Models and Deployment

Basketball Salaries Team

Load Primary Dataset

```
df_p.all <- read.csv('../data/pooled/primary.csv')
df_p.all$year <- as.factor(df_p.all$year)
str(df_p.all)
```

```
## 'data.frame':    733 obs. of  51 variables:
##  $ name  : Factor w/ 439 levels "aaron brooks",..: 1 1 2 2 3 4 5 5 6 6 ...
##  $ year  : Factor w/ 2 levels "2016","2017": 1 2 1 2 1 2 1 2 1 2 ...
##  $ salary: num  2700000 2116955 4351320 5504420 2022240 ...
##  $ Pos   : Factor w/ 5 levels "C","PF","PG",..: 3 3 2 4 2 1 4 4 1 1 ...
##  $ Age   : int  31 32 20 21 24 24 25 26 29 30 ...
##  $ Tm    : Factor w/ 31 levels "ATL","BOS","BRK",..: 4 12 22 22 18 7 25 25 1 2 ...
##  $ G     : int  69 65 78 80 52 22 82 61 82 68 ...
##  $ GS    : int  0 0 37 72 2 0 82 25 82 68 ...
##  $ MP    : int  1108 894 1863 2298 486 163 2341 1773 2631 2193 ...
##  $ PER   : num  11.8 9.5 17 14.4 5.6 8.4 12.7 11.3 19.4 17.7 ...
##  $ TS.   : num  0.494 0.507 0.541 0.53 0.422 0.472 0.533 0.506 0.565 0.553 ...
##  $ X3PAr : num  0.394 0.427 0.245 0.309 0.221 0.238 0.485 0.455 0.244 0.302 ...
##  $ FTr   : num  0.136 0.133 0.333 0.251 0.179 0.476 0.217 0.292 0.123 0.169 ...
##  $ ORB.  : num  2 2.3 9 5.3 4.8 5.4 4.5 4.8 6.3 4.9 ...
##  $ DRB.  : num  7.5 6.3 21.3 14.1 21.5 20.9 18.6 23.5 18.2 18.6 ...
##  $ TRB.  : num  4.8 4.3 15.1 9.6 13.3 12.8 11.5 14.1 12.4 11.8 ...
##  $ AST.  : num  26 20.7 10.3 10.5 8.9 3.8 8.8 7.9 16.7 24.4 ...
##  $ STL.  : num  1.4 1.4 1.6 1.4 1.7 0.3 1.5 1.7 1.3 1.2 ...
##  $ BLK.  : num  0.7 0.9 2.4 1.4 1.8 7.2 1.8 2 3.6 3.3 ...
##  $ TOV.  : num  14.2 17.2 9 8.5 18.7 16.4 13.2 15.2 8.8 11.9 ...
##  $ USG.  : num  22.9 19.2 17.3 20.1 17.7 17.6 16.9 15.4 20.6 19.8 ...
##  $ OWS   : num  0.2 -0.2 3.2 2 -0.9 -0.2 1.7 -0.1 4.9 3.6 ...
##  $ DWS   : num  0.7 0.5 2.2 1.7 0.4 0.2 2.3 2 4.5 2.7 ...
##  $ WS    : num  0.9 0.3 5.4 3.7 -0.5 0 4 1.9 9.4 6.3 ...
##  $ WS.48 : num  0.04 0.016 0.139 0.076 -0.047 -0.001 0.082 0.051 0.172 0.137 ...
##  $ OBPM  : num  -0.5 -2.1 0.6 -0.2 -5.9 -7.5 -0.4 -2.3 1.5 1 ...
##  $ DBPM  : num  -2.8 -2.6 1.2 -0.4 -0.2 1.9 0.7 1.2 2.6 2.1 ...
##  $ BPM   : num  -3.3 -4.6 1.8 -0.7 -6.1 -5.6 0.2 -1.1 4.1 3.1 ...
##  $ VORP  : num  -0.4 -0.6 1.8 0.8 -0.5 -0.1 1.3 0.4 4.1 2.8 ...
##  $ FG    : int  188 121 274 393 53 17 299 183 529 379 ...
##  $ FGA   : int  469 300 579 865 145 42 719 466 1048 801 ...
##  $ FG.   : num  0.401 0.403 0.473 0.454 0.366 0.405 0.416 0.393 0.505 0.473 ...
##  $ X3P   : int  66 48 42 77 9 5 126 70 88 86 ...
##  $ X3PA  : int  185 128 142 267 32 10 349 212 256 242 ...
##  $ X3P.  : num  0.357 0.375 0.296 0.288 0.281 0.5 0.361 0.33 0.344 0.355 ...
##  $ X2P   : int  122 73 232 316 44 12 173 113 441 293 ...
##  $ X2PA  : int  284 172 437 598 113 32 370 254 792 559 ...
##  $ X2P.  : num  0.43 0.424 0.531 0.528 0.389 0.375 0.468 0.445 0.557 0.524 ...
##  $ eFG.  : num  0.471 0.483 0.509 0.499 0.397 0.464 0.503 0.468 0.547 0.527 ...
##  $ FT    : int  49 32 129 156 17 9 115 96 103 108 ...
##  $ FTA   : int  64 40 193 217 26 20 156 136 129 135 ...
##  $ FT.   : num  0.766 0.8 0.668 0.719 0.654 0.45 0.737 0.706 0.798 0.8 ...
##  $ ORB   : int  21 18 154 116 20 8 98 77 148 95 ...
##  $ DRB   : int  80 51 353 289 91 28 401 374 448 369 ...
##  $ TRB   : int  101 69 507 405 111 36 499 451 596 464 ...
##  $ AST   : int  180 125 128 150 29 4 138 99 263 337 ...
```

```
##  $ STL   : int  30 25 59 64 16 1 72 60 68 52 ...
##  $ BLK   : int  10 9 55 40 11 13 53 44 121 87 ...
##  $ TOV   : int  82 66 66 89 36 10 120 94 107 116 ...
##  $ PF    : int  132 93 153 172 77 21 171 102 163 138 ...
##  $ PTS   : int  491 322 719 1019 132 48 839 532 1249 952 ...
```

```
head(df_p.all)
```

```
##            name year  salary Pos Age  Tm  G GS   MP  PER   TS. X3PAr   FTr ORB.
## 1  aaron brooks 2016 2700000  PG  31 CHI 69  0 1108 11.8 0.494 0.394 0.136  2.0
## 2  aaron brooks 2017 2116955  PG  32 IND 65  0  894  9.5 0.507 0.427 0.133  2.3
## 3  aaron gordon 2016 4351320  PF  20 ORL 78 37 1863 17.0 0.541 0.245 0.333  9.0
## 4  aaron gordon 2017 5504420  SF  21 ORL 80 72 2298 14.4 0.530 0.309 0.251  5.3
## 5 adreian payne 2016 2022240  PF  24 MIN 52  2  486  5.6 0.422 0.221 0.179  4.8
## 6    aj hammons 2017 1312611   C  24 DAL 22  0  163  8.4 0.472 0.238 0.476  5.4
##   DRB. TRB. AST. STL. BLK. TOV. USG.  OWS DWS   WS  WS.48 OBPM DBPM  BPM VORP
## 1  7.5  4.8 26.0  1.4  0.7 14.2 22.9  0.2 0.7  0.9  0.040 -0.5 -2.8 -3.3 -0.4
## 2  6.3  4.3 20.7  1.4  0.9 17.2 19.2 -0.2 0.5  0.3  0.016 -2.1 -2.6 -4.6 -0.6
## 3 21.3 15.1 10.3  1.6  2.4  9.0 17.3  3.2 2.2  5.4  0.139  0.6  1.2  1.8  1.8
## 4 14.1  9.6 10.5  1.4  1.4  8.5 20.1  2.0 1.7  3.7  0.076 -0.2 -0.4 -0.7  0.8
## 5 21.5 13.3  8.9  1.7  1.8 18.7 17.7 -0.9 0.4 -0.5 -0.047 -5.9 -0.2 -6.1 -0.5
## 6 20.9 12.8  3.8  0.3  7.2 16.4 17.6 -0.2 0.2  0.0 -0.001 -7.5  1.9 -5.6 -0.1
##    FG FGA   FG. X3P X3PA  X3P. X2P X2PA  X2P.  eFG.  FT FTA   FT. ORB DRB TRB
## 1 188 469 0.401  66  185 0.357 122  284 0.430 0.471  49  64 0.766  21  80 101
## 2 121 300 0.403  48  128 0.375  73  172 0.424 0.483  32  40 0.800  18  51  69
## 3 274 579 0.473  42  142 0.296 232  437 0.531 0.509 129 193 0.668 154 353 507
## 4 393 865 0.454  77  267 0.288 316  598 0.528 0.499 156 217 0.719 116 289 405
## 5  53 145 0.366   9   32 0.281  44  113 0.389 0.397  17  26 0.654  20  91 111
## 6  17  42 0.405   5   10 0.500  12   32 0.375 0.464   9  20 0.450   8  28  36
##   AST STL BLK TOV  PF  PTS
## 1 180  30  10  82 132  491
## 2 125  25   9  66  93  322
## 3 128  59  55  66 153  719
## 4 150  64  40  89 172 1019
## 5  29  16  11  36  77  132
## 6   4   1  13  10  21   48
```

