Technical Report

Group 2: Sky Peterson, Emma Thoron, Calvin Beeman-Weber, and Jakob Shimer

12/16/2021

Abstract

A very brief description of the topic you are investigating, the research question of interest, your methods of analysis, and the general conclusion of your study.

We are investing how couples meet in the modern age and what makes them stay together. With the advent of the digital era, the methods by which individuals develop new, or strengthen existing, interpersonal relationships has changed drastically, from the way that people meet and get to know potential romantic partners to marriages. Thus, our research question is: what variables tend to lead to marriage relationships? Through the Logistic Regression Model, LASSO, LDA, the Tree Model, and Random Forests, we will investigate the relationships between predictors to conclude what contributes to marriages. Based on this analysis, our study concludes that PPT18OV, partnerAge, ppage, ppincimp, and age_when_met tend to be the most important predictors when looking at people who are married.

Introduction

An overview of the topic and relevant background information, a discussion of existing theories and models, a description of how your investigation differs from prior ones, and a precise statement of your research question.

For many people, marriage is the ultimate goal of dating and relationships. In recent years, the dating scene has drastically changed. The role of social media and dating apps for instance, has allowed unprecedented access to a pool of eligible suitors. Events such as financial crisis, health crisis, and climate change have altered the political and economic landscape in ways that change people's goals and definitions for companionship. Our definitions of marriage are evolving. Recent historic cases such as Obergefell v. Hodges expanded marriage rights giving previously discriminated against people a chance to marry. Legally and socially marriage is a cornerstone of our country. In our research we want to look into factors of marriage.

In order to study the aspects of modern relationships that relate to marriage, we are using a data set from the Stanford University Libraries called "How Couples Meet and Stay Together 2017 fresh sample." The funding for HCMST 2017 comes from Stanford's United Parcel Service Endowment. The data is the latest collection of their "How Couples Meet and Stay Together" (HCMST) series taken over a series of a few years. The latest edition to the data from 2017 has new questions about people's use of dating apps including Tinder and Grindr. This project focuses on what aspects of relationships are associated with marriage. Previous scholarship using the HCMST 2017 data that we use in our analysis has focused on understanding the increase in online dating in recent years. For example, a study done on the data found that between 1995 and 2017, there was a 37% increase in couples meeting online. Our focus is slightly different than this dating app focused research. While understanding how couples meet will be taken into consideration by our analysis, we primarily want to look at attributes that contribute to marriage. In short, we want to look at the types of

questions that people ask and the types of attributes that lead to marriages. Our research question is, what variables tend to lead to marriage relationships?

Methods

A description of the data sets used, a discussion of where the data came from and how it was obtained, a summary of the data itself (including the number of observations and variables, and what each observational unit represents), an explanation of data processing implemented to prepare the data for analysis.

When we first settled on the topic of dating, marriage, and relationships for this project, we initially selected the the OkCupid dataset released in 2016 by researchers Emil Kirkegaard and Julius Daugbjerg Bjerrekær. However, after studying how this data was obtained through scraping data from OkCupid without users permission, as well as the highly publicized backlash for Kirkegaard and Bjerrekærs research, we decided that this data was not ethically sourced and thus shouldn't be used for our research.

When then decided to look for data that covered the topics relating to marriage that we were interested but that were ethically sourced. Enter the "How Couples Meet and Stay Together 2017 fresh sample" from Stanford University Libraries. As mentioned in the introduction, this project utilizes data from "How Couples Meet and Stay Together 2017 fresh sample" which we have dubbed HCMST 2017. Unlike the OkCupid data, this data was collected through survey and the people represented knew they were going to have their information recorded. The researchers at Stanford University contracted the survey company GfK to perform the online survey to collect the data. Gfk recruits subjects by phone and by Address Based Sampling. They give subjects without Internet access at home access to the Internet to help make the sample more representative. The data set is nationally representative. We fell confident that this data was collected in both a viable and ethical manner.

When we loaded the raw HCMST 2017 we had 3510 observations and 285 variables. Each observational unit of the dataset represents a (as in one person's) survey response. While this data set is amazing in many ways, it did contain a significant number of NAs. In fact there was no column that didn't contain at least one NA. Working through the NA issue in our wrangling section was an important step in our process and is touched on in depth in the following paragraphs. Throughout our data wrangling section below, we combine certain columns that can be and remove columns with to high a proportion of NAs which decreases our variables to 110.

Load the Data & Wrangling

Before even loading the dataset, we created two functions for use throughout the wrangling process. The first one called 'nMaxCor' can take a model matrix and extract the n variables with the largest correlations. The second, called modMatMerge, can convert a model matrix back to a data frame. These were critical functions in dealing with the imputation and wrangling of our data. The code for this can be found in the Methods section of our Code Appendix.

In the beginning out data wrangling is fairly standard. We load our data from its package. We factorize the data frame. Then rename several of its factors from names like 'w6_q4' to names like 'partnerGender' and 'sameCollege', so that it's easier to understand what they're referring to down the line. We found that when a column name started with 'w6' it contained preprocessed data from the survey.

```
HCMST <- read_dta("HCMST 2017 fresh sample for public sharing draft v1.1.dta")
```

Also in this section, work is done to combine certain variables. For example, there are three different questions in the survey referring to relationship status that we combined into one: one which separated people into married and unmarried groups, one which further differentiated the unmarried group into people in sexual relationships, non-sexual relationships, and no relationship, and one which split the people not in a relationship into people who had been in a relationship before and people who had never been in a relationship before. We

combined these into one factor with levels 'married', 'sexual relationship', 'unsexual relationship', 'previous relationships', 'no relationships'. Besides making our data set smaller, and thus slightly more wieldy, the question that was only asked to the unmarried people output NAs for every married person so combining it with that question reduced the number of NAs in our data set. Going through this combining process for the rest of the nested questions in the data set reduced the NAs further.

The final bit of data wrangling done in this section was a function we ran to remove any columns with more than one third NAs and any rows with more than one fifth NAs, the idea being that any columns left with that many NAs might not have enough data points to run imputation well and that too many un-answered survey questions from a given respondent calls their respect for the survey into question and weakens our trust in the rest of their responses. These changes are exported on a data frame called HCMST and a matrix called HCMSTmat.

Imputation for NA

Imputation is done here using the impute.knn function from the impute package from Biocunductor.org. The modMatMerge function is used to convert the output of the imputation from a matrix to a data frame. The dummy variables "Not Married" and "Married" that were created in the imputation process are combined into one variable called "S1", which will be our response variable.

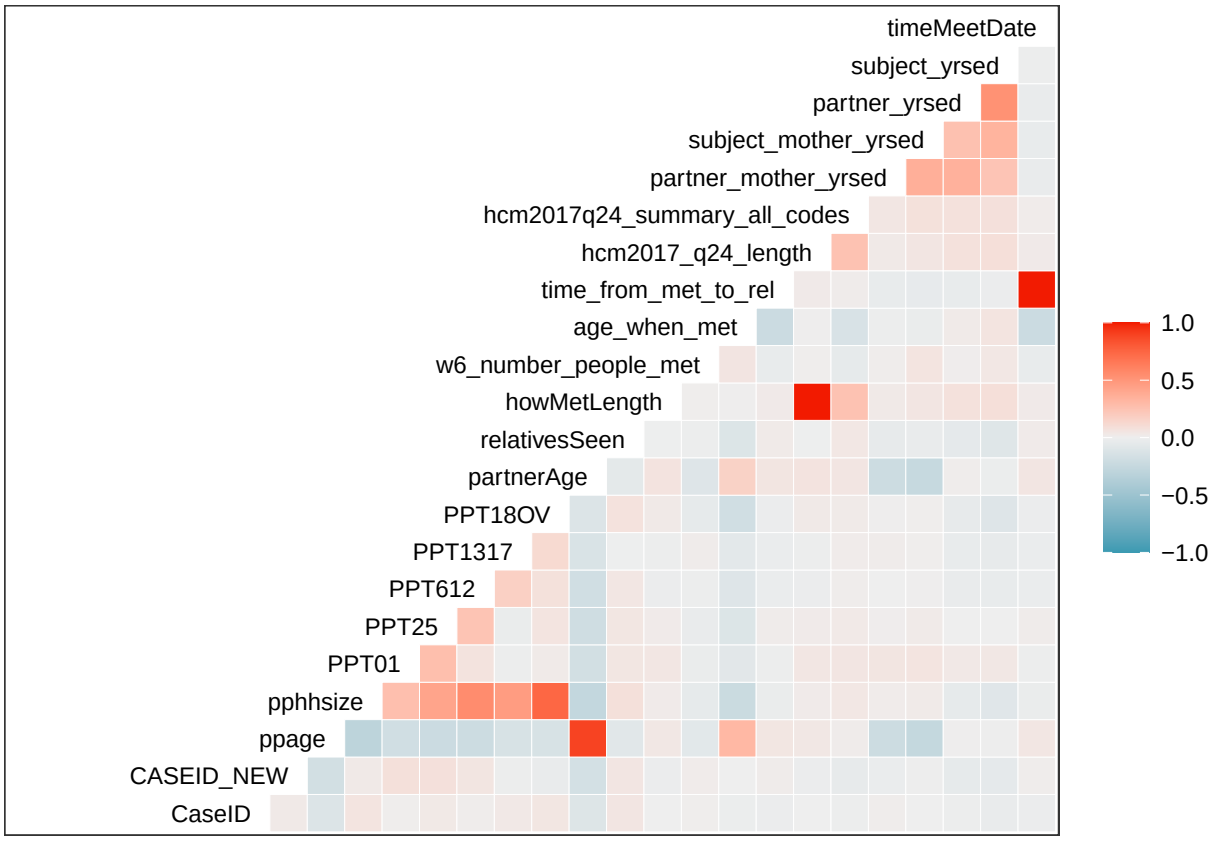
```
Imputed <- impute.knn(HCMSTmat, k = 30, maxp = 4000, rng.seed=12)

iHCMST <- modMatMerge(as.data.frame(Imputed$data), colnames(HCMST))
for(k in colnames(iHCMST)){
  levels(iHCMST[[k]]) <- str_replace_all(levels(iHCMST[[k]]), "\\.", " ")
}
```

