avoidant","Dependent Decision Style" = "Dependent", "Vigilant Decision Style"
= "Vigilant", "Spontaneous Decision Style"= "Spontaneous", "Intuitive Decision
Style" = "Intuitive", "Respected Decision Style" = "Respected", "Brooding
Decision Style" = "Brooding" , "Social Desirability" = "Social.Desiribility",
"Metacognitive Declarative Memory" = "Declarative", "Metacognitive Procedural
Memory" = "Procedural", "Metacognitive Conditional Memory" = "Conditional",
"Metacognitive Planning" = "Planning", "Metacognitive Information Management"
= "Information.Management", "Metacognitive Comprehension Monitoring" =
"Comprehension.Monitoring","Metacognitive Debugging Strategies" =
"Debugging.Strategies",  "Metacognitive Evaluation" = "Evaluation",
"Perception of Difficulty" = "OverallPostDif"))
```

```
plot_summs(overprecise.model, scale = T, colors= "purple", coefs = c("Overall
Skill (% Correct)" = "Overall.AccPercent", "Years of Experience"
="Years.Experience", "Internal LOC"= "Internal", "Chance LOC" = "Chance",
"Powerful Others LOC"= "Powerful.Others","Extraversion" = "Extraversion",
"Agreeableness" = "Agreeableness", "Conscientiousness" = "Conscientiousness",
"Openness to Experience" = "Openness", "Neuroticism"= "Neuroticism",
"Overclaiming Bias" = "OCQ.Bias", "Avoidant Decision Style" =
"Avoidant","Dependent Decision Style" = "Dependent", "Vigilant Decision Style"
= "Vigilant", "Spontaneous Decision Style"= "Spontaneous", "Intuitive Decision
Style" = "Intuitive", "Respected Decision Style" = "Respected", "Brooding
Decision Style" = "Brooding" , "Social Desirability" = "Social.Desiribility",
"Metacognitive Declarative Memory" = "Declarative", "Metacognitive Procedural
Memory" = "Procedural", "Metacognitive Conditional Memory" = "Conditional",
"Metacognitive Planning" = "Planning", "Metacognitive Information Management"
= "Information.Management", "Metacognitive Comprehension Monitoring" =
"Comprehension.Monitoring","Metacognitive Debugging Strategies" =
"Debugging.Strategies",  "Metacognitive Evaluation" = "Evaluation",
"Perception of Difficulty" = "OverallPostDif"))


plot_summs(logic.model, scale = T, coefs = c("Logic Skill (# Correct)" =
"Logic.Acc", "Years of Experience" ="Years.Experience", "Internal LOC"=
"Internal", "Chance LOC" = "Chance", "Powerful Others LOC"=
"Powerful.Others","Extraversion" = "Extraversion", "Agreeableness" =
"Agreeableness", "Conscientiousness" = "Conscientiousness", "Openness to
Experience" = "Openness", "Neuroticism"= "Neuroticism", "Overclaiming Bias" =
"OCQ.Bias", "Avoidant Decision Style" = "Avoidant","Dependent Decision Style"
= "Dependent", "Vigilant Decision Style" = "Vigilant", "Spontaneous Decision
Style"= "Spontaneous", "Intuitive Decision Style" = "Intuitive", "Respected
Decision Style" = "Respected", "Brooding Decision Style" = "Brooding" ,
"Social Desirability" = "Social.Desiribility", "Metacognitive Declarative
Memory" = "Declarative", "Metacognitive Procedural Memory" = "Procedural",
"Metacognitive Conditional Memory" = "Conditional", "Metacognitive Planning" =
"Planning", "Metacognitive Information Management" = "Information.Management",
"Metacognitive Comprehension Monitoring" =
"Comprehension.Monitoring","Metacognitive Debugging Strategies" =
"Debugging.Strategies",  "Metacognitive Evaluation" = "Evaluation",  "Logic
Perception of Difficulty" = "Post.Log.Dif"))

