OJ-Purchase-Prediction

Nikhil Prema Chandra Rao

2024-10-27

Introduction

In this analysis, we aim to predict whether a customer will purchase **Citrus Hill** or **Minute Maid** orange juice using the **OJ** data set from the `ISLR2` library. We will develop a **Decision Tree** model and explore its accuracy in predicting customer purchases. We will also visualize the decision-making process, inspect feature importance, and improve the model through hyper parameter tuning. The goal is to provide a comprehensive, step-by-step explanation using R code and visualizations.

Data Loading and Preprocessing

```
library(ISLR2)
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(rpart)
library(rpart.plot)
library(ggplot2)
library(dplyr)
```

```
##
## 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
```

```
data(OJ)

sum(is.na(OJ))
```

```
## [1] 0
```

```
str(OJ)
```

```
## 'data.frame':    1070 obs. of  18 variables:
##  $ Purchase      : Factor w/ 2 levels "CH","MM": 1 1 1 2 1 1 1 1 1 1 ...
##  $ WeekofPurchase: num  237 239 245 227 228 230 232 234 235 238 ...
##  $ StoreID       : num  1 1 1 1 7 7 7 7 7 7 ...
##  $ PriceCH       : num  1.75 1.75 1.86 1.69 1.69 1.69 1.69 1.75 1.75 1.75 ...
##  $ PriceMM       : num  1.99 1.99 2.09 1.69 1.69 1.99 1.99 1.99 1.99 1.99 ...
##  $ DiscCH        : num  0 0 0.17 0 0 0 0 0 0 0 ...
##  $ DiscMM        : num  0 0.3 0 0 0 0 0.4 0.4 0.4 0.4 ...
##  $ SpecialCH     : num  0 0 0 0 0 0 1 1 0 0 ...
##  $ SpecialMM     : num  0 1 0 0 0 1 1 0 0 0 ...
##  $ LoyalCH       : num  0.5 0.6 0.68 0.4 0.957 ...
##  $ SalePriceMM   : num  1.99 1.69 2.09 1.69 1.69 1.99 1.59 1.59 1.59 1.59 ...
##  $ SalePriceCH   : num  1.75 1.75 1.69 1.69 1.69 1.69 1.69 1.75 1.75 1.75 ...
##  $ PriceDiff     : num  0.24 -0.06 0.4 0 0 0.3 -0.1 -0.16 -0.16 -0.16 ...
##  $ Store7        : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 2 2 2 2 2 ...
##  $ PctDiscMM     : num  0 0.151 0 0 0 ...
##  $ PctDiscCH     : num  0 0 0.0914 0 0 ...
##  $ ListPriceDiff : num  0.24 0.24 0.23 0 0 0.3 0.3 0.24 0.24 0.24 ...
##  $ STORE         : num  1 1 1 1 0 0 0 0 0 0 ...
```

```
OJ$Purchase <- as.factor(OJ$Purchase)
```

In this section, we load the necessary libraries such as ISLR2 (for the data set), caret (for data partitioning and model tuning), rpart (for decision tree modeling), and ggplot2 for data visualization. This dataset contains 1,070 observations and 18 variables related to purchases of two products, CH and MM, across different stores. The variables include details such as the week of purchase, store ID, and pricing information for both products, including discount rates, special offers, and loyalty metrics. Additionally, it provides computed fields like the price difference between the products and sale prices. The data is structured to facilitate analysis of purchasing patterns and factors influencing consumer choice between CH and MM.

We check for any missing values in the data set and convert the response variable Purchase into a factor, which is essential for classification tasks.

