Faculté des Sciences et Techniques
Département d'Informatique
Master SIDI- DM– 2023/2024

MINI PROJET: MOVIELENS

MINI PROJET : APPLICATION SOUS R STUDIO POUR UN SYSTEME DE RECOMMANDATION

Réalisé par :

Ismail CHARIHI

Encadré par :

M. Mohamed SABIRI

Table des matières

I. Introduction : 3
II. Problematique 3
III. OUTILS UTILISE 3
IV. Démarches 4
1. Ingestion de données. 4
2. Analyse exploratoire 5
3. Stratégies d'analyse des données : 10
4. Création de model 13
5. Validation 13
6. Test 14
7. Evaluation 15
8. Résultats 15
V. Conclusion 15

I. INTRODUCTION :

Les systèmes de recommandations exploitent les notes (Rating) que les utilisateurs ont attribuées aux films pour formuler des recommandations spécifiques.

Les articles pour lesquels une note élevée est prévue pour un utilisateur donné sont ensuite recommandées à cet utilisateur.

Le but du ce projet consiste à développer un algorithme d'apprentissage automatique utilisant les entrées d'un sous-ensemble pour prédire les classements des films dans l'ensemble de validation. On utilisant le Langage R par RStudio .

Plusieurs algorithmes d'apprentissage automatique ont été utilisés et le modèle final est celui qui donne une précision maximale possible dans la prédiction. Ce rapport contient la définition du problème, l'ingestion de données, analyse exploratoire, modélisation et analyse des données, résultats et remarques finales.

Le projet utilise Pénalisé approche des moindres carrés motivée par les défis Netfix. Le film, l'utilisateur, l'année, le genre sont quelques-uns des fonctionnalités qui ont un plus grand effet sur les erreurs. Nous allons essayer de réduire ces effets en utilisant la méthode proposée pour améliorer la précision.

II. PROBLEMATIQUE

Un système de recommandation de films prédit la classification du film par un utilisateur sur la base de la classification antérieure des films par l'utilisateur.

Il existe différents types de préjugés présents dans les critiques de films. Cela peut être déférent d'ordre social, psychologique, des variations démographiques qui changent le goût de chaque utilisateur pour un film particulier donné. Cependant le problème peut être résolu en exprimant des biais majeurs dans les équations mathématiques.

III. OUTILS UTILISE

Dans le cadre de ce projet d'analyse des avis des clients, nous avons utilisé plusieurs outils de développement pour faciliter le traitement des données et l'analyse des sentiments. Voici un aperçu des principaux outils que nous avons utilisés :

🗁 Langage R :

Le langage **R** est un langage de programmation et un environnement logiciel largement utilisé dans le domaine de la statistique et de l'analyse de données. Il a été développé initialement par Ross Ihaka et Robert Gentleman à l'Université d'Auckland, en Nouvelle-Zélande, dans les années 1990.

R offre une grande variété de fonctionnalités pour la manipulation, la visualisation et l'analyse des données. Il est particulièrement apprécié pour son large éventail de packages, qui fournissent des outils spécialisés pour des tâches spécifiques telles que l'apprentissage automatique, la bioinformatique, l'économétrie, la visualisation de données, etc. Ces packages peuvent être installés et chargés dans l'environnement R pour étendre ses fonctionnalités de base.

L'environnement R est basé sur une ligne de commande, ce qui signifie que les instructions sont généralement saisies et exécutées une par une. Cependant, il existe également des interfaces graphiques conviviales pour R, telles que RStudio, qui facilitent le développement et l'exécution de scripts R.

R dispose d'une syntaxe expressive et concise, qui permet aux utilisateurs d'effectuer des opérations complexes sur les données avec peu de lignes de code. Il prend en charge les opérations mathématiques de base, les opérations sur les vecteurs et les matrices, les structures de contrôle (boucles, conditions), les fonctions, les graphiques, etc.

🗁 R-Studio :

R-Studio est un environnement de développement intégré (IDE) spécialement conçu pour le langage R. Nous avons utilisé R-Studio pour écrire, exécuter et déboguer notre code R. Cet outil nous a offert une interface conviviale et des fonctionnalités avancées pour améliorer notre productivité en matière de développement.

IV. DEMARCHES

1. Ingestion de données.

Le morceau de code ci-dessous génère une partition de l'ensemble de données pour l'entraînement et le test de nos données. Il supprime également les fichiers inutiles du répertoire de travail.

a) Installation de packages nécessaires

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")

library(tidyverse)
library(caret)
library(data.table)
```

b) Téléchargement de Dataset (films et ratings)

```
dl <- tempfile()
download.file("http://files.grouplens.org/datasets/movielens/ml-10m.zip", dl)

ratings <- fread(text = gsub("::", "\t", readLines(unzip(dl, "ml-10M100K/ratings.dat"))),
                 col.names = c("userId", "movieId", "rating", "timestamp"))

movies <- str_split_fixed(readLines(unzip(dl, "ml-10M100K/movies.dat")), "\\::", 3)
colnames(movies) <- c("movieId", "title", "genres")


# if using R 4.0 or later:
movies <- as.data.frame(movies) %>% mutate(movieId = as.numeric(movieId),
                                           title = as.character(title),
                                           genres = as.character(genres))
```

c) Les données de validation

