DA5030.P2.VelosoSriramKarwarkar

Josiah Veloso

November 3, 2018

```
#Added my own headers and renamed the original file + made into a csv
Income<-read.csv("Cenadults.csv")
str(Income)
```

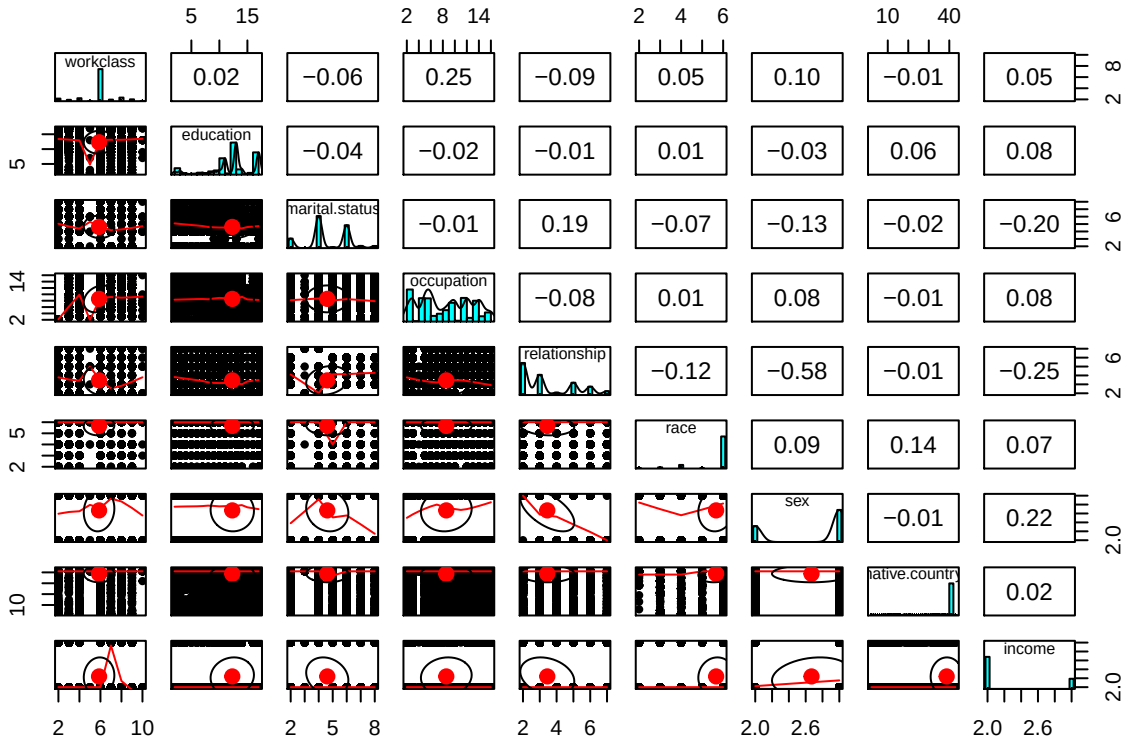
```
## 'data.frame':    32562 obs. of  15 variables:
##  $ age           : int  39 50 38 53 28 37 49 52 31 42 ...
##  $ workclass     : Factor w/ 10 levels "","? ","Federal-gov",..: 9 8 6 6 6 6 6 8 6 6 ...
##  $ fnlwgt        : int  77516 83311 215646 234721 338409 284582 160187 209642 45781 159449 ...
##  $ education     : Factor w/ 17 levels "","10th","11th",..: 11 11 13 3 11 14 8 13 14 11 ...
##  $ education.num : int  13 13 9 7 13 14 5 9 14 13 ...
##  $ marital.status: Factor w/ 8 levels "","Divorced",..: 6 4 2 4 4 4 5 4 6 4 ...
##  $ occupation    : Factor w/ 16 levels "","? ","Adm-clerical",..: 3 6 8 8 12 6 10 6 12 6 ...
##  $ relationship  : Factor w/ 7 levels "","Husband",..: 3 2 3 2 7 7 3 2 3 2 ...
##  $ race          : Factor w/ 6 levels "","Amer-Indian-Eskimo",..: 6 6 6 4 4 6 4 6 6 6 ...
##  $ sex           : Factor w/ 3 levels "","Female","Male": 3 3 3 3 2 2 2 3 2 3 ...
##  $ capital.gain  : int  2174 0 0 0 0 0 0 0 14084 5178 ...
##  $ capital.loss  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hours.per.week: int  40 13 40 40 40 40 16 45 50 40 ...
##  $ native.country: Factor w/ 43 levels "","? ","Cambodia",..: 41 41 41 41 7 41 25 41 41 41 ...
##  $ income        : Factor w/ 3 levels "","<=50K"," >50K": 2 2 2 2 2 2 2 3 3 3 ...
```

```
Income<-Income[-32562,]#removes the extra space from the original dataframe
#The value of 32561 is the TOTAL ROWS OF DATA overall
```

```
library(psych)
```

```
## Warning: package 'psych' was built under R version 3.4.4
```

```
pairs.panels(Income[c("workclass","education","marital.status","occupation","relationship","race","sex"
```




```
#in this initial examination WITHOUT SCALING ANYTHING there is little significant correlation between m
#Relationship and sex have noticable correlation
#if there was need to use the categorical variables - many of them seem to have a very prominent binomi
#an extreme skew so a transform might have to be necessary for some ML algorithims. However for Naive b
#This is because we are using only categorical variables to predict rather than continuous variables.
#At best occupation seems to be rather spread out and non-binomial.
