Loading necessary libraries

```
library(ggplot2)
library(rsample)     # data splitting
library(dplyr)       # data wrangling
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(rpart)       # performing regression trees
library(rpart.plot)  # plotting regression trees
```

Loading the dataset

```
movies_data <- read.csv("Final_Project_FlixGem.csv")
```

Task 1

Data Pre-Processing

Since the company doesn't want results for Series, we have to drop rows containing Series.

```
movies_data_v2<-movies_data[!(movies_data$Series.or.Movie=="Series"),]
```

Moreover, we can drop rows which has missing values (NA) as well. So our polished and cleaned data looks like this:

```
movies_data_v3 <- na.omit(movies_data_v2)
```

Part (a)

Hidden Gem Scores and Languages

Converting the languages in string to character vectors and assigning the first language in the string as the language because after studying the data, the languages are mostly listed by abundance and so the top language is automatically selected as the categorical value for that movie. Here, there might be some information loss, but this seems like a good way to encode the data since there are just too many categories.

```
for (i in 1:length(movies_data_v3$Languages)) {
  movies_data_v3$Languages[i]<-unlist(strsplit(movies_data_v3$Languages[i],split = ", "))

}

summary_table <-movies_data_v3 %>%
   group_by(Languages) %>%
   summarise(n = n())

summary_table <- summary_table %>% arrange(desc(n))
knitr::kable(head(summary_table))
```

Languages	n
English	1891
French	34
Spanish	29
Japanese	27
Mandarin	19
Korean	17

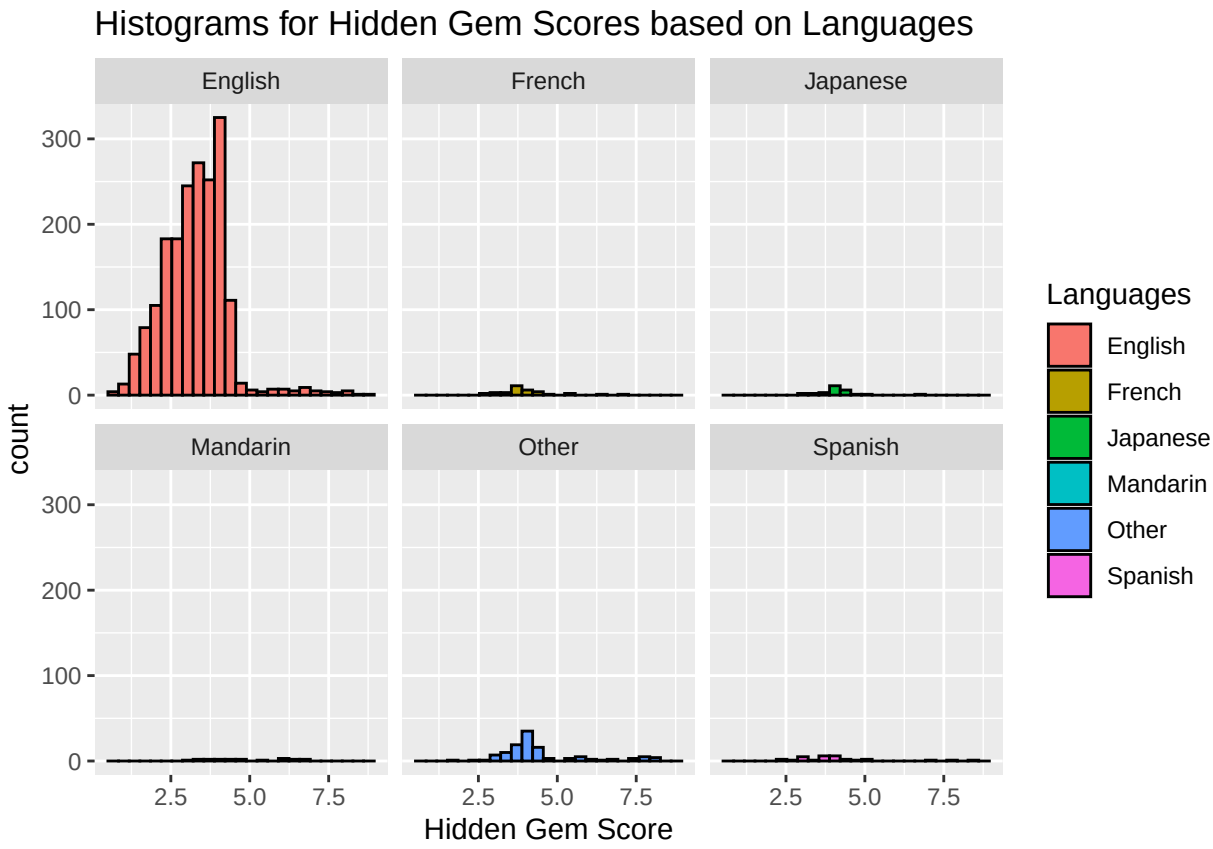
We are deciding to chose the first 5 languages and encode the rest as "Other". The top five languages are English, French, Spanish, Japanese and Mandarin.

```
important_langs <- c("English","French","Spanish","Japanese","Mandarin")

for (i in 1:length(movies_data_v3$Languages)) {
  if(!(movies_data_v3$Languages[i] %in% important_langs)){
    movies_data_v3$Languages[i]<-"Other"
  }
}

(unique(movies_data_v3$Languages))
```

```
## [1] "English"  "French"   "Other"    "Spanish"  "Mandarin" "Japanese"
```