Load Complete (Primary + Secondary) Dataset

```
df_c.all <- read.csv('../data/pooled/complete.csv')
df_c.all$year <- as.factor(df_c.all$year)
str(df_c.all)
```

```
## 'data.frame':    733 obs. of  58 variables:
##  $ name  : Factor w/ 439 levels "aaron brooks",..: 1 1 2 2 3 4 5 5 6 6 ...
##  $ year  : Factor w/ 2 levels "2016","2017": 1 2 1 2 1 2 1 2 1 2 ...
##  $ salary: num  2700000 2116955 4351320 5504420 2022240 ...
##  $ Pos   : Factor w/ 5 levels "C","PF","PG",..: 3 3 2 4 2 1 4 4 1 1 ...
##  $ Age   : int  31 32 20 21 24 24 25 26 29 30 ...
##  $ Tm    : Factor w/ 31 levels "ATL","BOS","BRK",..: 4 12 22 22 18 7 25 25 1 2 ...
##  $ G     : int  69 65 78 80 52 22 82 61 82 68 ...
##  $ GS    : int  0 0 37 72 2 0 82 25 82 68 ...
##  $ MP    : int  1108 894 1863 2298 486 163 2341 1773 2631 2193 ...
##  $ PER   : num  11.8 9.5 17 14.4 5.6 8.4 12.7 11.3 19.4 17.7 ...
##  $ TS.   : num  0.494 0.507 0.541 0.53 0.422 0.472 0.533 0.506 0.565 0.553 ...
##  $ X3PAr : num  0.394 0.427 0.245 0.309 0.221 0.238 0.485 0.455 0.244 0.302 ...
##  $ FTr   : num  0.136 0.133 0.333 0.251 0.179 0.476 0.217 0.292 0.123 0.169 ...
##  $ ORB.  : num  2 2.3 9 5.3 4.8 5.4 4.5 4.8 6.3 4.9 ...
##  $ DRB.  : num  7.5 6.3 21.3 14.1 21.5 20.9 18.6 23.5 18.2 18.6 ...
##  $ TRB.  : num  4.8 4.3 15.1 9.6 13.3 12.8 11.5 14.1 12.4 11.8 ...
##  $ AST.  : num  26 20.7 10.3 10.5 8.9 3.8 8.8 7.9 16.7 24.4 ...
##  $ STL.  : num  1.4 1.4 1.6 1.4 1.7 0.3 1.5 1.7 1.3 1.2 ...
##  $ BLK.  : num  0.7 0.9 2.4 1.4 1.8 7.2 1.8 2 3.6 3.3 ...
```

```
##  $ TOV.  : num  14.2 17.2 9 8.5 18.7 16.4 13.2 15.2 8.8 11.9 ...
##  $ USG.  : num  22.9 19.2 17.3 20.1 17.7 17.6 16.9 15.4 20.6 19.8 ...
##  $ OWS   : num  0.2 -0.2 3.2 2 -0.9 -0.2 1.7 -0.1 4.9 3.6 ...
##  $ DWS   : num  0.7 0.5 2.2 1.7 0.4 0.2 2.3 2 4.5 2.7 ...
##  $ WS    : num  0.9 0.3 5.4 3.7 -0.5 0 4 1.9 9.4 6.3 ...
##  $ WS.48 : num  0.04 0.016 0.139 0.076 -0.047 -0.001 0.082 0.051 0.172 0.137 ...
##  $ OBPM  : num  -0.5 -2.1 0.6 -0.2 -5.9 -7.5 -0.4 -2.3 1.5 1 ...
##  $ DBPM  : num  -2.8 -2.6 1.2 -0.4 -0.2 1.9 0.7 1.2 2.6 2.1 ...
##  $ BPM   : num  -3.3 -4.6 1.8 -0.7 -6.1 -5.6 0.2 -1.1 4.1 3.1 ...
##  $ VORP  : num  -0.4 -0.6 1.8 0.8 -0.5 -0.1 1.3 0.4 4.1 2.8 ...
##  $ FG    : int  188 121 274 393 53 17 299 183 529 379 ...
##  $ FGA   : int  469 300 579 865 145 42 719 466 1048 801 ...
##  $ FG.   : num  0.401 0.403 0.473 0.454 0.366 0.405 0.416 0.393 0.505 0.473 ...
##  $ X3P   : int  66 48 42 77 9 5 126 70 88 86 ...
##  $ X3PA  : int  185 128 142 267 32 10 349 212 256 242 ...
##  $ X3P.  : num  0.357 0.375 0.296 0.288 0.281 0.5 0.361 0.33 0.344 0.355 ...
##  $ X2P   : int  122 73 232 316 44 12 173 113 441 293 ...
##  $ X2PA  : int  284 172 437 598 113 32 370 254 792 559 ...
##  $ X2P.  : num  0.43 0.424 0.531 0.528 0.389 0.375 0.468 0.445 0.557 0.524 ...
##  $ eFG.  : num  0.471 0.483 0.509 0.499 0.397 0.464 0.503 0.468 0.547 0.527 ...
##  $ FT    : int  49 32 129 156 17 9 115 96 103 108 ...
##  $ FTA   : int  64 40 193 217 26 20 156 136 129 135 ...
##  $ FT.   : num  0.766 0.8 0.668 0.719 0.654 0.45 0.737 0.706 0.798 0.8 ...