Exploratory Data Analysis

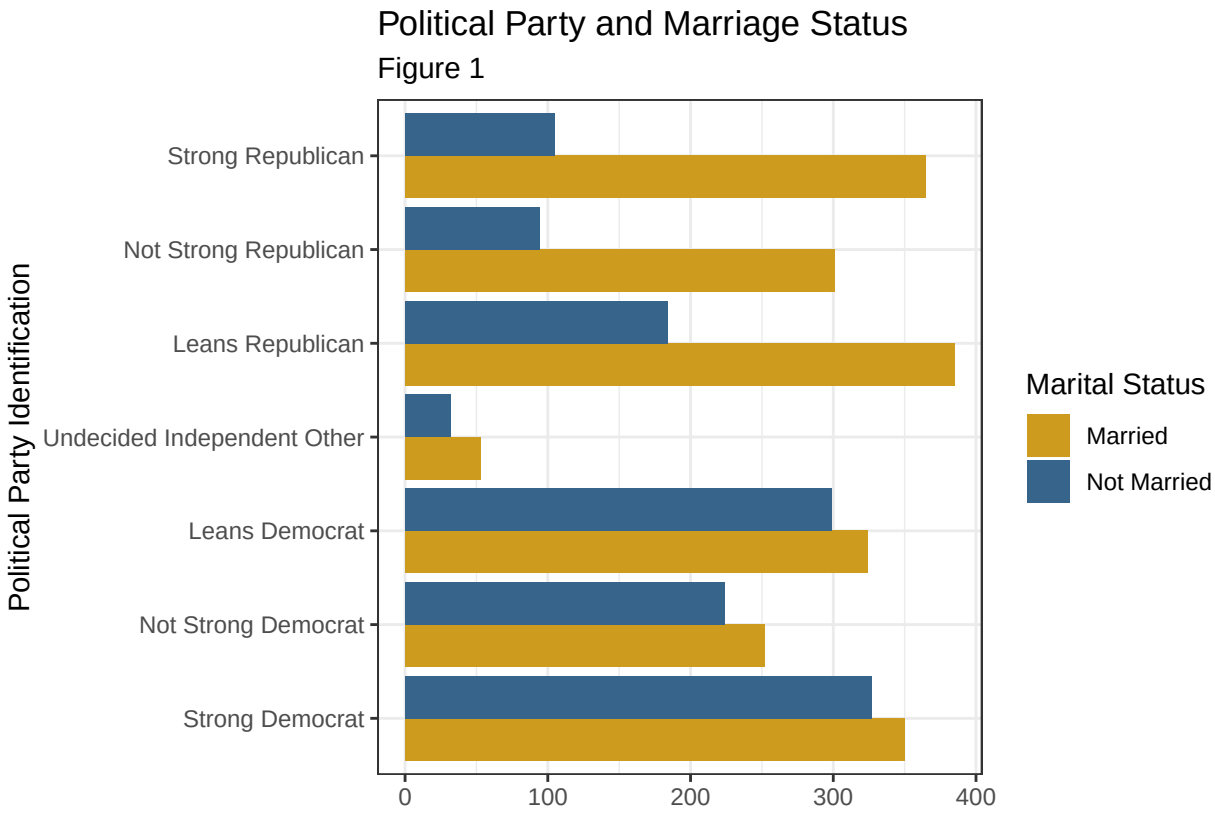
A presentation of graphical and numerical summaries of the data (along with a discussion of their relevance to modeling assumptions and further analysis), a description of the statistical methods used to analyze your data, and diagnostics of the appropriateness of any models or inference procedures you will apply in the Results section.

Summaries of the Data



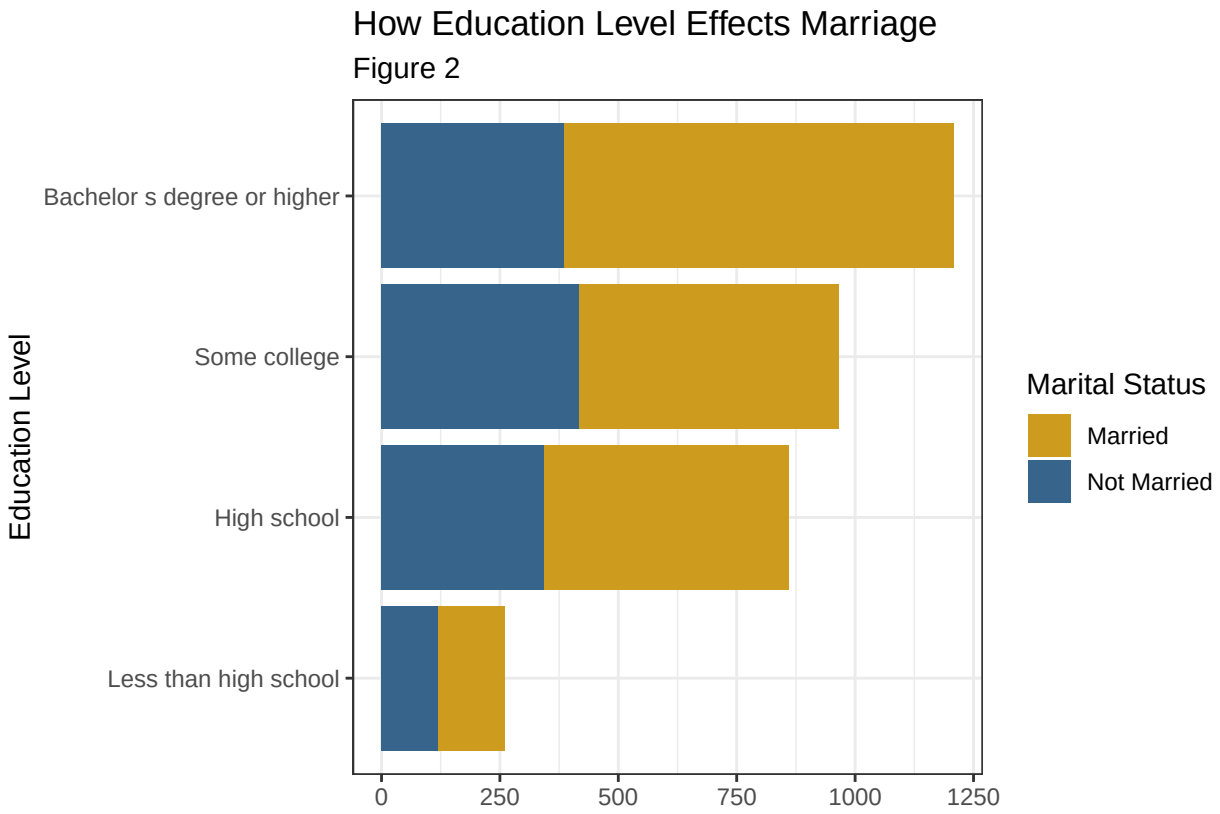
In the first element of our exploratory data analysis, we are going to look at the correlations for the remaining qualitative predictors through the ggcorr() function. From this correlation analysis, we find that there are a few highly correlated predictors. The variables with the highly correlated predictors close to or at one includes hcm2017_q24_length & howMetLength, timeMeetDate & time_from_met_to_rel, and partnerAge & ppage. It makes sense that these predictors are quite close together. For example hcm2017_q24_length & howMetLength are basically the same variable due to both overlap of questioning and our data wrangling. In the case of the partnerAge & ppage predictors—which stand for the partners age and the age of the respondent respectivley—the high correlation is probably due to the fact that the majority of people pick parners around the same age as themselves.