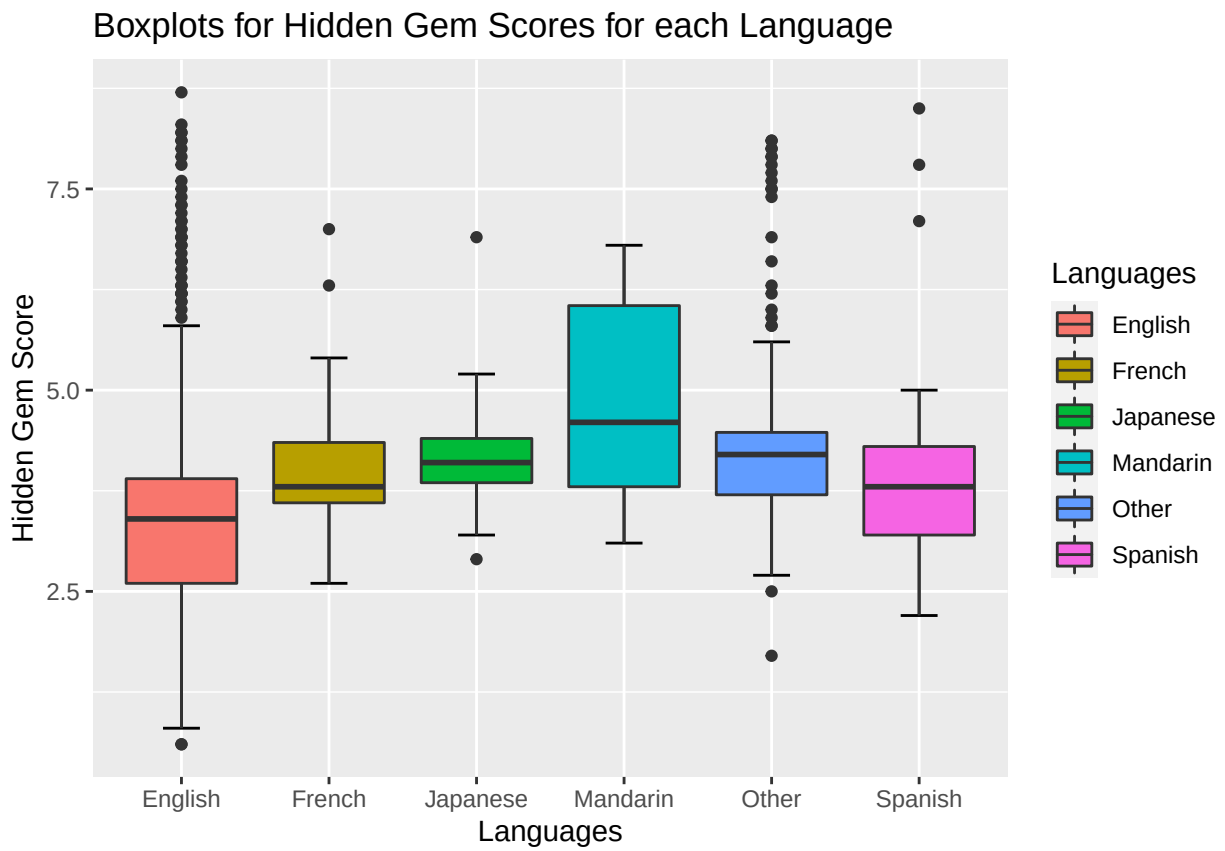
These are the language categories we will be working with. Since the Hidden Gem Score is a continuous variable, we can use histograms to see the distribution of the Hidden Gem Scores for each Language.

```
ggplot(movies_data_v3, aes(x=Hidden.Gem.Score,fill=Languages)) +
     geom_histogram(bins=25,col="black") +
     facet_wrap(~as.factor(Languages))+
  labs(x="Hidden Gem Score",title = "Histograms for Hidden Gem Scores based on Languages")
```



