PGA Stats Analysis and ML

Ryan Kyaw

June 5, 2020

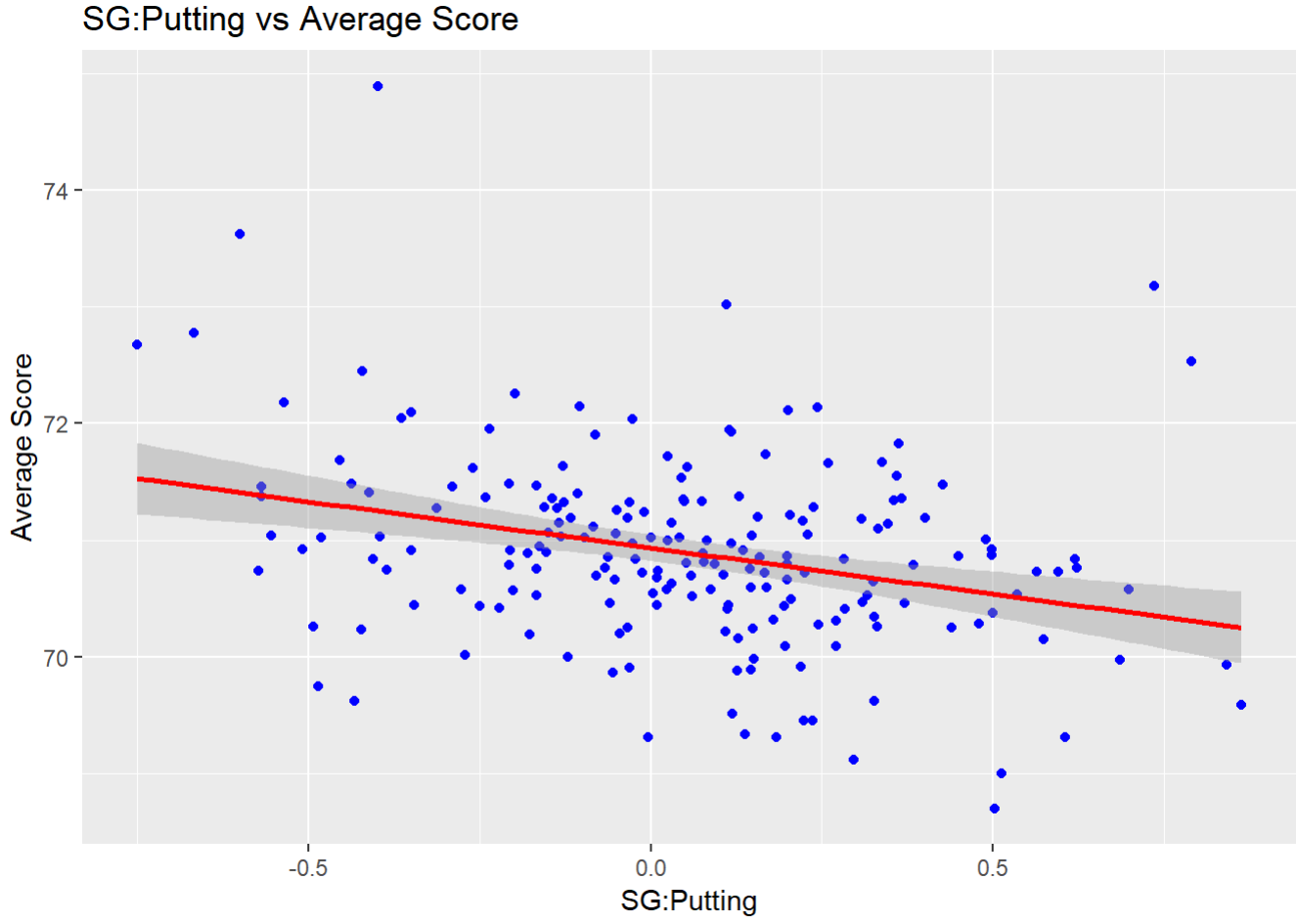
Is there statistical significance that any Strokes Gained statistics directly correlates to lower golf scores?

Let's look at the 2017 PGA Tour Strokes Gained and Scoring Average data to see.

```
df_2017 <- read.csv("2017.csv")
```

Looking at the data from the 2017 golf season, we can use a graphical analysis and correlation coefficient to see how strokes gained compares to average score.