png("OverestModelsForest.png", width=900, height=800)
```

```
plot_summs(OverallOverest.model, logic.model, grammar.model, NW.model, scale =
T,  coefs = c("Skill (# Correct)" = "Logic.Acc","Skill (# Correct)" =
"Overall.Correct.Answers", "Skill (# Correct)" = "Gram.Acc","Skill (#
Correct)" = "NW.Acc", "Years of Experience" ="Years.Experience", "Internal
LOC"= "Internal", "Chance LOC" = "Chance", "Powerful Others LOC"=
"Powerful.Others","Extraversion" = "Extraversion", "Agreeableness" =
"Agreeableness", "Conscientiousness" = "Conscientiousness", "Openness to
Experience" = "Openness", "Neuroticism"= "Neuroticism", "Overclaiming Bias" =
"OCQ.Bias", "Avoidant Decision Style" = "Avoidant","Dependent Decision Style"
= "Dependent", "Vigilant Decision Style" = "Vigilant", "Spontaneous Decision
Style"= "Spontaneous", "Intuitive Decision Style" = "Intuitive", "Respected
Decision Style" = "Respected", "Brooding Decision Style" = "Brooding" ,
"Social Desirability" = "Social.Desiribility", "Metacognitive Declarative
Memory" = "Declarative", "Metacognitive Procedural Memory" = "Procedural",
"Metacognitive Conditional Memory" = "Conditional", "Metacognitive Planning" =
"Planning", "Metacognitive Information Management" = "Information.Management",
"Metacognitive Comprehension Monitoring" =
"Comprehension.Monitoring","Metacognitive Debugging Strategies" =
"Debugging.Strategies",  "Metacognitive Evaluation" = "Evaluation",
"Perception of Difficulty" = "Post.Log.Dif", "Perception of Difficulty" =
"OverallPostDif", "Perception of Difficulty" = "Post.Gram.Dif", "Perception of
Difficulty" = "Post.NW.Dif"), model.names = c("Overall Overestimation", "Logic
Overestimation", "Grammar Overestimation", "NW Overestimation"))

apatheme=theme_bw()+
  theme(panel.grid.major=element_blank(),
        panel.grid.minor=element_blank(),
        panel.border=element_blank(),
        axis.line=element_line(),
        text=element_text(family='Helvetica'),
        legend.title=element_blank(),
        axis.text=element_text(size=12),
        axis.title=element_text(size=12),
        legend.text = element_text(size = 12))

png("OverestModelsForest.png", width=900, height=800)
plot.new()
Overestimation_Coeff + apatheme +labs(x = "\n Beta Estimate \n ", y = NULL) +
title("Relationships Between Individual Traits and Overestimation Across
Tasks")
dev.copy(png, "OverestModelsForest.png")
dev.off()

effect_plot(OverallOverest.model, pred = Years.Experience, interval = TRUE,
plot.points = TRUE)

# #Overestimation
# OverallOverest2.model<-summ(OverallOverest.model, scale = T, confint = T,
part.corr = TRUE)
# summ(logic.model, scale = T, confint = T, part.corr = TRUE)
# summ(grammar.model, scale = T, confint = T, part.corr = TRUE)
# summ(NW.model, scale = T, confint = T, part.corr = TRUE)
#
# #Full Model
# coef_names= c("(Intercept)" = "(Intercept)", "Skill (# Correct)" =
"Logic.Acc","Skill (# Correct)" = "Overall.Correct.Answers", "Skill (#
Correct)" = "Gram.Acc","Skill (# Correct)" = "NW.Acc", "Years of Experience"
="Years.Experience", "Internal LOC"= "Internal", "Chance LOC" = "Chance",
"Powerful Others LOC"= "Powerful.Others","Extraversion" = "Extraversion",
"Agreeableness" = "Agreeableness", "Conscientiousness" = "Conscientiousness",
"Openness to Experience" = "Openness", "Neuroticism"= "Neuroticism",
"Overclaiming Bias" = "OCQ.Bias", "Avoidant Decision Style" =
"Avoidant","Dependent Decision Style" = "Dependent", "Vigilant Decision Style"
= "Vigilant", "Spontaneous Decision Style"= "Spontaneous", "Intuitive Decision
Style" = "Intuitive", "Respected Decision Style" = "Respected", "Brooding
Decision Style" = "Brooding"  , "Social Desirability" = "Social.Desiribility",
"Metacognitive Declarative Memory" = "Declarative", "Metacognitive Procedural
Memory" = "Procedural", "Metacognitive Conditional Memory" = "Conditional",
"Metacognitive Planning" = "Planning", "Metacognitive Information Management"
= "Information.Management", "Metacognitive Comprehension Monitoring" =
"Comprehension.Monitoring","Metacognitive Debugging Strategies" =
"Debugging.Strategies",  "Metacognitive Evaluation" = "Evaluation",
"Perception of Difficulty" = "Post.Log.Dif", "Perception of Difficulty" =
"OverallPostDif", "Perception of Difficulty" = "Post.Gram.Dif", "Perception of
Difficulty" = "Post.NW.Dif")
#
```

