SPServer Analysis

Sahil Patel

7/15/2022

R Markdown

We must first load in the data from SPServer and convert the results to a binary 0 or 1, telling us whether the protein fluoresces or not. We then create the training/testing data split to verify our results

```
data <- read_excel("SPServerData.xlsx")
data <- data %>% mutate(Fluorescent = ifelse(`Median Brightness` < 3, 0, 1))

split1 <- sample(c(rep(0, 0.8 * nrow(data)), rep(1, 0.2 * nrow(data))))
split1 <- append(split1, 1)
train <- data[split1 == 0, ]
test <- data[split1 == 1, ]

x_test <- test %>% select(PAIR, ECOMB, ES3DC, ELOCAL, E3DC, E3D, ZPAIR, ZECOMB, ZES3DC, ZELOCAL, ZE3DC)
y_test <- test %>% select(Fluorescent)
glimpse(data)
```

```
## Rows: 39
## Columns: 15
## $ ID                    <chr> "high_brightness14", "high_brightness15", "high_~
## $ `Amino Acid Sequence` <chr> "REHMVLLEFATAAGIT", "RGHMVLLEFVTAAGIT", "RDHMVLL~
## $ `Median Brightness`   <dbl> 3.504147, 2.637185, 3.637395, 3.758577, 3.694205~
## $ PAIR                  <dbl> 318.99, 309.27, 307.27, 307.99, 307.06, 316.33, ~
## $ ECOMB                 <dbl> -3833.44, -3870.86, -3903.63, -3916.74, -3877.26~
## $ ES3DC                 <dbl> 133.90, 126.64, 125.92, 130.69, 127.90, 124.06, ~
## $ ELOCAL                <dbl> 9239.55, 9202.40, 9228.40, 9155.60, 9274.55, 928~
## $ E3DC                  <dbl> 93.71, 95.69, 88.25, 80.77, 91.29, 99.16, 95.39,~
## $ E3D                   <dbl> -13300.6, -13295.6, -13346.2, -13283.8, -13371.0~
## $ ZPAIR                 <dbl> -0.50, -0.86, -0.93, -0.88, -0.84, -0.83, -0.84,~
## $ ZECOMB                <dbl> -2.62, -2.69, -2.49, -2.27, -2.63, -2.42, -2.82,~
## $ ZES3DC                <dbl> -1.23, -1.48, -1.43, -1.39, -1.47, -1.89, -1.53,~
## $ ZELOCAL               <dbl> -2.16, -2.13, -1.97, -1.80, -2.06, -1.93, -2.25,~
## $ ZE3DC                 <dbl> -3.32, -4.02, -3.70, -3.49, -3.78, -3.57, -3.71,~
## $ Fluorescent           <dbl> 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, ~
```

Building a Model

Given that we are predicting flourescsence as a binary result, we use a logistic regression and start by using all of the given predictors

```
mylogit <- glm(Fluorescent ~ PAIR + ECOMB + ES3DC + ELOCAL+ E3DC + E3D + ZPAIR + ZECOMB + ZES3DC + ZELO
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(mylogit)
```

```
##
## Call:
## glm(formula = Fluorescent ~ PAIR + ECOMB + ES3DC + ELOCAL + E3DC +
##     E3D + ZPAIR + ZECOMB + ZES3DC + ZELOCAL + ZE3DC, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -8.743e-06   2.110e-08   2.110e-08   2.944e-06   8.164e-06
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.694e+03  2.082e+07       0        1
## PAIR        -1.071e+00  2.756e+04       0        1
## ECOMB       -2.207e+03  2.190e+07       0        1
## ES3DC        2.209e+03  2.188e+07       0        1
## ELOCAL       2.207e+03  2.190e+07       0        1
## E3DC         2.204e+03  2.190e+07       0        1
## E3D          2.206e+03  2.191e+07       0        1
## ZPAIR        3.474e+01  7.749e+05       0        1
## ZECOMB      -1.408e+02  5.966e+06       0        1
## ZES3DC      -6.612e+01  4.435e+05       0        1
## ZELOCAL      2.833e+01  6.428e+06       0        1
## ZE3DC        7.479e+01  5.524e+05       0        1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3.3118e+01  on 30  degrees of freedom
## Residual deviance: 4.1225e-10  on 19  degrees of freedom
## AIC: 24
##
## Number of Fisher Scoring iterations: 25
```