```r
# L'ensemble de validation représentera 10 % des données de MovieLens
set.seed(1, sample.kind = "Rounding")  # si vous utilisez R 3.5 ou une version antérieure, utilisez `set.seed(1)`
test_index <- createDataPartition(y = movielens$rating, times = 1, p = 0.1, list = FALSE)
edx <- movielens[-test_index,]
temp <- movielens[test_index,]

# Assurez-vous que userId et movieId dans l'ensemble de validation sont également dans l'ensemble edx
validation <- temp %>%
  semi_join(edx, by = "movieId") %>%
  semi_join(edx, by = "userId")

# Ajouter les lignes supprimées de l'ensemble de validation à l'ensemble edx
removed <- anti_join(temp, validation)
edx <- rbind(edx, removed)

rm(dl, ratings, movies, test_index, temp, movielens, removed)
```

d) Prétraitement des données

Modification des colonnes vers un format convenable pour l'analyse

```r
# Prétraitement des données
# Modifier l'année en tant que colonne dans les deux ensembles de données
edx <- edx %>% mutate(year = as.numeric(str_sub(title, -5, -2)))
validation <- validation %>% mutate(year = as.numeric(str_sub(title, -5, -2)))
```

e) Calcule de RMSE

La valeur utilisée pour évaluer la performance de l'algorithme est l'erreur quadratique moyenne (RMSE, pour Root Mean Square Error). Le RMSE est l'une des mesures les plus utilisées pour évaluer les différences entre les valeurs prédites par un modèle et les valeurs observées.

```r
# Fonction de perte RMSE (Root Mean Square Error)
RMSE <- function(true_ratings, predicted_ratings) {
  sqrt(mean((true_ratings - predicted_ratings)^2, na.rm = TRUE))
}
```

Le RMSE est une mesure de précision, plus le RMSE est bas, meilleur est le modèle que plus il est élevé. L'effet de chaque erreur sur le RMSE est proportionnel à la taille de l'erreur au carré; ainsi, les erreurs plus importantes auront un impact plus important sur le RMSE. Le RMSE est sensible aux valeurs aberrantes. Le critère d'évaluation pour cet algorithme est un RMSE devant être inférieur à 0,8775.

2. Analyse exploratoire.

Il y a six variables dans le sous-ensemble : "userId", "movieID", "rating", "timestamp", "title", "genres". Chaque ligne représente une seule évaluation d'un utilisateur pour un seul film.

Pour visualiser les premiers éléments

```
# Statistiques sommaires de l'ensemble edx
summary(edx)
```

```
> head(edx)
   userId movieId rating timestamp                         title                        genres year
1:      1     122      5 838985046              Boomerang (1992)                Comedy|Romance 1992
2:      1     185      5 838983525               Net, The (1995)         Action|Crime|Thriller 1995
3:      1     292      5 838983421               Outbreak (1995)  Action|Drama|Sci-Fi|Thriller 1995
4:      1     316      5 838983392               Stargate (1994)       Action|Adventure|Sci-Fi 1994
5:      1     329      5 838983392 Star Trek: Generations (1994) Action|Adventure|Drama|Sci-Fi 1994
6:      1     355      5 838984474       Flintstones, The (1994)       Children|Comedy|Fantasy 1994
```

Pour assurer que le Data n'a pas des données manquants voir un sommaire globale

```
# Statistiques sommaires de l'ensemble edx
summary(edx)
```

```
> summary(edx)
     userId          movieId          rating        timestamp           title          
 Min.   :    1   Min.   :    1   Min.   :0.500   Min.   :7.897e+08   Length:9000055    
 1st Qu.:18124   1st Qu.:  648   1st Qu.:3.000   1st Qu.:9.468e+08   Class :character  
 Median :35738   Median : 1834   Median :4.000   Median :1.035e+09   Mode  :character  
 Mean   :35870   Mean   : 4122   Mean   :3.512   Mean   :1.033e+09                     
 3rd Qu.:53607   3rd Qu.: 3626   3rd Qu.:4.000   3rd Qu.:1.127e+09                     
 Max.   :71567   Max.   :65133   Max.   :5.000   Max.   :1.231e+09                     
    genres                year     
 Length:9000055     Min.   :1915  
 Class :character   1st Qu.:1987  
 Mode  :character   Median :1994  
                    Mean   :1990  
                    3rd Qu.:1998  
                    Max.   :2008  
```

Le nombre total de films uniques et d'utilisateurs dans le sous-ensemble edx est fourni dans le morceau de code ci-dessous.

```
# Nombre de films et d'utilisateurs uniques dans l'ensemble edx
edx %>% summarize(n_users = n_distinct(userId), n_movies = n_distinct(movieId))
```

```
> # Number of unique movies and users in the edx dataset
> edx %>% summarize(n_users = n_distinct(userId), n_movies = n_distinct(movieId))
  n_users n_movies
1   69878    10677
```

Le nombre des films par Genre dans l'ensemble data est

```r
# Genres des films dans l'ensemble edx
genres = c("Drama", "Comedy", "Thriller", "Romance")
sapply(genres, function(g) {
  sum(str_detect(edx$genres, g))
})
```

```
   Drama   Comedy Thriller  Romance
3910127  3540930  2325899  1712100
```

Une statistique sommaire des évaluations dans le sous-ensemble edx. La note 4 est la plus courante, suivie de 3, et la note 0.5 est la moins fréquente.

