Analysis of Vienna City Marathon 2023 data

Mikhail Vsevolodov 11837434

0)Preparation for data analysis

Adding necesarry libraries

```
library(psych)
library(knitr)
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```
library(patchwork)
```

Loading of data:

```
marathon <- read.csv(file = "./marathon.csv", header = TRUE, sep = ";", stringsAsFactors = FALSE)
head(marathon, 3)
```

```
##   rank born country category time_total time_1half time_2half time_0to5km
## 1    1   93     KEN     W-30   02:24:12   01:12:05   01:12:08    00:17:03
## 2    2   88     KEN     W-35   02:24:25   01:12:05   01:12:20    00:17:03
## 3    3   97     ETH      W-H   02:24:50   01:12:05   01:12:46    00:17:03
##   time_5to10km time_10to15km time_15to20km time_20to25km time_25to30km
## 1     00:16:46      00:17:35      00:16:57      00:16:47      00:16:52
## 2     00:16:45      00:17:34      00:16:59      00:16:47      00:16:53
## 3     00:16:46      00:17:33      00:16:59      00:16:48      00:16:53
##   time_30to35km time_35to40km time_40to42km
## 1      00:16:55      00:17:22      00:07:58
## 2      00:16:52      00:17:34      00:08:01
## 3      00:17:33      00:17:39      00:07:41
```

1) Data preprocessing and cleaning

1.1)Data preprocessing

Data preprocessing is key for comfortable and correct data analysis. First of all we want to know the datatypes of the columns:

```
temp <- marathon[, c("rank","born","country","category","time_total","time_1half","time_0to5km")]
sapply(temp, class)
```

```
##        rank        born     country    category  time_total  time_1half
##   "integer"   "integer" "character" "character" "character" "character"
## time_0to5km
## "character"
```

As we see can see almost all columns have "character" type which is not practical for our analysis. Hence we would like to transform country and category into categorical data and time intervals into number variables which report the corresponding time (in minutes).

```
marathon$category <- factor(marathon$category)
marathon$country <- factor(marathon$country)

convert_time_to_minutes <- function(time_string) {
  time <- strptime(time_string, format = "%H:%M:%S")
  total_minutes <- as.numeric(time$hour) * 60 + as.numeric(time$min) + as.numeric(time$sec) / 60
  return(total_minutes)
}

marathon$time_total <- sapply(marathon$time_total, convert_time_to_minutes)
marathon$time_1half <- sapply(marathon$time_1half, convert_time_to_minutes)
marathon$time_2half <- sapply(marathon$time_2half, convert_time_to_minutes)
marathon$time_0to5km <- sapply(marathon$time_0to5km, convert_time_to_minutes)
marathon$time_5to10km <- sapply(marathon$time_5to10km, convert_time_to_minutes)
marathon$time_10to15km <- sapply(marathon$time_10to15km, convert_time_to_minutes)
marathon$time_15to20km <- sapply(marathon$time_15to20km, convert_time_to_minutes)
marathon$time_20to25km <- sapply(marathon$time_20to25km, convert_time_to_minutes)
marathon$time_25to30km <- sapply(marathon$time_25to30km, convert_time_to_minutes)
marathon$time_30to35km <- sapply(marathon$time_30to35km, convert_time_to_minutes)
marathon$time_35to40km <- sapply(marathon$time_35to40km, convert_time_to_minutes)
marathon$time_40to42km <- sapply(marathon$time_40to42km, convert_time_to_minutes)
```