```

```
#Frequencies calculated first and then likelihoods - all in one table each
#Pred1<-NBC(FrequencyTIncome,FrequencyTRace[5,3],FrequencyTSex[1,3],FrequencyTAdult[2,3],FrequencyTWork
Income$IsAdult<-ifelse(Income$age>=18,1,0) #The next question asks if she is an adult so a binary boole
Income2<-as.data.frame(Income[,c(15,9,10,16,2,4,14,6,7,8)])#We will only use the first 7 columns eventu
Income2$income<-ifelse(as.character(Income2$income)==as.character(" <=50K"),1,0)

PrR<-nrow(Income2)
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:PrR,]<-NA;
TestSplitter[,1:10]<-NA;#TestSplitter[,1:7]<-NA; is the parts we will eventually use
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:PrR,]<-NA;
OtherTestSplitter[,1:10]<-NA;#TestSplitter[,1:7]<-NA; is the parts we will eventually use
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country","mari
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country"
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)
```

```r
for(i in 1:PrR)
{
   for(a in 1:10) #REMINDER change to 1:7 in future tables
     ifelse(Income2[i,1]==1, #splits into two tables for categorical probability calculation of <=50K a
     TestSplitter[i,a]<-as.vector(Income2[i,a]), #TestSplitter and OtherTestSplitter dataframes are kep
     OtherTestSplitter[i,a]<-as.vector(Income2[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean iss


OATot<-nrow(Income2)#Total overall rows
TotalInL<-nrow(Splitter)#number of rows for people <=50K
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTMS<-as.data.frame(table(Splitter$marital.status))
FrequencyTMS[,3]<-as.data.frame(FrequencyTMS[,2]/TotalInL)
FrequencyTOccupation<-as.data.frame(table(Splitter$occupation))
FrequencyTOccupation[,3]<-as.data.frame(FrequencyTOccupation[,2]/TotalInL)
FrequencyTRelationship<-as.data.frame(table(Splitter$relationship))
FrequencyTRelationship[,3]<-as.data.frame(FrequencyTRelationship[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
##
FrequencyTNCNAMES<-as.data.frame(table(Splitter$native.country)) #This will be used to deal with questi
##
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)
```

```r
TotalInG<-nrow(AltSplitter)#number of rows for people >50K
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTMSA<-as.data.frame(table(AltSplitter$marital.status))
FrequencyTMSA[,3]<-as.data.frame(FrequencyTMSA[,2]/TotalInG)
FrequencyTOccupationA<-as.data.frame(table(AltSplitter$occupation))
FrequencyTOccupationA[,3]<-as.data.frame(FrequencyTOccupationA[,2]/TotalInG)
FrequencyTRelationshipA<-as.data.frame(table(AltSplitter$relationship))
FrequencyTRelationshipA[,3]<-as.data.frame(FrequencyTRelationshipA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
```

```
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)
```

```
#The custom naive bayes classifier
#fits 7 arguments fitting exactly the next question
#The first argument "x" is going to be the potential primary category to classify
#FrequencyTRace[5,2] Testing if index in dataframe works - note index number labels are wrong
#Naive Bayes = Likelihood(concentrated variable likelihood * x-features)
#/ DIVIDED BY
#Total Likelihood (concentrated variable likelihood * x-feature + other alternate variable likelihood *
#We are detecting income so it would be if income less than 50K (<=50K) of x[1,3] VS income greater tha
NBC<-function(x,r,s,a,w,e,nc){
  A=(x[1,3]*r*s*a*w*e*nc) #probability of <=50K
  B=(FrequencyTIncomeA[1,3]*FrequencyTRaceA[5,3]*FrequencyTSexA[1,3]*FrequencyTAdultA[1,3]*FrequencyTWo
  NBC<-A/(A+B)
}
#note this function has been fitted so the first argument is only going to be FrequencyTincome aka the
```

```
#Initial prediction
FrequencyTRace[5,1] #check race = white
```

```
## [1]  White
## 5 Levels:  Amer-Indian-Eskimo  Asian-Pac-Islander  Black ...  White
```

```
FrequencyTSex[1,1] #check sex = female
```

```
## [1]  Female
## Levels:  Female  Male
```

```
FrequencyTAdult[2,1] #check if adult = 1 = yes
```

```
## [1] 1
## Levels: 0 1
```

```
FrequencyTWorkclass[2,1] #check working class = federal govt. worker
```

```
## [1]  Federal-gov
## 9 Levels:  ?  Federal-gov  Local-gov  Never-worked ...  Without-pay
```

```
FrequencyTEducation[10,1] #check education = bachelors degree
```

```
## [1]  Bachelors
## 16 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
FrequencyTNC[20,1] #check native country = India
```

```
## [1]  India
## 42 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
Pred1<-NBC(FrequencyTIncome,FrequencyTRace[5,3],FrequencyTSex[1,3],FrequencyTAdult[2,3],FrequencyTWorkcl
Pred1
```

```
## [1] 0.6376822
```

There is a 63.77 percent chance that a person that is a white female adult working for the federal govt. who graduated from a bachelors from india, has less than or equal to 50K of income a year.

```
#Each fold end is calculated
Income<-Income[-32561,] #This will ensure equality so one row is omitted
Income2<-as.data.frame(Income[,c(15,9,10,16,2,4,14)])#new version of income removing the other columns
Income2$income<-ifelse(as.character(Income2$income)==as.character(" <=50K"),1,0)
TRows<-round(nrow(Income2))
Onefold<- round(TRows/10) #This is the total rows of test data
TrainTotal<- TRows-Onefold #This is the total rows of the training data
Twofold <- Onefold*2 #Twofold and up is actually not needed - but I will keep them for checking reasons
Threefold <- Onefold*3
Fourfold <- Onefold*4
Fivefold <- Onefold*5
Sixfold <- Onefold*6
Sevenfold <- Onefold*7
Eightfold <- Onefold*8
Ninefold <- Onefold*9
Tenfold <- Onefold*10
#Each actual fold in the actual whole data is calculated (Test sets)
OneFI<-Income2[1:Onefold,]
TwoFI<-Income2[Onefold+1:Onefold,]#This indexing is apparently additive so this works
ThreeFI<-Income2[Twofold+1:Onefold,]
FourFI<-Income2[Threefold+1:Onefold,]
FiveFI<-Income2[Fourfold+1:Onefold,]
SixFI<-Income2[Fivefold+1:Onefold,]
SevenFI<-Income2[Sixfold+1:Onefold,]
EightFI<-Income2[Sevenfold+1:Onefold,]
NineFI<-Income2[Eightfold+1:Onefold,]
TenFI<-Income2[Ninefold+1:Onefold,]
#Training sets
OneOff<-1:Onefold
OneTrain<-Income2[-OneOff,]
TwoOff<-Onefold+1:Onefold #This also has the same behavior
TwoTrain<-Income2[-TwoOff,]
ThreeOff<-Twofold+1:Onefold
ThreeTrain<-Income2[-ThreeOff,]
FourOff<-Threefold+1:Onefold
FourTrain<-Income2[-FourOff,]
FiveOff<-Fourfold+1:Onefold
FiveTrain<-Income2[-FiveOff,]
SixOff<-Fivefold+1:Onefold
SixTrain<-Income2[-SixOff,]
SevenOff<-Sixfold+1:Onefold
SevenTrain<-Income2[-SevenOff,]
EightOff<-Sevenfold+1:Onefold
EightTrain<-Income2[-EightOff,]
NineOff<-Eightfold+1:Onefold
NineTrain<-Income2[-NineOff,]
TenOff<-Ninefold+1:Onefold
TenTrain<-Income2[-TenOff,]
```

```
#These functions are to check for probabilities in likelihood table based on the factor <=50K

Income2<-data.frame(Income[,c(15,9,10,16,2,4,14)])


FindProbIncome<-function(x){
  FindProbIncome<-ifelse(x==1,FrequencyTIncome[1,3],FrequencyTIncomeA[1,3])
}

t1<-FindProbIncome(0)

FindProbRace<-function(x){
  FindProbRace<-ifelse(as.character(x)==as.character(" Amer-Indian-Eskimo"),FrequencyTRace[1,3],
                       ifelse(as.character(x)==as.character(" Asian-Pac-Islander"),FrequencyTRace[2,3],
                              ifelse(as.character(x)==as.character(" Black"),FrequencyTRace[3,3],
                                     ifelse(as.character(x)==as.character(" Other"),FrequencyTRace[4,3]
}

t2A<-FindProbRace(" Black")
t2B<-FindProbRace(" White")

FindProbSex<-function(x){
  ifelse(as.character(x)==as.character(" Female"),FrequencyTSex[1,3],FrequencyTSex[2,3])
}

t3<-FindProbSex(as.character("Female"))

FindProbAdult<-function(x){
  ifelse(x==0,FrequencyTAdult[1,3],FrequencyTAdult[2,3])
}

t4<-FindProbAdult(0)

FindProbWorkClass<-function(x){
  ifelse(as.character(x)==as.character("?"),FrequencyTWorkclass[1,3],
         ifelse(as.character(x)==as.character(" Federal-gov"),FrequencyTWorkclass[2,3],
                ifelse(as.character(x)==as.character(" Local-gov"),FrequencyTWorkclass[3,3],
                       ifelse(as.character(x)==as.character(" Never-worked"),FrequencyTWorkclass[4,3],
                              ifelse(as.character(x)==as.character(" Private"),FrequencyTWorkclass[5,3]
                                     ifelse(as.character(x)==as.character(" Self-emp-inc"),FrequencyTWo
                                            ifelse(as.character(x)==as.character(" Self-emp-not-inc"),F
