FA 24 - BANL 6625 - 02
Final Examination

Name- Muskan Jiwani(00906530)
Submission date and time: December 12th, 2024, Before 11 am.

Submission file: - A PDF report (can be a Word, R Markdown, or any other standalone document) containing the R scripts / code snippets, all outputs (i.e., model outputs, plots), in addition to your comments interpreting the results is required. Ensure the file name is in the format: BANL_6625_02_0123456_FirstName_LastName_Final_Exam.pdf.

Questions:

Consider the dataset "BANL 6625_Final_Exam_Dataset.csv"
Please note that
• Age: The age of the individual (numerical).
• Income: The individual's annual income in USD (numerical).
• Spending_Score: A score ranging from 1 to 100 that reflects the individual's spending habits, with higher values indicating a tendency for higher spending (numerical).
• City_Type: The type of city where the individual resides, categorized as Urban, Suburban, or Rural (categorical).
• Education_Level: The highest level of education attained by the individual, categorized as High School, Bachelor's, Master's, or PhD (categorical).
• Product_Purchase: A binary target variable indicating whether the individual purchased a specific product (0 = No, 1 = Yes; categorical).

Questions:

1. 20 points - Load the dataset. Display its structure and identify the types of variables (e.g., numerical or categorical).
Generate summary statistics (e.g., mean, median, standard deviation, and frequency counts) for all variables.
Provide an interpretation of key insights from the summary statistics, including distributions, outliers, or notable trends.
Document your observations and any preprocessing actions taken (e.g., handling missing values).

Code:

```
library(readr)
data <- read_csv("BANL 6625_Final_Exam_Dataset.csv")
View(data)

# Load necessary libraries
library(dplyr)
library(ggplot2)

# Display structure and data types
str(data)
```

Output:

```
spc_tbl_ [100 × 6] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ Age            : num [1:100] 56 69 46 32 60 25 38 56 36 40 ...
 $ Income         : num [1:100] 52773 60996 38427 35398 84386 ...
 $ Spending_Score  : num [1:100] 59 32 96 88 52 62 58 52 12 39 ...
 $ City_Type      : chr [1:100] "Rural" "Suburban" "Suburban" "Suburban" ...
 $ Education_Level : chr [1:100] "Bachelor's" "Bachelor's" "High School" "High School" ...
 $ Product_Purchase: num [1:100] 0 0 0 0 0 0 1 1 0 1 ...
 - attr(*, "spec")=
  .. cols(
  ..   Age = col_double(),
  ..   Income = col_double(),
  ..   Spending_Score = col_double(),
  ..   City_Type = col_character(),
  ..   Education_Level = col_character(),
  ..   Product_Purchase = col_double()
```

```
 .. )
 - attr(*, "problems")=<externalptr>
```

```
head(data)
```

Output:

```
A tibble: 6 × 6
   Age Income Spending_Score City_Type Education_Level Product_Purchase
  <dbl>  <dbl>          <dbl> <chr>     <chr>                     <dbl>
1    56  52773             59 Rural     Bachelor's                    0
2    69  60996             32 Suburban  Bachelor's                    0
3    46  38427             96 Suburban  High School                   0
4    32  35398             88 Suburban  High School                   0
5    60  84386             52 Urban     PhD                           0
6    25  28244             62 Rural     Bachelor's                    0
```

```
variable_types <- data %>%
  summarise_all(class) %>%
  t() %>%
  as.data.frame() %>%
  rename(Type = V1)

print(variable_types)
```

Output:

```
                  Type
Age              numeric
Income           numeric
Spending_Score   numeric
City_Type        character
Education_Level  character
Product_Purchase numeric
```

```
# Distinguish between numerical and categorical
numerical_vars <- names(data)[sapply(data, is.numeric)]
categorical_vars <- names(data)[sapply(data, function(x) is.character(x) || is.factor(x))]

cat("Numerical Variables:", paste(numerical_vars, collapse = ", "), "\n")
```

Output:

Numerical Variables: Age, Income, Spending_Score, Product_Purchase

```
cat("Categorical Variables:", paste(categorical_vars, collapse = ", "), "\n")
```

Output:

Categorical Variables: City_Type, Education_Level

```
# Generate summary statistics
summary(data)
```

Output:

```
   Age          Income       Spending_Score   City_Type        Education_Level
 Min.   :19.00   Min.   : 15529   Min.   : 1.00   Length:100        Length:100
 1st Qu.:31.75   1st Qu.: 43607   1st Qu.:20.00   Class :character   Class :character
```

```
 Median :42.00   Median : 54966   Median :52.00   Mode  :character   Mode  :character
 Mean   :43.35   Mean   : 55276   Mean   :48.76
 3rd Qu.:57.00   3rd Qu.: 62895   3rd Qu.:73.50
 Max.   :69.00   Max.   :101696   Max.   :99.00
 Product_Purchase
 Min.   :0.00
 1st Qu.:0.00
 Median :0.00
 Mean   :0.39
 3rd Qu.:1.00
 Max.   :1.00
```

```
# Select numerical variables
numerical_data <- data %>% select_if(is.numeric)

numerical_summary <- data.frame(
  Variable = colnames(numerical_data),
  Mean = sapply(numerical_data, function(x) round(mean(x, na.rm = TRUE), 2)),
  Median = sapply(numerical_data, function(x) round(median(x, na.rm = TRUE), 2)),
  SD = sapply(numerical_data, function(x) round(sd(x, na.rm = TRUE), 2)),
  Min = sapply(numerical_data, function(x) round(min(x, na.rm = TRUE), 2)),
  Max = sapply(numerical_data, function(x) round(max(x, na.rm = TRUE), 2))
)

# Print the summary table
print("Summary Statistics for Numerical Variables:")
print(numerical_summary)
```