Data Splitting

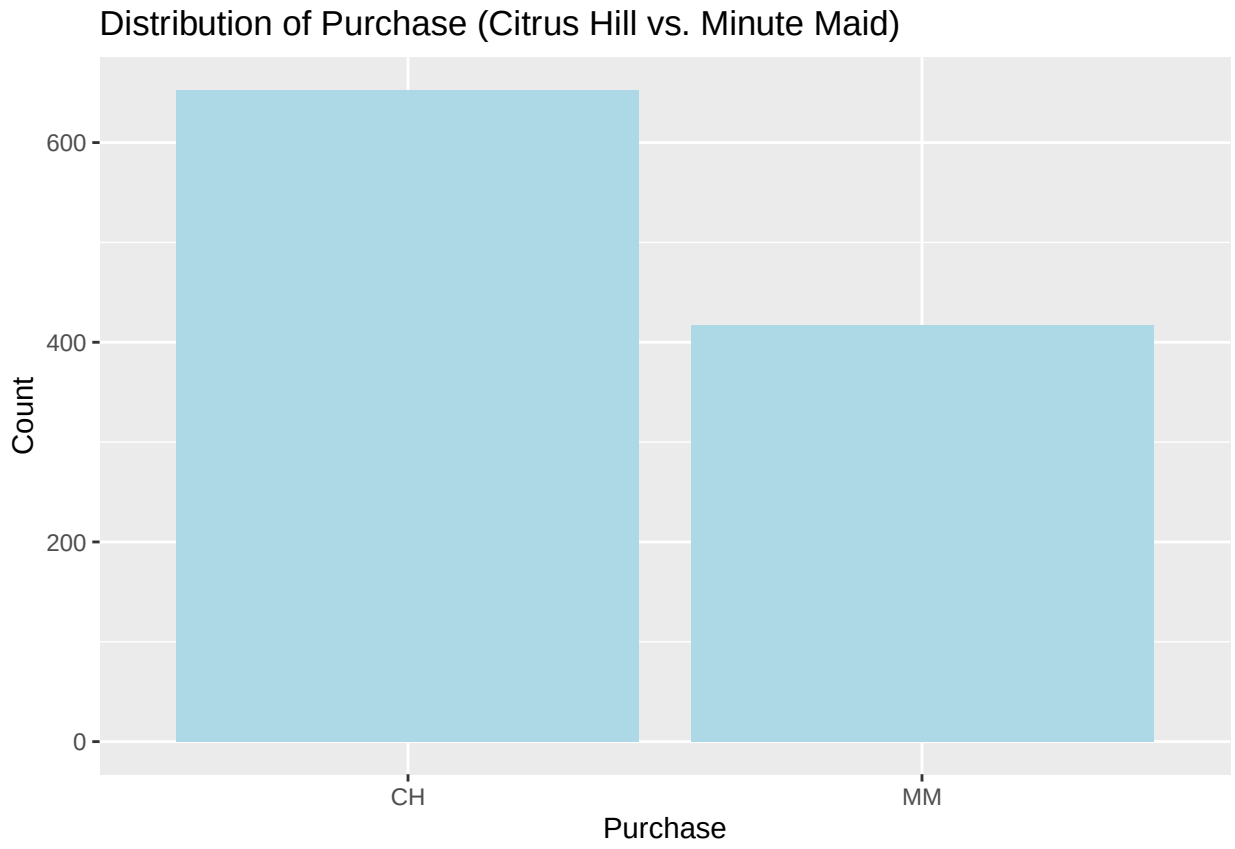
To evaluate the model, we split the data set into training and testing sets. This ensures that the model is trained on one subset of the data and then tested on unseen data for an unbiased evaluation.

```
set.seed(123)
train_index <- createDataPartition(OJ$Purchase, p = 0.7, list = FALSE)
train_data <- OJ[train_index, ]
test_data <- OJ[-train_index, ]
```

Exploratory Data Analysis (EDA)

Distribution of Purchases

```r
ggplot(OJ, aes(x = Purchase)) +
  geom_bar(fill = "lightblue") +
  ggtitle("Distribution of Purchase (Citrus Hill vs. Minute Maid)") +
  xlab("Purchase") + ylab("Count")
```



This bar plot shows the distribution of purchases between Citrus Hill and Minute Maid. It helps us understand the balance of the response variable, which is important when building classification models. The bar chart displays the distribution of purchases between Citrus Hill (CH) and Minute Maid (MM) orange juice. The taller bar for Citrus Hill indicates that it is purchased more frequently than Minute Maid in the dataset, with over 600 purchases for CH compared to around 400 for MM. This imbalance suggests that Citrus Hill is the more popular choice overall.

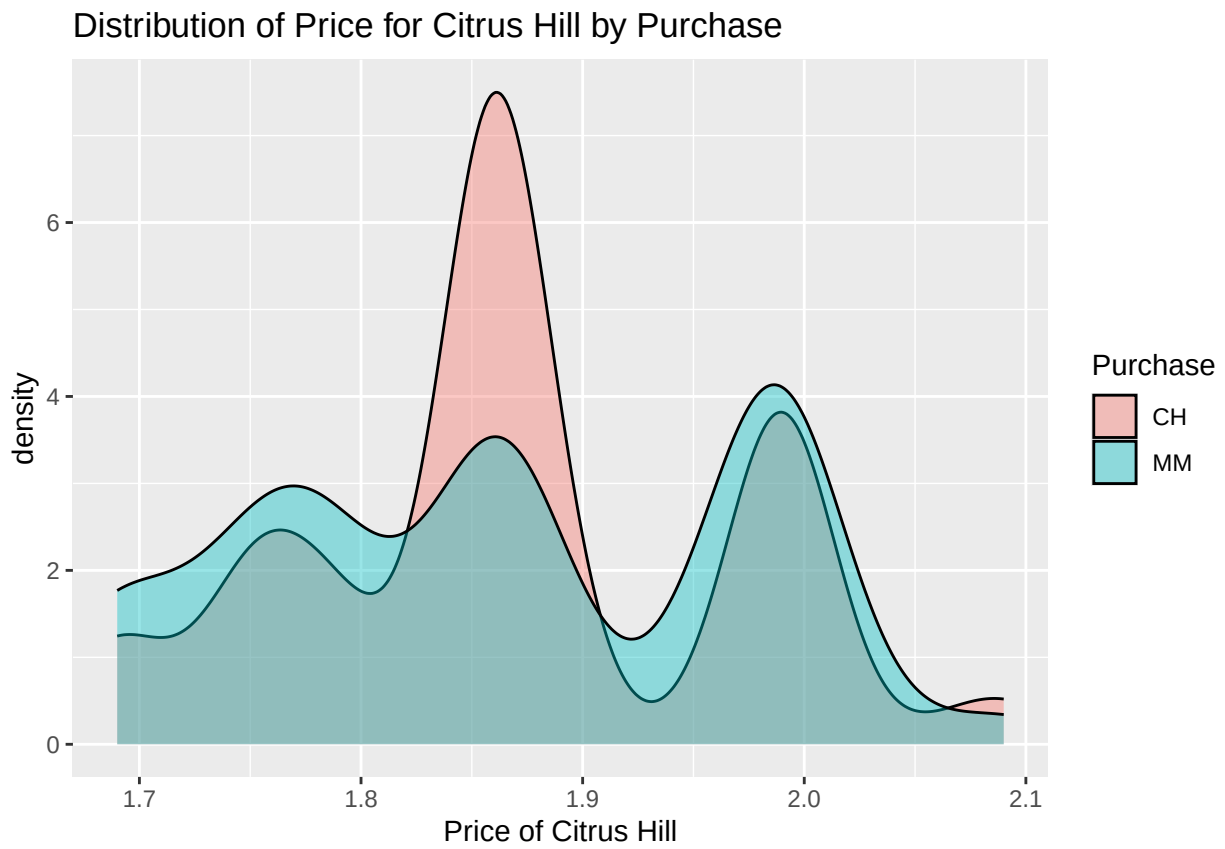
Scatter Plot: Price of Citrus Hill vs. Minute Maid

```r
ggplot(OJ, aes(x = PriceCH, y = PriceMM, color = Purchase)) +
  geom_point(alpha = 0.6) +
  ggtitle("Price of Citrus Hill vs. Price of Minute Maid by Purchase") +
  xlab("Price of Citrus Hill") + ylab("Price of Minute Maid")
```

