Lab 9 - Introduction to Linear Regression

Sanaz Saadatifar

4/5

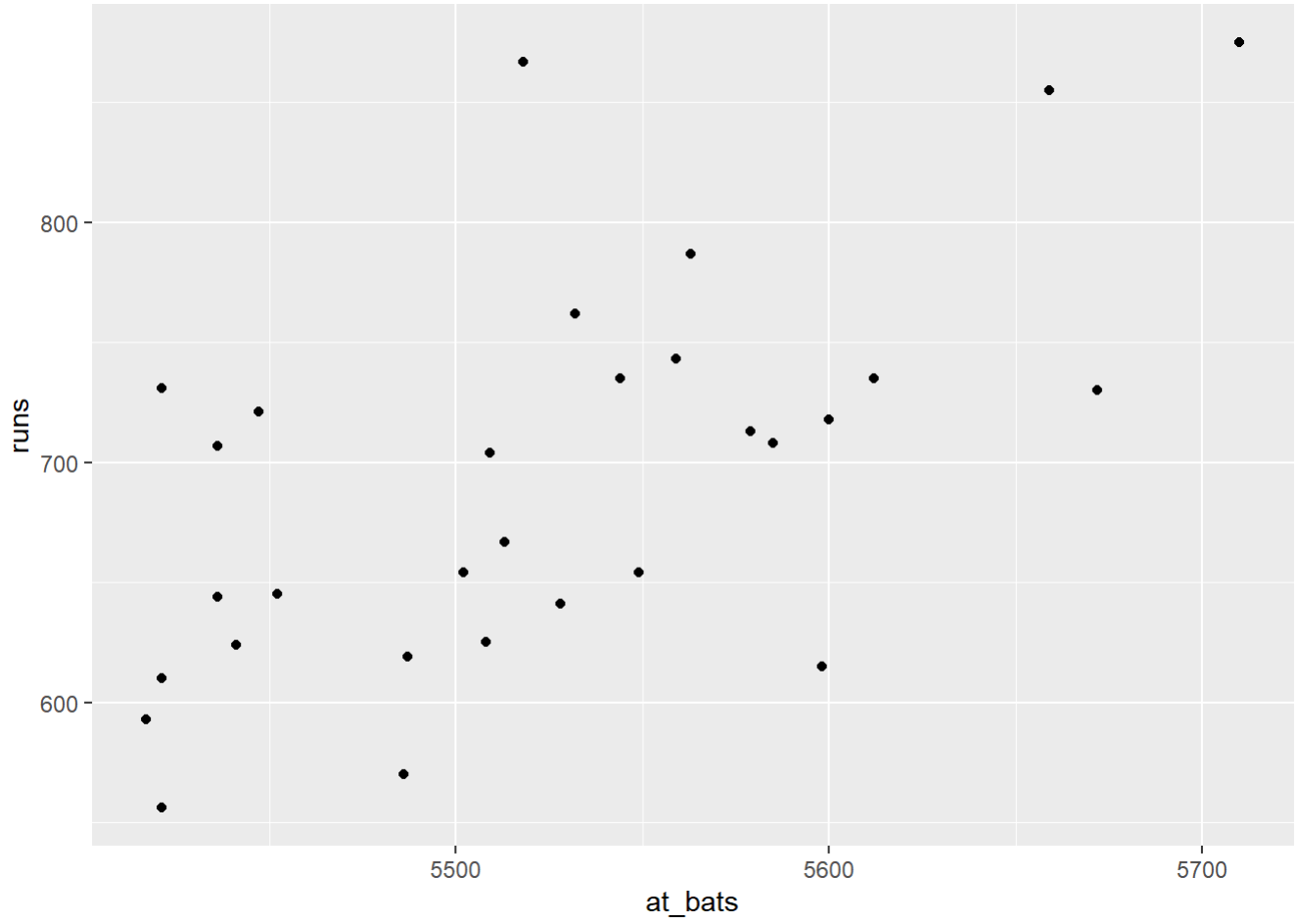
Lab report

Load data

```
load(url("https://dyurovsky.github.io/85309/data/lab9/mlb11.RData"))
```

Exercise 1:

```
ggplot(mlb11, aes(x = at_bats, y = runs)) +
  geom_point()
```



```
mlb11 %>%
  summarise(cor = cor(runs, at_bats)) %>%
  pull()
```

```
## [1] 0.610627
```

To look at the relationship between two numeric variables we can use a scatter plot. Based on this plot of at-bats versus runs, I would feel pretty good using a linear model to predict runs from at-bats.the relationship looks roughly linear and moderately strong.

Exercise 2:

the relationship looks approximately linear and positive more at bats leads to more runs. The relationship is moderately strong, and there does appear to be one unusual point with a moderate at bats but a high number of runs the New York Yankees.

Exercise 3:

```
plot_ss(mlb11, x = at_bats, y = runs)
```

The smallest sum of square I got was 123721.9.others also got a similar range.

Exercise 4:

```
m1 <- lm(runs ~ at_bats, data = mlb11)
summary(m1)
```

```
##
## Call:
## lm(formula = runs ~ at_bats, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -125.58  -47.05  -16.59   54.40  176.87
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2789.2429   853.6957  -3.267 0.002871 **
## at_bats         0.6305     0.1545   4.080 0.000339 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 66.47 on 28 degrees of freedom
## Multiple R-squared:  0.3729, Adjusted R-squared:  0.3505
## F-statistic: 16.65 on 1 and 28 DF,  p-value: 0.0003388
```

```
m2 <- lm(runs ~ homeruns, data = mlb11)
summary(m2)
```