##  $ ORB   : int  21 18 154 116 20 8 98 77 148 95 ...
##  $ DRB   : int  80 51 353 289 91 28 401 374 448 369 ...
##  $ TRB   : int  101 69 507 405 111 36 499 451 596 464 ...
##  $ AST   : int  180 125 128 150 29 4 138 99 263 337 ...
##  $ STL   : int  30 25 59 64 16 1 72 60 68 52 ...
##  $ BLK   : int  10 9 55 40 11 13 53 44 121 87 ...
##  $ TOV   : int  82 66 66 89 36 10 120 94 107 116 ...
##  $ PF    : int  132 93 153 172 77 21 171 102 163 138 ...
##  $ PTS   : int  491 322 719 1019 132 48 839 532 1249 952 ...
##  $ out   : int  79 87 87 86 56 47 90 75 81 80 ...
##  $ ovr   : int  75 85 90 92 69 66 91 83 83 91 ...
##  $ ins   : int  52 51 91 91 65 64 77 72 76 82 ...
##  $ pla   : int  74 81 69 49 43 40 60 59 58 82 ...
##  $ ath   : int  77 82 86 86 66 58 81 75 75 77 ...
##  $ def   : int  52 57 69 75 64 57 76 66 70 80 ...
##  $ reb   : int  36 37 87 94 68 71 94 65 73 87 ...
```

```
head(df_c.all)
```

```
##            name year  salary Pos Age  Tm  G GS   MP  PER   TS. X3PAr   FTr ORB.
## 1  aaron brooks 2016 2700000  PG  31 CHI 69  0 1108 11.8 0.494 0.394 0.136  2.0
## 2  aaron brooks 2017 2116955  PG  32 IND 65  0  894  9.5 0.507 0.427 0.133  2.3
## 3  aaron gordon 2016 4351320  PF  20 ORL 78 37 1863 17.0 0.541 0.245 0.333  9.0
## 4  aaron gordon 2017 5504420  SF  21 ORL 80 72 2298 14.4 0.530 0.309 0.251  5.3
## 5 adreian payne 2016 2022240  PF  24 MIN 52  2  486  5.6 0.422 0.221 0.179  4.8
## 6    aj hammons 2017 1312611   C  24 DAL 22  0  163  8.4 0.472 0.238 0.476  5.4
##   DRB. TRB. AST. STL. BLK. TOV. USG.  OWS DWS   WS  WS.48 OBPM DBPM  BPM VORP
## 1  7.5  4.8 26.0  1.4  0.7 14.2 22.9  0.2 0.7  0.9  0.040 -0.5 -2.8 -3.3 -0.4
## 2  6.3  4.3 20.7  1.4  0.9 17.2 19.2 -0.2 0.5  0.3  0.016 -2.1 -2.6 -4.6 -0.6
## 3 21.3 15.1 10.3  1.6  2.4  9.0 17.3  3.2 2.2  5.4  0.139  0.6  1.2  1.8  1.8
## 4 14.1  9.6 10.5  1.4  1.4  8.5 20.1  2.0 1.7  3.7  0.076 -0.2 -0.4 -0.7  0.8
## 5 21.5 13.3  8.9  1.7  1.8 18.7 17.7 -0.9 0.4 -0.5 -0.047 -5.9 -0.2 -6.1 -0.5
## 6 20.9 12.8  3.8  0.3  7.2 16.4 17.6 -0.2 0.2  0.0 -0.001 -7.5  1.9 -5.6 -0.1
##    FG FGA   FG. X3P X3PA  X3P. X2P X2PA  X2P.  eFG.  FT FTA   FT. ORB DRB TRB
## 1 188 469 0.401  66  185 0.357 122  284 0.430 0.471  49  64 0.766  21  80 101
## 2 121 300 0.403  48  128 0.375  73  172 0.424 0.483  32  40 0.800  18  51  69
## 3 274 579 0.473  42  142 0.296 232  437 0.531 0.509 129 193 0.668 154 353 507
## 4 393 865 0.454  77  267 0.288 316  598 0.528 0.499 156 217 0.719 116 289 405
## 5  53 145 0.366   9   32 0.281  44  113 0.389 0.397  17  26 0.654  20  91 111
## 6  17  42 0.405   5   10 0.500  12   32 0.375 0.464   9  20 0.450   8  28  36
##   AST STL BLK TOV  PF  PTS out ovr ins pla ath def reb
```

```
## 1 180  30  10  82 132  491  79  75  52  74  77  52  36
## 2 125  25   9  66  93  322  87  85  51  81  82  57  37
## 3 128  59  55  66 153  719  87  90  91  69  86  69  87
## 4 150  64  40  89 172 1019  86  92  91  49  86  75  94
## 5  29  16  11  36  77  132  56  69  65  43  66  64  68
## 6   4   1  13  10  21   48  47  66  64  40  58  57  71
```

