Case Study 3

Instructions:

Get into groups of 3 and answer the following questions. Submit your knitted report as a pdf/html file and include the original .Rmd file also. Submit one report per group with all members' names on it before the end of class. You currently have the fundamental R tools to complete this exercise, but you will may still have to explore new techniques and packages. **For each question, write the name of the team member who answered/coded it.**

1. Regression Modeling & Functional Programming

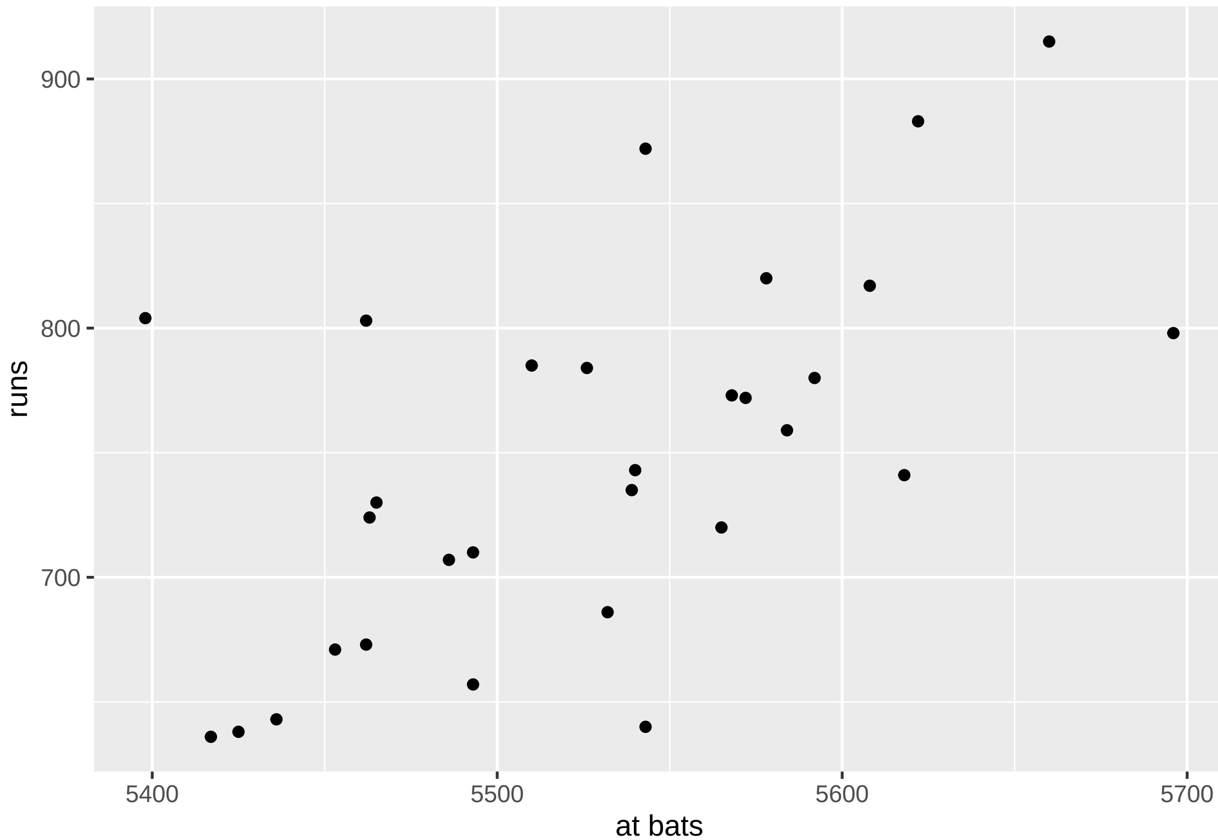
The movie Moneyball is about how proper use of statistics in baseball (called "sabermetrics") can bring unexpected success to a low-ranked, low-budget team. In it, the manager of the Oakland A's believes that (then) unpopular statistics, like a player's ability to get on base, can predict the team's ability to score runs better than traditional statistics, such as homerun counts and batting averages. By recruiting players who scored high in these underused statistics, he was able to improve the record of the team without needing to spend exorbitant amounts of money on the more mainstream players.

We will examine the data from the 30 MLB teams during the 2009 season. We will search for linear relationships between potential explanatory variables and the response variable: the number of runs scored in a season, which we treat as a measure of "success" for this data analysis. You don't need to know the rules of baseball to understand this question, but if you would like a refresher you can check out Wikipedia: https://en.wikipedia.org/wiki/Baseball_rules#Gameplay

In addition to runs scored, there are seven traditionally-used variables in the data set: at-bats, hits, homeruns, batting average, strikeouts, walks and stolen bases. The last three variables in the data set are "nontraditional": on-base percentage, slugging percentage, and on base plus slugging.