The Born column, although it carries information about the age of the participants, can nevertheless be interpreted rather ambiguously.Therefore we would like to bring this to age value. I will assume that the minimal required age of participant is 16. To do this we define and apply a function which convert year to age and compute the age on 2023:

```
year_to_age <- function(year){
  if (year > 7) {
    return(2023 - (1900 + year))
  } else {
    return(2023 - (2000 + year))
  }
}
marathon$born <- sapply(marathon$born, year_to_age)
names(marathon)[names(marathon) == "born"] <- "age"
head(marathon, 3)
```

```
##   rank age country category time_total time_1half time_2half time_0to5km
## 1    1  30     KEN     W-30   144.2000   72.08333   72.13333       17.05
## 2    2  35     KEN     W-35   144.4167   72.08333   72.33333       17.05
## 3    3  26     ETH      W-H   144.8333   72.08333   72.76667       17.05
##   time_5to10km time_10to15km time_15to20km time_20to25km time_25to30km
## 1     16.76667      17.58333      16.95000      16.78333      16.86667
## 2     16.75000      17.56667      16.98333      16.78333      16.88333
## 3     16.76667      17.55000      16.98333      16.80000      16.88333
##   time_30to35km time_35to40km time_40to42km
```

```
## 1      16.91667      17.36667      7.966667
## 2      16.86667      17.56667      8.016667
## 3      17.55000      17.65000      7.683333
```

1.2)Data cleaning

Data cleaning helps us get rid of broken data, which will help us avoid errors in further analysis. At this point, I will not delve deeply into the search for errors in the data, but I will give an example of such an error in the columns that we will subsequently use for our analysis.

```
marathon[marathon$time_2half == 0,][c(0,1,2,5,6,7)]
```

```
##      rank age time_total time_1half time_2half
## 889   895  39   267.4000          0          0
## 1366 1375  71   312.5667          0          0
## 1408 1417  55   320.5500          0          0
```

In these lines we can observe an obvious discrepancy between the values, since the values time_1half and time_2half do not sum up to time_total. However, we will not drop these lines since the information we need(time_total) is retained.

Now we are ready to start out analysis.

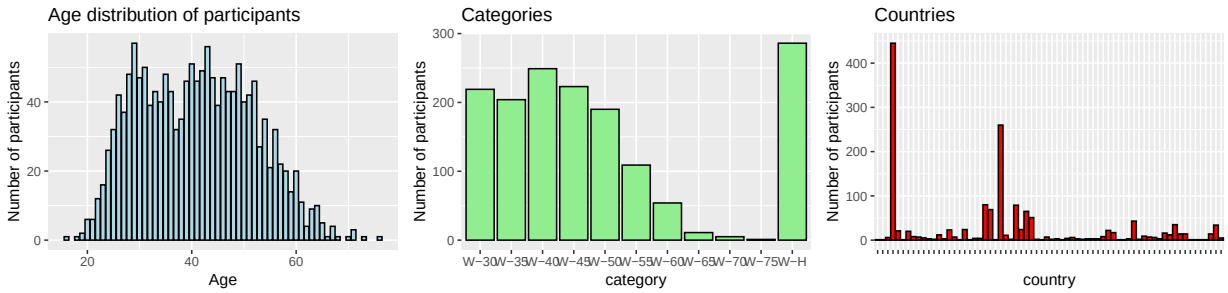
2) Empirical analysis

Before we begin to analyze the numerical characteristics of the participants, we will consider the categorical ones (we will include age in this group). We would like to know the age and national distribution of participants.

categorical values

In the case of categorical values, the best idea is to show the graph as a barplot.

```
p1 <- ggplot(data = marathon, aes(x = age)) +
  geom_bar(fill = "lightblue", color = "black") +
  labs(title = "Age distribution of participants",
       x = "Age",
       y = "Number of participants")
p2 <- ggplot(data = marathon, aes(x = category)) +
  geom_bar(fill = "lightgreen", color = "black") +
  labs(title = "Categories",
       x = "category",
       y = "Number of participants")
p3 <- ggplot(data = marathon, aes(x = country)) +
  geom_bar(fill = "red", color = "black") +
  labs(title = "Countries",
       x = "country",
       y = "Number of participants") + theme(axis.text.x=element_blank())
p1 + p2 + p3
```



In this graph we observe 2 pronounced peaks in the area of 29 and 43 years. In addition to this, a slight decline in the region of 37 - 38 years is interesting, due to the fact that participation in marathons is associated with interest in such events. It is also easy to notice the most common age groups: 25 - 36 and 39 - 52. There is also a noticeably smaller number of participants aged <25 and >50. : The age distribution appears to be relatively symmetrical with a slight shift to the left. It seems that in some age groups the number of participants is small or equal to zero, especially among people over 70. For category, the coding "W-. . . " denotes age groups of female participants. The distribution generally corresponds to the conclusions we made earlier, with the exception of the group W-H. W-H apparently represents a separate category not related to age. The "age"-category with the most participants is W-40.

