#**MD Salek Miah**  
##B.Sc. in Statistics, Shahjalal University of Science and Technology, Sylhet  
###Statistician — Epidemiology & Public Health Enthusiast — Machine Learning Specialist  
###Sylhet, Bangladesh  
###📞 +880 1687 831074  
###[GitHub](https://github.com/yourusername) — ###[LinkedIn](https://www.linkedin.com/in/yourprofile) — ###[ORCID](https://orcid.org/yourorcid)

I have developed an interactive movie recommendation application using the MovieLens dataset. You can explore the app here:

<https://huggingface.co/spaces/salekml/MovieLens_App1>

Feel free to try it out and let me know your thoughts.

# MovieLens Recommendation System - Full R Workflow

# CRAN packages  
install.packages("shiny")

## Warning: package 'shiny' is in use and will not be installed

install.packages("dplyr")

## Warning: package 'dplyr' is in use and will not be installed

install.packages("shinycssloaders")

## Warning: package 'shinycssloaders' is in use and will not be installed

install.packages("shinyWidgets")

## Warning: package 'shinyWidgets' is in use and will not be installed

install.packages("recommenderlab")

## Warning: package 'recommenderlab' is in use and will not be installed

# For recosystem (from CRAN)  
install.packages("recosystem")

## Warning: package 'recosystem' is in use and will not be installed

# Optional: dependencies for shinyWidgets and shinycssloaders  
install.packages("htmltools")

## Warning: package 'htmltools' is in use and will not be installed

install.packages("RColorBrewer")

## package 'RColorBrewer' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

install.packages("magrittr")

## package 'magrittr' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'magrittr'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying E:\all  
## software\R s  
## tudio\RStudio\R-4.5.1\library\00LOCK\magrittr\libs\x64\magrittr.dll to E:\all  
## software\R s tudio\RStudio\R-4.5.1\library\magrittr\libs\x64\magrittr.dll:  
## Permission denied

## Warning: restored 'magrittr'

##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

install.packages("data.table")

## package 'data.table' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'data.table'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying E:\all  
## software\R s  
## tudio\RStudio\R-4.5.1\library\00LOCK\data.table\libs\x64\data\_table.dll to  
## E:\all software\R s  
## tudio\RStudio\R-4.5.1\library\data.table\libs\x64\data\_table.dll: Permission  
## denied

## Warning: restored 'data.table'

##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

#### 🎬 MovieLens Recommendation System ####  
#### Required Libraries ####  
  
  
# 1️⃣ Data Handling  
install.packages("data.table") # দ্রুত ডেটা লোড ও প্রক্রিয়াকরণ

## package 'data.table' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'data.table'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying E:\all  
## software\R s  
## tudio\RStudio\R-4.5.1\library\00LOCK\data.table\libs\x64\data\_table.dll to  
## E:\all software\R s  
## tudio\RStudio\R-4.5.1\library\data.table\libs\x64\data\_table.dll: Permission  
## denied

## Warning: restored 'data.table'

##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

# 2️⃣ Recommendation Algorithms  
install.packages("recommenderlab") # UBCF, IBCF, evaluation tools

## Warning: package 'recommenderlab' is in use and will not be installed

# 3️⃣ Visualization (optional but helpful)  
install.packages("ggplot2") # Evaluation plotting (Precision/Recall curves)

## package 'ggplot2' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

install.packages("gridExtra") # একাধিক প্লট একসাথে দেখাতে

## package 'gridExtra' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

install.packages("caret") # Machine learning helper tools (optional)

## package 'caret' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'caret'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying E:\all  
## software\R s tudio\RStudio\R-4.5.1\library\00LOCK\caret\libs\x64\caret.dll to  
## E:\all software\R s tudio\RStudio\R-4.5.1\library\caret\libs\x64\caret.dll:  
## Permission denied