Since the majority of our predictors are categorical, it is critical for us to look at some of these predictors graphically since we are unable to do to numerically. We use ggplot + the geom_bar function to create the basic geometry of our visualizations, with a slew of other functions being used to improve readability in ways you can see in the graphs below. Based on these graphs below we can make several conclusions about predictors correlations to marriage. First, we can conclude that within our data, Republicans are more likely to be married than Democrats. Not only were there more people in the dataset who attended college, but they were also more likely to be married than those with other levels of education. While the number of respondents who reported each Relationship Quality Ranking are vastly different, the proportions of married to not married for each level of the predictor is about the same.



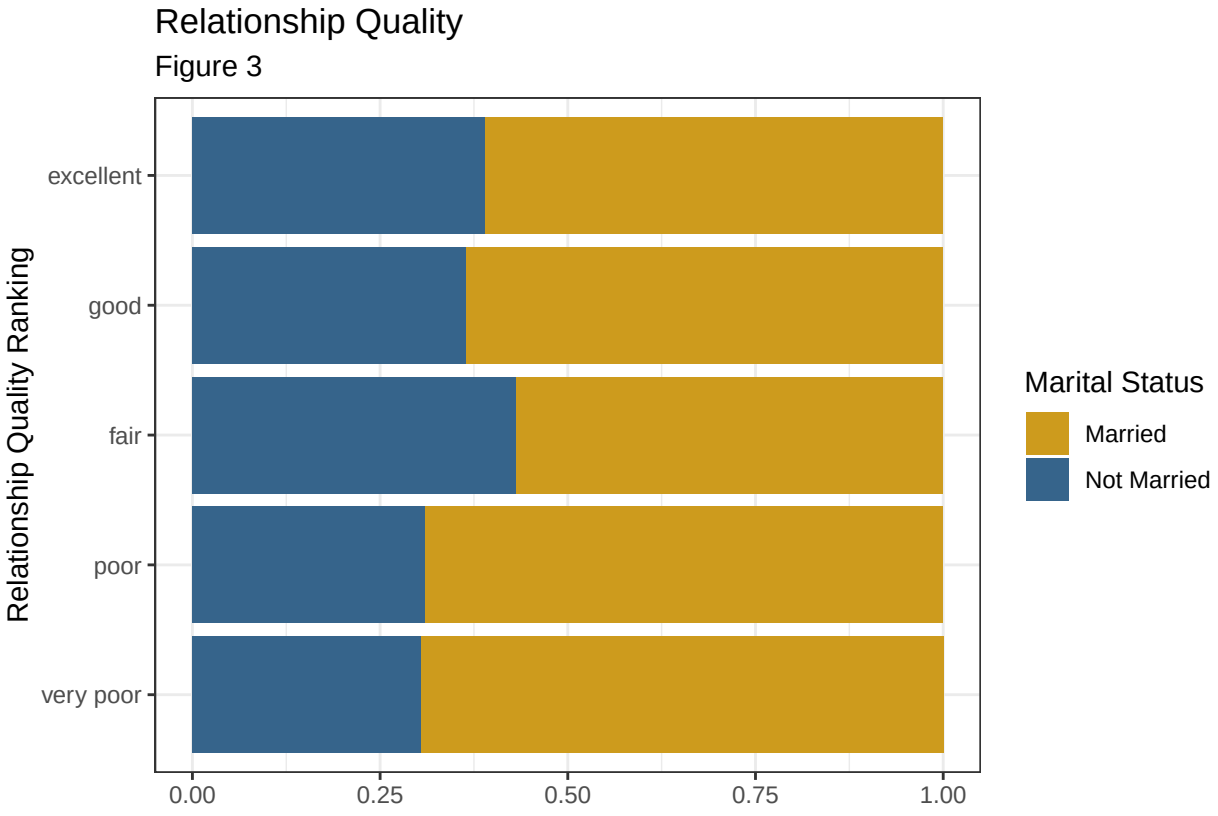
1-1.pdf

Figure 1 depicts the the political party respondents reported and their marital status. Our thought was that since there are differences in beliefs about relationships between Democrats and Republicans, this graph might provide insight into what our data represents. This graph illustrates a common theme in the dataset which is that there are more married people surveyed than not married people. We do tend to see that people who identify as Republican are married in higher rates compared to Democrats.



2-1.pdf

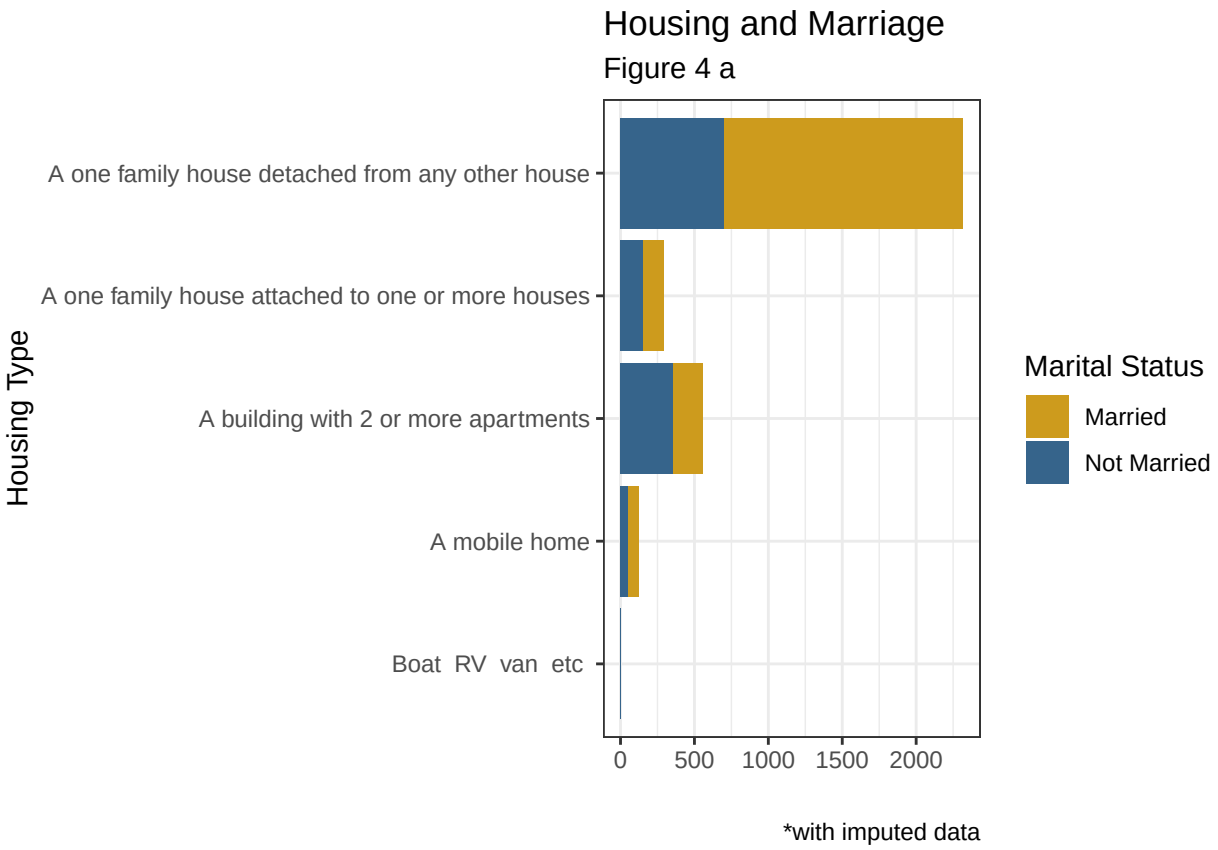
As seen in Figure 2, we can conclude that having a Bachelor's degree means that you are more likely to be married than if you have a different education level. Education level is a particularity interesting predictor to look at. When GfK performed the survey, they asked quite a few questions relating to education level, including education level of the partner's mother. Education is a component of income and social class an plays a vital role in connecting people. For these reasons we considered it important to plot this relationship between marital status and education.