(a) Import the 2009 MLB dataset into R Studio. The dataset can be found on Canvas in the file "mlb09.csv". Using `ggplot`, plot `at_bats` on the x-axis and `runs` on the y-axis. Describe the relationship between the two variables in terms of direction (positively or negatively correlated), shape (linear? quadratic? exponential?) and strength. How confident would you rate your ability to predict a team's season runs scored, if you just knew the team's at-bats?

```r
mlb <- read.csv("~/Desktop/repos/Intro-DS-F23/Data/mlb09.csv")


library(ggplot2)

ggplot(mlb, aes(x = at_bats, y = runs)) + geom_point() + xlab("at bats")
```




```r
cor(mlb$at_bats, mlb$runs)
```

```
## [1] 0.6016984
```

```
There is a moderate, positive, linear association between a team's at bats and the number of runs they s
```

```
(b) Fit the regression of  `runs` onto  `at_bats`. Write the equation for the least squares regression l
```

```r
model1 = lm(runs ~ at_bats, data = mlb)
summary1 = summary(model1)
summary1
```

```
##
## Call:
## lm(formula = runs ~ at_bats, data = mlb)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -116.19  -46.16  -13.05   33.95  135.45
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2594.0311   838.2929  -3.094 0.004441 **
## at_bats         0.6044     0.1516   3.986 0.000436 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 60.6 on 28 degrees of freedom
## Multiple R-squared:  0.362,  Adjusted R-squared:  0.3393
## F-statistic: 15.89 on 1 and 28 DF,  p-value: 0.000436
```

The least squares regression line is:

$$\hat{y} = -2594.0311 + 0.6044x$$

The intercept does not have a meaningful interpretation ("We expect the number of runs of a team with 0 at-bats to be -2594.0311.").

An interpretation for the slope is: "For every additional at-bat a team has, we expect on average a 0.6044 increase in number of runs."

(c) Identify the coefficient of determination for this regression. Interpret R^2 in terms of the regre

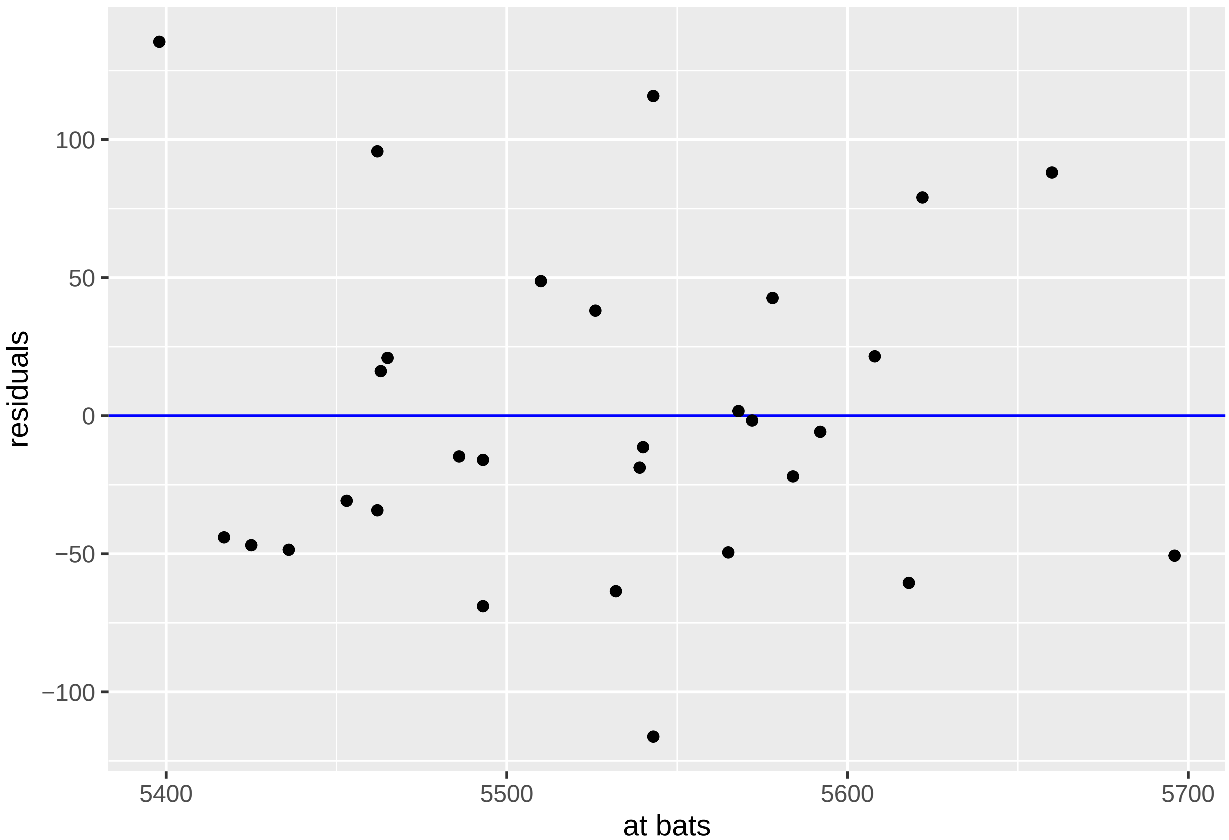
```r
paste0("The coefficient of determination is ", round(summary1$r.squared,
    3), ".")
```

```
## [1] "The coefficient of determination is 0.362."
```