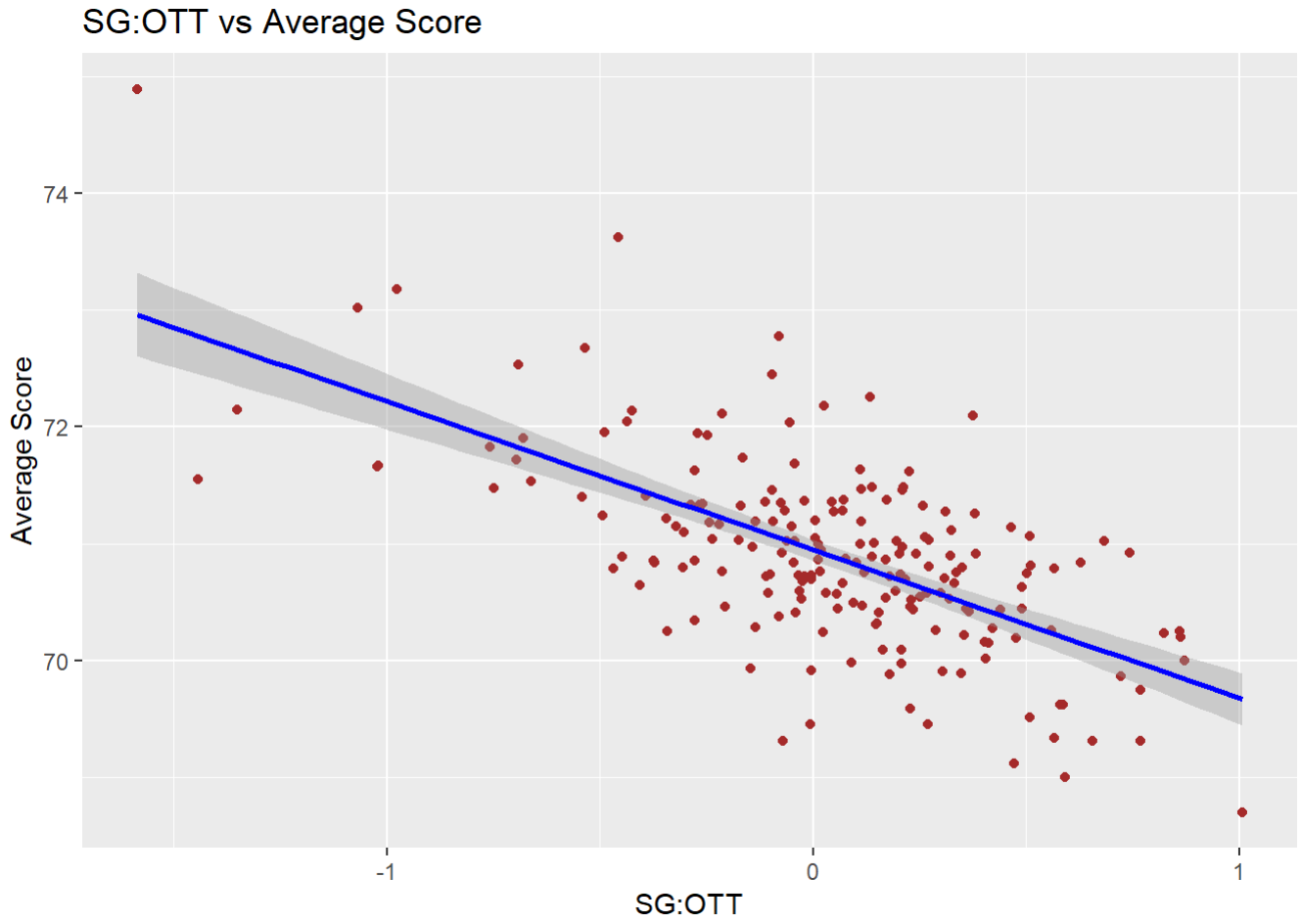
```
ggplot(df_2017) +
  geom_point(aes(x = df_2017$SG_PUTTING_PER_ROUND, y = df_2017$AVG_SCORE), color = "blue") +
  ggtitle("SG:Putting vs Average Score")+
  xlab("SG:Putting")+
  ylab("Average Score")+
  geom_smooth(method = "lm", aes(x = df_2017$SG_PUTTING_PER_ROUND, y = df_2017$AVG_SCORE), color
= "red")
```




```
cor(x = df_2017$AVG_SCORE, y = df_2017$SG_PUTTING_PER_ROUND)
```

```
## [1] -0.3028644
```

