Data Mining Project 3

Olivia Hofmann, Michael Perkins

2024-11-24

Introduction

The objective of this project is to classify counties into risk levels (**high**, **medium**, or **low**) for future pandemics using COVID-19 data. This report follows the CRISP-DM framework, focusing on **Data Preparation**, **Modeling**, **Evaluation**, and **Deployment**. The results can help stakeholders prepare for and mitigate the impact of future pandemics.

---

1. Data Preparoation

```r
# Load required libraries
library(tidyverse)
library(lubridate)
library(caret)
library(rpart)  # Decision Tree
library(randomForest)  # Random Forest
library(e1071)  # SVM
library(DMwR2)
library(PRROC)
library(ggplot2)
library(gridExtra)
library(pROC)
library(xgboost)
library(class)
library(nnet)
library(iml)
```

Define Classes

- The classes are based on confirmed COVID-19 cases per 10,000 population per week:
  - **High Risk:** 50 cases.
  - **Medium Risk:** 10–49 cases.
  - **Low Risk:** < 10 cases.
- These thresholds were chosen based on observed patterns in case severity and the need to trigger timely interventions.

```r
# Load mobility data
final_merged_dataset <- read_csv("data/final_merged_dataset.csv")
```

Data Preparation Steps

1. **Merge and Clean Data**: Ensure a single dataset with a class attribute.

2. **Select Predictive Features**: Extract features with potential predictive power.
3. **Handle Missing Data**: Use imputation or remove incomplete rows for models that cannot handle missing data.

```
# Define risk levels
final_data <- final_merged_dataset %>%
  mutate(risk_level = case_when(
    count >= 50 ~ "high",
    count >= 10 ~ "medium",
    TRUE ~ "low"
  )) %>%
  mutate(risk_level = factor(risk_level, levels = c("low", "medium", "high")))

# Select relevant features
classification_data <- final_data %>%
  select(retail_and_recreation_percent_change_from_baseline,
         grocery_and_pharmacy_percent_change_from_baseline,
         workplaces_percent_change_from_baseline,
         PC1, PC2, week, risk_level) %>%
  drop_na()

# Split data into training and testing
set.seed(123)
training_index <- sample(1:nrow(classification_data), 0.7 * nrow(classification_data))
training_data <- classification_data[training_index, ]
testing_data <- classification_data[-training_index, ]

# Balance training data
min_class_size <- min(table(training_data$risk_level))
balanced_training_data <- training_data %>%
  group_by(risk_level) %>%
  sample_n(min_class_size) %>%
  ungroup()
```

---

2. Modeling

Model 1: Decision Tree

- **Advantages**: Simple and interpretable.

```
set.seed(123)
dt_model <- rpart(risk_level ~ ., data = balanced_training_data, method = "class",
                  control = rpart.control(cp = 0.05, maxdepth = 3))
```

Model 2: Random Forest

- **Advantages**: Handles large datasets and captures feature interactions.

```
set.seed(123)
rf_model <- randomForest(risk_level ~ ., data = balanced_training_data, ntree = 100, mtry = 2)
```

Model 3: Support Vector Machine (SVM)

- **Advantages**: Effective for high-dimensional spaces.

```r
set.seed(123)
subsample_index <- sample(1:nrow(balanced_training_data), 0.01 * nrow(balanced_training_data))
subsample_data <- balanced_training_data[subsample_index, ]

svm_model <- svm(risk_level ~ ., data = subsample_data, cost = 0.1, gamma = 0.01, kernel = 'linear')
```

Model 4: Gradient Boosting

- **Advantages**:

```r
# Prepare training features and labels
x_train <- model.matrix(risk_level ~ . - 1, data = balanced_training_data)  # Remove intercept and non-
y_train <- as.numeric(balanced_training_data$risk_level) - 1  # Convert factor to 0-indexed numeric

# Prepare testing features and labels
x_test <- model.matrix(risk_level ~ . - 1, data = testing_data)
y_test <- as.numeric(testing_data$risk_level) - 1

# Train the Gradient Boosting model
set.seed(123)
xgb_model <- xgboost(
  data = x_train,
  label = y_train,
  objective = "multi:softprob",  # Multiclass classification
  num_class = length(unique(balanced_training_data$risk_level)),  # Number of classes
  nrounds = 100,  # Number of boosting rounds
  eta = 0.1,      # Learning rate
  max_depth = 3,  # Tree depth
  verbose = 0     # Suppress training logs
)
```

Model 5: Logistic Regression

- **Advantages**:

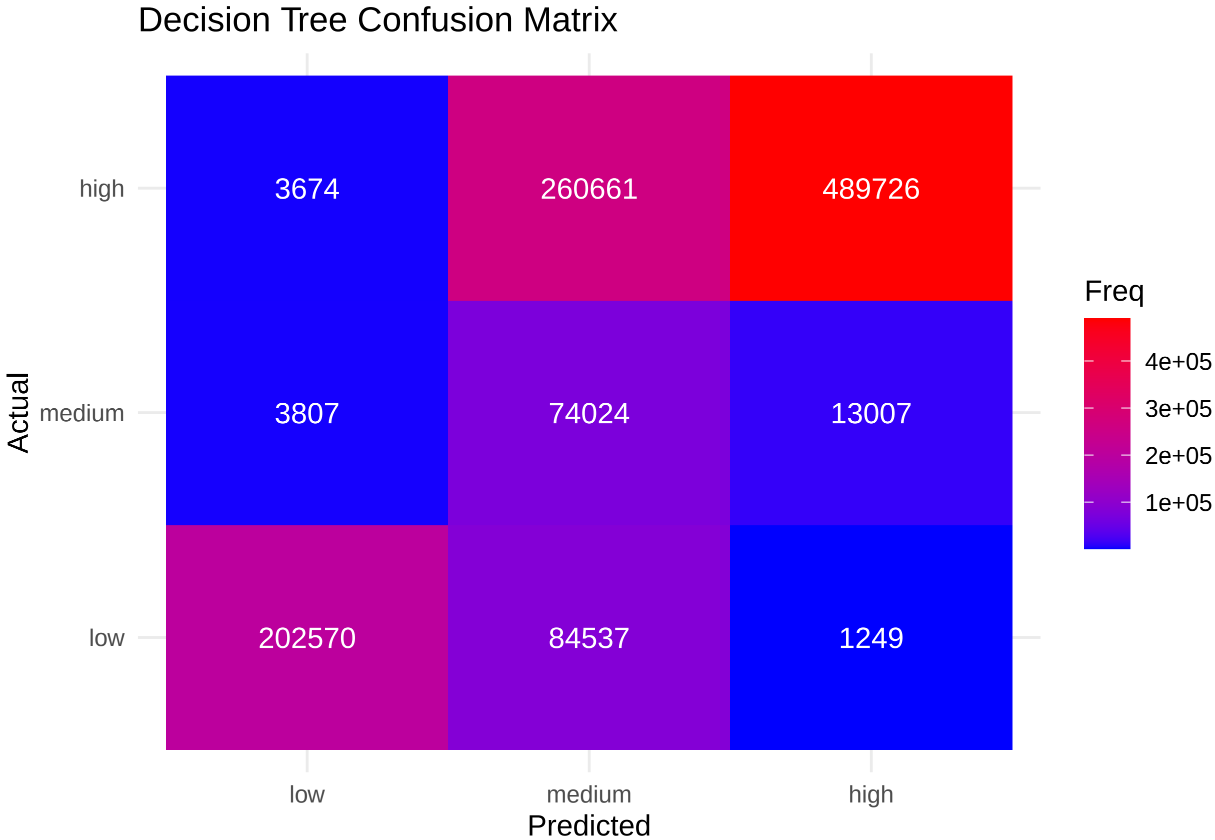
```r
# Train a multinomial logistic regression model
logistic_model <- multinom(risk_level ~ ., data = balanced_training_data)
```

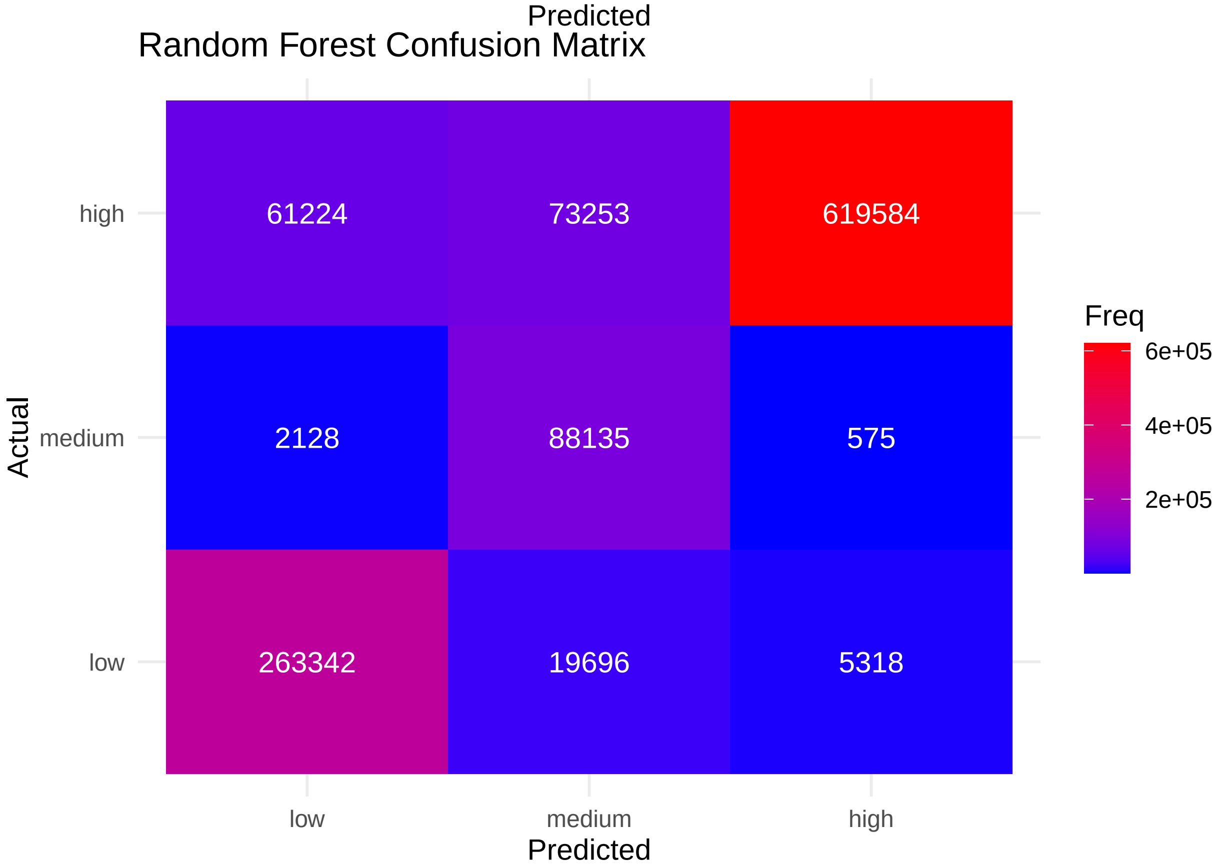
```
## # weights:  24 (14 variable)
## initial  value 700935.513809
## iter  10 value 696309.742437
## iter  20 value 399017.359757
## final  value 398574.755340
## converged
```