## Warning: restored 'caret'

##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

# 5️⃣ Neural Embedding (optional advanced)  
install.packages("keras") # Deep learning-based recommendation (optional)

## package 'keras' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

# 6️⃣ Deployment API (for Hugging Face)  
install.packages("plumber") # API তৈরি করার জন্য

## package 'plumber' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

# 7️⃣ Miscellaneous  
install.packages("stringr") # string manipulation

## package 'stringr' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

install.packages("Matrix") # sparse matrix operations

## Warning: package 'Matrix' is in use and will not be installed

install.packages("readr") # fast CSV/TSV reading

## package 'readr' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'readr'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying E:\all  
## software\R s tudio\RStudio\R-4.5.1\library\00LOCK\readr\libs\x64\readr.dll to  
## E:\all software\R s tudio\RStudio\R-4.5.1\library\readr\libs\x64\readr.dll:  
## Permission denied

## Warning: restored 'readr'

##   
## The downloaded binary packages are in  
## C:\Users\Acer\AppData\Local\Temp\RtmpGQcgTT\downloaded\_packages

# 1. Install / Load Packages (non-interactive)  
# Non-interactive install helper (only install if missing)  
packages <- c(  
 "data.table","dplyr","tidyr","reshape2",  
 "recommenderlab","recosystem","ggplot2","gridExtra",  
 "Metrics","caret","keras","plumber","stringr","Matrix","readr"  
)  
  
  
install\_if\_missing <- function(pkgs){  
 for(p in pkgs){  
 if(!suppressWarnings(requireNamespace(p, quietly = TRUE))){  
 message("Installing: ", p)  
 if(p == "recommenderlab"){  
 # try CRAN first, then BiocManager  
 try(install.packages("recommenderlab", dependencies=TRUE), silent = TRUE)  
 if(!requireNamespace("recommenderlab", quietly = TRUE)){  
 if(!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")  
 BiocManager::install("recommenderlab", ask = FALSE)  
 }  
 } else {  
 install.packages(p, dependencies = TRUE)  
 }  
 }  
 }  
}  
  
  
install\_if\_missing(packages)  
  
  
#### Load Required Packages ####  
library(data.table)

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

library(dplyr)  
library(tidyr)

##   
## Attaching package: 'tidyr'

## The following objects are masked from 'package:Matrix':  
##   
## expand, pack, unpack

library(reshape2)

##   
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':  
##   
## smiths

## The following objects are masked from 'package:data.table':  
##   
## dcast, melt

library(recommenderlab)  
library(recosystem)  
library(ggplot2)  
library(gridExtra)

##   
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':  
##   
## combine

library(Metrics)  
library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following objects are masked from 'package:Metrics':  
##   
## precision, recall

## The following objects are masked from 'package:recommenderlab':  
##   
## MAE, RMSE

library(keras)

## The keras package is deprecated. Use the keras3 package instead.

##   
## Attaching package: 'keras'

## The following objects are masked from 'package:recommenderlab':  
##   
## evaluate, normalize

library(plumber)  
library(stringr)  
library(Matrix)  
library(readr)

## MovieLens Recommendation System - Full Workflow

# Load libraries  
library(data.table)  
library(dplyr)  
library(tidyr)  
library(tibble)  
library(ggplot2)  
library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:data.table':  
##   
## hour, isoweek, mday, minute, month, quarter, second, wday, week,  
## yday, year

## The following objects are masked from 'package:arules':  
##   
## intersect, setdiff, union

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

library(recommenderlab)  
library(recosystem)  
  
# 1️⃣ Set base path and load datasets  
base\_path <- "D:/job task/data/ml-100k/ml-100k/"  
  
ratings <- fread(file.path(base\_path, "u.data"), sep="\t",  
 col.names=c("userId","movieId","rating","timestamp"))  
  
movies <- fread(file.path(base\_path, "u.item"), sep="|", header=FALSE, quote="",  
 encoding="Latin-1", fill=TRUE) %>% select(movieId=V1, title=V2)

## Explore Ratings

head(ratings)

## userId movieId rating timestamp  
## <int> <int> <int> <int>  
## 1: 196 242 3 881250949  
## 2: 186 302 3 891717742  
## 3: 22 377 1 878887116  
## 4: 244 51 2 880606923  
## 5: 166 346 1 886397596  
## 6: 298 474 4 884182806

str(ratings)