```r
# Statistiques sommaires des évaluations dans l'ensemble edx
summary(edx$rating)
```

```
> summary(edx$rating)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.500   3.000   4.000   3.512   4.000   5.000
```

Les films avec le plus grand nombre d'évaluation est :

```r
# Film avec le plus grand nombre d'évaluations
edx %>% group_by(title) %>% summarise(number = n()) %>% arrange(desc(number))
```

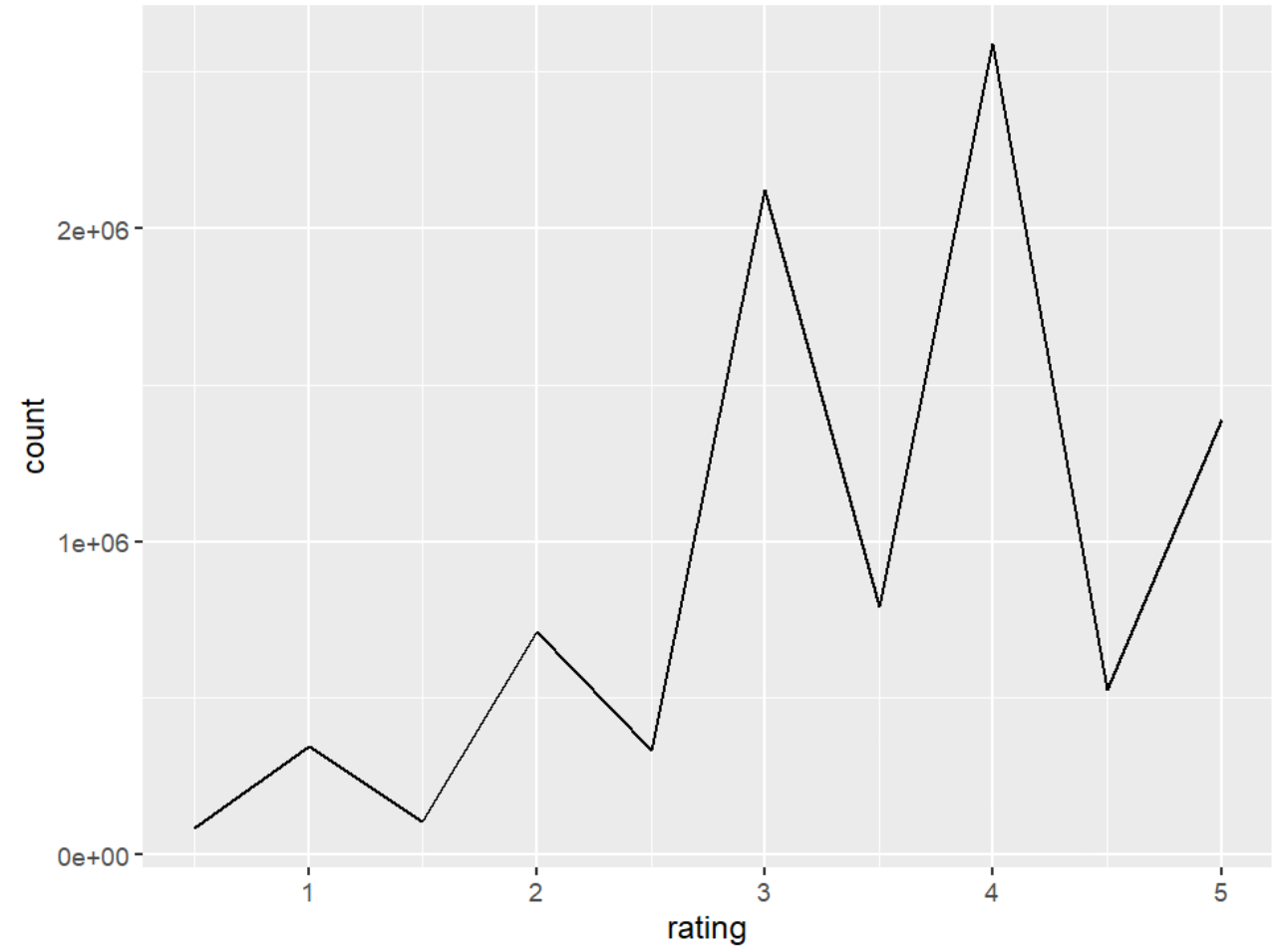
```
> edx %>% group_by(title)%>%summarise(number = n())%>%arrange(desc(number))
# A tibble: 10,676 × 2
   title                                                     number
   <chr>                                                      <int>
 1 Pulp Fiction (1994)                                        31362
 2 Forrest Gump (1994)                                        31079
 3 Silence of the Lambs, The (1991)                           30382
 4 Jurassic Park (1993)                                       29360
 5 Shawshank Redemption, The (1994)                           28015
 6 Braveheart (1995)                                          26212
 7 Fugitive, The (1993)                                       25998
 8 Terminator 2: Judgment Day (1991)                          25984
 9 Star Wars: Episode IV - A New Hope (a.k.a. Star Wars) (1977) 25672
10 Apollo 13 (1995)                                           24284
# ℹ 10,666 more rows
# ℹ Use `print(n = ...)` to see more rows
```

