ETM640ProjectDraft

Randolph Anderson

March 4, 2019

Introduction

The 532d Training Squadron (532 TRS) is a unit in the United States Air Force (USAF). The 532 TRS trains the nation's intercontinental ballistic missile (ICBM) operators. The unit also trains maintenance forces for ICBM and air launched cruise missile (ALCM) systems. 532 TRS instructors receive professional development (ProD) to increase their knowledge of nuclear enterprise.
Professional development involves sending one or more nstructors on a trip to a nuclear facility in the United States.While on their trip, instructors receive tours and briefings from onsite personnel. One purpose of providing professional development is to increase effectiveness of instructors. Instructors can help motivate students by illuminating the importance of what students are learning. However, instructors also perform jobs that do not involve actively teaching students. When in these other jobs instructors are not in the best position to integrate what they learned into their instruction.

Problem Definition

Some instructors do not receive professional development until most of their instructing time is over. As a result, they never get a chance to share what they have learned with the students. Since the goal of ProD is, in part, to increase the effectiveness of instructors, the unit may be able to do a better job assigning instructors to professional development trip in a way that balances several objectives (goals).

Objectives/Goals

The squadron has several objectives for the professional development program. Objectives include:

1. (Budget) ProD budget should fully utilize remaining travel funds after all mission required trips are funded; ProD Budget = Total Budget - Mission Required Trips Costs.
2. (Improve Instruction) ProD should help instructors be more effective teaching.
3. (Fair development of instructor core) Each instructor is to be given the opportunity to go on at least one ProD trip per year and there should a limit on the max number of trips a person goes on.
4. (Merit): Instructors merit is considered when assigning instructors to ProD opportunities that are tagged as being a "glory trip". A glory trip is does something that is extra special. The idea is to reward high performing instructors with a special trip.

Goal Programming Model

Goal programming is an optimization techniques to solve problems with multiple objectives. Additionally, Goal programming allows for flexibility in meeting constraints by allowing constraints to be deviated/broken in order to balance multiple goals. The model will determine which instructor to send on which ProD trip for each quarter (3 month period). Goal programming models can be formulated in many different ways. For example, some models might aim to minimized the sum of deviations from the goal targets, while other models, may be set up to minimize the largest deviation (minimax model). This project will use a version of the minimax model. A weighted percentage based minimax model is selected for the following reasons and assumptions. It is assumed that harm from deviations may increase rapidly such that it may not be advantage to allow one goal to float to far from target. For example, if objective 3 was allowed to deviate it could create harmful perceptions of unfairness. The percentage based minimax model was selected because harm of deviation was assumed to be more closely related to percentage deviation then nominal deviation. Weights were applied because squadron leadership has communicated that a loose prioritization between the goals. Weights will allow the model to capture these priorities into the results.

Decision Variables

There are j instructors and i ProD trips. The model aims to determine which instructors to send on which trip. For each instructor a decision is made to either send or don’t send for each trip. To capture all this decisions a discison matrix is defined. $DV[i,j]$ captures the decision for each instructor trip combination. Each element of the matrix is either a 0 or a 1. A 0 indicates that the jth instructor was not assigned to the ith trip. While a 1 indicates that the jth intructor is assigned to the ith trip. For example, DV[1,2]=1 indicates the decision to send instructor 2 to Pro-D trip 1. While DV[12,34] =0 indicates that the 34th instructor is not assigned to 12th trip.

Objective

Min: Q, Where Q is less than or equal to the weighted ratio of goal related deviations (e.g Q <= (weight) *Deviation/GoalTarget)

```
suppressPackageStartupMessages(library(ggplot2))
suppressPackageStartupMessages(library(ggrepel))
suppressPackageStartupMessages(library (magrittr, quietly = TRUE))     #Used for pipe
s/dplyr
suppressPackageStartupMessages(library (pander, quietly = TRUE))       # for nicely fo
rmated tables
suppressPackageStartupMessages(library(dplyr, quietly=TRUE))           # For data stru
cture manipulation
suppressPackageStartupMessages(library(ROI, quietly=TRUE))             # R Optimizatio
n Interface package
suppressPackageStartupMessages(library(ROI.plugin.glpk, quietly=TRUE)) # Connection t
o glpk as solver
suppressPackageStartupMessages(library(ompr, quietly=TRUE))            # Optimization
Modeling using R
suppressPackageStartupMessages(library(ompr.roi, quietly=TRUE))        # Connective ti
ssue
suppressPackageStartupMessages(library(tidyr))
```