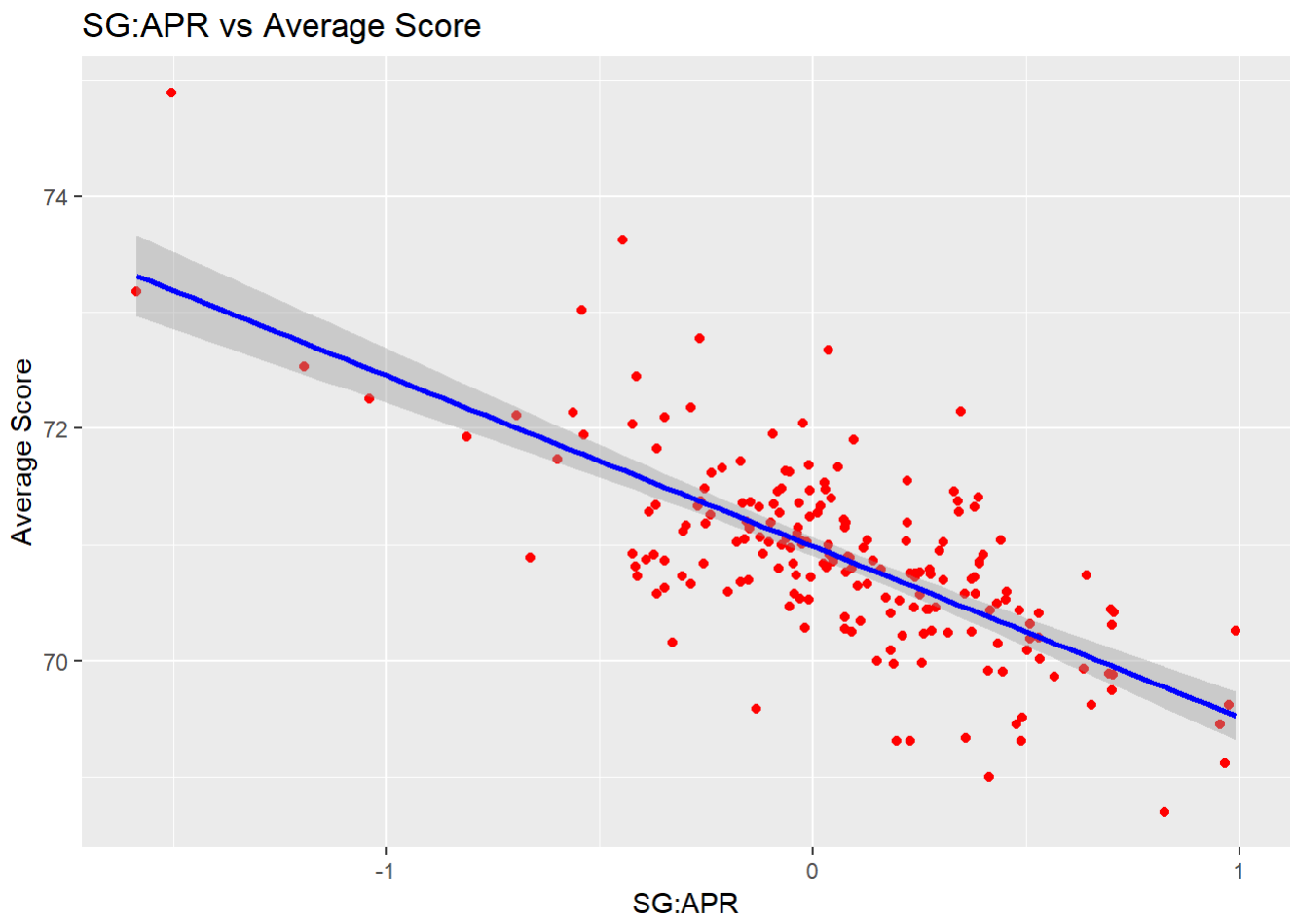
```
ggplot(df_2017) +
  geom_point(aes(x = df_2017$SG.OTT, y = df_2017$AVG_SCORE), color = "brown")+
  ggtitle("SG:OTT vs Average Score")+
  xlab("SG:OTT")+
  ylab("Average Score")+
  geom_smooth(method = "lm", aes(x = df_2017$SG.OTT, y = df_2017$AVG_SCORE), color = "blue")
```




```
cor(x = df_2017$AVG_SCORE, y = df_2017$SG.OTT)
```

```
## [1] -0.6471942
```

