PleBUS/CSC 328 Data Analytics
Exam #3 Take Home Edition

Step 0: Your company is developing a product that will only be attractive to people with higher incomes, and you've been asked by the Marketing department to create an algorithmic model that can accurately predict which people make more than $50K per year. Every person that Marketing will target costs the firm $500, so it's important to target only potential consumers. The dataset "Income_Pred" will used to make your predictive models, but Marketing told you that they cannot reliably obtain country of residence, so that attribute must not be included in any modeling.

```
Income_datum <- Income_data %>%
  dplyr::select(Income, everything()) %>%
  dplyr::select(-Country, -FamilyRole) #due to high correlation with Sex
```

Create a Word document, in which you will document everything (code, output, plots, etc.) in your quest for scientific discovery.

Step 1: Perform a complete inspection and analysis of the raw data.

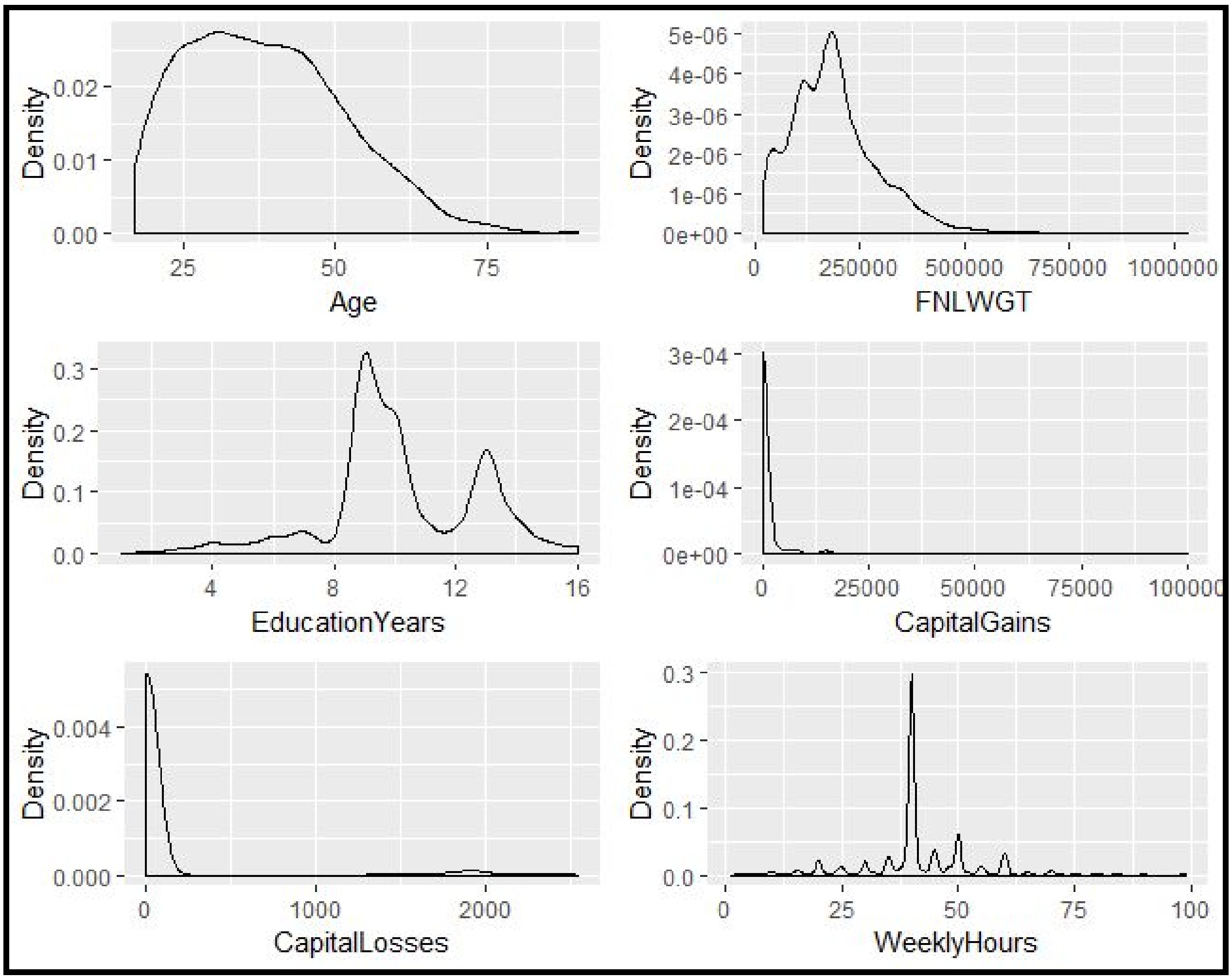
```
> IncomeData <- Income_Pred
> Income_data <- IncomeData[1:5000, ]
> glimpse(Income_data)
```