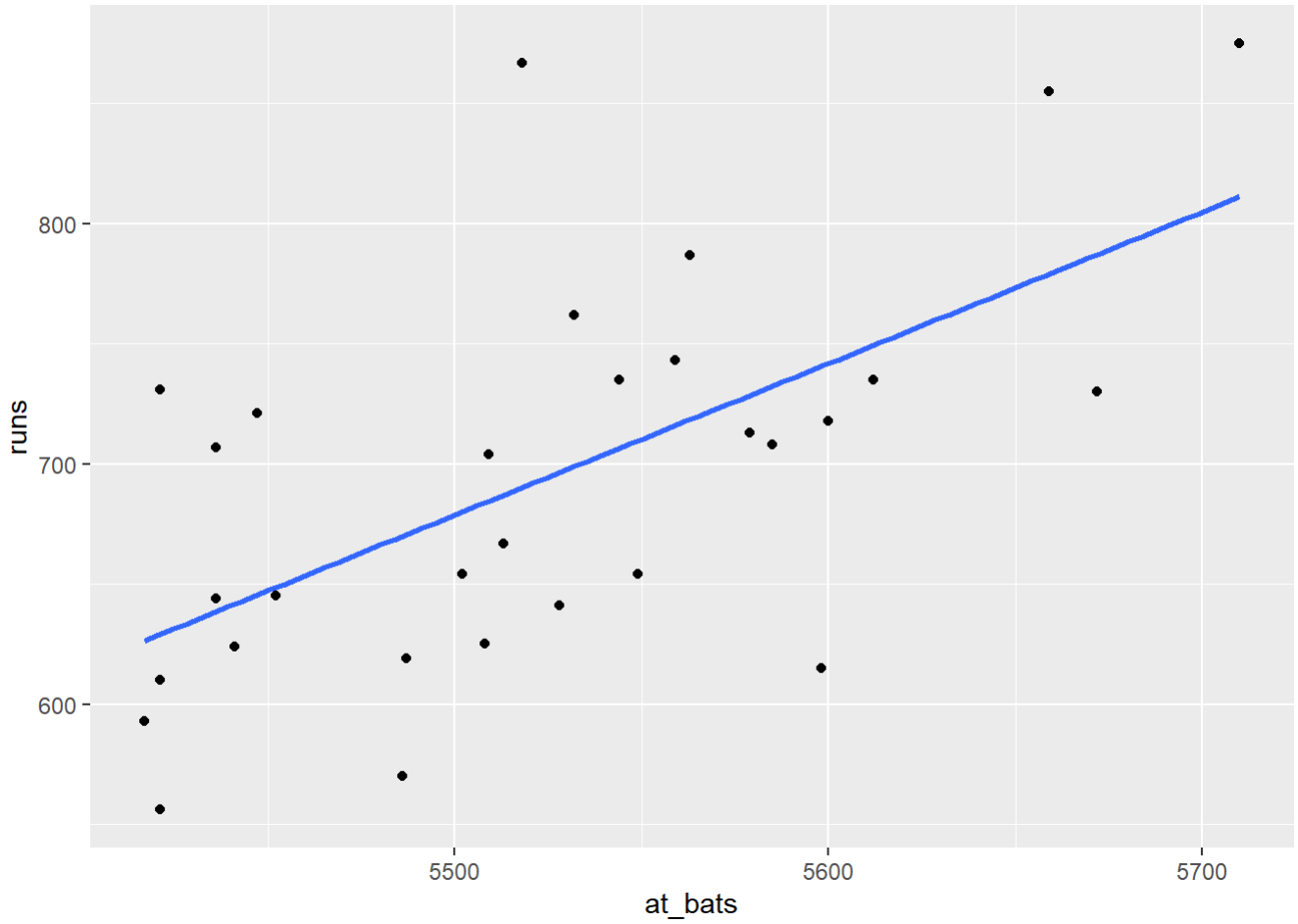
```
##
## Call:
## lm(formula = runs ~ homeruns, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -91.615 -33.410   3.231  24.292 104.631
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 415.2389    41.6779   9.963 1.04e-10 ***
## homeruns      1.8345     0.2677   6.854 1.90e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 51.29 on 28 degrees of freedom
## Multiple R-squared:  0.6266, Adjusted R-squared:  0.6132
## F-statistic: 46.98 on 1 and 28 DF,  p-value: 1.9e-07
```

Runs = 415.24 + 1.83*homeruns this means that for every home runs a team scores, they score about 1.83 total runs.

Exercise 5:

```
ggplot(mlb11, aes(x = at_bats, y = runs)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

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




```
#predicted_5579 <- -2789.2429 + 0.6305 * 5579
#predicted_5579

predict(m1, newdata = tibble(at_bats = 5579))
```

```
##        1
## 728.5955
```

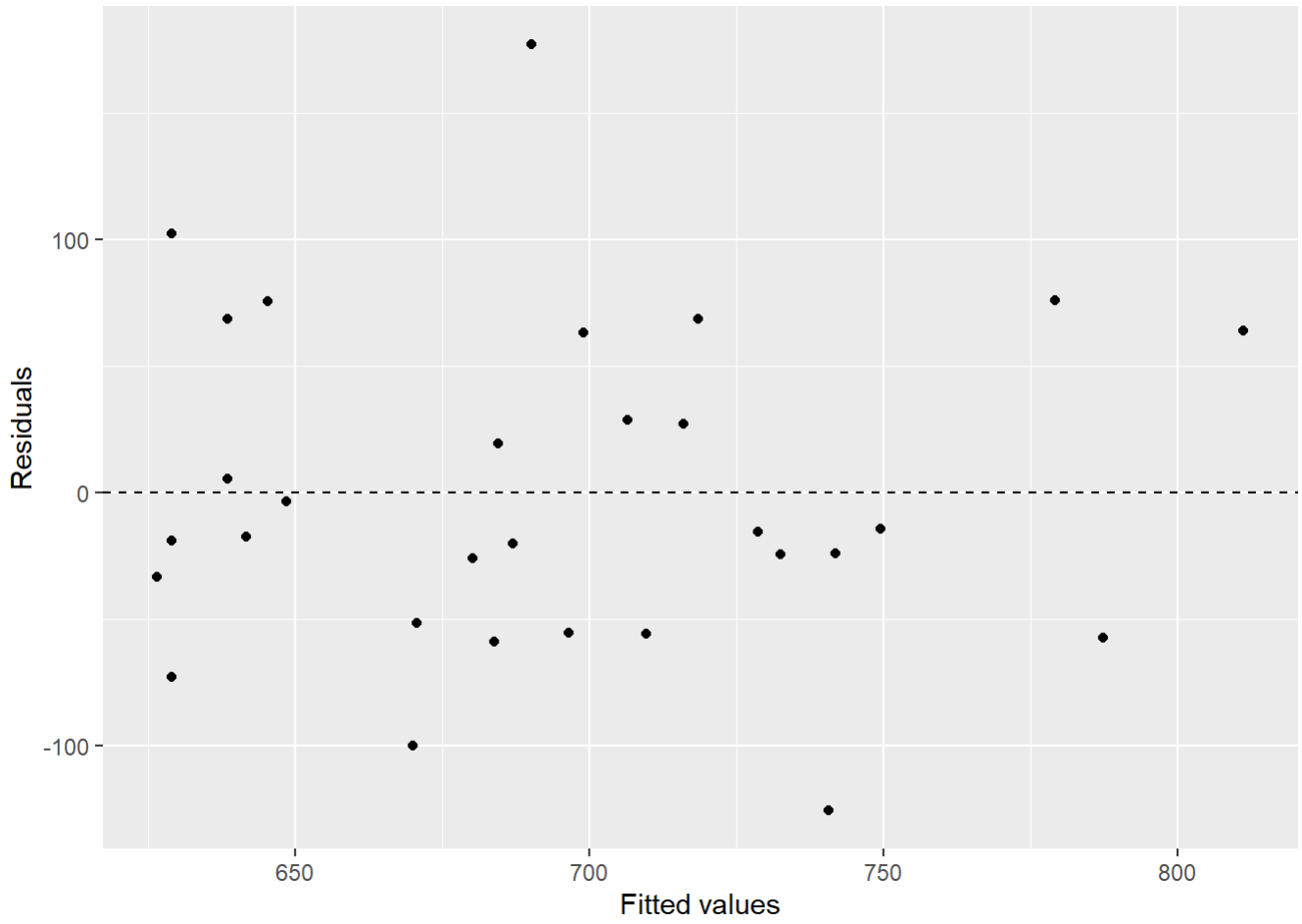
```
empirical_value <- filter(mlb11, at_bats == 5579) %>%
  pull(runs)
empirical_value
```