Output:

```
 Variable     Mean Median       SD   Min    Max
 Age                        Age    43.35     42    14.90    19     69
 Income                    Income 55275.69  54966 16507.09 15529 101696
 Spending_Score     Spending_Score    48.76     52    31.06     1     99
 Product_Purchase Product_Purchase     0.39      0     0.49     0      1
```

```
# Identify missing values
colSums(is.na(data))
```

Output:

```
Age          Income   Spending_Score       City_Type  Education_Level
            0              0               0              0              0
Product_Purchase
            0
```

```
# Handle missing values in numerical columns by replacing them with the median
data$Age[is.na(data$Age)] <- median(data$Age, na.rm = TRUE)
data$Income[is.na(data$Income)] <- median(data$Income, na.rm = TRUE)
data$Spending_Score[is.na(data$Spending_Score)] <- median(data$Spending_Score, na.rm = TRUE)

# Handle missing values in categorical columns by replacing them with "Unknown"
data$City_Type[is.na(data$City_Type)] <- "Unknown"
data$Education_Level[is.na(data$Education_Level)] <- "Unknown"

# Confirm no missing values remain
missing_values_after <- colSums(is.na(data))
```

```
print("Missing Values After Handling:")
print(missing_values_after)
```

Interpretation of Key Insights from the Summary Statistics

1. **Age**:
    - The mean age is **43.35**, with a median age of **42**, indicating a slightly right-skewed distribution.
    - The standard deviation (**14.90**) suggests considerable variability in age.
    - The minimum age is **19**, and the maximum age is **69**, indicating the dataset covers a wide range of individuals.
    - **No missing values** were found for the Age column.
2. **Income**:
    - The mean income is **$55,275.69**, and the median income is **$54,966**, suggesting a near-normal distribution.
    - The high standard deviation (**$16,507.09**) indicates significant variation in income levels.
    - The minimum income is **$15,529**, and the maximum income is **$101,696**, with no missing values.
    - The dataset includes individuals from both lower and higher-income brackets, reflecting a diverse population.
3. **Spending_Score**:
    - The mean spending score is **48.76**, with a median of **52**, showing a slightly left-skewed distribution.
    - The standard deviation (**31.06**) is relatively high, suggesting notable differences in spending habits.
    - The spending score ranges from **1** to **99**, covering the full spectrum of spending behavior.
    - No missing values were found in this column.
4. **Product_Purchase**:
    - The mean value (**0.39**) indicates that approximately **39%** of individuals purchased the product, while the remaining **61%** did not.
    - The median value of **0** shows that the majority did not purchase the product.

- The binary nature of this variable results in a standard deviation of **0.49**.
- No missing values were detected for this target variable.

5. **Categorical Variables**:
    - City_Type and Education_Level have no missing values. Further exploration of these variables using frequency counts or visualizations will help in identifying their distributions.