```
# coef_names3= c("(Intercept)" = "(Intercept)", "Skill" =
"Overall.Correct.Answers", "Skill" = "OverallPercentileRank" , "Skill" =
"Overall.AccPercent","Years of Experience" ="Years.Experience", "Internal
LOC"= "Internal", "Chance LOC" = "Chance", "Powerful Others LOC"=
"Powerful.Others","Extraversion" = "Extraversion", "Agreeableness" =
"Agreeableness", "Conscientiousness" = "Conscientiousness", "Openness to
Experience" = "Openness", "Neuroticism"= "Neuroticism", "Overclaiming Bias" =
"OCQ.Bias", "Avoidant Decision Style" = "Avoidant","Dependent Decision Style"
= "Dependent", "Vigilant Decision Style" = "Vigilant", "Spontaneous Decision
Style"= "Spontaneous", "Intuitive Decision Style" = "Intuitive", "Respected
Decision Style" = "Respected", "Brooding Decision Style" = "Brooding" ,
"Social Desirability" = "Social.Desiribility", "Metacognitive Declarative
Memory" = "Declarative", "Metacognitive Procedural Memory" = "Procedural",
"Metacognitive Conditional Memory" = "Conditional", "Metacognitive Planning" =
"Planning", "Metacognitive Information Management" = "Information.Management",
"Metacognitive Comprehension Monitoring" =
"Comprehension.Monitoring","Metacognitive Debugging Strategies" =
"Debugging.Strategies",  "Metacognitive Evaluation" = "Evaluation",
"Perception of Difficulty" = "OverallPostDif")
#
# #huxreg takes the output of the linear regression model and puts it into a
really nice table
# OverestCompare.ht<-huxreg("Overall Overestestimation" =
OverallOverest2.model, "Logic Overestimation" = logic.model, "Grammar
Overestimation" = grammar.model, "NW Overestimation" = NW.model, ci_level = .
99, coefs = coef_names, statistics = c("N. obs." = "nobs", "R squared" =
"r.squared", "Adj. R Squared" = "adj.r.squared", "F statistic" = "statistic",
#       "P value" = "p.value"), bold_signif = 0.05)
#
# Overestft <- as_flextable(OverestCompare.ht)
# ## Not run:
# Overestft<- autofit(Overestft)
#   my_doc <- officer::read_docx()
#   my_doc <- flextable::body_add_flextable(
#         my_doc, Overestft)
#   print(my_doc, target =
#         "OverestCompare2.docx")
#
# OverestCompare.tb<-export_summs(
# OverallOverest.model, OverallOverplacing.model, overprecise.model,
#   error_format = "({std.error})",
#   error_pos = c("below"),
#   ci_level = 0.95,
#   model.names =  c("Overall Overestimation", "Overall Overplacing", "Overall
Overprecision"),
#   coefs = coef_names3,
#   file.name = "OverestCompare2.docx",
# bold_signif = 0.05,
```

~~~
# statistics = c("N. obs." = "nobs", "R squared" = "r.squared", "Adj. R
Squared" = "adj.r.squared", "F statistic" = "statistic")
# )
#
#
#
# huxtable::quick_docx(OverestCompare.tb, file = "OverestCompareMod.docx")