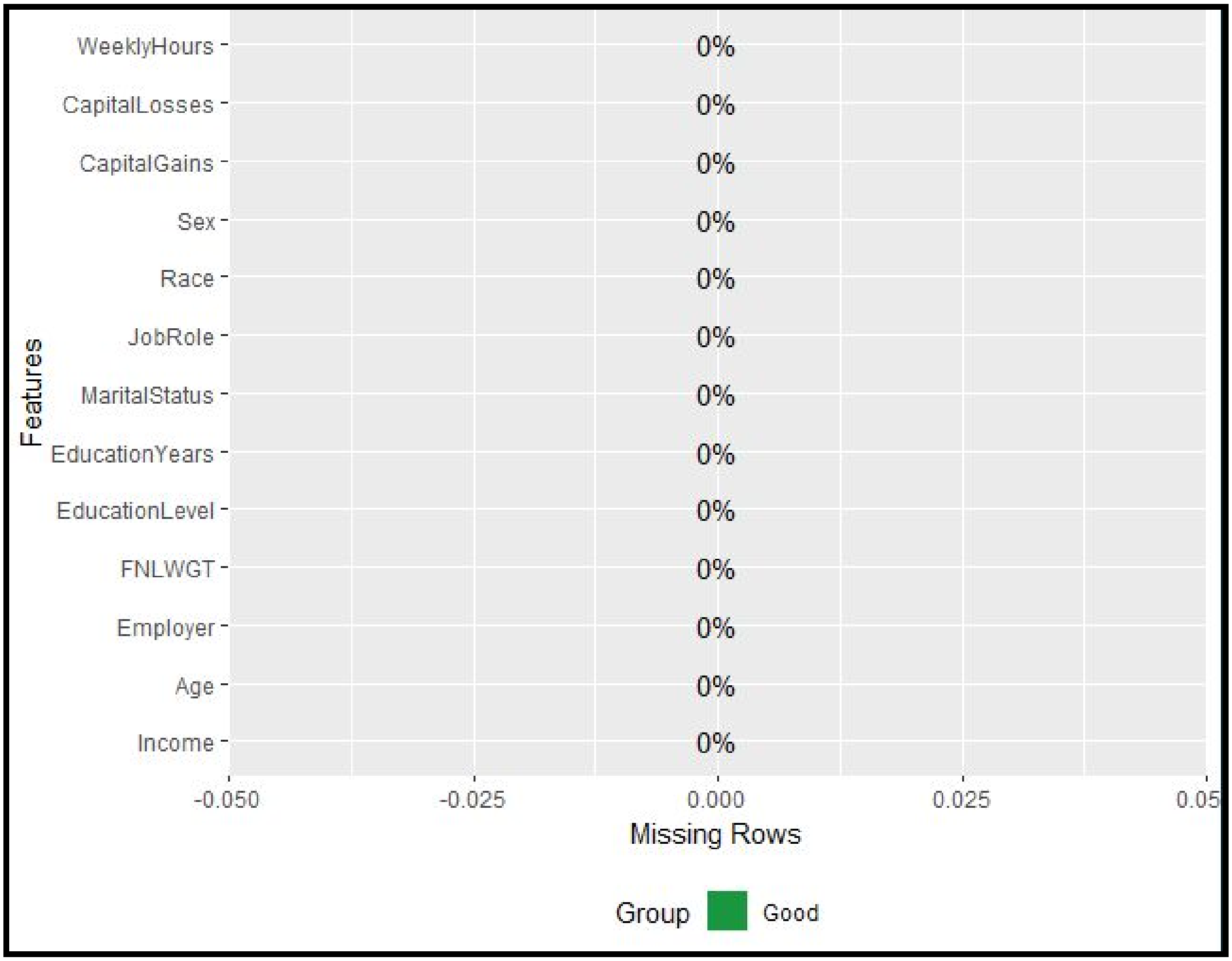
```
> IncomeData <- Income_Pred
> Income_data <- IncomeData[1:5000, ]
> glimpse(Income_data)
Observations: 5,000
Variables: 15
$ Age            <int> 39, 50, 38, 53, 28, 37, 49, 52, 31, 42, 37, 30, 23, 32, 40, 34, 25, 32, 38, 43, 40, 54, 35, 43, 59, 56, 19, 54, 39, 49, 23, 20, 45, 30, 22, 4...
$ Employer       <fct>  State-gov,  Self-emp-not-inc,  Private,  Private,  Private,  Private,  Private,  Self-emp-not-inc,  Private,  Private,  Private,  State-gov,...
$ FNLWGT         <int> 77516, 83311, 215646, 234721, 338409, 284582, 160187, 209642, 45781, 159449, 280464, 141297, 122272, 205019, 121772, 245487, 176756, 186824, ...
$ EducationLevel <fct>  Bachelors,  Bachelors,  HS-grad,  11th,  Bachelors,  Masters,  9th,  HS-grad,  Masters,  Bachelors,  Some-college,  Bachelors,  Bachelors,  ...
$ EducationYears <int> 13, 13, 9, 7, 13, 14, 5, 9, 14, 13, 10, 13, 13, 12, 11, 4, 9, 9, 7, 14, 16, 9, 5, 7, 9, 13, 9, 10, 9, 9, 12, 10, 13, 10, 10, 7, 10, 9, 10, 12...
$ MaritalStatus  <fct>  Never-married,  Married-civ-spouse,  Divorced,  Married-civ-spouse,  Married-civ-spouse,  Married-civ-spouse,  Married-spouse-absent,  Marri...
$ JobRole        <fct>  Adm-clerical,  Exec-managerial,  Handlers-cleaners,  Handlers-cleaners,  Prof-specialty,  Exec-managerial,  Other-service,  Exec-managerial,...
$ FamilyRole     <fct>  Not-in-family,  Husband,  Not-in-family,  Husband,  Wife,  Wife,  Not-in-family,  Husband,  Not-in-family,  Husband,  Husband,  Husband,  Ow...
$ Race           <fct>  White,  White,  White,  Black,  Black,  White,  Black,  White,  White,  White,  Black,  Asian-Pac-Islander,  White,  Black,  Asian-Pac-Islan...
$ Sex            <fct>  Male,  Male,  Male,  Male,  Female,  Female,  Female,  Male,  Female,  Male,  Male,  Male,  Female,  Male,  Male,  Male,  Male,  Male,  Male...
$ CapitalGains   <int> 2174, 0, 0, 0, 0, 0, 0, 0, 14084, 5178, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
```

```
Income           <fct>  <=50K,  <=50K,  <=50K,  <=50K,  <=50K,  <=50K,  <=50K,  >50K,  >50K,  >50K,  >50K,  >50K,  <=50K,  <=50K,  >50K,  <=50K,  <=50K,  <=50K
Income_data[Income_data== " ?"] = NA
Income_data[Income_data== "?"] = NA
levels(Income_data$Employer)[levels(Income_data$Employer)==" ?"] = NA
levels(Income_data$JobRole)[levels(Income_data$JobRole)==" ?"] = NA
Income_data$JobRole
  [1]  Adm-clerical       Exec-managerial    Handlers-cleaners  Handlers-cleaners  Prof-specialty     Exec-managerial    Other-service      Exec-managerial
  [9]  Prof-specialty     Exec-managerial    Exec-managerial    Prof-specialty     Adm-clerical       Sales              Craft-repair       Transport-moving
 [17]  Farming-fishing    Machine-op-inspct  Sales              Exec-managerial    Prof-specialty     Other-service      Farming-fishing    Transport-moving
 [25]  Tech-support       Tech-support       Craft-repair       <NA>               Exec-managerial    Craft-repair       Protective-serv    Sales
 [33]  Exec-managerial    Adm-clerical       Other-service      Machine-op-inspct  Machine-op-inspct  Adm-clerical       Sales              Prof-specialty
 [41]  Machine-op-inspct  Prof-specialty     Tech-support       Adm-clerical       Handlers-cleaners  Prof-specialty     Machine-op-inspct  Exec-managerial
 [49]  Craft-repair       Prof-specialty     Exec-managerial    Other-service      Prof-specialty     Exec-managerial    Exec-managerial    Tech-support
 [57]  Machine-op-inspct  Other-service      Adm-clerical       Machine-op-inspct  Sales              <NA>               Transport-moving   Prof-specialty
 [65]  Tech-support       Craft-repair       Adm-clerical       Adm-clerical       Exec-managerial    <NA>               Prof-specialty     Sales
 [73]  Sales              Machine-op-inspct  Prof-specialty     Other-service      Adm-clerical       <NA>               Other-service      Farming-fishing
 [81]  Sales              Other-service      Other-service      Sales              Craft-repair       Sales              Protective-serv    Prof-specialty
 [89]  Sales              Prof-specialty     Prof-specialty     Craft-repair       Machine-op-inspct  Sales              Protective-serv    Handlers-cleaners
 [97]  Prof-specialty     Sales              Exec-managerial    Other-service      Exec-managerial    Exec-managerial    Prof-specialty     Tech-support
[105]  Craft-repair       Craft-repair       <NA>               Handlers-cleaners  Adm-clerical       Handlers-cleaners  Sales              Prof-specialty
[113]  Other-service      Sales              Machine-op-inspct  Handlers-cleaners  Sales              Craft-repair       Sales              Craft-repair
[121]  Other-service      Exec-managerial    Exec-managerial    Prof-specialty     Other-service      Exec-managerial    Adm-clerical       Adm-clerical
[129] <NA>                Craft-repair       Sales              Other-service      Craft-repair       Sales              Tech-support       Prof-specialty
[137]  Craft-repair       Adm-clerical       Sales              Craft-repair       Craft-repair       Sales              Other-service      Prof-specialty
[145]  Tech-support       Transport-moving   Other-service      Other-service      Craft-repair       <NA>               Adm-clerical       Adm-clerical
[153]  Exec-managerial    Craft-repair       <NA>               Craft-repair       Handlers-cleaners  Sales              Craft-repair       Other-service
```

```
levels(Income_data$Employer)[levels(Income_data$Employer)==" ?"] = NA
levels(Income_data$Employer)
levels(Income_data$JobRole)[levels(Income_data$JobRole)==" ?"] = NA
Income_data$JobRole
levels(Income_data$JobRole)
glimpse(Income_data)

Income_data$Income <- ifelse(Income_data$Income ==" <=50K", "Bad", "Good")
Income_data$Income <- as.factor(Income_data$Income)
```

```
IncomeData <- na.omit(Income_datum)
plot_density(IncomeData)
glimpse(IncomeData)
plot_missing(IncomeData)
summary(IncomeData)
hetcor(IncomeData)
```






```
> summary(IncomeData)
  Income          Age                     Employer          FNLWGT               EducationLevel EducationYears                MaritalStatus              JobRole
 Bad :3481   Min.   :17.00   Private         :3435   Min.   :  19302   HS-grad     :1531   Min.   : 1.00   Divorced             : 645   Prof-specialty : 625
 Good:1188   1st Qu.:28.00   Self-emp-not-inc: 383   1st Qu.: 117502   Some-college:1021   1st Qu.: 9.00   Married-AF-spouse    :   4   Craft-repair   : 619
             Median :37.00   Local-gov       : 329   Median : 179717   Bachelors   : 805   Median :10.00   Married-civ-spouse   :2176   Exec-managerial: 618
             Mean   :38.56   State-gov       : 193   Mean   : 190859   Masters     : 242   Mean   :10.14   Married-spouse-absent:  60   Sales          : 588
             3rd Qu.:47.00   Self-emp-inc    : 182   3rd Qu.: 241962   Assoc-voc   : 204   3rd Qu.:13.00   Never-married        :1503   Adm-clerical   : 576
             Max.   :90.00   Federal-gov     : 146   Max.   :1033222   11th        : 173   Max.   :16.00   Separated            : 151   Other-service  : 495
                            (Other)          :   1                    (Other)      : 693                   Widowed              : 130  (Other)         :1148
                Race             Sex         CapitalGains     CapitalLosses        WeeklyHours
  Amer-Indian-Eskimo:  43   Female:1482   Min.   :    0   Min.   :   0.00   Min.   : 1.00
  Asian-Pac-Islander: 144   Male  :3187   1st Qu.:    0   1st Qu.:   0.00   1st Qu.:40.00
  Black             : 475                 Median :    0   Median :   0.00   Median :40.00
  Other             :  27                 Mean   : 1080   Mean   :  97.35   Mean   :41.16
  White             :3980                 3rd Qu.:    0   3rd Qu.:   0.00   3rd Qu.:45.00
                                          Max.   :99999   Max.   :2547.00   Max.   :99.00
```

```
> hetcor(IncomeData)

Two-Step Estimates

Correlations/Type of Correlation:
                Income        Age   Employer     FNLWGT EducationLevel EducationYears MaritalStatus    JobRole       Race        Sex CapitalGains CapitalLosses
Income               1 Polyserial Polychoric Polyserial     Polychoric     Polyserial    Polychoric Polychoric Polychoric Polychoric   Polyserial    Polyserial
Age             0.3162          1 Polyserial    Pearson     Polyserial        Pearson    Polyserial Polyserial Polyserial Polyserial      Pearson       Pearson
Employer       0.07339       <NA>          1 Polyserial     Polychoric     Polyserial    Polychoric Polychoric Polychoric Polychoric   Polyserial    Polyserial
FNLWGT        -0.03192   -0.08286       <NA>          1     Polyserial        Pearson    Polyserial Polyserial Polyserial Polyserial      Pearson       Pearson
EducationLevel 0.02052   -0.03827    0.01073   -0.04749              1     Polyserial    Polychoric Polychoric Polychoric Polychoric   Polyserial    Polyserial
EducationYears  0.4384     0.0321       <NA>   -0.05872         0.2268              1    Polyserial Polyserial Polyserial Polyserial      Pearson       Pearson
MaritalStatus  -0.3042    -0.2481   -0.03585    0.05154       -0.01036       -0.05743             1 Polychoric Polychoric Polychoric   Polyserial    Polyserial
JobRole        0.08767  0.0006056    0.02872   0.008494         -0.044        0.06835      0.006655          1 Polychoric Polychoric   Polyserial    Polyserial
Race            0.1718    0.04198     0.1333   -0.03861        0.03245        0.07479       -0.1303    0.01202          1 Polychoric   Polyserial    Polyserial
Sex             0.3849    0.08503     0.1424    0.04699       -0.03925        0.02876       -0.1271     0.1577     0.1473          1   Polyserial    Polyserial
CapitalGains    0.5418    0.06713       <NA>  -0.001329        0.02558         0.1152      -0.05424    0.02849    0.02618    0.08103            1       Pearson
CapitalLosses   0.1803    0.06087       <NA>   0.001296        0.01609        0.09389      -0.03595   0.007359    0.03895    0.09298     -0.03462             1
WeeklyHours     0.3339     0.0814       <NA>   -0.01376       0.007703         0.1581       -0.2024    0.04934    0.08783      0.309      0.07204       0.07999
               WeeklyHours
Income          Polyserial
Age                Pearson
Employer        Polyserial
FNLWGT             Pearson
EducationLevel  Polyserial
EducationYears     Pearson
MaritalStatus   Polyserial
JobRole         Polyserial
Race            Polyserial
Sex             Polyserial
CapitalGains       Pearson
CapitalLosses      Pearson
WeeklyHours              1

Standard Errors:
                Income     Age Employer  FNLWGT EducationLevel EducationYears MaritalStatus JobRole    Race     Sex CapitalGains CapitalLosses
Income
Age            0.01765
Employer        0.0227       0
FNLWGT         0.02014 0.01454        0
EducationLevel 0.02098 0.01521  0.01796 0.01519
EducationYears 0.01665 0.01462        0 0.01459        0.01396
MaritalStatus  0.02071 0.01439  0.01851 0.01564        0.01637        0.01564
JobRole        0.02028 0.01495  0.01766 0.01492        0.01557        0.01479       0.01612
Race           0.02996 0.02219  0.02513 0.02113        0.02263        0.02165        0.0228 0.02243
Sex            0.02407 0.01893  0.02218 0.01925        0.01988        0.01927       0.01947 0.01903 0.02768
CapitalGains    0.0206 0.01457        0 0.01464        0.01497        0.01444       0.01579 0.01474 0.02451  0.0254
CapitalLosses  0.01764 0.01458        0 0.01464         0.0151        0.01451       0.01575 0.01486 0.02357 0.02097      0.01462
WeeklyHours    0.01825 0.01454        0 0.01463        0.01527        0.01427       0.01493 0.01483 0.02216   0.018      0.01456       0.01454

n = 4669

P-values for Tests of Bivariate Normality:
                  Income       Age   Employer     FNLWGT EducationLevel EducationYears MaritalStatus    JobRole     Race Sex CapitalGains CapitalLosses
Income
Age            6.094e-52
```