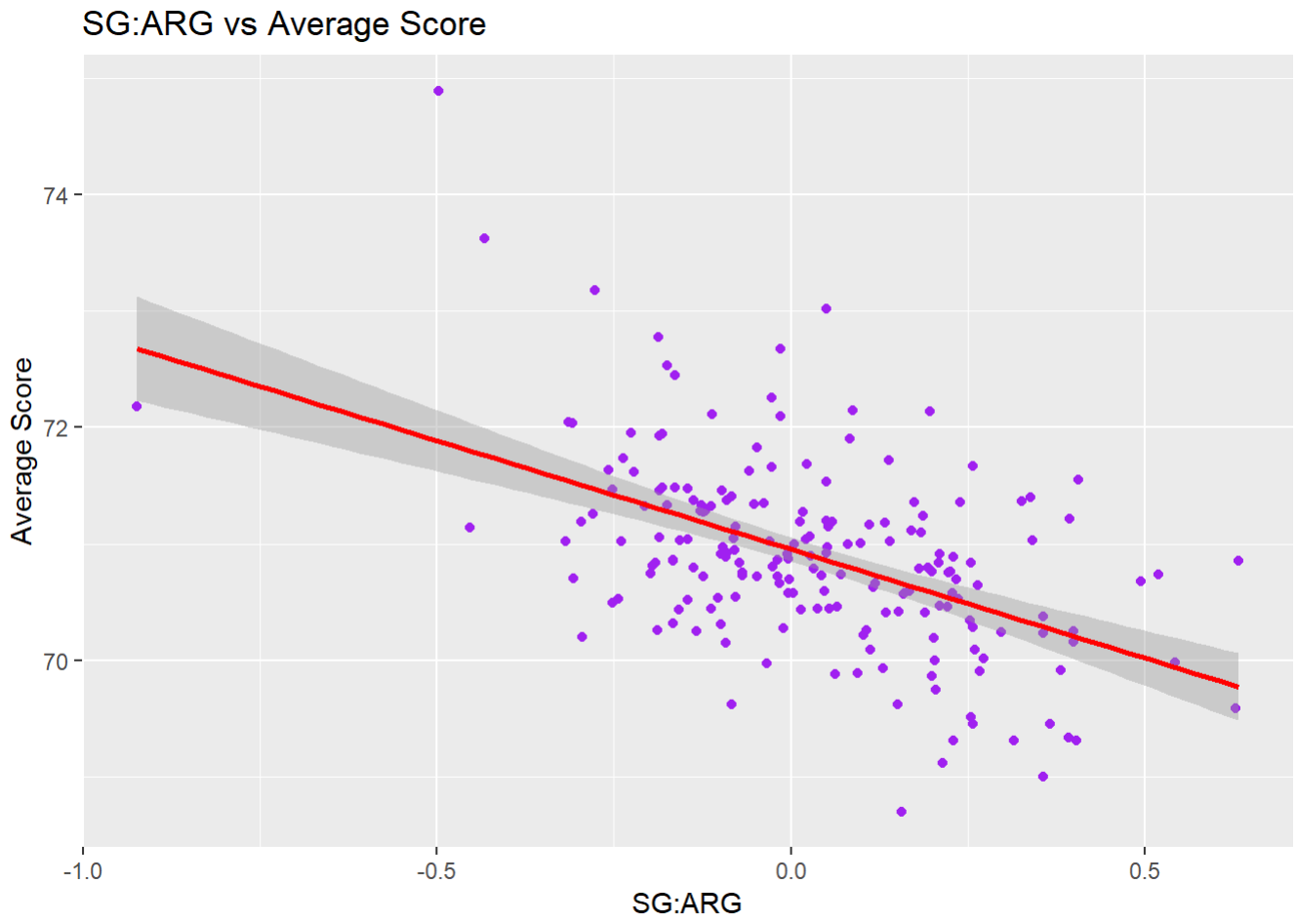
```
ggplot(df_2017) +
  geom_point(aes(x = df_2017$SG.APR, y = df_2017$AVG_SCORE), color = "red")+
  ggtitle("SG:APR vs Average Score")+
  xlab("SG:APR")+
  ylab("Average Score")+
  geom_smooth(method = "lm", aes(x = df_2017$SG.APR, y = df_2017$AVG_SCORE), color = "blue")
```




```
cor(x = df_2017$AVG_SCORE, y = df_2017$SG.APR)
```

```
## [1] -0.7094093
```

```
ggplot(df_2017) +
  geom_point(aes(x = df_2017$SG.ARG, y = df_2017$AVG_SCORE), color = "purple")+
  ggtitle("SG:ARG vs Average Score")+
  xlab("SG:ARG")+
  ylab("Average Score")+
  geom_smooth(method = "lm", aes(x = df_2017$SG.ARG, y = df_2017$AVG_SCORE), color = "red")
```




```
cor(x = df_2017$AVG_SCORE, y = df_2017$SG.ARG)
```

```
## [1] -0.5021516
```