# #Only Significant
# coef_names2= c("(Intercept)" = "(Intercept)", "Skill (# Correct)" =
"Overall.Correct.Answers", "Skill" = "OverallPercentileRank" , "Skill" =
"Overall.AccPercent", "Years of Experience" ="Years.Experience", "Internal
LOC"= "Internal", "Chance LOC" = "Chance", "Openness to Experience" =
"Openness", "Neuroticism"= "Neuroticism", "Dependent Decision Style" =
"Dependent", "Metacognitive Debugging Strategies" = "Debugging.Strategies",
"Perception of Difficulty" = "Post.Log.Dif", "Perception of Difficulty" =
"OverallPostDif")
#
# df.TotalDKE$Overall.AccPercent
#
# #huxreg takes the output of the linear regression model and puts it into a
really nice table
# OverestCompare2.ht<-huxreg("Overall Overestestimation" =
OverallOverest2.model, "Logic Overestimation" = logic.model, "Grammar
Overestimation" = grammar.model, "NW Overestimation" = NW.model, ci_level = .
99, coefs = coef_names2, statistics = c("N. obs." = "nobs", "R squared" =
"r.squared", "Adj. R Squared" = "adj.r.squared", "F statistic" = "statistic",
#       "P value" = "p.value"), bold_signif = 0.05)
#
# Overestft2 <- as_flextable(OverestCompare2.ht)
# ## Not run:
# Overestft<- autofit(Overestft2)
#   my_doc <- officer::read_docx()
#   my_doc <- flextable::body_add_flextable(
#         my_doc, Overestft2)
#   print(my_doc, target =
#         "OverestCompareshort.docx")
#
# #Overplacing
#
# summ(OverallOverplacing.model, scale = T, confint = T, part.corr = TRUE)
# summ(overprecise.model, scale = T, confint = T)


```

```{r, mediation testing}
~~~

```
#Sobel Method
# mediation test (mediating variable, IV, DV)

#Overall
mediation.test(df.TotalDKE$Years.Experience,
df.TotalDKE$Overall.Correct.Answers, df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Internal, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Chance, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Powerful.Others,
df.TotalDKE$Overall.Correct.Answers, df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Extraversion, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Agreeableness, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Conscientiousness,
df.TotalDKE$Overall.Correct.Answers, df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Openness, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Neuroticism, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$OCQ.Bias, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$Avoidant, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Dependent, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Vigilant, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Spontaneous, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Intuitive, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Respected, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Brooding, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$Social.Desiribility,
df.TotalDKE$Overall.Correct.Answers, df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$Declarative, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Procedural, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
```

```
mediation.test(df.TotalDKE$Conditional, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Planning, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Information.Management,
df.TotalDKE$Overall.Correct.Answers, df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Comprehension.Monitoring,
df.TotalDKE$Overall.Correct.Answers, df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Debugging.Strategies,
df.TotalDKE$Overall.Correct.Answers, df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Evaluation, df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Correct.Answers, df.TotalDKE$Overall.Overestimation.Score)

#Difficulty


mediation.test(df.TotalDKE$Years.Experience, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$Internal, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Chance, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Powerful.Others, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$Extraversion, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Agreeableness, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Conscientiousness, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Openness, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Neuroticism, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$OCQ.Bias, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$Avoidant, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Dependent, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
```

```
mediation.test(df.TotalDKE$Vigilant, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Spontaneous, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Intuitive, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Respected, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Brooding, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$Social.Desiribility, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$Declarative, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Procedural, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Conditional, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Planning, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Information.Management, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Comprehension.Monitoring,
df.TotalDKE$OverallPostDif, df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Debugging.Strategies, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)
mediation.test(df.TotalDKE$Evaluation, df.TotalDKE$OverallPostDif,
df.TotalDKE$Overall.Overestimation.Score)

mediation.test(df.TotalDKE$Overall.Correct.Answers,
df.TotalDKE$OverallPostDif, df.TotalDKE$Overall.Overestimation.Score)

Overall.AccPercent

#Overplacing
mediation.test(df.TotalDKE$Years.Experience,
df.TotalDKE$OverallPercentileRank, df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$Internal, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Chance, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Powerful.Others, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
```

```
mediation.test(df.TotalDKE$Extraversion, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Agreeableness, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Conscientiousness,
df.TotalDKE$OverallPercentileRank, df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Openness, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Neuroticism, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$OCQ.Bias, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$Avoidant, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Dependent, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Vigilant, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Spontaneous, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Intuitive, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Respected, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Brooding, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$Social.Desiribility,
df.TotalDKE$OverallPercentileRank, df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$Declarative, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Procedural, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Conditional, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Planning, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Information.Management,
df.TotalDKE$OverallPercentileRank, df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Comprehension.Monitoring,
df.TotalDKE$OverallPercentileRank, df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Debugging.Strategies,
df.TotalDKE$OverallPercentileRank, df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Evaluation, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)
```

```
mediation.test(df.TotalDKE$OverallPostDif, df.TotalDKE$OverallPercentileRank,
df.TotalDKE$OverallOverplacing)