Step 2: Prepare the data for any type of modeling exercise, including ensembles.

```
There were 33 warnings (use warnings() to see them)
> gulab <- recipe(Income~., data=IncomeData) %>%
+   step_center(all_numeric(), -all_outcomes()) %>%
+   step_scale(all_numeric(), -all_outcomes()) %>%
+   #step_bagimpute(all_nominal(), -all_outcomes()) %>%
+   step_YeoJohnson(all_numeric(), -all_outcomes()) %>%
+   step_nzv(all_predictors())%>%
+   step_dummy(all_nominal(), -all_outcomes(), one_hot = TRUE) %>%
+   prep(data = IncomeData)
> data_clean <- bake(gulab, new_data = IncomeData)
> glimpse(data_clean)
Observations: 4,669
Variables: 57
$ Income                        <fct> Bad, Bad, Bad, Bad, Bad, Bad, Bad, Good, Good, Good
$ Age                           <dbl> 0.03377929, 0.78036273, -0.04304184, 0.96199337, -0
$ FNLWGT                        <dbl> -1.326448701, -1.247522901, 0.221475208, 0.37755474
$ EducationYears                <dbl> 1.16806114, 1.16806114, -0.44447542, -1.20170852, 1
$ WeeklyHours                   <dbl> -0.09981299, -2.50414436, -0.09981299, -0.09981299,
$ Employer_X.Federal.gov        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
$ Employer_X.Local.gov          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
$ Employer_X.Never.worked       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
$ Employer_X.Private            <dbl> 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0,
$ Employer_X.Self.emp.inc       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
$ Employer_X.Self.emp.not.inc   <dbl> 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
$ Employer_X.State.gov          <dbl> 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
$ Employer_X.Without.pay        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
$ EducationLevel_X.10th         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
$ EducationLevel_X.11th         <dbl> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
$ EducationLevel_X.12th         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
$ EducationLevel_X.1st.4th      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
```

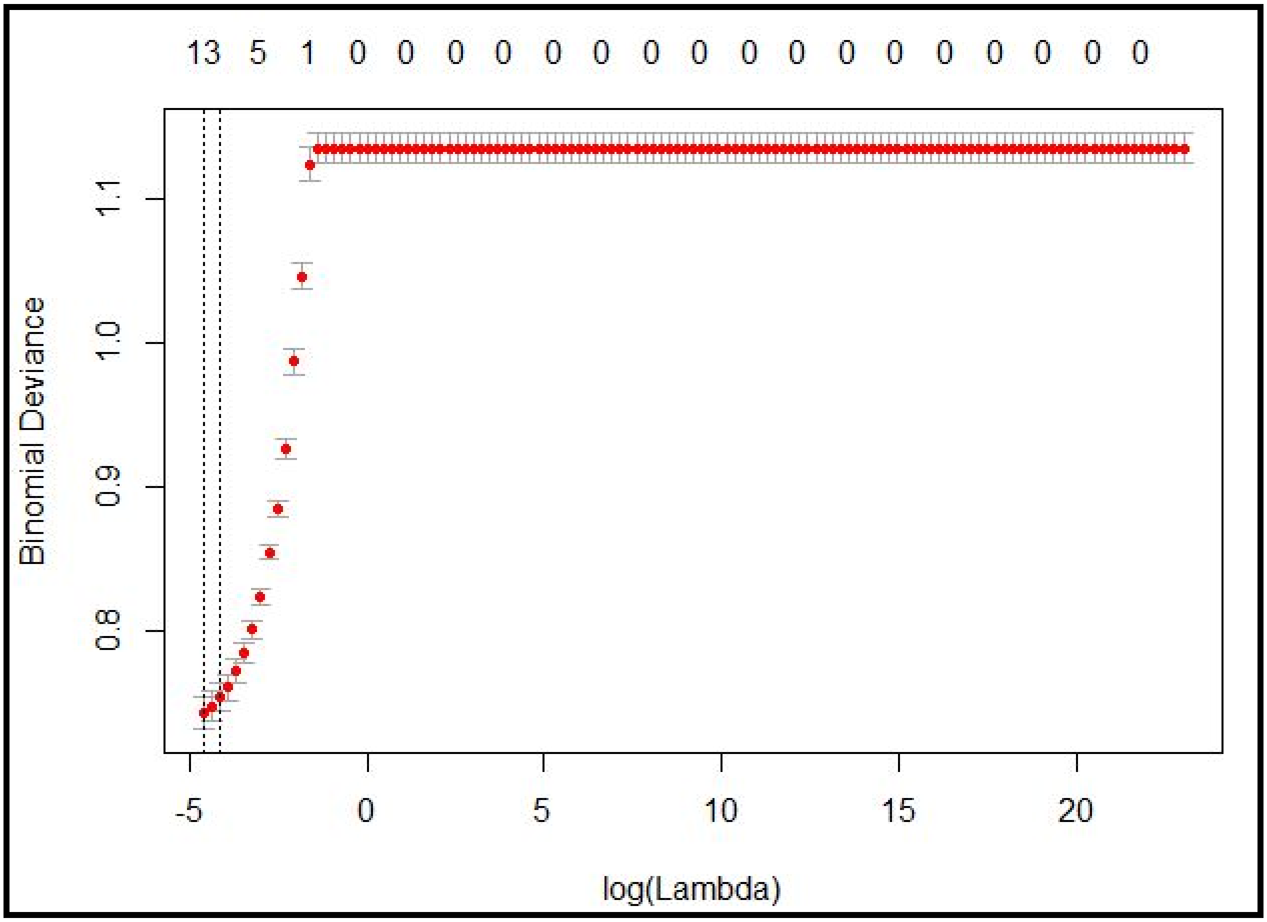
```
> plot_missing(data_clean)
>
```

Step 3: Identify only relevant predictors to be included in your model.

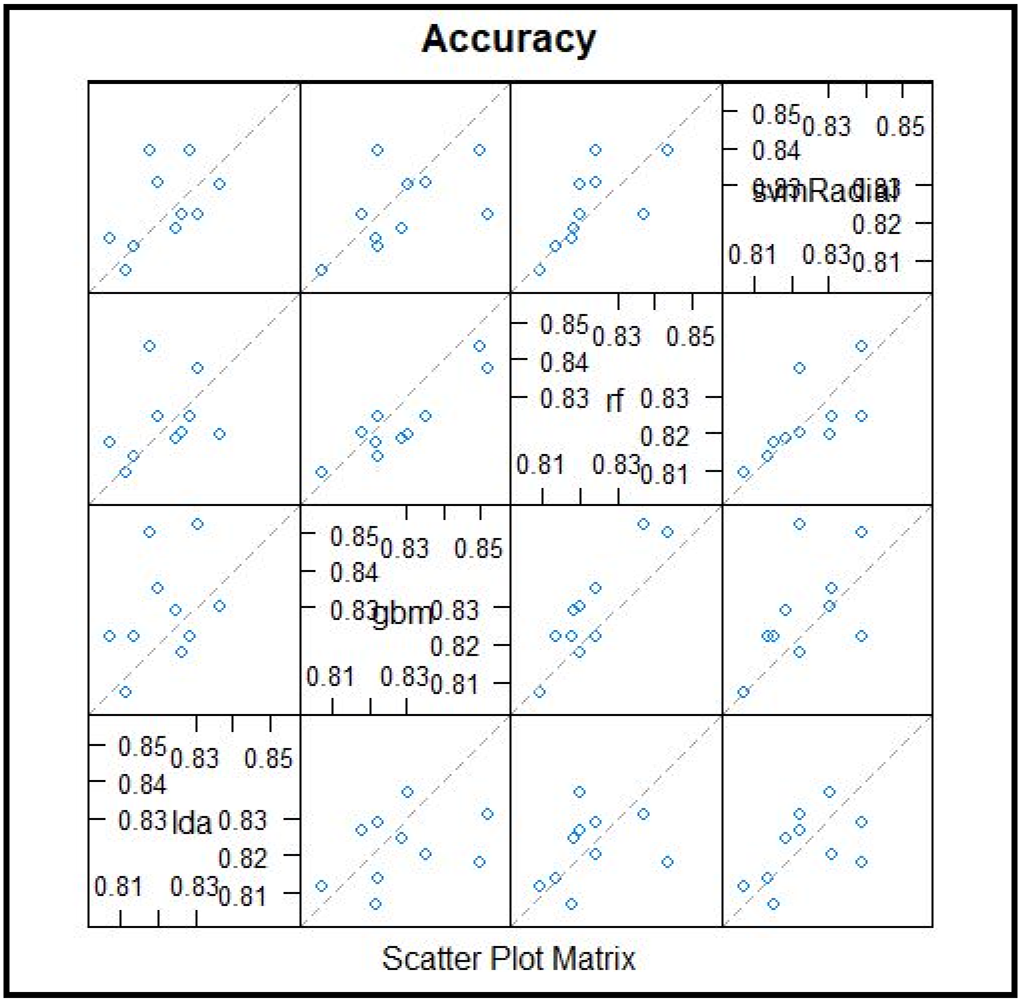
Ans: I used two approaches to find relevant predictors.