```
## [1] 713
```

the model predictive value is 728.5955, the true value is 713, so this is an overestimate.

Exercise 6:

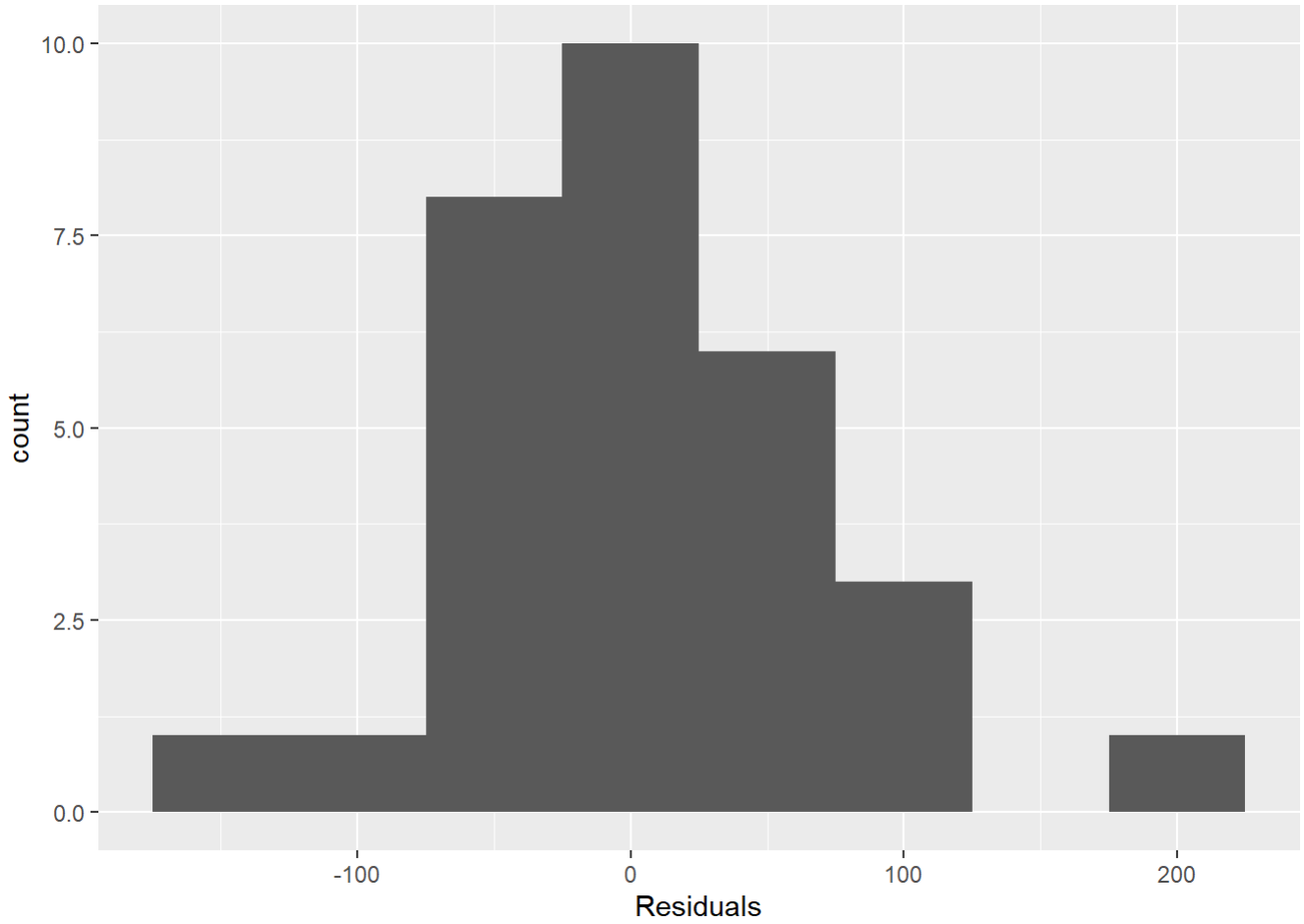
```
m1_residuals <- tibble(x = nrow(mlb11),
                       fitted = fitted(m1),
                       resid = residuals(m1))

ggplot(m1_residuals, aes(x = fitted, y = resid)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  xlab("Fitted values") +
  ylab("Residuals")
```



because there is no obvious pattern in the residuals over the range of predicted values, we think it's OK to assume a linear relationship

Exercise 7:

```
ggplot(m1_residuals, aes(x = resid)) +
  geom_histogram(binwidth = 50) +
  xlab("Residuals")
```



the residuals don't look too skewed, and most of the residuals are near the center of the distribution. So it's probably OK to assume they are nearly normal.

Exercise 8:

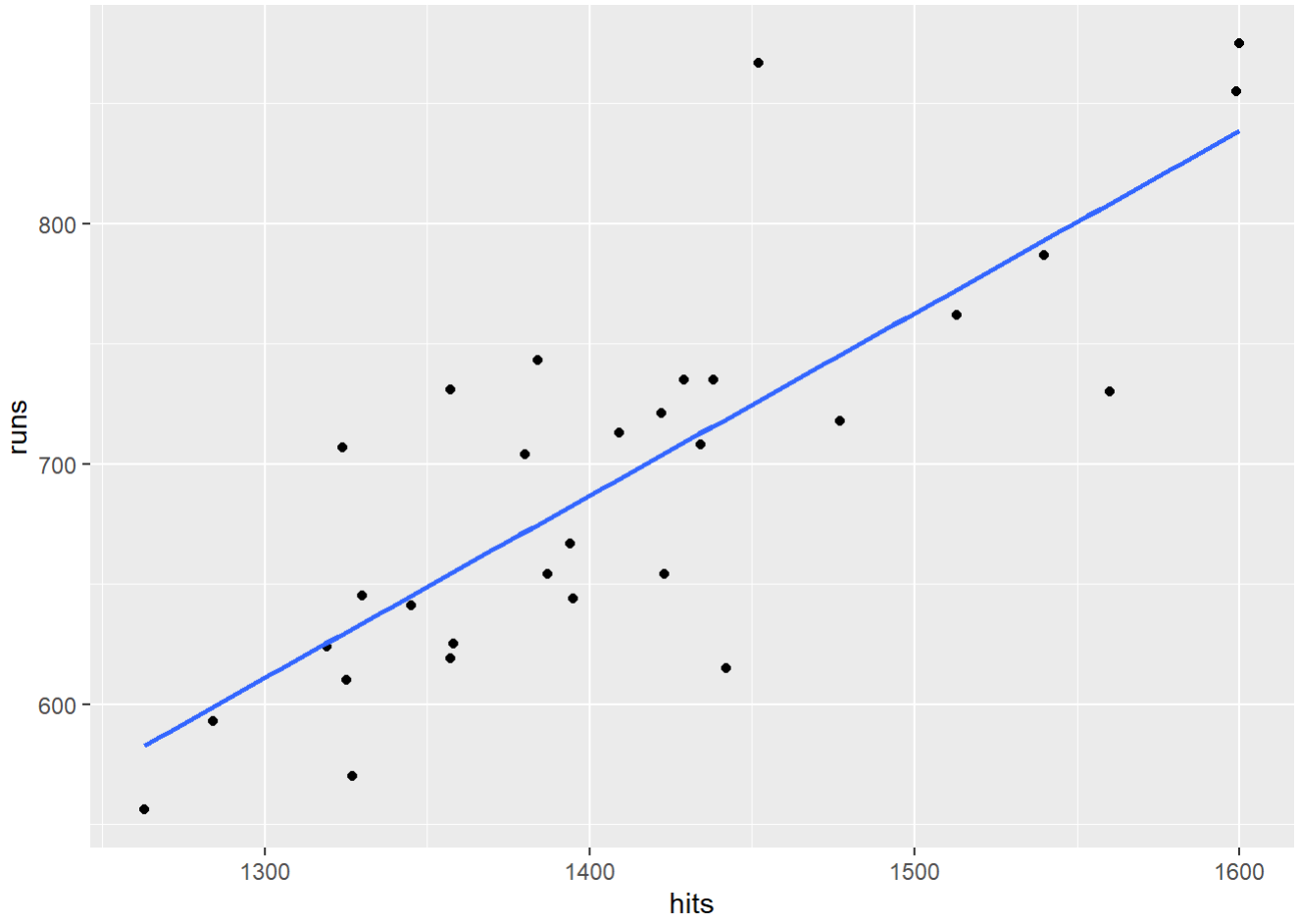
eyeballing the variability of residuals in the plot for exercise 6, we see that they look roughly constant over the range.

More Practice

Exercise 9:

```
ggplot(mlb11, aes(x = hits, y = runs)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

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




```
m_hits <- lm(runs ~ hits, data = mlb11)
summary(m_hits)
```

```
##
## Call:
## lm(formula = runs ~ hits, data = mlb11)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -103.718  -27.179   -5.233   19.322  140.693
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -375.5600   151.1806  -2.484   0.0192 *
## hits           0.7589     0.1071   7.085 1.04e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 50.23 on 28 degrees of freedom
## Multiple R-squared:  0.6419, Adjusted R-squared:  0.6292
## F-statistic:  50.2 on 1 and 28 DF,  p-value: 1.043e-07
```

the relationship between hits and runs looks strongly and positively linear more hits resulting more runs.

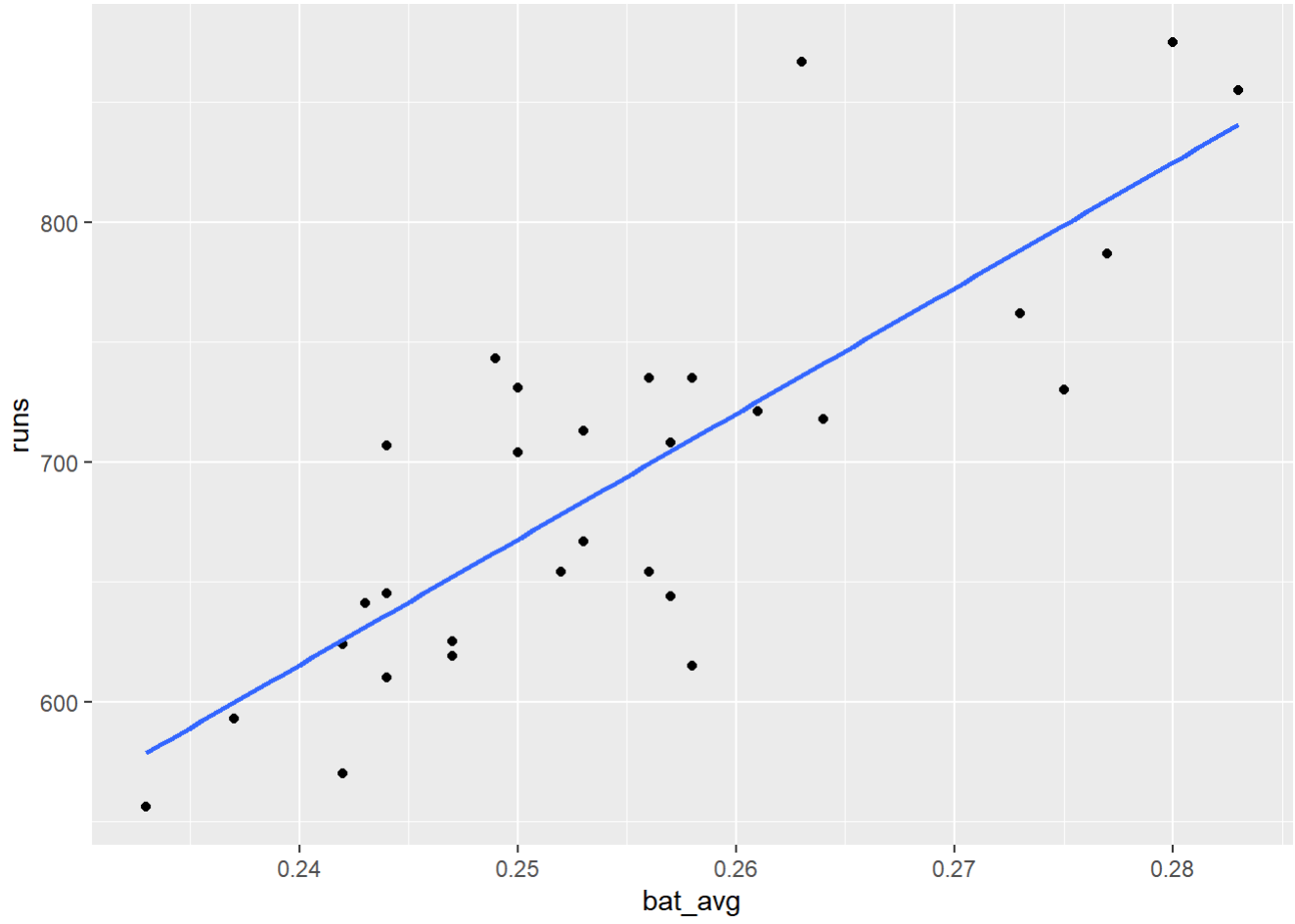
Exercise 10:

based on the code results from exercise 9 for hits and exercise 4 for at-bats we can see that R squared for at-bats is 37.29% while R squared for hits is 64.19% we can conclude that hits's model due to having a higher R squared provides us with better prediction for runs compared to the at-bats because a higher proportion of variance of runs is explained by the hits model.

Exercise 11:

```
#ggplot(mlb11, aes(x = homeruns, y = runs)) +
#  geom_point() +
#  geom_smooth(method = "lm", se = FALSE)


#m_homeruns <- lm(runs ~ homeruns, data = mlb11)
#summary(m_homeruns)



ggplot(mlb11, aes(x = bat_avg, y = runs)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

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



```
m_bat_avg <- lm(runs ~ bat_avg, data = mlb11)
summary(m_bat_avg)
```

```
##
## Call:
## lm(formula = runs ~ bat_avg, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -94.676 -26.303  -5.496  28.482 131.113
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -642.8      183.1  -3.511  0.00153 **
## bat_avg       5242.2      717.3   7.308 5.88e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 49.23 on 28 degrees of freedom
## Multiple R-squared:  0.6561, Adjusted R-squared:  0.6438
## F-statistic: 53.41 on 1 and 28 DF,  p-value: 5.877e-08
```

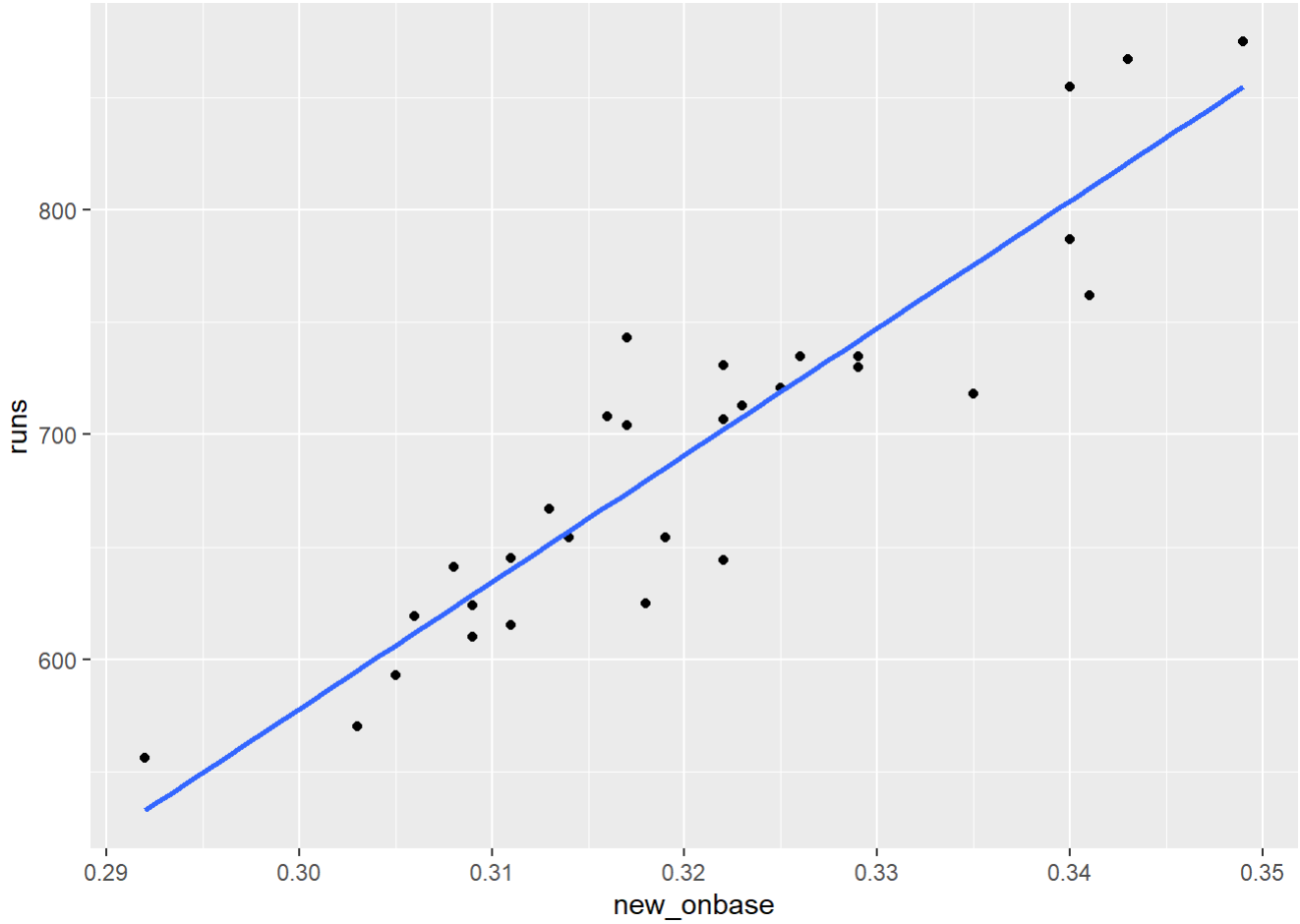
```
#ggplot(mlb11, aes(x = strikeouts, y = runs)) +
#  geom_point() +
#  geom_smooth(method = "lm", se = FALSE)