#Difficulty

mediation.test(df.TotalDKE$Years.Experience, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$Internal, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Chance, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Powerful.Others, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$Extraversion, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Agreeableness, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Conscientiousness, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Openness, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Neuroticism, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$OCQ.Bias, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$Avoidant, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Dependent, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Vigilant, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Spontaneous, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Intuitive, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Respected, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Brooding, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$Social.Desiribility, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
```

```
mediation.test(df.TotalDKE$Declarative, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Procedural, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Conditional, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Planning, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Information.Management, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Comprehension.Monitoring,
df.TotalDKE$OverallPostDif, df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Debugging.Strategies, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)
mediation.test(df.TotalDKE$Evaluation, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)

mediation.test(df.TotalDKE$OverallPercentileRank, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverplacing)

#Overprecision

mediation.test(df.TotalDKE$Years.Experience, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$Internal, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Chance, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Powerful.Others, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$Extraversion, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Agreeableness, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Conscientiousness, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Openness, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Neuroticism, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$OCQ.Bias, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$Avoidant, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
```

```
mediation.test(df.TotalDKE$Dependent, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Vigilant, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Spontaneous, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Intuitive, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Respected, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Brooding, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$Social.Desiribility,
df.TotalDKE$Overall.AccPercent, df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$Declarative, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Procedural, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Conditional, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Planning, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Information.Management,
df.TotalDKE$Overall.AccPercent, df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Comprehension.Monitoring,
df.TotalDKE$Overall.AccPercent, df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Debugging.Strategies,
df.TotalDKE$Overall.AccPercent, df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Evaluation, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$OverallPostDif, df.TotalDKE$Overall.AccPercent,
df.TotalDKE$OverallOverPrecision)

#Difficulty

mediation.test(df.TotalDKE$Years.Experience, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$Internal, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Chance, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Powerful.Others, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
```

```
mediation.test(df.TotalDKE$Extraversion, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Agreeableness, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Conscientiousness, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Openness, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Neuroticism, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$OCQ.Bias, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$Avoidant, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Dependent, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Vigilant, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Spontaneous, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Intuitive, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Respected, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Brooding, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$Social.Desiribility, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)

mediation.test(df.TotalDKE$Declarative, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Procedural, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Conditional, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Planning, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Information.Management, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Comprehension.Monitoring,
df.TotalDKE$OverallPostDif, df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Debugging.Strategies, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
mediation.test(df.TotalDKE$Evaluation, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)
```

~~~
mediation.test(df.TotalDKE$Overall.AccPercent, df.TotalDKE$OverallPostDif,
df.TotalDKE$OverallOverPrecision)


```

```{r, moderation analysis - task difficulty}

# #moderate.lm(IV, mod, DV, data, mc = FALSE)
#
# #Overplacing
#
# Years.mod <- moderate.lm(OverallPercentileRank, Years.Experience,
OverallOverplacing, data=df.TotalDKE, mc = FALSE)
# summary(Years.mod)
```

```{r, moderation analysis}

#Overprecision
overprecise.model <- lm(OverallOverPrecision ~ Overall.AccPercent +
Years.Experience + Internal + Chance + Powerful.Others + Extraversion +
Agreeableness + Conscientiousness + Openness + Neuroticism + OCQ.Bias +
Avoidant + Dependent + Vigilant + Spontaneous + Intuitive + Respected +
Brooding + Social.Desiribility + Declarative + Procedural + Conditional +
Planning + Information.Management + Comprehension.Monitoring +
Debugging.Strategies + Evaluation + OverallPostDif, data=df.TotalDKE)
summ(overprecise.model)

interact_plot(overprecise.model, pred =Overall.AccPercent , modx =
OverallPostDif, plot.points = T)

OverallOverest.model<- lm(Overall.Overestimation.Score~
Overall.Correct.Answers + Years.Experience + Internal + Chance +
Powerful.Others + Extraversion + Agreeableness + Conscientiousness + Openness
+ Neuroticism + OCQ.Bias + Avoidant + Dependent + Vigilant + Spontaneous +
Intuitive + Respected + Brooding + Social.Desiribility + Declarative +
Procedural + Conditional + Planning + Information.Management +
Comprehension.Monitoring + Debugging.Strategies + Evaluation + OverallPostDif,
data=df.TotalDKE)
summ(OverallOverest.model)

interact_plot(OverallOverest.model, pred =Overall.Correct.Answers , modx =
OverallPostDif, plot.points = T)
~~~