From the graph showing the distribution of participants by country, we see that there are 2 pronounced peaks, and 5 bars with significant number of participants, which means that about half of the participants are citizens of 5 countries. We will demonstrate number of countries and the countries with the majority of participants:

```
cat("Number of countries: ", length(table(marathon$country)))
```

```
## Number of countries:  68
```

```
head(sort(table(marathon$country), decreasing = TRUE), 20)
```

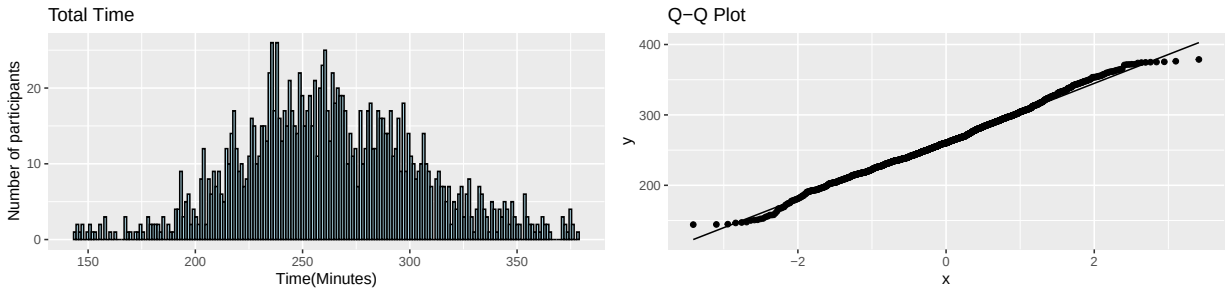
```
##
## AUT GER FRA HUN GBR ISR ITA POL SUI USA ESP IRL CZE NED BEL BRA NOR SLO SVK SWE
## 445 260  80  79  69  65  51  43  35  34  24  24  23  22  21  20  17  16  14  14
```

As expected, most of the participants are residents of Austria and Germany.

Numerical values

For numerical values we will use histograms and QQ-Plot:

```
p3 <- ggplot(data = marathon, aes(x = time_total)) +
  geom_histogram(bins =200, fill = "lightblue", color = "black") +
  labs(title = "Total Time", x = "Time(Minutes)",y = "Number of participants")
p4 <- qq_plot <- ggplot(data = marathon, aes(sample = time_total)) +
  stat_qq() + stat_qq_line() + ggtitle("Q-Q Plot")
p3+p4
```



In the histogram we see a fairly symmetrical distribution.The histogram appears to have multiple peaks and troughs, suggesting it is multimodal rather than unimodal. I would attribute this to the desire of people to move in groups. We are seeing some tendency towards a normal distribution. To make sure of this let's look at Q-Q-Plot: The deviation from the line suggests that the sample data may not be perfectly normally distributed. There is a noticeable "S" shape, with the tails of the distribution (the leftmost and rightmost points on the plot) deviating from the line. The points that deviate significantly from the line in the tails might also suggest potential outliers in the data.We do not observe a significant skewness.

Now let's look at some statistics of numerical values:

```
describe(marathon[, c("time_total","time_1half","time_2half","time_0to5km","time_5to10km","time_10to15k
```

```
##                  mean    sd median trimmed   mad   min    max  range  skew
## time_total    262.83 41.78 260.23  261.94 40.67 144.2 378.77 234.57  0.15
## time_1half    125.85 19.39 125.20  125.76 17.12   0.0 183.82 183.82 -0.41
## time_2half    136.41 24.81 134.78  135.75 23.97   0.0 210.73 210.73 -0.03
## time_0to5km    28.91  4.39  28.92   28.99  3.61   0.0  47.85  47.85 -1.02
## time_5to10km   29.11  4.55  28.97   29.17  3.71   0.0  46.93  46.93 -1.05
## time_10to15km  30.92  5.27  30.77   30.87  4.45   0.0  50.25  50.25 -0.46
## time_15to20km  30.09  5.41  29.77   30.01  4.65   0.0  52.72  52.72 -0.50
## time_20to25km  30.41  5.60  30.08   30.31  4.65   0.0  52.23  52.23 -0.52
## time_25to30km  31.63  6.14  31.17   31.41  5.44   0.0  61.35  61.35 -0.07
```