                                                   ifelse(as.character(x)==as.character(" State-gov"),F
                                                          ))))))))
}

t5<-FindProbWorkClass(as.character(" Self-emp-not-inc"))

FindProbEducation<-function(x){
 ifelse(as.character(x)==as.character(" 10th"),FrequencyTEducation[1,3],
         ifelse(as.character(x)==as.character(" 11th"),FrequencyTEducation[2,3],
                ifelse(as.character(x)==as.character(" 12th"),FrequencyTEducation[3,3],
                       ifelse(as.character(x)==as.character(" 1st-4th"),FrequencyTEducation[4,3],
                              ifelse(as.character(x)==as.character(" 5th-6th"),FrequencyTEducation[5,3]
```

```
                                    ifelse(as.character(x)==as.character(" 7th-8th"),FrequencyTEducati
                                           ifelse(as.character(x)==as.character(" 9th"),FrequencyTEduca
                                                  ifelse(as.character(x)==as.character(" Assoc-acdm"),
                                                         ifelse(as.character(x)==as.character(" Assoc-
        ifelse(as.character(x)==as.character(" Bachelors"),FrequencyTEducation[10,3],
               ifelse(as.character(x)==as.character(" Doctorate"),FrequencyTEducation[11,3],
                      ifelse(as.character(x)==as.character(" HS-grad"),FrequencyTEducation[12,3],
                             ifelse(as.character(x)==as.character(" Masters"),FrequencyTEducation[13,3
                                    ifelse(as.character(x)==as.character(" Preschool"),FrequencyTEducat
                                           ifelse(as.character(x)==as.character(" Prof-school"),Freque
}

t6A<-FindProbEducation(" 5th-6th")

t6B<-FindProbEducation(" Masters")

FindProbNative<-function(x){
 ifelse(as.character(x)==as.character(" ?"),FrequencyTNC[1,3],
        ifelse(as.character(x)==as.character(" Cambodia"),FrequencyTNC[2,3],
               ifelse(as.character(x)==as.character(" Canada"),FrequencyTNC[3,3],
                      ifelse(as.character(x)==as.character(" China"),FrequencyTNC[4,3],
                             ifelse(as.character(x)==as.character(" Columbia"),FrequencyTNC[5,3],
                                    ifelse(as.character(x)==as.character(" Cuba"),FrequencyTNC[6,3],
                                           ifelse(as.character(x)==as.character(" Dominican-Republic")
                                                  ifelse(as.character(x)==as.character(" Ecuador"),Fre
                                                         ifelse(as.character(x)==as.character(" El-Sal
        ifelse(as.character(x)==as.character(" England"),FrequencyTNC[10,3],
               ifelse(as.character(x)==as.character(" France"),FrequencyTNC[11,3],
                      ifelse(as.character(x)==as.character(" Germany"),FrequencyTNC[12,3],
                             ifelse(as.character(x)==as.character(" Greece"),FrequencyTNC[13,3],
                                    ifelse(as.character(x)==as.character(" Guatemala"),FrequencyTNC[14
                                           ifelse(as.character(x)==as.character(" Haiti"),FrequencyTNC
        ifelse(as.character(x)==as.character(" Honduras"),FrequencyTNC[17,3],
               ifelse(as.character(x)==as.character(" Hong"),FrequencyTNC[18,3],
                      ifelse(as.character(x)==as.character(" Hungary"),FrequencyTNC[19,3],
                             ifelse(as.character(x)==as.character(" India"),FrequencyTNC[20,3],
                                    ifelse(as.character(x)==as.character(" Iran"),FrequencyTNC[21,3],
                                           ifelse(as.character(x)==as.character(" Ireland"),FrequencyT
                                                  ifelse(as.character(x)==as.character(" Italy"),Freque
                                                         ifelse(as.character(x)==as.character(" Jamaic
        ifelse(as.character(x)==as.character(" Japan"),FrequencyTNC[25,3],
               ifelse(as.character(x)==as.character(" Laos"),FrequencyTNC[26,3],
                      ifelse(as.character(x)==as.character(" Mexico"),FrequencyTNC[27,3],
                             ifelse(as.character(x)==as.character(" Nicaragua"),FrequencyTNC[28,3],
                                    ifelse(as.character(x)==as.character(" Outlying-US(Guam-USVI-etc)"
                                           ifelse(as.character(x)==as.character(" Peru"),FrequencyTNC[
        ifelse(as.character(x)==as.character(" Poland"),FrequencyTNC[32,3],
               ifelse(as.character(x)==as.character(" Portugal"),FrequencyTNC[33,3],
                      ifelse(as.character(x)==as.character(" Puerto-Rico"),FrequencyTNC[34,3],
                             ifelse(as.character(x)==as.character(" Scotland"),FrequencyTNC[35,3],
                                    ifelse(as.character(x)==as.character(" South"),FrequencyTNC[36,3],
                                           ifelse(as.character(x)==as.character(" Taiwan"),FrequencyTN
                                                  ifelse(as.character(x)==as.character(" Thailand"),Fr
```

```
                                                        ifelse(as.character(x)==as.character(" Trinad
        ifelse(as.character(x)==as.character(" United-States"),FrequencyTNC[40,3],
               ifelse(as.character(x)==as.character(" Vietnam"),FrequencyTNC[41,3],FrequencyTNC[42,3]
                           ))))))))))))))))))))))))))))))))))))))))))))))))))))))))))))))))

}


t7A<-FindProbNative(" China")
t7B<-FindProbNative(" Greece")
t7C<-FindProbNative(" India")
t7D<-FindProbNative(" Outlying-US(Guam-USVI-etc)")
t7E<-FindProbNative(" Philippines")
t7F<-FindProbNative(" Yugoslavia")
```

```
#These functions are to check for probabilities in likelihood table based on the factor >50K

FindProbIncomeOther<-function(x){
  FindProbIncomeOther<-ifelse(x==1,FrequencyTIncomeA[1,3],FrequencyTIncome[1,3])
}

#EDUCATION AND WORKCLASS AND NCA HAVE LAPLACE ISSUES

FindProbRaceOther<-function(x){

  FindProbRaceOther<-ifelse(as.character(x)==as.character(" Amer-Indian-Eskimo"),FrequencyTRaceA[1,3],
                        ifelse(as.character(x)==as.character(" Asian-Pac-Islander"),FrequencyTRaceA[2,3]
                               ifelse(as.character(x)==as.character(" Black"),FrequencyTRaceA[3,3],
                                      ifelse(as.character(x)==as.character(" Other"),FrequencyTRaceA[4,3
  #Error prone area due to very likely low count - applying "laplace"
}

t2A<-FindProbRaceOther(" Black")
t2B<-FindProbRaceOther(" White")

FindProbSexOther<-function(x){
  ifelse(as.character(x)==as.character(" Female"),FrequencyTSexA[1,3],FrequencyTSexA[2,3])
  #Error prone area due to very likely low count - applying "laplace"
}

t3<-FindProbSexOther(as.character(" Female"))

FindProbAdultOther<-function(x){
  ifelse(x==0,FrequencyTAdultA[1,3],1)#Error prone area due to very likely low count - applying "laplac
}

t4<-FindProbAdultOther(0)

FindProbWorkClassOther<-function(x){
  ifelse(as.character(x)==as.character(" ?"),FrequencyTWorkclassA[1,3],
         ifelse(as.character(x)==as.character(" Federal-gov"),FrequencyTWorkclassA[2,3],
                ifelse(as.character(x)==as.character(" Local-gov"),FrequencyTWorkclassA[3,3],
                       ifelse(as.character(x)==as.character(" Never-worked"),FrequencyTWorkclassA[4,3],
                              ifelse(as.character(x)==as.character(" Private"),FrequencyTWorkclassA[5,3
```

```
                                    ifelse(as.character(x)==as.character(" Self-emp-inc"),FrequencyTWo
                                           ifelse(as.character(x)==as.character(" Self-emp-not-inc"),F
                                                  ifelse(as.character(x)==as.character(" State-gov"),F
}

t5<-FindProbWorkClassOther(as.character(" Self-emp-not-inc"))

FindProbEducationOther<-function(x){
 #for(l in 1:16)
    #for(t in 1:7)
  #  {
  #    ifelse(x[l,1]!= FrequencyTEducation[l,1], x[l,]<-c(FrequencyTEducation[l,1],x[l,2]<-as.vector(1)
  #  }

 ifelse(as.character(x)==as.character(" 10th"),FrequencyTEducationA[1,3],
         ifelse(as.character(x)==as.character(" 11th"),FrequencyTEducationA[2,3],
                ifelse(as.character(x)==as.character(" 12th"),FrequencyTEducationA[3,3],
                       ifelse(as.character(x)==as.character(" 1st-4th"),FrequencyTEducationA[4,3],
