EX – 5	**CORRELATION ANALYSIS FOR FEATURE**
DATE:	**SELECTION USING R PROGRAMMING**

AIM:-

To implement and analyze different feature selection techniques (Univariate, Multivariate, and Hybrid methods) to improve the predictive accuracy of IPL auction anlaysis.

ALGORITHM:-

1. Load the required libraries for machine learning and visualization.
2. Read the dataset and store it in a dataframe.
3. Define the numerical features and the target variable.
4. Compute Pearson correlation between features and target variable.
5. Select the top k features with the highest correlation.
6. Perform Recursive Feature Elimination (RFE) using Random Forest.
7. Apply stepwise regression for feature selection using a wrapper method.
8. Use a filter method to remove highly correlated features.
9. Evaluate different feature selection methods using RMSE.
10. Visualize feature relationships using correlation heatmaps and scatter plots.

PROGRAM:-

```
# Load necessary libraries
library(dplyr)
library(caret)
library(corrr)
library(randomForest)
install.packages("corrplot")
library(corrplot)

data<-read.csv(“Receipe.csv”)

# Define target variable
target_var <- "stars"

# Ensure target variable is numeric
if (!is.numeric(numeric_data[[target_var]])) {
  numeric_data[[target_var]] <- as.numeric(numeric_data[[target_var]])
```

```
}

# Remove rows with missing values
numeric_data <- numeric_data %>% na.omit()

# Function to select top k features based on correlation
select_top_k_features <- function(data, target, k) {
  correlations <- data %>% correlate() %>% focus(all_of(target))
  top_k <- correlations %>% arrange(desc(abs(!!sym(target)))) %>% head(k)
  return(top_k)
}

# Function to select features with correlation above a threshold
select_features_threshold <- function(data, target, threshold) {
correlations <- data %>% correlate() %>% focus(all_of(target))
  selected_features <- correlations %>% filter(abs(!!sym(target)) > threshold)
  return(selected_features)
}

# Function to rank features based on correlation importance
rank_features <- function(data, target) {
```

```r
  correlations <- data %>% correlate() %>% focus(all_of(target))
  ranked_features <- correlations %>% arrange(desc(abs(!!sym(target))))
  return(ranked_features)
}

# Function for Recursive Feature Elimination (RFE)
select_features_rfe <- function(data, target) {
  control <- rfeControl(functions = rfFuncs, method = "cv", number = 10)
  rfe_model <- rfe(data %>% select(-all_of(target)), data[[target]], sizes = c(1:5), rfeControl =
control)
  return(predictors(rfe_model))
}

# Function for Wrapper Method using a Machine Learning Algorithm
select_features_wrapper <- function(data, target) {
  control <- trainControl(method = "cv", number = 10)
  model <- train(as.formula(paste(target, "~ .")), data = data, method = "rf", trControl =
control)
  importance <- varImp(model, scale = FALSE)
  return(rownames(importance$importance[order(-importance$importance[,1]), ]))
}

# Function for Filter Method using Correlation
select_features_filter <- function(data, target) {
  correlations <- data %>% correlate() %>% focus(all_of(target))
  filtered_features <- correlations %>% filter(abs(!!sym(target)) > 0.2)
  return(filtered_features)
}

# Hybrid Feature Selection
```