## Classes 'data.table' and 'data.frame': 100000 obs. of 4 variables:  
## $ userId : int 196 186 22 244 166 298 115 253 305 6 ...  
## $ movieId : int 242 302 377 51 346 474 265 465 451 86 ...  
## $ rating : int 3 3 1 2 1 4 2 5 3 3 ...  
## $ timestamp: int 881250949 891717742 878887116 880606923 886397596 884182806 881171488 891628467 886324817 883603013 ...  
## - attr(\*, ".internal.selfref")=<externalptr>

summary(ratings)

## userId movieId rating timestamp   
## Min. : 1.0 Min. : 1.0 Min. :1.00 Min. :874724710   
## 1st Qu.:254.0 1st Qu.: 175.0 1st Qu.:3.00 1st Qu.:879448710   
## Median :447.0 Median : 322.0 Median :4.00 Median :882826944   
## Mean :462.5 Mean : 425.5 Mean :3.53 Mean :883528851   
## 3rd Qu.:682.0 3rd Qu.: 631.0 3rd Qu.:4.00 3rd Qu.:888259984   
## Max. :943.0 Max. :1682.0 Max. :5.00 Max. :893286638

colSums(is.na(ratings))

## userId movieId rating timestamp   
## 0 0 0 0

length(unique(ratings$userId))

## [1] 943

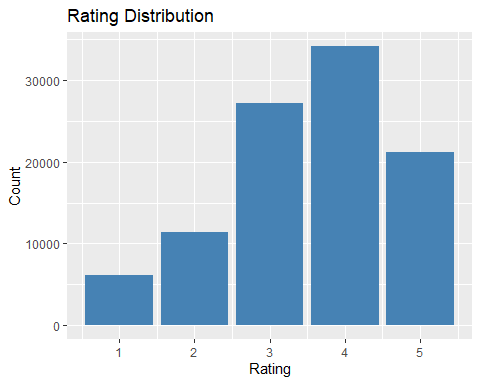
length(unique(ratings$movieId))

## [1] 1682

table(ratings$rating)

##   
## 1 2 3 4 5   
## 6110 11370 27145 34174 21201

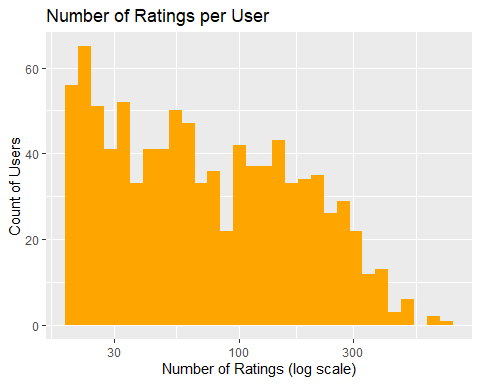
ggplot(ratings, aes(x=rating)) +  
 geom\_bar(fill="steelblue") +  
 labs(title="Rating Distribution", x="Rating", y="Count")



# Ratings per user  
user\_ratings <- ratings %>% group\_by(userId) %>% summarise(n\_ratings = n())  
summary(user\_ratings$n\_ratings)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 20 33 65 106 148 737

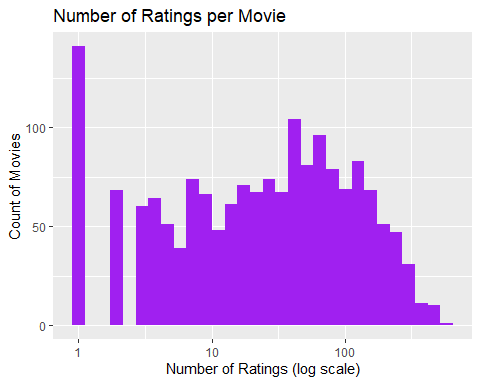
ggplot(user\_ratings, aes(x=n\_ratings)) +   
 geom\_histogram(bins=30, fill="orange") +  
 scale\_x\_log10() +  
 labs(title="Number of Ratings per User", x="Number of Ratings (log scale)", y="Count of Users")



# Ratings per movie  
movie\_ratings <- ratings %>% group\_by(movieId) %>% summarise(n\_ratings = n())  
summary(movie\_ratings$n\_ratings)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 1.00 6.00 27.00 59.45 80.00 583.00

ggplot(movie\_ratings, aes(x=n\_ratings)) +   
 geom\_histogram(bins=30, fill="purple") +  
 scale\_x\_log10() +  
 labs(title="Number of Ratings per Movie", x="Number of Ratings (log scale)", y="Count of Movies")