#m_strikeouts <- lm(runs ~ strikeouts, data = mlb11)
#summary(m_strikeouts)



#ggplot(mlb11, aes(x = stolen_bases, y = runs)) +
#  geom_point() +
#  geom_smooth(method = "lm", se = FALSE)


#m_stolen_bases <- lm(runs ~ stolen_bases, data = mlb11)
#summary(m_stolen_bases)



#ggplot(mlb11, aes(x = wins, y = runs)) +
#  geom_point() +
#  geom_smooth(method = "lm", se = FALSE)


#m_wins <- lm(runs ~ wins, data = mlb11)
#summary(m_wins)
```

Bat_avg best describes and predicts runs because it has very strong and positive linear relationship with runs and also compared to the other variables it has the highest r ^2 which is 65.61%

Exercise 12:

```
ggplot(mlb11, aes(x = new_onbase, y = runs)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

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

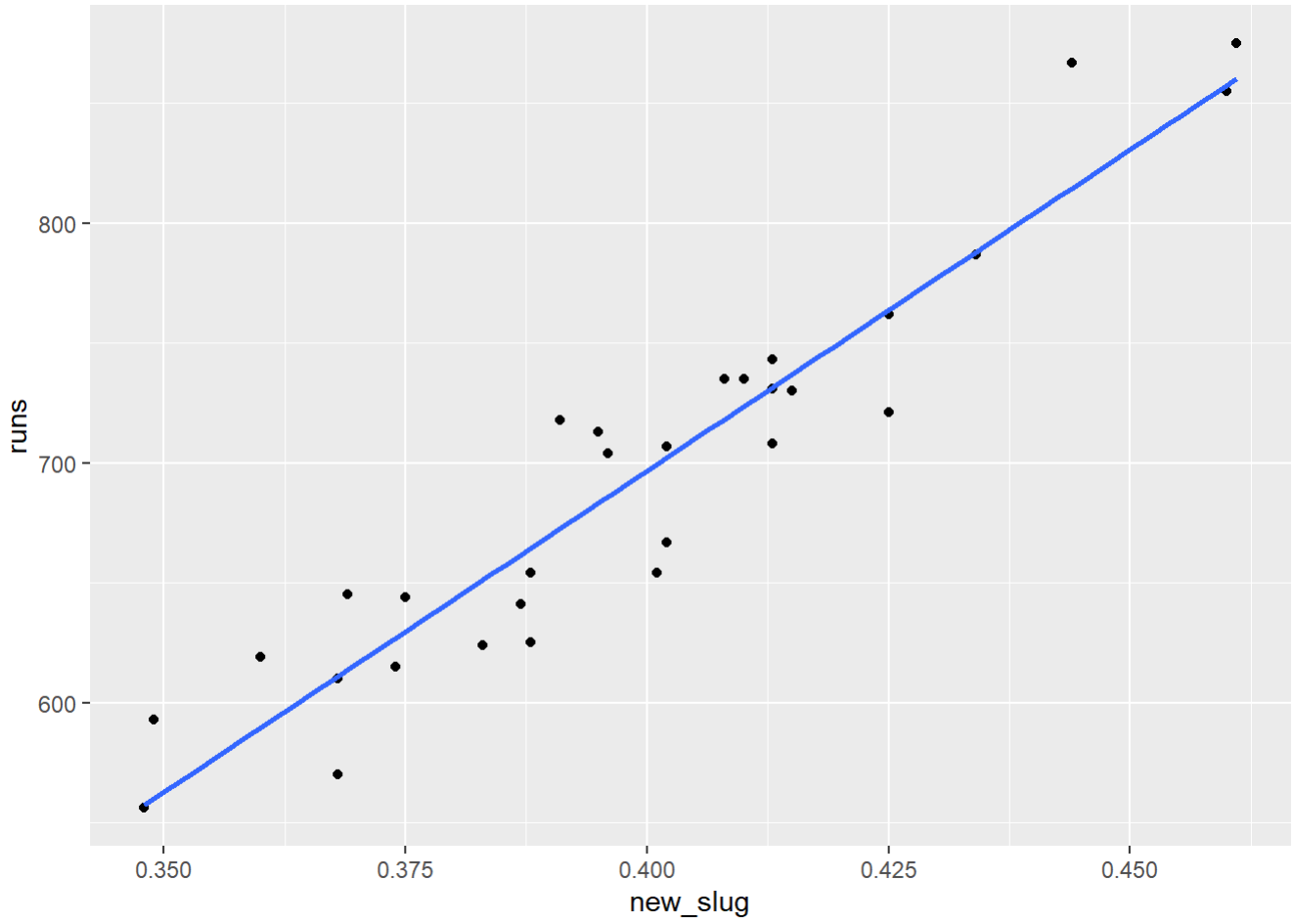



```
m_new_onbase <- lm(runs ~ new_onbase, data = mlb11)
summary(m_new_onbase)
```

```
##
## Call:
## lm(formula = runs ~ new_onbase, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -58.270 -18.335   3.249  19.520  69.002
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1118.4      144.5  -7.741 1.97e-08 ***
## new_onbase    5654.3      450.5  12.552 5.12e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32.61 on 28 degrees of freedom
## Multiple R-squared:  0.8491, Adjusted R-squared:  0.8437
## F-statistic: 157.6 on 1 and 28 DF,  p-value: 5.116e-13
```

```
ggplot(mlb11, aes(x = new_slug, y = runs)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

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

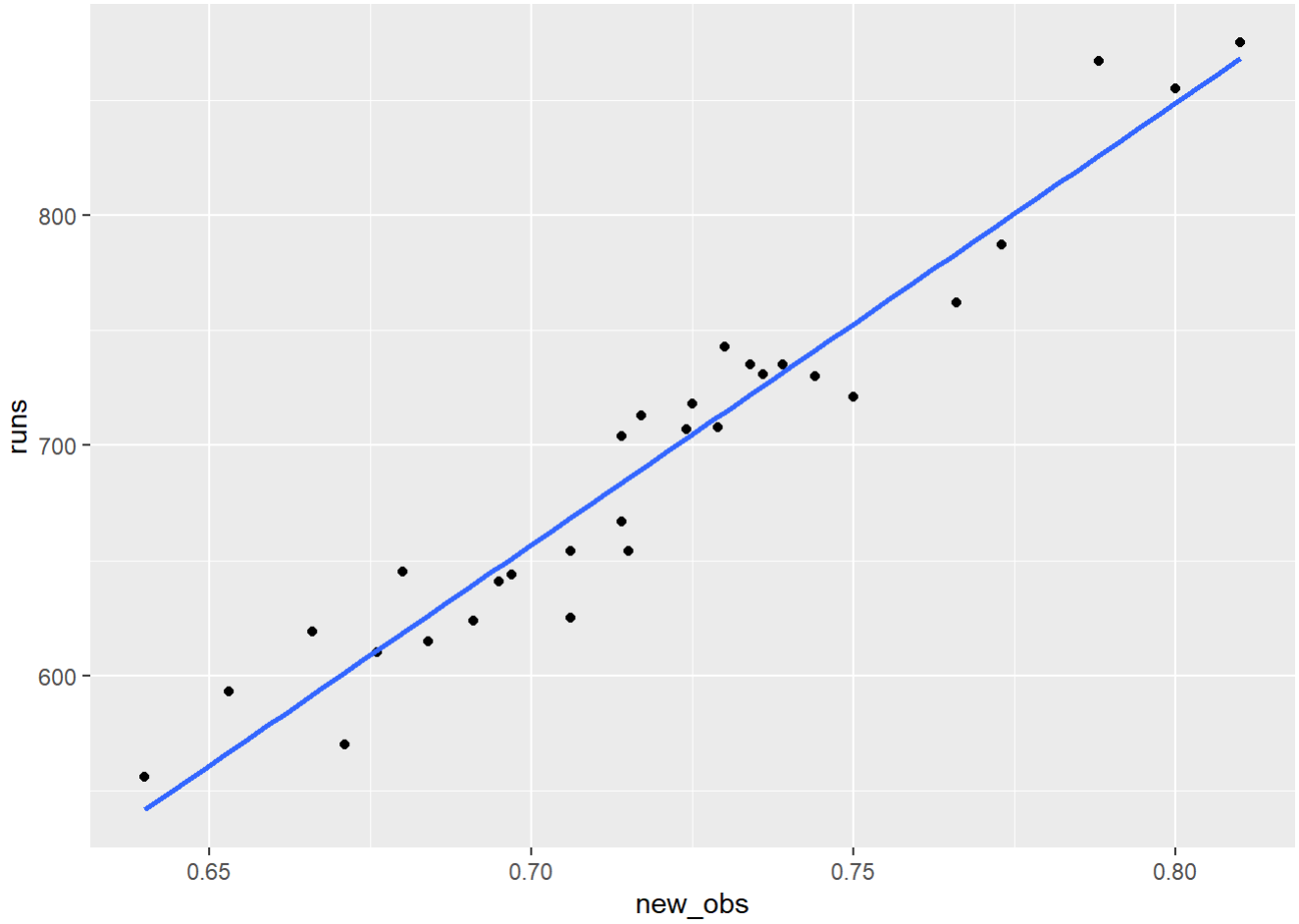