Plotting a boxplot to see the median, interquartile range and outlier values for the Hidden Gem Scores for each Language category.

```
ggplot(movies_data_v3,aes(x=Languages,y=Hidden.Gem.Score,fill=Languages)) +
    stat_boxplot(geom="errorbar",width=0.25) + geom_boxplot() +
     ylab("Hidden Gem Score")+
  labs(title="Boxplots for Hidden Gem Scores for each Language")
```



Explanation

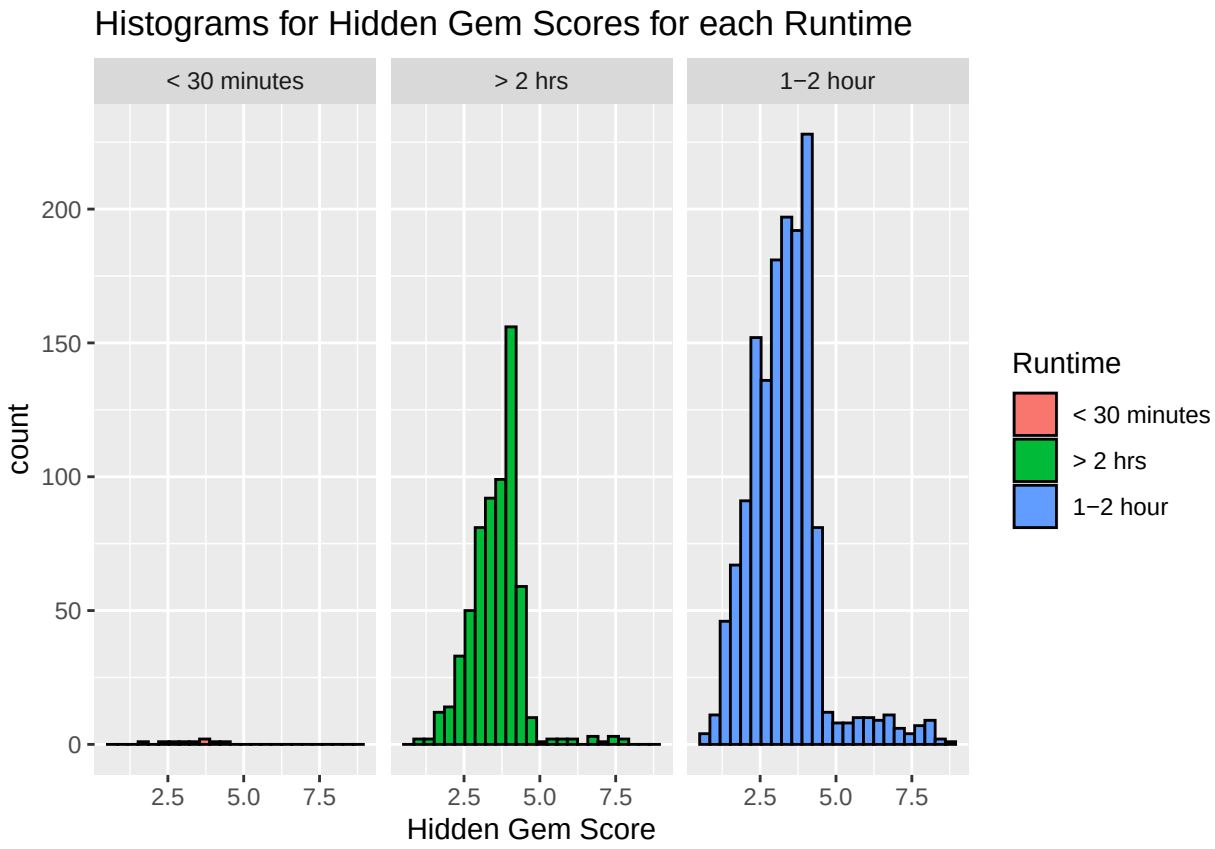
From the histrograms we can see that the majority of the movies are in English, with the shape of the distribution of the English movies being negatively skewed. From the histogram, the shape of the distribution for the rest of the languages are not really clear because of their relatively low count. From the boxplots, we see that the median Hidden Gem score for movies having English as one of the release-languages are the lowest while those that have Mandarin are the highest. The median Hidden Gem score for the rest of the languages are relatively the same (around 3.7). From the boxplot, we also see that English, Spanish and the 'other' categoires have a negative skew while movies in Mandarin, and French have a postive skew. Finally, the ditribution for the movies in Japanese is symmetric. We also note that moveis supporting English has the most number of outliers and those in Mandarin have the lowest.

