HarvardX PH125.9 Independent Capstone Project: Can We Predict Happiness?

Oscar Mak

March 2022

1. Introduction

The study of happiness has been in the popular zeitgeist recently. On Jan 23, 2022 CNN published an article with the headline "Two years into the pandemic, Yale's 'happiness' course is more popular than ever". More than 3.7 million people have enrolled in the free online course referenced by CNN in this article. The course appears to be well-liked, earning a rating of 4.9 stars (out of 5) from 31,743 ratings as of March 3, 2022.

More recently, the Wall Street Journal published "Harvard Wants M.B.A.s to Learn How to Be Happy at Work" on Feb 14, 2022. The article describes an oversubscribed course in the Harvard MBA program called "Leadership and Happiness" taught by Arthur Brooks. Professor Brooks also writes a popular series of articles in The Atlantic titled "How to Build a Life".

Motivated by my experience taking the course mentioned by CNN and by reading some of Professor Brooks's writing, I performed an analysis of General Social Survey (GSS) data to see if self-rating of happiness can be predicted using other responses to the survey.

Survey respondents rated themselves as either very happy, pretty happy, or not too happy. A naïve model predicting the modal (most frequent) response correctly predicted the correct outcome with 56.0% accuracy. A classification model using the XGBoost package improved the accuracy only slightly to 63.0%. Satisfaction with present financial situation, whether someone found life exciting, routine, or dull, and marital status were the three most important features in predicting the happiness outcome.

2. Methods and analysis

2.1. About the data

The GSS is a survey of Americans' well-being and attitudes conducted annually by the National Opinion Research Center (NORC) at the University of Chicago since 1972.

The GSS includes a set of core demographic, behavioral, and attitudinal questions that are asked every year (the replicating core). In addition, topics of special interest are added from time to time. Past examples of such special interest topics include national spending priorities, intergroup tolerance, and attitudes toward morality.

2.1.1. Loading the data The complete GSS dataset is available to the public in STATA format as a Zip file. The following code downloads the data, unzips it, and loads it into an R dataframe.

```
# Download and unzip General Social Survery (GSS) data file
dl <- tempfile()
download.file("https://gss.norc.org/documents/stata/GSS_stata.zip", dl)
```

```
unzip(dl)
gss_data <- read_dta("gss7221_r1b.dta")
```

2.1.2. Exploring and cleaning the data A quick look at the data reveals that it is a pretty big dataset.

```
str(gss_data, list.len = 3, width = 80, strict.width="cut")
```

```
## tibble [68,846 x 6,309] (S3: tbl_df/tbl/data.frame)
##  $ year           : num [1:68846] 1972 1972 1972 1972 1972 ...
##   ..- attr(*, "label")= chr "gss year for this respondent"
##   ..- attr(*, "format.stata")= chr "%8.0g"
##  $ id             : num [1:68846] 1 2 3 4 5 6 7 8 9 10 ...
##   ..- attr(*, "label")= chr "respondent id number"
##   ..- attr(*, "format.stata")= chr "%8.0g"
##  $ wrkstat        : dbl+lbl [1:68846] 1, 5, 2, 1, 7, 1, 1, 1, 2, 1, 7, 1, 1, 1..
##    ..@ label       : chr "labor force status"
##    ..@ format.stata: chr "%8.0g"
##    ..@ labels      : Named num [1:8] 1 2 3 4 5 6 7 8
##    .. ..- attr(*, "names")= chr [1:8] "working full time" "working part time""..
##   [list output truncated]
```

The dataset contains 68,846 observations of 6,309 variables. Each variable is additionally tagged with metadata. For example "wrkstat" is labeled as "labor force status" and a value of 1 here indicates that the respondent is working full time.

2.1.2.1. Filter for replicating core 6,309 is a very large number of variables so we begin filtering out unhelpful variables. One such filter is to only include variables that correspond to the replicating core of questions that are asked every year. Variables not included in the replicating core are not asked every year and will therefore contain many NAs.