                              ifelse(as.character(x)==as.character(" 5th-6th"),FrequencyTEducationA[5,3
                                     ifelse(as.character(x)==as.character(" 7th-8th"),FrequencyTEducati
                                            ifelse(as.character(x)==as.character(" 9th"),FrequencyTEduca
                                                   ifelse(as.character(x)==as.character(" Assoc-acdm"),
                                                          ifelse(as.character(x)==as.character(" Assoc-
         ifelse(as.character(x)==as.character(" Bachelors"),FrequencyTEducationA[10,3],
                ifelse(as.character(x)==as.character(" Doctorate"),FrequencyTEducationA[11,3],
                       ifelse(as.character(x)==as.character(" HS-grad"),FrequencyTEducationA[12,3],
                              ifelse(as.character(x)==as.character(" Masters"),FrequencyTEducationA[13,
                                     ifelse(as.character(x)==as.character(" Preschool"),FrequencyTEduca
                                            ifelse(as.character(x)==as.character(" Prof-school"),Freque


}

t6A<-FindProbEducationOther(" 5th-6th")

t6B<-FindProbEducationOther(" Masters")

FindProbNativeOther<-function(x){
 ifelse(as.character(x)==as.character(" ?"),FrequencyTNCA[1,3],
         ifelse(as.character(x)==as.character(" Cambodia"),FrequencyTNCA[2,3],
                ifelse(as.character(x)==as.character(" Canada"),FrequencyTNCA[3,3],
                       ifelse(as.character(x)==as.character(" China"),FrequencyTNCA[4,3],
                              ifelse(as.character(x)==as.character(" Columbia"),FrequencyTNCA[5,3],
                                     ifelse(as.character(x)==as.character(" Cuba"),FrequencyTNCA[6,3],
                                            ifelse(as.character(x)==as.character(" Dominican-Republic")
                                                   ifelse(as.character(x)==as.character(" Ecuador"),Fre
                                                          ifelse(as.character(x)==as.character(" El-Sal
         ifelse(as.character(x)==as.character(" England"),FrequencyTNCA[10,3],
                ifelse(as.character(x)==as.character(" France"),FrequencyTNCA[11,3],
                       ifelse(as.character(x)==as.character(" Germany"),FrequencyTNCA[12,3],
                              ifelse(as.character(x)==as.character(" Greece"),FrequencyTNCA[13,3],
                                     ifelse(as.character(x)==as.character(" Guatemala"),FrequencyTNCA[1
                                            ifelse(as.character(x)==as.character(" Haiti"),FrequencyTNC
```

```
        ifelse(as.character(x)==as.character(" Honduras"),FrequencyTNCA[17,3],
               ifelse(as.character(x)==as.character(" Hong"),FrequencyTNCA[18,3],
                      ifelse(as.character(x)==as.character(" Hungary"),FrequencyTNCA[19,3],
                             ifelse(as.character(x)==as.character(" India"),FrequencyTNCA[20,3],
                                    ifelse(as.character(x)==as.character(" Iran"),FrequencyTNCA[21,3],
                                           ifelse(as.character(x)==as.character(" Ireland"),FrequencyTN
                                                  ifelse(as.character(x)==as.character(" Italy"),Frequ
                                                         ifelse(as.character(x)==as.character(" Jamaica
        ifelse(as.character(x)==as.character(" Japan"),FrequencyTNCA[25,3],
               ifelse(as.character(x)==as.character(" Laos"),FrequencyTNCA[26,3],
                      ifelse(as.character(x)==as.character(" Mexico"),FrequencyTNCA[27,3],
                             ifelse(as.character(x)==as.character(" Nicaragua"),FrequencyTNCA[28,3],
                                    ifelse(as.character(x)==as.character(" Outlying-US(Guam-USVI-etc)"
                                           ifelse(as.character(x)==as.character(" Peru"),FrequencyTNCA
        ifelse(as.character(x)==as.character(" Poland"),FrequencyTNCA[32,3],
               ifelse(as.character(x)==as.character(" Portugal"),FrequencyTNCA[33,3],
                      ifelse(as.character(x)==as.character(" Puerto-Rico"),FrequencyTNCA[34,3],
                             ifelse(as.character(x)==as.character(" Scotland"),FrequencyTNCA[35,3],
                                    ifelse(as.character(x)==as.character(" South"),FrequencyTNCA[36,3]
                                           ifelse(as.character(x)==as.character(" Taiwan"),FrequencyTNC
                                                  ifelse(as.character(x)==as.character(" Thailand"),Fre
                                                         ifelse(as.character(x)==as.character(" Trinada
        ifelse(as.character(x)==as.character(" United-States"),FrequencyTNCA[40,3],
               ifelse(as.character(x)==as.character(" Vietnam"),FrequencyTNCA[41,3],FrequencyTNCA[42,3
                      ))))))))))))))))))))))))))))))))))))))))))))))))))))

}
```

```
#Newly modified custom Naive bays to take in dataframe variables with probabilities for current and alt
NBCMod<-function(x,r,s,a,w,e,nc){
  A=FindProbIncome(x)*FindProbRace(r)*FindProbSex(s)*FindProbAdult(a)*FindProbWorkClass(w)*FindProbEduca
  B=FindProbIncomeOther(x)*FindProbRaceOther(r)*FindProbSexOther(s)*FindProbAdultOther(a)*FindProbWorkCl
  NBCMod<-(A/((A)+(B)))
}

TF<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India") #It works exactly the same as b
```

```
#Onefold recycling vectors
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;#TestSplitter[,1:7]<-NA; is the parts we will eventually use
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country"
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
```

```
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(OneTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50K
     TestSplitter[i,a]<-as.vector(OneTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are ke
     OtherTestSplitter[i,a]<-as.vector(OneTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean iss

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)

TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

NBCMod<-function(x,r,s,a,w,e,nc){
  A=FindProbIncome(x)*FindProbRace(r)*FindProbSex(s)*FindProbAdult(a)*FindProbWorkClass(w)*FindProbEduc
  B=FindProbIncomeOther(x)*FindProbRaceOther(r)*FindProbSexOther(s)*FindProbAdultOther(a)*FindProbWorkC
  NBCMod<-(A/((A)+(B)))
}

#TF<-NBCMod(1,"White","Female",1,"Federal-gov","Bachelors","India") #It works exactly the same as befor

Prob1f<-0 #probability collection
```

```
Prob1f<-as.data.frame(Prob1f)
Prob1f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:T
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestNC) #LAPLACE
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNC
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[TestNC,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:TestNC-1,],c(NA,1,1/Tra
```

```
## [[1]]
## [1]  Yugoslavia
## 38 Levels:  ?  Cambodia  Canada  China  Columbia  Cuba ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)

for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:T
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov        <NA>
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
OneTrain[1,1]
```

```
## [1] 0
```

```
OneTrain[1,3]
```

```
## [1]  Male
## Levels:   Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob1f[i,]<-NBCMod(OneTrain[i,1],OneTrain[i,2],OneTrain[i,3],OneTrain[i,4],OneTrain[i,5],OneTrain[i
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred1<-as.data.frame(0)
Pred1[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
Pred1<-as.data.frame(MeanTrainer(Prob1f))
summary(Pred1)
```

```
##  MeanTrainer(Prob1f)
##  Min.   :0.1859
##  1st Qu.:0.5370
##  Median :0.6162
##  Mean   :0.6117
##  3rd Qu.:0.6900
##  Max.   :0.9320
```

```
Prediction1<-0
Prediction1<-as.data.frame(ifelse(Pred1>.5,1,0))#threshold to assume prediction that a person has <=50K
summary(Prediction1)
```

```
##  MeanTrainer(Prob1f)
##  Min.   :0.00
##  1st Qu.:1.00
##  Median :1.00
##  Mean   :0.84
##  3rd Qu.:1.00
##  Max.   :1.00
```

```
Res1f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res1f[i]<-ifelse(Prediction1[i,1]==OneFI[i,1],1,0)
}
Accuracy1f<-mean(Res1f) #The accuracy of predicting a person in this FIRST fold is 67.35%!