```r
# Les cinq évaluations les plus fréquemment données, de la plus à la moins fréquente
head(sort(-table(edx$rating)), 5)
```

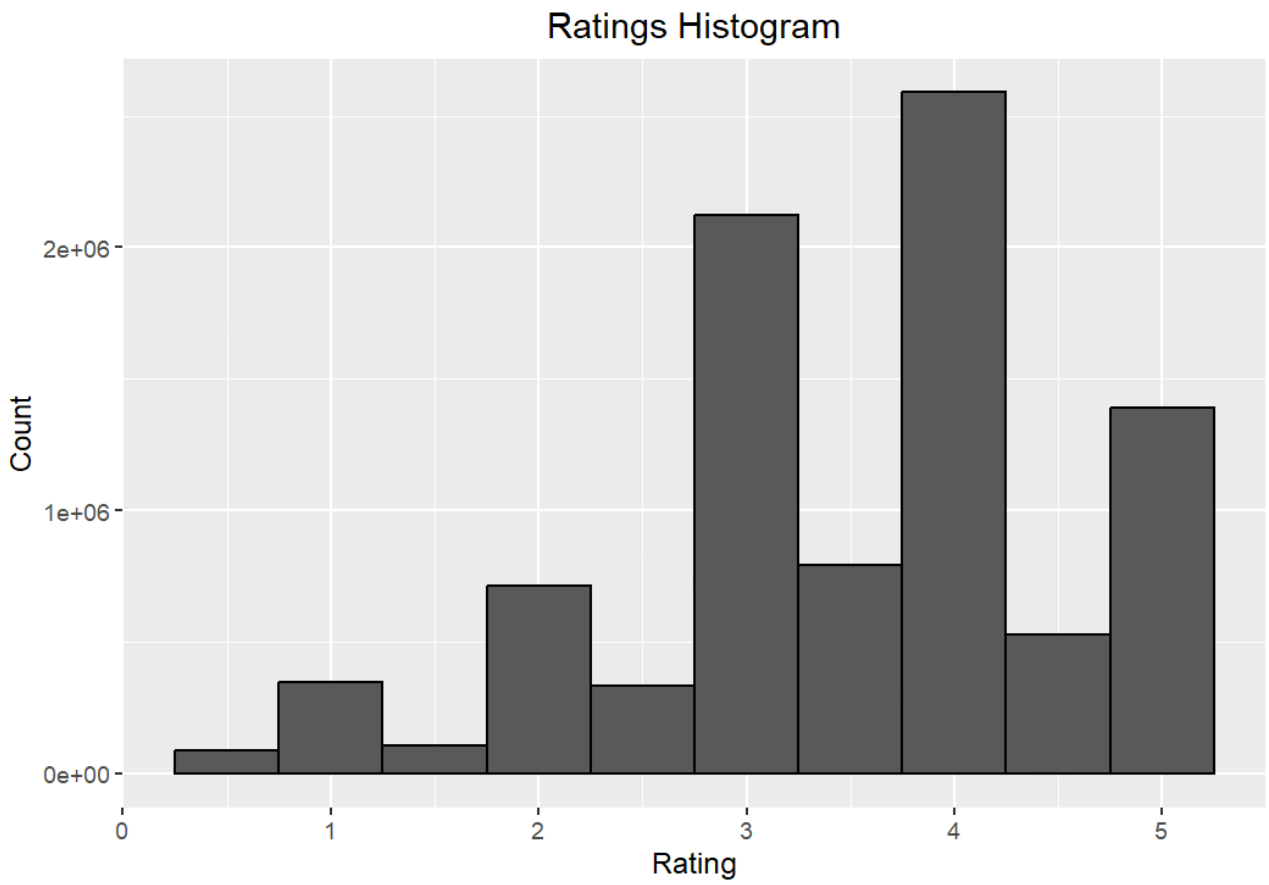
```
> head(sort(-table(edx$rating)),5)
      4       3       5     3.5       2
-2588430 -2121240 -1390114  -791624  -711422
```

Graphique des évaluations

```r
# Graphique des évaluations
table(edx$rating)
edx %>%
  group_by(rating) %>%
  summarize(count = n()) %>%
  ggplot(aes(x = rating, y = count)) +
  geom_line()
```



Histogramme des évaluations