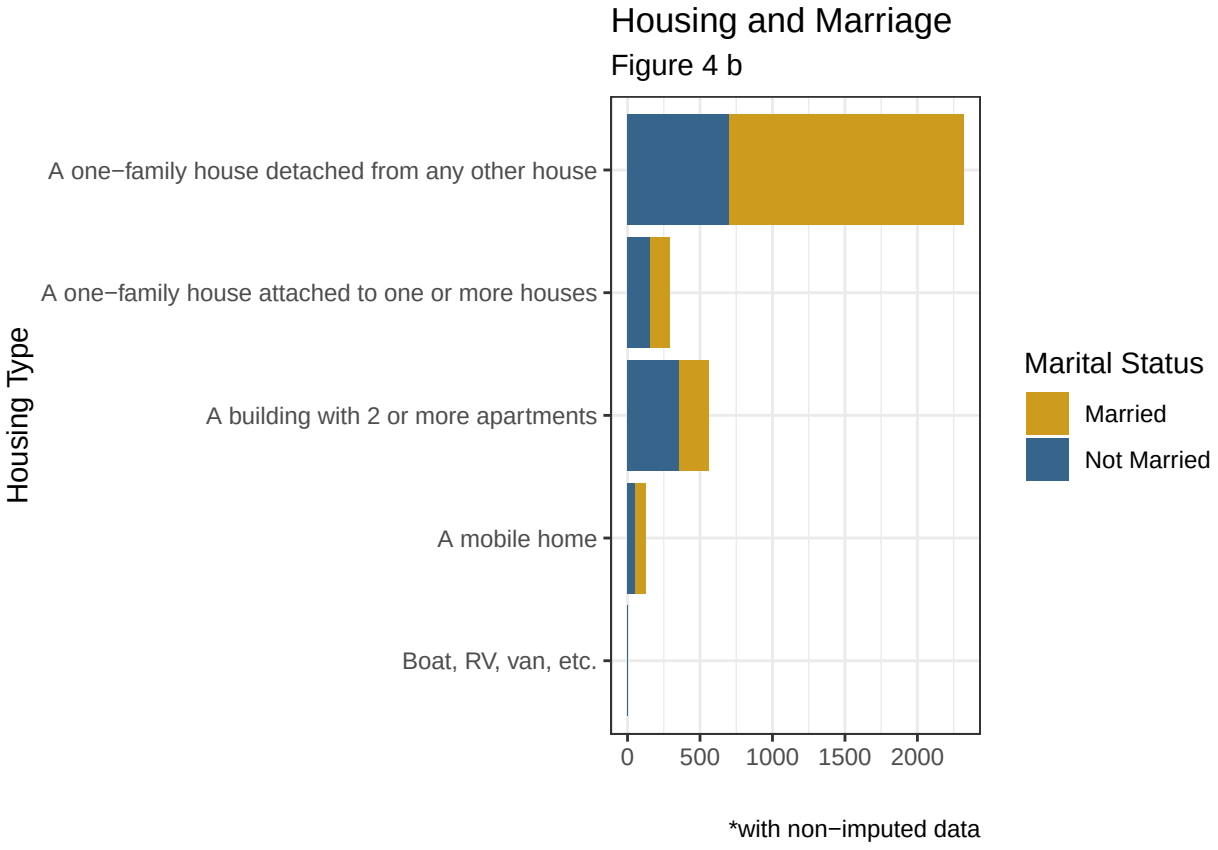
3-1.pdf

```
##   w6_relationship_quality    n
## 1                very poor   23
## 2                     poor   29
## 3                     fair  197
## 4                     good  877
## 5                excellent 2169
```

As seen in Figure 3, married and not married couples have approximately the same proportion of relationship quality rankings. However, those with "good" and "excellent" relationships do skew towards married. What this graph doesn't show, but that is present in the data, is that there are lot more "excellent" relationships than "poor" ones. A quick run of the count() function will show that there are 2169 "excellent" relationships while there are only 36 "poor" ones. While we didn't study Relationship Quality (because using a binary predictor open up more models and methods of analysis) we think that this is a predictor that deserves to be a response at some point. This is for a few reasons, but primary because relationship happiness and satisfaction it's an important aspect of human connection. Given the scope of our data, as well as the Stanford Research study on online dating and our study on marriage, we think that Relationship Quality is a great next step for this data.


Housing and Marriage
Figure 4 a
A one family house detached from any other house
A one family house attached to one or more houses
A building with 2 or more apartments
A mobile home
Boat RV van etc
Housing Type
0
500
1000
1500
2000
Marital Status
Married
Not Married
*with imputed data

4-1.pdf


Housing and Marriage
Figure 4 b
A one−family house detached from any other house
A one−family house attached to one or more houses
A building with 2 or more apartments
A mobile home
Boat, RV, van, etc.
Housing Type
0
500
1000
1500
2000
Marital Status
Married
Not Married
*with non−imputed data

4-2.pdf

Figures 4a and 4b are slightly different than the earlier visualizations. These two visualizations were made with the same methods and aesthetics as above, but with the specific purpose of comparing the way one of our factors changed before and after imputation. The impute function that we run on the HCMST dataset to get out iHCMST dataset is working correctly and we see a quite similar relationship between the dataset. In our non-imputed data from Fig. 4b, there are about five couples (3 married, 2 unmarried) who report their house as being "Boat/RV/van." When we look at the imputed data in Fig. 4a we can pretty clearly see that same reslaionship in "Boat/RV/van" and the other variables. The imputed data in Fig. 4b does not fill in any one-family detached houses. Overall, our impute does a good job as it does exactly what we thought our KNN-impute would do.

Statistical Methods Used & Diagnostics

We are going to be using several statistical methods to analyze our data. These methods include, data imputation using KNN (done in the Methods section), a logistic regression model, LASSO, LDA, Tree, and Random Forests.

Let's start with data imputation using KNN. We are using data imputation to solve an issue with our original dataset, which is that there are a significan number of NAs. We are actually first wrangling the dataset to remove both rows and columns with too many NAs (normally set at at least 3/4 NA). From this point, data imputation will fill in the NAs that are remaining based on KNN for that column.

Next lets move onto actually building our model. The first model we're making is a complete logistic regression model. We chose logistic regression rather than linear regression because it works nicer with binary classifier response variables like ours. Mostly this model will serve as a baseline on which to compare our improved models, but by looking at what factors are weighted more heavily by the model we can begin to understand which are most relevant.

We decided not to use best_subsets() (or any of the best_subset functions) because of the computational time. Our chosen method of model building is LASSO. We have selected LASSO as an appropriate method to analyze our data due to the size of our dataset. LASSO performs variable selection and regularization which is important to helping us find a good model since we still have a good 145 possible predictors (with the response and case numbers for keeping track of people gone). This method will also help us handle the possibility of collinearity in the data.

We are also using a Tree and randomForests in our model building efforts. Tree is a useful function for our purposes. It narrows down the number of predictors to relevant ones automatically and can point us to turning points in those predictors' values. Trees will be influenced by categorical variables with lots of categories. For this reason, we are also going to be running randomForests as a check on the Tree model.

We are using randomForest, both bagged and not bagged with the imputed dataset. Random Forests can handle lots of NA variables such as those in our original dataset, so we could make this model as well with our non-imputed data, however, we have chosen to look into the bagged model since with have other checks of how our KNN-impute is working.Random Forests' output is less immediately interpretative than that of Tree in an inference situation, but we can use RStudio to pull out information about which predictors were found useful most often.

Results

A description of the tools and methods used to build your models, an overview of the models themselves and a summary of their attributes, a discussion of model comparisons and accuracy, a presentation of model predictions, classifications, and/or parameter inference.

We will be splitting the data into a training and test with initial_split() so that we can test the accuracy of our models in making inferences.

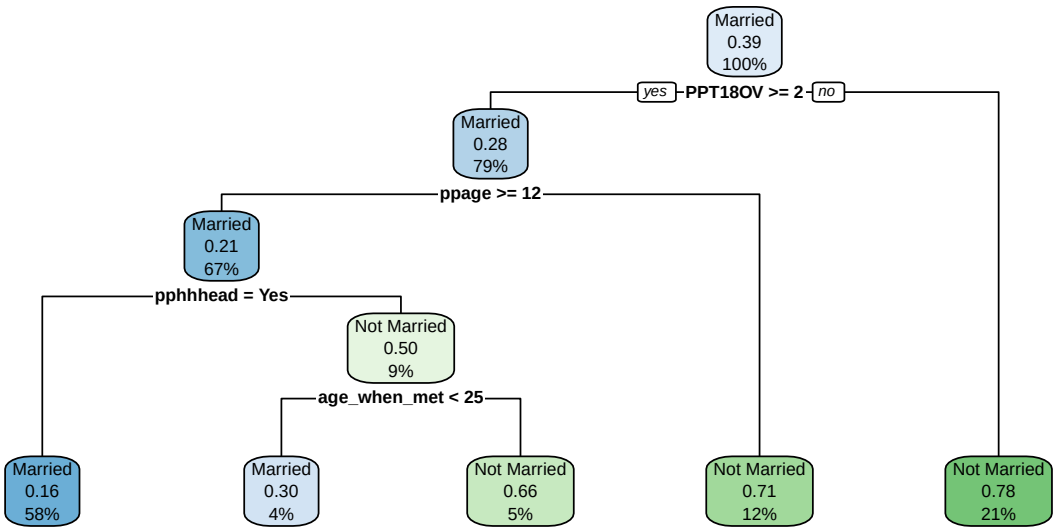
The first model that we create is the logistic regression model. We used the glm() function to create our model, and then created a data frame comparing the predictions made by the model and the true values on our test set created above. These were used to create an accuracy value, stored as logitAcc.