36.2% of the variation in runs scored is explained by the least squares regression line onto at-bats.

(d) Create the residual plot and discuss what it indicates. Does the plot suggest linearity or not? Do

```r
mlb$residuals = model1$residuals

ggplot(mlb, aes(x = at_bats, y = residuals)) + geom_hline(aes(yintercept = 0),
    color = "blue") + geom_point() + xlab("at bats") + ylab("residuals")
```



The residuals appear to be approximately random scattered about 0, so the fit appears to be OK. We are reassured that fitting a linear model is appropriate. We do not suspect anything wrong with our model.

(e) Suppose the manager of a team comes and asks you to predict how many runs his team will score if the

```
predict_model1 = function(x) {
    if (!is.numeric(x)) {
        "Error: need numeric x"
        break
    } else {
        yhat = model1$coefficients[1] + model1$coefficients[2] *
            x
    }
    df = data.frame(x = x, yhat = yhat)
    return(df)
}
```

(f) Use your function from (e) to predict how many runs his team will score if they get 5000 at-bats, 55

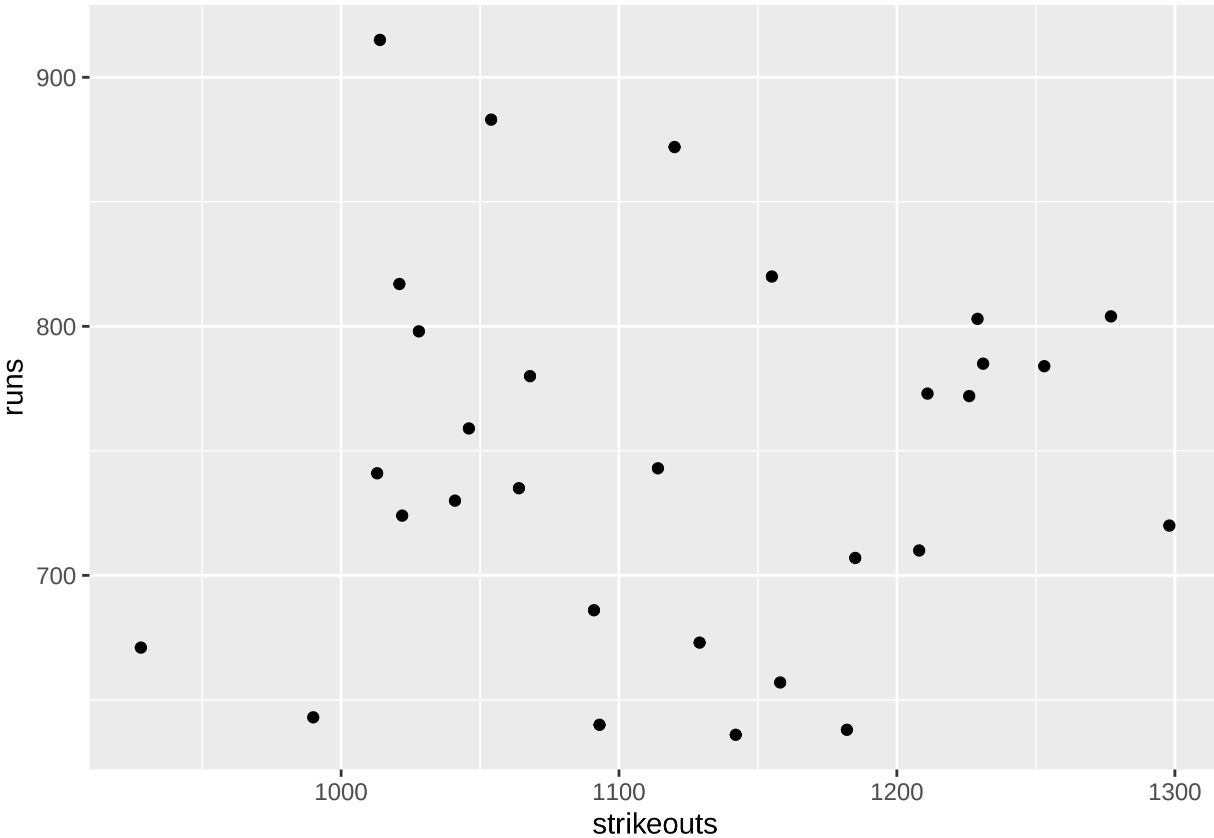
```
predict_model1(x = c(5000, 5500, 6000))
```

```
##      x      yhat
## 1 5000  427.9929
## 2 5500  730.1953
## 3 6000 1032.3977
```