```
m_new_slug <- lm(runs ~ new_slug, data = mlb11)
summary(m_new_slug)
```

```
##
## Call:
## lm(formula = runs ~ new_slug, data = mlb11)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -45.41 -18.66  -0.91  16.29  52.29
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -375.80      68.71   -5.47 7.70e-06 ***
## new_slug     2681.33     171.83   15.61 2.42e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 26.96 on 28 degrees of freedom
## Multiple R-squared:  0.8969, Adjusted R-squared:  0.8932
## F-statistic: 243.5 on 1 and 28 DF,  p-value: 2.42e-15
```

```
ggplot(mlb11, aes(x = new_obs, y = runs)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

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



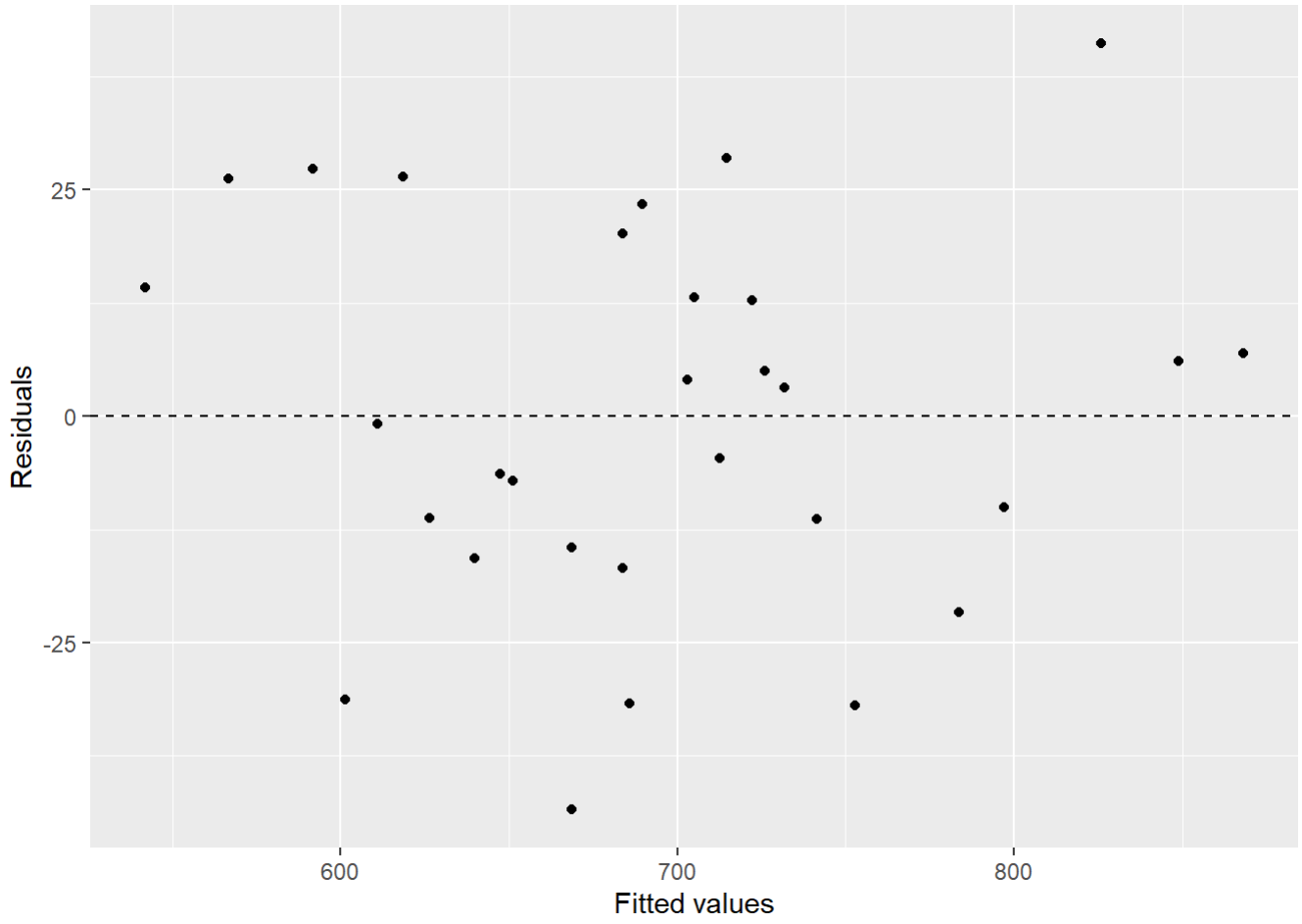

```
m_new_obs <- lm(runs ~ new_obs, data = mlb11)
summary(m_new_obs)
```

```
##
## Call:
## lm(formula = runs ~ new_obs, data = mlb11)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -43.456 -13.690   1.165  13.935  41.156
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -686.61      68.93  -9.962 1.05e-10 ***
## new_obs      1919.36      95.70  20.057  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.41 on 28 degrees of freedom
## Multiple R-squared:  0.9349, Adjusted R-squared:  0.9326
## F-statistic: 402.3 on 1 and 28 DF,  p-value: < 2.2e-16
```

