Lab #6 Report

Nick Huo & Severin Johnson

2022-11-10

Question 1

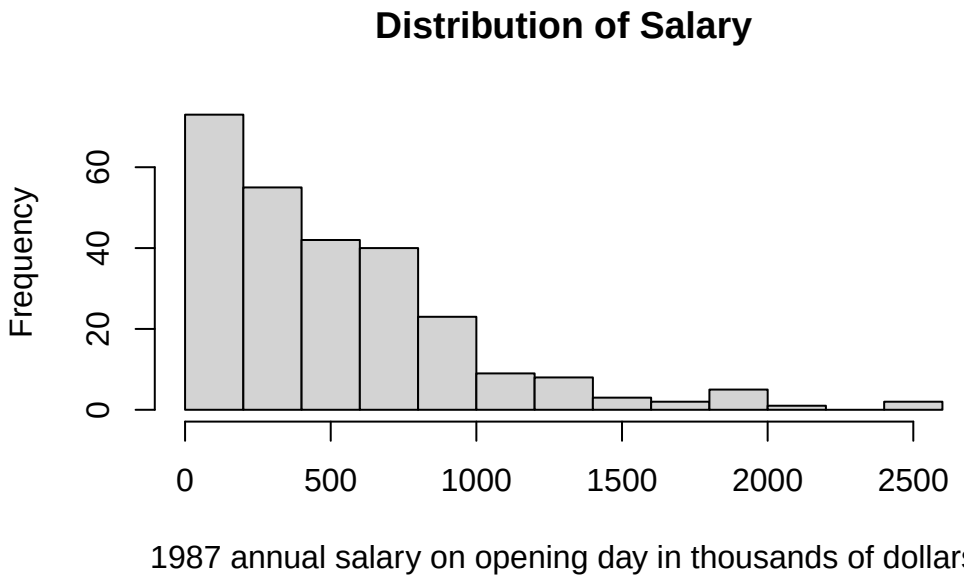
1. Remove incomplete rows and categorical variables

```
library(ISLR)
data(Hitters)

Hitters <- na.omit(Hitters)
cat.cols <- c("League","Division","NewLeague")
Hitters <- Hitters[ ,-which(colnames(Hitters)%in%cat.cols)]
attach(Hitters)
```

2. Distribution of Salary

```
hist(Salary, main="Distribution of Salary",
     xlab="1987 annual salary on opening day in thousands of dollars")
```



The distribution looks like exponential with a right skew (tail). It means that most people get relatively low salaries while a few gets really high salaries. It makes sense because we would expect that there are a few sport stars that get paid a lot more than others.

3. Model 1: Full Model – Salary

```
model1 <- lm(Salary ~ ., data=Hitters)
summary(model1)
```

```
##
## Call:
## lm(formula = Salary ~ ., data = Hitters)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -982.81 -187.84  -35.66  130.61 1947.43
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 126.10553   83.62448   1.508 0.132838
## AtBat        -2.20302    0.63605  -3.464 0.000629 ***
## Hits          7.82776    2.40198   3.259 0.001276 **
## HmRun         2.16355    6.23618   0.347 0.728937
## Runs         -2.09957    3.00849  -0.698 0.485911
## RBI          -0.02292    2.61033  -0.009 0.993003
## Walks         6.15106    1.84028   3.342 0.000960 ***
## Years        -2.59237   12.45401  -0.208 0.835280
## CAtBat       -0.17628    0.13667  -1.290 0.198325
## CHits         0.06976    0.67874   0.103 0.918221
## CHmRun       -0.23309    1.63561  -0.143 0.886795
## CRuns         1.61005    0.75162   2.142 0.033168 *
## CRBI          0.80143    0.70000   1.145 0.253367
## CWalks       -0.79394    0.33243  -2.388 0.017681 *
## PutOuts       0.29457    0.07830   3.762 0.000211 ***
## Assists       0.38400    0.22383   1.716 0.087499 .
## Errors       -2.87871    4.42077  -0.651 0.515539
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 319.9 on 246 degrees of freedom
## Multiple R-squared:  0.5279, Adjusted R-squared:  0.4972
## F-statistic: 17.19 on 16 and 246 DF,  p-value: < 2.2e-16
```

We see $R^2_{adj} = 0.497$. This means that about 49.7% of the variability in Salary is explained by Model 1.