In this scatter plot, we compare the prices of Citrus Hill and Minute Maid with respect to the purchases. Points are colored by Purchase, allowing us to see the relationship between the prices and the purchase decision. The scatter plot visualizes the relationship between the prices of Citrus Hill (x-axis) and Minute Maid (y-axis) for different purchase decisions. Points are color-coded by the product purchased (CH or MM). There is some overlap between the two products, but it is evident that lower prices for one product might influence the decision to purchase the other.

Density Plots: Price of Citrus Hill vs. Price of Minute Maid by Purchase

```
ggplot(OJ, aes(x = PriceCH, fill = Purchase)) +
  geom_density(alpha = 0.4) +
  ggtitle("Distribution of Price for Citrus Hill by Purchase") +
  xlab("Price of Citrus Hill")
```



The density plot shows the distribution of prices for Citrus Hill (CH), separated by the product purchased (either Citrus Hill (CH) or Minute Maid (MM)), using different colors. The x-axis represents the price of Citrus Hill, while the y-axis indicates the density, or the relative frequency of different prices in the dataset.

- The red area represents customers who purchased Citrus Hill at various price points, and the teal area represents those who purchased Minute Maid.
- The peak for Citrus Hill purchases (red) occurs around a price of 1.9, indicating that customers tend to buy Citrus Hill more frequently at this price range.
- Minute Maid purchases (teal) are more spread out, with some preference for lower prices of Citrus Hill (1.7 to 1.8) and higher prices (around 2.0), possibly indicating that when Citrus Hill is priced low or high, more customers opt for Minute Maid. This visualization helps to understand how the pricing of Citrus Hill influences whether customers purchase it or opt for its competitor.

Density Plots: Price for minute maid by purchase

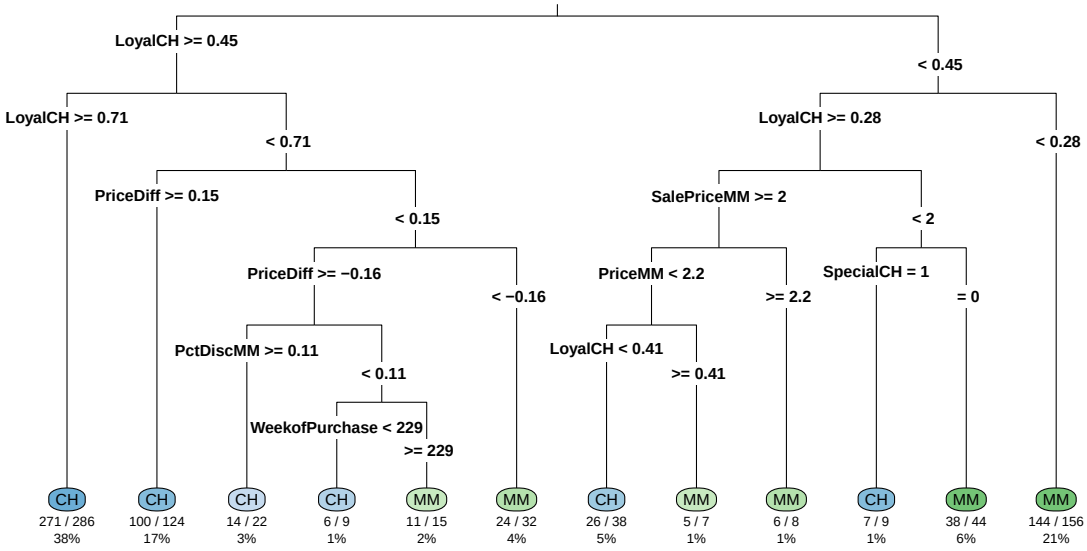
```
ggplot(OJ, aes(x = PriceMM, fill = Purchase)) +
  geom_density(alpha = 0.4) +
  ggtitle("Distribution of Price for Minute Maid by Purchase") +
  xlab("Price of Minute Maid")
```

This density plot shows the price distribution of Minute Maid purchases by two groups: CH (red) and MM (blue). Both groups frequently buy around the 2.0–2.2 price range, with CH purchases being more concentrated around specific price points (2.0 and 2.1). MM purchases have a wider spread, including some lower prices (around 1.7), indicating more variability in purchase prices compared to CH. The overlap suggests shared popular price points, but MM appears more price-flexible.