For 5000 at-bats, we predict 427.999 runs. For 5500 at-bats, we predict 730.195 runs. For 6000 at-bats,

(g) Suppose a rival team coach claims that `strikeouts` are the most important factor in scoring runs.

```
ggplot(mlb, aes(x = strikeouts, y = runs)) + geom_point()
```




```
cor(mlb$strikeouts, mlb$runs)
```

```
## [1] 0.02185242
```

```
model2 = lm(runs ~ strikeouts, data = mlb)
summary2 = summary(model2)
summary2$r.squared
```

```
## [1] 0.0004775284
```

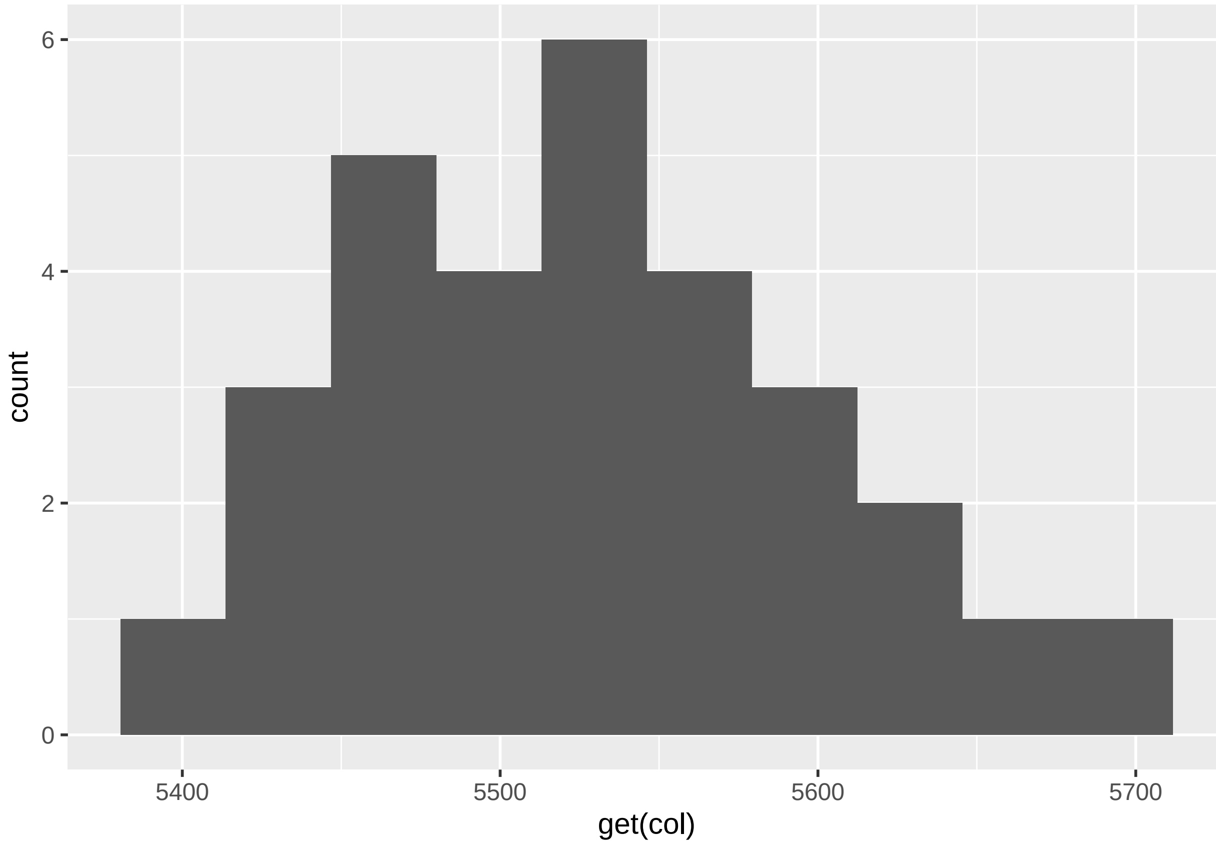
The plot suggests only the vaguest possibility of a relationship between strikeouts and runs. Additional