Split Primary & Complete Datasets into Train Test

```
library(caret)
set.seed(7)
# primary dataset
train_rows.p <- createDataPartition(y=df_p.all[,'salary'], list=FALSE, p=.8)
df_p.train <- df_p.all[train_rows.p,]
df_p.test <- df_p.all[-train_rows.p,]
nrow(df_p.all)
```

```
## [1] 733
```

```
nrow(df_p.train)
```

```
## [1] 588
```

```
nrow(df_p.test)
```

```
## [1] 145
```

```
# complete dataset
train_rows.c <- createDataPartition(y=df_c.all[,'salary'], list=FALSE, p=.8)
df_c.train <- df_c.all[train_rows.c,]
df_c.test <- df_c.all[-train_rows.c,]
nrow(df_c.all)
```

```
## [1] 733
```

```
nrow(df_c.train)
```

```
## [1] 588
```

```
nrow(df_c.test)
```

```
## [1] 145
```

Load Train/Test Datasets Resulting from Variable Selection

```
df_p_vs.train = read.csv('../data/train_test/primary/train_selected.csv')
df_p_vs.train$year <- as.factor(df_p_vs.train$year)
df_p_vs.test = read.csv('../data/train_test/primary/test_selected.csv')
df_p_vs.test$year <- as.factor(df_p_vs.test$year)
df_c_vs.train = read.csv('../data/train_test/complete/train_selected.csv')
df_c_vs.test = read.csv('../data/train_test/complete/test_selected.csv')
```

Modeling Helper Functions

```
r_squared <- function(y,yHat){1-sum((y-yHat)^2)/sum((y-mean(y))^2)}
mse <- function(y,yHat){mean((y-yHat)^2)}
model_results <- function(model,dataset,y,yHat){
  r2_test <- r_squared(y,yHat)
  mse_test <- mse(y,yHat)
  cat(sprintf('Model: %-25s Dataset: %-15s R^2 Test: %-10.3f MSE: %-10.3e\n',model,dataset,r2_test,mse_test))}
```