The negative correlation values appear to imply that the higher SG values correlate to lower scores.

Let's create linear models for each of the Strokes Gained categories versus Scoring Average.

```
lm_putt <- lm(AVG_SCORE ~ SG_PUTTING_PER_ROUND, data = df_2017)
summary(lm_putt)
```

```
##
## Call:
## lm(formula = AVG_SCORE ~ SG_PUTTING_PER_ROUND, data = df_2017)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.8340 -0.4115 -0.0352  0.3846  3.6416
##
## Coefficients:
##                      Estimate Std. Error  t value Pr(>|t|)
## (Intercept)          70.93431    0.05691 1246.361  < 2e-16 ***
## SG_PUTTING_PER_ROUND -0.79180    0.17935   -4.415 1.68e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7841 on 193 degrees of freedom
## Multiple R-squared:  0.09173,    Adjusted R-squared:  0.08702
## F-statistic: 19.49 on 1 and 193 DF,  p-value: 1.681e-05
```

```
lm_drive <- lm(AVG_SCORE ~ SG.OTT, data = df_2017)
summary(lm_drive)
```

```
##
## Call:
## lm(formula = AVG_SCORE ~ SG.OTT, data = df_2017)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.72500 -0.38127 -0.03402  0.38564  2.09652
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 70.94710    0.04515 1571.49   <2e-16 ***
## SG.OTT      -1.26997    0.10768  -11.79   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6272 on 193 degrees of freedom
## Multiple R-squared:  0.4189, Adjusted R-squared:  0.4158
## F-statistic: 139.1 on 1 and 193 DF,  p-value: < 2.2e-16
```

```
lm_ARG <- lm(AVG_SCORE ~ SG.ARG, data = df_2017)
summary(lm_ARG)
```

```
##
## Call:
## lm(formula = AVG_SCORE ~ SG.ARG, data = df_2017)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.95838 -0.41526 -0.06013  0.32838  3.01056
##
## Coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept) 70.95327    0.05149 1377.980  < 2e-16 ***
## SG.ARG      -1.86555    0.23126   -8.067 7.48e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7115 on 193 degrees of freedom
## Multiple R-squared:  0.2522, Adjusted R-squared:  0.2483
## F-statistic: 65.08 on 1 and 193 DF,  p-value: 7.484e-14
```

```
lm_APR <- lm(AVG_SCORE ~ SG.APR, data = df_2017)
summary(lm_APR)
```

```
##
## Call:
## lm(formula = AVG_SCORE ~ SG.APR, data = df_2017)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.59364 -0.29726 -0.01785  0.31393  1.98287
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 70.98417    0.04203 1688.85   <2e-16 ***
## SG.APR      -1.46972    0.10510  -13.98   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5799 on 193 degrees of freedom
## Multiple R-squared:  0.5033, Adjusted R-squared:  0.5007
## F-statistic: 195.5 on 1 and 193 DF,  p-value: < 2.2e-16
```

Despite the relatively low R-squared values, especially with Strokes Gained Putting vs Average Score, the significant coefficient estimates make these models relatively strong in predicting average score. Human subjects can result in highly varied golf results, and this amount of noise can explain the low R-squared values.

Now, let's run a hypothesis test to see if the correlation between the Strokes Gained variables and Average Score is statistically significant.