To identify which variables are in the replicating core, we download a PDF titled "Repeated Items in the General Social Survey" and extract the PDF text using the following code.

```
rep_core_pdf <- tempfile()
download.file("https://gss.norc.org/Documents/other/Replicating%20Core.pdf",
              rep_core_pdf)
rep_core_text <- pdf_text(rep_core_pdf)     # Extract pdf text
```

Reading the PDF reveals that all replicating core variable names are included in pages 2-12 so we use the following code to remove the unnecessary pages.

```
rep_core_text <- rep_core_text[2:12]
```

The variable names are three or more all caps text characters, followed by zero to three digits (for example, RACE and FAMILY16). We can therefore use regex to extract a list of variable names that we want and then keep only columns for those variables with this code.

```
# Extracts variables names for questions in replicating core into one long list
pattern <- "[A-Z]{3,}[0-9]{0,3}"
rep_core_codes <- str_extract_all(rep_core_text, pattern) %>%
```

```r
  unlist() %>%
  tolower()     # variables names are lowercase in gss_data
rep_core_codes <- rep_core_codes[rep_core_codes != "gss"]  # GSS is survey name

# Keep only columns in the replicating core
gss_data <- gss_data[, colnames(gss_data) %in% rep_core_codes]
```

After filtering for the replicating core we are left with 484 variables.

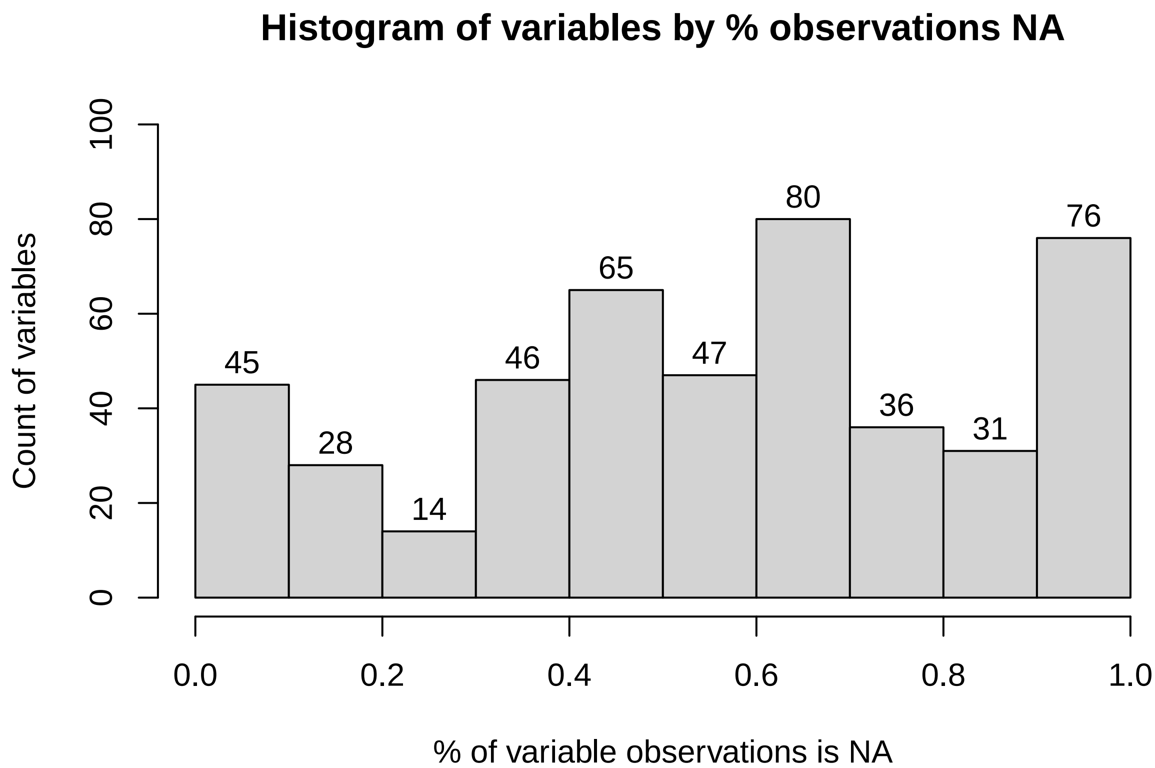
2.1.2.2. Filter for near zero variance The following code removes variables with near zero variance. Removing such variables is desirable because they have little predictive power and can cause some models to become unstable or crash.

```r
nzv <- nearZeroVar(gss_data)
gss_data <- gss_data[, -nzv]
```

468 variables remain after filtering for variables with near zero variance.