```
## Tableau des 20 films évalués une seule fois
# Ce sont des estimations bruyantes qui peuvent augmenter notre RMSE
edx %>%
  group_by(movieId) %>%
  summarize(count = n()) %>%
  filter(count == 1) %>%
  left_join(edx, by = "movieId") %>%
  group_by(title) %>%
  summarize(rating = rating, n_rating = count) %>%
  slice(1:20) %>%
  knitr::kable()
```

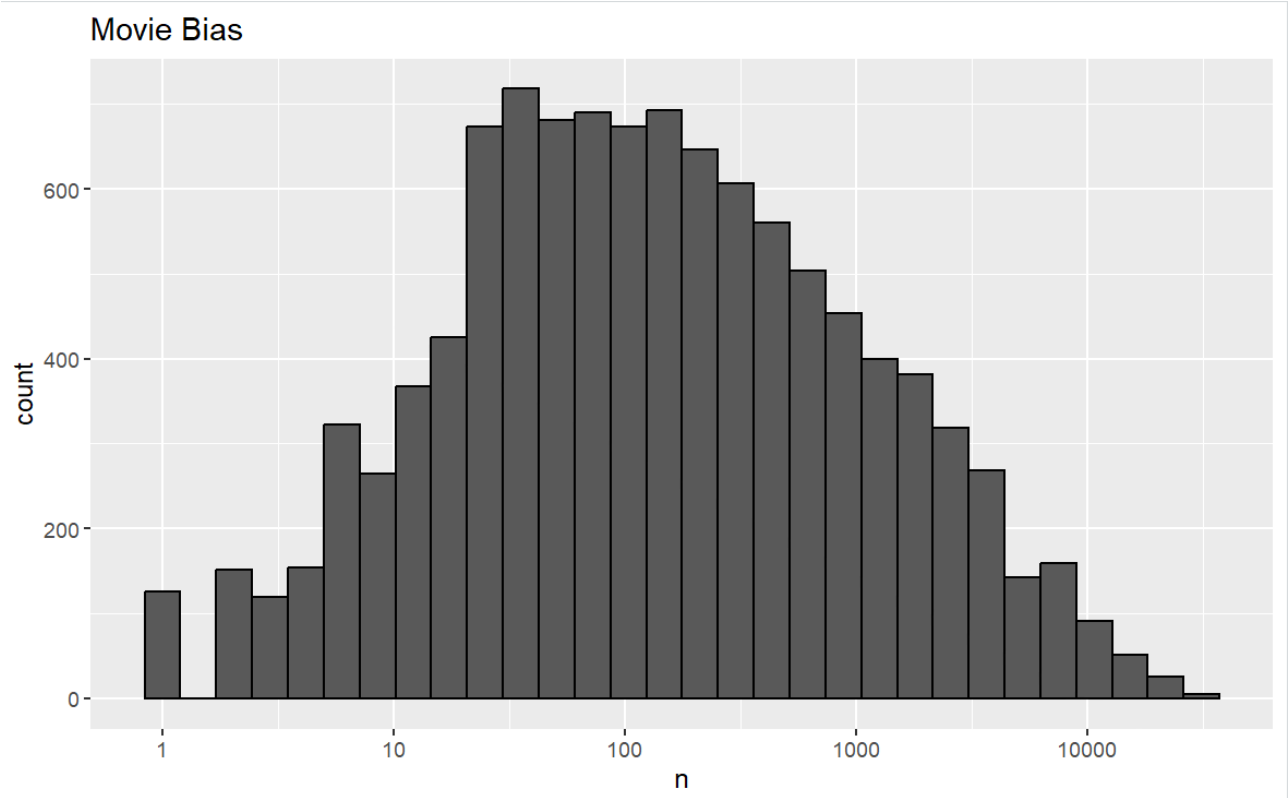
title	rating	n_rating
1, 2, 3, Sun (Un, deux, trois, soleil) (1993)	2.0	1
100 Feet (2008)	2.0	1
4 (2005)	2.5	1
Accused (Anklaget) (2005)	0.5	1
Ace of Hearts (2008)	2.0	1
Ace of Hearts, The (1921)	3.5	1
Adios, Sabata (Indio Black, sai che ti dico: Sei un gran figlio di...) (1971)	1.5	1
Africa addio (1966)	3.0	1
Aleksandra (2007)	3.0	1
Bad Blood (Mauvais sang) (1986)	4.5	1
Battle of Russia, The (Why We Fight, 5) (1943)	3.5	1
Bellissima (1951)	4.0	1
Big Fella (1937)	3.0	1
Black Tights (1-2-3-4 ou Les Collants noirs) (1960)	3.0	1
Blind Shaft (Mang jing) (2003)	2.5	1
Blue Light, The (Das Blaue Licht) (1932)	5.0	1
Borderline (1950)	3.0	1
Brothers of the Head (2005)	2.5	1
Chapayev (1934)	1.5	1
Cold Sweat (De la part des copains) (1970)	2.5	1

>

3. Stratégies d'analyse des données :

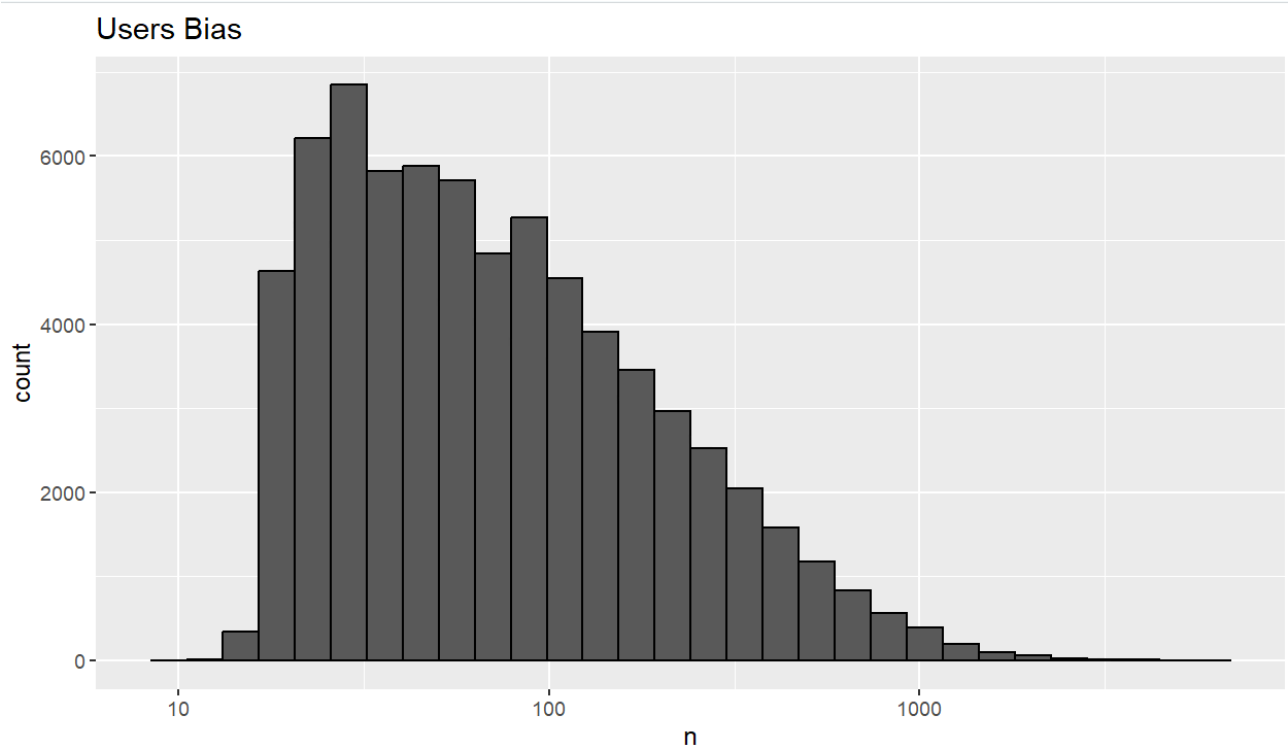
Certains films sont évalués plus fréquemment que d'autres (par exemple, les blockbusters reçoivent des évaluations plus élevées). Cela s'appelle le biais des films. La distribution de l'effet de biais des films (b_i) est présentée ci-dessous