# Average movie ratings  
avg\_movie\_ratings <- ratings %>%  
 group\_by(movieId) %>%  
 summarise(avg\_rating = mean(rating), n\_ratings = n()) %>%  
 inner\_join(movies, by="movieId")  
  
top\_movies <- avg\_movie\_ratings %>% filter(n\_ratings >= 50) %>%  
 arrange(desc(avg\_rating)) %>% head(10)  
top\_movies

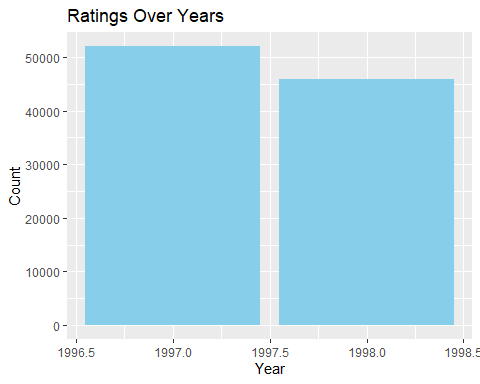
## # A tibble: 10 × 4  
## movieId avg\_rating n\_ratings title   
## <int> <dbl> <int> <chr>   
## 1 408 4.49 112 Close Shave, A (1995)   
## 2 318 4.47 298 Schindler's List (1993)   
## 3 169 4.47 118 Wrong Trousers, The (1993)   
## 4 483 4.46 243 Casablanca (1942)   
## 5 114 4.45 67 Wallace & Gromit: The Best of Aardman Animation…  
## 6 64 4.45 283 Shawshank Redemption, The (1994)   
## 7 603 4.39 209 Rear Window (1954)   
## 8 12 4.39 267 Usual Suspects, The (1995)   
## 9 50 4.36 583 Star Wars (1977)   
## 10 178 4.34 125 12 Angry Men (1957)

## Filter sparse dat

ratings\_filtered <- ratings %>%  
 group\_by(userId) %>% filter(n() >= 5) %>%  
 ungroup() %>%  
 group\_by(movieId) %>% filter(n() >= 10) %>%  
 ungroup()  
  
num\_users <- length(unique(ratings\_filtered$userId))  
num\_movies <- length(unique(ratings\_filtered$movieId))  
num\_ratings <- nrow(ratings\_filtered)  
sparsity <- 1 - (num\_ratings / (num\_users \* num\_movies))  
sparsity

## [1] 0.9098318

# Add datetime column  
ratings\_filtered <- ratings\_filtered %>%  
 mutate(rating\_date = as\_datetime(timestamp))  
  
ggplot(ratings\_filtered, aes(x=year(rating\_date))) +  
 geom\_bar(fill="skyblue") +  
 labs(title="Ratings Over Years", x="Year", y="Count")



## Create Rating Matrix

rating\_matrix <- ratings\_filtered %>%  
 select(userId, movieId, rating) %>%  
 pivot\_wider(names\_from = movieId, values\_from = rating) %>%  
 column\_to\_rownames("userId") %>%  
 as.matrix() %>%  
 as("realRatingMatrix")

## Train/Test Spli

set.seed(123)  
train\_index <- sample(x = c(TRUE, FALSE), size = nrow(rating\_matrix), replace = TRUE, prob = c(0.8, 0.2))  
train <- rating\_matrix[train\_index, ]  
test <- rating\_matrix[!train\_index, ]

## Build Collaborative Filtering Models ## User-Based CF

ubcf\_model <- Recommender(train, method = "UBCF", param = list(method = "Cosine", nn = 30))  
pred\_ubcf <- predict(ubcf\_model, test, type = "topNList", n = 10)  
pred\_list\_ubcf <- as(pred\_ubcf, "list")  
recommended\_titles\_ubcf <- lapply(pred\_list\_ubcf, function(ids) {  
 movies$title[movies$movieId %in% as.numeric(ids)]  
})  
  