2.1.2.3. Filter for excess NAs We currently have a 68,846 by 468 data frame. If every respondent answered every question then we would have a grid of 32,219,928 data points. We can use the is.na function to see that there are 17,990,906 NAs. A quick calculation finds that this is 56% of the grid, so we have a lot of NAs. We can plot the number of variables by percentage of observations that are NA with the code below.

```r
colMeans(is.na(gss_data)) %>%
  hist(main = "Histogram of variables by % observations NA",
       xlab = "% of variable observations is NA",
       ylab = "Count of variables",
       ylim = c(0, 100),
       labels = TRUE)
```



The plot confirms that NAs are not evenly distributed. For example, 76 variables have observations that are 90-100% NAs (almost all NA) while 45 variables have 0-10% NAs. The following code filters out variables that are >40% NA.

```
low_NAs <- colMeans(is.na(gss_data)) <= 0.4
gss_data <- gss_data[,low_NAs]
```

2.1.2.4. Manual inspection of remaining variables We are left with 133 variables after filtering for NAs. This is a manageable number to inspect manually for inclusion or exclusion. The following code creates a table of variables, the label assigned to each variable in the metadata, and the % NA for each variable.

```
n <- length(gss_data)
data_labels <- vector(length = n)      # create an empty vector
for(i in 1:n){
  data_labels[i] <- attributes(gss_data[[i]])$label
  } # fill vector with label metadata

# Create table with data for each variable
names_labels <- data.frame(
  index = seq.int(length(data_labels)),
  label = data_labels,
  pct_NA = percent(colMeans(is.na(gss_data)), accuracy = 0.1)
  )
names_labels     # Display table of variables with labels
```

```
##          index                                      label pct_NA
## year         1              gss year for this respondent   0.0%
## wrkstat      2                        labor force status   5.9%
## wrkslf       3            r self-emp or works for somebody  11.4%
## occ10        4           r's census occupation code (2010)  12.8%
## indus10      5             r's industry code (naics 2007)  12.8%
## marital      6                            marital status   0.1%
## divorce      7            ever been divorced or separated  38.4%
## pawrkslf     8   father self-emp. or worked for somebody  24.0%
## paocc10      9    father's census occupation code (2010)  24.6%
## paind10     10       father's industry code (naics 2007)  26.3%
## sibs        11             number of brothers and sisters   8.3%
## childs      12                        number of children   6.1%
## age         13                         age of respondent   0.8%
## educ        14           highest year of school completed   0.4%
## paeduc      15     highest year school completed, father  33.4%
## maeduc      16     highest year school completed, mother  21.8%
## degree      17                        r's highest degree   0.3%
## padeg       18                   father's highest degree  29.2%
## madeg       19                    mothers highest degree  17.8%
## sex         20                           respondents sex   0.1%
## race        21                        race of respondent   5.9%
## res16       22    type of place lived in when 16 yrs old   8.2%
## reg16       23                 region of residence, age 16   5.9%
## mobile16    24           geographic mobility since age 16   8.8%
## family16    25       living with parents when 16 yrs old   8.1%
## incom16     26        r's family income when 16 yrs old  19.9%
## born        27                was r born in this country  13.6%
## parborn     28    were r's parents born in this country  19.4%
## granborn    29  how many grandparents born outside u.s.  18.7%
## hompop      30              number of persons in household   5.9%
## babies      31     household members less than 6 yrs old   6.4%
## preteen     32       household members 6 thru 12 yrs old   6.4%
## teens       33      household members 13 thru 17 yrs old   6.3%
## adults      34        household members 18 yrs and older   6.0%
## unrelat     35          number in household not related  31.5%
## earnrs      36          how many in family earned money   6.7%
## income      37                       total family income  12.7%
## region      38                       region of interview   5.9%
## xnorcsiz    39                   expanded norc size code   5.9%
## srcbelt     40                              src beltcode   5.9%
## size        41                    size of place in 1000s   5.9%
## partyid     42               political party affiliation   0.7%
## polviews    43 think of self as liberal or conservative  13.9%
## natroad     44                      highways and bridges  28.7%
## natsoc      45                           social security  29.2%
## natmass     46                       mass transportation  31.9%
## natpark     47                      parks and recreation  28.3%
## spkath      48           allow anti-religionist to speak  38.7%
## colath      49           allow anti-religionist to teach  38.3%
## libath      50      allow anti-religious book in library  39.6%
## spkcom      51                  allow communist to speak  39.8%
## cappun      52 favor or oppose death penalty for murder  16.5%
## gunlaw      53               favor or oppose gun permits  33.2%
```

```