The first method was based on running a glm on the the baked data but it led to warnings and the algorithm did not converge. I decided to use lasso to only keep the relevant predictors which resulted in a smaller and more concise number of predictors.

```
> set.seed(4557)
> train_x <-  model.matrix(Income ~ . -1, data = data_clean)
> train_y <-  data_clean$Income
> grid = 10^seq(10,-2,by=-.1)
> cv.lasso <-  cv.glmnet(train_x, train_y, family="binomial", alpha=1, lambda=grid)
> plot(cv.lasso)
> best_lambda = cv.lasso$lambda.min
> coef(cv.lasso)
57 x 1 sparse Matrix of class "dgCMatrix"
                                          1
(Intercept)                     -2.62588659
Age                              0.28909723
FNLWGT                           .
EducationYears                   0.62311489
WeeklyHours                      0.27184005
Employer_X.Federal.gov           .
Employer_X.Local.gov             .
Employer_X.Never.worked          .
Employer_X.Private               .
Employer_X.Self.emp.inc          0.31056262
Employer_X.Self.emp.not.inc     -0.02289290
Employer_X.State.gov             .
Employer_X.Without.pay           .
EducationLevel_X.10th            .
EducationLevel_X.11th            .
EducationLevel_X.12th            .
EducationLevel_X.1st.4th         .
EducationLevel_X.5th.6th         .
EducationLevel_X.7th.8th         .
EducationLevel_X.9th             .
EducationLevel_X.Assoc.acdm      .
EducationLevel_X.Assoc.voc       .
EducationLevel_X.Bachelors       .
EducationLevel_X.Doctorate       .
EducationLevel_X.HS.grad         .
EducationLevel_X.Masters         .
EducationLevel_X.Preschool       .
```




```
> relevant_clean = data_clean %>%
+   dplyr::select(Income, Age, EducationYears, WeeklyHours, Employer_X.Self.emp.inc,
+                 Employer_X.Self.emp.not.inc, MaritalStatus_X.Married.civ.spouse,
+                 JobRole_X.Exec.managerial, JobRole_X.Farming.fishing,
+                 JobRole_X.Other.service, JobRole_X.Prof.specialty)
> glimpse(relevant_clean)
Observations: 4,669
Variables: 11
$ Income                              <fct> Bad, Bad, Bad, Bad, Bad, Bad, Bad, Good, Good, Good, Good, Good,
$ Age                                 <dbl> 0.03377929, 0.78036273, -0.04304184, 0.96199337, -0.90720615, -0
$ EducationYears                      <dbl> 1.16806114, 1.16806114, -0.44447542, -1.20170852, 1.16806114, 1.
$ WeeklyHours                         <dbl> -0.09981299, -2.50414436, -0.09981299, -0.09981299, -0.09981299,
$ Employer_X.Self.emp.inc             <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
$ Employer_X.Self.emp.not.inc         <dbl> 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0
$ MaritalStatus_X.Married.civ.spouse  <dbl> 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0
$ JobRole_X.Exec.managerial           <dbl> 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0
$ JobRole_X.Farming.fishing           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0
$ JobRole_X.Other.service             <dbl> 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1
$ JobRole_X.Prof.specialty            <dbl> 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0
>
```



Step 4: Set up a cross-validated model scenario in which there are at least 10 folds to generate in-training predictions. You will be building models on basic accuracy measures.

```
> control <- trainControl( method= "cv",
+                          number = 10,
+                          verboseIter = TRUE,
+                          classProbs = TRUE, savePredictions = TRUE)
```

Step 5: Using the caretStack function, create four base models (LDA, Random Forest, GBM, and SVM Radial) and then, if they all meet the required correlation criteria, ensemble them using a generalized linear model, a Random Forest model, and a XG boosted model. If there are high correlations, substitute the offending model with another of your choosing and rerun with the ensembling algorithm.

```
> set.seed(85)
> models <- caretList(Income~., data=relevant_clean, trControl = control, methodList = algorithmList)
+ Fold01: parameter=none
- Fold01: parameter=none
+ Fold02: parameter=none
- Fold02: parameter=none
+ Fold03: parameter=none
- Fold03: parameter=none
+ Fold04: parameter=none
- Fold04: parameter=none
+ Fold05: parameter=none
- Fold05: parameter=none
+ Fold06: parameter=none
- Fold06: parameter=none
+ Fold07: parameter=none
- Fold07: parameter=none
+ Fold08: parameter=none
- Fold08: parameter=none
+ Fold09: parameter=none
- Fold09: parameter=none
+ Fold10: parameter=none
- Fold10: parameter=none
Aggregating results
Fitting final model on full training set
+ Fold01: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
Iter   TrainDeviance   ValidDeviance   StepSize   Improve
     1        1.0908             nan     0.1000    0.0211
     2        1.0577             nan     0.1000    0.0171
     3        1.0274             nan     0.1000    0.0138
     4        1.0054             nan     0.1000    0.0113
     5        0.9863             nan     0.1000    0.0094
```

```
> #correlation between results
> modelCor(results)
                lda       gbm        rf svmRadial
lda       1.0000000 0.3591851 0.3121134 0.5165051
gbm       0.3591851 1.0000000 0.9080622 0.5452788
rf        0.3121134 0.9080622 1.0000000 0.6926914
svmRadial 0.5165051 0.5452788 0.6926914 1.0000000
> splom(results)
```

Ans: Since the generalized boosted regression model and the random forest model are highly correlated, I decided to take out random forest because it also appears highly correlated to svmRadial, instead I decided to substitute knn in.

```
   150        0.6708             nan     0.1000   -0.0003

- Fold09: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
+ Fold10: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
Iter   TrainDeviance   ValidDeviance   StepSize   Improve
     1        1.0948             nan     0.1000    0.0203
     2        1.0625             nan     0.1000    0.0164
     3        1.0350             nan     0.1000    0.0133
     4        1.0172             nan     0.1000    0.0085
     5        0.9947             nan     0.1000    0.0106
     6        0.9755             nan     0.1000    0.0086
     7        0.9602             nan     0.1000    0.0070
     8        0.9482             nan     0.1000    0.0063
     9        0.9370             nan     0.1000    0.0052
    10        0.9254             nan     0.1000    0.0057
    20        0.8525             nan     0.1000    0.0026
    40        0.7905             nan     0.1000    0.0011
    60        0.7595             nan     0.1000    0.0003
    80        0.7420             nan     0.1000   -0.0000
   100        0.7298             nan     0.1000    0.0003
   120        0.7228             nan     0.1000   -0.0001
   140        0.7178             nan     0.1000   -0.0001
   150        0.7157             nan     0.1000   -0.0002

- Fold10: shrinkage=0.1, interaction.depth=1, n.minobsinnode=10, n.trees=150
+ Fold10: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
Iter   TrainDeviance   ValidDeviance   StepSize   Improve
     1        1.0804             nan     0.1000    0.0266
     2        1.0386             nan     0.1000    0.0212
     3        1.0066             nan     0.1000    0.0161
     4        0.9792             nan     0.1000    0.0132
     5        0.9561             nan     0.1000    0.0107
     6        0.9369             nan     0.1000    0.0086
     7        0.9207             nan     0.1000    0.0074
     8        0.9067             nan     0.1000    0.0064
     9        0.8940             nan     0.1000    0.0061
    10        0.8847             nan     0.1000    0.0040
    20        0.8082             nan     0.1000    0.0020
    40        0.7489             nan     0.1000    0.0010
    60        0.7253             nan     0.1000    0.0006
    80        0.7142             nan     0.1000    0.0000
   100        0.7082             nan     0.1000   -0.0000
   120        0.7025             nan     0.1000   -0.0002
   140        0.6988             nan     0.1000   -0.0002
   150        0.6973             nan     0.1000   -0.0001

- Fold10: shrinkage=0.1, interaction.depth=2, n.minobsinnode=10, n.trees=150
+ Fold10: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
Iter   TrainDeviance   ValidDeviance   StepSize   Improve
     1        1.0784             nan     0.1000    0.0270
     2        1.0351             nan     0.1000    0.0221
     3        0.9991             nan     0.1000    0.0175
     4        0.9699             nan     0.1000    0.0149
     5        0.9467             nan     0.1000    0.0118
     6        0.9266             nan     0.1000    0.0104
     7        0.9091             nan     0.1000    0.0086
     8        0.8940             nan     0.1000    0.0068
     9        0.8795             nan     0.1000    0.0065
    10        0.8657             nan     0.1000    0.0066
    20        0.7850             nan     0.1000    0.0022
```