Simple Linear Regression Models

```
# modeling function
slr_modeling <- function(dataset,df_train,df_test){
  model <- 'SLR'
  x_vars <- names(df_train)[!(names(df_train)%in%c('name','salary','X2P','X2PA','TRB','PTS'))]
  f <- as.formula(sprintf('salary ~ `%s`',paste(x_vars,collapse='` + `')))
  slr_model <- lm(f,data=df_train)
  yhat <- predict(slr_model,df_test)
  model_results(model,dataset,df_test[['salary']],yhat)
  return(slr_model)}
# train/test Simple Linear Regression models
names(df_p.train)
```

```
##  [1] "name"   "year"   "salary" "Pos"    "Age"    "Tm"     "G"      "GS"
##  [9] "MP"     "PER"    "TS."    "X3PAr"  "FTr"    "ORB."   "DRB."   "TRB."
## [17] "AST."   "STL."   "BLK."   "TOV."   "USG."   "OWS"    "DWS"    "WS"
## [25] "WS.48"  "OBPM"   "DBPM"   "BPM"    "VORP"   "FG"     "FGA"    "FG."
## [33] "X3P"    "X3PA"   "X3P."   "X2P"    "X2PA"   "X2P."   "eFG."   "FT"
## [41] "FTA"    "FT."    "ORB"    "DRB"    "TRB"    "AST"    "STL"    "BLK"
## [49] "TOV"    "PF"     "PTS"
```

```
ignore <- slr_modeling('primary',df_p.train,df_p.test)
```

```
## Model: SLR                       Dataset: primary         R^2 Test: 0.478      MSE: 2.633e+13
```

```
ignore <- slr_modeling('complete',df_c.train,df_c.test)
```

```
## Model: SLR                       Dataset: complete        R^2 Test: 0.544      MSE: 2.329e+13
```

```
summary(ignore)
```

```
##
## Call:
## lm(formula = f, data = df_train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -13351794  -2556875    -76052   2494544  13719401
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.081e+07  9.273e+06   1.165  0.24449
## year2017     7.657e+05  6.065e+05   1.263  0.20731
## PosPF        1.694e+05  8.192e+05   0.207  0.83627
## PosPG       -4.181e+06  1.555e+06  -2.689  0.00740 **
## PosSF       -9.588e+05  1.134e+06  -0.846  0.39804
## PosSG       -2.810e+06  1.334e+06  -2.107  0.03559 *
## Age          1.640e+05  5.997e+04   2.735  0.00646 **
## TmBOS        3.611e+05  1.861e+06   0.194  0.84627
## TmBRK       -9.970e+05  2.212e+06  -0.451  0.65235
## TmCHI       -3.774e+05  1.906e+06  -0.198  0.84315
## TmCHO       -1.204e+06  1.918e+06  -0.628  0.53023
## TmCLE        1.624e+06  1.951e+06   0.832  0.40561
## TmDAL        1.546e+04  1.963e+06   0.008  0.99372
## TmDEN       -1.593e+06  2.090e+06  -0.762  0.44627
## TmDET       -4.683e+05  1.945e+06  -0.241  0.80981
## TmGSW       -3.516e+05  1.868e+06  -0.188  0.85078
## TmHOU       -9.426e+05  2.029e+06  -0.464  0.64250
## TmIND       -1.617e+06  1.793e+06  -0.902  0.36739
## TmLAC        9.555e+05  1.873e+06   0.510  0.61018
## TmLAL        8.175e+05  2.461e+06   0.332  0.73991
## TmMEM        1.235e+06  2.007e+06   0.615  0.53867
## TmMIA       -1.012e+05  1.825e+06  -0.055  0.95581
## TmMIL        1.400e+06  1.986e+06   0.705  0.48107
## TmMIN       -1.231e+06  2.211e+06  -0.557  0.57804
```

```
## TmNOP        1.690e+06  1.981e+06   0.853  0.39400
## TmNYK       -2.602e+05  2.048e+06  -0.127  0.89893
## TmOKC        1.457e+06  2.006e+06   0.726  0.46801
## TmORL        6.014e+05  1.987e+06   0.303  0.76223
## TmPHI       -2.120e+06  2.091e+06  -1.014  0.31115
## TmPHO        1.148e+06  2.122e+06   0.541  0.58877
## TmPOR        3.375e+06  2.057e+06   1.641  0.10143
## TmSAC       -1.696e+05  2.000e+06  -0.085  0.93242
## TmSAS       -2.162e+06  1.933e+06  -1.119  0.26381
## TmTOR        4.821e+04  1.965e+06   0.025  0.98044
## TmTOT       -1.440e+06  1.680e+06  -0.857  0.39181
## TmUTA       -2.242e+06  1.918e+06  -1.169  0.24299
## TmWAS        1.740e+06  1.972e+06   0.882  0.37814
## G           -7.875e+04  2.589e+04  -3.042  0.00248 **
## GS           1.938e+04  1.382e+04   1.402  0.16161
## MP           3.300e+03  2.045e+03   1.614  0.10723
## PER         -2.214e+05  4.932e+05  -0.449  0.65371
## TS.         -1.466e+07  3.049e+07  -0.481  0.63086
## X3PAr       -7.174e+06  8.729e+06  -0.822  0.41151
## FTr         -4.377e+06  3.646e+06  -1.201  0.23044
## ORB.         2.176e+04  1.126e+06   0.019  0.98459
## DRB.         2.911e+05  1.094e+06   0.266  0.79036
## TRB.        -5.060e+05  2.208e+06  -0.229  0.81885
## AST.        -2.894e+04  1.111e+05  -0.261  0.79456
## STL.         6.052e+05  6.618e+05   0.914  0.36091
## BLK.         4.543e+05  5.246e+05   0.866  0.38691
## TOV.         2.509e+05  1.188e+05   2.112  0.03515 *
## USG.         2.470e+04  2.115e+05   0.117  0.90707
## OWS          1.948e+06  4.123e+06   0.473  0.63675
## DWS          5.164e+06  4.166e+06   1.240  0.21568
## WS          -1.848e+06  4.102e+06  -0.451  0.65249
## WS.48        3.890e+06  2.779e+07   0.140  0.88876
## OBPM        -1.821e+06  4.254e+06  -0.428  0.66880
## DBPM        -3.785e+06  4.177e+06  -0.906  0.36533
## BPM          3.305e+06  4.174e+06   0.792  0.42883
## VORP        -1.273e+06  6.855e+05  -1.857  0.06388 .
## FG           2.959e+04  3.546e+04   0.834  0.40441
## FGA         -1.461e+04  1.745e+04  -0.837  0.40279
## FG.         -6.822e+06  5.062e+07  -0.135  0.89286
## X3P         -1.417e+04  4.554e+04  -0.311  0.75580
## X3PA         1.529e+04  1.853e+04   0.825  0.40972
## X3P.         2.805e+05  2.376e+06   0.118  0.90608
## X2P.        -2.833e+06  8.378e+06  -0.338  0.73543
## eFG.        -2.713e+06  4.889e+07  -0.055  0.95577
## FT          -6.212e+03  2.869e+04  -0.216  0.82870
## FTA          2.189e+04  1.723e+04   1.271  0.20440
## FT.          3.144e+06  3.404e+06   0.923  0.35623
## ORB          3.967e+02  1.590e+04   0.025  0.98011
## DRB          3.286e+03  7.486e+03   0.439  0.66093
## AST          1.329e+04  1.069e+04   1.244  0.21415
## STL         -5.551e+04  2.165e+04  -2.565  0.01062 *
## BLK         -1.774e+04  2.125e+04  -0.835  0.40410
## TOV         -2.628e+04  2.743e+04  -0.958  0.33858
## PF          -2.587e+04  1.022e+04  -2.531  0.01168 *
## out         -1.313e+05  4.213e+04  -3.116  0.00194 **
## ovr          2.557e+05  9.240e+04   2.767  0.00586 **
## ins         -1.512e+03  5.116e+04  -0.030  0.97643
## pla         -1.550e+04  3.315e+04  -0.467  0.64036
## ath         -3.101e+04  5.035e+04  -0.616  0.53823
## def          2.031e+04  4.221e+04   0.481  0.63065
## reb         -5.435e+04  2.815e+04  -1.930  0.05412 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 4604000 on 503 degrees of freedom
## Multiple R-squared:  0.6613, Adjusted R-squared:  0.6048
## F-statistic: 11.69 on 84 and 503 DF,  p-value: < 2.2e-16
```

```
names(df_p_vs.train)
```

```
##  [1] "name"   "salary" "year"   "Pos"    "Age"    "Tm"     "GS"     "TS."
##  [9] "AST."   "WS"     "VORP"   "FG."    "X3P"    "FT"     "PF"
```

```
ignore <- slr_modeling('primary VS',df_p_vs.train,df_p_vs.test) # primary variable subset
```