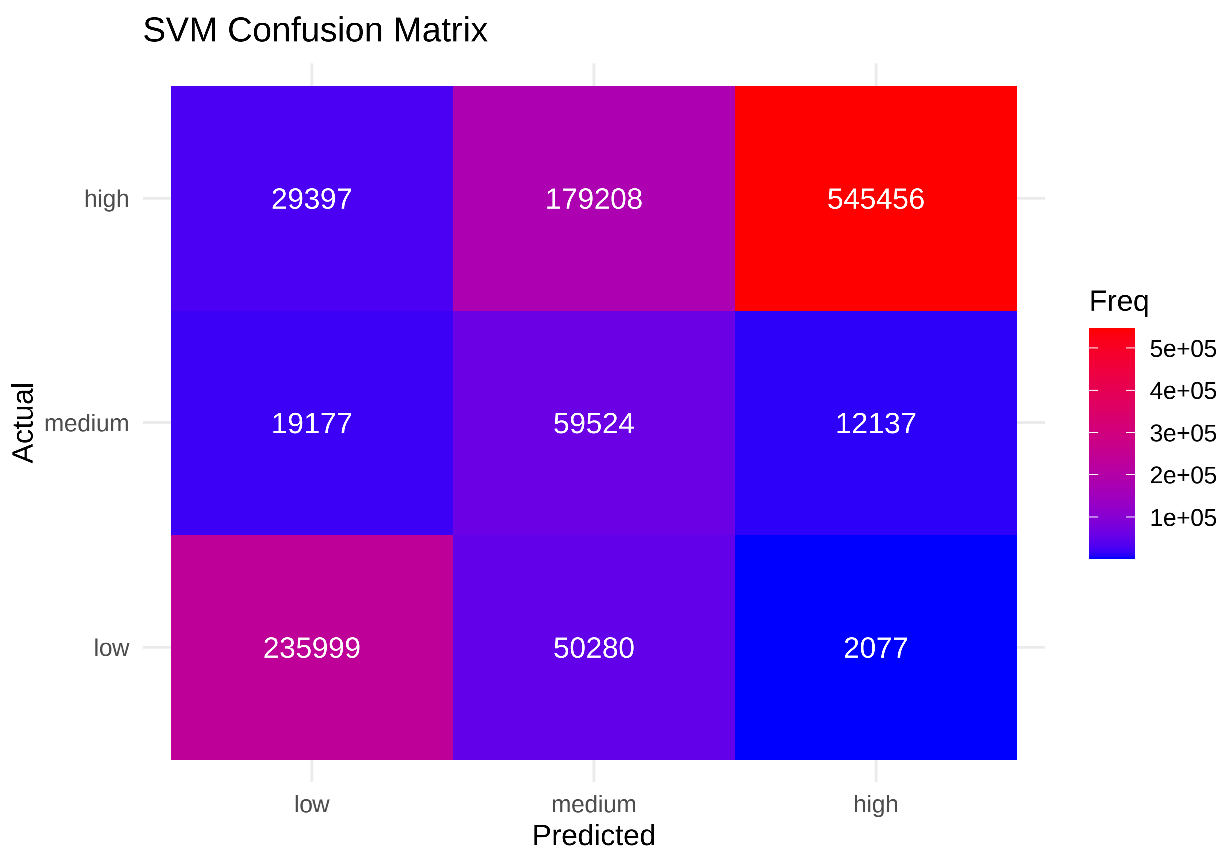
---

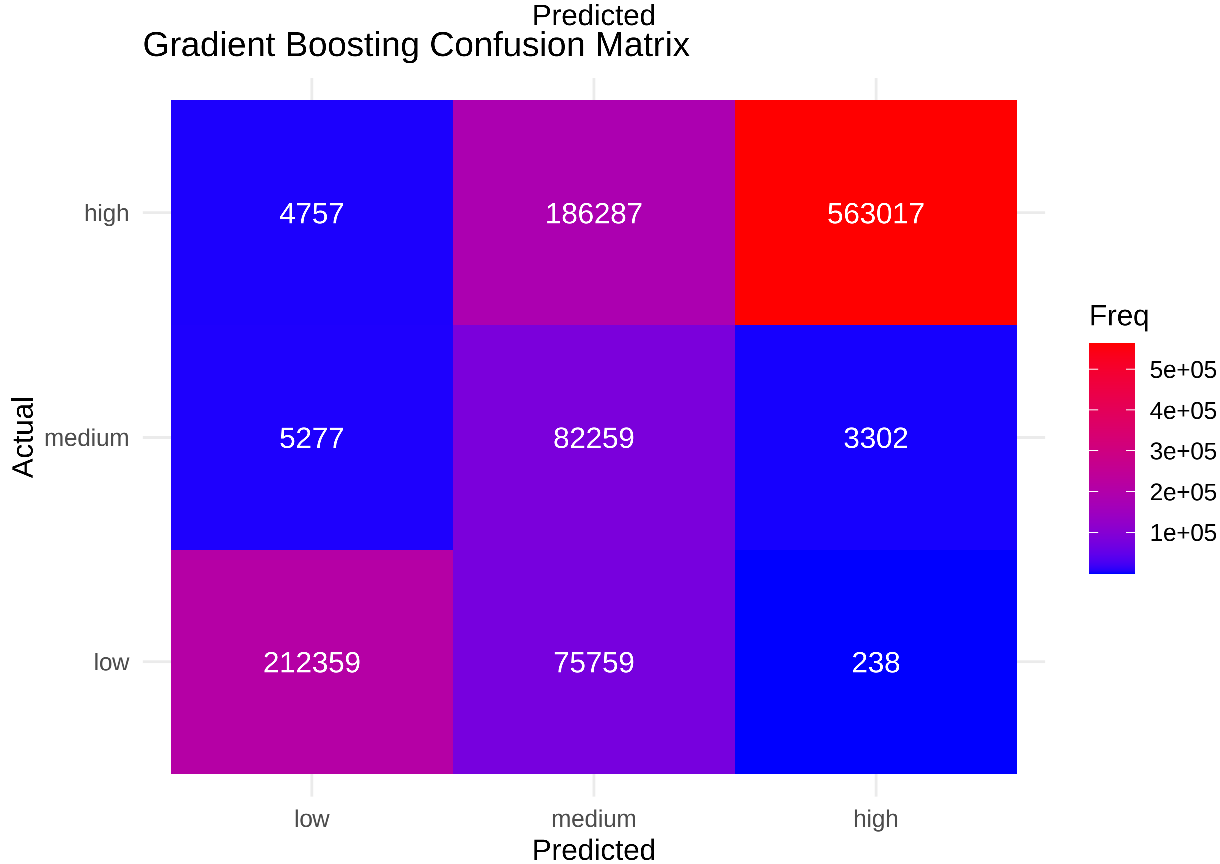
3. Evaluation

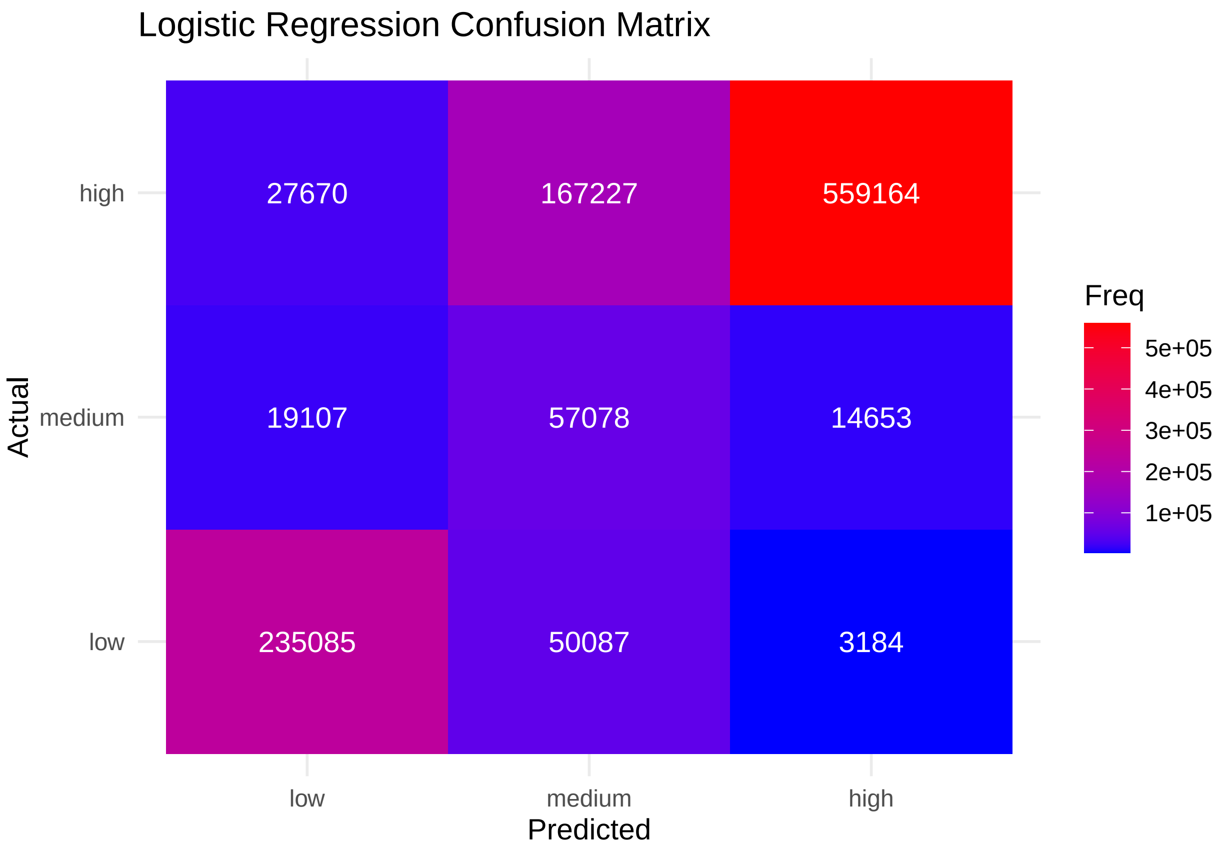