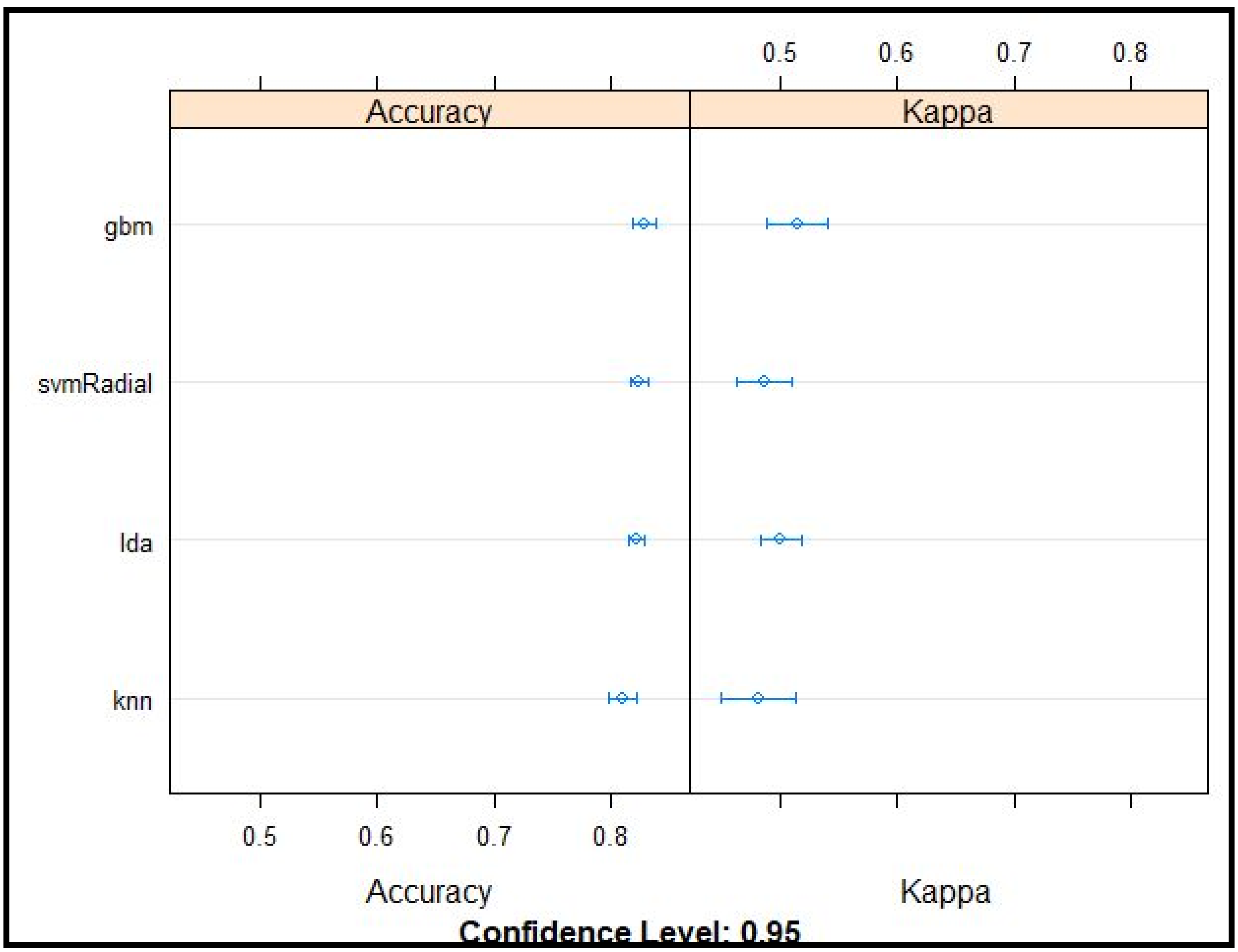
```
+ Fold09: parameter=none
- Fold09: parameter=none
+ Fold10: parameter=none
- Fold10: parameter=none
Aggregating results
Fitting final model on full training set
> print(stack.glm)
A glm ensemble of 2 base models: lda, gbm, knn, svmRadial

Ensemble results:
Generalized Linear Model

4669 samples
   4 predictor
   2 classes: 'Bad', 'Good'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4203, 4202, 4202, 4202, 4202, 4202, ...
Resampling results:

  Accuracy  Kappa
  0.825875  0.504431

>
> #stack using rf
> set.seed(13)
> stack.rf <- caretStack(models, method = "rf", metric = "Accuracy", trControl = control)
+ Fold01: mtry=2
- Fold01: mtry=2
+ Fold01: mtry=3
- Fold01: mtry=3
+ Fold01: mtry=4
- Fold01: mtry=4
+ Fold02: mtry=2
- Fold02: mtry=2
+ Fold02: mtry=3
- Fold02: mtry=3
+ Fold02: mtry=4
- Fold02: mtry=4
+ Fold03: mtry=2
- Fold03: mtry=2
+ Fold03: mtry=3
- Fold03: mtry=3
+ Fold03: mtry=4
- Fold03: mtry=4
+ Fold04: mtry=2
- Fold04: mtry=2
+ Fold04: mtry=3
```

```
   140        0.6833             nan     0.1000   -0.0004
   150        0.6810             nan     0.1000   -0.0001

- Fold10: shrinkage=0.1, interaction.depth=3, n.minobsinnode=10, n.trees=150
Aggregating results
Selecting tuning parameters
Fitting n.trees = 150, interaction.depth = 2, shrinkage = 0.1, n.minobsinnode = 10 on full training set
Iter   TrainDeviance   ValidDeviance   StepSize   Improve
     1        1.0796             nan     0.1000    0.0268
     2        1.0385             nan     0.1000    0.0209
     3        1.0040             nan     0.1000    0.0169
     4        0.9769             nan     0.1000    0.0136
     5        0.9555             nan     0.1000    0.0111
     6        0.9363             nan     0.1000    0.0094
     7        0.9195             nan     0.1000    0.0085
     8        0.9052             nan     0.1000    0.0067
     9        0.8920             nan     0.1000    0.0066
    10        0.8803             nan     0.1000    0.0053
    20        0.8020             nan     0.1000    0.0015
    40        0.7389             nan     0.1000    0.0002
    60        0.7170             nan     0.1000   -0.0001
    80        0.7070             nan     0.1000   -0.0001
   100        0.7009             nan     0.1000   -0.0000
   120        0.6958             nan     0.1000   -0.0001
   140        0.6918             nan     0.1000   -0.0001
   150        0.6907             nan     0.1000   -0.0002

+ Fold01: k=5
- Fold01: k=5
+ Fold01: k=7
- Fold01: k=7
+ Fold01: k=9
- Fold01: k=9
+ Fold02: k=5
- Fold02: k=5
+ Fold02: k=7
- Fold02: k=7
+ Fold02: k=9
- Fold02: k=9
+ Fold03: k=5
- Fold03: k=5
+ Fold03: k=7
- Fold03: k=7
+ Fold03: k=9
- Fold03: k=9
+ Fold04: k=5
- Fold04: k=5
+ Fold04: k=7
- Fold04: k=7
+ Fold04: k=9
- Fold04: k=9
+ Fold05: k=5
- Fold05: k=5
+ Fold05: k=7
- Fold05: k=7
+ Fold05: k=9
- Fold05: k=9
+ Fold06: k=5
- Fold06: k=5
```

```
+ Fold09: eta=0.3, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
- Fold09: eta=0.3, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
+ Fold09: eta=0.3, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.75, nrounds=150
- Fold09: eta=0.3, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.75, nrounds=150
+ Fold09: eta=0.3, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.00, nrounds=150
- Fold09: eta=0.3, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.00, nrounds=150
+ Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.50, nrounds=150
- Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.50, nrounds=150
+ Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.75, nrounds=150
- Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.75, nrounds=150
+ Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=1.00, nrounds=150
- Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=1.00, nrounds=150
+ Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
- Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
+ Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.75, nrounds=150
- Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.75, nrounds=150
+ Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.00, nrounds=150
- Fold09: eta=0.3, max_depth=2, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.00, nrounds=150
+ Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.50, nrounds=150
- Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.50, nrounds=150
+ Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.75, nrounds=150
- Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.75, nrounds=150
+ Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=1.00, nrounds=150
- Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=1.00, nrounds=150
+ Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
- Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
+ Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.75, nrounds=150
- Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.75, nrounds=150
+ Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.00, nrounds=150
- Fold09: eta=0.3, max_depth=3, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.00, nrounds=150
+ Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.50, nrounds=150
- Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.50, nrounds=150
+ Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.75, nrounds=150
- Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.75, nrounds=150
+ Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=1.00, nrounds=150
- Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=1.00, nrounds=150
+ Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
- Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
+ Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.75, nrounds=150
- Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.75, nrounds=150
+ Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.00, nrounds=150
- Fold09: eta=0.4, max_depth=1, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.00, nrounds=150
+ Fold09: eta=0.4, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.50, nrounds=150
- Fold09: eta=0.4, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.50, nrounds=150
+ Fold09: eta=0.4, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.75, nrounds=150
- Fold09: eta=0.4, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=0.75, nrounds=150
+ Fold09: eta=0.4, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=1.00, nrounds=150
- Fold09: eta=0.4, max_depth=2, gamma=0, colsample_bytree=0.6, min_child_weight=1, subsample=1.00, nrounds=150
+ Fold09: eta=0.4, max_depth=2, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
- Fold09: eta=0.4, max_depth=2, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.50, nrounds=150
+ Fold09: eta=0.4, max_depth=2, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=0.75, nrounds=150
```

```
+ Fold10: eta=0.4, max_depth=3, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.
- Fold10: eta=0.4, max_depth=3, gamma=0, colsample_bytree=0.8, min_child_weight=1, subsample=1.
Aggregating results
Selecting tuning parameters
Fitting nrounds = 50, max_depth = 2, eta = 0.3, gamma = 0, colsample_bytree = 0.8, min_child_we
> print(stack.xgboost)
A xgbTree ensemble of 2 base models: lda, gbm, knn, svmRadial

Ensemble results:
eXtreme Gradient Boosting

4669 samples
   4 predictor
   2 classes: 'Bad', 'Good'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4202, 4203, 4201, 4202, 4202, 4202, ...
Resampling results across tuning parameters:

  eta  max_depth  colsample_bytree  subsample  nrounds  Accuracy   Kappa
  0.3  1          0.6               0.50        50      0.8239432  0.5167278
  0.3  1          0.6               0.50       100      0.8235131  0.5189909
  0.3  1          0.6               0.50       150      0.8173032  0.5005943
  0.3  1          0.6               0.75        50      0.8258681  0.5271846
  0.3  1          0.6               0.75       100      0.8232962  0.5190704
  0.3  1          0.6               0.75       150      0.8226520  0.5169235
  0.3  1          0.6               1.00        50      0.8224397  0.5108682
  0.3  1          0.6               1.00       100      0.8230835  0.5135561
  0.3  1          0.6               1.00       150      0.8230835  0.5135408
  0.3  1          0.8               0.50        50      0.8228739  0.5149259
  0.3  1          0.8               0.50       100      0.8228776  0.5113918
  0.3  1          0.8               0.50       150      0.8230885  0.5123081
  0.3  1          0.8               0.75        50      0.8224470  0.5185133
  0.3  1          0.8               0.75       100      0.8252321  0.5239440
  0.3  1          0.8               0.75       150      0.8241624  0.5212674
  0.3  1          0.8               1.00        50      0.8243710  0.5207427
  0.3  1          0.8               1.00       100      0.8243705  0.5208105
  0.3  1          0.8               1.00       150      0.8248002  0.5211967
  0.3  2          0.6               0.50        50      0.8200888  0.5078022
  0.3  2          0.6               0.50       100      0.8132274  0.4860881
  0.3  2          0.6               0.50       150      0.8117298  0.4821297
  0.3  2          0.6               0.75        50      0.8196601  0.5050286
  0.3  2          0.6               0.75       100      0.8179484  0.4987911
  0.3  2          0.6               0.75       150      0.8155870  0.4910256
  0.3  2          0.6               1.00        50      0.8224438  0.5087786
  0.3  2          0.6               1.00       100      0.8215891  0.5070523
```

```
Call:
summary.resamples(object = results)

Models: lda, gbm, knn, svmRadial
Number of resamples: 10

Accuracy
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
lda       0.8068670 0.8147752 0.8224574 0.8218028 0.8281585 0.8369099    0
gbm       0.8072805 0.8219838 0.8256648 0.8288701 0.8339564 0.8522484    0
knn       0.7905983 0.7965739 0.8070668 0.8102368 0.8260171 0.8308351    0
svmRadial 0.8072805 0.8161820 0.8222698 0.8237318 0.8300275 0.8394004    0

Kappa
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
lda       0.4558053 0.4871496 0.5086918 0.5012903 0.5164982 0.5404578    0
gbm       0.4751236 0.4883706 0.5028100 0.5156553 0.5424604 0.5718841    0
knn       0.4288561 0.4429616 0.4710382 0.4818151 0.5245229 0.5483172    0
svmRadial 0.4432629 0.4612176 0.4786557 0.4870337 0.5199210 0.5289236    0

>
```