Data Input

```
# reads in the instructor data from a csv file.
RawInstructorData <- as.matrix(read.table(file="InstructorData.csv", row.names=1,sep
=","))
colnames(RawInstructorData) <- c("Instructor","DAID", "4QHrs", "Top Performer","Activ
e/Inactive")
RawInstructorData
```

```
##    Instructor    DAID        4QHrs Top Performer Active/Inactive
## 1  "Carswell"    "21-Sep-18" "408" "0"           "1"
## 2  "Bostic"      "18-Sep-18" "424" "0"           "1"
## 3  "Frank"       "18-Sep-18" "408" "0"           "1"
## 4  "Schell"      "18-Sep-18" "416" "0"           "1"
## 5  "McLarty"     "6-Aug-18"  "368" "0"           "1"
## 6  "Johnson"     "19-Jun-18" "328" "1"           "1"
## 7  "Parsons"     "19-Jun-18" "328" "0"           "1"
## 8  "Ramie"       "19-Jun-18" "368" "0"           "1"
## 9  "McKenzie"    "4-May-18"  "368" "0"           "0"
## 11 "Corrigan, B" "19-Mar-18" " 64" "0"           "0"
## 10 "Corrigan, P" "19-Mar-18" "248" "0"           "1"
## 12 "Moselle"     "12-Dec-17" "360" "0"           "1"
## 13 "Payne"       "30-Oct-17" "  8" "0"           "0"
## 15 "Boyd"        "26-Sep-17" " 64" "0"           "0"
## 14 "Meno"        "26-Sep-17" "184" "1"           "1"
## 16 "Hudson"      "21-Aug-17" " 48" "0"           "0"
## 17 "Moore"       "21-Aug-17" " 64" "0"           "0"
## 18 "Hagstrom"    "5-Jul-17"  "104" "0"           "0"
## 22 "Spratt"      "5-Jul-17"  "296" "0"           "0"
## 19 "Temple"      "5-Jul-17"  " 24" "0"           "0"
## 20 "Westfall"    "5-Jul-17"  "280" "0"           "0"
## 21 "Wyatt, B"    "5-Jul-17"  "312" "0"           "1"
## 23 "Wyatt, J"    "5-Jul-17"  "328" "0"           "1"
## 24 "Romanofski"  "8-May-17"  " 40" "1"           "1"
## 26 "Cook"        "6-Feb-17"  " 64" "1"           "1"
## 25 "Glass"       "6-Feb-17"  "104" "0"           "0"
## 27 "Dalrymple"   "12-Dec-16" " 80" "0"           "0"
## 28 "White, C"    "31-Oct-16" "240" "0"           "1"
## 30 "Chaney"      "22-Aug-16" " 40" "0"           "0"
## 29 "Gunther"     "22-Aug-16" " 40" "0"           "0"
## 33 "Anderson"    "2-May-16"  "200" "1"           "1"
## 31 "Kuhls"       "2-May-16"  "144" "0"           "0"
## 32 "Turner"      "2-May-16"  " 32" "0"           "0"
## 35 "Fox"         "9-Dec-15"  " 24" "0"           "0"
## 34 "Raines"      "9-Dec-15"  " 48" "0"           "0"
## 36 "Dean"        "3-Nov-15"  " 24" "1"           "0"
## 37 "Burnside, S" "15-Sep-15" " 72" "0"           "0"
```

```
# reads in the instructor data from a csv file.
RawTripData <- as.matrix(read.table(file="C:/Users/rpatr/Documents/ProDTripData.csv",
row.names=1,sep=","))
colnames(RawTripData) <- c("Trip Name","MeritBased", "Cost/Instr", "instrSlotsAvailabl
e","NumberOfDays","TotalCost")
RawTripData
```

```