(h) We want to determine which of the 7 traditional variables (at-bats, hits, homeruns, batting average

Implement defensive programming to ensure that you receive the expected input; write a descriptive erro

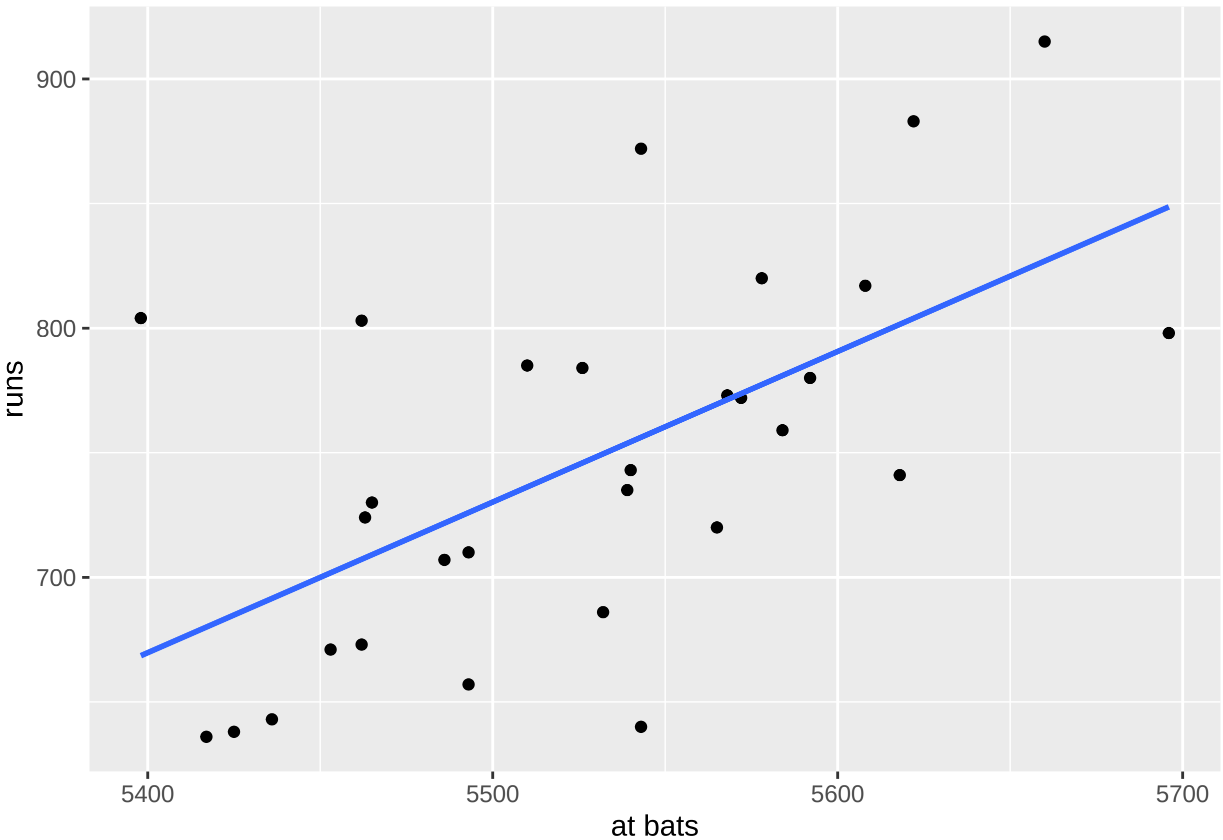
```
ex_function = function(col) {
    gg = ggplot(mlb, aes(x = get(col))) + geom_histogram(bins = 10)
    return(gg)
}
```

```r
ex_function("at_bats")
```




```r
check_plot = function(col) {
    gg = ggplot(mlb, aes(x = get(col), y = runs)) + geom_point() +
        geom_smooth(method = "lm", se = FALSE) + xlab(str_replace(col,
        pattern = "_", replacement = " "))
    return(gg)
}

# test it out
check_plot("at_bats")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



(i) Now we want to which of the 7 traditional variables (at-bats, hits, homeruns, batting average, stri

Write a function which fits a linear model and returns the R^2 value for any given variable on the x-a