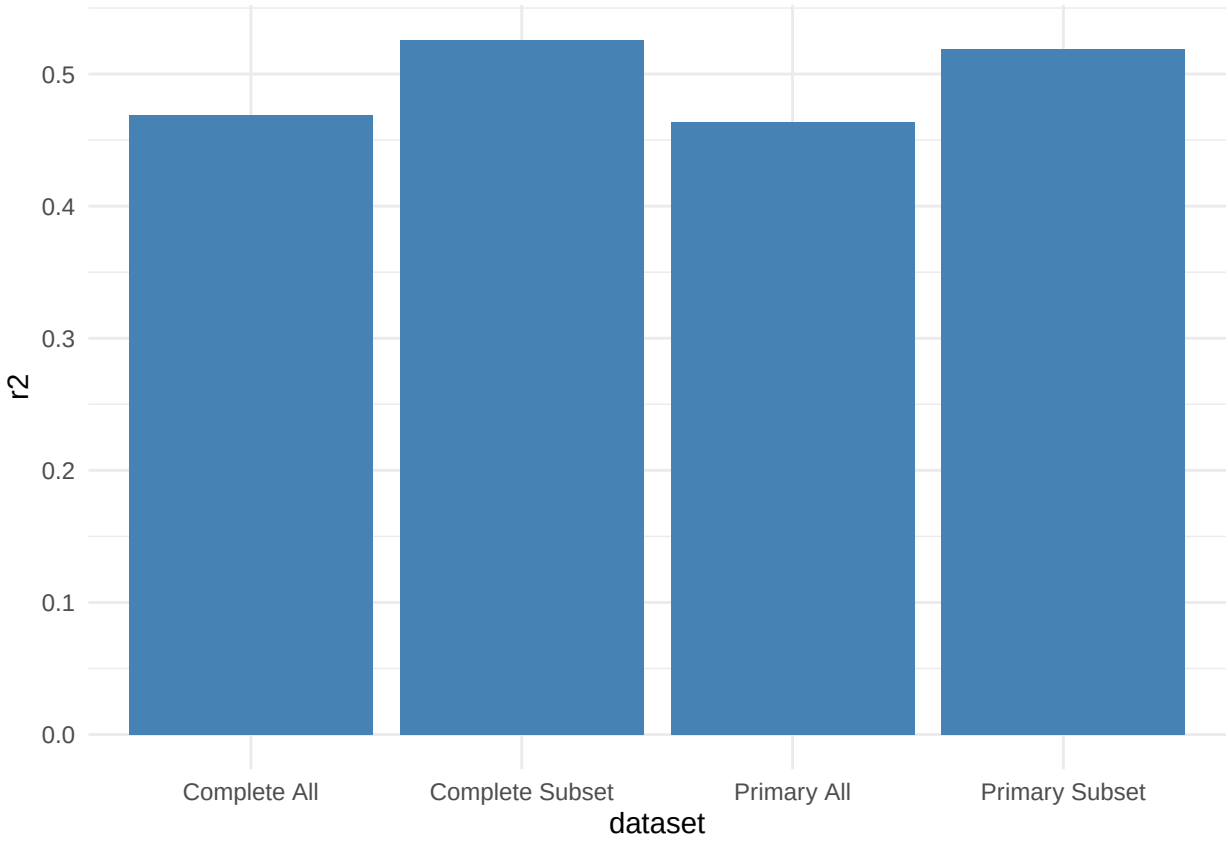
```
## Model: SLR                       Dataset: primary VS      R^2 Test: 0.440      MSE: 2.825e+13
```

```
ignore <- slr_modeling('complete VS',df_c_vs.train,df_c_vs.test) # complete variable subset
```

```
## Model: SLR                       Dataset: complete VS     R^2 Test: 0.472      MSE: 2.664e+13
```

SLR Plot R^2

```
library(ggplot2)
r2_data <- as.data.frame(list(
  dataset = c('Primary All','Complete All','Primary Subset','Complete Subset'),
  r2 =      c(.464,         .469,            .519,          .526)))
ggplot(data=r2_data, aes(x=dataset, y=r2)) +
  geom_bar(stat="identity", fill="steelblue") +
  theme_minimal()
```

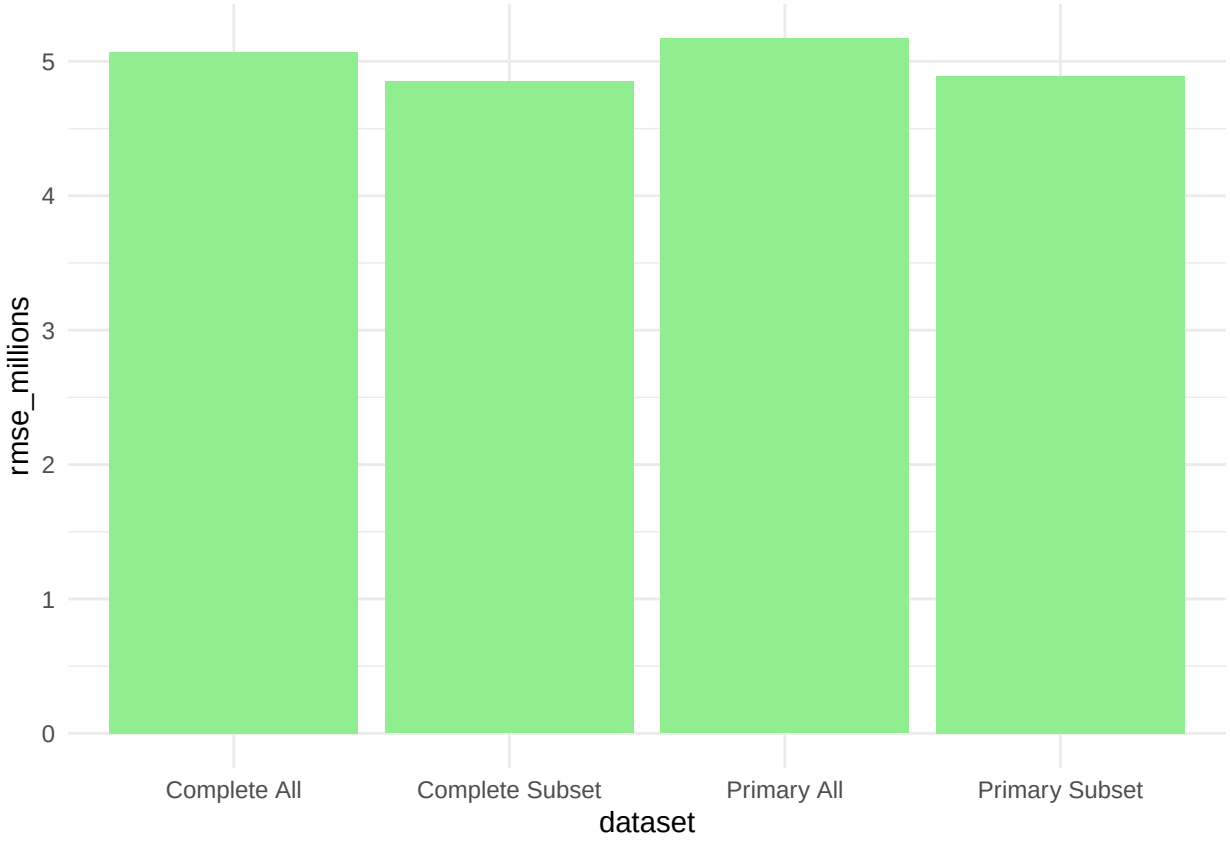



```
ggsave("../figures/slr_r2_bar.png");
```