##    Trip Name                        MeritBased Cost/Instr
## 1  "Los Alamos/Sandia National Lab" "0"        "1276.56"
## 2  "Nuc 200"                        "1"        "1282.94"
## 3  "Y-12"                           "0"        "1858.48"
## 4  "NNSS"                           "0"        " 448.47"
## 5  "PANTEX"                         "0"        "1267.52"
## 6  "Bangor NAS"                     "0"        "1288.12"
## 7  "NMCC"                           "1"        "2310.44"
## 8  "LLNL"                           "0"        "1048.10"
## 9  "AF WERX"                        "1"        "2555.28"
## 10 "Kingsbay"                       "0"        "1160.29"
## 11 "AFA Symposium"                  "1"        "3318.60"
## 12 "Honeywell Tour"                 "0"        "1380.22"
## 13 "Nuclear Symposium"              "1"        "1841.13"
##    instrSlotsAvailable NumberOfDays TotalCost
## 1  "4"                 "4"          " 5106.24"
## 2  "4"                 "6"          " 5131.76"
## 3  "6"                 "3"          "11150.88"
## 4  "4"                 "4"          " 1793.88"
## 5  "2"                 "3"          " 2535.04"
## 6  "3"                 "4"          " 3864.36"
## 7  "2"                 "5"          " 4620.88"
## 8  "5"                 "3"          " 5240.50"
## 9  "3"                 "5"          " 7665.84"
## 10 "4"                 "4"          " 4641.16"
## 11 "1"                 "5"          " 3318.60"
## 12 "4"                 "5"          " 5520.88"
## 13 "4"                 "4"          " 7364.52"
```

```r
TotalTravelBudget <- 188000 #budget for all unit funded travel. This is a hard limit.
MissionTravelCosts <- 0 #(ssumes no mission related trips accomplished prior to model
being run.
ForcastedMissionTravelCosts <- 105000 #current mission related travel forcasted by res
ource advisor
ProDTravelCosts <- 0  #(assumes no ProD trips accomplished prior to model being run.
TotalRemainingTravelBudget <- 188000-MissionTravelCosts-ProDTravelCosts
ProDTripBudgetTarget <- .9*(TotalTravelBudget-ForcastedMissionTravelCosts) # Target i
s to use 90% of open funds
NumOfTrips <- length(RawTripData[,1])
NumOfInstructors <- length(RawInstructorData[,1])
TDYCostPerPerson <- as.numeric(RawTripData[,3])
InstructorStatus <- as.numeric(RawInstructorData[,5])
InstructorMerit <-  as.numeric(RawInstructorData[,4])
TripMeritStatus <-  as.numeric(RawTripData[,2])
InstrSlotsAvailable <- as.numeric(RawTripData[,4])
NumOfTripsSlotsAvailable <- sum(InstrSlotsAvailable)

TotalNumOfActiveInstructors<-tabulate(as.numeric(RawInstructorData[,5]),nbins = 1)

#Target for the number of active instructors to send on Trips.
TotalNumOfProDTripsForActiveInstructorsTarget<-TotalNumOfActiveInstructors*(floor(NumO
fTripsSlotsAvailable/NumOfInstructors))+min((NumOfTripsSlotsAvailable%%NumOfInstructor
s),TotalNumOfActiveInstructors)

#Calc the TargetNumOfInstructorsToSendOnMeritTrips
NumOfMeritTripsSlotsAvailable <- TripMeritStatus %*% InstrSlotsAvailable
NumOfInstructorsWithMeritStatus <- tabulate(as.numeric(RawInstructorData[,4]),nbins =
1)
TargetNumOfInstructorsToSendOnMeritTrips <- min(NumOfInstructorsWithMeritStatus,NumOfM
eritTripsSlotsAvailable)
MinNumOfMeritTripsPerTopPerformer <- floor(NumOfMeritTripsSlotsAvailable/NumOfInstruct
orsWithMeritStatus)
MiniumTripsTarget <- floor(NumOfTripsSlotsAvailable/NumOfInstructors)
MaxTripsTarget <- ceiling(NumOfTripsSlotsAvailable/NumOfInstructors)

Weights <- as.matrix(c(1,1,1,1,1))
```