```
# Distribution des biais des films, car la plupart des films à succès sont bien notés - Effet film
edx %>%
  count(movieId) %>%
  ggplot(aes(n)) +
  geom_histogram(bins = 30, color = "black") +
  scale_x_log10() +
  ggtitle("Biais des films")
```



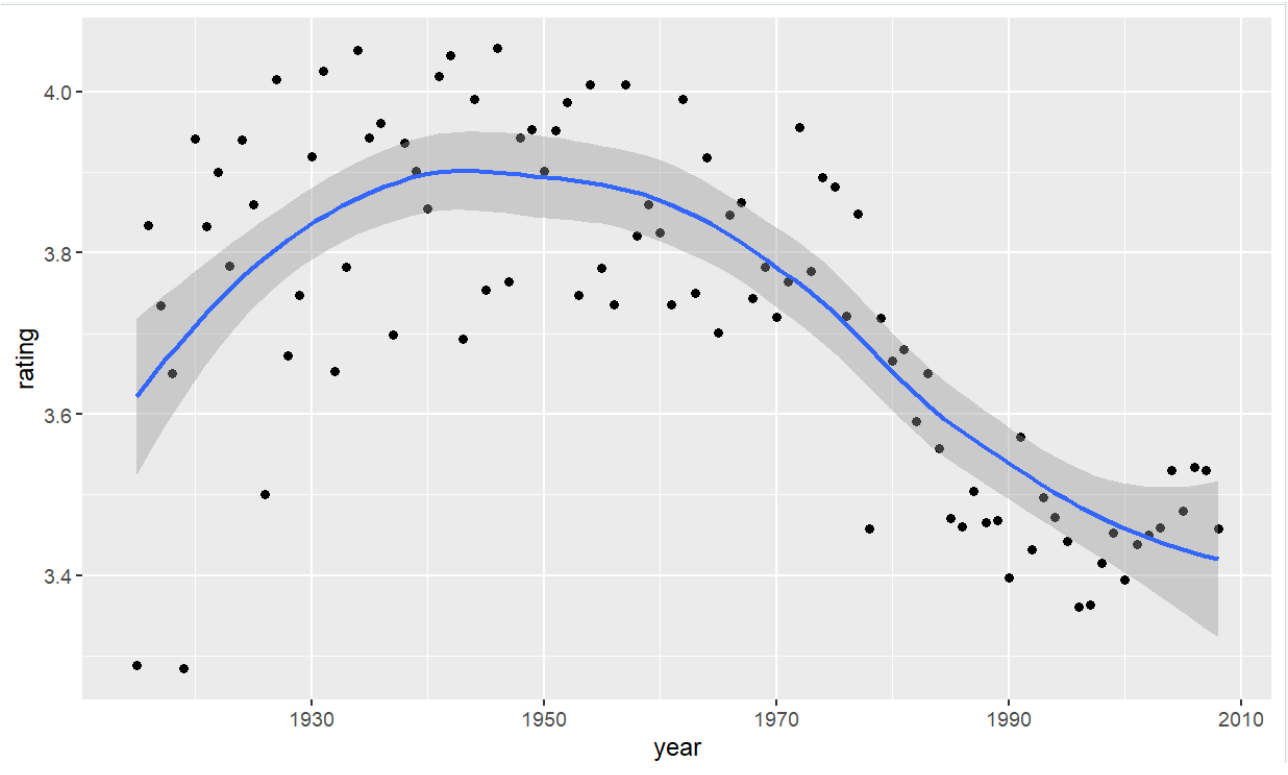
Certains utilisateurs émettent des avis positifs, tandis que d'autres ont des avis négatifs en raison de leurs préférences personnelles, indépendamment du film. La distribution de l'effet de biais des utilisateurs (b_u) est présentée ci-dessous.