~~~
OverallOverplacing.model<- lm(OverallOverplacing~ OverallPercentileRank +
Years.Experience + Internal + Chance + Powerful.Others + Extraversion +
Agreeableness + Conscientiousness + Openness + Neuroticism + OCQ.Bias +
Avoidant + Dependent + Vigilant + Spontaneous + Intuitive + Respected +
Brooding + Social.Desiribility +  Declarative + Procedural + Conditional +
Planning + Information.Management + Comprehension.Monitoring +
Debugging.Strategies + Evaluation + OverallPostDif,  data=df.TotalDKE)
summ(OverallOverplacing.model)
interact_plot(OverallOverplacing.model, pred = OverallPercentileRank , modx =
OverallPostDif, plot.points = T)

```

```{r, reimaging graphs as bars}

#This function allows for calculation of 95% confidence intervals. x is the
data frame.
errorBarFunc <- function(x) {
  output <- c(mean(x) - (1.96*((sd(x))/(sqrt(length(x))))), mean(x) +
(1.96*((sd(x))/(sqrt(length(x))))))
  names(output) <- c("ymin", "ymax")
  output

}

#Overall Plots

df.OverallPlot<- df.TotalDKE[c(73,74,262)]
#view(df.OverallPlot)
df.OverallPlot<- melt(df.OverallPlot, id.vars='OverallQuantile')

OverallOverest<-ggplot(df.OverallPlot, aes(x=OverallQuantile, y=value,
fill=variable))+ stat_summary(fun=mean, geom = "bar", position="dodge",
size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 8, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("Overall Global Overestimation (Out of 81 Questions)")+
scale_fill_discrete(name = "Legend", labels = c("Questions Answered
Correctly", "Estimated Answered Correctly"))
#ggsave("OverallGlobalBar.png")

#Overprecision
##OverallAccPercent & Overall.Confidence.Average
df.TotalDKE$Overall.Confidence.Average<- apply(df.conf, 1, mean, na.rm =T)
~~~

```
df.OverallOverpecisePlot<-
df.TotalDKE[c("Overall.AccPercent","Overall.Confidence.Average",
"OverallQuartilePerc")]
df.OverallOverpecisePlot<- melt(df.OverallOverpecisePlot,
id.vars='OverallQuartilePerc')
#view(df.OverallOverpecisePlot)

OverallOverprecise<-ggplot(df.OverallOverpecisePlot,
aes(x=OverallQuartilePerc, y=value, fill=variable))+ stat_summary(fun=mean,
geom = "bar", position="dodge", size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 5, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("Overall Overprecision (in %)")+  theme(plot.title =
element_text(hjust = 0.5)) + scale_fill_discrete(name = "Legend", labels = c(
"Percentage Correct", "Estimated Percentage Correct"))
#ggsave("OverallOverprecisionBar.png")

#Overplacing
##OverallPercentileRank, OverallEstimatedPercentile, OverallQuartileRank
df.OverallOverplacingPlot<- df.TotalDKE[c("OverallPercentileRank",
"OverallEstimatedPercentile", "OverallQuartileRank")]
df.OverallOverplacingPlot<- melt(df.OverallOverplacingPlot,
id.vars='OverallQuartileRank')
#view(df.OverallOverplacingPlot)

OverallOverplace<-ggplot(df.OverallOverplacingPlot, aes(x=OverallQuartileRank,
y=value, fill=variable))+ stat_summary(fun=mean, geom = "bar",
position="dodge", size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 4.5, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("Overall Overplacing (Percentile Rank)")+  theme(plot.title =
element_text(hjust = 0.5)) + scale_fill_discrete(name = "Legend", labels = c(
"Achieved Percentile Rank", "Estimated Percentile Rank"))

#ggsave("OverallOverplacingBar.png")
ggarrange(OverallOverplace, OverallOverest, OverallOverprecise,
                    labels = c("A", "B", "C"),
                    ncol = 2, nrow = 2)
ggsave("OverallFig.png", width = 14, height = 12)
```

~~~
```
```{r, reimaging graphs as bars - Logic}


#Logic Plots
df.TotalDKE$Post.Log.Est
df.LogicPlot<- df.TotalDKE[c("Logic.Acc", "Post.Log.Est", "LogicQuantile")]
#view(df.LogicPlot)
df.LogicPlot<- melt(df.LogicPlot, id.vars='LogicQuantile')