Goal Programming Code

```
model <- MIPModel() %>%

  # VARIABLES
  add_variable (DV[i, j], i=1:NumOfTrips, j=1:NumOfInstructors, type="binary") %>%
  add_variable(BDminus, type = "continuous", lb = 0) %>%
  add_variable(BDpostive, type = "continuous", lb = 0) %>%
  add_variable(IDminus, type = "integer", lb = 0) %>%
  add_variable(IDpostive, type = "integer", lb = 0) %>%
  add_variable(TDminus[j], j=1:NumOfInstructors, type="integer", lb=0) %>%
  add_variable(TDpostive[j], j=1:NumOfInstructors, type="integer", lb=0) %>%
  add_variable(MDminus, type = "integer", lb = 0) %>%
  add_variable(MDpostive, type = "integer", lb = 0) %>%

  # OBJECTIVE
  #Minimize the weighted sum of deviations
  set_objective(Weights[1]*(BDminus + BDpostive) + Weights[2]*IDminus +
                  sum_expr(Weights[3]*TDminus[j]+Weights[4]*TDpostive[j], j=1:NumOfIns
tructors)+Weights[5]*MDminus, "min") %>%

  # GOAL CONSTRAINTS
  # 1. ProD Trip Budget Goal
  add_constraint(sum_expr(TDYCostPerPerson[i]*DV[i,j],i=1:NumOfTrips, j=1:NumOfInstruc
tors)
                 + BDminus - BDpostive == ProDTripBudgetTarget)%>%

  # 2. Improve instruction Goal
  add_constraint(sum_expr(InstructorStatus[j]*DV[i,j],i=1:NumOfTrips, j=1:NumOfInstruc
tors) + IDminus - IDpostive
                 == TotalNumOfProDTripsForActiveInstructorsTarget)%>%

  # 3. Fairly Develop Instructors: Max Trips
  add_constraint(sum_expr(DV[i,j],i=1:NumOfTrips)-TDpostive[j] <= MaxTripsTarget, j=1:
NumOfInstructors)%>%

  # 4. Fairly Develop Instructors: Minimum Trips
  add_constraint(sum_expr(DV[i,j],i=1:NumOfTrips)+TDminus[j] >= MiniumTripsTarget, j=
1:NumOfInstructors)%>%

  # 5. Instructor Merit
  add_constraint(sum_expr(InstructorMerit[j]*TripMeritStatus[i]*DV[i,j],i=1:NumOfTrip
s,j=1:NumOfInstructors) + MDminus -MDpostive  == TargetNumOfInstructorsToSendOnMeritTr
ips)%>%


  # # HARD CONSTRAINTS
  #1. Instructor Merit (if spots are avialble each Top performer will go on a min numb
er of available merit trips)
  add_constraint(sum_expr(InstructorMerit[j]*TripMeritStatus[i]*DV[i,j],i=1:NumOfTrip
s)
                 >= InstructorMerit[j]*MinNumOfMeritTripsPerTopPerformer,j=1:NumOfInst
ructors)%>%

  # 1. Budget Constraint
  add_constraint(sum_expr(TDYCostPerPerson[i]*DV[i,j],i=1:NumOfTrips, j=1:NumOfInstruc
tors)
                 <= TotalRemainingTravelBudget)%>%

  # 2. Slots Open for Trip
  add_constraint(sum_expr(DV[i,j], j=1:NumOfInstructors) <= InstrSlotsAvailable[i], i=
1:NumOfTrips)

result <- solve_model(model, with_ROI(solver = "glpk", verbose = TRUE))
```

```
## <SOLVER MSG>  ----
## GLPK Simplex Optimizer, v4.47
## 128 rows, 561 columns, 2766 non-zeros
##       0: obj =  0.000000000e+000  infeas = 7.478e+004 (3)
## *   146: obj =  6.756460000e+003  infeas = 0.000e+000 (0)
## *   174: obj =  6.746460000e+003  infeas = 0.000e+000 (0)
## OPTIMAL SOLUTION FOUND
## GLPK Integer Optimizer, v4.47
## 128 rows, 561 columns, 2766 non-zeros
## 559 integer variables, 481 of which are binary
## Integer optimization begins...
## +   174: mip =     not found yet >=              -inf        (1; 0)
## +   246: >>>>>  6.746460000e+003 >=  6.746460000e+003   0.0% (1; 0)
## +   246: mip =  6.746460000e+003 >=     tree is empty   0.0% (0; 1)
## INTEGER OPTIMAL SOLUTION FOUND
## <!SOLVER MSG> ----
```

```r
# Active assignments (A) of each jth instructor to ith trips.
A <- result %>%
  get_solution(DV[i,j]) %>%
  filter(value == 1) %>%
  select(i, j)

# Text name of the ith trip
TDYNames <- RawTripData[,1]

# Total Num of instructors going on the ith trip
NumAssignedToEachTDY <- tabulate(as.numeric(A[,1]),nbins = NumOfTrips)

# Total Num of top performers going on the ith trip
NumOfTopPerformersAssignedToEachTDY <- matrix(rep(0, each=NumOfTrips), ncol=1)

# Manual counting of all the top performers for each trip.
for(i in 1:nrow(A))
{
  if(A[i,2]*InstructorMerit[A[i,2]] >= 1)
  {
    NumOfTopPerformersAssignedToEachTDY[A[i,1]] = NumOfTopPerformersAssignedToEachTDY
[A[i,1]] + 1
  }
}

#TripSummary (TDYName, #Assigned, Capacity, MeritTrip? AssignedOnMerit  Cost/Person)
TripSummaryChart <- cbind(TDYNames,NumAssignedToEachTDY,InstrSlotsAvailable,
