Data Preparation

Zachary Levonian

11/02/2018

```
library(alr4)
```

```
## Loading required package: car
## Loading required package: carData
## Loading required package: effects
## lattice theme set by effectsTheme()
## See ?effectsTheme for details.
```

```
library(mice)  # for multiple imputation
```

```
## Loading required package: lattice
##
## Attaching package: 'mice'
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(BaylorEdPsych)  # For Little's MCAR test
library(polycor)  # To compute correlation between heterogenous variables
```

Load data

```
train <- read.csv("../../data/raw/train.csv", stringsAsFactors=FALSE)
test <- read.csv("../../data/raw/test.csv", stringsAsFactors=FALSE)
```

Combine the data into a single dataframe to make it easier to work with. I denote data in the test set with `Survived = 2`.

```
test$Survived = 2
df <- rbind(train, test)
```

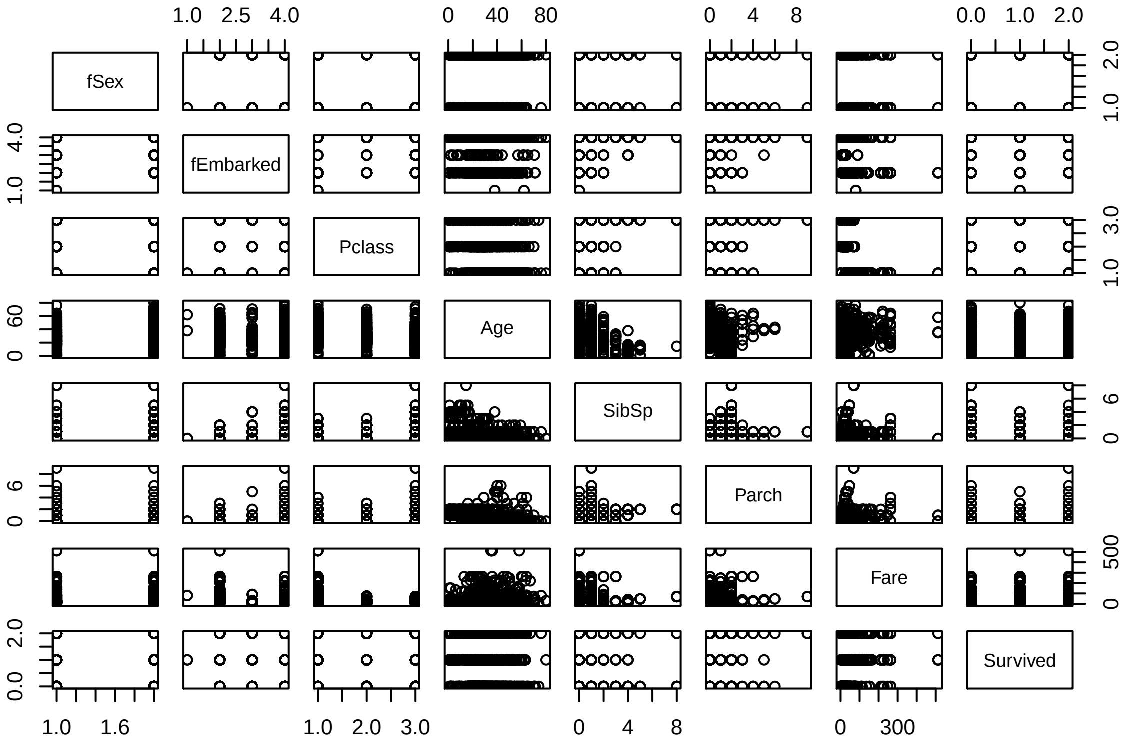
Data exploration

Build factors from data

```
df$fSex = factor(df$Sex)
df$fEmbarked = factor(df$Embarked)
```

High-level summaries and visualization

```
pairs(df[c("fSex", "fEmbarked", "Pclass", "Age", "SibSp", "Parch", "Fare", "Survived")])
```



Missing data

```
sapply(df, function(x) sum(is.na(x)))
```

```
## PassengerId    Survived      Pclass        Name         Sex         Age
##           0           0           0           0           0         263
##       SibSp       Parch      Ticket        Fare       Cabin    Embarked
##           0           0           0           1           0           0
##        fSex   fEmbarked
##           0           0
```

It looks like the only data that's missing is Age data, and a single instance of missing Fare data (in the test set).

Fare missing data

```
# print the row where Fare info is missing
df[is.na(df["Fare"])]
```

```
##  [1] "1044"                "2"                   "3"
##  [4] "Storey, Mr. Thomas" "male"                "60.50"
##  [7] "0"                   "0"                   "3701"
## [10] NA                    ""                    "S"
## [13] "male"                "S"
```

We need to impute this value, but as there's only a single missing value it's impossible to determine if the data is missing at random or not.

We will assume the data is missing at random and impute a value for Thomas Storey's fare using `mice`.

```
#TODO use mice to impute the data
```

Age missing data

263 passengers are missing age data.