```

```
#Twofold recycling vectors
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
```

```
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(TwoTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50K
     TestSplitter[i,a]<-as.vector(TwoTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are key
     OtherTestSplitter[i,a]<-as.vector(TwoTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean issu

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)

TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
```

```
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

Prob2f<-0 #probability collection
Prob2f<-as.data.frame(Prob2f)
Prob2f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:T
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestNC) #LAPLACE
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNC
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[TestNC,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:TestNC-1,],c(NA,1,1/Tra
```

```
## [[1]]
## [1]  Yugoslavia
## 40 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)

for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:T
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov        <NA>
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
```

```
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
TwoTrain[1,1]
```

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

```
TwoTrain[1,3]
```

```
## [1]  Male
## Levels:  Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob2f[i,]<-NBCMod(TwoTrain[i,1],TwoTrain[i,2],TwoTrain[i,3],TwoTrain[i,4],TwoTrain[i,5],TwoTrain[i
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred2<-as.data.frame(0)
Pred2[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
Pred2<-as.data.frame(MeanTrainer(Prob2f))
summary(Pred2)
```

```
##  MeanTrainer(Prob2f)
##  Min.   :0.1907
##  1st Qu.:0.5343
##  Median :0.6138
##  Mean   :0.6098
##  3rd Qu.:0.6880
##  Max.   :0.9318
```

```
Prediction2<-0
Prediction2<-as.data.frame(ifelse(Pred2>.5,1,0))#threshold to assume prediction that a person has <=50K
summary(Prediction2)
```

```
##  MeanTrainer(Prob2f)
##  Min.   :0.0000
##  1st Qu.:1.0000
##  Median :1.0000
##  Mean   :0.8378
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```
Res2f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res2f[i]<-ifelse(Prediction2[i,1]==TwoFI[i,1],1,0)
```

```
}
Accuracy2f<-mean(Res2f) #The accuracy of predicting a person in this FIRST fold is 66.83%!
```

```
#Threefold recycling vectors
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(ThreeTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50
     TestSplitter[i,a]<-as.vector(ThreeTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are
     OtherTestSplitter[i,a]<-as.vector(ThreeTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean iss

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)

TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
```

```
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

Prob3f<-0 #probability collection
Prob3f<-as.data.frame(Prob3f)
Prob3f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:Te
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestNC) #LAPLACE
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNCA
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[TestNC,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:TestNC-1,],c(NA,1,1/Tra
```

```
## [[1]]
## [1]  Yugoslavia
## 40 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)

for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
```

```
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:T
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov        <NA>
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
ThreeTrain[1,1]
```

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

```
ThreeTrain[1,3]
```

```
## [1]  Male
## Levels:   Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob3f[i,]<-NBCMod(ThreeTrain[i,1],ThreeTrain[i,2],ThreeTrain[i,3],ThreeTrain[i,4],ThreeTrain[i,5],
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred3<-as.data.frame(0)
Pred3[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
Pred3<-as.data.frame(MeanTrainer(Prob3f))
summary(Pred3)
```

```
##  MeanTrainer(Prob3f)
##  Min.   :0.1868
##  1st Qu.:0.5333
##  Median :0.6134
##  Mean   :0.6079
##  3rd Qu.:0.6855
##  Max.   :0.8962
```

```
Prediction3<-0
Prediction3<-as.data.frame(ifelse(Pred3>.5,1,0))#threshold to assume prediction that a person has <=50K
summary(Prediction3)
```

```
##  MeanTrainer(Prob3f)
##  Min.   :0.0000
##  1st Qu.:1.0000
##  Median :1.0000
##  Mean   :0.8292
##  3rd Qu.:1.0000
```

```
##  Max.   :1.0000
```

```
Res3f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res3f[i]<-ifelse(Prediction3[i,1]==ThreeFI[i,1],1,0)
}
Accuracy3f<-mean(Res3f) #The accuracy of predicting a person in this FIRST fold is 67.51%!
#Replace XTrain XFI, ProbXf
#REPLACE Pred2,Prediction2,Res2f,Accuracy2f
```

```
#Fourfold recycling vectors
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country"
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(FourTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50K
     TestSplitter[i,a]<-as.vector(FourTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are k
     OtherTestSplitter[i,a]<-as.vector(FourTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean iss

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)
```

```
TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

Prob4f<-0 #probability collection
Prob4f<-as.data.frame(Prob4f)
Prob4f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:T
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestNC) #LAPLACE
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNC
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[TestNC,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:TestNC-1,],c(NA,1,1/Tra
```

```
## [[1]]
## [1]  Yugoslavia
## 40 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)
```

```
for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:T
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov        <NA>
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
FourTrain[1,1]
```

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

```
FourTrain[1,3]
```

```
## [1]  Male
## Levels:   Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob4f[i,]<-NBCMod(FourTrain[i,1],FourTrain[i,2],FourTrain[i,3],FourTrain[i,4],FourTrain[i,5],FourT
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred4<-as.data.frame(0)
Pred4[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
Pred4<-as.data.frame(MeanTrainer(Prob4f))
summary(Pred4)
```

```
##  MeanTrainer(Prob4f)
##  Min.   :0.1896
##  1st Qu.:0.5349
##  Median :0.6150
##  Mean   :0.6108
##  3rd Qu.:0.6878
##  Max.   :0.9107
```

```
Prediction4<-0
Prediction4<-as.data.frame(ifelse(Pred4>.5,1,0))#threshold to assume prediction that a person has <=50K
summary(Prediction4)
```

```
##  MeanTrainer(Prob4f)
##  Min.   :0.0000
##  1st Qu.:1.0000
##  Median :1.0000
##  Mean   :0.8375
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```
Res4f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res4f[i]<-ifelse(Prediction4[i,1]==FourFI[i,1],1,0)
}
Accuracy4f<-mean(Res4f) #The accuracy of predicting a person in this FIRST fold is 67.81%!