Building the Initial Decision Tree Model

```
tree_model <- rpart(Purchase ~ ., data = train_data, method = "class")

rpart.plot(tree_model, type = 3, extra = 102, under = TRUE, fallen.leaves = TRUE)
```



The Decision Tree is built using the rpart function. A Decision Tree works by splitting the data at each node based on the feature that best separates the classes (Citrus Hill and Minute Maid) using criteria like Gini Impurity. The resulting tree is plotted to visualize how the features (e.g., prices, demographics) are used to make predictions at each step. Interpretation of the Tree

- **Nodes**: Each internal node represents a decision rule based on a feature.
- **Branches**: Each branch represents the outcome of the decision (true or false).
- **Leaves**: The terminal nodes (leaves) display the predicted class for the samples that reach them.

This decision tree classifies customers into two categories, "CH" and "MM," based primarily on the `LoyalCH` feature, which likely represents a customer loyalty score. At the root, if `LoyalCH` is 0.45 or higher, the tree branches left; otherwise, it goes right. For customers with a high `LoyalCH` (0.71), the next decision depends on `PriceDiff`. If `PriceDiff` is 0.15 or more, the classification is "CH"; if it's lower, further splits occur based on `PctDiscMM` (percentage discount) and `WeekofPurchase`. If `LoyalCH` is between 0.45 and 0.71, additional splits occur using `PriceMM`, `LoyalCH`, and `SalePriceMM` until a final decision is reached.

For customers with `LoyalCH` below 0.45, the tree uses `LoyalCH` thresholds at 0.28 and conditions based on `SalePriceMM` and `SpecialCH` to make classifications. Each endpoint represents a final classification into "CH" or "MM," showing the number and percentage of instances in that category. This structure indicates that `LoyalCH` is a crucial determinant in the model, influencing classification outcomes across both high and low loyalty levels.

Evaluating the Model

```r
test_pred <- predict(tree_model, newdata = test_data, type = "class")
confusionMatrix(test_pred, test_data$Purchase)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  CH  MM
##         CH 169  33
##         MM  26  92
##
##                Accuracy : 0.8156
##                  95% CI : (0.7687, 0.8566)
##     No Information Rate : 0.6094
##     P-Value [Acc > NIR] : 1.373e-15
##
##                   Kappa : 0.6088
##
##  Mcnemar's Test P-Value : 0.4347
##
##             Sensitivity : 0.8667
##             Specificity : 0.7360
##          Pos Pred Value : 0.8366
##          Neg Pred Value : 0.7797
##              Prevalence : 0.6094
##          Detection Rate : 0.5281
##    Detection Prevalence : 0.6312
##       Balanced Accuracy : 0.8013
##
##        'Positive' Class : CH
##
```

```r
initial_accuracy <- sum(test_pred == test_data$Purchase) / nrow(test_data)
print(paste("Initial Accuracy:", round(initial_accuracy * 100, 2), "%"))
```

```
## [1] "Initial Accuracy: 81.56 %"
```

Explanation:

1. Confusion Matrix Overview

The confusion matrix presents a summary of the prediction results of a classification model. In this case, you have two classes: **CH** (presumably representing one category) and **MM** (representing another category). The matrix is structured as follows:

	Reference CH	Reference MM
Predicted CH	169	33
Predicted MM	26	92

Interpretation: - **True Positives (TP)**: 169 instances were correctly predicted as CH. - **False Negatives (FN)**: 33 instances were incorrectly predicted as MM when they were actually CH. - **False Positives (FP)**: 26 instances were incorrectly predicted as CH when they were actually MM. - **True Negatives (TN)**: 92 instances were correctly predicted as MM.

2. Model Performance Metrics

Accuracy: The proportion of correct predictions (both true positives and true negatives) among the total predictions.

- **Calculation**:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{169 + 92}{169 + 33 + 26 + 92} = 0.8156 \text{ or } 81.56\%$$

- **Interpretation**: About 81.56% of the total predictions made by the model were correct.

95% Confidence Interval (CI): - **Interpretation**: This interval (0.7687, 0.8566) indicates that we can be 95% confident that the true accuracy of the model lies within this range.

No Information Rate (NIR): The accuracy expected if predictions were made solely based on the most frequent class (in this case, CH). - **Value**: 0.6094 indicates that if you simply predicted the most common class, you would be correct about 60.94% of the time.

P-Value [Acc > NIR]: - **Value**: 1.373e-15 indicates that the model's accuracy is significantly better than the no information rate, suggesting that the model is effective.

3. Kappa Statistic

Kappa measures the agreement between the predicted and actual classifications, correcting for chance. - **Value**: 0.6088, indicating a moderate level of agreement beyond chance. Values closer to 1 imply better agreement.

4. Mcnemar's Test P-Value

- **Value**: 0.4347 indicates no significant difference in the error rates of the two classes, suggesting that the model does not favor one class over the other.