Hidden Gem Scores and Runtime

Since the Hidden Gem Score is a continuous variable, we can use histograms to see the distribution of the Hidden Gem Scores for each different Runtime Category to see if there is any relationship between them:

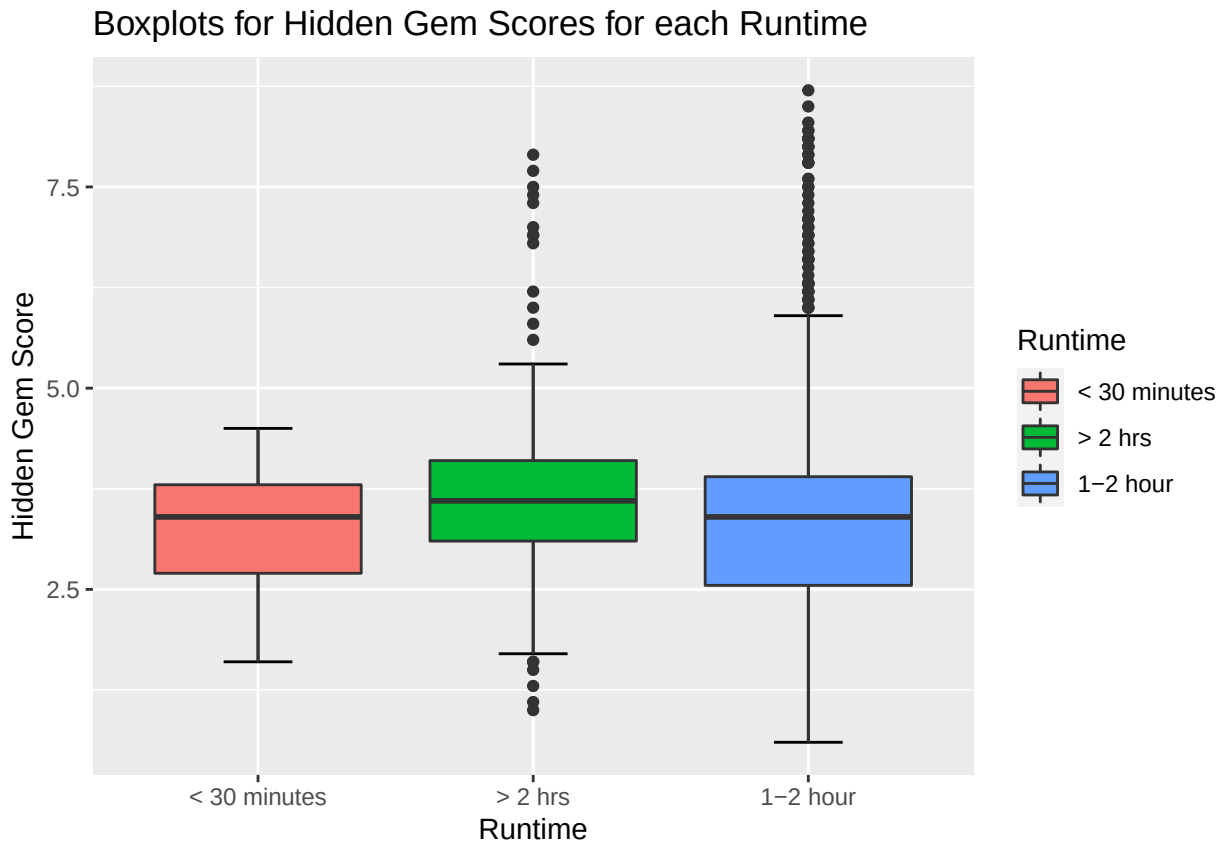
Plotting a histogram for each runtime category to see distribution of Hidden Gem Scores.

```
ggplot(movies_data_v3, aes(x=Hidden.Gem.Score,fill=Runtime)) +
     geom_histogram(bins=25,col="black") +
     facet_wrap(~as.factor(Runtime))+
  labs(x="Hidden Gem Score",title = "Histograms for Hidden Gem Scores for each Runtime")
```



Plotting a boxplot to see the median, interquartile range and outlier values for the Hidden Gem Scores for each different Runtime Category.

```
ggplot(movies_data_v3,aes(x=Runtime,y=Hidden.Gem.Score,fill=Runtime)) +
    stat_boxplot(geom="errorbar",width=0.25) + geom_boxplot() +
     ylab("Hidden Gem Score")+
  labs(title="Boxplots for Hidden Gem Scores for each Runtime")
```