1. Confusion Matrix Heatmaps





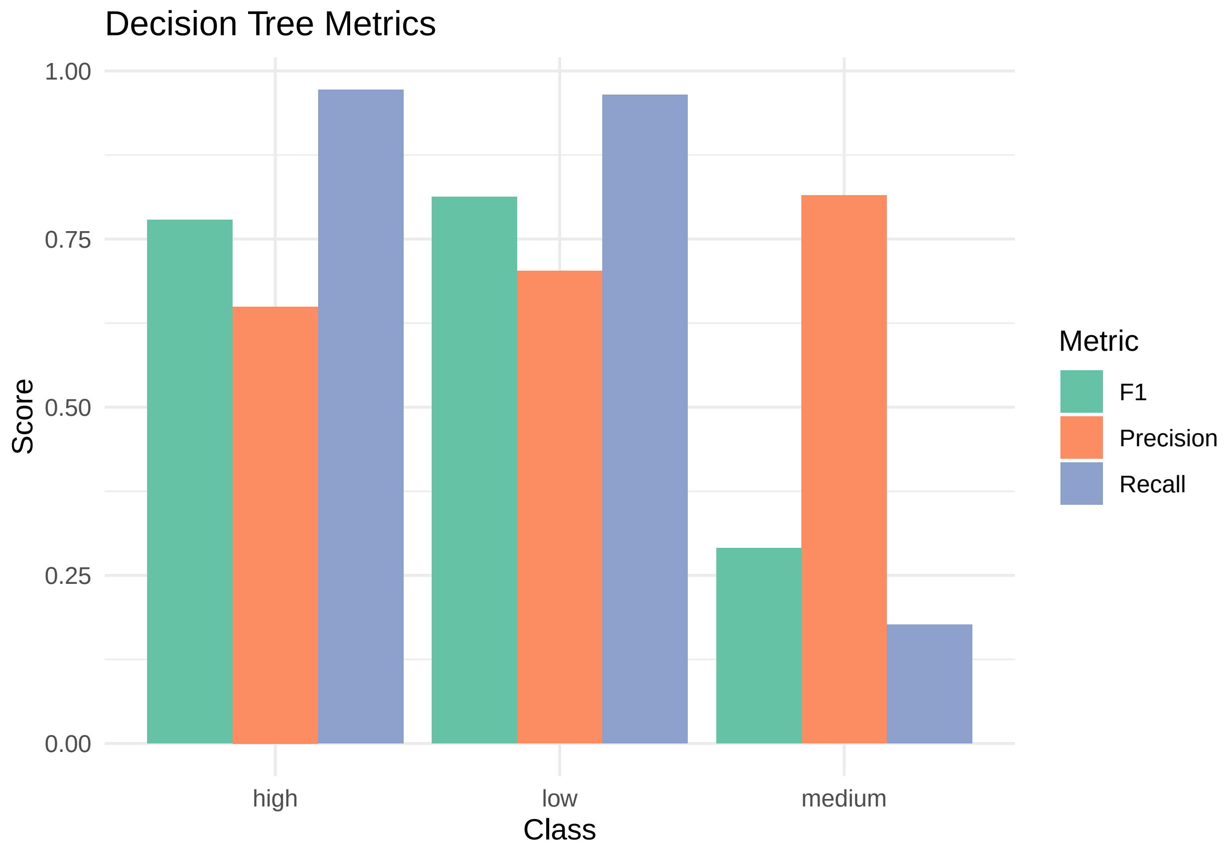

SVM Confusion Matrix
Actual
high
medium
low
29397
179208
545456
19177
59524
12137
235999
50280
2077
low
medium
high
Predicted
Freq
5e+05
4e+05
3e+05
2e+05
1e+05