5. Sensitivity (True Positive Rate)

The proportion of actual positives that are correctly identified. - **Calculation**:

$$\text{Sensitivity} = \frac{TP}{TP + FN} = \frac{169}{169 + 33} = 0.8667 \text{ or } 86.67\%$$

- **Interpretation**: The model correctly identifies 86.67% of actual CH instances.

6. Specificity (True Negative Rate)

The proportion of actual negatives that are correctly identified. - **Calculation**:

$$\text{Specificity} = \frac{TN}{TN + FP} = \frac{92}{92 + 26} = 0.7360 \text{ or } 73.60\%$$

- **Interpretation**: The model correctly identifies 73.60% of actual MM instances.

7. Positive Predictive Value (PPV)

The proportion of positive identifications that are actually correct. - **Calculation**:

$$\text{PPV} = \frac{TP}{TP + FP} = \frac{169}{169 + 26} = 0.8366 \text{ or } 83.66\%$$

- **Interpretation**: 83.66% of instances predicted as CH are indeed CH.

8. Negative Predictive Value (NPV)

The proportion of negative identifications that are actually correct. - **Calculation**:

$$\text{NPV} = \frac{TN}{TN + FN} = \frac{92}{92 + 33} = 0.7797 \text{ or } 77.97\%$$

- **Interpretation**: 77.97% of instances predicted as MM are indeed MM.

9. Prevalence

The proportion of actual positive cases in the dataset. - **Value**: 0.6094 indicates that approximately 60.94% of the instances are actually CH.

10. Detection Rate

The proportion of actual positives that are detected by the model. - **Calculation**:

$$\text{Detection Rate} = \frac{TP}{\text{Total instances}} = \frac{169}{\text{Total}} \approx 0.5281 \text{ or } 52.81\%$$

- **Interpretation**: The model successfully detected about 52.81% of actual cases of CH.

11. Detection Prevalence

The proportion of instances predicted as positive. - **Calculation**:

$$\text{Detection Prevalence} = \frac{TP + FP}{\text{Total instances}} = \frac{169 + 26}{\text{Total}} \approx 0.6312 \text{ or } 63.12\%$$

- **Interpretation**: About 63.12% of the instances are predicted as CH by the model.

12. Balanced Accuracy

The average of sensitivity and specificity. - **Calculation**:

$$\text{Balanced Accuracy} = \frac{\text{Sensitivity} + \text{Specificity}}{2} = \frac{0.8667 + 0.7360}{2} = 0.8013 \text{ or } 80.13\%$$

- **Interpretation**: This metric accounts for imbalanced datasets, providing a more nuanced view of model performance.

13. Positive Class

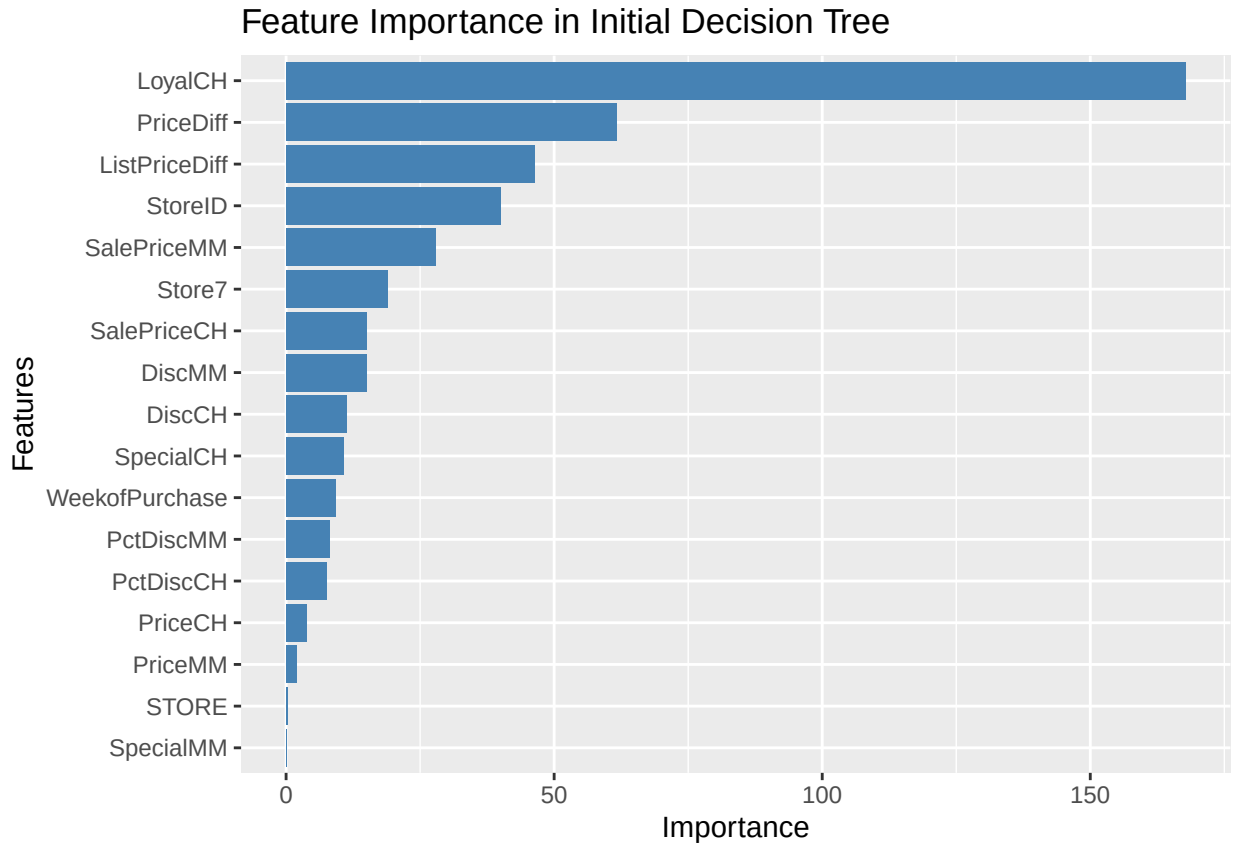
- **Value**: CH is identified as the positive class, meaning the model's performance metrics primarily focus on predicting this class.