Explanation

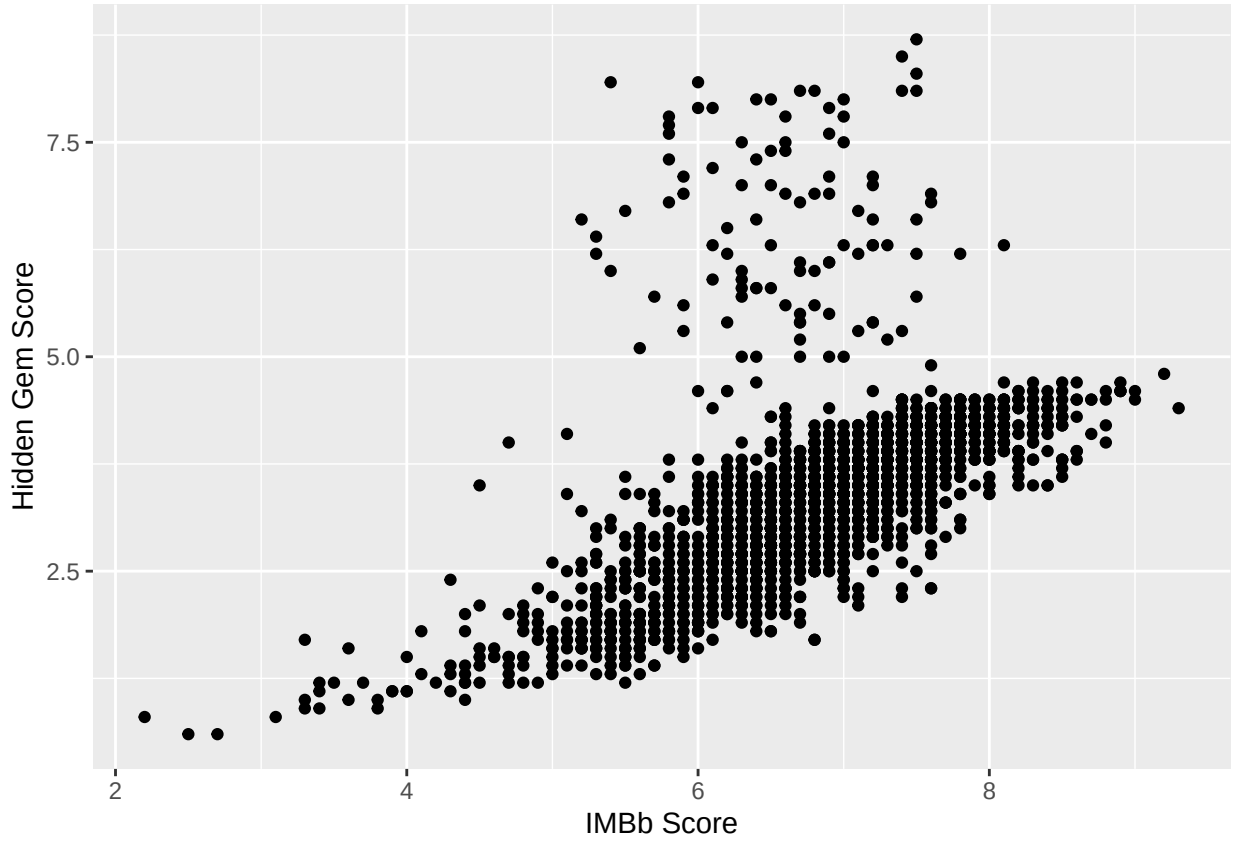
From the boxlpots we see that for all three categories of run time the median and the upper quartile values are relatively the same, although the lower quartile values vary quite a bit. From both the histogram and the boxplots, we see that the less than 30 mins and 1-2 hour run time category all have the same negative skew. From the hsitrogram, we see that the shape of the distribution for the greater than 2 hour category and the 1-2 hour category are relatively the same. Therefore, we can say the greater than 2 hours category of run time also has the same negative skew.

Part (b)

IMDb Scores and Hidden Gem Scores

Analyzing if IMDb Scores are correlated with Hidden Gem Scores. Plotting a graph with IMDb Scores on the x-axis and Hidden Gem Scores on the y-axis.

```
ggplot(movies_data_v3,aes(x=IMDb.Score,y=Hidden.Gem.Score)) + geom_point() +
xlab("IMBb Score") + ylab("Hidden Gem Score")
```