## courts      54                   courts dealing with criminals  20.7%
## relig       55                      r's religious preference   0.5%
## fund        56               how fundamentalist is r currently   4.1%
## attend      57          how often r attends religious services   1.0%
## reliten     58                         strength of affiliation  13.2%
## postlife    59                     belief in life after death  38.2%
## relig16     60                        religion in which raised  10.9%
## fund16      61              how fundamentalist was r at age 16   8.3%
## raclive     62                  any opp. race in neighborhood  12.6%
## happy       63                              general happiness   6.9%
## health      64                            condition of health  25.0%
## life        65                      is life exciting or dull  39.1%
## confinan    66       confid in banks & financial institutions  37.7%
## conbus      67                  confidence in major companies  34.7%
## conclerg    68               confidence in organized religion  34.5%
## coneduc     69                        confidence in education  33.1%
## confed      70             confid. in exec branch of fed govt  34.1%
## conlabor    71                  confidence in organized labor  35.9%
## conpress    72                            confidence in press  33.6%
## conmedic    73                         confidence in medicine  33.0%
## contv       74                       confidence in television  33.2%
## conjudge    75         confid. in united states supreme court  34.9%
## consci      76             confidence in scientific community  36.9%
## conlegis    77                         confidence in congress  34.1%
## conarmy     78                         confidence in military  34.0%
## satjob      79                               work satisfaction  32.2%
## class       80                subjective class identification   5.1%
## satfin      81          satisfaction with financial situation   6.8%
## finalter    82                  change in financial situation   6.9%
## finrela     83                       opinion of family income   7.3%
## union       84               does r or spouse belong to union  31.4%
## abdefect    85                strong chance of serious defect  33.8%
## abnomore    86                married--wants no more children  33.9%
## abhlth      87            woman's health seriously endangered  33.5%
## abpoor      88          low income--cant afford more children  34.0%
## abrape      89                     pregnant as result of rape  34.1%
## absingle    90                                    not married  34.0%
## chldidel    91                       ideal number of children  38.0%
## xmarsex     92               sex with person other than spouse  39.3%
## pornlaw     93                feelings about pornography laws  39.3%
## xmovie      94                seen x-rated movie in last year  39.6%
## fear        95        afraid to walk at night in neighborhood  36.9%
## owngun      96                               have gun in home  36.7%
## pistol      97                     pistol or revolver in home  38.7%
## shotgun     98                                shotgun in home  38.7%
## rifle       99                                  rifle in home  38.7%
## news       100                how often does r read newspaper  36.8%
## tvhours    101                    hours per day watching tv  39.5%
## phone      102                        does r have telephone   8.4%
## coop       103                 r's attitude toward interview   8.9%
## comprend   104                 r's understanding of questions   6.7%
## form       105             form of split questionnaire asked   6.7%
## realinc    106                  family income in constant $  10.2%
## coninc     107             family income in constant dollars  10.2%
```

```
## ethnic     108                       country of family origin  25.9%
## eth1       109                 1st mentioned country of origin  35.6%
## ethnum     110              type of response about ethnicity:r   6.5%
## dwelling   111                               type of structure  24.6%
## gender1    112                              gender of 1st person  13.8%
## old1       113                                 age of 1st person  15.2%
## mar1       114                      marital status of 1st person  14.5%
## hefinfo    115                          number of hef informant  22.0%
## hhrace     116                                race of household  15.9%
## respnum    117                             number in family of r  14.7%
## hhtype     118                                    household type  14.0%
## hhtype1    119                        household type (condensed)  14.0%
## famgen     120            number of family generations in household  13.6%
## rplace     121                 r's relationship to household head  15.2%
## dateintv   122                                 date of interview  12.7%
## isco88     123 respondent's occupation, 1980 census & 1988 isco code  11.5%
## paisco88   124 r's father's occupation, 1980 census & 1988 isco code  24.6%
## cohort     125                                     year of birth   0.3%
## zodiac     126                     respondents astrological sign  11.1%
## ballot     127                          ballot used for interview  31.8%
## issp       128                              filter for issp cases  35.0%
## sampcode   129                                sampling error code   8.2%
## sample     130                          sampling frame and method   5.9%
## wtssall    131                                    weight variable   5.9%
## vstrat     132                                   variance stratum   6.7%
## vpsu       133                      variance primary sampling unit   6.7%
```