```
check_r2 = function(col) {
    model = lm(runs ~ get(col), data = mlb)
    r2 = round(summary(model)$r.squared, 3)
    return(r2)
}

check_r2("at_bats")
```

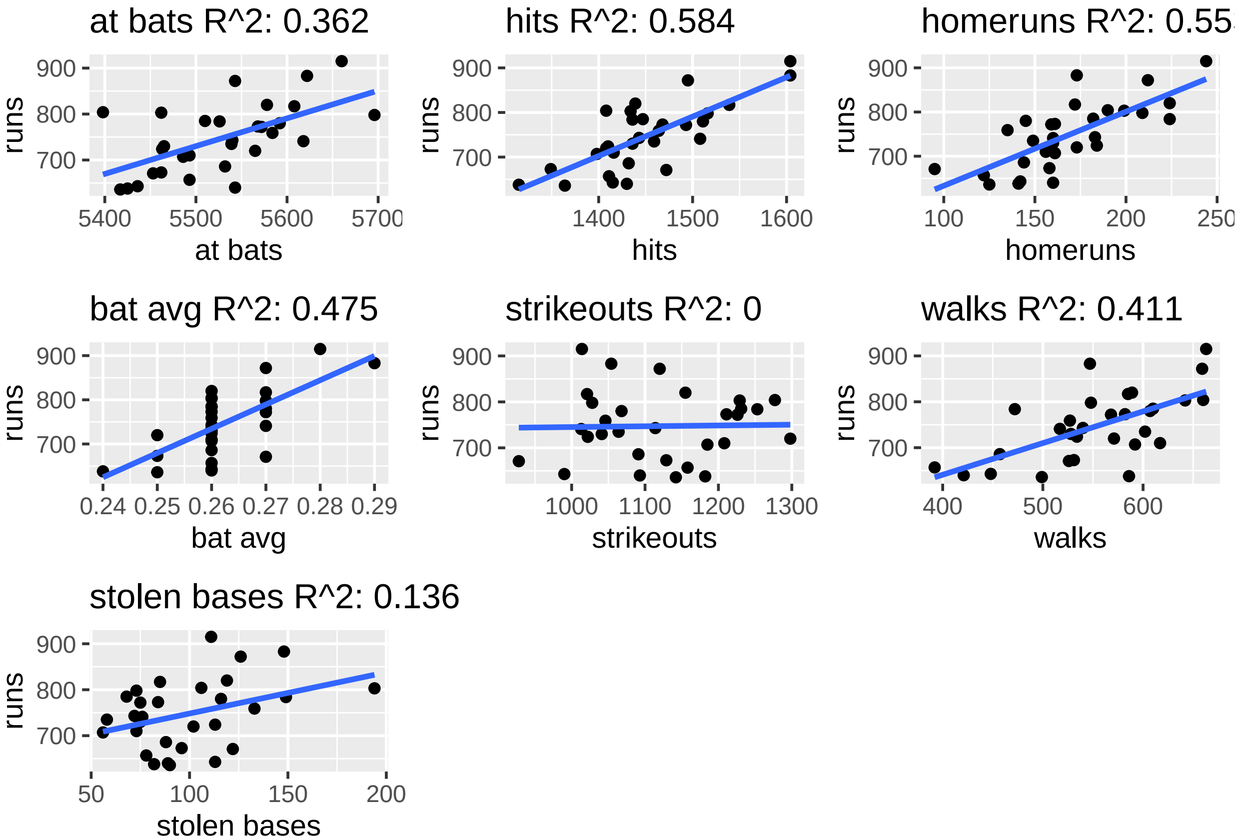
```
## [1] 0.362
```

(j)
Apply your functions from (h) and (i) to the dataset *by using a for loop* over the 7 traditional varial

```
trad_vars = colnames(mlb)[3:9]

plot_list = list()
r2_list = list()
for (i in 1:7) {
    r2 = check_r2(trad_vars[i])
    gg = check_plot(trad_vars[i]) + ggtitle(paste(str_replace(trad_vars[i],
        pattern = "_", replacement = " "), "R^2:", r2))
    r2_list[[i]] = r2
    plot_list[[i]] = gg
}
```

```r
library(ggpubr)
ggarrange(plotlist = plot_list, nrow = 3, ncol = 3)
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