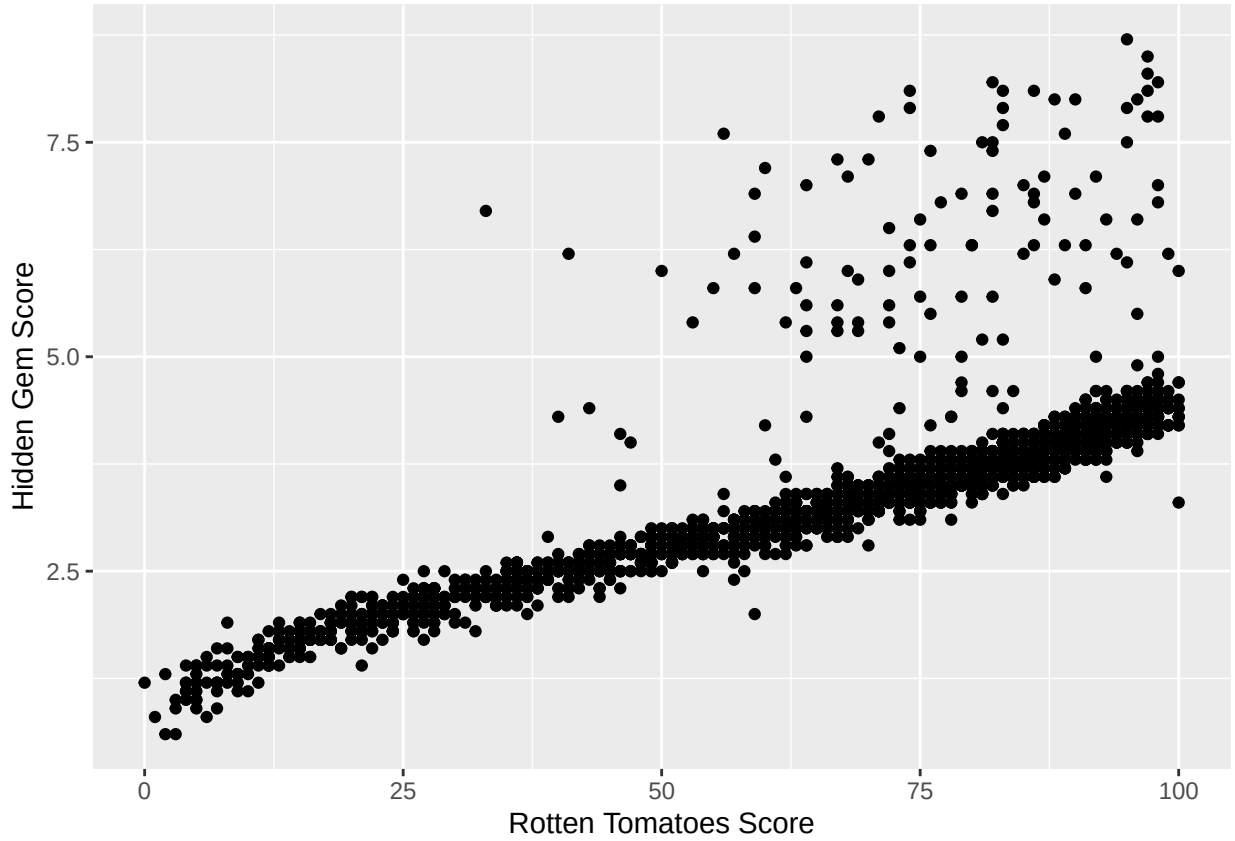


From the graph, we can see that there is some positive correlation between IMDb Scores and Hidden Gem Scores since the graph has a positive gradient except some randomly scattered Hidden Gem Score values(around 5 to 8) when the IMDb Scores vary between 6 and 8.

Rotten Tomatoes Scores and Hidden Gem Scores

Analyzing if Rotten Tomatoes Scores are correlated with Hidden Gem Scores. Plotting a graph with Rotten Tomatoes Scores on the x-axis and Hidden Gem Scores on the y-axis.

```
ggplot(movies_data_v3,aes(x=Rotten.Tomatoes.Score,y=Hidden.Gem.Score)) + geom_point() +
xlab("Rotten Tomatoes Score") + ylab("Hidden Gem Score")
```