The first half of the marathon (125.85 minutes) is less than the second half (136.41 minutes), which may indicate that participants start the race with more energy and get slower as fatigue accumulates. The same with 5km intervals. The minimum race time is 144.2 minutes, and the maximum is 378.77 minutes, what shows a wide variety in the performance of the participants. Total time has a positive skewness (0.15), indicating a small number of participants with very long race times.

3)Hypothesis testing

We take a closer look at the total running time and assume for this exercise that it follows an approximate normal distribution. We will consider the claim $\mu = 289$. Based on our data we are skeptical about this statement. Let's apply t-test to test the hypothesis: Null hypothesis (H0): $\mu = 289$ minutes (μ = the mean running time) Alternative hypothesis (H1): $\mu < 289$ minutes

```
null.mean <- 289
n = length(marathon$time_total)
sample.mean <-  sum(marathon$time_total) / n
sample.variance <- (1/(n-1)) * sum((marathon$time_total - sample.mean)^2)
t.statistic = (sample.mean - null.mean)/(sqrt(sample.variance/n))
print(t.statistic)
```

```
## [1] -24.67085
```

```
p.val <-pt(t.statistic, df = n - 1)
print(p.val)
```

```
## [1] 6.245486e-114
```

At this point we already can see that the p-value is much less than the typical alpha level of 0.05, so we can reject the null hypothesis. Lets run t.test function to recheck the result:

```
t_test_result <- t.test(marathon$time_total, alternative = "less", mu = 289)
print(t_test_result)
```

```
##
##  One Sample t-test
```

```
##
## data:  marathon$time_total
## t = -24.671, df = 1550, p-value < 2.2e-16
## alternative hypothesis: true mean is less than 289
## 95 percent confidence interval:
##      -Inf 264.5737
## sample estimates:
## mean of x
##  262.8277
```

As we can see there is sufficient evidence to conclude that the true mean is less than 289 minutes. The confidence interval further supports this conclusion, as it indicates that we can be 95% confident that the true mean lies below 264.5737 minutes, which is well below 289 minutes.The negative t-value indicates that the sample mean is less than the hypothesized mean, and given the very small p-value, the result is statistically significant. We can conclude that the real mean value is significantly lower than considered claim.

4)Linear regression model

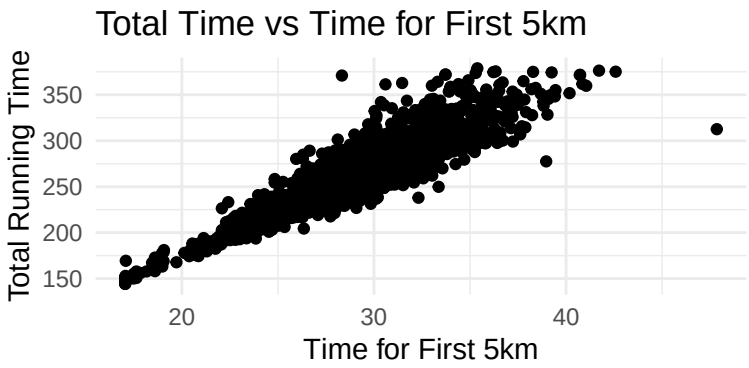
We now want to model the running time time total using the first split time time 0to5km. As we discussed in paragraph "1.2)Data cleaning" in our data there is erroneous measurements. In order to avoid distortions in our linear model, we will drop them:

```
marathon <- marathon[marathon$time_0to5km != 0,]
length(marathon$rank)
```

```
## [1] 1545
```

As we can see we got rid of 6 measurements. Now can visualize the relationship between these two variables using a scatter plot:

```
ggplot(marathon, aes(x = time_0to5km, y = time_total)) +
  geom_point() +
  labs(title = "Total Time vs Time for First 5km",
       x = "Time for First 5km",
       y = "Total Running Time") +
  theme_minimal()
```



There is clear positive correlation between the time taken for the first 5km and the total running time. We can explain it with assumption,that participants who take longer to complete the first 5km tend to have longer overall running times.The lower density of points at the upper end indicates greater variability in total running time among runners who are slower in the first 5km. There are also some outliers. The outlier on the left side of the graph is especially interesting, as it seems particularly atypical to have a slower pace in the first 5 kilometers. The trend seems to be linear.