4. Model 2: Full Model – Log(Salary)

```
model2 <- lm(log(Salary) ~ ., data=Hitters)
summary(model2)
```

```
##
## Call:
## lm(formula = log(Salary) ~ ., data = Hitters)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.27815 -0.45173  0.06541  0.41628  2.87554
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.6180205  0.1622340  28.465  < 2e-16 ***
## AtBat       -0.0034410  0.0012340  -2.789  0.00571 **
## Hits         0.0139192  0.0046599   2.987  0.00310 **
## HmRun        0.0087610  0.0120984   0.724  0.46966
## Runs        -0.0014271  0.0058366  -0.245  0.80704
## RBI         -0.0003191  0.0050641  -0.063  0.94980
## Walks        0.0110596  0.0035702   3.098  0.00218 **
## Years        0.0558455  0.0241611   2.311  0.02164 *
## CAtBat       0.0001446  0.0002651   0.546  0.58588
## CHits       -0.0006355  0.0013168  -0.483  0.62980
## CHmRun      -0.0003842  0.0031731  -0.121  0.90373
## CRuns        0.0017605  0.0014582   1.207  0.22846
## CRBI         0.0002197  0.0013580   0.162  0.87164
## CWalks      -0.0014416  0.0006449  -2.235  0.02630 *
## PutOuts      0.0003654  0.0001519   2.406  0.01689 *
## Assists      0.0006273  0.0004342   1.445  0.14983
## Errors      -0.0100029  0.0085764  -1.166  0.24461
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6206 on 246 degrees of freedom
## Multiple R-squared:  0.5427, Adjusted R-squared:  0.5129
## F-statistic: 18.25 on 16 and 246 DF,  p-value: < 2.2e-16
```

We see $R^2_{adj} = 0.513$. This means that about 51.3% of the variability in Log(Salary) is explained by Model 2.

5. Convert dataset to matrix format

```
X <- as.matrix(Hitters[,-which(colnames(Hitters) == "Salary")])
```

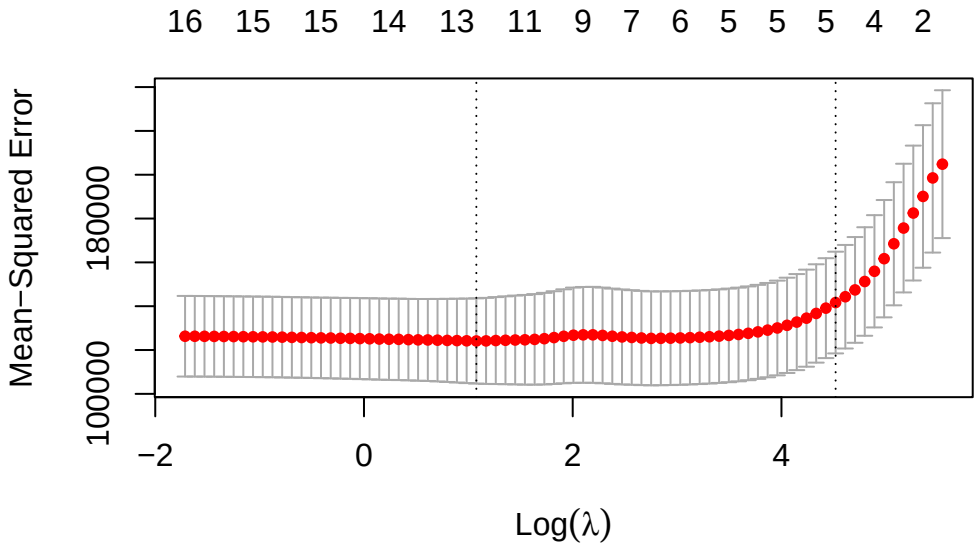
6. Model 3: Use LASSO with cross-validation – Salary

a. LASSO with 10-fold cv

```
library(glmnet)
lasso.cv <- cv.glmnet(x=X, y=Salary, nfolds=10, type.measure="mse")
```

b. MSPE for the sequence of λ values

```
plot(lasso.cv)
```



c. Use the λ value that leads to a more parsimonious model

```
lasso.cv$lambda.1se
```

```
## [1] 91.74363
```

d. Covaroates associated with this λ

```
lasso.1se <- glmnet(x=X,y=Salary, lambda=lasso.cv$lambda.1se)
lasso.1se$beta
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                 s0
## AtBat    .
## Hits     1.21576856
## HmRun    .
## Runs     .
## RBI      .
## Walks    1.29215655
## Years    .
## CAtBat   .
## CHits    .
## CHmRun   .
## CRuns    0.12267555
## CRBI     0.32146185
## CWalks   .
## PutOuts  0.02499026
## Assists  .
## Errors   .
```