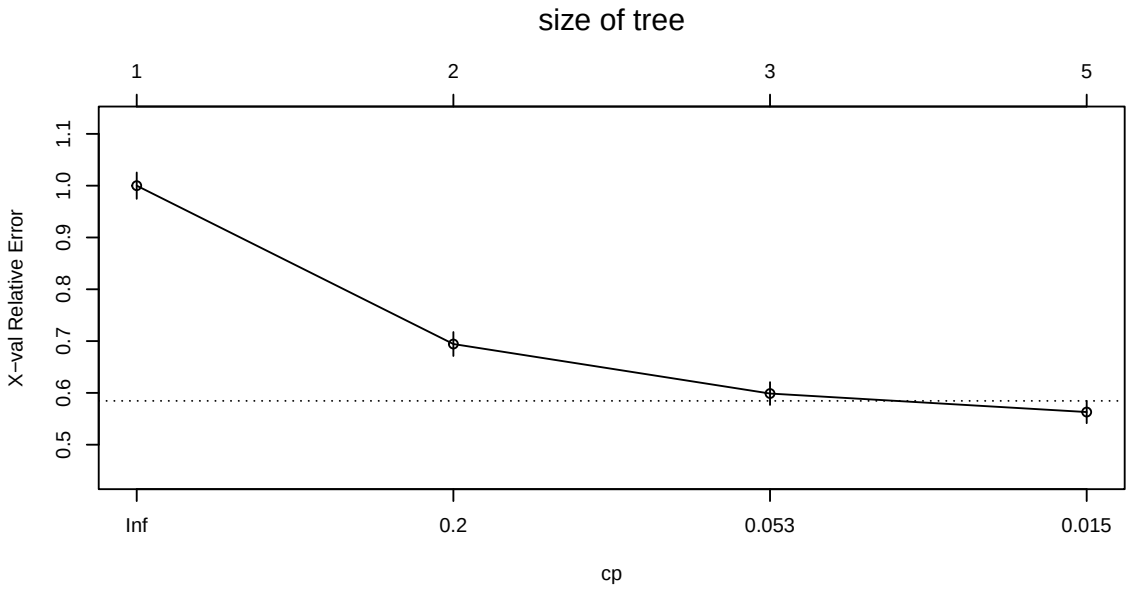
The LASSO is critical to our analysis as well. We make a model matrix out of the imputed data, then turn it into a training and test set matrix. We also create a grid of lambdas. We use both glmnet() to get our function and cv.glmnet to test for our best lambda value. As seen in the table below our best best minimum lambda value is 0.01 and our best lambda value within one standard error is 0.01321941. Pulling out this value, we create a data frame with observed and predicted values, as above.

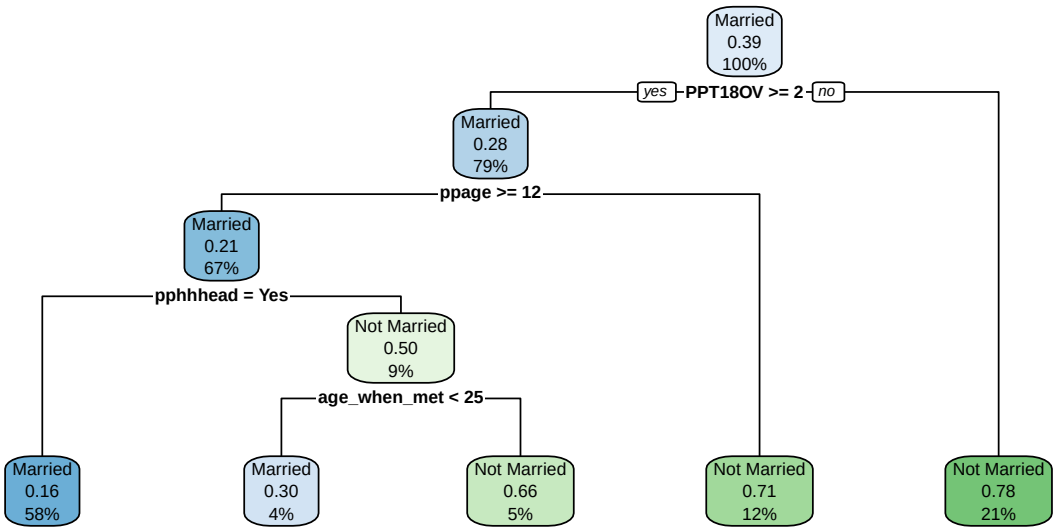
```
##   bestL        seL
## 1  0.01 0.01747528
```

We also create a model using LDA to deepen our analysis. This is done using the 'lda' function from the MASS package. We create a data frame for cross validation analysis, as above.

We are using a Pruned Tree in order to find the best predictors. The first tree is created using a basic rpart function with a minsplit of 1. prune.rpart is used to prune the tree, using a cp value decided on from a plot made by the plotcp function. That plot shown below, with the plot of the original tree and the pruned tree. Data frame made as in the previous sections.


Married
0.39
100%
yes
PPT18OV >= 2
no
Married
0.28
79%
ppage >= 12
Married
0.21
67%
pphhhead = Yes
Not Married
0.50
9%
age_when_met < 25
Married
0.16
58%
Married
0.30
4%
Not Married
0.66
5%
Not Married
0.71
12%
Not Married
0.78
21%


size of tree
1
2
3
5
X−val Relative Error
0.5
0.6
0.7
0.8
0.9
1.0
1.1
Inf
0.2
0.053
0.015
cp


Married
0.39
100%
yes
PPT18OV >= 2
no
Married
0.28
79%
ppage >= 12
Married
0.21
67%
pphhhead = Yes
Not Married
0.50
9%
age_when_met < 25
Married
0.16
58%
Married
0.30
4%
Not Married
0.66
5%
Not Married
0.71
12%
Not Married
0.78
21%

As mentioned in the Statistical Methods Used & Diagnostics section, we are using two different Random Forest models that serve as a check both on the prunned tree and on how the impute function is performing. The first model is run on the original HCMST dataset that has been wrangled but not imputed yet. We use the randomForest function with ntree = 10 and make a cross-validation data set.

We also have a bagged random forest which uses the imputed HCMST data. It is made on the unimputed dataset with the randomForest function and ntree=10, but with the mtry function set to 1 less than the number of columns in the input dataset in order to make the function do bagging.

In order to assess which model selected the best predictors for inference, we are using both the confusion materix and the tests of accuracy as seen below. The confusion matrices are made with the conf_mat function and the obs and preds information we generated for each model above. Based on the results of the different confusion matrices, the Logistical model, Trees model, Forest model, and Bagged model all had similar ranges for their Type I and II Errors both compared to each other and compared to the errors. However, the LASSO had vastly different Type I and II errors and these errors were either much higher or much lower than what the other models returned.

```
conf_mat(logitRes, truth = obs, estimate = preds)
```

```
##              Truth
## Prediction    Married Not Married
##   Married         438          97
##   Not Married      73         216
```

```
conf_mat(lassoRes, truth = obs, estimate = preds)
```

```
##              Truth
## Prediction    Married Not Married
##   Married         449         115
##   Not Married      62         198
```

```
conf_mat(treeRes, truth = obs, estimate = preds)
```

```
##              Truth
## Prediction    Married Not Married
##   Married         419          92
##   Not Married      92         221
```

```
conf_mat(forestRes, truth = obs, estimate = preds)
```

```
##              Truth
## Prediction    Married Not Married
##   Married         432          89
##   Not Married      79         224
```

```
conf_mat(bagRes, truth = obs, estimate = preds)
```