SLR Plot RMSE

```
library(ggplot2)
rmse_data <- as.data.frame(list(
  dataset = c('Primary All','Complete All','Primary Subset','Complete Subset'),
  rmse_millions =      c(5.17,         5.07,          4.89,          4.85)))
ggplot(data=rmse_data, aes(x=dataset, y=rmse_millions)) +
```

```
  geom_bar(stat="identity", fill="lightgreen") +
  theme_minimal()
```




```
ggsave("../figures/slr_rmse_bar.png");
```

Lasso, Ridge, and Elastic Net Models with 10-fold Cross validation for alpha = seq(0,1,by=.05)

```
library(glmnet)
# modeling function
lre_modeling <- function(dataset,x_train,y_train,x_test,y_test,alphas,mkplot){
  # fit models
  for (i in alphas){
    set.seed(7) # seed for reproducibility
    model_name <- sprintf('fit_alpha_%.2f',i)
    assign(model_name, cv.glmnet(x_train, y_train, type.measure="mse",alpha=i,family="gaussian"))
    model <- get(model_name)
    yhat <- predict(model,s=model$lambda.min,newx=x_test)
    model_results(model_name,dataset,y_test,yhat)
    # plot
    if(mkplot){
      path = sprintf("../figures/elasticnet_models/alpha_%.2f.%s.png",i,dataset)
      png(file=path)
      par(mfrow=c(2,1))
      glmnet_model <- glmnet(x_train, y_train, family="gaussian",alpha=i)
      plot(glmnet_model)
      title(sprintf('Elasticnet Model, %s dataset, alpha = %.2f',dataset,i),line=3)
      plot(model,xvar='lambda')
      dev.off()}}
  return(model)} # return final model created
# extract train/test datasets of only numeric variables as required by glmnet models
#    primary dataset
numeric_vars.p <- names(Filter(is.numeric,df_p.train))
numeric_x_vars.p <- numeric_vars.p[!(numeric_vars.p%in%c('salary'))]
x_train.p <- data.matrix(df_p.train[,numeric_x_vars.p])
y_train.p <- df_p.train[['salary']]
```

```
x_test.p <- data.matrix(df_p.test[,numeric_x_vars.p])
y_test.p <- df_p.test[['salary']]
#    complete dataset
numeric_vars.c <- names(Filter(is.numeric,df_c.train))
numeric_x_vars.c <- numeric_vars.c[!(numeric_vars.c%in%c('salary'))]
x_train.c <- data.matrix(df_c.train[,numeric_x_vars.c])
y_train.c <- df_c.train[['salary']]
x_test.c <- data.matrix(df_c.test[,numeric_x_vars.c])
y_test.c <- df_c.test[['salary']]
# train/test Simple Linear Regression models
mkplots <- FALSE # change to TRUE if you want to generate plots
ignore <- lre_modeling('primary',x_train.p,y_train.p,x_test.p,y_test.p,seq(0,1,by=.05),mkplots)
```

```
## Model: fit_alpha_0.00           Dataset: primary         R^2 Test: 0.471      MSE: 2.668e+13
## Model: fit_alpha_0.05           Dataset: primary         R^2 Test: 0.470      MSE: 2.673e+13
## Model: fit_alpha_0.10           Dataset: primary         R^2 Test: 0.471      MSE: 2.672e+13
## Model: fit_alpha_0.15           Dataset: primary         R^2 Test: 0.470      MSE: 2.672e+13
## Model: fit_alpha_0.20           Dataset: primary         R^2 Test: 0.470      MSE: 2.673e+13
## Model: fit_alpha_0.25           Dataset: primary         R^2 Test: 0.470      MSE: 2.673e+13
## Model: fit_alpha_0.30           Dataset: primary         R^2 Test: 0.470      MSE: 2.673e+13
## Model: fit_alpha_0.35           Dataset: primary         R^2 Test: 0.470      MSE: 2.672e+13
## Model: fit_alpha_0.40           Dataset: primary         R^2 Test: 0.470      MSE: 2.672e+13
## Model: fit_alpha_0.45           Dataset: primary         R^2 Test: 0.469      MSE: 2.681e+13
## Model: fit_alpha_0.50           Dataset: primary         R^2 Test: 0.468      MSE: 2.683e+13
## Model: fit_alpha_0.55           Dataset: primary         R^2 Test: 0.468      MSE: 2.685e+13
## Model: fit_alpha_0.60           Dataset: primary         R^2 Test: 0.468      MSE: 2.687e+13
## Model: fit_alpha_0.65           Dataset: primary         R^2 Test: 0.467      MSE: 2.687e+13
## Model: fit_alpha_0.70           Dataset: primary         R^2 Test: 0.467      MSE: 2.688e+13
## Model: fit_alpha_0.75           Dataset: primary         R^2 Test: 0.468      MSE: 2.686e+13
## Model: fit_alpha_0.80           Dataset: primary         R^2 Test: 0.468      MSE: 2.686e+13
## Model: fit_alpha_0.85           Dataset: primary         R^2 Test: 0.467      MSE: 2.687e+13
## Model: fit_alpha_0.90           Dataset: primary         R^2 Test: 0.467      MSE: 2.688e+13
## Model: fit_alpha_0.95           Dataset: primary         R^2 Test: 0.467      MSE: 2.688e+13
## Model: fit_alpha_1.00           Dataset: primary         R^2 Test: 0.467      MSE: 2.688e+13
```

```
ignore <- lre_modeling('complete',x_train.c,y_train.c,x_test.c,y_test.c,seq(0,1,by=.05),mkplots)
```

```
## Model: fit_alpha_0.00           Dataset: complete        R^2 Test: 0.619      MSE: 1.947e+13
## Model: fit_alpha_0.05           Dataset: complete        R^2 Test: 0.613      MSE: 1.977e+13
## Model: fit_alpha_0.10           Dataset: complete        R^2 Test: 0.614      MSE: 1.974e+13
## Model: fit_alpha_0.15           Dataset: complete        R^2 Test: 0.615      MSE: 1.967e+13
## Model: fit_alpha_0.20           Dataset: complete        R^2 Test: 0.609      MSE: 1.997e+13
## Model: fit_alpha_0.25           Dataset: complete        R^2 Test: 0.606      MSE: 2.011e+13
## Model: fit_alpha_0.30           Dataset: complete        R^2 Test: 0.602      MSE: 2.034e+13
## Model: fit_alpha_0.35           Dataset: complete        R^2 Test: 0.603      MSE: 2.029e+13
## Model: fit_alpha_0.40           Dataset: complete        R^2 Test: 0.601      MSE: 2.036e+13
## Model: fit_alpha_0.45           Dataset: complete        R^2 Test: 0.600      MSE: 2.041e+13
## Model: fit_alpha_0.50           Dataset: complete        R^2 Test: 0.600      MSE: 2.045e+13
## Model: fit_alpha_0.55           Dataset: complete        R^2 Test: 0.599      MSE: 2.049e+13
## Model: fit_alpha_0.60           Dataset: complete        R^2 Test: 0.599      MSE: 2.051e+13
## Model: fit_alpha_0.65           Dataset: complete        R^2 Test: 0.598      MSE: 2.053e+13
## Model: fit_alpha_0.70           Dataset: complete        R^2 Test: 0.597      MSE: 2.057e+13
## Model: fit_alpha_0.75           Dataset: complete        R^2 Test: 0.599      MSE: 2.047e+13
## Model: fit_alpha_0.80           Dataset: complete        R^2 Test: 0.599      MSE: 2.049e+13
## Model: fit_alpha_0.85           Dataset: complete        R^2 Test: 0.598      MSE: 2.051e+13
## Model: fit_alpha_0.90           Dataset: complete        R^2 Test: 0.598      MSE: 2.053e+13
## Model: fit_alpha_0.95           Dataset: complete        R^2 Test: 0.598      MSE: 2.053e+13
## Model: fit_alpha_1.00           Dataset: complete        R^2 Test: 0.598      MSE: 2.054e+13
```