We see that many of the variables can be removed for the following reasons:
1. *Redundant* – occ10 (respondent's occupation code) and indus10 (respondent's industry code) are very similar. realinc (family income in constant $) and coninc (family income in constant dollars) are nearly identical. For each such pair of similar variables, only the one with lower % NA is kept.
2. *Interview logistics* – variables regarding interview logistics such as region (geographic region of interview) and size (population of interview location in 1000s) were omitted. Where an interview took place should not have much bearing on a respondent's happiness.
3. *Political and social beliefs* – variables concerning specific political and social issues were omitted. Examples include natmass (does the country spend enough on mass transportation) and conlegis (degree of confidence in Congress).
4. *Parent info* – variables concerning demographic information on the respondent's parents were omitted. Examples include paind10 (father's industry code) and madeg (mother's highest degree).

The following code drops the unwanted variables.

```
# Create vector of column numbers for variables to keep
keep_index <- c(1:4, 6, 7, 11:14, 17, 20:29, 37, 42, 43, 55:65, 79:83, 94:96,
                100, 107, 108, 111, 116, 117, 119:121, 123, 126)
gss_data <- gss_data[, keep_index]  # Drop unwanted variables
```

We are left with 54 variables for our model as listed below.

```
names_labels <- names_labels[keep_index,]
names_labels
```

```
##          index                                                 label pct_NA
```

```
## year          1                          gss year for this respondent   0.0%
## wrkstat       2                                   labor force status   5.9%
## wrkslf        3                    r self-emp or works for somebody  11.4%
## occ10         4                   r's census occupation code (2010)  12.8%
## marital       6                                       marital status   0.1%
## divorce       7                     ever been divorced or separated  38.4%
## sibs         11                       number of brothers and sisters   8.3%
## childs       12                                   number of children   6.1%
## age          13                                    age of respondent   0.8%
## educ         14                    highest year of school completed   0.4%
## degree       17                                   r's highest degree   0.3%
## sex          20                                      respondents sex   0.1%
## race         21                                   race of respondent   5.9%
## res16        22            type of place lived in when 16 yrs old   8.2%
## reg16        23                          region of residence, age 16   5.9%
## mobile16     24                    geographic mobility since age 16   8.8%
## family16     25                 living with parents when 16 yrs old   8.1%
## incom16      26                   r's family income when 16 yrs old  19.9%
## born         27                           was r born in this country  13.6%
## parborn      28             were r's parents born in this country  19.4%
## granborn     29           how many grandparents born outside u.s.  18.7%
## income       37                                  total family income  12.7%
## partyid      42                         political party affiliation   0.7%
## polviews     43            think of self as liberal or conservative  13.9%
## relig        55                            r's religious preference   0.5%
## fund         56                   how fundamentalist is r currently   4.1%
## attend       57            how often r attends religious services   1.0%
## reliten      58                             strength of affiliation  13.2%
## postlife     59                          belief in life after death  38.2%
## relig16      60                             religion in which raised  10.9%
## fund16       61                  how fundamentalist was r at age 16   8.3%
## raclive      62                       any opp. race in neighborhood  12.6%
## happy        63                                   general happiness   6.9%
## health       64                                 condition of health  25.0%
## life         65                             is life exciting or dull  39.1%
## satjob       79                                    work satisfaction  32.2%
## class        80                      subjective class identification   5.1%
## satfin       81              satisfaction with financial situation   6.8%
## finalter     82                       change in financial situation   6.9%
## finrela      83                             opinion of family income   7.3%
## xmovie       94                      seen x-rated movie in last year  39.6%
## fear         95           afraid to walk at night in neighborhood  36.9%
## owngun       96                                     have gun in home  36.7%
## news        100                      how often does r read newspaper  36.8%
## coninc      107                   family income in constant dollars  10.2%
## ethnic      108                             country of family origin  25.9%
## dwelling    111                                    type of structure  24.6%
## hhrace      116                                    race of household  15.9%
## respnum     117                                number in family of r  14.7%
## hhtype1     119                          household type (condensed)  14.0%
## famgen      120         number of family generations in household  13.6%
## rplace      121                    r's relationship to household head  15.2%
## isco88      123 respondent's occupation, 1980 census & 1988 isco code  11.5%
## zodiac      126                         respondents astrological sign  11.1%
```