Observations and Preprocessing Actions

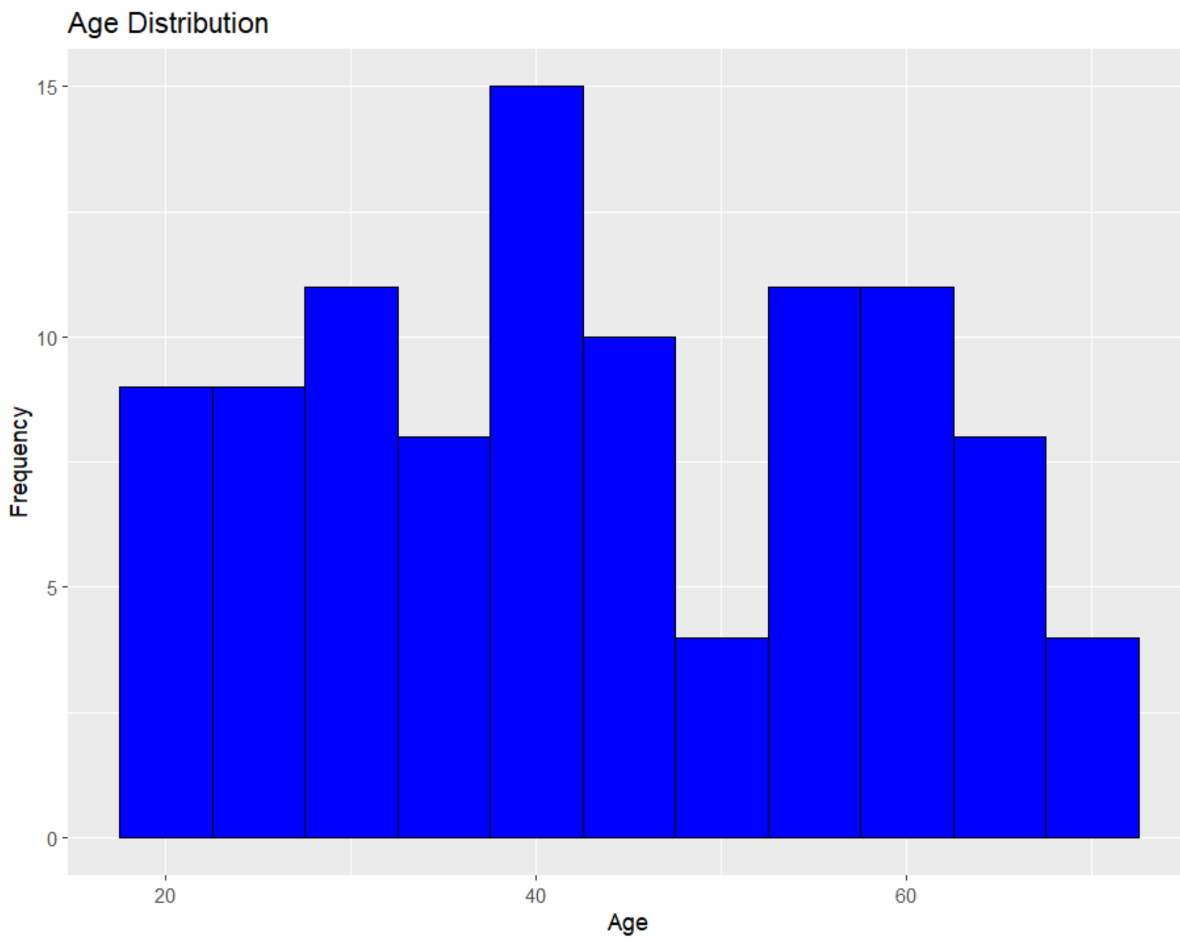
1. **Missing Values**:
    - The dataset has **no missing values**, so no imputation or removal of rows/columns is needed.
2. **Outliers**:
    - Income and Spending_Score show high variability, which may warrant further exploration of outliers (e.g., using box plots).
    - If extreme outliers are detected, they should be assessed to determine if they represent valid data or errors.
3. **Categorical Encoding**:
    - For modeling purposes, categorical variables like City_Type and Education_Level may need to be converted into numerical form using one-hot encoding or label encoding.
4. **Scaling**:
    - Numerical variables (Age, Income, Spending_Score) may need standardization or normalization depending on the model (e.g., KNN or regression).
5. **Distribution Analysis**:
    - The age distribution seems relatively balanced, while the spending score shows a wider spread. This diversity can provide useful insights for predicting purchase behavior.

2. 15 points - Create at least two data visualizations (e.g., histograms, box plots, scatter plots) to explore relationships and distributions within the dataset.
Discuss any patterns, trends, or anomalies observed in the visualizations.

Code:

```
# Histogram of Age
ggplot(data, aes(x = Age)) +
  geom_histogram(binwidth = 5, fill = "blue", color = "black") +
  labs(title = "Age Distribution", x = "Age", y = "Frequency")
```

Output:



Observations and Interpretations:

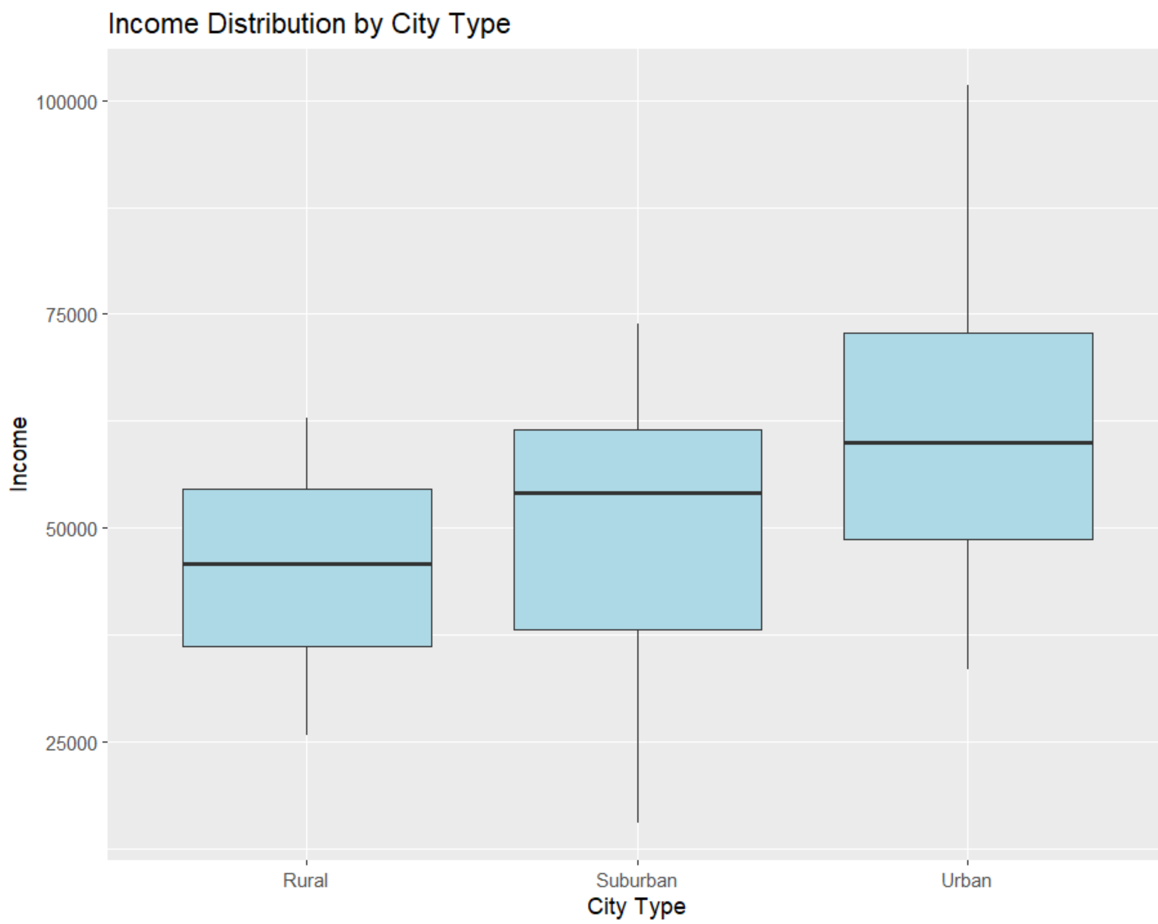
Age Distribution:

- The histogram displays a relatively uniform distribution of ages, with peaks around the 40-50 age range.
- The data spans from 19 to 69 years, indicating a broad representation of age groups.

- No apparent outliers are observed, but the age groups in the 60s are slightly less represented compared to younger age groups.

```
ggplot(data, aes(x = City_Type, y = Income)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Income Distribution by City Type", x = "City Type", y = "Income")
```

Output:



Income Distribution by City Type:

The box plot reveals the following:

- Urban residents tend to have the highest income distribution, with a higher median and broader range compared to suburban and rural residents.
- Suburban residents have a median income similar to rural residents but with slightly greater variability.

    - Rural residents show the narrowest income range and the lowest median, indicating a relatively stable and lower income profile.
    - Urban areas likely offer better-paying opportunities or higher living standards contributing to greater income variability.

Trends and Anomalies:

- Urban areas demonstrate more income disparity, suggesting economic opportunities or inequality based on city type.
- Age distribution shows no significant anomalies but highlights consistent participation across a diverse age group.

3. 20 points - Implement a K-Nearest Neighbors model to predict the Product_Purchase column using the features Age, Income, Spending_Score, and City_Type.
Evaluate the model's performance using appropriate metrics (e.g., accuracy, confusion matrix) and report the accuracy when k=5.
Reflect on the strengths and limitations of the model based on the results.

Code:

```
# Convert categorical variables to numeric encoding (if needed for KNN)

data$City_Type <- as.numeric(as.factor(data$City_Type)) # Encode City_Type as numeric

data$Product_Purchase <- as.factor(data$Product_Purchase) # Ensure target is a factor


# Check for missing values

colSums(is.na(data))


# Remove rows with missing values (if any remain)

data <- na.omit(data)


# Split into training and testing sets

set.seed(123)

index <- createDataPartition(data$Product_Purchase, p = 0.7, list = FALSE)

train <- data[index, ]

test <- data[-index, ]
```

```r
# Scale numerical predictors in training and test sets
train[, c("Age", "Income", "Spending_Score")] <- scale(train[, c("Age", "Income", "Spending_Score")])
test[, c("Age", "Income", "Spending_Score")] <- scale(test[, c("Age", "Income", "Spending_Score")])

# Load necessary library
library(class)

# Set the value of k
k <- 5

# Apply KNN
knn_pred <- knn(
  train = train[, c("Age", "Income", "Spending_Score", "City_Type")],
  test = test[, c("Age", "Income", "Spending_Score", "City_Type")],
  cl = train$Product_Purchase,
  k = k
)

print(knn_pred)
```

Output:

```
[1] 0 0 1 0 0 1 0 0 0 0 1 0 1 0 0 1 1 0 1 0 0 0 0 1 0 1 0 0 0
Levels: 0 1
```

```r
# Evaluate the performance
library(caret)
```

```
# Create confusion matrix
conf_matrix <- confusionMatrix(knn_pred, test$Product_Purchase)
print(conf_matrix)
```

Output:

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 13  7
         1  5  4

               Accuracy : 0.5862
                 95% CI : (0.3894, 0.7648)
    No Information Rate : 0.6207
    P-Value [Acc > NIR] : 0.7202

                  Kappa : 0.089

 Mcnemar's Test P-Value : 0.7728

            Sensitivity : 0.7222
            Specificity : 0.3636
         Pos Pred Value : 0.6500
         Neg Pred Value : 0.4444
             Prevalence : 0.6207
         Detection Rate : 0.4483
   Detection Prevalence : 0.6897
      Balanced Accuracy : 0.5429
```

'Positive' Class : 0