                          NumOfTopPerformersAssignedToEachTDY,TripMeritStatus,TDYCostP
erPerson)
colnames(TripSummaryChart)<-list("Names", "Total", "Cap",
                                 "NumOfTPs", "GloryTrip?", "Cost/Per")

# Total Num of instructors going on the ith trip
NumTripsAssignedToEachInstructor <- tabulate(as.numeric(A[,2]),nbins = NumOfInstructor
s)

# Blank vector to be build up
NumOfMeritTripsForEachInstructor<- matrix(rep(0, each=NumOfInstructors), ncol=1)

# Of the trips each instructor went on how many trips had merit based classification?
for(t in 1:nrow(A))
{
  if(A[t,1]*TripMeritStatus[A[t,1]] >= 1)
  {
    NumOfMeritTripsForEachInstructor[A[t,2]] = NumOfMeritTripsForEachInstructor[A[t,
2]] + 1
  }
}
```

```
# The first 3 trips assigned to each instructor
AssignmentByInstructorMatrix <- matrix(rep(0, each=NumOfInstructors*3), ncol=3) #blan
k matrix to be built

 #Count which trips each instructor went on to a max of 3 trips.
for(k in 1:nrow(A))
{
  # if the first and second slot are filled then use the thrid slot, unless already fi
lled
  if ((AssignmentByInstructorMatrix[A[k,2],3] == 0) && (AssignmentByInstructorMatrix[A
[k,2],2] != 0) && (AssignmentByInstructorMatrix[A[k,2],1] != 0))
  {
    AssignmentByInstructorMatrix[A[k,2],3] = A[k,1]
  }
    #if first slot is filled then use the second slot, unless already filled
  if((AssignmentByInstructorMatrix[A[k,2],2] == 0) && (AssignmentByInstructorMatrix[A
[k,2],1] != 0))
  {
    AssignmentByInstructorMatrix[A[k,2],2] = A[k,1]
    }
    #if first trip slot isn't filled then record trip, otherwise use next slot
  if(AssignmentByInstructorMatrix[A[k,2],1] == 0)
  {
    AssignmentByInstructorMatrix[A[k,2],1] = A[k,1]
     }
 }

InstructorNumber <- matrix(c(1:NumOfInstructors),ncol=1)

# InstructorChart (TotalTrips, MeritTrips, Active?, MeritStatus, Trip1, Trip2, Trip3)
InstructorChart <- cbind(InstructorNumber,NumTripsAssignedToEachInstructor,NumOfMeritT
ripsForEachInstructor,
                         InstructorMerit,InstructorStatus,AssignmentByInstructorMatri
x)
colnames(InstructorChart)<-list("Instructor","TotalTrips", "MeritTrips","MeritStatu
s","Active?", "Trip1", "Trip2", "Trip3")


# Goal Charts ()
MaxWeightedPercentDeviation <- objective_value(result)
# Goal Charts (Goal,MinusDev, PostiveDev)

# Deviation Results Extacted
BDminusResult <- get_solution(result, BDminus)
BDpostiveResult <-get_solution(result, BDpostive)
IDminusResult <- get_solution(result, IDminus)
IDpostiveResult <- get_solution(result, IDpostive)
MDminusResult <- get_solution(result, MDminus)
```

```
MDpostiveResult <- get_solution(result, MDpostive)

# GoalResults
TotalSpent <- ProDTripBudgetTarget - BDminusResult + BDpostiveResult
TotalNumOfProDTripsForActiveInstructors <- TotalNumOfProDTripsForActiveInstructorsTarg
et - IDminusResult + IDpostiveResult
NumOfTPsSendOnMeritTrips <- TargetNumOfInstructorsToSendOnMeritTrips - MDminusResult
+ MDpostiveResult

# GoalDataVectors
G1 <- matrix(c(TotalSpent,ProDTripBudgetTarget,BDminusResult,BDpostiveResult),nrow=1)
G2 <- matrix(c(TotalNumOfProDTripsForActiveInstructors,
               TotalNumOfProDTripsForActiveInstructorsTarget,IDminusResult,IDpostiveRe
sult),nrow=1)
G3 <- matrix(c("SeeIntrChart",MiniumTripsTarget,"SeeIntrChart","SeeIntrChart"),nrow=1)
G4 <- matrix(c("SeeIntrChart",MaxTripsTarget,"SeeIntrChart","SeeIntrChart"),nrow=1)
G5 <- matrix(c(NumOfTPsSendOnMeritTrips,TargetNumOfInstructorsToSendOnMeritTrips,MDmin
usResult,MDpostiveResult),nrow=1)
GoalChart <- rbind(G1,G2,G3,G4,G5)
colnames(GoalChart)<-list("GoalResult","Target","MinusDev","PosDev")
rownames(GoalChart)<-list("Budget","TDYs4ActiveInstructors","MinTripsGoal","MaxTripsGo
al", "GloryTrips")

#Weights Data
WeightsByGoal <- t(Weights)
colnames(WeightsByGoal)<-list("Goal1","Goal2","Goal3","Goal4", "Goal5")
rownames(WeightsByGoal)<-list("Weights")

#display the formated results
print(paste0("Objective value= ", objective_value(result)))
```

```
## [1] "Objective value= 6746.45999999998"
```

```
pander(WeightsByGoal,
       caption="WeightsSummary")
```