2.1.2.5. Explore years variable A quick look at the years variable shows that we have observations in most years from 1972 to 2021.

```
unique(gss_data$year)
```

```
##  [1] 1972 1973 1974 1975 1976 1977 1978 1980 1982 1983 1984 1985 1986 1987 1988
## [16] 1989 1990 1991 1993 1994 1996 1998 2000 2002 2004 2006 2008 2010 2012 2014
## [31] 2016 2018 2021
```

Since 2021 was an unusual year for happiness due to the COVID-19 pandemic, we remove observations from that year using the following code.

```
gss_data <- gss_data %>% filter(year != 2021)
```

2.1.2.6. Explore happy variable Since the happy variable is the outcome that we want to predict, we ought to take a closer look at it. First, we use the is.na function to find that we have 4,760 NA observations in happy. The following code drops observations with NA in the happy variable.

```
gss_data <- gss_data %>% drop_na(happy)
```

The following code provides additional information on the happy variable.

```
table(gss_data$happy)
```

```
##
##     1     2     3
## 18823 33563  7668
```

```
attributes(gss_data$happy)
```

```
## $label
## [1] "general happiness"
##
## $format.stata
## [1] "%8.0g"
##
## $labels
##    very happy  pretty happy not too happy
##             1             2             3
##
## $class
## [1] "haven_labelled" "vctrs_vctr"     "double"
```

We see that the values are categorical with 1 = very happy, 2 = pretty happy, and 3 = not too happy. We also see that 2 (pretty happy) is the most common response.

2.2. Preparing data for XGBoost

XGBoost requires the features for prediction to be formatted as a matrix. It also requires categorical outcome values to be formatted as integers starting from zero. The following code extracts the happy variable (our outcome) and adjusts the values such that 0 = very happy, 1 = pretty happy, and 2 = not too happy. It additionally puts the features in matrix format as required.

```
outcomes <- as.integer(gss_data$happy) - 1  # Extract happy, convert to integer from zero
gss_data$happy <- NULL                      # Remove happy column from gss_data
features <- as.matrix(gss_data)
```

2.3. Creating validation data set and splitting development data into train and test

We carve out 10% of the data as a holdout validation data set that will be used to evaluate our final model. The remaining data will be used to develop and tune the model. This development data is further divided in a 90/10 split where 90% of the development data is used to train the model and 10% is used to test and tune the model. This partitioning of the data is accomplished with the following code.

```
# Carve out 10 percent of data as validation set
set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(y = outcomes, times = 1, p = 0.1, list = FALSE)
dev_features <- features[-test_index,]
dev_outcomes <- outcomes[-test_index]
validation_features <- features[test_index,]
validation_outcomes <- outcomes[test_index]

# Split development data 90 pct into train_set and 10 pct into test_set
set.seed(1, sample.kind="Rounding")
test_index <- createDataPartition(y = dev_outcomes, times = 1, p = 0.1, list = FALSE)
train_features <- dev_features[-test_index,]
train_outcomes <- dev_outcomes[-test_index]
test_features <- dev_features[test_index,]
test_outcomes <- dev_outcomes[test_index]
```

2.4. Model approach

A naïve model predicting the most common happy response is used as a baseline. XGBoost is used to try to improve upon this naive baseline.