# Item-Based CF  
ibcf\_model <- Recommender(train, method = "IBCF", param = list(method = "Cosine", k = 30))  
pred\_ibcf <- predict(ibcf\_model, test, type = "topNList", n = 10)  
pred\_list\_ibcf <- as(pred\_ibcf, "list")  
recommended\_titles\_ibcf <- lapply(pred\_list\_ibcf, function(ids) {  
 movies$title[movies$movieId %in% as.numeric(ids)]  
})

## SVD / Matrix Factorization

train\_file <- tempfile()  
write.table(ratings\_filtered %>% select(userId, movieId, rating),   
 file = train\_file, sep = " ", row.names = FALSE, col.names = FALSE)  
  
r <- Reco()  
r$train(data\_file(train\_file), opts = list(dim = 20, lrate = 0.1, niter = 20, verbose = TRUE))

## iter tr\_rmse obj  
## 0 1.2318 2.1871e+05  
## 1 0.9554 1.5910e+05  
## 2 0.9420 1.5699e+05  
## 3 0.9256 1.5538e+05  
## 4 0.9065 1.5328e+05  
## 5 0.8924 1.5179e+05  
## 6 0.8814 1.5084e+05  
## 7 0.8712 1.4982e+05  
## 8 0.8629 1.4893e+05  
## 9 0.8551 1.4842e+05  
## 10 0.8481 1.4789e+05  
## 11 0.8419 1.4739e+05  
## 12 0.8361 1.4701e+05  
## 13 0.8308 1.4674e+05  
## 14 0.8259 1.4653e+05  
## 15 0.8211 1.4600e+05  
## 16 0.8169 1.4569e+05  
## 17 0.8135 1.4557e+05  
## 18 0.8098 1.4531e+05  
## 19 0.8070 1.4518e+05

# Example: Predict Top-N for a user  
recommend\_movies <- function(user\_id, N = 10, model\_type = "SVD") {  
   
 if(model\_type == "SVD") {  
 user\_movies <- ratings\_filtered %>% filter(userId == user\_id) %>% pull(movieId)  
 unseen\_movies <- setdiff(unique(ratings\_filtered$movieId), user\_movies)  
 unseen\_df <- data.frame(user = user\_id, item = unseen\_movies)  
 pred\_ratings <- r$predict(data\_memory(user = unseen\_df$user, item = unseen\_df$item))  
   
 top\_n\_df <- data.frame(movieId = unseen\_movies, pred\_rating = pred\_ratings) %>%  
 arrange(desc(pred\_rating)) %>% head(N)  
   
 movies$title[movies$movieId %in% top\_n\_df$movieId]  
   
 } else if(model\_type == "UBCF") {  
 pred <- predict(ubcf\_model, rating\_matrix[user\_id,], type="topNList", n=N)  
 ids <- as(pred, "list")[[1]]  
 movies$title[movies$movieId %in% as.numeric(ids)]  
   
 } else if(model\_type == "IBCF") {  
 pred <- predict(ibcf\_model, rating\_matrix[user\_id,], type="topNList", n=N)  
 ids <- as(pred, "list")[[1]]  
 movies$title[movies$movieId %in% as.numeric(ids)]  
   
 } else stop("Invalid model\_type. Choose 'SVD', 'UBCF', or 'IBCF'.")  
}  
  
# Example usage  
recommend\_movies(user\_id = 5, N = 10, model\_type = "SVD")

## [1] "Wallace & Gromit: The Best of Aardman Animation (1996)"   
## [2] "Delicatessen (1991)"   
## [3] "Brazil (1985)"   
## [4] "Clockwork Orange, A (1971)"   
## [5] "My Man Godfrey (1936)"   
## [6] "Ran (1985)"   
## [7] "Rosencrantz and Guildenstern Are Dead (1990)"   
## [8] "City of Lost Children, The (1995)"   
## [9] "Double vie de Véronique, La (Double Life of Veronique, The) (1991)"  
## [10] "Hard Eight (1996)"

recommend\_movies(user\_id = 5, N = 10, model\_type = "UBCF")