Pearson Correlation Coefficient

```
putttest <- cor.test(df_2017$SG_PUTTING_PER_ROUND, df_2017$AVG_SCORE, method = "pearson")
putttest
```

```
##
##  Pearson's product-moment correlation
##
## data:  df_2017$SG_PUTTING_PER_ROUND and df_2017$AVG_SCORE
## t = -4.4149, df = 193, p-value = 1.681e-05
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.4252785 -0.1695682
## sample estimates:
##        cor
## -0.3028644
```

```
argtest <- cor.test(df_2017$SG.ARG, df_2017$AVG_SCORE, method = "pearson")
argtest
```

```
##
##  Pearson's product-moment correlation
##
## data:  df_2017$SG.ARG and df_2017$AVG_SCORE
## t = -8.0669, df = 193, p-value = 7.484e-14
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.6003072 -0.3890930
## sample estimates:
##        cor
## -0.5021516
```

```
aprtest <- cor.test(df_2017$SG.APR, df_2017$AVG_SCORE, method = "pearson")
aprtest
```

```
##
##  Pearson's product-moment correlation
##
## data:  df_2017$SG.APR and df_2017$AVG_SCORE
## t = -13.983, df = 193, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.7728804 -0.6318836
## sample estimates:
##        cor
## -0.7094093
```

```
otttest <- cor.test(df_2017$SG.OTT, df_2017$AVG_SCORE, method = "pearson")
otttest
```

```
##
##  Pearson's product-moment correlation
##
## data:  df_2017$SG.OTT and df_2017$AVG_SCORE
## t = -11.794, df = 193, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.7220447 -0.5573682
## sample estimates:
##       cor
## -0.6471942
```

All of these p-values are significant, and this means that we can confirm that the true correlation is not equal to 0. Therefore, we can trust our calculate correlation coefficients.

Now, let's use linear regression to create a model that uses the Strokes Gained data to predict a players average scores.

First, lets Train/Test split our data.

```
df_2017$id <- 1:nrow(df_2017)
df_train_2017 <- df_2017 %>% sample_frac(0.7)
df_test_2017 <- df_2017 %>% anti_join(df_train_2017)
```

```
## Joining, by = c("Player", "ROUNDS_PLAYED", "SG_PUTTING_PER_ROUND", "SG.OTT", "SG.APR", "SG.AR
G", "AVG_SCORE", "id")
```

```
df_train_2017$id <- NULL
df_test_2017$id <- NULL
write.csv(df_test_2017, "test.csv", row.names = FALSE)
rm(df_test_2017)
```

Now, we will use the training data to create our model.

```
train_model <- lm(AVG_SCORE ~ SG_PUTTING_PER_ROUND + SG.OTT + SG.APR + SG.ARG,
                  data = df_train_2017)
summary(train_model)
```

```
##
## Call:
## lm(formula = AVG_SCORE ~ SG_PUTTING_PER_ROUND + SG.OTT + SG.APR +
##     SG.ARG, data = df_train_2017)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5550 -0.1360 -0.0264  0.1266  0.6962
##
## Coefficients:
##                       Estimate Std. Error  t value Pr(>|t|)
## (Intercept)          71.09401    0.02033 3497.557  < 2e-16 ***
## SG_PUTTING_PER_ROUND -0.97868    0.07106  -13.772  < 2e-16 ***
## SG.OTT               -0.99150    0.06130  -16.174  < 2e-16 ***
## SG.APR               -1.02987    0.05982  -17.215  < 2e-16 ***
## SG.ARG               -0.92067    0.09905   -9.295 4.27e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2224 on 131 degrees of freedom
## Multiple R-squared:  0.9195, Adjusted R-squared:  0.9171
## F-statistic: 374.3 on 4 and 131 DF,  p-value: < 2.2e-16
```

Then, we load back the test data and use the model to predict on the test data.

```
df_test_2017 <- read.csv("test.csv")
avg_score_actual <- df_test_2017$AVG_SCORE
avg_score_pred <- predict(train_model, df_test_2017)
pred_vs_actual_scores <- data.frame(cbind(actual = avg_score_actual, predicted = avg_score_pre
d))
```

To analyze how effective our model is, we calculate the mean absolute percentage error

```
MAPE_home <- mean(abs((pred_vs_actual_scores$predicted - pred_vs_actual_scores$actual))
                  /pred_vs_actual_scores$actual)
```

This is a significantly low MAPE value, and this implies that our model does a good job in being accurate in predicting a player's average score from their Strokes Gained Data.

This further proves the strength of the Strokes Gained statistic in golf analytics.