It appears that out of the 7 traditional variables, hits is the best predictor of runs (since the R2 is highest).

```
(k)
Apply your functions from (h) & (i) to the dataset *without using a for loop* but still using functional
```

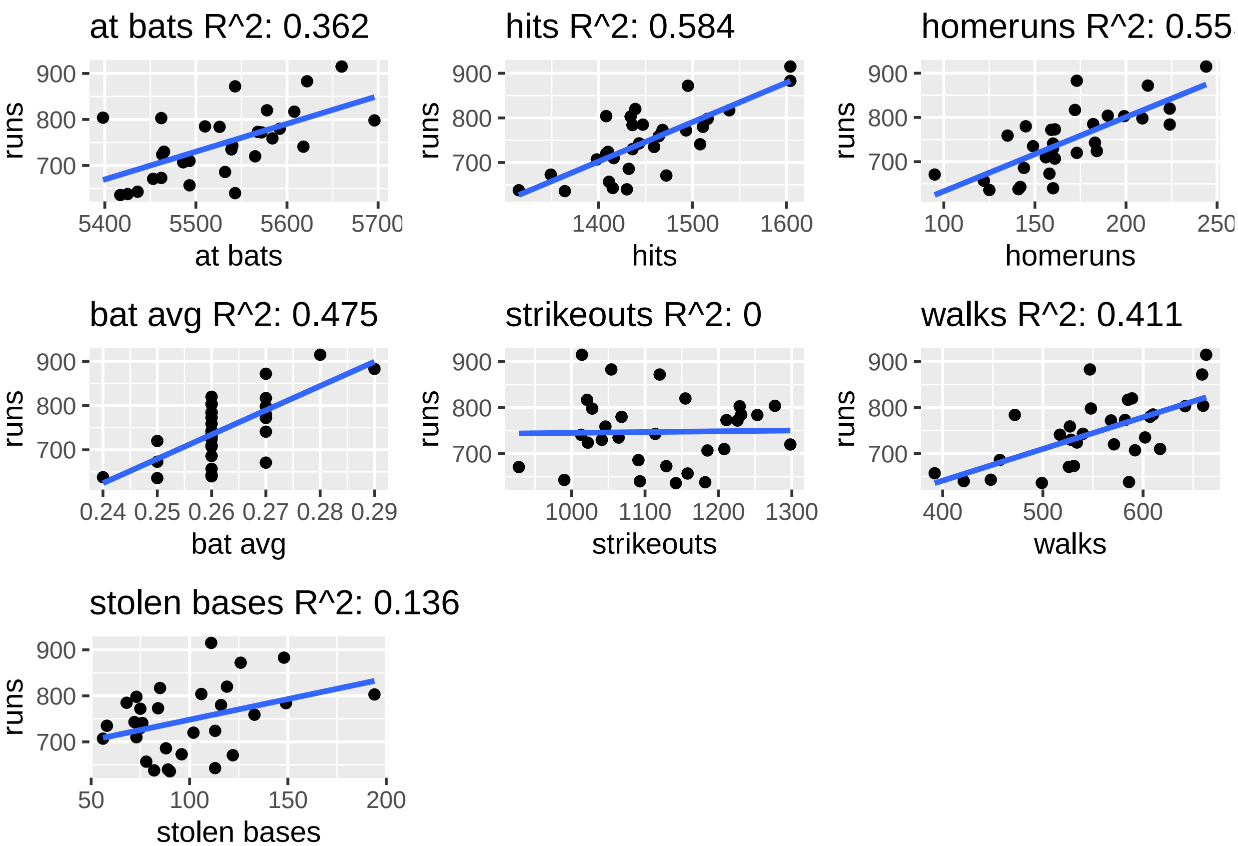
```r
plot_list = list()
r2_list = list()

r2_list = lapply(trad_vars, check_r2)
plot_list = lapply(1:7, function(i) {
    gg = check_plot(trad_vars[i]) + ggtitle(paste(str_replace(trad_vars[i],
        pattern = "_", replacement = " "), "R^2:", r2_list[[i]]))
    return(gg)
})

summarydf = data.frame(vars = trad_vars, r2 = r2_list %>%
    unlist)
```

```
summarydf
```

```
##          vars    r2
## 1     at_bats 0.362
## 2        hits 0.584
## 3    homeruns 0.553
## 4     bat_avg 0.475
## 5  strikeouts 0.000
## 6       walks 0.411
## 7 stolen_bases 0.136
```

```
ggarrange(plotlist = plot_list, nrow = 3, ncol = 3)
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



Same results – they agree with part j.