Further we will estimate the correlation coefficient:

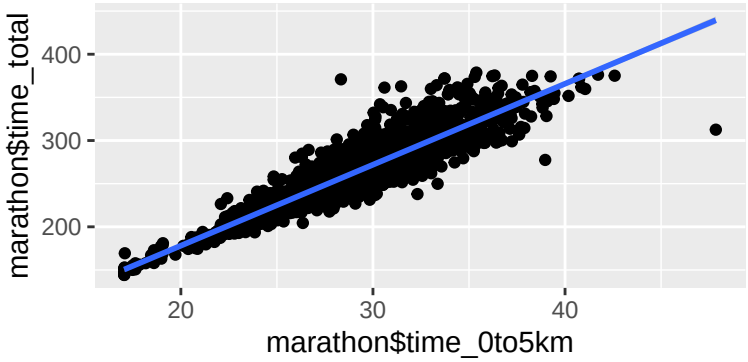
```
correlation <- cor(marathon$time_0to5km, marathon$time_total)
print(correlation)
```

```
## [1] 0.9034136
```

Such correlation coefficient signalizes a very strong positive linear relationship between first 5km and the total running time, which implies that the time for the first 5 km is a good predictor for a total running time. Lets build the linear model and make a prediction with given value:

```
model <- lm(time_total ~ time_0to5km, data=marathon)
ggplot(marathon,aes(marathon$time_0to5km, marathon$time_total)) +
  geom_point() +
  geom_smooth(method='lm')
```

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



Now we can make a prediction based on our linear model:

```
predicted_time <- predict(model, newdata=data.frame(time_0to5km=26))
predicted_time
```

```
##        1
## 234.4335
```

According to our linear model person who runs for the first 5 km 26 minutes will need 234.4 minutes as total running time.

Next we could expand our linear model and look at the result:

```
extended_model <- lm(time_total ~ time_0to5km + age + time_1half, data=marathon)
summary(extended_model)$coefficients
```

```
##                Estimate Std. Error   t value      Pr(>|t|)
## (Intercept) -16.0834056 2.41940065 -6.647682  4.120448e-11
## time_0to5km   1.9929660 0.19505438 10.217489  9.290543e-24
## age           0.1624865 0.02918931  5.566644  3.058354e-08
## time_1half    1.7012957 0.04178491 40.715550 1.159903e-246
```

The Pr(>|t|) column shows the p-value for predictors. We can see that for each predictor p-value if significantly less than 0.05, which implies statistical significance of them.

Lets remove another erroneous measurement which reports time_1half = 0 minutes and update the model.

```
marathon <- marathon[marathon$time_1half != 0,]
updated_model <- lm(time_total ~ time_0to5km + age + time_1half, data=marathon)
summary(updated_model)$coefficients
```

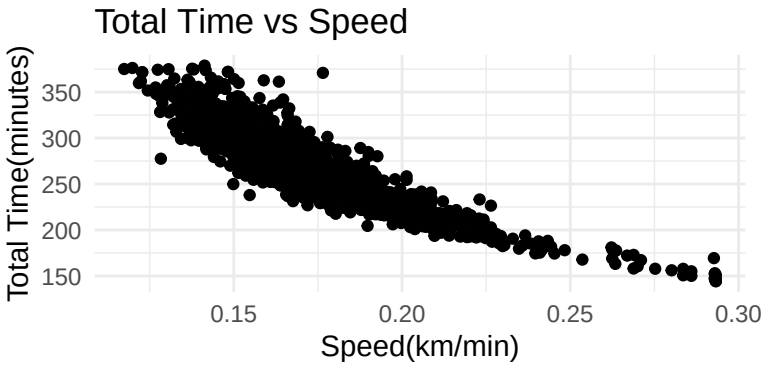
```
##                Estimate Std. Error    t value     Pr(>|t|)
## (Intercept) -3.54979928 1.83311035  -1.936490 5.299051e-02
## time_0to5km -3.17585498 0.20595610 -15.420058 5.350351e-50
## age          0.02562442 0.02204092   1.162584 2.451784e-01
## time_1half   2.83435852 0.04464153  63.491515 0.000000e+00
```