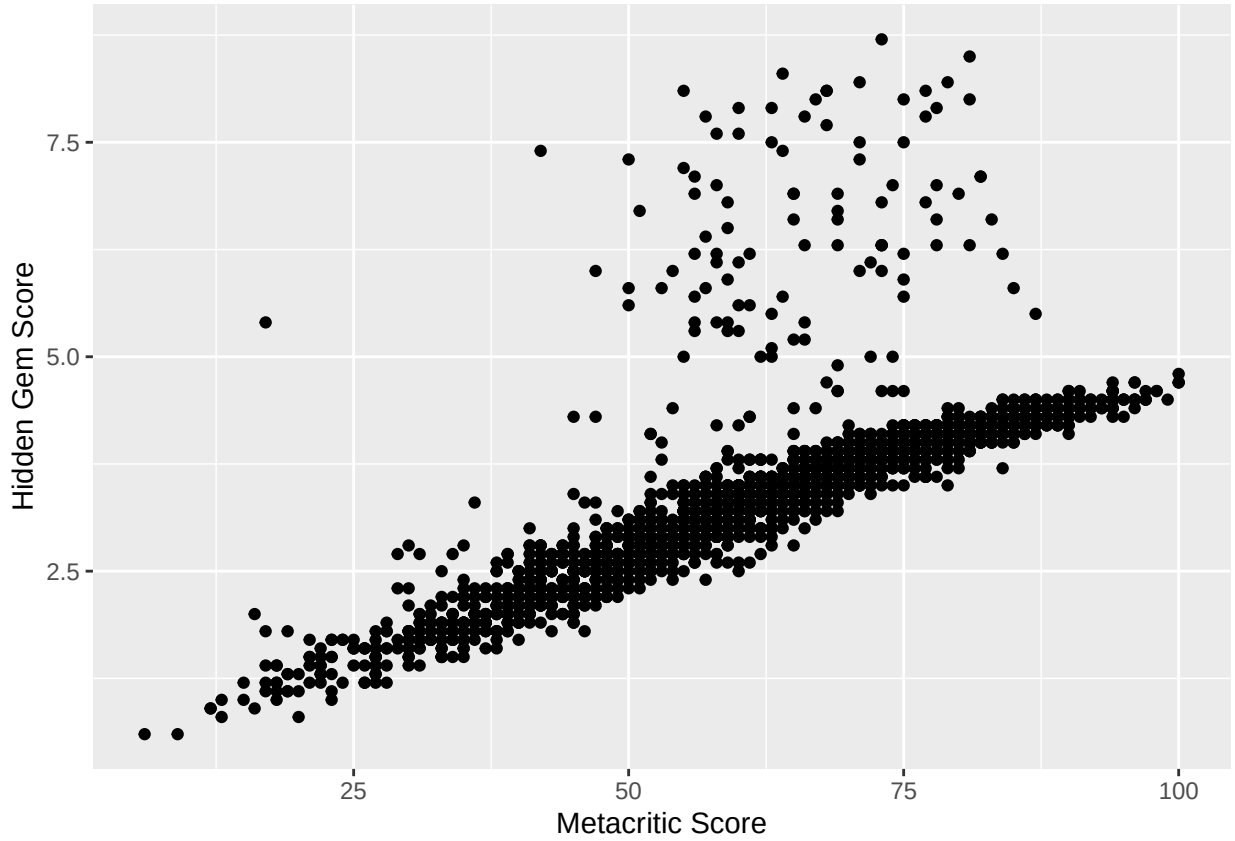


From the graph, we can deduce that there is a strong positive correlation between Rotten Tomatoes Score and Hidden Gem Score since the graph has a positive gradient since it is upward sloping. However, there are a few movies within high rotten tomatoes scores which have much higher hidden gem scores in proportion.

Metacritic Scores and Hidden Gem Scores

Analyzing if Metacritic Scores are correlated with Hidden Gem Scores. Plotting a graph with Metacritic Scores on the x-axis and Hidden Gem Scores on the y-axis.

```
ggplot(movies_data_v3,aes(x=Metacritic.Score,y=Hidden.Gem.Score)) + geom_point() +
xlab("Metacritic Score") + ylab("Hidden Gem Score")
```



Again, it looks like there is a positive correlation due to the upward sloping nature of the scatterplot. This suggests there is strong positive correlation.

Part(c)

From the boxplpots we can see that the median values of the Hidden Gem scores don't change much as the length of the runtime increase. Also, from the histograms for the 1-2 hour category and the >2 hour category, the shape of the distribution remains relatively same. Therefore, we can't really claim that people are becoming more acceptable of longer movies.