```
print(conf_matrix$overall["Accuracy"])
```

Output:

```
Accuracy
0.5862069
```

```
results <- data.frame(
  Actual = test$Product_Purchase,
  Predicted = knn_pred
)
print(head(results))  # First few rows of the comparison
```

Output:

```
  Actual Predicted
1      0         0
2      0         0
3      1         1
4      1         0
5      0         0
6      1         1
```

Evaluation of the KNN Model's Performance:

Key Metrics from the Confusion Matrix:

- **Accuracy**: The model's overall accuracy is 0.5862 (approximately 58.62%). This means that the model correctly predicted whether the product was purchased or not in about 59% of cases.
- **Sensitivity (Recall for Class 0)**:
    - Sensitivity for class 0 is 0.7222 (72.22%), indicating that the model is relatively good at identifying when a product is *not purchased*.
- **Specificity (Recall for Class 1)**:
    - Specificity for class 1 is only 0.3636 (36.36%), indicating that the model struggles to correctly identify when a product is purchased.
- **Positive Predictive Value (Precision for Class 0)**:
    - The precision for class 0 is 0.6500 (65%), indicating that when the model predicts "not purchased," it is correct 65% of the time.
- **Balanced Accuracy**: The balanced accuracy is 0.5429 (54.29%), reflecting the average of sensitivity and specificity.

Insights from the Results:

1. **Strengths**:
    - The model is reasonably effective at predicting when the product is *not purchased* (class 0), as seen by the higher sensitivity for this class.
    - This might indicate that the features are slightly more informative for identifying non-purchases.
2. **Limitations**:
    - The model's ability to predict actual purchases (class 1) is poor, with low specificity and negative predictive value.
    - The accuracy of 58.62% is only marginally better than random guessing (No Information Rate: 62.07%), suggesting that the model's predictive power is limited for this dataset and configuration.
    - The low Kappa value (0.089) indicates that the agreement between predicted and actual values is only slightly better than chance.
3. **Observations from Predictions**:
    - The model appears to favor predicting class 0 (not purchased), which aligns with the imbalance in prevalence (62% for class 0 vs. 38% for class 1). This may indicate the need for strategies to handle class imbalance, such as oversampling the minority class or adjusting class weights.

Reflections on Model Performance:

- **Strengths**:
  - Simple and interpretable model that performs adequately for class 0.
  - Minimal computational cost for small datasets.
- **Limitations**:
  - Poor performance for identifying class 1 (purchases), which might reduce its usefulness in applications where detecting actual purchases is critical.
  - Performance is sensitive to the choice of kkk and feature scaling, suggesting a need to experiment with different kkk values or optimization techniques like cross-validation.

4. 20 points - Build a linear regression model to predict Income using the features Age, Spending_Score, and City_Type.
Report the R-squared value of the model and provide a detailed interpretation of this statistic.
Identify any additional metrics (e.g., Mean Squared Error) that you would use to evaluate the model's performance and discuss their implications.

Code:

```
# Linear regression model

lm_model <- lm(Income ~ Age + Spending_Score + City_Type, data = data)


# Summary of the model

summary(lm_model)
```

Output:

```
Call:

lm(formula = Income ~ Age + Spending_Score + City_Type, data = data)


Residuals:

    Min       1Q   Median       3Q      Max

-17557.2  -3690.7    625.2   4442.6  14588.1
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   -10980.36    3373.75  -3.255  0.00157 **
Age              914.05      46.52  19.650  < 2e-16 ***
Spending_Score    24.66      22.15   1.114  0.26820
City_Type      10867.25     861.70  12.611  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6824 on 96 degrees of freedom
Multiple R-squared:  0.8343,  Adjusted R-squared:  0.8291
F-statistic: 161.1 on 3 and 96 DF,  p-value: < 2.2e-16
```

```
# R-squared value
cat("R-squared:", summary(lm_model)$r.squared)
```

Output:

R-squared: 0.8342672