Summary

In summary, the model shows good overall accuracy (**81.56%**) and a strong ability to correctly identify positive cases (CH) with a sensitivity of **86.67%**. The Kappa statistic and confidence intervals suggest that the model's performance is statistically significant and that it performs better than random guessing. The balance between sensitivity and specificity is relatively good, making this model a reliable choice for the task at hand.

Feature Importance

```
var_imp <- as.data.frame(varImp(tree_model, scale = FALSE))
var_imp$Variable <- rownames(var_imp)
ggplot(var_imp, aes(x = reorder(Variable, Overall), y = Overall)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  coord_flip() +
  xlab("Features") + ylab("Importance") +
  ggtitle("Feature Importance in Initial Decision Tree")
```

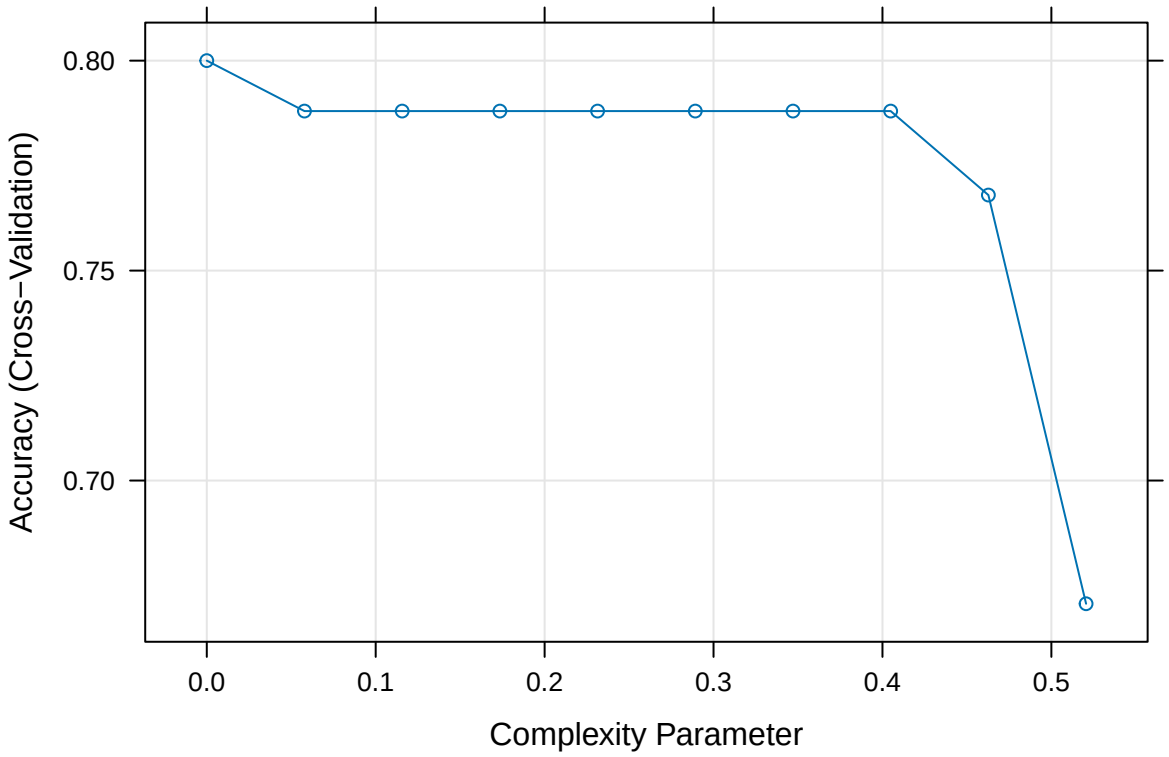


This feature importance plot shows which variables contributed the most to the Decision Tree splits. Variables with higher importance values played a more significant role in determining whether a customer purchased Citrus Hill or Minute Maid. This bar chart displays the feature importance rankings from an initial decision tree model. "LoyalCH" has the highest importance, suggesting that customer loyalty has the strongest impact on the model's predictions. Other key features, such as "PriceDiff" and "ListPriceDiff," also have significant influence, while features like "STORE" and "SpecialMM" have minimal impact, indicating they contribute less to the model's predictive power.