2.4.1. Naive prediction using mode The simplest model for our categorical happy outcome would be to predict the modal (most frequent) response in all cases. We saw in section 2.1.2.6 that “pretty happy” is the most frequest response. The following code predicts “pretty happy” in all cases yields an accuracy of 56.0% when applied to the test data set.

```
most_freq <- names(sort(-table(train_outcomes)))[1]
most_freq
```

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

```
mean(test_outcomes == most_freq)
```

```
## [1] 0.5597484
```

2.4.2. XGBoost classification using trees Can XGBoost classification using trees beat the 56.0% accuracy achieved by naively predicting the mode? We convert our train and test data into XGBoost matrix objects and fit the XGBoost model using default values for tuning parameters eta, nrounds, and max.depth.

```
# Convert train and test sets into XGBoost matrix objects
xgboost_train <- xgb.DMatrix(data = train_features, label = train_outcomes)
xgboost_test <- xgb.DMatrix(data = test_features, label = test_outcomes)

# Train the model using defaults
num_class <- length(levels(as.factor(train_outcomes)))
fit <- xgb.train(data = xgboost_train,
                 objective = "multi:softprob",    # output class probabilities
                 booster = "gbtree",              # use tree for classification problems
                 eval_metric = "mlogloss",
                 watchlist = list(val_1 = xgboost_train),
                 eta = 0.3,                       # tuning parameter default = 0.3
                 nrounds = 100,                   # tuning parameter default = 100
                 max.depth = 6,                   # tuning parameter default = 6
                 num_class = num_class,
                 early_stopping_rounds = 10,
                 verbose = FALSE)                 # turns off progress reports

# Make predictions on test features. Output is probability by class
pred <- as_tibble(predict(fit, test_features, reshape = TRUE))
colnames(pred) <- levels(as.factor(train_outcomes))

# Predict the class with the highest probability
pred$prediction <- apply(pred, 1, function(x) colnames(pred)[which.max(x)])
head(pred)   # Display first few rows
```

```
## # A tibble: 6 x 4
##     `0`   `1`    `2` prediction
##   <dbl> <dbl>  <dbl> <chr>
## 1 0.256 0.599 0.145  1
## 2 0.691 0.293 0.0156 0
## 3 0.673 0.281 0.0459 0
## 4 0.196 0.634 0.170  1
## 5 0.278 0.468 0.254  1
## 6 0.498 0.459 0.0438 0
```

```
# Calculate accuracy of predictions
mean(pred$prediction == test_outcomes)
```

```
## [1] 0.6167222
```

When applied to our test data set, XGBoost (with default parameters) yields an accuracy of 61.7%, which is a modest improvement over the naïve model accuracy of 56.0%. Can we do better by tuning the XGBoost model?

2.4.2.1. Tuning eta in XGBoost We use the following code to tune the eta parameter in our XGBoost model.

```
# This takes a while to run so turned off evaluation
# Delete eval=FALSE to run, but be prepared to wait!
etas <- seq(0, 1, 0.1)
accuracies <- sapply(etas, function(n){
  fit <- xgb.train(data = xgboost_train,
                   objective = "multi:softmax",   # output class probabilities
                   booster = "gbtree",            # use tree for classification problems
                   eval_metric = "error",
                   eta = n,                       # parameter being tuned
                   nrounds = 100,                 # tuning parameter default = 100
                   max.depth = 6,                 # tuning parameter default = 6
                   num_class = num_class,
                   verbose = FALSE)               # turns off progress reports
  pred <- as_tibble(predict(fit, test_features, reshape = TRUE))
  return(mean(pred == test_outcomes))
})
best_eta <- etas[which.max(accuracies)]
best_eta
max(accuracies)
```

The best value for eta is found to be 0.1. This further improves our model accuracy from 61.7% (using default parameters) to 62.5% when applied to our test data set.

2.4.2.2. Tuning nrounds in XGBoost We use the tuned value of eta and the following code to tune the nrounds parameter in our XGBoost model.

```
# This takes a while to run so turned off evaluation
# Delete eval=FALSE to run, but be prepared to wait!
nrounds <- seq(100, 200, 10)
accuracies <- sapply(nrounds, function(n){
  fit <- xgb.train(data = xgboost_train,
                 objective = "multi:softmax",   # output class probabilities
                 booster = "gbtree",            # use tree for classification problems
                 eval_metric = "error",
                 eta = best_eta,                # use prior tuning result
                 nrounds = n,                   # parameter being tuned
                 max.depth = 6,                 # tuning parameter default = 6
                 num_class = num_class,
                 verbose = FALSE)               # turns off progress reports
  pred <- as_tibble(predict(fit, test_features, reshape = TRUE))
  return(mean(pred == test_outcomes))
})
best_nrounds <- nrounds[which.max(accuracies)]
best_nrounds
max(accuracies)
```

The best value for nrounds is found to be 150. This further improves our model accuracy from 62.5% with only eta tuned to 62.7% with both eta and nrounds tuned (when applied to our test data set).