#Replace XTrain XFI, ProbXf
#REPLACE Pred2,Prediction2,Res2f,Accuracy2f
```

```
#Fivefold recycling vectors
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country"
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(FiveTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50K
     TestSplitter[i,a]<-as.vector(FiveTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are k
     OtherTestSplitter[i,a]<-as.vector(FiveTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean iss

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
```

```
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)

TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

Prob5f<-0 #probability collection
Prob5f<-as.data.frame(Prob5f)
Prob5f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:T
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestNC) #LAPLACE
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNC
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[TestNC,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:TestNC-1,],c(NA,1,1/Tra
```

```
## [[1]]
## [1]  Yugoslavia
## 40 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)

for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:T
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov        <NA>
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
FiveTrain[1,1]
```

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

```
FiveTrain[1,3]
```

```
## [1]  Male
## Levels:   Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob5f[i,]<-NBCMod(FiveTrain[i,1],FiveTrain[i,2],FiveTrain[i,3],FiveTrain[i,4],FiveTrain[i,5],FiveT
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred5<-as.data.frame(0)
Pred5[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
Pred5<-as.data.frame(MeanTrainer(Prob5f))
summary(Pred5)
```

```
##  MeanTrainer(Prob5f)
##  Min.   :0.2009
##  1st Qu.:0.5337
##  Median :0.6129
```

```
##  Mean   :0.6088
##  3rd Qu.:0.6858
##  Max.   :0.9114
```

```
Prediction5<-0
Prediction5<-as.data.frame(ifelse(Pred5>.5,1,0))#threshold to assume prediction that a person has <=50K
summary(Prediction5)
```

```
##  MeanTrainer(Prob5f)
##  Min.   :0.0000
##  1st Qu.:1.0000
##  Median :1.0000
##  Mean   :0.8338
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```
Res5f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res5f[i]<-ifelse(Prediction5[i,1]==FiveFI[i,1],1,0)
}
Accuracy5f<-mean(Res5f) #The accuracy of predicting a person in this FIRST fold is 67.11%!
#Replace XTrain*12 XFI*1, ProbXf*6
#REPLACE PredX*5,PredictionX*4,Res5f,Accuracy2f*1
```

```
#Sixfold recycling vectors
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(SixTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50K
     TestSplitter[i,a]<-as.vector(SixTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are key
     OtherTestSplitter[i,a]<-as.vector(SixTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean iss

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
```

```
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)

TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

Prob6f<-0 #probability collection
Prob6f<-as.data.frame(Prob6f)
Prob6f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:T
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestNC) #LAPLACE
```

```
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNC
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[TestNC,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:TestNC-1,],c(NA,1,1/Tra
```

```
## [[1]]
## [1]  Yugoslavia
## 40 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)

for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:T
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov        <NA>
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
SixTrain[1,1]
```

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

```
SixTrain[1,3]
```

```
## [1]  Male
## Levels:  Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob6f[i,]<-NBCMod(SixTrain[i,1],SixTrain[i,2],SixTrain[i,3],SixTrain[i,4],SixTrain[i,5],SixTrain[i
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred6<-as.data.frame(0)
Pred6[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
```

```r
Pred6<-as.data.frame(MeanTrainer(Prob6f))
summary(Pred6)
```

```
##  MeanTrainer(Prob6f)
##  Min.   :0.2017
##  1st Qu.:0.5353
##  Median :0.6131
##  Mean   :0.6091
##  3rd Qu.:0.6860
##  Max.   :0.9102
```

```r
Prediction6<-0
Prediction6<-as.data.frame(ifelse(Pred6>.5,1,0))#threshold to assume prediction that a person has <=50K
summary(Prediction6)
```

```
##  MeanTrainer(Prob6f)
##  Min.   :0.0000
##  1st Qu.:1.0000
##  Median :1.0000
##  Mean   :0.8385
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```r
Res6f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res6f[i]<-ifelse(Prediction6[i,1]==SixFI[i,1],1,0)
}
Accuracy6f<-mean(Res6f) #The accuracy of predicting a person in this FIRST fold is 67.54%!
#Replace XTrain*12 XFI*1, ProbXf*6
#REPLACE PredX*5,PredictionX*4,ResXf*3,Accuracy2f*1
```

```r
#Sevenfold recycling vectors
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(SevenTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50
     TestSplitter[i,a]<-as.vector(SevenTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are
     OtherTestSplitter[i,a]<-as.vector(SevenTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
```

```
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean iss

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)

TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

Prob7f<-0 #probability collection
Prob7f<-as.data.frame(Prob7f)
Prob7f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:T
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestMODNC<-nrow(FrequencyTNCNAMES) #Special table to fit laplace due to unique issue with table size
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestMODNC) #LAPLACE FOR FrequencyTNC TABLE
{
  ifelse(as.vector(FrequencyTNCNAMES[i,1])!=as.vector(FrequencyTNC[i,1]), FrequencyTNC<-rbind(FrequencyT
}
NCRows<-nrow(FrequencyTNC) #Adds laplace estimator at the end just in case
FrequencyTNC<-FrequencyTNC[-NCRows,] #removes extra row added
ifelse(is.na(FrequencyTNC[TestNC,1])==TRUE,FrequencyTNC<-rbind(FrequencyTNC[1:TestNC-1,],c(NA,1,1/TrainT
```

```
## [[1]]
## [1]  Vietnam
## 41 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
for(i in 1:NCRows) #LAPLACE- fixed NC table
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNCA
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[NCRows,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:NCRows-1,],c(NA,1,1/Tra
```

```
## [[1]]
##  [1]  ?                   Cambodia            Canada
##  [4]  China               Columbia            Cuba
##  [7]  Dominican-Republic  Ecuador             El-Salvador
## [10]  England             France              Germany
## [13]  Greece              Guatemala           Haiti
## [16]  Honduras            <NA>                Hong
## [19]  Hungary             India               Iran
## [22]  Ireland             Italy               Jamaica
## [25]  Japan               Laos                Mexico
## [28]  Nicaragua           <NA>                Peru
## [31]  Philippines         Poland              Portugal
## [34]  Puerto-Rico         Scotland            South
## [37]  Taiwan              Thailand            Trinadad&Tobago
## [40]  United-States       Vietnam             Yugoslavia
## [43] <NA>
## 40 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)

for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
```

```
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:T
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov         <NA>
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
SevenTrain[1,1]
```

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

```
SevenTrain[1,3]
```

```
## [1]  Male
## Levels:   Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob7f[i,]<-NBCMod(SevenTrain[i,1],SevenTrain[i,2],SevenTrain[i,3],SevenTrain[i,4],SevenTrain[i,5],S
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred7<-as.data.frame(0)
Pred7[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
Pred7<-as.data.frame(MeanTrainer(Prob7f))
Mean7<-colMeans(Pred7,na.rm = TRUE)
typeof(Mean7)
```

```
## [1] "double"
```

```
#for(i in 1:ValidationTrows)
#{
 #Pred7[i,1]<-ifelse(is.na(Pred7[i,1])==TRUE,Pred7[i,1]<-Mean7,Pred7[i,1]) #NAs have been found ~.008% 
#}
summary(Pred7)
```

```
##  MeanTrainer(Prob7f)
##  Min.   :0.2024
##  1st Qu.:0.5339
##  Median :0.6117
##  Mean   :0.6083
##  3rd Qu.:0.6852
##  Max.   :0.9112
```

```
Prediction7<-0
Prediction7<-as.data.frame(ifelse(Pred7>.5,1,0))#threshold to assume prediction that a person has <=50K
```

```r
summary(Prediction7)
```

```
##  MeanTrainer(Prob7f)
##  Min.   :0.0000
##  1st Qu.:1.0000
##  Median :1.0000
##  Mean   :0.8378
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```r
Res7f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res7f[i]<-ifelse(Prediction7[i,1]==SevenFI[i,1],1,0)
}
Accuracy7f<-mean(Res7f) #The accuracy of predicting a person in this FIRST fold is 67.75%!