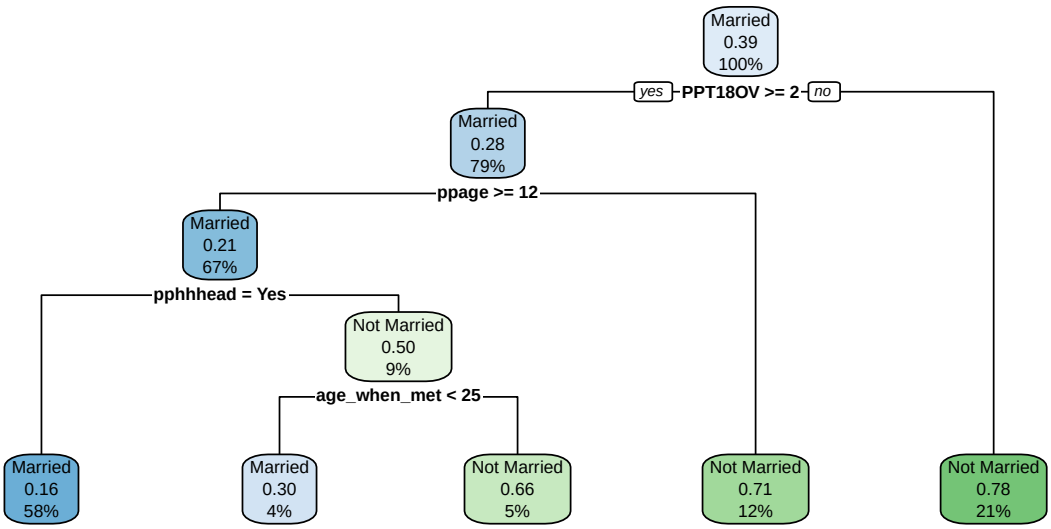
```
##              Truth
## Prediction    Married Not Married
##   Married         420          84
##   Not Married      91         229
```

Generating an accuracy value was part of the cross-validation work we did above. Here those values are aggregated into a data frame and presented. As seen below, the Logistic model has an accuracy of 0.7936893, the LASSO has an accuracy of 0.7924757, and the Tree has an accuracy of 0.7924757. Of the Random Forest models, the Bagged model returns an accuracy of 0.7912621 while the plain-jane Random Forest model returns an accuracy of 0.7961165. The closer to 1 the accuracy is, the more accurate the model. These results are very very similar, probably to the point that variation is do to the chance of the training and test data split rather than a significant difference in model accuracy. However, this does tell us one interesting

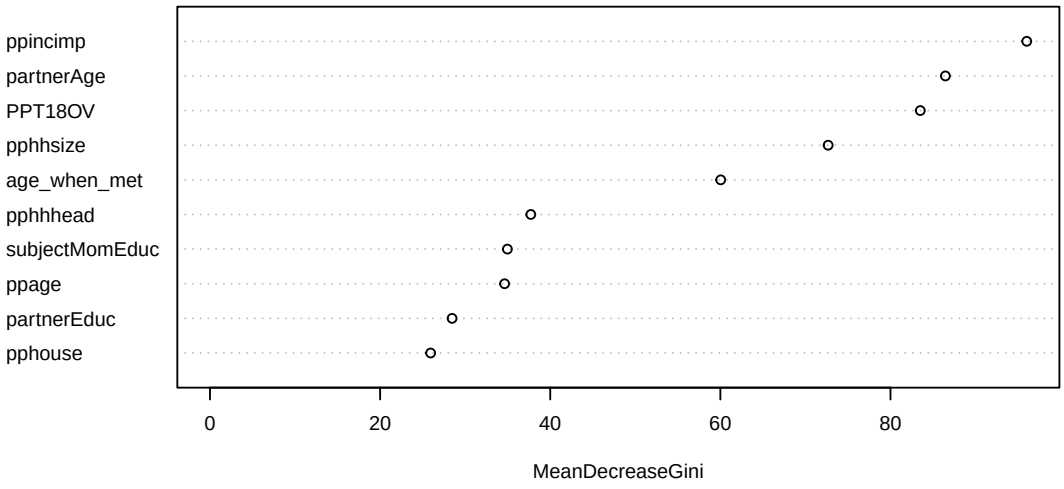
thing about the accuracy of the Tree model. Since the Tree model is susceptible being adversely affected by categorical variables with many levels, we know that that is not happening with our data since the accuracy is so close to all the other models accuracies, especially the Random Forest accuracy.

```
if(Binom){
  Accs <- data.frame(
    model = c("logit", "lasso", "tree", "forest", "bag"),
    acc = c(logitAcc, lassoAcc, treeAcc, forestAcc, bagAcc)
  )
}else{
  Accs <- data.frame(
    model = c("lda", "tree", "forest", "bag"),
    acc = c(ldaAcc, treeAcc, forestAcc, bagAcc)
  )
}
Accs
```

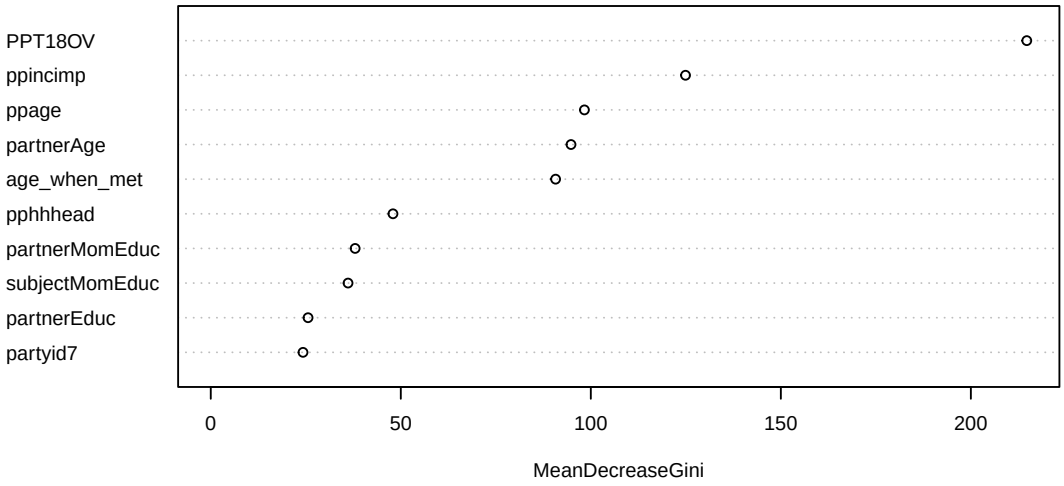
```
##    model       acc
## 1  logit 0.7936893
## 2  lasso 0.7851942
## 3   tree 0.7851942
## 4 forest 0.7961165
## 5    bag 0.7876214
```



forestMod



bagMod



Discussion

A review of the results generated from the model and synthesis with the context from which the data was generated or observed, a restatement of research objective and an answer to the original research question, a discussion limitations of the study as well as areas for further research.

The reserach objective when we looked at this dataset was to determine what variables tend to lead to marriage relationships. Our results show that PPT18OV, partnerAge, ppage, ppincimp, and age_when_met tend to be the most important variables used to predict whether a couple is married. PPT18OV measures the number of children over the age of 18 in a particular household, so couples with older children tend to be married. Additionally, partnerAge and ppage both measure the age of the members of the relationship, while age_when_met captures the length of the relationship. Finally, ppincimp measures the income of the couple. These particular variables show that older, wealthier couples with established households are more likely to be married. Interestingly, the variable howMetLength seems to indicate that couples with longer stories surrounding how they met tend to have higher rates of marriage. However, the overwhelming theme in the variable importance plots is that age and income are the best predictors of whether a couple is married.

There are a few limitations of our data and some exciting areas of future research. One of the limitations of our study was the nature of survey data which needed lots of wrangling to get it into a usable form. This survey data also had several years worth of data prior to 2017 but since every year they ask different question, it was not usable for this project. One of the analysis obstacles of the data is that "Self-identified Lesbian, Gay, and Bisexual respondents were oversampled" which means that the conclusions that we are going to be drawing will represent queer relationships at a level which is not representative of the overall U.S. population. Although we don't consider this obstacle to affect our particular study in an impactful way. In terms of areas for further research, we consider relationship quality/relationship happiness to something that deserves further investigation as discussed with Figure 3 either as something analyzed with this data or not. Looking at relationship happiness might be a more holistic way of viewing relationships compared to looking at online dating or predictors of marriage. These would be fun things to look into.