```
# Distribution des évaluations de chaque utilisateur pour les films - Effet utilisateur
edx %>% count(userId) %>%
  ggplot(aes(n)) +
  geom_histogram(bins = 30, color = "black") +
  scale_x_log10() +
  ggtitle("Biais des utilisateurs")
```



L'état d'esprit des utilisateurs évolue également avec le temps. Cela peut également affecter la note moyenne des films au fil des ans. Le graphique de l'effet de biais annuel (b_y) est présenté ci-dessous. La tendance générale montre que les utilisateurs modernes notent relativement les films plus bas.

```
# Estimation de la tendance des évaluations par rapport à l'année de sortie - Effet année
edx %>% group_by(year) %>%
  summarize(rating = mean(rating)) %>%
  ggplot(aes(year, rating)) +
  geom_point() +
  geom_smooth()
```



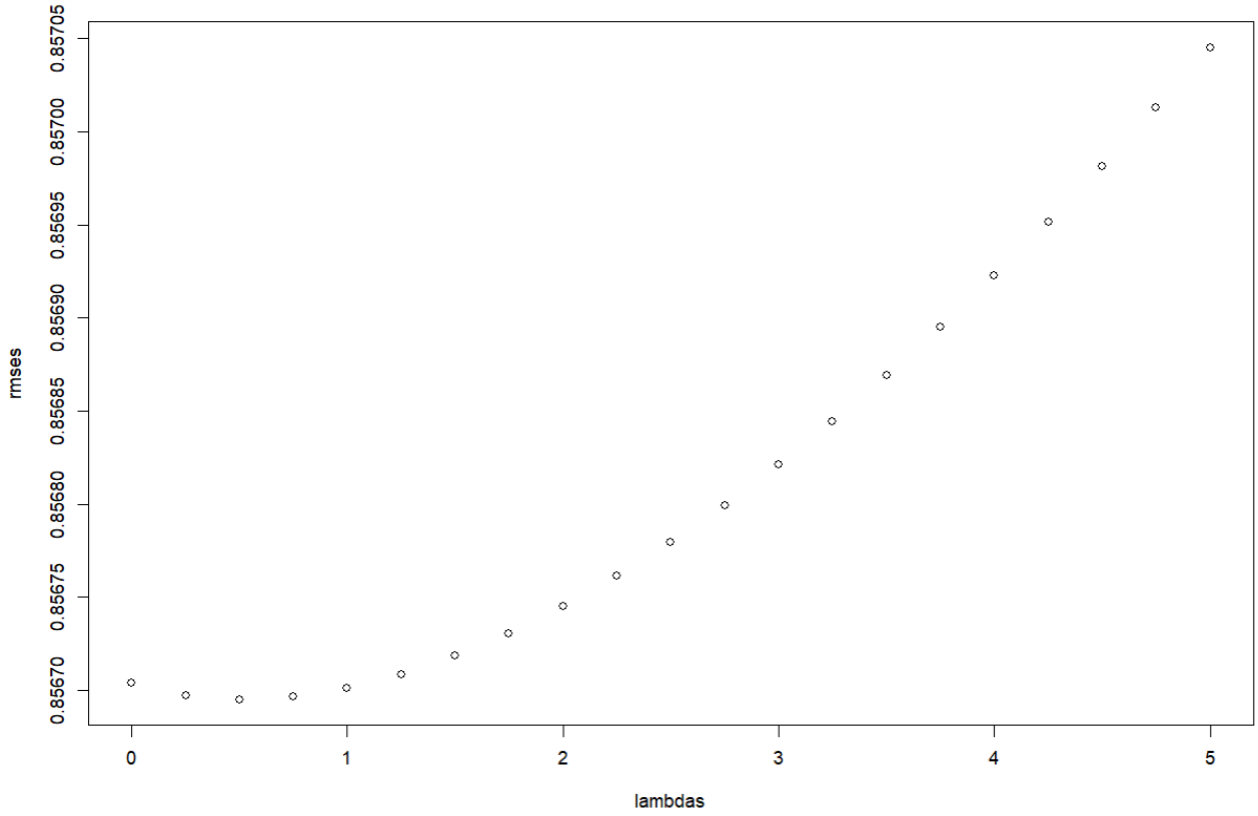
À partir de la distribution des évaluations que nous avons vue dans le module précédent, nous pouvons observer que certains films ne sont évalués qu'une seule fois. Cela sera important pour notre modèle car un nombre très faible d'évaluations peut entraîner des estimations peu fiables pour nos prédictions. Dans le tableau ci-dessous, les 20 films évalués une seule fois semblent obscurs, et la prédiction de futures évaluations pour eux sera difficile.

4. Création de model