LogicOverest<-ggplot(df.LogicPlot, aes(x=LogicQuantile, y=value,
fill=variable))+ stat_summary(fun=mean, geom = "bar", position="dodge",
size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 8, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("Logic Global Overestimation (Out of 81 Questions)")+
scale_fill_discrete(name = "Legend", labels = c("Questions Answered
Correctly", "Estimated Answered Correctly"))
#ggsave("LogicGlobalBar.png")

#Overprecision
##LogicAccPercent & Logic.Confidence.Average
df.LogicOverpecisePlot<- df.TotalDKE[c("Logic.AccPerc","Logic.Confidence.Avg",
"LogicQuartilePerc")]
df.LogicOverpecisePlot<- melt(df.LogicOverpecisePlot,
id.vars='LogicQuartilePerc')
#view(df.LogicOverpecisePlot)

LogicOverprecise<-ggplot(df.LogicOverpecisePlot, aes(x=LogicQuartilePerc,
y=value, fill=variable))+ stat_summary(fun=mean, geom = "bar",
position="dodge", size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 5, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("Logic Overprecision (in %)")+  theme(plot.title = element_text(hjust
= 0.5)) + scale_fill_discrete(name = "Legend", labels = c( "Percentage
Correct", "Estimated Percentage Correct"))
#ggsave("LogicOverprecisionBar.png")

#Overplacing
##LogicPercentileRank, LogicEstimatedPercentile, LogicQuartileRank
~~~

~~~
df.LogicOverplacingPlot<- df.TotalDKE[c("LogicPercentileRank",
"LogicEstimatedPercentile", "LogicQuartileRank")]
df.LogicOverplacingPlot<- melt(df.LogicOverplacingPlot,
id.vars='LogicQuartileRank')
#view(df.LogicOverplacingPlot)

LogicOverplace<-ggplot(df.LogicOverplacingPlot, aes(x=LogicQuartileRank,
y=value, fill=variable))+ stat_summary(fun=mean, geom = "bar",
position="dodge", size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 5.5, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("Logic Overplacing (Percentile Rank)")+  theme(plot.title =
element_text(hjust = 0.5)) + scale_fill_discrete(name = "Legend", labels = c(
"Achieved Percentile Rank", "Estimated Percentile Rank"))

#ggsave("LogicOverplacingBar.png")
ggarrange(LogicOverplace, LogicOverest, LogicOverprecise,
                    labels = c("A", "B", "C"),
                    ncol = 2, nrow = 2)
ggsave("LogicFig.png", width = 14, height = 12)

```

```{r, reimaging graphs as bars - Grammar}

#Grammar Plots
df.TotalDKE$Post.Log.Est
df.GrammarPlot<- df.TotalDKE[c("Gram.Acc", "Post.Gram.Est",
"GrammarQuantile")]
#view(df.GrammarPlot)
df.GrammarPlot<- melt(df.GrammarPlot, id.vars='GrammarQuantile')

GrammarOverest<-ggplot(df.GrammarPlot, aes(x=GrammarQuantile, y=value,
fill=variable))+ stat_summary(fun=mean, geom = "bar", position="dodge",
size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 6, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("Grammar Global Overestimation (Out of 81 Questions)")+
scale_fill_discrete(name = "Legend", labels = c("Questions Answered
Correctly", "Estimated Answered Correctly"))
~~~

```
#ggsave("GrammarGlobalBar.png")

#Overprecision
##GrammarAccPercent & Grammar.Confidence.Average
df.GrammarOverpecisePlot<-
df.TotalDKE[c("Gram.AccPerc","Grammar.Confidence.Avg", "GrammarQuartilePerc")]
df.GrammarOverpecisePlot<- melt(df.GrammarOverpecisePlot,
id.vars='GrammarQuartilePerc')
#view(df.GrammarOverpecisePlot)

GrammarOverprecise<-ggplot(df.GrammarOverpecisePlot,
aes(x=GrammarQuartilePerc, y=value, fill=variable))+ stat_summary(fun=mean,
geom = "bar", position="dodge", size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 5, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("Grammar Overprecision (in %)")+  theme(plot.title =
element_text(hjust = 0.5)) + scale_fill_discrete(name = "Legend", labels = c(
"Percentage Correct", "Estimated Percentage Correct"))
#ggsave("GrammarOverprecisionBar.png")