Code Appendix

A collection of code used to process data, perform analysis, and build models. To avoid excessive run-times when compiling the document, consider adding eval = F to the chunk options (which will force the code not to be run when compiling the document into .pdf or .html)

###Libraries

```
# Here is a list of libraries that we are using
library(MASS)
library(tidyverse)
library(tidymodels)
library(GGally)
library(haven)
library(glmnet)
library(rpart)
library(rpart.plot)
library(randomForest)

if(!require(impute)){
  if(!require("BiocManager")){install.packages("BiocManager")}
```

```
  BiocManager::install("impute")}
library(impute)

if(!require(caret)){install.packages("caret")}
library(caret)

theme_set(theme_bw())
```

###Custom Functions

```
#Extracts variables with largest correlations
nMaxCor <- function(Matrix, n){
  Matrix <- round(Matrix, 5)
  Cor <- Matrix
  diag(Cor) <- 0
  Cor[upper.tri(Cor)] <- 0
  Cor <- abs(Cor)
  Names <- rownames(Cor)
  Results <- data.frame()
  for (i in 1:n) {
    Results[i,1] <- Names[which(Cor == max(Cor), arr.ind = T)[1]]
    Results[i,2] <- Names[which(Cor == max(Cor), arr.ind = T)[2]]
    Results[i,3] <- Matrix[which.max(Cor)]
    Cor[which.max(Cor)] <- 0
  }
  names(Results) <- c("Var1", "Var2", "Cor")
  Results
}

#Converts dataframe w/ dummies back to original
modMatMerge <- function(DataMat, DataNames){
  library(tidyverse)
  mNames <- colnames(DataMat)
  Groups <- list()
  Subsets <- list()
  Levels <- list()
  Merged <- list()

  for(k in 1:length(DataNames)){
    Var <- DataNames[k]
    Groups[[k]] <- mNames[grep(paste0("^", Var), mNames)]
    Subsets[[k]] <- dplyr::select(DataMat, Groups[[k]])
    if(length(Groups[[k]]) > 1){
      Levels[[k]] <- str_remove(colnames(Subsets[[k]]), Var)
      Merged[[k]] <- as.factor(Levels[[k]][max.col(Subsets[[k]])])
    } else {
      Merged[[k]] <- as.numeric(DataMat[[Groups[[k]]]])
    }
  }
  Data <- data.frame(Merged[1:length(DataNames)])
  colnames(Data) <- DataNames
  Data
}
```

###Data Wrangling

```
HCMST <- read_dta("HCMST 2017 fresh sample for public sharing draft v1.1.dta")
HCMST <- as_factor(HCMST)

HCMST <- HCMST %>%
  rename(
    partnerGender = w6_q4,
    sameSex = w6_q5,
    partnerHispanic = w6_q6a,
    partnerRace = w6_q6b,
    partnerAge = w6_q9,
    partnerEduc = w6_q10,
    partnerMomEduc = w6_q11,
    partnerPolitic = w6_q12,
    subjectMomEduc = w6_q14,
    relativesSeen = w6_q16,
    topEarner = w6_q23,
    howMetLength = w6_q24_length,
    sameHighSch = w6_q25,
    sameCollege = w6_q26,
    sameHometown = w6_q27,
    parentsFirst = w6_q28,
    meetOnline = w6_q32,
    relQuality = w6_q34
  ) %>%
  mutate(
    S1 = as.factor(case_when(S1 == levels(S1)[2] ~ "Married",
                             S1 == levels(S1)[3] ~ "Not Married")),
    S2 = as.factor(case_when(S2 == levels(S2)[2] ~ "Sexual Partner",
                             S2 == levels(S2)[3] ~ "Nonsexual Partner")),
    S3 = as.factor(case_when(S3 == levels(S3)[2] ~ "Past Relationship",
                             S3 == levels(S3)[3] ~ "No Relationships")),
    sameCollege = as.factor(case_when(sameCollege == 1 ~ "Yes",
                                      sameCollege == 2 ~ "No")),
    relStatus = as.factor(coalesce(S3, S2, S1)),
    relEnd = coalesce(w6_relationship_end_mar, w6_relationship_end_nonmar),
    timeMeetDate = w6_q21b_year - w6_q21a_year,
    timeDateMarry = w6_q21d_year - w6_q21b_year,
    timeDateCohab = w6_q21c_year - w6_q21b_year,
  ) %>%
  dplyr::select(-c(grep("weight", names(HCMST)),
                   grep("[Yy]ear|duration", names(HCMST)),
                   grep(paste(paste0("^Q",1:34),collapse = "|"), names(HCMST)),
                   grep("^w6_friend_connect.*[^all]$", colnames(HCMST)),
                   S2, S3,
                   DOV_Branch, speed_flag, qflag, consent,
                   xlgb,
                   ppagecat, ppagect4,
                   ppeduc, ppmarit, ppreg9,
                   Race_1, Race_2, Race_3, Race_4, Race_5, Race_6,
                   race1, race2, race3, race4, race5,
                   race6, race7, race8, race9, race10,
                   race11, race12, race13, race14, race15,
                   w6_took_the_survey,
```

```
                    w6_q21a_month, w6_q21a_month_flag,
                    w6_q21b_month, w6_q21b_month_flag,
                    w6_married, w6_identity, w6_identity_2,
                    w6_q15a1_truncated, w6_q15a4_truncated, w6_q15a7,
                    w6_otherdate,
                    w6_q17,
                    partnership_status,
                    female))

levels(HCMST$relEnd) <- c("Refused", "Divorce", "Separation", "Death", "Breakup", "Together")
HCMST$relEnd <- replace_na(HCMST$relEnd, "Together")
HCMST[HCMST == "Refused"] <- NA
HCMST <- droplevels(HCMST)

#Removes columns with many NA
for(i in names(HCMST)){
  if(sum(is.na(HCMST[i])) > nrow(HCMST)/5){
    HCMST <- dplyr::select(HCMST,-all_of(i))
  }
}

#Removes rows with many NA
for(i in nrow(HCMST):1){
  if(sum(is.na(HCMST[i,])) >= ncol(HCMST)/3){
    HCMST <- HCMST[-i,]
  }
}

#Ensures numeric vars are encoded as such
numVars <- c("partnerAge", "ppage", "pphhsize",
             "PPT01", "PPT25", "PPT612", "PPT1317", "PPT18OV",
             names(HCMST)[grep("yrsed", names(HCMST))])

for(i in numVars){
  HCMST[i] <- as.numeric(unlist(HCMST[i]))
}

HCMST <- droplevels(HCMST) #Drops unused levels (e.g. "Refused") from all factors

HCMSTdummy <- dummyVars(~ ., data = HCMST, sep = "", na.action = "na.pass")
HCMSTmat <- as.matrix(data.frame(predict(HCMSTdummy, newdata = HCMST)))

#Imputes data w/ KNN
Imputed <- impute.knn(HCMSTmat, k = 30, maxp = 4000, rng.seed=12)

#Collapses dummy variables back to original data frame format
iHCMST <- modMatMerge(as.data.frame(Imputed$data), colnames(HCMST))
for(k in colnames(iHCMST)){
  levels(iHCMST[[k]]) <- str_replace_all(levels(iHCMST[[k]]), "\\.", " ")
}
```

###Data Exploration

```r
#Reordering Levels
iHCMST <- iHCMST %>%
  mutate(partyid7 = fct_relevel(partyid7, c("Strong Democrat",
                                            "Not Strong Democrat",
                                            "Leans Democrat",
                                            "Undecided Independent Other",
                                            "Leans Republican",
                                            "Not Strong Republican",
                                            "Strong Republican")),
         ppeducat = fct_relevel(ppeducat, c("Less than high school",
                                            "High school",
                                            "Some college",
                                            "Bachelor s degree or higher")),
         w6_relationship_quality = fct_relevel(w6_relationship_quality,c("very poor",
                                                                         "poor",
                                                                         "fair",
                                                                         "good",
                                                                         "excellent")),
         pphouse = fct_relevel(pphouse, c("Boat  RV  van  etc ",
                                          "A mobile home",
                                          "A building with 2 or more apartments",
                                          "A one family house attached to one or more houses",
                                          "A one family house detached from any other house"))
  )
HCMST <- mutate(HCMST, pphouse = fct_relevel(pphouse, c("Boat, RV, van, etc.",
                                                        "A mobile home",
                                                        "A building with 2 or more apartments",
                                                        "A one-family house attached to one or more hous
                                                        "A one-family house detached from any other hous

ggcorr(HCMST, hjust = 1, size = 3, layout.exp = 6)

ggplot(data = iHCMST, mapping = aes(x = as.factor(partyid7), fill = as.factor(S1))) +
  geom_bar(position = "dodge") +
  coord_flip() +
  scale_fill_manual(values = c("goldenrod3", "steelblue4")) +
  labs(title = "Political Party and Marriage Status", subtitle = "Figure 1",
       y = "", x = "Political Party Identification") +
  guides(fill = guide_legend(title = "Marital Status"))

ggplot(data = iHCMST, mapping = aes(x = as.factor(ppeducat), fill = as.factor(S1))) +
  geom_bar() +
  coord_flip() +
  scale_fill_manual(values = c("goldenrod3", "steelblue4")) +
  labs(title = "How Education Level Effects Marriage", subtitle = "Figure 2", y = "",
       x = "Education Level") +
  guides(fill = guide_legend(title = "Marital Status"))

ggplot(data = iHCMST, mapping = aes(x = as.factor(w6_relationship_quality), fill = as.factor(S1))) +
  geom_bar(position = "fill") +
  coord_flip() +
  scale_fill_manual(values = c("goldenrod3", "steelblue4")) +
  labs(title = "Relationship Quality", subtitle = "Figure 3", y = "", x = "Relationship Quality Ranking
```

```
  guides(fill = guide_legend(title = "Marital Status"))

iHCMST %>%
   count(w6_relationship_quality)

# The following two graphs show the effect of our imputation on our data
ggplot(data = iHCMST, mapping = aes(x = as.factor(pphouse),
                                    fill = as.factor(S1))) +
  geom_bar() +
  coord_flip() +
  scale_fill_manual(values = c("goldenrod3", "steelblue4")) +
  labs(title = "Housing and Marriage",
       subtitle = "Figure 4 a", y = "",
       x = "Housing Type", caption = "*with imputed data") +
  guides(fill = guide_legend(title = "Marital Status"))

ggplot(data = HCMST, mapping = aes(x = as.factor(pphouse), fill = as.factor(S1))) +
  geom_bar() +
  coord_flip() +
  scale_fill_manual(values = c("goldenrod3", "steelblue4")) +
  labs(title = "Housing and Marriage",
       subtitle = "Figure 4 b", y = "",
       x = "Housing Type", caption = "*with non-imputed data") +
  guides(fill = guide_legend(title = "Marital Status"))
```