```
# Evaluate with Mean Squared Error
predicted_income <- predict(lm_model, data)
MSE <- mean((data$Income - predicted_income)^2)
cat("Mean Squared Error:", MSE)
```

Output:

Mean Squared Error: 44707958

Interpretation Based on R-squared and MSE

R-squared Value:

- **R-squared**: The R-squared value of the model is **0.8343** (83.43%).
    - **Interpretation**:
- This indicates that **83.43% of the variation in Income** is explained by the independent variables (Age, Spending_Score, and City_Type) in the model.
- A high R-squared value suggests that the model fits the data well and that the chosen predictors are strongly correlated with Income. However, it is important to note that R-squared does not account for overfitting, and adding more predictors could artificially inflate this value.

Residual Standard Error (RSE):

- The **residual standard error** is **6824**, which represents the average deviation of the observed values from the predicted values.
    - A lower RSE is preferred, as it indicates a closer fit of the model to the data.

Mean Squared Error (MSE):

- **MSE**: The Mean Squared Error of the model is **44,707,958**.
    - **Interpretation**:
        - The MSE represents the average squared difference between the actual Income values and the predicted values.
        - While this metric gives an absolute measure of error, it is in squared units of the dependent variable (Income), making interpretation less intuitive. The high MSE value indicates room for improvement in the model.

Additional Metrics to Evaluate the Model:

1. **Adjusted R-squared**:
    - **Value**: 0.8291 (82.91%).
    - **Interpretation**: This value adjusts R-squared for the number of predictors in the model, providing a more reliable measure of model performance when comparing models with differing numbers of variables. The high adjusted R-squared value confirms that the predictors in the model are meaningful and not just overfitting.
2. **Root Mean Squared Error (RMSE)**:

RMSE is the square root of MSE, which puts the error back into the original units of the dependent variable (Income).

Interpretation: The RMSE of approximately 6,686 indicates that, on average, the predicted Income values deviate from the actual values by around $6,686.

Residual Analysis:

- Examining the residuals for patterns, outliers, or heteroscedasticity can provide additional insights into model performance. Ideally, residuals should be randomly distributed with no discernible pattern.

F-statistic:

- The F-statistic for the model is **161.1** with a p-value of **< 2.2e-16**, indicating that the overall model is statistically significant and at least one predictor variable significantly explains the variation in Income.

Short Discussion of Implications

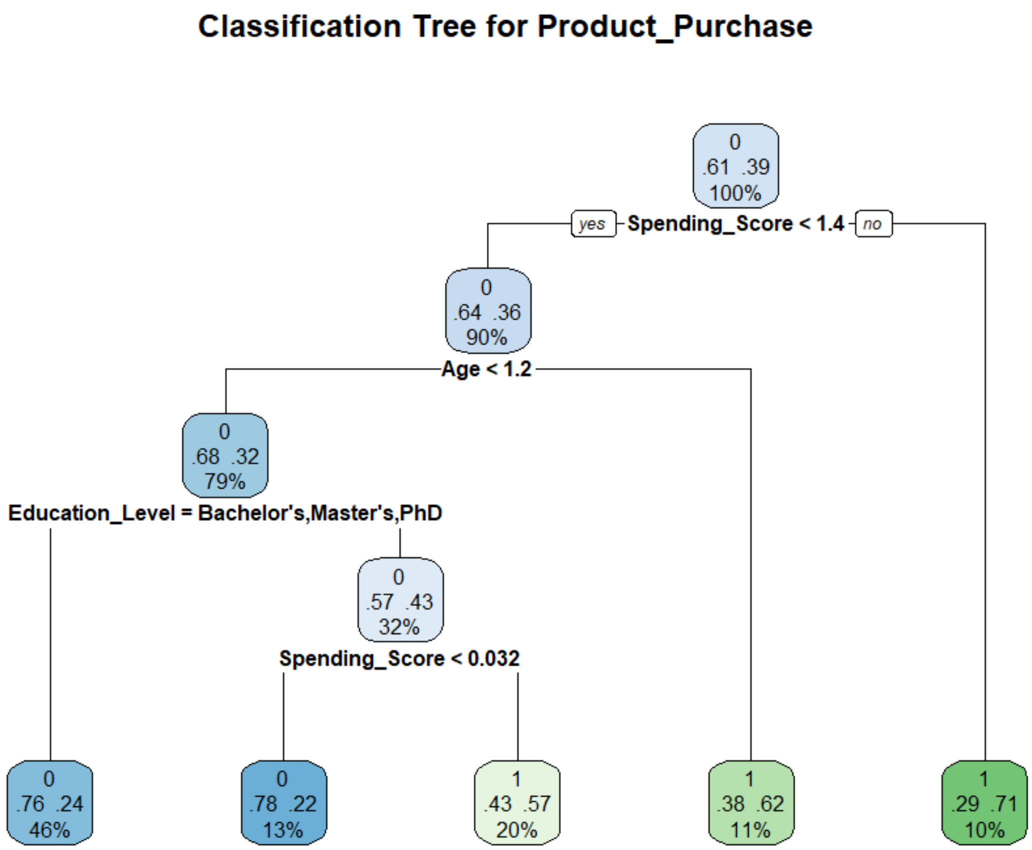
1. **R-squared (83.43%)**: The model explains a significant portion of income variability using Age, Spending_Score, and City_Type. This indicates the predictors are relevant, though care is needed to avoid overfitting.
2. **Significant Predictors**:
    - Age positively impacts income, aligning with career growth trends.
    - City_Type shows income disparities by location, suggesting urban residents earn more.
    - Spending_Score is not statistically significant, questioning its relevance.
3. **Residuals and MSE**: The **Residual Standard Error (6824)** and **Mean Squared Error (44,707,958)** reveal prediction accuracy but highlight potential outliers or missing predictors.
4. **Practical Implications**: Insights can guide targeted marketing strategies (e.g., age-based segmentation or location-specific campaigns) and policies addressing income inequalities.
5. **Model Limitations**: Non-linear effects or missing variables (e.g., education, job type) may require further exploration or alternative modeling techniques for better accuracy.

5. 20 points - Create a classification tree to predict Product_Purchase using all other variables in the dataset as predictors.
Visualize the tree and identify key decision splits (e.g., What is the top split? What variables are involved in significant splits?).
Summarize the tree's decision-making process and evaluate its performance using appropriate metrics.

Code:

```
# Load required libraries

library(rpart)

library(rpart.plot)



# Build the classification tree

tree_model <- rpart(Product_Purchase ~ ., data = train, method = "class")



# Plot the tree with better visualization

rpart.plot(tree_model, type = 2, extra = 104, fallen.leaves = TRUE, cex = 0.8,

        main = "Classification Tree for Product_Purchase")
```

Output:



View the summary of the tree

summary(tree_model)

Output:

Call:

rpart(formula = Product_Purchase ~ ., data = train, method = "class")

n= 71

	CP	nsplit	rel error	xerror	xstd
1	0.10714286	0	1.0000000	1.000000	0.1470707

```
2 0.07142857      1 0.8928571 1.357143 0.1500934
3 0.03571429      2 0.8214286 1.464286 0.1486501
4 0.01000000      4 0.7500000 1.500000 0.1479232

Variable importance
 Spending_Score           Age Education_Level        Income      City_Type
             49            23              16            10              2

Node number 1: 71 observations,    complexity param=0.1071429
  predicted class=0  expected loss=0.3943662  P(node) =1
    class counts:    43    28
   probabilities: 0.606 0.394
  left son=2 (64 obs) right son=3 (7 obs)
  Primary splits:
      Spending_Score  < 1.419495    to the left,  improve=1.5896000, (0 missing)
      Income          < -1.06658    to the left,  improve=1.4974140, (0 missing)
      Age             < 1.239161    to the left,  improve=0.9591438, (0 missing)
      Education_Level splits as  LRRL, improve=0.5671059, (0 missing)
      City_Type       < 1.5         to the left,  improve=0.4117336, (0 missing)

Node number 2: 64 observations,    complexity param=0.07142857
  predicted class=0  expected loss=0.359375  P(node) =0.9014085
    class counts:    41    23
   probabilities: 0.641 0.359
  left son=4 (56 obs) right son=5 (8 obs)
  Primary splits:
      Age             < 1.239161    to the left,  improve=1.2901790, (0 missing)
      Education_Level splits as  LRLL, improve=0.9140941, (0 missing)
```

```
      Income          < -1.108125   to the left,  improve=0.8375322, (0 missing)
      Spending_Score  < -1.483292   to the right, improve=0.7068452, (0 missing)
      City_Type       < 1.5         to the left,  improve=0.1944643, (0 missing)
  Surrogate splits:
      Income < 1.846729    to the left,  agree=0.906, adj=0.25, (0 split)

Node number 3: 7 observations
  predicted class=1  expected loss=0.2857143  P(node) =0.09859155
    class counts:     2     5
   probabilities: 0.286 0.714

Node number 4: 56 observations,    complexity param=0.03571429
  predicted class=0  expected loss=0.3214286  P(node) =0.7887324
    class counts:    38    18
   probabilities: 0.679 0.321
  left son=8 (33 obs) right son=9 (23 obs)
  Primary splits:
      Education_Level splits as  LRLL, improve=1.00301100, (0 missing)
      Spending_Score  < -1.403545   to the right, improve=1.00000000, (0 missing)
      Income          < 0.3881933   to the right, improve=0.80357140, (0 missing)
      Age             < -1.26788    to the right, improve=0.48513710, (0 missing)
      City_Type       < 1.5         to the left,  improve=0.04761905, (0 missing)
  Surrogate splits:
      Income         < -1.795928   to the right, agree=0.643, adj=0.130, (0 split)
      Age            < -1.669006   to the right, agree=0.625, adj=0.087, (0 split)
      Spending_Score < 0.1435444   to the left,  agree=0.607, adj=0.043, (0 split)

Node number 5: 8 observations
```

predicted class=1 expected loss=0.375 P(node) =0.1126761
class counts: 3   5
probabilities: 0.375 0.625

Node number 8: 33 observations
predicted class=0 expected loss=0.2424242 P(node) =0.4647887
class counts: 25   8
probabilities: 0.758 0.242

Node number 9: 23 observations,   complexity param=0.03571429
predicted class=0 expected loss=0.4347826 P(node) =0.3239437
class counts: 13   10
probabilities: 0.565 0.435
left son=18 (9 obs) right son=19 (14 obs)
Primary splits:
Spending_Score < 0.03189876 to the left, improve=1.3360940, (0 missing)
Age   < 0.002354029 to the right, improve=0.4472050, (0 missing)
Income   < -0.4913602 to the right, improve=0.4313320, (0 missing)
City_Type   < 2.5   to the right, improve=0.1043478, (0 missing)
Surrogate splits:
Income   < -1.628664   to the left, agree=0.652, adj=0.111, (0 split)
City_Type < 1.5   to the left, agree=0.652, adj=0.111, (0 split)

Node number 18: 9 observations
predicted class=0 expected loss=0.2222222 P(node) =0.1267606
class counts: 7   2
probabilities: 0.778 0.222

Node number 19: 14 observations

predicted class=1 expected loss=0.4285714 P(node) =0.1971831

class counts: 6   8

probabilities: 0.429 0.571

```
# Predict on test set
tree_pred <- predict(tree_model, newdata = test, type = "class")