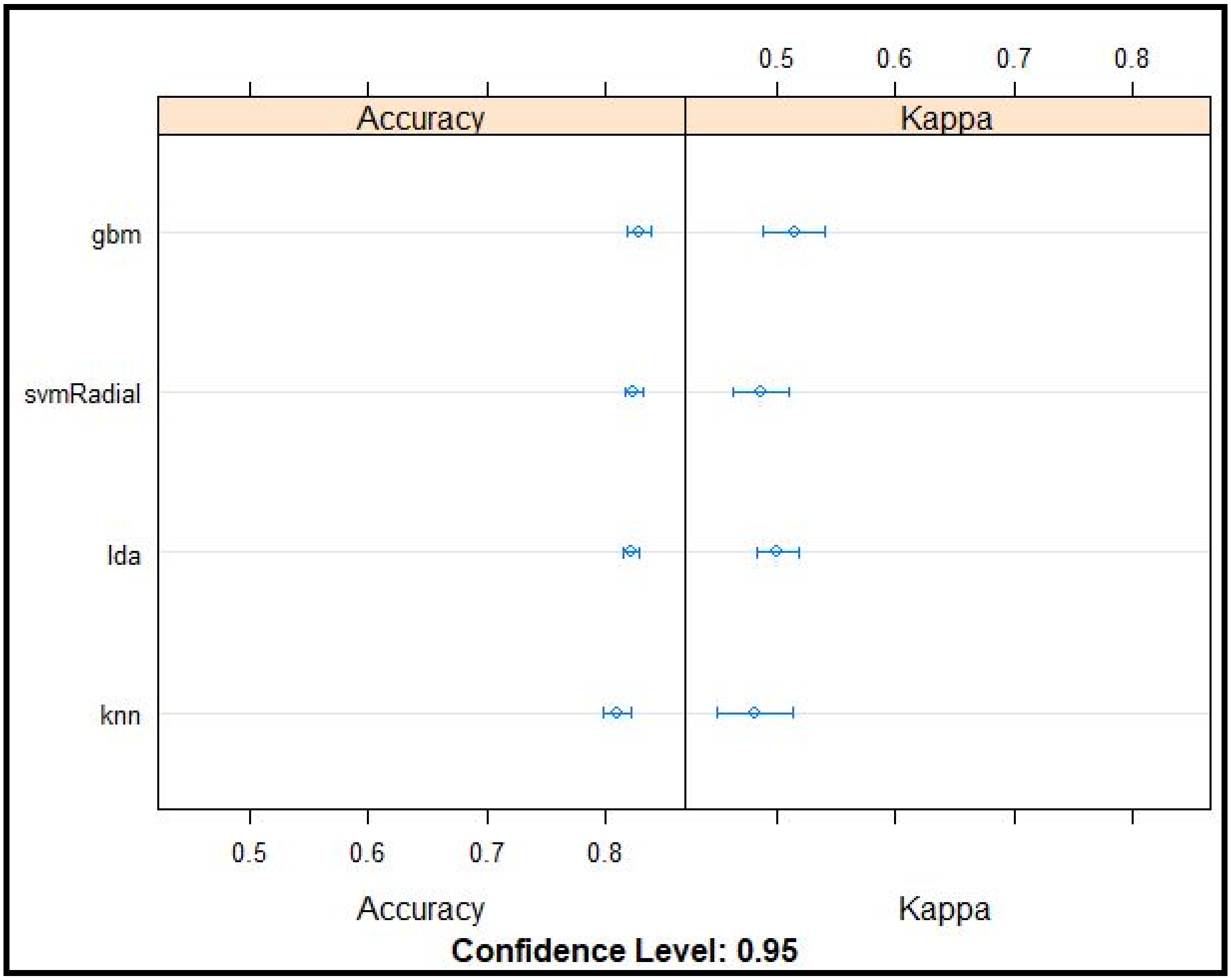


Step 6: Using the caretStack function, create three base models (Random Forest, GBM, and SVM Radial) and add a fourth model from the Caret package set of models [https://topepo.github.io/caret/available-models.html] that we have not worked with. If they all meet the required correlation criteria, ensemble them using a generalized linear model, a Random Forest model, and a XG boosted model. If there are high correlations, substitute the offending model with another of your choosing and rerun with the ensembling algorithm.

Ans:

```
Tuning parameter 'gamma' was held constant at a value of 0
Tuning parameter 'min_child_weight' was held constant at a value of 1
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were nrounds = 50, max_depth = 2, eta = 0.3, gamma = 0, colsample_bytree = 0.8, min_child_weight = 1 and subsample = 0.75.
> algorithmList2 = c('rf', 'gbm', 'ada', 'svmRadial')
> set.seed(85)
> set.seed(84)
> models2 <- caretList(Income~., data=relevant_clean, trControl = control, methodList = algorithmList2)
+ Fold01: mtry= 2
- Fold01: mtry= 2
+ Fold01: mtry= 6
- Fold01: mtry= 6
+ Fold01: mtry=10
- Fold01: mtry=10
+ Fold02: mtry= 2
- Fold02: mtry= 2
+ Fold02: mtry= 6
- Fold02: mtry= 6
+ Fold02: mtry=10
- Fold02: mtry=10
+ Fold03: mtry= 2
- Fold03: mtry= 2
+ Fold03: mtry= 6
- Fold03: mtry= 6
+ Fold03: mtry=10
- Fold03: mtry=10
+ Fold04: mtry= 2
- Fold04: mtry= 2
```

```
1 package is needed for this model and is not installed. (ada). Wou
1: yes
2: no

Selection: 1
Installing package into 'C:/Users/khattakm/Documents/R/win-library/
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/ada_2.
Content type 'application/zip' length 734616 bytes (717 KB)
downloaded 717 KB

package 'ada' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\khattakm\AppData\Local\Temp\RtmpgB2B6d\downloaded_
+ Fold01: nu=0.1, maxdepth=1, iter=150
- Fold01: nu=0.1, maxdepth=1, iter=150
+ Fold01: nu=0.1, maxdepth=2, iter=150
- Fold01: nu=0.1, maxdepth=2, iter=150
+ Fold01: nu=0.1, maxdepth=3, iter=150
- Fold01: nu=0.1, maxdepth=3, iter=150
+ Fold02: nu=0.1, maxdepth=1, iter=150
- Fold02: nu=0.1, maxdepth=1, iter=150
+ Fold02: nu=0.1, maxdepth=2, iter=150
- Fold02: nu=0.1, maxdepth=2, iter=150
+ Fold02: nu=0.1, maxdepth=3, iter=150
- Fold02: nu=0.1, maxdepth=3, iter=150
+ Fold03: nu=0.1, maxdepth=1, iter=150
- Fold03: nu=0.1, maxdepth=1, iter=150
+ Fold03: nu=0.1, maxdepth=2, iter=150
- Fold03: nu=0.1, maxdepth=2, iter=150
+ Fold03: nu=0.1, maxdepth=3, iter=150
- Fold03: nu=0.1, maxdepth=3, iter=150
+ Fold04: nu=0.1, maxdepth=1, iter=150
- Fold04: nu=0.1, maxdepth=1, iter=150
+ Fold04: nu=0.1, maxdepth=2, iter=150
- Fold04: nu=0.1, maxdepth=2, iter=150
+ Fold04: nu=0.1, maxdepth=3, iter=150
- Fold04: nu=0.1, maxdepth=3, iter=150
+ Fold05: nu=0.1, maxdepth=1, iter=150
- Fold05: nu=0.1, maxdepth=1, iter=150
+ Fold05: nu=0.1, maxdepth=2, iter=150
- Fold05: nu=0.1, maxdepth=2, iter=150
```