## [1] "Heavy Metal (1981)" "Last of the Mohicans, The (1992)"   
## [3] "Muriel's Wedding (1994)" "Sunset Blvd. (1950)"   
## [5] "Wes Craven's New Nightmare (1994)" "Body Snatchers (1993)"   
## [7] "Sirens (1994)" "Farewell My Concubine (1993)"   
## [9] "Hamlet (1996)" "Old Man and the Sea, The (1958)"

recommend\_movies(user\_id = 5, N = 10, model\_type = "IBCF")

## [1] "Braveheart (1995)" "Net, The (1995)"   
## [3] "Madness of King George, The (1994)" "Lion King, The (1994)"   
## [5] "So I Married an Axe Murderer (1993)" "Reservoir Dogs (1992)"   
## [7] "Raiders of the Lost Ark (1981)" "Blues Brothers, The (1980)"   
## [9] "Sneakers (1992)" "Parent Trap, The (1961)"

## Evaluate Models ([Precision@K](mailto:Precision@K), [Recall@K](mailto:Recall@K), NDCG

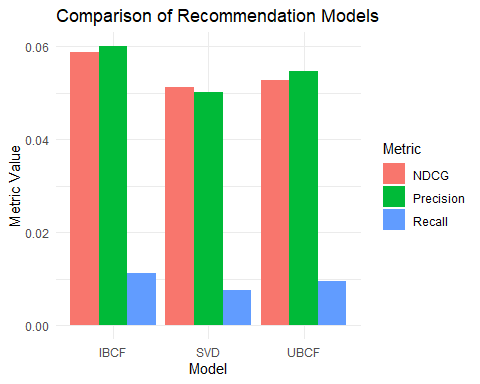
compute\_metrics <- function(topN\_list, test\_matrix, K = 10, goodRating = 4) {  
 precisions <- c(); recalls <- c(); ndcgs <- c()  
   
 for (i in 1:length(topN\_list)) {  
 user\_id <- names(topN\_list)[i]  
 recommended <- topN\_list[[i]]  
 user\_ratings <- as(test\_matrix[user\_id, ], "matrix")[1, ]  
 actual <- which(user\_ratings >= goodRating)  
   
 p <- length(intersect(recommended[1:K], actual)) / K  
 r <- if(length(actual) > 0) length(intersect(recommended[1:K], actual)) / length(actual) else 0  
 rel <- ifelse(recommended[1:K] %in% actual, 1, 0)  
 dcg <- sum(rel / log2(2:(K+1)))  
 idcg <- sum(1 / log2(2:(min(K, length(actual)) + 1)))  
 ndcg <- ifelse(idcg == 0, 0, dcg / idcg)  
   
 precisions <- c(precisions, p)  
 recalls <- c(recalls, r)  
 ndcgs <- c(ndcgs, ndcg)  
 }  
   
 data.frame(  
 Precision = mean(precisions, na.rm = TRUE),  
 Recall = mean(recalls, na.rm = TRUE),  
 NDCG = mean(ndcgs, na.rm = TRUE)  
 )  
}  
  
# Compute metrics  
topN\_svd\_list <- lapply(1:nrow(rating\_matrix), function(u) {  
 user\_movies <- ratings\_filtered %>% filter(userId == as.numeric(rownames(rating\_matrix)[u])) %>% pull(movieId)  
 unseen\_movies <- setdiff(unique(ratings\_filtered$movieId), user\_movies)  
 unseen\_df <- data.frame(user = as.numeric(rownames(rating\_matrix)[u]), item = unseen\_movies)  
 pred\_ratings <- r$predict(data\_memory(user = unseen\_df$user, item = unseen\_df$item))  
 unseen\_movies[order(-pred\_ratings)][1:10]  
})  
names(topN\_svd\_list) <- rownames(rating\_matrix)  
  
topN\_ubcf\_list <- lapply(1:nrow(rating\_matrix), function(u) {  
 pred <- predict(ubcf\_model, rating\_matrix[u, ], type="topNList", n=10)  
 as(pred, "list")[[1]]  
})  
names(topN\_ubcf\_list) <- rownames(rating\_matrix)  
  
topN\_ibcf\_list <- lapply(1:nrow(rating\_matrix), function(u) {  
 pred <- predict(ibcf\_model, rating\_matrix[u, ], type="topNList", n=10)  
 as(pred, "list")[[1]]  
})  
names(topN\_ibcf\_list) <- rownames(rating\_matrix)  
  
metrics\_svd <- compute\_metrics(topN\_svd\_list, rating\_matrix, K=10)  
metrics\_ubcf <- compute\_metrics(topN\_ubcf\_list, rating\_matrix, K=10)  
metrics\_ibcf <- compute\_metrics(topN\_ibcf\_list, rating\_matrix, K=10)  
metrics\_svd; metrics\_ubcf; metrics\_ibcf