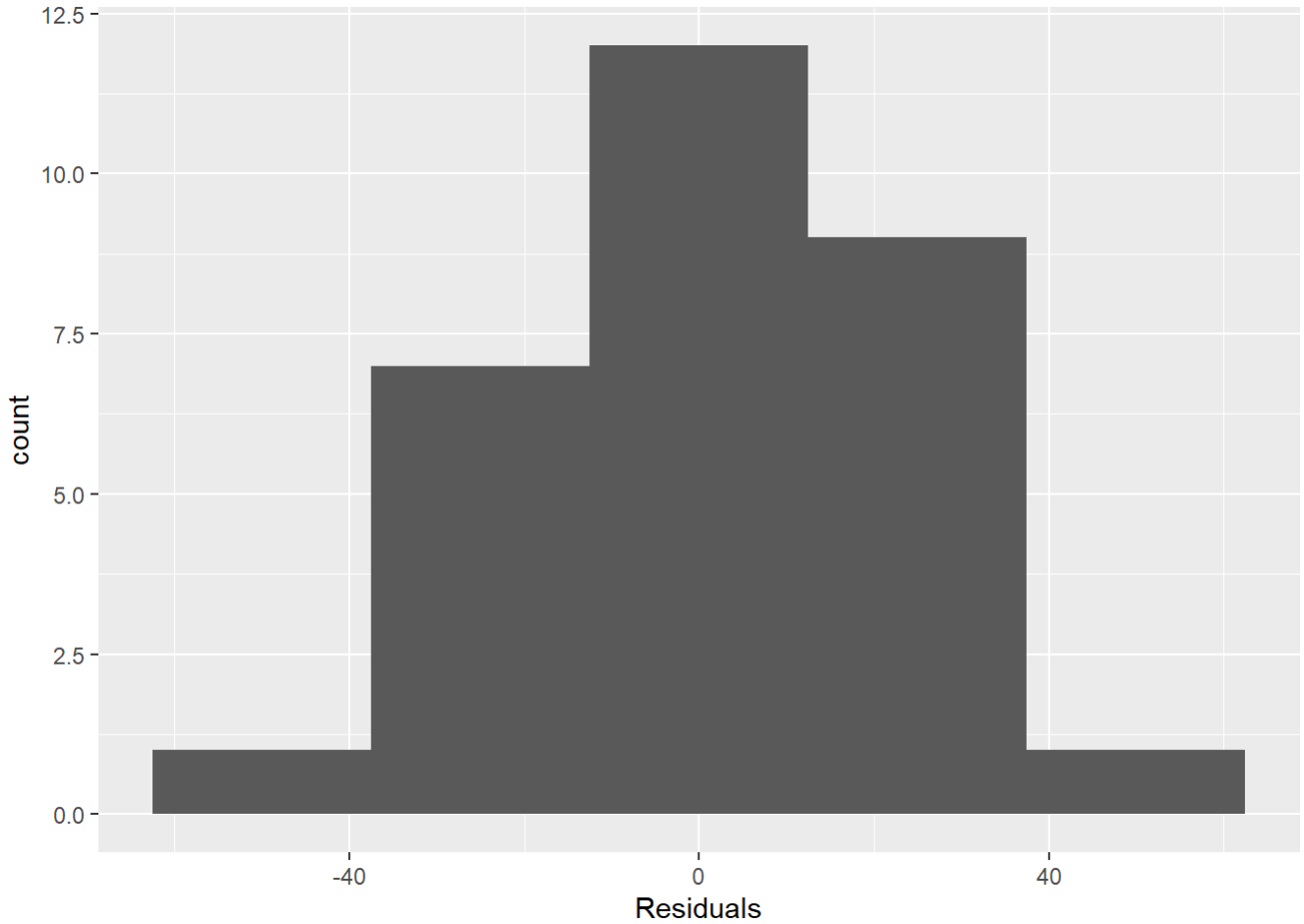
overall the new variables are more efficient and better in predicting the runs compared to ditional 7 valuables. because they all have strong positive linearity compared to the previous seven ones and also there R squares have much higher value compared to the previous ones. between these three new_obs is the best because it has a higher r ^2 , 93.49% , compared to the other two. But overall it makes sense because apparently the new variables are created based on the traditional seven variables so probably the values that we're not that efficient in prediction of run's are not included a new ones.

Exercise 13:

```
m_new_obs_residuals <- tibble(x = nrow(mlb11),
                        fitted = fitted(m_new_obs),
                        resid = residuals(m_new_obs))

ggplot(m_new_obs_residuals, aes(x = fitted, y = resid)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed") +
  xlab("Fitted values") +
  ylab("Residuals")
```




```
ggplot(m_new_obs_residuals, aes(x = resid)) +
  geom_histogram(binwidth = 25) +
  xlab("Residuals")
```



because there is no obvious pattern in the residuals over the range of predicted values, we think it's OK to assume a linear relationship the residuals don't look too skewed, and most of the residuals are near the center of the distribution. So it's probably OK to assume they are nearly normal. eyeballing the variability of residuals in the plot, we see that they look roughly constant over the range