# Confusion Matrix
conf_matrix_tree <- confusionMatrix(tree_pred, test$Product_Purchase)
print(conf_matrix_tree)
```

Output:

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 12  5
         1  6  6

               Accuracy : 0.6207
                 95% CI : (0.4226, 0.7931)
    No Information Rate : 0.6207
    P-Value [Acc > NIR] : 0.5815

                  Kappa : 0.2084

 Mcnemar's Test P-Value : 1.0000
```

Sensitivity : 0.6667

Specificity : 0.5455

Pos Pred Value : 0.7059

Neg Pred Value : 0.5000

Prevalence : 0.6207

Detection Rate : 0.4138

Detection Prevalence : 0.5862

Balanced Accuracy : 0.6061

'Positive' Class : 0

Key Decision Splits in the Tree

1. **Top Split**: The primary decision split is on the **Spending_Score**. If the value is less than 1.4, the classification is skewed towards no product purchase (0).
2. **Subsequent Splits**:
   - If **Spending_Score < 1.4**, the next split is on **Age**. For Age < 1.2, further splits depend on **Education_Level** or Spending_Score thresholds.
   - If **Spending_Score >= 1.4**, the probability of product purchase (1) increases significantly.

Summary of the Tree's Decision-Making Process

- **Spending_Score** is the most influential predictor, followed by **Age** and **Education_Level**.
- The tree systematically narrows predictions by applying conditions to maximize classification accuracy for each subset of data.
- The **leaf nodes** reveal the likelihood of purchase for specific groups.

Model Performance Metrics

- **Accuracy**: 62.07%—close to the no-information rate (NIR), indicating limited predictive improvement.
- **Balanced Accuracy**: 60.61%—suggests the model is slightly better than random at classifying both classes.

- **Sensitivity (66.67%)**: The model performs moderately in identifying non-purchasers (0).
- **Specificity (54.55%)**: It struggles more with identifying purchasers (1).
- **Kappa (0.2084)**: Reflects weak agreement beyond chance.

6. 5 points - Provide a summary of your overall workflow, including data preparation, model development, and key findings.
Highlight any actionable insights derived from the analysis or recommendations based on your results.

Summary of the Workflow

1. Data Preparation

- **Data Cleaning**: Verified the dataset for missing values, finding none.
- **Exploratory Analysis**:
    - Analyzed the distributions of key variables (e.g., Age, Income, Spending_Score).
    - Identified patterns and outliers through visualizations like histograms and box plots.
    - Noted trends in income distributions by city type, emphasizing disparities between urban, suburban, and rural areas.
- **Feature Engineering**: Prepared data for modeling by ensuring numerical and categorical variables were appropriately represented.

2. Model Development

1. **K-Nearest Neighbors (KNN)**:
    - Built a KNN model to predict product purchases.
    - Evaluated the model's performance with metrics like accuracy (58.62%) and balanced accuracy (54.29%).
    - Found moderate sensitivity (72.22%) but poor specificity (36.36%), indicating challenges in identifying purchasers.
2. **Linear Regression**:
    - Predicted income based on Age, Spending_Score, and City_Type.

    - Achieved an R-squared value of 0.834, indicating the model explained 83.4% of the variance in income.
    - Mean Squared Error (MSE) was 44,707,958, suggesting opportunities to improve predictive accuracy.
3. **Decision Tree**:
    - Built a classification tree for product purchases.
    - Key predictors included Spending_Score, Age, and Education_Level.
    - The model had a balanced accuracy of 60.61%, with an accuracy of 62.07% and limited agreement (Kappa = 0.2084).
    - Provided interpretable decision rules for purchase predictions.

3. Key Findings

- **Income Trends**: Urban residents tend to have higher incomes than suburban and rural residents. Spending behavior may align with these disparities.
- **Product Purchase Patterns**: Spending_Score, Age, and Education_Level were the most influential factors affecting purchase likelihood.
- **Model Performance**: Models performed moderately well but indicated room for improvement, particularly in predictive precision for purchasers.

Actionable Insights and Recommendations

1. **Targeted Marketing**:
    - Focus on urban and higher-income groups with higher spending scores for promotional campaigns.
    - Tailor messaging for younger, educated demographics as they show a higher likelihood of purchase.
2. **Improving Predictive Models**:
    - Explore ensemble methods like random forests or boosting to enhance predictive performance.
    - Address class imbalance issues using techniques like oversampling, undersampling, or cost-sensitive modeling.
3. **Further Data Exploration**:

- Investigate additional variables that might impact income or purchasing behavior (e.g., lifestyle preferences, family size).
- Conduct segmentation analysis to refine target customer profiles.

4. **Business Strategy**:
    - Design loyalty programs or personalized offers for high-spending customers.
    - Leverage insights from Spending_Score and Education_Level to develop customer retention strategies.

By applying these recommendations, businesses can optimize customer targeting and improve overall revenue generation.

CITATION:

https://chatgpt.com/share/675af077-1ebc-8002-993f-4ac36010d62c

American Statistical Association. (2019). *Guide to exploratory data analysis*. Retrieved from https://www.amstat.org/

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. doi:10.1023/A:1010933404324

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). New York: Springer.

Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling*. New York: Springer.