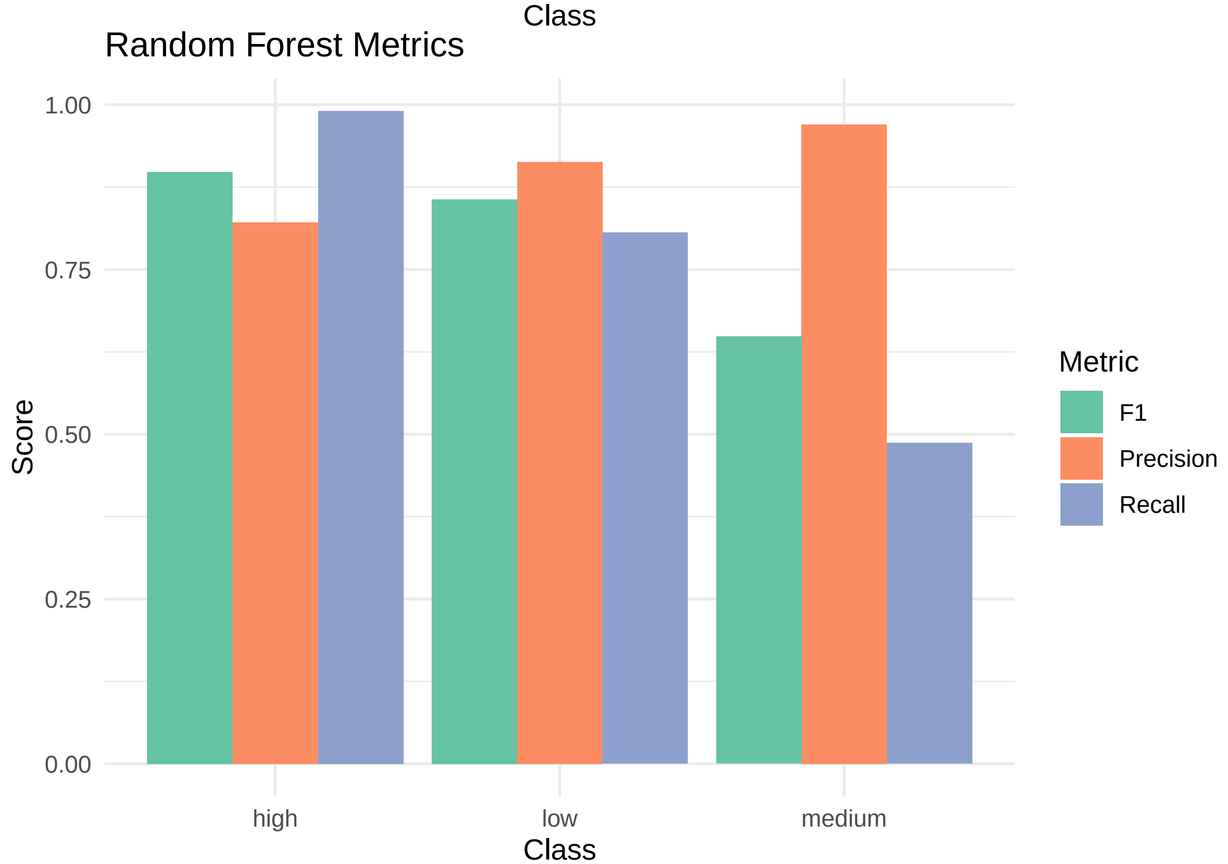

Gradient Boosting Confusion Matrix
Actual
high
medium
low
4757
186287
563017
5277
82259
3302
212359
75759
238
low
medium
high
Predicted
Freq
5e+05
4e+05
3e+05
2e+05
1e+05