This time the predictor "age" seems to be not statistically significant, because of it's p-value > 0.05.

5)Variable transformation

Define a new variable speed measuring the average velocity on the first 5 kilometers and plot total time against it:

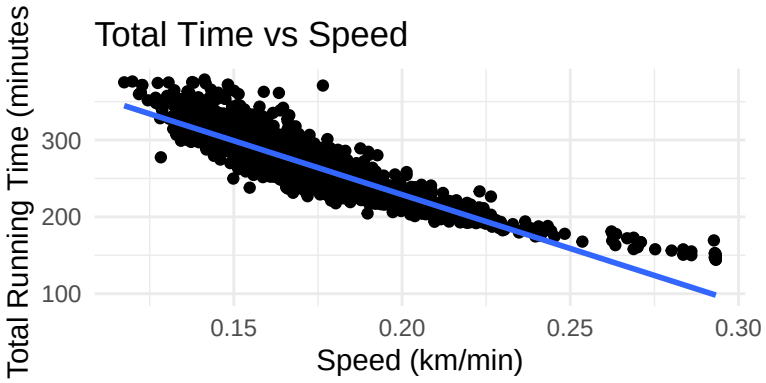
```
marathon$speed <- 5 / marathon$time_0to5km
ggplot(marathon, aes(x = speed, y = time_total)) +
  geom_point() +
  labs(title = "Total Time vs Speed",
       x = "Speed(km/min)", y = "Total Time(minutes)") +
  theme_minimal()
```



We can see non-linear curve-like pattern, a linear model might not be the best fit for this data. Let's show it:

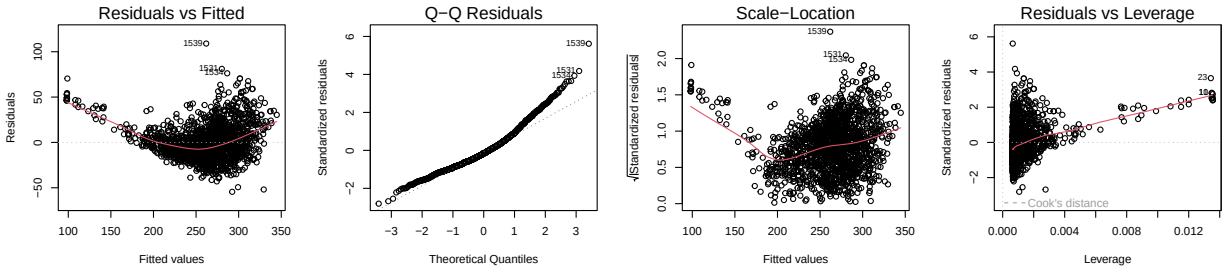
```
model_speed <- lm(time_total ~ speed, data = marathon)
ggplot(marathon, aes(x = speed, y = time_total)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Total Time vs Speed",
       x = "Speed (km/min)", y = "Total Running Time (minutes)") +
  theme_minimal()
```

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



Let's take a look at diagnostic plots:

```
par(mfrow = c(1, 4))
plot(model_speed)
```