#Overplacing
##GrammarPercentileRank, GrammarEstimatedPercentile, GrammarQuartileRank
df.GrammarOverplacingPlot<- df.TotalDKE[c("GrammarPercentileRank",
"GrammarEstimatedPercentile", "GrammarQuartileRank")]
df.GrammarOverplacingPlot<- melt(df.GrammarOverplacingPlot,
id.vars='GrammarQuartileRank')
#view(df.GrammarOverplacingPlot)

GrammarOverplace<-ggplot(df.GrammarOverplacingPlot, aes(x=GrammarQuartileRank,
y=value, fill=variable))+ stat_summary(fun=mean, geom = "bar",
position="dodge", size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 5.5, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("Grammar Overplacing (Percentile Rank)")+  theme(plot.title =
element_text(hjust = 0.5)) + scale_fill_discrete(name = "Legend", labels = c(
"Achieved Percentile Rank", "Estimated Percentile Rank"))

#ggsave("GrammarOverplacingBar.png")
ggarrange(GrammarOverplace, GrammarOverest, GrammarOverprecise,
                    labels = c("A", "B", "C"),
                    ncol = 2, nrow = 2)
ggsave("GrammarFig.png", width = 14, height = 12)

```
```{r, reimaging graphs as bars - NW}

#NW Plots
df.TotalDKE$Post.Log.Est
df.NWPlot<- df.TotalDKE[c("NW.Acc", "Post.NW.Est", "NWQuantile")]
#view(df.NWPlot)
df.NWPlot<- melt(df.NWPlot, id.vars='NWQuantile')

NWOverest<-ggplot(df.NWPlot, aes(x=NWQuantile, y=value, fill=variable))+
stat_summary(fun=mean, geom = "bar", position="dodge", size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 8.3, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("NW Global Overestimation (Out of 81 Questions)")+
scale_fill_discrete(name = "Legend", labels = c("Questions Answered
Correctly", "Estimated Answered Correctly"))
#ggsave("NWGlobalBar.png")

#Overprecision
##NWAccPercent & NW.Confidence.Average
df.NWOverpecisePlot<- df.TotalDKE[c("NW.AccPerc","NW.Confidence.Avg",
"NWQuartilePerc")]
df.NWOverpecisePlot<- melt(df.NWOverpecisePlot, id.vars='NWQuartilePerc')
#view(df.NWOverpecisePlot)

NWOverprecise<-ggplot(df.NWOverpecisePlot, aes(x=NWQuartilePerc, y=value,
fill=variable))+ stat_summary(fun=mean, geom = "bar", position="dodge",
size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 7, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("NW Overprecision (in %)")+  theme(plot.title = element_text(hjust =
0.5)) + scale_fill_discrete(name = "Legend", labels = c( "Percentage Correct",
"Estimated Percentage Correct"))
#ggsave("NWOverprecisionBar.png")

#Overplacing
##NWPercentileRank, NWEstimatedPercentile, NWQuartileRank
```

~~~
df.NWOverplacingPlot<- df.TotalDKE[c("NWPercentileRank",
"NWEstimatedPercentile", "NWQuartileRank")]
df.NWOverplacingPlot<- melt(df.NWOverplacingPlot, id.vars='NWQuartileRank')
#view(df.NWOverplacingPlot)

NWOverplace<-ggplot(df.NWOverplacingPlot, aes(x=NWQuartileRank, y=value,
fill=variable))+ stat_summary(fun=mean, geom = "bar", position="dodge",
size=1.0) +
stat_summary(fun.data="errorBarFunc", geom="errorbar", color="black",
width=0.2, position = position_dodge(width=0.9) ) +
stat_summary(aes(label=round(..y..,2)), fun=mean, geom="text", size = 4, vjust
= 6, color = "black", position=position_dodge(width=0.9))+
labs(y="Estimated/Achieved Accuracy", x = "") +  scale_x_discrete(labels =
c("Quartile 1", "Quartile 2", "Quartile 3", "Quartile 4")) + theme_bw() +
ggtitle("NW Overplacing (Percentile Rank)")+  theme(plot.title =
element_text(hjust = 0.5)) + scale_fill_discrete(name = "Legend", labels = c(
"Achieved Percentile Rank", "Estimated Percentile Rank"))

#ggsave("NWOverplacingBar.png")
ggarrange(NWOverplace, NWOverest, NWOverprecise,
                    labels = c("A", "B", "C"),
                    ncol = 2, nrow = 2)
ggsave("NWFig.png", width = 14, height = 12)

```
~~~