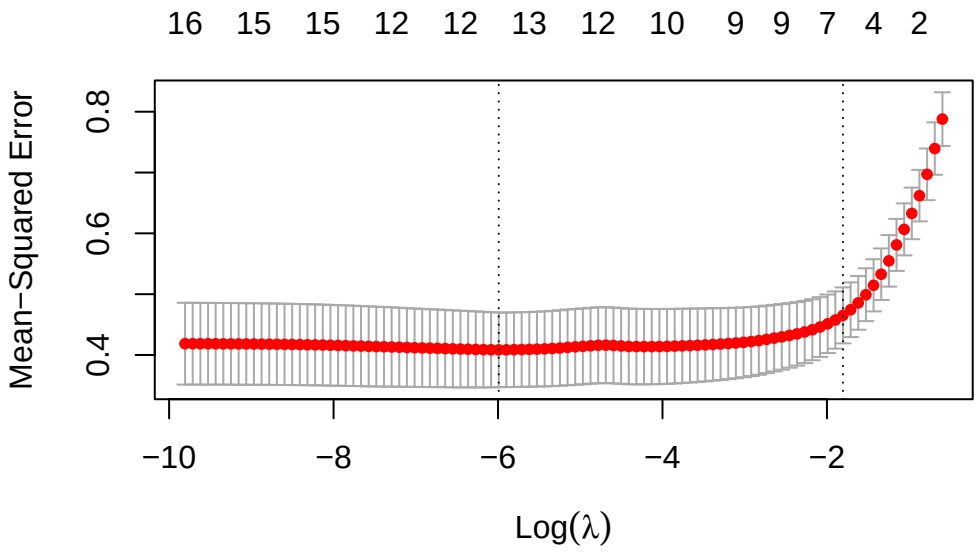
7. Model 4: LASSO for Log(Salary)

a. LASSO with 10-fold cv

```
lasso2.cv <- cv.glmnet(x=X, y=log(Salary), nfolds=10, type.measure="mse")
```

b. MSPE for the sequence of λ values

```
plot(lasso2.cv)
```



c. Use the λ value that leads to a more parsimonious model

```
lasso2.cv$lambda.1se
```

```
## [1] 0.1644637
```

d. Covaroates associated with this λ

```
lasso2.1se <- glmnet(x=X,y=log(Salary), lambda=lasso2.cv$lambda.1se)
lasso2.1se$beta
```

```
## 16 x 1 sparse Matrix of class "dgCMatrix"
##                      s0
## AtBat    .
## Hits     2.784740e-03
## HmRun    .
## Runs     .
## RBI      5.778134e-05
```

```
## Walks   2.063297e-03
## Years   .
## CAtBat  .
## CHits   3.690392e-04
## CHmRun  .
## CRuns   2.814887e-04
## CRBI    4.423564e-05
## CWalks  .
## PutOuts .
## Assists .
## Errors  .
```

8. Fit and compare Model 3 and Model 4

```
model3 <- lm(Salary ~ Hits+Walks+CRuns+CRBI+PutOuts)
summary(model3)
```

```
##
## Call:
## lm(formula = Salary ~ Hits + Walks + CRuns + CRBI + PutOuts)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -914.21 -171.94  -33.26   97.63 2197.08
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -96.96096   55.62583  -1.743 0.082513 .
## Hits          2.09338    0.57376   3.649 0.000319 ***
## Walks         2.51513    1.22010   2.061 0.040269 *
## CRuns         0.26490    0.19463   1.361 0.174679
## CRBI          0.39549    0.19755   2.002 0.046339 *
## PutOuts       0.26620    0.07857   3.388 0.000814 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 333 on 257 degrees of freedom
## Multiple R-squared:  0.4654, Adjusted R-squared:  0.455
## F-statistic: 44.75 on 5 and 257 DF,  p-value: < 2.2e-16
```

```
model4 <- lm(log(Salary) ~ Hits+RBI+Walks+Years+CHits+CRuns+CRBI)
summary(model4)
```

```
##
## Call:
## lm(formula = log(Salary) ~ Hits + RBI + Walks + Years + CHits +
##     CRuns + CRBI)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.06335 -0.49503  0.04739  0.42930  3.13549
```

```
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.417e+00  1.434e-01  30.808  < 2e-16 ***
## Hits        5.284e-03  1.771e-03   2.984  0.00312 **
## RBI         1.314e-03  3.082e-03   0.426  0.67015
## Walks       5.919e-03  2.552e-03   2.319  0.02117 *
## Years       5.101e-02  2.096e-02   2.433  0.01566 *
## CHits       3.302e-04  4.418e-04   0.747  0.45554
## CRuns       4.369e-05  7.682e-04   0.057  0.95469
## CRBI        8.257e-06  4.756e-04   0.017  0.98616
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6316 on 255 degrees of freedom
## Multiple R-squared:  0.509,  Adjusted R-squared:  0.4955
## F-statistic: 37.76 on 7 and 255 DF,  p-value: < 2.2e-16
```

Model 3 has 5 covariates (4 significant) with $R^2_{adj} = 0.455$.