#Replace XTrain*12 XFI*1, ProbXf*6
#REPLACE PredX*10,PredictionX*4,ResXf*3,Accuracy2f*1
```

```r
#Eightfold recycling vectors
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country"
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(EightTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50
     TestSplitter[i,a]<-as.vector(EightTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are
     OtherTestSplitter[i,a]<-as.vector(EightTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean iss

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
```

```
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)

TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

Prob8f<-0 #probability collection
Prob8f<-as.data.frame(Prob8f)
Prob8f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:T
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestNC) #LAPLACE
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNC
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
```

```
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[TestNC,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:TestNC-1,],c(NA,1,1/Tra
```

```
## [[1]]
## [1]  Yugoslavia
## 40 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)

for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:Te
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov        <NA>
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
EightTrain[1,1]
```

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

```
EightTrain[1,3]
```

```
## [1]  Male
## Levels:   Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob8f[i,]<-NBCMod(EightTrain[i,1],EightTrain[i,2],EightTrain[i,3],EightTrain[i,4],EightTrain[i,5],E
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred8<-as.data.frame(0)
Pred8[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
Pred8<-as.data.frame(MeanTrainer(Prob8f))
Mean7<-colMeans(Pred8,na.rm = TRUE)
typeof(Mean7)
```

```
## [1] "double"
```

```
#for(i in 1:ValidationTrows)
#{
 #Pred8[i,1]<-ifelse(is.na(Pred8[i,1])==TRUE,Pred8[i,1]<-Mean7,Pred8[i,1]) #NO NAs found commenting out
#}
summary(Pred8)
```

```
##  MeanTrainer(Prob8f)
##  Min.   :0.2055
##  1st Qu.:0.5351
##  Median :0.6133
##  Mean   :0.6097
##  3rd Qu.:0.6866
##  Max.   :0.9107
```

```
Prediction8<-0
Prediction8<-as.data.frame(ifelse(Pred8>.5,1,0))#threshold to assume prediction that a person has <=50K
summary(Prediction8)
```

```
##  MeanTrainer(Prob8f)
##  Min.   :0.0000
##  1st Qu.:1.0000
##  Median :1.0000
##  Mean   :0.8403
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```
Res8f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res8f[i]<-ifelse(Prediction8[i,1]==EightFI[i,1],1,0)
}
Accuracy8f<-mean(Res8f) #The accuracy of predicting a person in this FIRST fold is 66.85%!
#Replace XTrain*12 XFI*1, ProbXf*6
#REPLACE PredX*10,PredictionX*4,ResXf*3,AccuracyXf*1
```

```
#Ninefold recycling vectors
TestSplitter<-0
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(NineTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50K
```

```
    TestSplitter[i,a]<-as.vector(NineTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are k
    OtherTestSplitter[i,a]<-as.vector(NineTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean iss

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)

TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

Prob9f<-0 #probability collection
Prob9f<-as.data.frame(Prob9f)
Prob9f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
```

```
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:T
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestNC) #LAPLACE
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNC
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[TestNC,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:TestNC-1,],c(NA,1,1/Tra
```

```
## [[1]]
## [1]  Yugoslavia
## 40 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)

for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:T
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov        <NA>
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
NineTrain[1,1]
```

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

```
NineTrain[1,3]
```

```
## [1]  Male
## Levels:  Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob9f[i,]<-NBCMod(NineTrain[i,1],NineTrain[i,2],NineTrain[i,3],NineTrain[i,4],NineTrain[i,5],NineT
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred9<-as.data.frame(0)
Pred9[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
Pred9<-as.data.frame(MeanTrainer(Prob9f))
Mean7<-colMeans(Pred9,na.rm = TRUE)
typeof(Mean7)
```

```
## [1] "double"
```

```
#for(i in 1:ValidationTrows)
#{
 #Pred9[i,1]<-ifelse(is.na(Pred9[i,1])==TRUE,Pred9[i,1]<-Mean7,Pred9[i,1]) #NO NAs found commenting out
#}
summary(Pred9)
```

```
##  MeanTrainer(Prob9f)
##  Min.   :0.2250
##  1st Qu.:0.5349
##  Median :0.6131
##  Mean   :0.6089
##  3rd Qu.:0.6873
##  Max.   :0.9107
```

```
Prediction9<-0
Prediction9<-as.data.frame(ifelse(Pred9>.5,1,0))#threshold to assume prediction that a person has <=50K
summary(Prediction9)
```

```
##  MeanTrainer(Prob9f)
##  Min.   :0.0000
##  1st Qu.:1.0000
##  Median :1.0000
##  Mean   :0.8378
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```
Res9f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res9f[i]<-ifelse(Prediction9[i,1]==NineFI[i,1],1,0)
}
Accuracy9f<-mean(Res9f) #The accuracy of predicting a person in this FIRST fold is 67.26%!
#Replace XTrain*12 XFI*1, ProbXf*6
#REPLACE PredX*10,PredictionX*4,ResXf*3,AccuracyXf*1
```

```
#Tenfold recycling vectors
TestSplitter<-0
```

```
TestSplitter<-as.data.frame(TestSplitter)
TestSplitter[1:TrainTotal,]<-NA;
TestSplitter[,1:7]<-NA;
OtherTestSplitter<-0
OtherTestSplitter<-as.data.frame(OtherTestSplitter)
OtherTestSplitter[1:TrainTotal,]<-NA;
OtherTestSplitter[,1:7]<-NA;
colnames(TestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
colnames(OtherTestSplitter)<-c("income","race","sex","isAdult","workclass","education","native.country")
TestSplitter$income<-as.character(TestSplitter$income)
OtherTestSplitter$income<-as.character(OtherTestSplitter$income)


for(i in 1:TrainTotal)
{
   for(a in 1:7) #REMINDER change to 1:7 in future tables
     ifelse(TenTrain[i,1]==1, #splits into two tables for categorical probability calculation of <=50K
     TestSplitter[i,a]<-as.vector(TenTrain[i,a]), #TestSplitter and OtherTestSplitter dataframes are kep
     OtherTestSplitter[i,a]<-as.vector(TenTrain[i,a]))
}
Splitter<-TestSplitter[complete.cases(TestSplitter),]
AltSplitter<-OtherTestSplitter[complete.cases(OtherTestSplitter),]

Splitter$income<-as.factor(Splitter$income)#Remember to convert to integer to prevent boolean issues!