WeightsSummary

	Goal1	Goal2	Goal3	Goal4	Goal5
Weights	1	1	1	1	1

```
pander(GoalChart,
       caption="GoalSummary")
```

GoalSummary (continued below)

	GoalResult	Target	MinusDev
Budget	67954.54	74700	6745.45999999998
TDYs4ActiveInstructors	26	26	0
MinTripsGoal	SeeIntrChart	1	SeeIntrChart
MaxTripsGoal	SeeIntrChart	2	SeeIntrChart
GloryTrips	12	6	0

	PosDev
Budget	0
TDYs4ActiveInstructors	0
MinTripsGoal	SeeIntrChart
MaxTripsGoal	SeeIntrChart
GloryTrips	6

```
pander(TripSummaryChart ,
       caption="TripSummary")
```

TripSummary

Names	Total	Cap	NumOfTPs	GloryTrip?	Cost/Per
Los Alamos/Sandia National Lab	4	4	0	0	1276.56
Nuc 200	4	4	3	1	1282.94
Y-12	6	6	0	0	1858.48
NNSS	4	4	0	0	448.47
PANTEX	2	2	0	0	1267.52
Bangor NAS	3	3	0	0	1288.12
NMCC	2	2	2	1	2310.44
LLNL	5	5	0	0	1048.1
AF WERX	3	3	2	1	2555.28
Kingsbay	4	4	0	0	1160.29
AFA Symposium	1	1	1	1	3318.6

Names	Total	Cap	NumOfTPs	GloryTrip?	Cost/Per
Honeywell Tour	4	4	0	0	1380.22
Nuclear Symposium	4	4	4	1	1841.13

```
pander(InstructorChart,
       caption="InstructorSummary")
```

InstructorSummary

Instructor	TotalTrips	MeritTrips	MeritStatus	Active?	Trip1	Trip2	Trip3
1	2	0	0	1	8	12	0
2	2	0	0	1	8	12	0
3	1	0	0	1	3	0	0
4	1	0	0	1	3	0	0
5	1	0	0	1	3	0	0
6	2	2	1	1	2	9	0
7	1	0	0	1	3	0	0
8	1	0	0	1	3	0	0
9	1	0	0	0	3	0	0
10	1	0	0	0	12	0	0
11	1	0	0	1	4	0	0
12	2	1	0	1	8	9	0
13	1	0	0	0	1	0	0
14	1	1	0	0	2	0	0
15	2	2	1	1	2	11	0
16	1	0	0	0	6	0	0
17	1	0	0	0	4	0	0
18	1	0	0	0	1	0	0
19	1	0	0	0	1	0	0
20	1	0	0	0	1	0	0
21	1	0	0	0	5	0	0

Instructor	TotalTrips	MeritTrips	MeritStatus	Active?	Trip1	Trip2	Trip3
22	2	0	0	1	10	12	0
23	1	0	0	1	6	0	0
24	2	2	1	1	7	13	0
25	2	2	1	1	7	13	0
26	1	0	0	0	5	0	0
27	0	0	0	0	0	0	0
28	1	0	0	1	6	0	0
29	1	0	0	0	10	0	0
30	1	0	0	0	10	0	0
31	2	2	1	1	2	13	0
32	1	0	0	0	8	0	0
33	1	0	0	0	10	0	0
34	1	0	0	0	8	0	0
35	1	0	0	0	4	0	0
36	2	2	1	0	9	13	0
37	1	0	0	0	4	0	0