## Precision Recall NDCG  
## 1 0.05015907 0.007458741 0.05130249

## Precision Recall NDCG  
## 1 0.05471898 0.009391098 0.05269209

## Precision Recall NDCG  
## 1 0.06002121 0.01115444 0.05881633

# Comparison Plot  
metrics <- data.frame(  
 Model = c("SVD", "UBCF", "IBCF"),  
 Precision = c(metrics\_svd$Precision, metrics\_ubcf$Precision, metrics\_ibcf$Precision),  
 Recall = c(metrics\_svd$Recall, metrics\_ubcf$Recall, metrics\_ibcf$Recall),  
 NDCG = c(metrics\_svd$NDCG, metrics\_ubcf$NDCG, metrics\_ibcf$NDCG)  
)  
metrics\_long <- tidyr::pivot\_longer(metrics, cols = Precision:NDCG, names\_to = "Metric", values\_to = "Value")  
ggplot(metrics\_long, aes(x = Model, y = Value, fill = Metric)) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 theme\_minimal() +  
 labs(title = "Comparison of Recommendation Models", y = "Metric Value")



## Save Objects for Deployment

saveRDS(movies, "D:/job task/ML/MovieLensShinyApp/movies.rds")  
saveRDS(ratings\_filtered, "D:/job task/ML/MovieLensShinyApp/ratings\_filtered.rds")  
saveRDS(rating\_matrix, "D:/job task/ML/MovieLensShinyApp/rating\_matrix.rds")  
saveRDS(r, "D:/job task/ML/MovieLensShinyApp/svd\_model.rds")  
saveRDS(ubcf\_model, "D:/job task/ML/MovieLensShinyApp/ubcf\_model.rds")  
saveRDS(ibcf\_model, "D:/job task/ML/MovieLensShinyApp/ibcf\_model.rds")  
saveRDS(topN\_svd\_list, "D:/job task/ML/MovieLensShinyApp/topN\_svd.rds")

## Final Recommendation Function & Deployment Prep

library(dplyr)  
library(recommenderlab)  
library(recosystem)  
library(tibble)  
library(ggplot2)  
library(lubridate)  
  
# Reusable function for Top-N recommendations  
recommend\_movies <- function(user\_id, N = 10, model\_type = "SVD") {  
   
 if(model\_type == "SVD") {  
 # Movies already rated by user  
 user\_movies <- ratings\_filtered %>% filter(userId == user\_id) %>% pull(movieId)  
 unseen\_movies <- setdiff(unique(ratings\_filtered$movieId), user\_movies)  
   
 unseen\_df <- data.frame(user = user\_id, item = unseen\_movies)  
 pred\_ratings <- r$predict(data\_memory(user = unseen\_df$user, item = unseen\_df$item))  
   
 top\_n\_df <- data.frame(movieId = unseen\_movies, pred\_rating = pred\_ratings) %>%  
 arrange(desc(pred\_rating)) %>%  
 head(N)  
   
 movies$title[movies$movieId %in% top\_n\_df$movieId]  
   
 } else if(model\_type == "UBCF") {  
 pred <- predict(ubcf\_model, rating\_matrix[user\_id,], type="topNList", n=N)  
 ids <- as(pred, "list")[[1]]  
 movies$title[movies$movieId %in% as.numeric(ids)]  
   
 } else if(model\_type == "IBCF") {  
 pred <- predict(ibcf\_model, rating\_matrix[user\_id,], type="topNList", n=N)  
 ids <- as(pred, "list")[[1]]  
 movies$title[movies$movieId %in% as.numeric(ids)]  
   
 } else {  
 stop("Invalid model\_type. Choose 'SVD', 'UBCF', or 'IBCF'.")  
 }  
}  
  