AltSplitter$income<-as.factor(AltSplitter$income)#Remember to convert to integer to prevent boolean issu

OATot<-nrow(Income2)
TotalInL<-nrow(Splitter)
FrequencyTWorkclass<-as.data.frame(table(Splitter$workclass))
FrequencyTWorkclass[,3]<-as.data.frame(FrequencyTWorkclass[,2]/TotalInL)
FrequencyTEducation<-as.data.frame(table(Splitter$education))
FrequencyTEducation[,3]<-as.data.frame(FrequencyTEducation[,2]/TotalInL)
FrequencyTRace<-as.data.frame(table(Splitter$race))
FrequencyTRace[,3]<-as.data.frame(FrequencyTRace[,2]/TotalInL)
FrequencyTSex<-as.data.frame(table(Splitter$sex))
FrequencyTSex[,3]<-as.data.frame(FrequencyTSex[,2]/TotalInL)
FrequencyTNC<-as.data.frame(table(Splitter$native.country))
FrequencyTNC[,3]<-as.data.frame(FrequencyTNC[,2]/TotalInL)
FrequencyTIncome<-as.data.frame(table(Splitter$income))
FrequencyTIncome[,3]<-as.data.frame(FrequencyTIncome[,2]/OATot)
FrequencyTAdult<-as.data.frame(table(Splitter$isAdult))
FrequencyTAdult[,3]<-as.data.frame(FrequencyTAdult[,2]/TotalInL)

TotalInG<-nrow(AltSplitter)
FrequencyTWorkclassA<-as.data.frame(table(AltSplitter$workclass))
FrequencyTWorkclassA[,3]<-as.data.frame(FrequencyTWorkclassA[,2]/TotalInG)
FrequencyTEducationA<-as.data.frame(table(AltSplitter$education))
FrequencyTEducationA[,3]<-as.data.frame(FrequencyTEducationA[,2]/TotalInG)
FrequencyTRaceA<-as.data.frame(table(AltSplitter$race))
FrequencyTRaceA[,3]<-as.data.frame(FrequencyTRaceA[,2]/TotalInG)
FrequencyTSexA<-as.data.frame(table(AltSplitter$sex))
FrequencyTSexA[,3]<-as.data.frame(FrequencyTSexA[,2]/TotalInG)
FrequencyTNCA<-as.data.frame(table(AltSplitter$native.country))
```

```
FrequencyTNCA[,3]<-as.data.frame(FrequencyTNCA[,2]/TotalInG)
FrequencyTIncomeA<-as.data.frame(table(AltSplitter$income))
FrequencyTIncomeA[,3]<-as.data.frame(FrequencyTIncomeA[,2]/OATot) #Noticably only people who are adults
FrequencyTAdultA<-as.data.frame(table(AltSplitter$isAdult))
FrequencyTAdultA[,3]<-as.data.frame(FrequencyTAdultA[,2]/TotalInG)

Prob10f<-0 #probability collection
Prob10f<-as.data.frame(Prob10f)
Prob10f[1:TrainTotal,]<-NA;


TestEd<-nrow(FrequencyTEducation)
TestEdA<-nrow(FrequencyTEducationA)
for(i in 1:TestEd) #LAPLACE
{
  ifelse(as.vector(FrequencyTEducation[i,1])!=as.vector(FrequencyTEducationA[i,1]), FrequencyTEducationA
         FrequencyTEducationA[i,])
}
EducationARows<-nrow(FrequencyTEducationA) #Adds laplace estimator at the end just in case
FrequencyTEducationA<-FrequencyTEducationA[-EducationARows,] #removes extra row added
ifelse(is.na(FrequencyTEducationA[TestEd,1])==TRUE,FrequencyTEducationA<-rbind(FrequencyTEducationA[1:T
```

```
## [[1]]
## [1]  Some-college
## 15 Levels:  10th  11th  12th  1st-4th  5th-6th  7th-8th ...  Some-college
```

```
TestNC<-nrow(FrequencyTNC)
TestNCA<-nrow(FrequencyTNCA)

for(i in 1:TestNC) #LAPLACE
{
  ifelse(as.vector(FrequencyTNC[i,1])!=as.vector(FrequencyTNCA[i,1]), FrequencyTNCA<-rbind(FrequencyTNC
}
NCARows<-nrow(FrequencyTNCA) #Adds laplace estimator at the end just in case
FrequencyTNCA<-FrequencyTNCA[-NCARows,] #removes extra row added
ifelse(is.na(FrequencyTNCA[TestNC,1])==TRUE,FrequencyTNCA<-rbind(FrequencyTNCA[1:TestNC-1,],c(NA,1,1/Tra
```

```
## [[1]]
## [1]  Yugoslavia
## 40 Levels:  ?  Cambodia  Canada  China  Columbia ...  Yugoslavia
```

```
TestWC<-nrow(FrequencyTWorkclass)
TestWCA<-nrow(FrequencyTWorkclassA)

for(i in 1:TestWC) #LAPLACE
{
  ifelse(as.vector(FrequencyTWorkclass[i,1])!=as.vector(FrequencyTWorkclassA[i,1]), FrequencyTWorkclassA
         FrequencyTWorkclassA[i,])
}
WorkARows<-nrow(FrequencyTWorkclassA) #Adds laplace estimator at the end just in case
FrequencyTWorkclassA<-FrequencyTWorkclassA[-WorkARows,] #removes extra row added
ifelse(is.na(FrequencyTWorkclassA[TestWC,1])==TRUE,FrequencyTWorkclassA<-rbind(FrequencyTWorkclassA[1:T
```

```
## [[1]]
## [1]  ?                 Federal-gov       Local-gov         <NA>
```

```
## [5]  Private           Self-emp-inc      Self-emp-not-inc  State-gov
## [9] <NA>
## 7 Levels:  ?  Federal-gov  Local-gov  Private ...  State-gov
```

```
TestProb<-NBCMod(1," White"," Female",1," Federal-gov"," Bachelors"," India")
TenTrain[1,1]
```

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

```
TenTrain[1,3]
```

```
## [1]  Male
## Levels:  Female  Male
```

```
for(i in 1:TrainTotal)
{
    Prob10f[i,]<-NBCMod(TenTrain[i,1],TenTrain[i,2],TenTrain[i,3],TenTrain[i,4],TenTrain[i,5],TenTrain[
}
```

```
#This takes the means of the training data (of 9 rows EACH from 29304 rows) to accurately predict the t
Trainerrows<-round(TrainTotal/Onefold)
ValidationTrows<-TrainTotal/Trainerrows
Pred10<-as.data.frame(0)
Pred10[1:ValidationTrows,]<-NA #This stores the predictions
MeanTrainer<-function(probs){
  predstack<-0;
  for(s in seq(9,TrainTotal,9)) #Goes through a sequence of 9 to fit into the testing data
  {
    predstack[s/9]<-(probs[s-8,1]+probs[s-7,1]+probs[s-6,1]+probs[s-5,1]+probs[s-4,1]+probs[s-3,1]+probs
  }
  MeanTrainer<-predstack
}
Pred10<-as.data.frame(MeanTrainer(Prob10f))
Mean7<-colMeans(Pred10,na.rm = TRUE)
typeof(Mean7)
```

```
## [1] "double"
```

```
#for(i in 1:ValidationTrows)
#{
 #Pred10[i,1]<-ifelse(is.na(Pred10[i,1])==TRUE,Pred10[i,1]<-Mean7,Pred10[i,1]) #NO NAs found commenting
#}
summary(Pred10)
```

```
##  MeanTrainer(Prob10f)
##  Min.   :0.1864
##  1st Qu.:0.5357
##  Median :0.6150
##  Mean   :0.6108
##  3rd Qu.:0.6889
##  Max.   :0.9106
```

```
Prediction10<-0
Prediction10<-as.data.frame(ifelse(Pred10>.5,1,0))#threshold to assume prediction that a person has <=5
summary(Prediction10)
```

```
##  MeanTrainer(Prob10f)
##  Min.   :0.0000
```

```
##  1st Qu.:1.0000
##  Median :1.0000
##  Mean   :0.8403
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```
Res10f<-0 #measure accuracy
for(i in 1:ValidationTrows)
{
  Res10f[i]<-ifelse(Prediction10[i,1]==TenFI[i,1],1,0)
}
Accuracy10f<-mean(Res10f) #The accuracy of predicting a person in this FIRST fold is 67.54%!
#Replace XTrain*12 XFI*1, ProbXf*6
#REPLACE PredX*10,PredictionX*4,ResXf*3,AccuracyXf*1
```

```
NBCV10F<-(Accuracy1f+Accuracy2f+Accuracy3f+Accuracy4f+Accuracy5f+Accuracy6f+Accuracy7f+Accuracy8f+Accura
```

The model made has a likelihood of 67.36 percent. Optimistically it is fairly better than a random guess at best.