Task 2

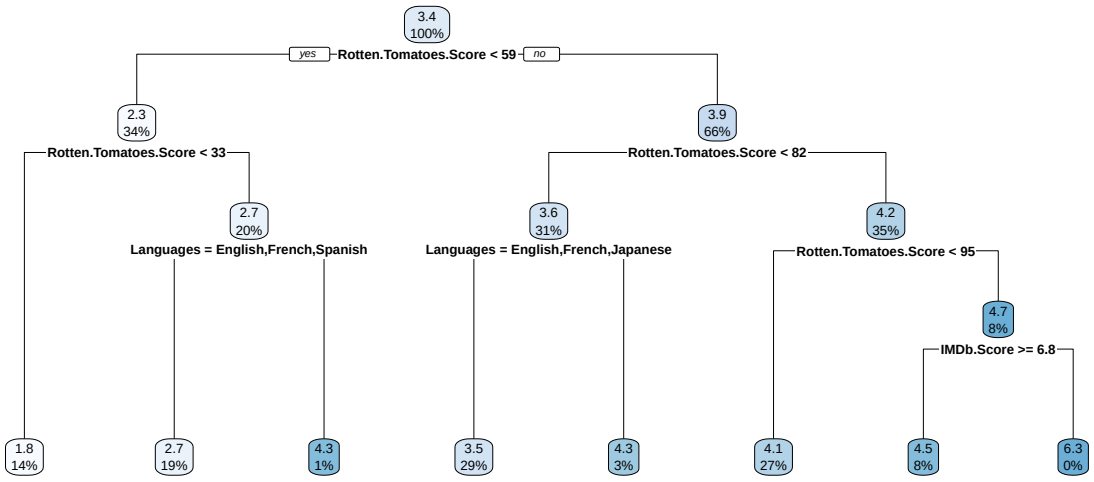
Creating the regression tree with outcome as Hidden Gem Score and predictors as Language, IMDb Score, Runtime, Rotten Tomatoes Score and Metacritic Score

```
regression_tree <- rpart(
  Hidden.Gem.Score ~ Languages + Runtime + IMDb.Score+ Rotten.Tomatoes.Score +
    Metacritic.Score,
  data = movies_data_v3,
  method = "anova"
)

regression_tree
```

```
## n= 2118
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 2118 2548.05300 3.396412
##    2) Rotten.Tomatoes.Score< 58.5 716  314.62640 2.347765
##      4) Rotten.Tomatoes.Score< 32.5 299   40.34562 1.809030 *
##      5) Rotten.Tomatoes.Score>=32.5 417  125.27650 2.734053
##       10) Languages=English,French,Spanish 406   61.90443 2.690640 *
##       11) Languages=Mandarin,Other 11   34.36545 4.336364 *
##    3) Rotten.Tomatoes.Score>=58.5 1402 1043.96800 3.931954
##      6) Rotten.Tomatoes.Score< 81.5 664  407.27110 3.583133
##       12) Languages=English,French,Japanese 609  276.83550 3.515928 *
##       13) Languages=Mandarin,Other,Spanish 55   97.22909 4.327273 *
##      7) Rotten.Tomatoes.Score>=81.5 738  483.21200 4.245799
##       14) Rotten.Tomatoes.Score< 94.5 568  287.35490 4.124648 *
##       15) Rotten.Tomatoes.Score>=94.5 170  159.66490 4.650588
##         30) IMDb.Score>=6.75 160  101.79940 4.548125 *
##         31) IMDb.Score< 6.75 10   29.30900 6.290000 *
```

Plotting the Regression Tree

```
rpart.plot(regression_tree)
```



Summary

Since the root of our tree is the Rotten Tomatoes score we can say that the Rotten Tomatoes Score is the most important in determining the Hidden Gem Score. We could also say that languages are the second most important factor followed by IMDb scores in determining the said score. The larger the number of leaves, the more accurate our predictions would be. We believe that our model has sufficient number of leaves, given the range of Hidden Gem scores in the data set, to predict the score with relative accuracy. We also note that if we had mores leaves in our tree our predictions would be more accurate.

Task 3

Creating empty data frame to store the Director Names and the HG-H Index.

```
director_data<-data.frame( matrix(ncol = 2, nrow = length(unique(movies_data_v3$Director))))
colnames(director_data) <- c("Director","HGHIndex")
director_data$Director <- unique(movies_data_v3$Director)
knitr::kable(head(director_data))
```

Director	HGHIndex
Todd Phillips	NA
George Lucas	NA
David Yates	NA
Francis Veber	NA
Robert Redford	NA
Rob Reiner	NA

Gathering the Hidden Gem Scores for each director

```
find_hgs <- function(director){
  scores <- c()
  for(j in 1:length(movies_data_v3$Director)){
    if(movies_data_v3$Director[j]==director){
      #take hidden gem scores
      scores <- append(scores,movies_data_v3$Hidden.Gem.Score[j])
    }
  }
  return(scores)
}
#function for finding the h_index
h_index <- function(gem_scores) {
    sorted_gemscores <- sort(gem_scores, decreasing = T)
    bools<- (sum(sorted_gemscores >= seq_along(sorted_gemscores)))
    return(bools)
}

for (i in 1:length(director_data$Director)) {
  director_data$HGHIndex[i]<-h_index(find_hgs(director_data$Director[i]))
}

knitr::kable(head(director_data))
```

Director	HGHIndex
Todd Phillips	3
George Lucas	2
David Yates	3
Francis Veber	1
Robert Redford	3
Rob Reiner	2

Top 10 Directors

```
director_data<-director_data %>% arrange(desc(HGHIndex))
knitr::kable(director_data[1:10,])
```

Director	HGHIndex
Steven Spielberg	4
Quentin Tarantino	4
Ang Lee	4
David Fincher	4
Bong Joon Ho	4
Woody Allen	4
David Mackenzie	4
Hayao Miyazaki	4
Peter Jackson	4
Paul Thomas Anderson	4

Note that there are more directors with the same Hidden Gem Score H-Index of 4 but they are not displayed here.