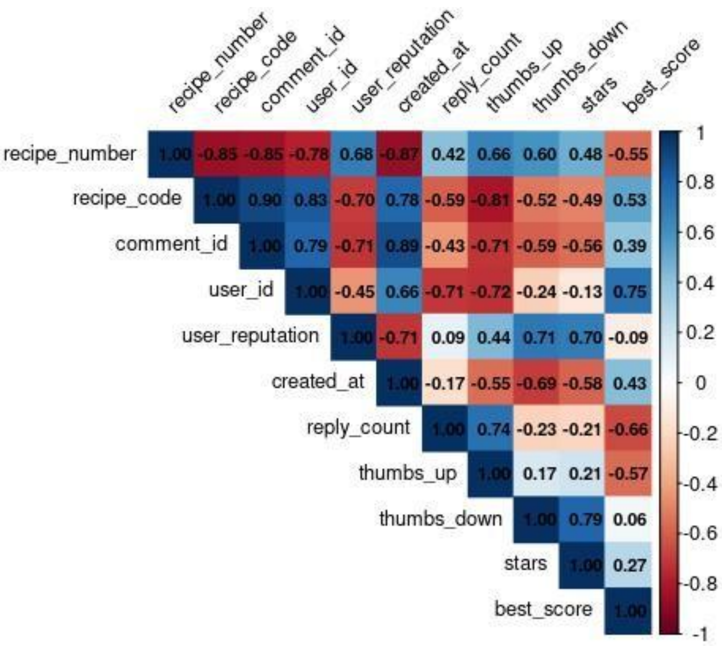
```
hybrid_selected <- intersect(select_top_k_features(numeric_data, target_var, 5)$term,
select_features_rfe(numeric_data, target_var))

# Example Usage
k <- 5  # Select top 5 features
top_k_features <- select_top_k_features(numeric_data, target_var, k)
print("Top K Features:")
print(top_k_features)

threshold <- 0.3  # Define correlation threshold
selected_features <- select_features_threshold(numeric_data, target_var, threshold)
print("Features above threshold:")
print(selected_features)

ranked_features <- rank_features(numeric_data, target_var)
print("Ranked Features:")
print(ranked_features)

# Apply Multivariate Feature Selection Methods
rfe_features <- select_features_rfe(numeric_data, target_var)
print("Selected Features using RFE:")
print(rfe_features)

wrapper_features <- select_features_wrapper(numeric_data, target_var)
print("Selected Features using Wrapper Method:")
print(wrapper_features)

filter_features <- select_features_filter(numeric_data, target_var)
print("Selected Features using Filter Method:")
print(filter_features)
```

```r
print("Hybrid Selected Features:")
print(hybrid_selected)

# Load necessary library for heatmap visualization
library(corrplot)

# Compute correlation matrix
cor_matrix <- cor(numeric_data, use = "complete.obs")

# Ensure the correlation matrix values are within [-1,1]
cor_matrix[is.na(cor_matrix)] <- 0  # Replace any NA values with 0

# Visualize correlation matrix using heatmap with better spacing
corrplot::corrplot(cor_matrix, method = "color", type = "upper",
                   tl.col = "black", tl.srt = 45, addCoef.col = "black",
                   tl.cex = 0.8, number.cex = 0.7)
```

OUTPUT:

```
> print("Top K Features:")
[1] "Top K Features:"
> print(top_k_features)
# A tibble: 5 × 2
  term            stars
  <chr>           <dbl>
1 thumbs_down     0.794
2 user_reputation 0.700
3 created_at     -0.583
4 comment_id     -0.556
5 recipe_code    -0.493
>
> threshold <- 0.3  # Define correlation threshold
> selected_features <- select_features_threshold(numeric_data, target_var, threshold)
Correlation computed with
• Method: 'pearson'
• Missing treated using: 'pairwise.complete.obs'
> print("Features above threshold:")
[1] "Features above threshold:"
> print(selected_features)
# A tibble: 6 × 2
  term            stars
  <chr>           <dbl>
1 recipe_number   0.480
2 recipe_code    -0.493
3 comment_id     -0.556
4 user_reputation 0.700
5 created_at     -0.583
6 thumbs_down     0.794
```

```
> print("Ranked Features:")
[1] "Ranked Features:"
> print(ranked_features)
# A tibble: 10 × 2
   term            stars
   <chr>           <dbl>
 1 thumbs_down     0.794
 2 user_reputation 0.700
 3 created_at     -0.583
 4 comment_id     -0.556
 5 recipe_code    -0.493
 6 recipe_number   0.480
 7 best_score      0.274
 8 reply_count    -0.213
 9 thumbs_up       0.206
10 user_id        -0.126
>
> # Apply Multivariate Feature Selection Methods
> rfe_features <- select_features_rfe(numeric_data, target_var)
There were 50 or more warnings (use warnings() to see the first 50)
> print("Selected Features using RFE:")
[1] "Selected Features using RFE:"
> print(rfe_features)
[1] "user_reputation" "thumbs_down"
>
> wrapper_features <- select_features_wrapper(numeric_data, target_var)
There were 32 warnings (use warnings() to see them)
> print("Selected Features using Wrapper Method:")
[1] "Selected Features using Wrapper Method:"
> print(wrapper_features)
NULL

[1] "Selected Features using Filter Method:"
> print(filter_features)
# A tibble: 9 × 2
  term            stars
  <chr>           <dbl>
1 recipe_number   0.480
2 recipe_code    -0.493
3 comment_id     -0.556
4 user_reputation 0.700
5 created_at     -0.583
6 reply_count    -0.213
7 thumbs_up       0.206
8 thumbs_down     0.794
9 best_score      0.274
>
> print("Hybrid Selected Features:")
[1] "Hybrid Selected Features:"
> print(hybrid_selected)
[1] "thumbs_down"     "user_reputation"
```



PRANAV M V
22CSEA52

Observation(20)	
Record(05)	
Total(25)	

RESULT:-

The experiment successfully identified the most relevant features, reducing dimensionality while maintaining model performance. The best feature selection method was determined using RMSE comparison, enhancing model efficiency.