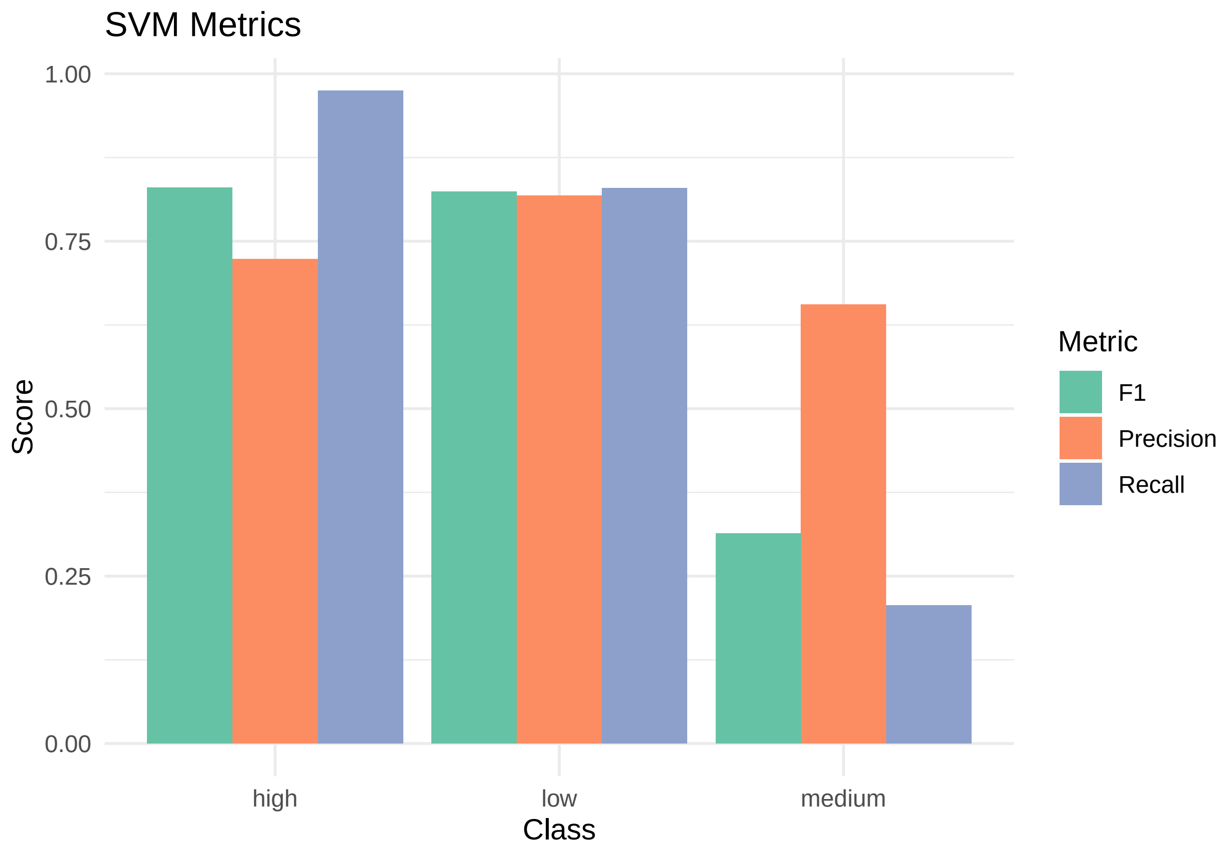


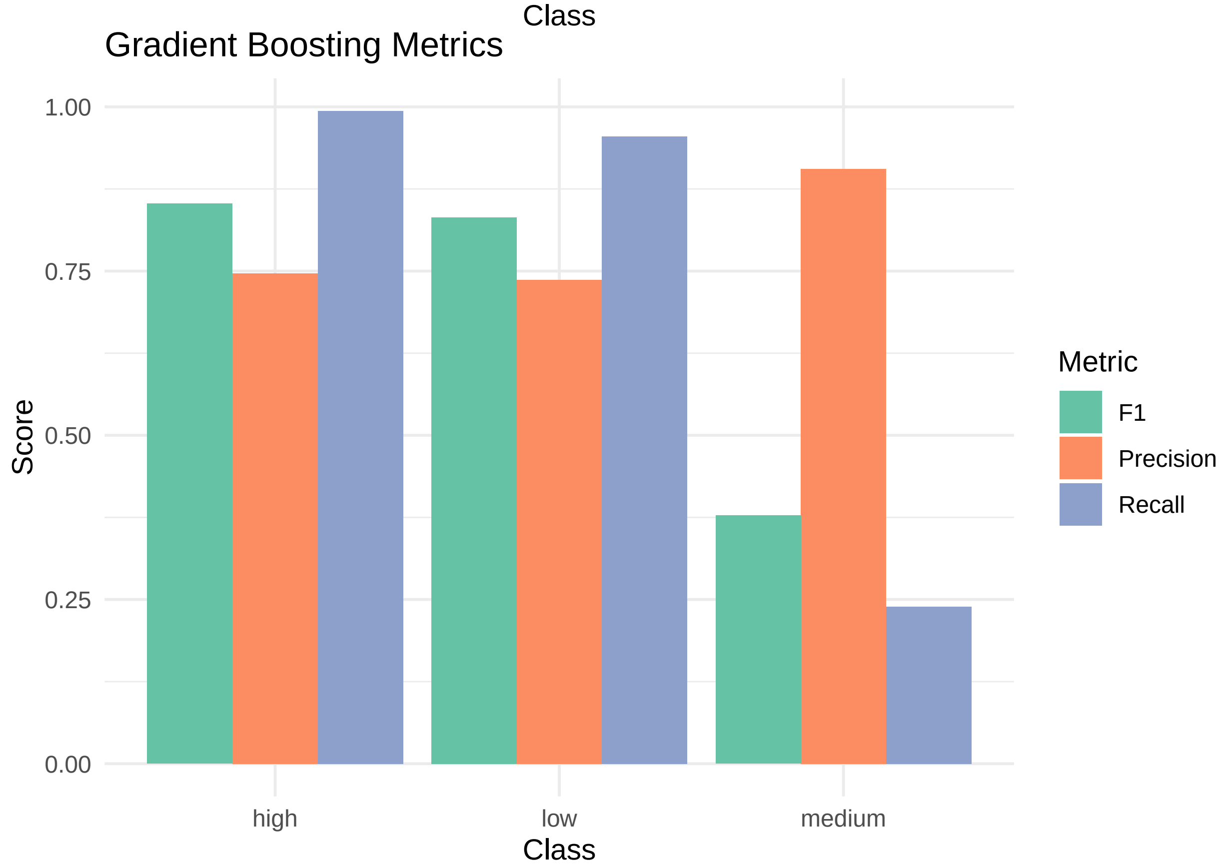
Logistic Regression Confusion Matrix
high
medium
low
Actual
27670
167227
559164
19107
57078
14653
235085
50087
3184
low
medium
high
Predicted
Freq
5e+05
4e+05
3e+05
2e+05
1e+05