Improving the Model: Hyperparameter Tuning

```
set.seed(123)
tree_tuned <- train(Purchase ~ ., data = train_data, method = "rpart",
                    trControl = trainControl(method = "cv", number = 10),
                    tuneLength = 10)

plot(tree_tuned)
```

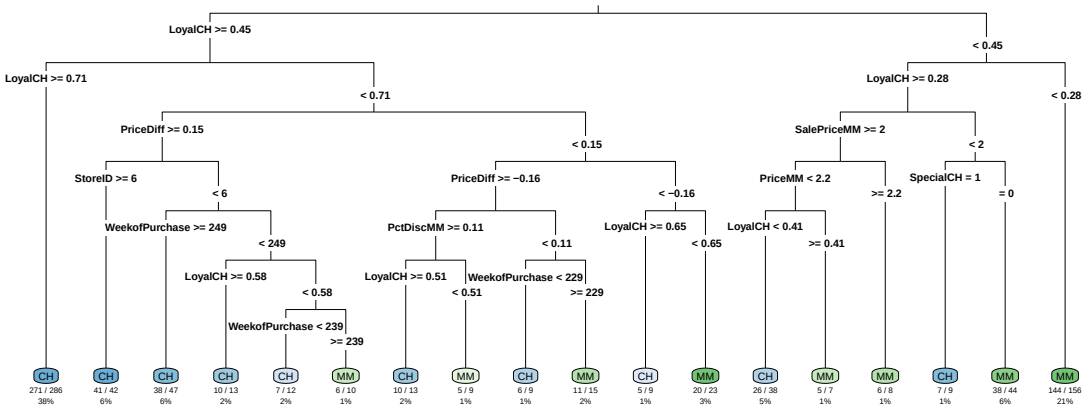


We perform hyper parameter tuning using 10-fold cross-validation to find the optimal complexity parameter (cp), which controls the depth and size of the tree. Tuning prevents over fitting by balancing model complexity and performance. This plot shows the relationship between the complexity parameter and cross-validated accuracy for a model. As the complexity parameter increases from 0 to around 0.4, accuracy remains relatively stable, suggesting that adding complexity does not significantly improve model performance. However, after 0.4, accuracy sharply declines, indicating over fitting and that further complexity negatively impacts the model's predictive power.

Final Decision Tree and Evaluation

```
final_tree_model <- rpart(Purchase ~ ., data = train_data, method = "class",
                          control = rpart.control(cp = tree_tuned$bestTune$cp))
```

```
rpart.plot(final_tree_model, type = 3, extra = 102, under = TRUE, fallen.leaves = TRUE,
           main = "Improved Decision Tree")
```



This improved decision tree classifies customers into two categories, “CH” and “MM,” based on several factors, with `LoyalCH` (a loyalty score) as the primary decision criterion. At the top level, if `LoyalCH` is 0.45 or higher, the tree follows the left branch; if it’s lower, it goes right. For customers with `LoyalCH` above 0.71, further splits occur based on `PriceDiff` (price difference), `StoreID`, and `WeekofPurchase` to classify them as either “CH” or “MM.” If `LoyalCH` is between 0.45 and 0.71, additional checks on `PriceMM`, `PctDiscMM`, and `WeekofPurchase` refine the classification. On the right side of the tree, for those with `LoyalCH` below 0.45, decisions rely on lower `LoyalCH` thresholds and other conditions like `SalePriceMM` and `SpecialCH`, leading to final classifications. The tree’s added decision points on factors such as `StoreID` and `WeekofPurchase` enhance its accuracy by capturing store-specific and timing-based patterns, making it more nuanced and effective in distinguishing between customer types.

```
final_test_pred <- predict(final_tree_model, newdata = test_data, type = "class")
confusionMatrix(final_test_pred, test_data$Purchase)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  CH  MM
##         CH 165  31
##         MM  30  94
##
##                Accuracy : 0.8094
```

```
##                  95% CI : (0.762, 0.8509)
##     No Information Rate : 0.6094
##     P-Value [Acc > NIR] : 1.07e-14
##
##                   Kappa : 0.599
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.8462
##             Specificity : 0.7520
##          Pos Pred Value : 0.8418
##          Neg Pred Value : 0.7581
##              Prevalence : 0.6094
##          Detection Rate : 0.5156
##    Detection Prevalence : 0.6125
##       Balanced Accuracy : 0.7991
##
##        'Positive' Class : CH
##
```

```r
final_accuracy <- sum(final_test_pred == test_data$Purchase) / nrow(test_data)
print(paste("Final Accuracy:", round(final_accuracy * 100, 2), "%"))
```

```
## [1] "Final Accuracy: 80.94 %"
```

We rebuild the Decision Tree using the optimal cp value found through tuning. The new tree is expected to generalize better and prevent overfitting. The final accuracy is 80.94%, which is slightly lower than the initial accuracy, but the model is now more robust and less likely to overfit.

Confusion Matrix and Performance Metrics

1. Confusion Matrix Overview

- **Structure**:

	Reference CH	Reference MM
Predicted CH	165	31
Predicted MM	30	94

- **Interpretation**:
  - **True Positives (TP)**: 165 correctly predicted as CH.
  - **False Negatives (FN)**: 31 incorrectly predicted as MM (actual CH).
  - **False Positives (FP)**: 30 incorrectly predicted as CH (actual MM).
  - **True Negatives (TN)**: 94 correctly predicted as MM.