Model 4 has 7 covariates (3 significant) with $R^2_{adj} = 0.496$.

Although Model 4 has fewer significant covariates, it has a slightly higher R^2_{adj}.

Question 2

1. Shuffle the dataset

```
set.seed(12345)
Hitters <- Hitters[sample(1:263), ]
```

2. How many partitions and nrows of data in each partition

```
nk <- c(rep(x=13,17),rep(x=14,3))
sum(nk)
```

```
## [1] 263
```

3. Partition table

```
partition <- c()
for (i in 1:length(nk)) {
    partition <- c(partition, rep(x=i,nk[i]))
}
```

4. The first fold

```
training <- Hitters[which(partition!=20),]
test <- Hitters[which(partition==20),]
```

5. Create an empty list to store the folds

```
my.folds <- vector(mode="list", length=20)
```

6. Populate this list

```
for (i in 1:20) {
    test_idx <- 21-i
    my.folds[[i]]$training <- Hitters[which(partition!=test_idx),]
    my.folds[[i]]$test <- Hitters[which(partition==test_idx),]
}
#my.folds[[1]]
```

Question 3

1. Fit Model 1 using 1st fold and calculate MSPE

```
mod1_fold1 <- lm(Salary ~ ., data=my.folds[[1]]$training)
mean((my.folds[[1]]$test[,1] - predict(mod1_fold1, my.folds[[1]]$test))^2)
```

```
## [1] 118921.7
```

2 & 3. Calculate MSPE for Model 1 and Model 3 with cv

```
MSPE.mod1 <- rep(NA,20)
MSPE.mod3 <- rep(NA,20)

for (i in 1:20) {
    mod1_fold_i <- lm(Salary ~ ., data=my.folds[[i]]$training)
    mod3_fold_i <- lm(Salary ~ Hits+Walks+CRuns+CRBI+PutOuts, data=my.folds[[i]]$training)

    MSPE.mod1[i] <- mean((my.folds[[i]]$test[,1] - predict(mod1_fold_i, my.folds[[i]]$test))^2)
    MSPE.mod3[i] <- mean((my.folds[[i]]$test[,1] - predict(mod3_fold_i, my.folds[[i]]$test))^2)
}
MSPE.mod1
```

```
##  [1] 118921.75  41321.59  70280.53 259586.91  80823.41  33215.43  36973.59
##  [8] 174760.76 132080.53  88782.35  88426.87  43518.99 323354.53  84973.54
## [15]  81076.38 158598.86  21111.42  84419.46  58676.98  59581.74
```

```
MSPE.mod3
```

```
##  [1] 106419.89  25307.23  55270.06 198243.47  57445.91  33899.36  44444.29
##  [8]  87590.17 100193.29  96424.64  81607.53  12429.62 200393.95  72462.70
## [15] 103586.40 112161.77  23785.17  39843.81  44076.35  39579.70
```

4. Calculate MSPE for Model 2 and Model 4 with cv

```
MSPE.mod2 <- rep(NA,20)
MSPE.mod4 <- rep(NA,20)
for (i in 1:20) {
    mod2_fold_i <- lm(log(Salary) ~ ., data=my.folds[[i]]$training)
    mod4_fold_i <- lm(log(Salary) ~ Hits+RBI+Walks+Years+CHits+CRuns+CRBI,
                      data=my.folds[[i]]$training)

    MSPE.mod2[i] <- mean((my.folds[[i]]$test[,1] -
                              exp(predict(mod2_fold_i, my.folds[[i]]$test)))^2)
    MSPE.mod4[i] <- mean((my.folds[[i]]$test[,1] -
                              exp(predict(mod4_fold_i, my.folds[[i]]$test)))^2)
}
MSPE.mod2
```

```
##  [1]   67389.22   46842.69   83915.91  534559.03   47183.59   31145.16
##  [7]   35766.88  259685.41   62716.85  331853.01   54890.11   55703.99
## [13] 1479954.08  155451.40  112934.89  264288.64   26322.91   66164.25
## [19]   25239.85   16330.13
```