# Example usage  
recommend\_movies(user\_id = 5, N = 10, model\_type = "SVD")

## [1] "Wallace & Gromit: The Best of Aardman Animation (1996)"   
## [2] "Delicatessen (1991)"   
## [3] "Brazil (1985)"   
## [4] "Clockwork Orange, A (1971)"   
## [5] "My Man Godfrey (1936)"   
## [6] "Ran (1985)"   
## [7] "Rosencrantz and Guildenstern Are Dead (1990)"   
## [8] "City of Lost Children, The (1995)"   
## [9] "Double vie de Véronique, La (Double Life of Veronique, The) (1991)"  
## [10] "Hard Eight (1996)"

recommend\_movies(user\_id = 5, N = 10, model\_type = "UBCF")

## [1] "Heavy Metal (1981)" "Last of the Mohicans, The (1992)"   
## [3] "Muriel's Wedding (1994)" "Sunset Blvd. (1950)"   
## [5] "Wes Craven's New Nightmare (1994)" "Body Snatchers (1993)"   
## [7] "Sirens (1994)" "Farewell My Concubine (1993)"   
## [9] "Hamlet (1996)" "Old Man and the Sea, The (1958)"

recommend\_movies(user\_id = 5, N = 10, model\_type = "IBCF")

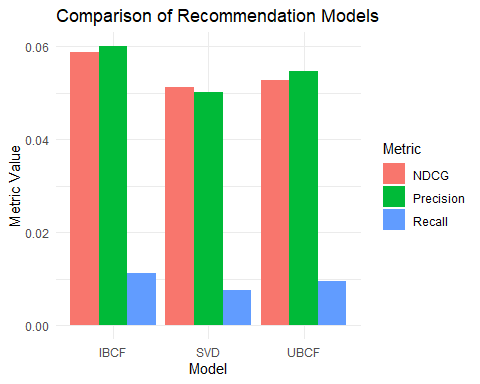
## [1] "Braveheart (1995)" "Net, The (1995)"   
## [3] "Madness of King George, The (1994)" "Lion King, The (1994)"   
## [5] "So I Married an Axe Murderer (1993)" "Reservoir Dogs (1992)"   
## [7] "Raiders of the Lost Ark (1981)" "Blues Brothers, The (1980)"   
## [9] "Sneakers (1992)" "Parent Trap, The (1961)"

## Save All Objects for Deployment (New Path)

saveRDS(movies, "D:/job task/ML/MovieLensShinyApp/movies.rds")  
saveRDS(ratings\_filtered, "D:/job task/ML/MovieLensShinyApp/ratings\_filtered.rds")  
saveRDS(rating\_matrix, "D:/job task/ML/MovieLensShinyApp/rating\_matrix.rds")  
saveRDS(r, "D:/job task/ML/MovieLensShinyApp/svd\_model.rds")  
saveRDS(ubcf\_model, "D:/job task/ML/MovieLensShinyApp/ubcf\_model.rds")  
saveRDS(ibcf\_model, "D:/job task/ML/MovieLensShinyApp/ibcf\_model.rds")  
saveRDS(topN\_svd\_list, "D:/job task/ML/MovieLensShinyApp/topN\_svd.rds")

## Short Report Preparation

# Compare metrics of SVD, UBCF, IBCF  
metrics <- data.frame(  
 Model = c("SVD", "UBCF", "IBCF"),  
 Precision = c(metrics\_svd$Precision, metrics\_ubcf$Precision, metrics\_ibcf$Precision),  
 Recall = c(metrics\_svd$Recall, metrics\_ubcf$Recall, metrics\_ibcf$Recall),  
 NDCG = c(metrics\_svd$NDCG, metrics\_ubcf$NDCG, metrics\_ibcf$NDCG)  
)  
  
metrics\_long <- tidyr::pivot\_longer(metrics, cols = Precision:NDCG, names\_to = "Metric", values\_to = "Value")  
  
ggplot(metrics\_long, aes(x = Model, y = Value, fill = Metric)) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 theme\_minimal() +  
 labs(title = "Comparison of Recommendation Models", y = "Metric Value")



# Save metrics plot for report  
ggsave("D:/job task/ML/MovieLensShinyApp/model\_comparison.png", width = 8, height = 5)