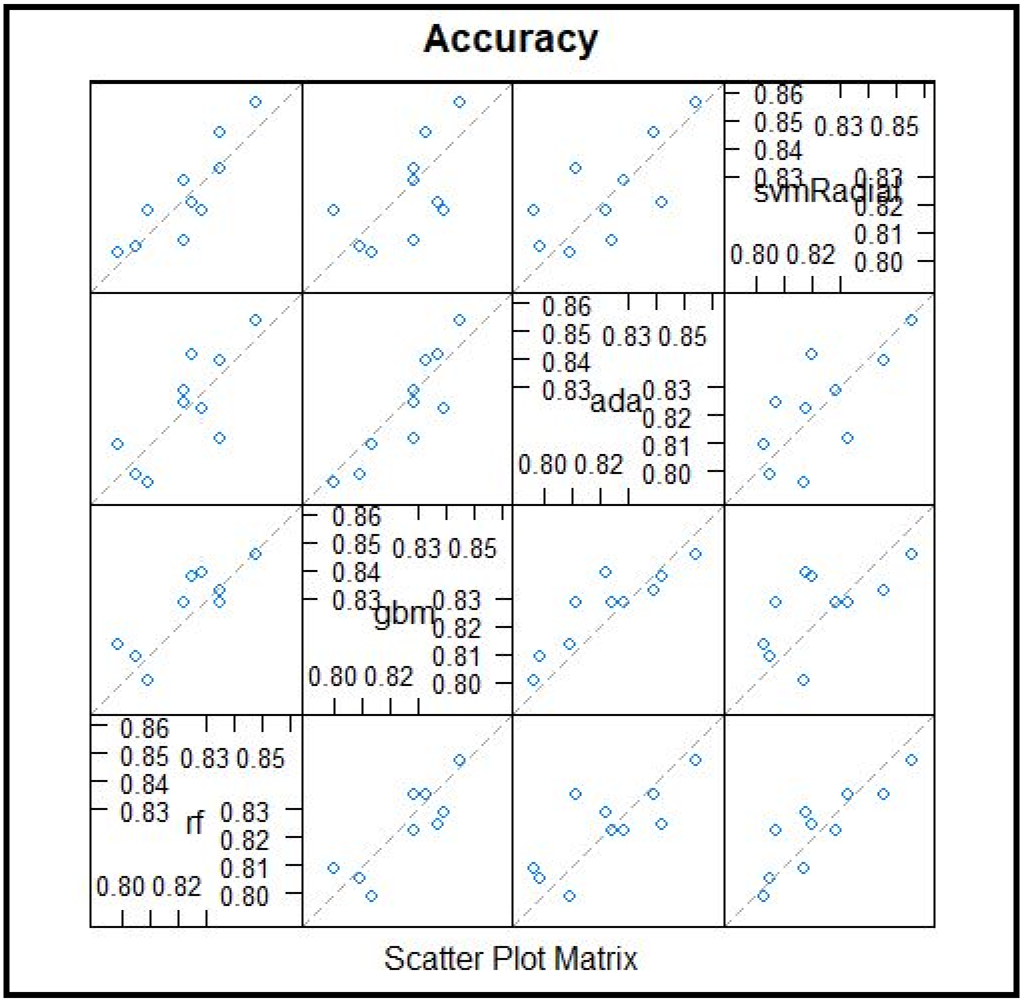


Accuracy
Kappa
gbm
svmRadial
lda
knn
0.5
0.6
0.7
0.8
Confidence Level: 0.95

```
> results2 <- resamples(models2)
> summary(results2)

Call:
summary.resamples(object = results2)

Models: rf, gbm, ada, svmRadial
Number of resamples: 10

Accuracy
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.7987152 0.8123271 0.8235281 0.8228762 0.8335118 0.8476395    0
gbm       0.8004292 0.8174518 0.8286938 0.8265114 0.8364492 0.8454936    0
ada       0.7961373 0.8099572 0.8233405 0.8226570 0.8367238 0.8540773    0
svmRadial 0.8029979 0.8098595 0.8192500 0.8235232 0.8319058 0.8562232    0

Kappa
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.4003497 0.4532921 0.4778753 0.4799264 0.5067266 0.5419862    0
gbm       0.4491726 0.4895947 0.5214991 0.5179131 0.5473238 0.5638503    0
ada       0.4306695 0.4593081 0.4965504 0.4974224 0.5327791 0.5813339    0
svmRadial 0.4239206 0.4560958 0.4798188 0.4874998 0.5193666 0.5758918    0

> |
```

```
> dotplot(results2)
> #correlation between results
> modelCor(results2)
                 rf       gbm       ada svmRadial
rf        1.0000000 0.8475198 0.7719794 0.8693456
gbm       0.8475198 1.0000000 0.8891874 0.6133984
ada       0.7719794 0.8891874 1.0000000 0.7023672
svmRadial 0.8693456 0.6133984 0.7023672 1.0000000
> |
```

There is a high correlation between ada and all other models which shows that maybe I should’ve chosen a different model instead of ada.



Step 7: Create four more ensembles (two-to-four base models + one ensembling algorithm) of your choosing and identify the combination with the highest accuracy.

Model 1:

```
#for step 7
#MODEL 1

algorithmList3 = c('rf', 'svmRadial')

set.seed(100)
models3 <- caretList(Income~., data=relevant_clean, trControl = control, methodList = algorithmList2)
results3 <- resamples(models3)
summary(results3)
dotplot(results3)

modelCor(results3)
splom(results3)

set.seed(101)
stack.glm3 <- caretStack(models3, method = "glm", metric = "Accuracy", trControl = control)
print(stack.glm3)
```

```
Call:
summary.resamples(object = results3)

Models: rf, gbm, ada, svmRadial
Number of resamples: 10

Accuracy
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.7918455 0.8137045 0.8235326 0.8213708 0.8307444 0.8394004    0
gbm       0.7961373 0.8202246 0.8276231 0.8280112 0.8387778 0.8543897    0
ada       0.8047210 0.8204693 0.8244111 0.8254394 0.8324411 0.8479657    0
svmRadial 0.7939914 0.8116604 0.8297645 0.8237291 0.8367238 0.8436831    0

Kappa
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.4009648 0.4372220 0.4839503 0.4761352 0.5118744 0.5510479    0
gbm       0.4501851 0.4893568 0.5132920 0.5196406 0.5536881 0.5941414    0
ada       0.4455057 0.4741389 0.5002472 0.5002906 0.5182042 0.5699425    0
svmRadial 0.4016692 0.4295874 0.4997378 0.4869620 0.5428883 0.5578282    0

> dotplot(results3)
>
> modelCor(results3)
                 rf       gbm       ada svmRadial
rf        1.0000000 0.6780299 0.5959730 0.9383420
gbm       0.6780299 1.0000000 0.9071064 0.7792846
ada       0.5959730 0.9071064 1.0000000 0.7020913
svmRadial 0.9383420 0.7792846 0.7020913 1.0000000
```

```
> print(stack.glm3)
A glm ensemble of 2 base models: rf, gbm, ada, svmRadial

Ensemble results:
Generalized Linear Model

4669 samples
   4 predictor
   2 classes: 'Bad', 'Good'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4203, 4202, 4202, 4202, 4201, 4203, ...
Resampling results:

  Accuracy   Kappa
  0.8226547  0.499778
```

Model 2:

```
#MODEL2

algorithmList4 = c('glm', 'C5.0')
set.seed(1000)
models4 <- caretList(Income~., data=relevant_clean, trControl = control, methodList = algorithmList2)
results3 <- resamples(models4)
summary(results4)
dotplot(results4)
modelCor(results4)
splom(results4)
set.seed(1010)
stack.glm4 <- caretStack(models4, method = "rf", metric = "Accuracy", trControl = control)
print(stack.glm4)
```

```
Call:
summary.resamples(object = results4)

Models: rf, gbm, ada, svmRadial
Number of resamples: 10

Accuracy
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.8004292 0.8120985 0.8192500 0.8222292 0.8294922 0.8522484    0
gbm       0.7961373 0.8126338 0.8306628 0.8265087 0.8426982 0.8501071    0
ada       0.7961373 0.8104925 0.8256602 0.8252271 0.8426124 0.8476395    0
svmRadial 0.7982833 0.8190578 0.8242227 0.8224406 0.8246925 0.8394004    0

Kappa
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.3962217 0.4596409 0.4783168 0.4826594 0.5035814 0.5612038    0
gbm       0.4060748 0.4754361 0.5199940 0.5111809 0.5547497 0.5747366    0
ada       0.4023867 0.4798671 0.5085335 0.5048712 0.5453569 0.5655703    0
svmRadial 0.3916787 0.4575001 0.4995455 0.4834610 0.5070490 0.5289236    0

> dotplot(results4)
> modelCor(results4)
                 rf       gbm       ada svmRadial
rf        1.0000000 0.7873612 0.7960455 0.7295619
gbm       0.7873612 1.0000000 0.9612309 0.7356018
ada       0.7960455 0.9612309 1.0000000 0.6997356
svmRadial 0.7295619 0.7356018 0.6997356 1.0000000
> splom(results4)
> set.seed(1010)
> stack.glm4 <- caretStack(models4, method = "rf", metric = "Accuracy", trControl = control)
+ Fold01: mtry=2
- Fold01: mtry=2
```

```
> print(stack.glm4)
A rf ensemble of 2 base models: rf, gbm, ada, svmRadial

Ensemble results:
Random Forest

4669 samples
   4 predictor
   2 classes: 'Bad', 'Good'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4202, 4202, 4203, 4202, 4202, 4202, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
  2     0.8113132  0.4831354
  3     0.8125966  0.4894883
  4     0.8089559  0.4788433

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 3.
>
```