```
(l) Now imagine you are the manager of the Oakland A's in the early 2000s when Moneyball was set. You su
```

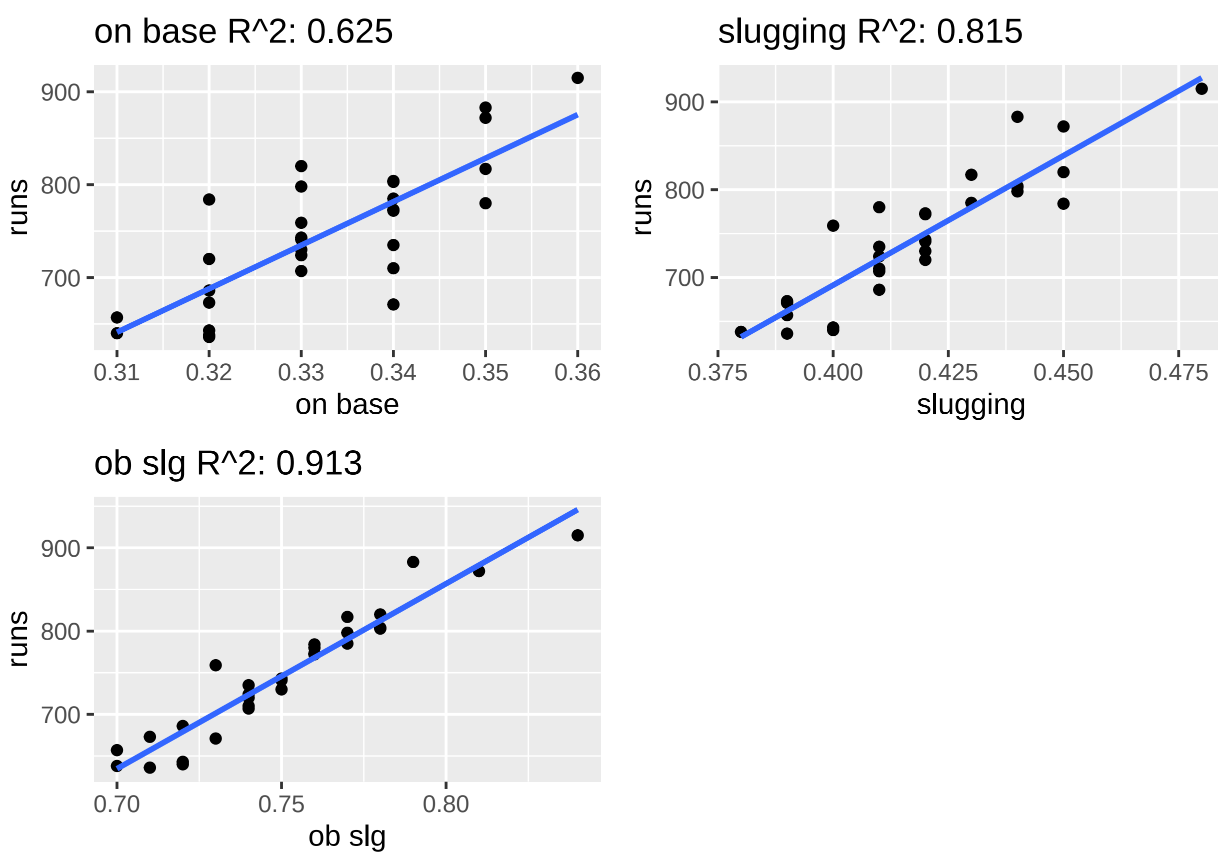
```
non_trad_vars = colnames(mlb)[10:12]

plot_list_nt = list()
r2_list_nt = list()
```

```r
r2_list_nt = lapply(non_trad_vars, check_r2)
plot_list_nt = lapply(1:3, function(i) {
    gg = check_plot(non_trad_vars[i]) + ggtitle(paste(str_replace(non_trad_vars[i],
        pattern = "_", replacement = " "), "R^2:", r2_list_nt[[i]]))
    return(gg)
})

summarydf_nt = data.frame(vars = non_trad_vars, r2 = r2_list_nt %>%
    unlist)
summarydf_nt
```

```
##       vars    r2
## 1  on_base 0.625
## 2 slugging 0.815
## 3   ob_slg 0.913
```

```r
ggarrange(plotlist = plot_list_nt, nrow = 2, ncol = 2)
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```



Based on the R^2 values we see that the non-traditional statistics seem to be better at predicting runs compared to the traditional statistics. On base plus slugging seems to be best with an R^2 of 0.913. This is considerably better than using hits as a predictor. The plot backs this claim up even more.