First, we want to determine if the data are missing at random (MAR) or completely at random (MCAR).

```
age_little_df <- df[,c("fSex", "fEmbarked", "Pclass", "Age", "SibSp", "Parch", "Survived")]
mcar <- LittleMCAR(age_little_df)
```

```
## Loading required package: mvnmle
```

```
## Warning in nlm(lf, startvals, ...): NA/Inf replaced by maximum positive
## value

## Warning in nlm(lf, startvals, ...): NA/Inf replaced by maximum positive
## value
```

```
## this could take a while
```

```
mcar$missing.patterns
```

```
## [1] 2
```

```
mcar$amount.missing
```

```
##                 fSex fEmbarked Pclass         Age SibSp Parch Survived
## Number Missing     0         0      0 263.0000000     0     0        0
## Percent Missing    0         0      0   0.2009167     0     0        0
```

Little's MCAR test tests the null hypothesis that the missing data are MCAR. Thus, we have evidence that we ought to reject the null hypothesis and the missing age data are MAR. [**?**] [1]

```
age_little_df <- df[,c("fSex", "fEmbarked", "Pclass", "SibSp", "Parch", "Fare", "Survived")]
age_little_df$AgeMissing = as.numeric(is.na(df["Age"]))
hetcor(age_little_df)
```

```
##
## Two-Step Estimates
##
## Correlations/Type of Correlation:
##                fSex  fEmbarked     Pclass       SibSp      Parch       Fare
## fSex              1 Polychoric Polyserial Polyserial Polyserial Polyserial
## fEmbarked   0.1786          1 Polyserial Polyserial Polyserial Polyserial
## Pclass      0.1562     0.2064          1     Pearson    Pearson    Pearson
## SibSp       -0.135     0.1112    0.06116          1    Pearson    Pearson
## Parch      -0.2612    0.08228    0.01862     0.3735          1    Pearson
## Fare       -0.2287    -0.2535    -0.5586     0.1602     0.2215          1
## Survived   -0.2836    -0.1667    -0.1531   -0.04395    0.03505     0.1231
## AgeMissing 0.08225     -0.167     0.2086  -0.007873   -0.08229    -0.1306
##              Survived AgeMissing
## fSex       Polyserial Polyserial
## fEmbarked  Polyserial Polyserial
```

```
## Pclass        Pearson    Pearson
## SibSp         Pearson    Pearson
## Parch         Pearson    Pearson
## Fare          Pearson    Pearson
## Survived            1    Pearson
## AgeMissing   -0.02785          1
##
## Standard Errors:
##               fSex fEmbarked  Pclass   SibSp   Parch    Fare Survived
## fSex
## fEmbarked  0.04391
## Pclass      0.0341   0.03242
## SibSp      0.03394   0.04095 0.02756
## Parch      0.03282   0.03927 0.02765 0.02381
## Fare       0.03365   0.03071 0.01904 0.02695  0.0263
## Survived   0.03236   0.03397 0.02701 0.02761 0.02763 0.02724
## AgeMissing 0.03593   0.03163 0.02646 0.02766 0.02747 0.02719  0.02764
##
## n = 1308
##
## P-values for Tests of Bivariate Normality:
##                  fSex  fEmbarked Pclass SibSp Parch Fare Survived
## fSex
## fEmbarked     0.02766
## Pclass     4.277e-213 5.744e-249
## SibSp               0          0      0
## Parch               0          0      0     0
## Fare                0          0      0     0     0
## Survived   2.255e-208 9.276e-168      0     0     0    0
## AgeMissing          0          0      0     0     0    0        0
```

A missing age value is correlated positively with passenger class ($r = 0.2082$) and negatively with point of embarkment ($r = -0.1672$) and passenger fare ($r = -0.1306$). All other correlations are < 0.1.

I'm inclined to think that the true mediator of missing age (among the covariates in the dataset) is passenger class, which embarkment and fare both correlate with.

```r
t.test(df[is.na(df["Age"]),"Fare"], df[!is.na(df["Age"]),"Fare"])
```

```
##
##  Welch Two Sample t-test
##
## data:  df[is.na(df["Age"]), "Fare"] and df[!is.na(df["Age"]), "Fare"]
## t = -6.9669, df = 852.61, p-value = 6.481e-12
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -21.61344 -12.11208
## sample estimates:
## mean of x mean of y
##  19.82332  36.68608
```

Save the cleaned-up data

Now, we save all the columns to be used as potential features to a file.

```r
toWrite <- df[,c("PassengerId", "fSex", "fEmbarked", "Pclass", "Age", "SibSp", "Parch", "Fare", "Survive
write.table(toWrite, file="../../data/derived/factorized_data.csv",
            row.names=FALSE, col.names=TRUE,
            sep=",", quote=FALSE)
```

Train a Model

The code below demonstrates: - Reading in the features - Splitting the data into the training and test data - Training a regression model - Predicting the test set based on the trained model - Saving the predictions in the Kaggle format for submission