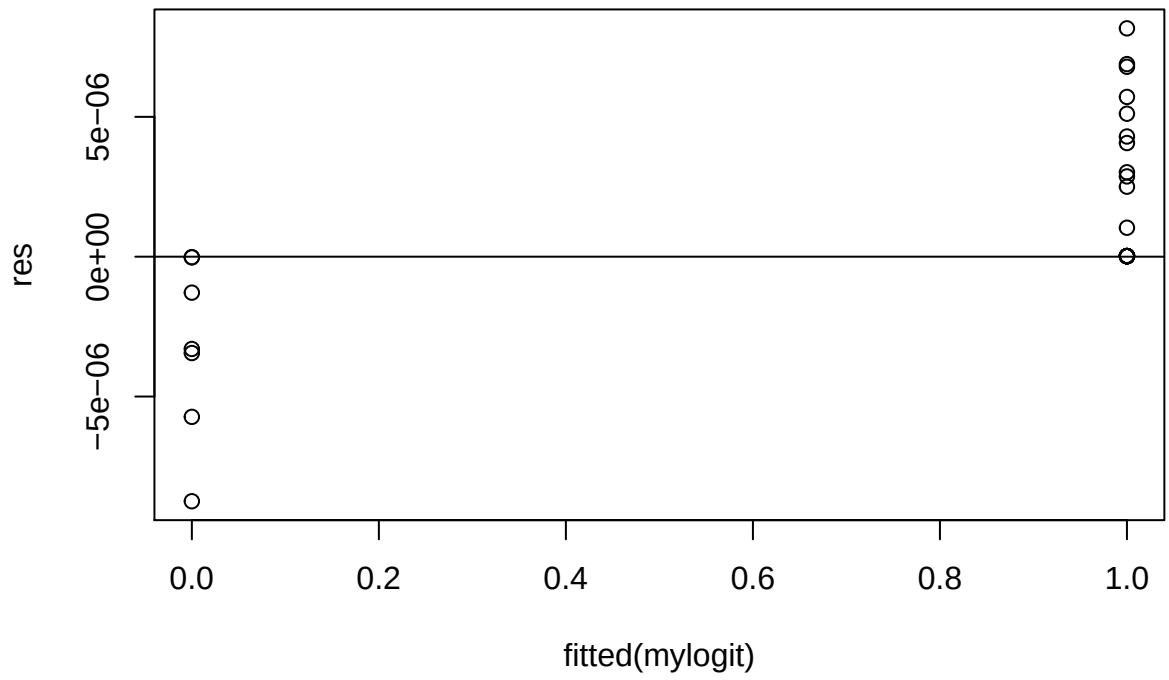
Residual Plot

Graphing the residual plot to see how well things worked (because there are no statistically significant variables)

```r
#get residuals
res <- resid(mylogit)

#produce residual vs. fitted plot
plot(fitted(mylogit), res)

#add a horizontal line at 0
abline(0,0)
```

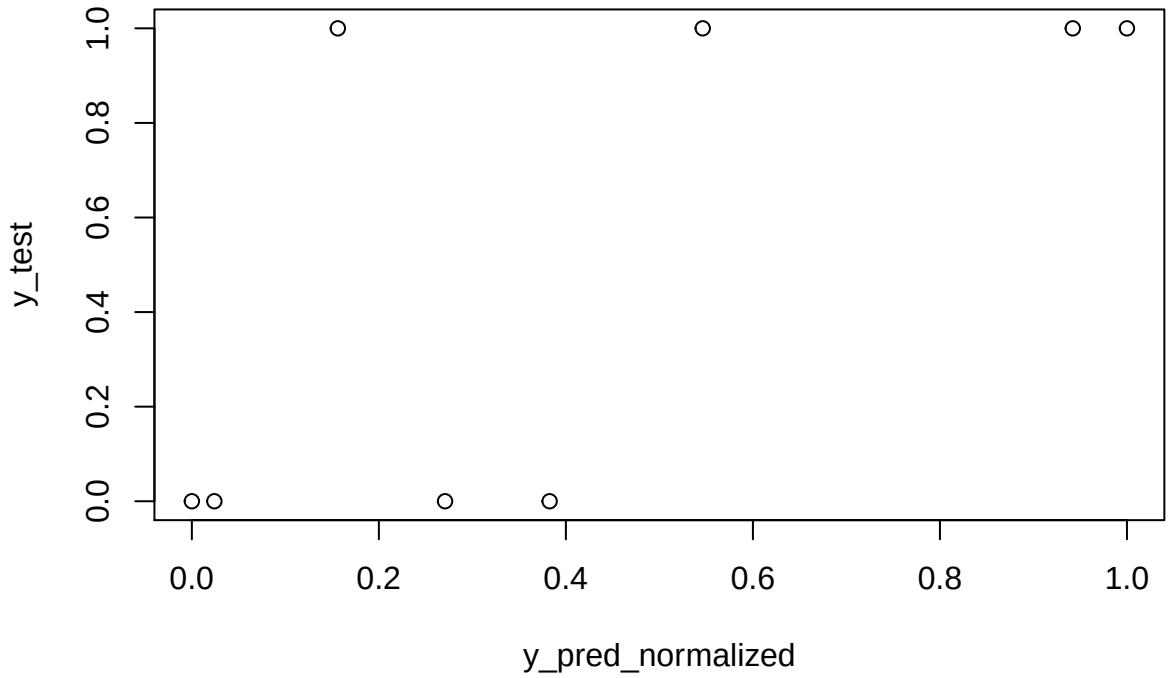


Testing the Model

To test the model, I am putting in all of the data and making a prediction and comparing it to the actual value

```
y_pred <- predict(mylogit, x_test)
y_pred_normalized <- (y_pred-min(y_pred))/(max(y_pred)-min(y_pred))
y_test <- y_test[[1]]

plot(y_pred_normalized, y_test)
```



By the prediction above, we notice that there is some classification error however the information can be quite useful in a machine learning setting.