Model 3:

```
#MODEL3

algorithmList5 = c('glm', 'rf')
set.seed(10000)
models5 <- caretList(Income~., data=relevant_clean, trControl = control, methodList = algorithmList2)
results5 <- resamples(models5)
summary(results5)
dotplot(results5)
modelCor(results5)
splom(results5)
set.seed(10100)
stack.glm5 <- caretStack(models5, method = "ada", metric = "Accuracy", trControl = control)
print(stack.glm5)
```

```
Call:
summary.resamples(object = results5)

Models: rf, gbm, ada, svmRadial
Number of resamples: 10

Accuracy
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.7901499 0.8143382 0.8199355 0.8213819 0.8281585 0.8497854    0
gbm       0.7987152 0.8190578 0.8252957 0.8275986 0.8340471 0.8626609    0
ada       0.7837259 0.8160490 0.8233405 0.8248121 0.8260171 0.8648069    0
svmRadial 0.8008565 0.8166763 0.8231475 0.8256668 0.8297645 0.8605150    0

Kappa
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.3900957 0.4467610 0.4723099 0.4775027 0.5015955 0.5690202    0
gbm       0.4356568 0.4851124 0.5087831 0.5119299 0.5192757 0.6175822    0
ada       0.3771278 0.4757006 0.4946289 0.5019305 0.5137940 0.6246644    0
svmRadial 0.4158653 0.4709903 0.4822414 0.4926886 0.4981677 0.5985846    0

> dotplot(results5)
```

```
> dotplot(results5)
> modelCor(results5)
                 rf       gbm       ada svmRadial
rf        1.0000000 0.8362034 0.9569057 0.8983271
gbm       0.8362034 1.0000000 0.9293900 0.9269925
ada       0.9569057 0.9293900 1.0000000 0.9585115
svmRadial 0.8983271 0.9269925 0.9585115 1.0000000
> splom(results5)
```

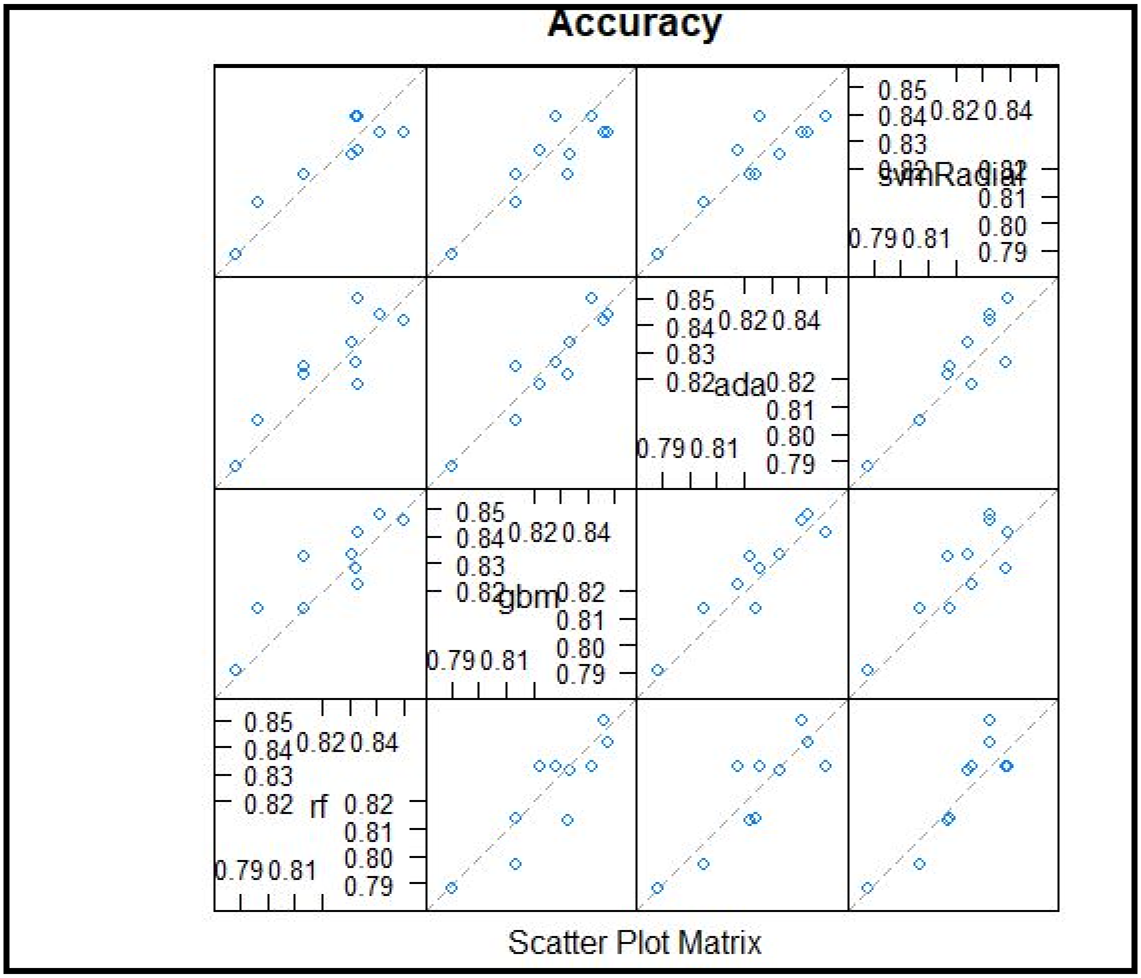
```
> print(stack.glm5)
A C5.0 ensemble of 2 base models: rf, gbm, ada, svmRadial

Ensemble results:
C5.0

4669 samples
   4 predictor
   2 classes: 'Bad', 'Good'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4202, 4202, 4201, 4203, 4202, 4202, ...
Resampling results across tuning parameters:

  model  winnow  trials  Accuracy   Kappa
  rules  FALSE    1      0.8220201  0.5313587
  rules  FALSE   10      0.8226630  0.5285391
  rules  FALSE   20      0.8226630  0.5285391
  rules   TRUE    1      0.8213777  0.5298400
  rules   TRUE   10      0.8203066  0.5245851
  rules   TRUE   20      0.8209490  0.5341815
  tree   FALSE    1      0.8220201  0.5313587
  tree   FALSE   10      0.8215923  0.5211304
  tree   FALSE   20      0.8215923  0.5211304
  tree    TRUE    1      0.8213777  0.5298400
  tree    TRUE   10      0.8200925  0.5361551
  tree    TRUE   20      0.8200925  0.5361551

Accuracy was used to select the optimal model using the largest value.
The final values used for the model were trials = 10, model = rules and winnow = FALSE.
```

Model 4:

```
#MODEL4

algorithmList6 = c('rf', 'C5.0')
set.seed(100000)
models6 <- caretList(Income~., data=relevant_clean, trControl = control, methodList = algorithmList2)
results6 <- resamples(models6)
summary(results6)
dotplot(results6)
modelCor(results6)
splom(results6)
set.seed(101000)
stack.glm6 <- caretStack(models6, method = "glm", metric = "Accuracy", trControl = control)
print(stack.glm6)
```

```
> results6 <- resamples(models6)
> summary(results6)

Call:
summary.resamples(object = results6)

Models: rf, gbm, ada, svmRadial
Number of resamples: 10

Accuracy
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.7880086 0.8134047 0.8319073 0.8233008 0.8329764 0.8501071    0
gbm       0.7901499 0.8158458 0.8304721 0.8269438 0.8394897 0.8479657    0
ada       0.7880086 0.8189625 0.8252957 0.8252280 0.8394897 0.8501071    0
svmRadial 0.7880086 0.8176942 0.8256694 0.8226621 0.8329764 0.8394004    0

Kappa
               Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
rf        0.3743250 0.4434261 0.5039261 0.4811814 0.5180967 0.5696079    0
gbm       0.3937957 0.4828950 0.5220727 0.5151957 0.5588312 0.5776834    0
ada       0.3857467 0.4784849 0.4996440 0.5017455 0.5485128 0.5772553    0
svmRadial 0.3857467 0.4634266 0.4905903 0.4858638 0.5254250 0.5374711    0

> dotplot(results6)
> modelCor(results6)
                 rf       gbm       ada svmRadial
rf        1.0000000 0.8784261 0.8718781 0.9050463
gbm       0.8784261 1.0000000 0.9324294 0.8657507
ada       0.8718781 0.9324294 1.0000000 0.8986147
svmRadial 0.9050463 0.8657507 0.8986147 1.0000000
```

```
> print(stack.glm6)
A glm ensemble of 2 base models: rf, gbm, ada, svmRadial

Ensemble results:
Generalized Linear Model

4669 samples
   4 predictor
   2 classes: 'Bad', 'Good'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 4202, 4202, 4202, 4203, 4202, 4202, ...
Resampling results:

  Accuracy   Kappa
  0.8243781  0.5065891
```



Step 8: Discuss why you think the highest accuracy was achieved with those specific ensembles.

Ans: The highest accuracy was achieved by the ensemble of model 3. Model 3 uses generalized linear model and additive logistic regression in the stack and runs it in the generalized linear model ensemble which achieved the highest accuracy of 86.48%. The second closest value was achieved by gradient boosted model which was also at 86%. This shows that it might be hard to assume if one model is better than another without running it through the dataset. I think different models have different strengths and as I become more experienced in R, it will be easy to guess what model to use with a particular dataset.

Furthermore, it would make a better model if we already knew what stack models that can be used to gain maximum accuracy and then use the most appropriate ensemble model.