```r
# lambdas est une séquence de valeurs de pénalité allant de 0 à 5 avec un pas de 0.25
lambdas <- seq(0, 5, 0.25)

# rmses stocke les résultats de l'erreur quadratique moyenne (RMSE) pour différentes valeurs de lambda
rmses <- sapply(lambdas, function(l) {_

  # Calculer la moyenne des évaluations à partir de l'ensemble d'entraînement edx
  mu <- mean(edx$rating)

  # Ajuster la moyenne par l'effet du film et pénaliser les faibles nombres d'évaluations
  b_i <- edx %>%
    group_by(movieId) %>%
    summarize(b_i = sum(rating - mu)/(n() + l))

  # Ajuster la moyenne par l'effet de l'utilisateur et du film et pénaliser les faibles nombres d'évaluations
  b_u <- edx %>%
    left_join(b_i, by = "movieId") %>%
    group_by(userId) %>%
    summarize(b_u = sum(rating - b_i - mu)/(n() + l))

  # Ajuster la moyenne par l'effet de l'utilisateur, du film et de l'année et pénaliser les faibles nombres d'évaluations
  b_y <- edx %>%
    left_join(b_i, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    group_by(year) %>%
    summarize(b_y = sum(rating - mu - b_i - b_u)/(n() + l), n_y = n())

  # Prédire les évaluations dans l'ensemble d'entraînement pour trouver la valeur optimale de la pénalité 'lambda'
  predicted_ratings <- edx %>%
    left_join(b_i, by = "movieId") %>%
    left_join(b_u, by = "userId") %>%
    mutate(pred = mu + b_i + b_u) %>%
    .$pred

  # Retourner l'erreur quadratique moyenne entre les évaluations réelles et prédites
  return(RMSE(edx$rating, predicted_ratings))
})
```

```r
# Tracer la relation entre lambdas et rmses
plot(lambdas, rmses)
```

5. Validation




```r
# Sélectionner la valeur optimale de lambda qui minimise l'erreur
lambda <- lambdas[which.min(rmses)]
```

```
> lambda
[1] 0.5
```

En appliquant la valeur de lambda à l'ensemble de validation, nous pouvons générer les prédictions pour la validation.

```r
# Appliquer lambda sur l'ensemble de validation
mu <- mean(edx$rating)
movie_effect_reg <- edx %>%
  group_by(movieId) %>%
  summarize(b_i = sum(rating - mu)/(n() + lambda), n_i = n())
user_effect_reg <- edx %>%
  left_join(movie_effect_reg, by = "movieId") %>%
  group_by(userId) %>%
  summarize(b_u = sum(rating - mu - b_i)/(n() + lambda), n_u = n())
year_reg_avgs <- edx %>%
  left_join(movie_effect_reg, by = "movieId") %>%
  left_join(user_effect_reg, by = "userId") %>%
  group_by(year) %>%
  summarize(b_y = sum(rating - mu - b_i - b_u)/(n() + lambda), n_y = n())
```

6. Test

```r
# Prédire les évaluations sur l'ensemble de validation avec le modèle régularisé
predicted_ratings <- validation %>%
  left_join(movie_effect_reg, by = "movieId") %>%
  left_join(user_effect_reg, by = "userId") %>%
  left_join(year_reg_avgs, by = "year") %>%
  mutate(pred = mu + b_i + b_u + b_y) %>%
  .$pred
```

7. Evaluation

```r
# Calculer le RMSE du modèle sur l'ensemble de validation
model_rmse <- RMSE(validation$rating, predicted_ratings)

# Afficher les résultats
rmse_results <- data_frame(method = "Modèle avec Effet Régularisé sur Films, Utilisateurs et Années",
                           RMSE = model_rmse)
rmse_results %>% knitr::kable()
```

8. Résultats

La valeur de l'erreur quadratique moyenne (RMSE) du modèle d'effet régularisé sur les films, les utilisateurs et les années est donnée ci-dessous.

method	RMSE
Reg Movie, User, Year Effect Model	0.8648841

V. CONCLUSION

Une analyse approfondie des données a révélé que certains points de données dans les caractéristiques ont un impact important sur les erreurs. Ainsi, un modèle de régularisation a été utilisé pour pénaliser de tels points de données. Le RMSE final est de 0,8648, inférieur au critère d'évaluation initial de 0,8775 fixé par l'objectif du projet. Nous pouvons également améliorer le RMSE en ajoutant d'autres effets tels que le genre, l'âge. Des modèles d'apprentissage automatique complexes tels que les réseaux neuronaux, la filtration collaborative basée sur les articles peuvent également améliorer les résultats, mais des limitations matérielles telles que la RAM sont contraignantes.