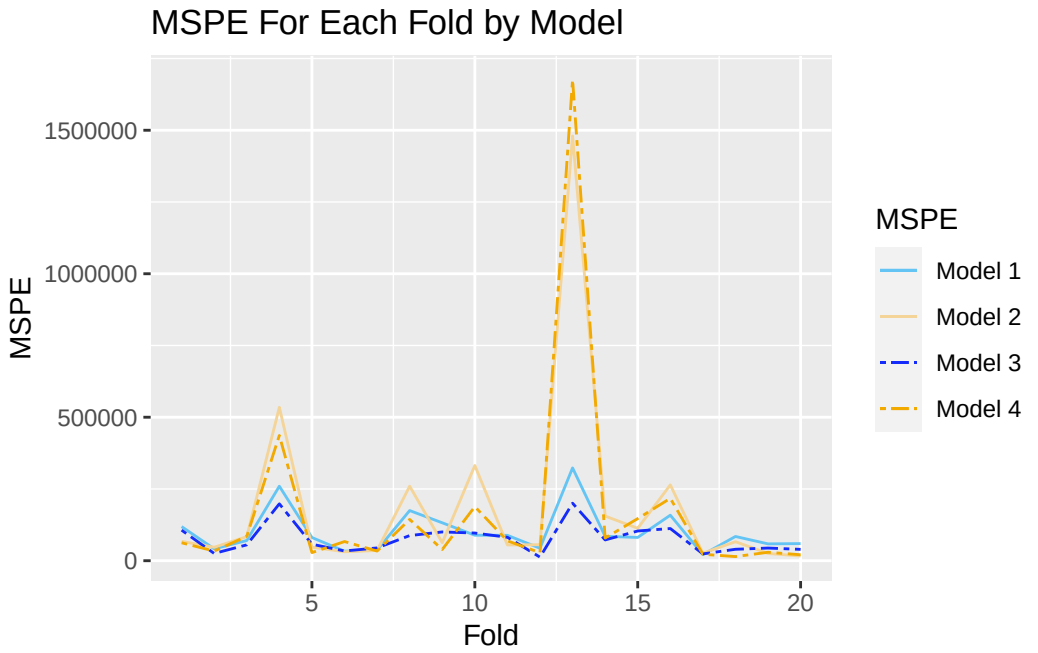
```
MSPE.mod4
```

```
##  [1]   62482.01   33507.37   84942.07  436476.76   28212.56   66804.00
##  [7]   33636.74  143806.68   39090.92  188539.98   69248.20   27893.78
## [13] 1676179.90   78993.61  146800.70  217346.33   22476.40   14342.01
## [19]   29410.98   20706.47
```

5. MSPE Plot

```
library(ggplot2)
mspe_df <- data.frame(Fold = 1:20,
                      "Model_1" = MSPE.mod1,
                      "Model_2" = MSPE.mod2,
                      "Model_3" = MSPE.mod3,
                      "Model_4" = MSPE.mod4)
ggplot(mspe_df, aes(x=Fold)) +
    geom_line(aes(y=Model_1, color="Model 1", linetype="Model 1")) +
    geom_line(aes(y=Model_2, color="Model 2", linetype="Model 2")) +
    geom_line(aes(y=Model_3, color="Model 3", linetype="Model 3")) +
    geom_line(aes(y=Model_4, color="Model 4", linetype="Model 4")) +
    scale_colour_manual("MSPE",
```

```
                      breaks = c("Model 1","Model 2","Model 3","Model 4"),
                      values = c("#64c5f5", "#f5d498", "#142bfa", "#f2a900")) +
    scale_linetype_manual("MSPE",
                      breaks = c("Model 1","Model 2","Model 3","Model 4"),
                      values = c("solid", "solid", "twodash", "twodash"))+
    ggtitle("MSPE For Each Fold by Model") +
    ylab("MSPE")
```