2.4.2.3. Tuning max.depth in XGBoost We use the tuned values of eta and nrounds and the following code to tune the max.depth parameter in our XGBoost model.

```
# This takes a while to run so turned off evaluation
# Delete eval=FALSE to run, but be prepared to wait!
depths <- seq(1, 10, 1)
accuracies <- sapply(depths, function(n){
  fit <- xgb.train(data = xgboost_train,
                   objective = "multi:softmax",   # output class probabilities
                   booster = "gbtree",            # use tree for classification problems
                   eval_metric = "error",
                   eta = best_eta,                # use prior tuning result
                   nrounds = best_nrounds,        # use prior tuning result
                   max.depth = n,                 # parameter being tuned
                   num_class = num_class,
                   verbose = FALSE)               # turns off progress reports
  pred <- as_tibble(predict(fit, test_features, reshape = TRUE))
  return(mean(pred == test_outcomes))
})
best_depth <- depths[which.max(accuracies)]
best_depth
max(accuracies)
```

The best value for max.depth is found to be 6, which is the same as the default value. The accuracy of the model when applied to the test data set is unchanged at 62.7%.

3. Results

The tuned XGBoost model yields an accuracy of 62.7% when applied to the test data set, a modest improvement over the naïve model accuracy of 56.0%. We use the following code to apply the tuned model to the validation data set.

```
# Fit the model using tuned parameters
# Turned off evaluation since prior tuning results are needed and those take
# a very long time to run. Delete eval=FALSE here an in prior tuning to run.
fit <- xgb.train(data = xgboost_train,
                 objective = "multi:softmax",
                 booster = "gbtree",
                 eval_metric = "error",
                 max.depth = best_depth,
                 eta = best_eta,
                 nrounds = best_nrounds,
                 num_class = num_class,
                 verbose = FALSE)

# Apply to validation set
pred <- as_tibble(predict(fit, validation_features, reshape = TRUE))
final_accuracy <- mean(pred == validation_outcomes)
final_accuracy   # 63.0% when applied to validation set
```

We achieve a final accuracy of 63.0%. Additionally, we can use the variable importance feature of XGBoost to see which features are the most important predictors of the happy outcome.

```r
# See note in prior code chunk about eval=FALSE
importance <- xgb.importance(model = fit)
head(importance, 5)
```

The five most important features for predicting the happy outcome are described as follows:
1. *satfin* - satisfaction with present financial situation
2. *life* - is life exciting, pretty routine, or dull
3. *marital* - current marital status
4. *satjob* - satisfaction with the work you do
5. *health* - is own health excellent, good, fair, or poor

4. Conclusion

It turns out that happiness is difficult to predict! We did a lot of data cleaning to trim the GSS dataset from 6,309 variables to 54 variables. A naïve model that simply predicted the most frequent response in all cases resulted in 56.0% accuracy when applied to our test data set. A tuned XGBoost classification model improved that accuracy to 62.7% when applied to the test data set. This tuned model resulted in 63.0% accuracy when applied to the holdout validation data set.

The variable importance function of XGBoost revealed that the most important variables for predicting happiness were:
1. Satisfaction with present financial situation
2. Is life exciting, routine, or dull
3. Current marital status
4. Satisfaction with work
5. State of health

Another look at the topic of happiness is possible by examining other happiness survey data sets. In one well-known example, Princeton economists Angus Deaton and Daniel Kahnemann analyzed Gallup-Healthways (now Gallup-Sharecare) survey data. This data set included two distinct measures of happiness. The first was emotional well-being, or day-to-day joy, sadness, anger, etc. The second was life evaluation, which measures the respondent's overall satisfaction with their life on a continuous 0-10 scale. The more nuanced measures of happiness, plus the fact that life evaluation is a continuous variable rather than categorical, may allow different analyses to be conducted. For example, Deaton and Kahnemann famously concluded that additional income above $75,000/year increases the life evaluation measure of happiness but not the emotional well-being measure of happiness. This data set is only available to subscribers or full-time students on campus and therefore was not easily accessible for this analysis.