###Model Building

```
# Data Split
Indeces <- data.frame(index = 1:nrow(iHCMST))
set.seed(60)
Split <- initial_split(Indeces)
trainIndeces <- training(Split)[[1]]
testIndeces <- testing(Split)[[1]]
set.seed(NULL)

# Create formula for regression
fmla <- S1 ~ . # where S1 is marriage status
resp <- fmla[[2]]
if(resp == "S1"){
  iHCMST <- dplyr::select(iHCMST, -c(relStatus, relEnd, w6_q19))
}
Binom <- ifelse(length(levels(HCMST[[resp]])) == 2, TRUE, FALSE)

trainHCMST <- iHCMST[trainIndeces,]
testHCMST <- iHCMST[testIndeces,]

#Logistic Model
logitMod <- glm(fmla, trainHCMST, family = "binomial")
logitRes <- data.frame(
  obs = testHCMST$S1,
  probs = predict(logitMod, testHCMST, type = "response")
)
logitRes$preds <- as.factor(ifelse(logitRes$probs >= 0.5, "Not Married", "Married"))
```

```
logitAcc <- accuracy(logitRes, truth = obs, estimate = preds)$.estimate

#LASSO Model
modMat <- model.matrix(fmla, iHCMST)[,-1]

trainX <- modMat[trainIndeces,]
trainY <- trainHCMST$S1
testX <- modMat[testIndeces,]
testY <- testHCMST$S1
grid = 10^(seq( -2, 10, length = 100))

lassoMod <- glmnet(trainX, trainY, alpha = 1, lambda = grid, family = "binomial")
lassoCV <- cv.glmnet(trainX, trainY, alpha = 1, lambda = grid, nfolds = 10, family = "binomial")
bestL <- lassoCV$lambda.min
seL <- lassoCV$lambda.1se
data.frame(bestL, seL)

lassoRes <- data.frame(
  obs = testY,
  preds = as.factor(predict(lassoMod, s = seL, newx = testX, type = "class"))
)

lassoAcc <- accuracy(lassoRes, truth = obs, estimate = preds)$.estimate

#Potential LDA Model. Only run if Binom = FALSE
ldaMod <- lda(fmla, trainHCMST)

ldaPred <- predict(ldaMod, newdata = testHCMST)
ldaRes <- data.frame(
  obs = as.factor(testHCMST[[resp]]),
  preds = ldaPred$class,
  probs = ldaPred$posterior[,2]
)

ldaAcc <- accuracy(ldaRes, truth = obs, estimate = preds)$.estimate

#Tree Model
treeMod <- rpart(fmla, trainHCMST, minsplit = 1)

par(mfrow=c(3,1))

rpart.plot(treeMod)

plotcp(treeMod)

pruneMod <- prune.rpart(treeMod, cp = 0.017)
rpart.plot(pruneMod)
treeRes <- data.frame(
  obs = testY,
  preds = predict(pruneMod, testHCMST, type = "class")
)
treeAcc <- accuracy(lassoRes, truth = obs, estimate = preds)$.estimate
```

```
#Forest Model
forestMod <- randomForest(fmla, data = trainHCMST, ntree = 10)

forestRes <- data.frame(
  obs = testY,
  preds = as.factor(predict(forestMod, testHCMST))
)
forestAcc <- accuracy(forestRes, truth = obs, estimate = preds)$.estimate

#Bag Model
bagMod <- randomForest(fmla, data = trainHCMST, ntree = 10, mtry = ncol(trainHCMST)-1)
bagRes <- data.frame(
  obs = testY,
  preds = as.factor(predict(bagMod, testHCMST))
)
bagAcc <- accuracy(bagRes, truth = obs, estimate = preds)$.estimate

#Confusion Matrices
conf_mat(logitRes, truth = obs, estimate = preds)
conf_mat(lassoRes, truth = obs, estimate = preds)
conf_mat(treeRes, truth = obs, estimate = preds)
conf_mat(forestRes, truth = obs, estimate = preds)
conf_mat(bagRes, truth = obs, estimate = preds)

#Accuracy Table
if(Binom){
  Accs <- data.frame(
    model = c("logit", "lasso", "tree", "forest", "bag"),
    acc = c(logitAcc, lassoAcc, treeAcc, forestAcc, bagAcc)
  )
}else{
  Accs <- data.frame(
    model = c("lda", "tree", "forest", "bag"),
    acc = c(ldaAcc, treeAcc, forestAcc, bagAcc)
  )
}
Accs

#ROC Curve areas, not working.
if(Binom){
  ROCareas <- data.frame(
    model = c("logit", "lasso", "tree", "bag", "forest"),
    ROCarea = c(roc_auc(logitRes, truth = obs, estimate = preds),
                roc_auc(lassoRes, truth = obs, estimate = preds),
                roc_auc(treeRes, truth = obs, estimate = preds),
                roc_auc(bagRes, truth = obs, estimate = preds),
                roc_auc(forestRes, truth = obs, estimate = preds))
  )
}else{
  ROCareas <- data.frame(
    model = c("lda", "tree", "bag", "forest"),
    ROCarea = c(roc_auc(ldaRes, truth = obs, estimate = preds),
                roc_auc(treeRes, truth = obs, estimate = preds),
```

```
                roc_auc(bagRes, truth = obs, estimate = preds),
                roc_auc(forestRes, truth = obs, estimate = preds))
  )
}
ROCareas

#Tree and variable importance plots.
par(mfrow=c(3,1))

# Pruned model
rpart.plot(pruneMod)

# Important variables in forest model
varImpPlot(forestMod, n.var = 10)

# Important variables in bagged model
varImpPlot(bagMod, n.var = 10)
```

References

The citations for any data sets, literature or resources directly or indirectly referenced in your report, along with any sources you consulted during your investigation that had a significant impact on your analysis. Citations should be made according to the ASA style guide: https://amstat.tfjournals.com/asa-style-guide/

IDE, Spyder. "Okcupid Datasets." Figshare, Figshare, 15 July 2021, https://figshare.com/articles/dataset/OKCupid_Datasets/14987388.

"Obergefell v. Hodges." Oyez, www.oyez.org/cases/2014/14-556. Accessed 8 Dec. 2021

Resnick, Brian. "Researchers Just Released Profile Data on 70,000 Okcupid Users without Permission." Vox, Vox, 12 May 2016, https://www.vox.com/2016/5/12/11666116/70000-okcupid-users-data-release.

Rosenfeld, Michael J., Reuben J. Thomas, and Sonia Hausen. 2019. *How Couples Meet and Stay Together 2017 fresh sample.* [Computer files]. Stanford, CA: Stanford University Libraries.

Rosenfeld, Michael J., Reuben J. Thomas, and Sonia Hausen. 2019. *Disintermediating your friends: How Online Dating in the United States displaces other ways of meeting* Proceedings of the National Academy of Sciences. vol. 116, iss. 36, https://doi.org/10.1073/pnas.1908630116.