Save Optimal Model to File

Optimal model with largest R^2 Test and smallest MSE is SLR on Complete Backwards Selected Variables

```
numeric_x_vars.c
```

```
##  [1] "Age"   "G"     "GS"    "MP"    "PER"   "TS."   "X3PAr" "FTr"   "ORB."
## [10] "DRB."  "TRB."  "AST."  "STL."  "BLK."  "TOV."  "USG."  "OWS"   "DWS"
## [19] "WS"    "WS.48" "OBPM"  "DBPM"  "BPM"   "VORP"  "FG"    "FGA"   "FG."
## [28] "X3P"   "X3PA"  "X3P."  "X2P"   "X2PA"  "X2P."  "eFG."  "FT"    "FTA"
## [37] "FT."   "ORB"   "DRB"   "TRB"   "AST"   "STL"   "BLK"   "TOV"   "PF"
## [46] "PTS"   "out"   "ovr"   "ins"   "pla"   "ath"   "def"   "reb"
```

```
optimal_model <- lre_modeling('complete',x_train.c,y_train.c,x_test.c,y_test.c,c(0),mkplots)
```

```
## Model: fit_alpha_0.00            Dataset: complete        R^2 Test: 0.619      MSE: 1.947e+13
```

```
optimal_model$lambda.min
```

```
## [1] 501360.3
```

```
saveRDS(optimal_model,file='../data/optimal_model/elasticnet/model.rds')
```

Deployment of Optimal Model

```
library(plumber)
paste(
  "curl -X POST 'http://localhost:8000/predict_salary?",
  "Age=31&G=76&GS=76&MP=2709&PER=27.5&TS.=.588&X3PAr=0.199&",
  "FTr=0.347&ORB.=4.7&DRB.=18.8&TRB.=11.8&AST.=36.0&STL.=2.0&",
  "BLK.=1.5&TOV.=13.2&USG.=31.4&OWS=9.6&DWS=4&WS=13.6&",
  "WS.48=0.242&OBPM=6.9&DBPM=2.3&BPM=9.1&VORP=7.6&FG=737&",
  "FGA=1416&FG.=0.520&X3P=87&X3PA=282&X3P.=0.309&X2P=650&",
  "X2PA=1134&X2P.=0.573&eFG.=0.551&FT=359&FTA=491&FT.=0.731&",
  "ORB=111&DRB=454&TRB=565&AST=514&STL=104&BLK=49&TOV=249&",
  "PF=143&PTS=1920&out=94&ovr=99&ins=89&pla=91&ath=92&def=91&reb=91'",
  sep='')
r <- plumb("./deploy_optimal_model.R")
r$run(port=8000)
```

Prediction from Optimal Model

```
df_c.train[df_c.train$name=='lebron james'&df_c.train$year=='2016',]
```

```
##             name year   salary Pos Age  Tm  G GS   MP  PER   TS. X3PAr   FTr
## 440 lebron james 2016 30963450  SF  31 CLE 76 76 2709 27.5 0.588 0.199 0.347
##     ORB. DRB. TRB. AST. STL. BLK. TOV. USG. OWS DWS   WS WS.48 OBPM DBPM BPM
## 440  4.7 18.8 11.8   36    2  1.5 13.2 31.4 9.6   4 13.6 0.242  6.9  2.3 9.1
##     VORP  FG  FGA  FG. X3P X3PA  X3P. X2P X2PA  X2P.  eFG.  FT FTA   FT. ORB
## 440  7.6 737 1416 0.52  87  282 0.309 650 1134 0.573 0.551 359 491 0.731 111
##     DRB TRB AST STL BLK TOV  PF  PTS out ovr ins pla ath def reb
## 440 454 565 514 104  49 249 143 1920  94  99  89  91  92  91  91
```

```
df_c.all.copy <- df_c.all
x <- data.matrix(df_c.all[,numeric_x_vars.c])
df_c.all.copy$salary_hat_elasticnet <- as.vector(predict(optimal_model,s=optimal_model$lambda.min,newx=x))
df_c.all.copy <- df_c.all.copy[,c('name','year','salary','salary_hat_elasticnet')]
names(df_c.all.copy) <- c('name','year','salary','salary_hat')
head(df_c.all.copy)
```

```
##            name year  salary salary_hat
## 1  aaron brooks 2016 2700000    3730156
## 2  aaron brooks 2017 2116955    3179791
## 3  aaron gordon 2016 4351320    8286105
## 4  aaron gordon 2017 5504420   10716914
## 5 adreian payne 2016 2022240    1125380
## 6    aj hammons 2017 1312611    1495870
```

```
write.csv(df_c.all.copy,'../data/predictions/elasticnet.csv',row.names=F)
```