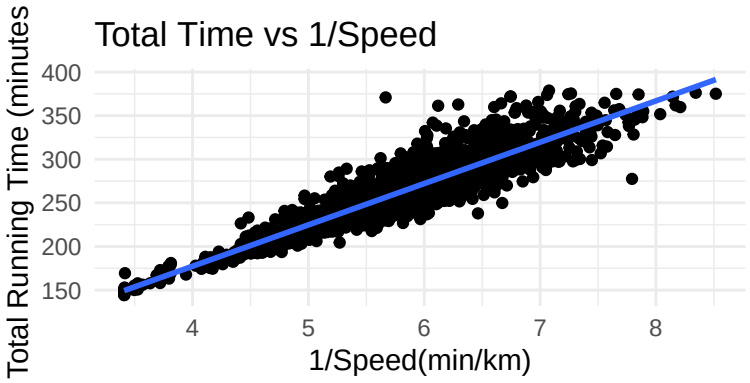


From the Residual vs Fitted plot and Scale-Location Plot, as long they are not to tend to follow linear pattern, we can clearly see that residuals contradict the linear assumption.

Now let's transform the predictor speed using the function f (x) = 1 / x and perform another linear regression of time total onto f (speed):

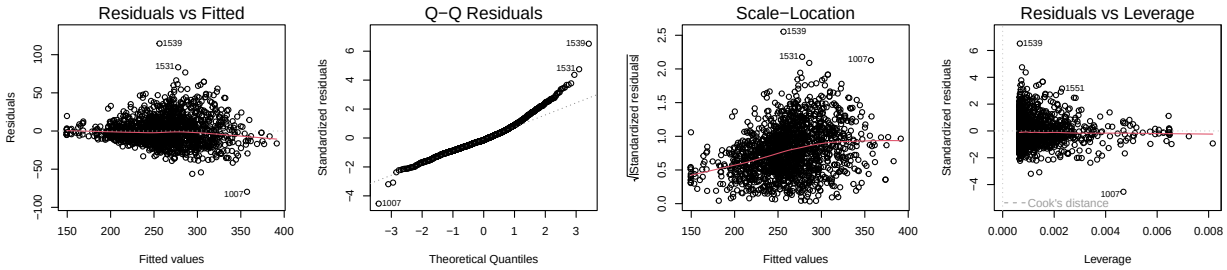
```
marathon$inv_speed <- 1 / marathon$speed
model_inv_speed <- lm(time_total ~ inv_speed, data = marathon)
ggplot(marathon, aes(x = inv_speed, y = time_total)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Total Time vs 1/Speed",
       x = "1/Speed(min/km)", y ="Total Running Time (minutes)") +
  theme_minimal()
```

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



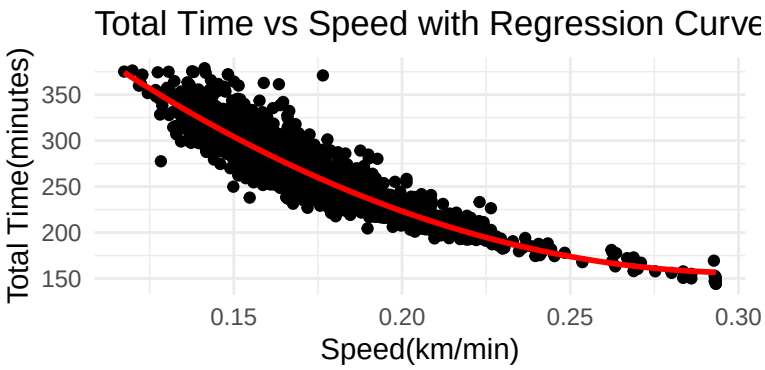
Here we can clearly see linear pattern. Let's discuss diagnostic plots for the new model:

```
par(mfrow = c(1, 4))
plot(model_inv_speed)
```



At the first plot we can see slight curvature, which could indicate some non-linearity, but in general linear model seems to be good for this data.

```
ggplot(marathon, aes(x = speed, y = time_total)) +
  geom_point() +
  geom_smooth(aes(x = speed, y = time_total), method = "lm",
              formula = y ~ poly(x, 2), se = FALSE, color = "red") +
  labs(title = "Total Time vs Speed with Regression Curve",
       x = "Speed(km/min)", y = "Total Time(minutes)") +
  theme_minimal()
```



To improve the model fit it is possible to apply some non-linear transformations on predictors, such as logs, roots or transformation demonstrated above. It could be also useful to get rid of significant outliers.