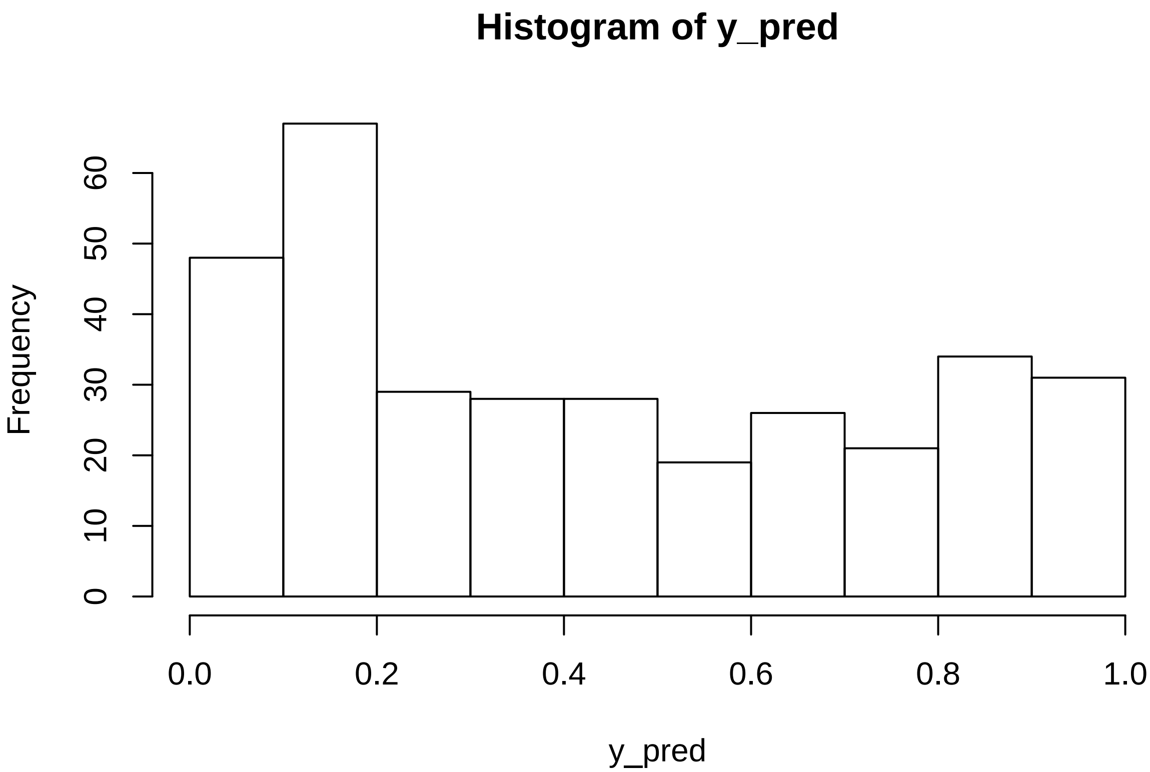
```r
df <- read.csv("../../data/derived/factorized_data.csv", stringsAsFactors=TRUE)
```

```r
train <- df[df$Survived != 2, ]
test <- df[df$Survived == 2, ]
```

```r
md <- glm(Survived ~ fSex + fEmbarked + Pclass + Age + SibSp + Parch + Fare, family="binomial", data=tra
summary(md)
```

```
##
## Call:
## glm(formula = Survived ~ fSex + fEmbarked + Pclass + Age + SibSp +
##     Parch + Fare, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7233  -0.6439  -0.3772   0.6288   2.4457
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   17.894850 607.855474   0.029  0.97651
## fSexmale      -2.638476   0.222256 -11.871  < 2e-16 ***
## fEmbarkedC   -12.257443 607.855250  -0.020  0.98391
## fEmbarkedQ   -13.080988 607.855453  -0.022  0.98283
## fEmbarkedS   -12.658656 607.855228  -0.021  0.98339
## Pclass        -1.199251   0.164619  -7.285 3.22e-13 ***
## Age           -0.043350   0.008232  -5.266 1.39e-07 ***
## SibSp         -0.363208   0.129017  -2.815  0.00487 **
## Parch         -0.060270   0.123900  -0.486  0.62666
## Fare           0.001432   0.002531   0.566  0.57165
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 964.52  on 713  degrees of freedom
## Residual deviance: 632.34  on 704  degrees of freedom
##   (177 observations deleted due to missingness)
## AIC: 652.34
##
## Number of Fisher Scoring iterations: 13
```

```r
# predict on the test set
y_pred <- predict(md, test, type="response")
hist(y_pred)
```




```r
# save the predictions to a file that can be submitted to Kaggle
test$Survived = as.numeric(y_pred > 0.5)
# while there shouldn't be NAs in the test output,
# here we apply a nasty hack to ensure any NAs are numeric in the output
test[is.na(test)] <- 0
toWrite <- test[,c("PassengerId", "Survived")]
write.table(toWrite, file="../../data/derived/testModelPredictions.csv",
            row.names=FALSE, col.names=TRUE,
            sep=",", quote=FALSE)
```

References

1. Craig K. Enders. 2010. *Applied Missing Data Analysis*. Guilford Press. Retrieved from http://www.appliedmissingdata.com/