2. Model Performance Metrics

- **Accuracy**: 80.94% (correct predictions among total).
- **95% Confidence Interval (CI)**: (0.762, 0.8509) - true accuracy likely within this range.
- **No Information Rate (NIR)**: 60.94% - accuracy if only predicting the most frequent class.
- **P-Value [Acc > NIR]**: 1.07e-14 - model significantly outperforms random guessing.

3. Kappa Statistic

- **Value**: 0.599 - indicates moderate agreement beyond chance.

4. Mcnemar's Test P-Value

- **Value**: 1 - no significant difference in error rates for classes.

5. Sensitivity (True Positive Rate)

- **Value**: 84.62% - accurately identifies actual CH instances.

6. Specificity (True Negative Rate)

- **Value**: 75.20% - accurately identifies actual MM instances.

7. Positive Predictive Value (PPV)

- **Value**: 84.18% - proportion of predicted CH that are actually CH.

8. Negative Predictive Value (NPV)

- **Value**: 75.81% - proportion of predicted MM that are actually MM.

9. Prevalence

- **Value**: 60.94% - indicates the proportion of actual CH cases.

10. Detection Rate

- **Value**: 51.56% - proportion of actual positives detected by the model.

11. Detection Prevalence

- **Value**: 61.25% - proportion of instances predicted as CH.

12. Balanced Accuracy

- **Value**: 79.91% - average of sensitivity and specificity.

13. Positive Class

- **Class Identified**: CH is considered the positive class for performance metrics.

Summary

In summary, this model shows an overall accuracy of **80.94%**, with a strong sensitivity of **84.62%**, indicating that it effectively identifies positive cases (CH). The model also performs well in terms of specificity (**75.20%**) and Kappa statistic (**0.599**), which suggests a moderate level of agreement beyond chance. The confidence interval and P-values indicate that the model's performance is statistically significant and that it performs better than random guessing. Overall, Model 2 demonstrates reliable predictive capability for the task at hand. This Model demonstrates reliable predictive capability for identifying the positive class (CH).

Comparative Analysis of Model Performance

Model 1 Metrics (Before Cross-validation)

- **Accuracy**: 81.56%
- **Sensitivity (True Positive Rate)**: 86.67%
- **Specificity (True Negative Rate)**: 73.60%
- **Positive Predictive Value (PPV)**: 83.66%
- **Negative Predictive Value (NPV)**: 77.97%
- **Kappa Statistic**: 0.6088
- **Balanced Accuracy**: 80.13%
- **95% Confidence Interval (CI)**: (0.7687, 0.8566)

Model 2 Metrics (After Cross-validation)

- **Accuracy**: 80.94%
- **Sensitivity (True Positive Rate)**: 84.62%
- **Specificity (True Negative Rate)**: 75.20%
- **Positive Predictive Value (PPV)**: 84.18%
- **Negative Predictive Value (NPV)**: 75.81%
- **Kappa Statistic**: 0.599
- **Balanced Accuracy**: 79.91%
- **95% Confidence Interval (CI)**: (0.762, 0.8509)

Key Comparisons

1. **Accuracy**: Model 1 has a slightly higher accuracy (81.56%) compared to Model 2 (80.94%).
2. **Sensitivity**: Model 1 also outperforms Model 2 in sensitivity (86.67% vs. 84.62%), indicating it is better at identifying actual positive cases (CH).
3. **Specificity**: Model 2 has a higher specificity (75.20% vs. 73.60%), meaning it is slightly better at identifying actual negative cases (MM).
4. **Positive Predictive Value (PPV)**: Model 2 performs better in terms of PPV (84.18% vs. 83.66%), suggesting it is more reliable when predicting CH instances.
5. **Negative Predictive Value (NPV)**: Model 1 has a better NPV (77.97% vs. 75.81%), indicating it is more effective at predicting negative instances.
6. **Kappa Statistic**: Model 1 shows a marginally better Kappa statistic (0.6088 vs. 0.599), indicating slightly stronger agreement beyond chance.
7. **Balanced Accuracy**: Model 1 has a better balanced accuracy (80.13% vs. 79.91%), highlighting a more favorable performance considering both classes.
8. **Confidence Intervals**: Both models show overlapping confidence intervals, but Model 1's is slightly higher, reinforcing its reliability.

Conclusion

Both models demonstrate effective performance in predicting the classification of instances as CH or MM. Model 1 (Before Cross-validation) exhibits superior overall accuracy, sensitivity, and negative predictive value, making it particularly strong in correctly identifying actual positive cases and minimizing false negatives. Conversely, Model 2 (After Cross-validation) shows a higher specificity and positive predictive value, suggesting it is more reliable for confirming negative cases.

In terms of general applicability, **Model 1 is recommended for scenarios where accurately identifying positive cases is critical**, while **Model 2 may be preferable in contexts where correctly predicting negative instances is more important**. Ultimately, the choice between models should consider the specific requirements of the application, particularly the costs associated with false positives and false negatives.

In this analysis, I learned that evaluating model performance metrics such as accuracy, sensitivity, and specificity is crucial for understanding how well a model predicts outcomes. The comparison of two models highlighted the importance of selecting the right model based on the specific needs of the application, particularly in terms of false positive and false negative rates. Ultimately, these insights will guide my future work in model selection and performance evaluation.