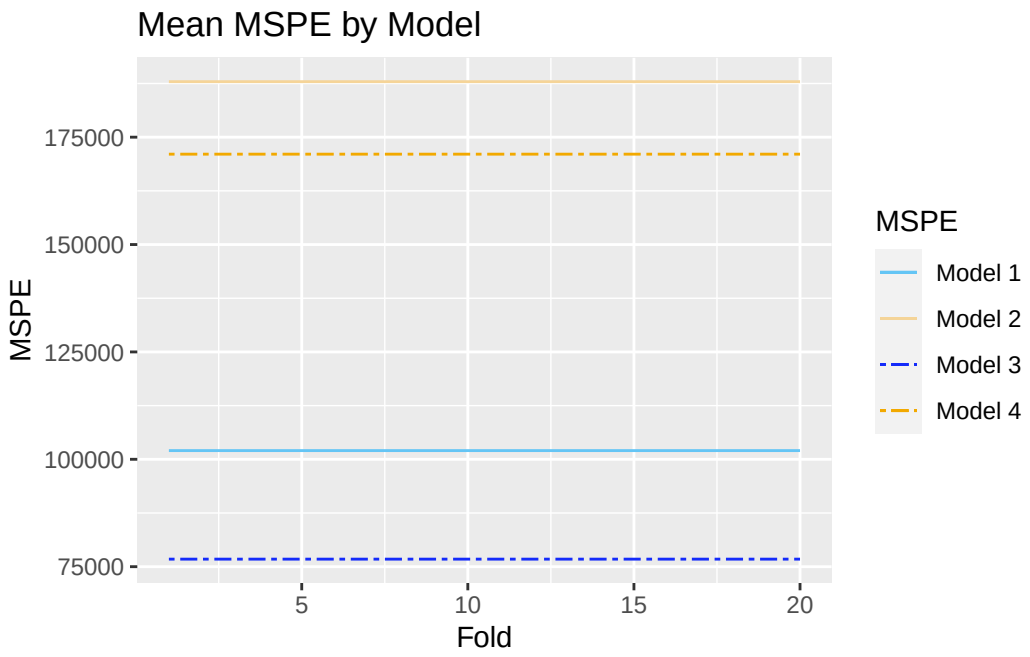


The untransformed models (1 & 3) are in blue; the Log models (2 & 4) are in orange.

The regular models (1 & 2) have solid lines; the LASSO models (3 & 4) have dashed lines.

6. Mean MSPE Plot

```
mspe_mean_df <- data.frame(Fold = 1:20,
                      "Model_1" = mean(MSPE.mod1),
                      "Model_2" = mean(MSPE.mod2),
                      "Model_3" = mean(MSPE.mod3),
                      "Model_4" = mean(MSPE.mod4))
ggplot(mspe_mean_df, aes(x=Fold)) +
    geom_line(aes(y=Model_1, color="Model 1", linetype="Model 1")) +
    geom_line(aes(y=Model_2, color="Model 2", linetype="Model 2")) +
    geom_line(aes(y=Model_3, color="Model 3", linetype="Model 3")) +
    geom_line(aes(y=Model_4, color="Model 4", linetype="Model 4")) +
    scale_colour_manual("MSPE",
                      breaks = c("Model 1","Model 2","Model 3","Model 4"),
                      values = c("#64c5f5", "#f5d498", "#142bfa", "#f2a900")) +
    scale_linetype_manual("MSPE",
                      breaks = c("Model 1","Model 2","Model 3","Model 4"),
                      values = c("solid", "solid", "twodash", "twodash"))+
    ggtitle("Mean MSPE by Model") +
    ylab("MSPE")
```



7. Model Recommendation

```
sd_mspe <- c(sd(MSPE.mod1),sd(MSPE.mod2),sd(MSPE.mod3),sd(MSPE.mod4))
results_df <- data.frame("adj R_sq" = c(0.497, 0.513, 0.455, 0.496),
                         "Mean MSPE" = t(mspe_mean_df[1,2:5])[1:4],
                         "SD MSPE" = sd_mspe,
                         row.names = c("Model 1", "Model 2","Model 3","Model 4"))
knitr::kable(results_df)
```

	adj.R_sq	Mean.MSPE	SD.MSPE
Model 1	0.497	102024.28	77000.10
Model 2	0.513	187916.90	331533.09
Model 3	0.455	76758.27	51866.74
Model 4	0.496	171044.87	368064.62

Looking at the R^2_{adj} values, we see that all four models are at around 50%, with the LASSO models (3 and 4) having slightly lower R^2_{adj} compared to the first two models. But if we look at mean MSPE, we see the LASSO models performing better, despite having lower R^2_{adj}. Also noticing that the log models (2 and 4) have much higher mean MSPE and variability in MSPE values, so we would choose the untransformed model (1 and 2). Then, since Model 3 has the lowest mean MSPE and lowest MSPE for almost every fold, and the lowest variability (judging by the plot), I recommend using Model 3 to predict the salaries of players.