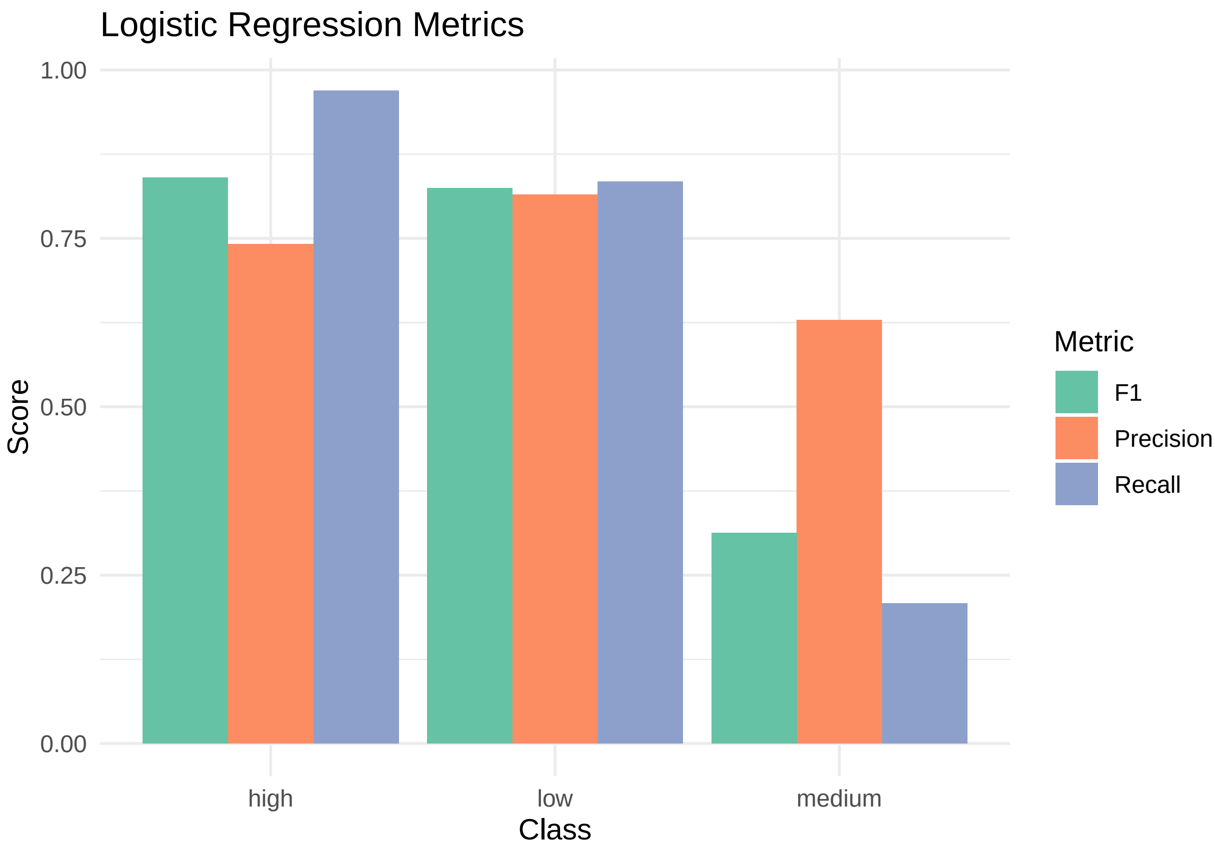
2. Precision, Recall and F1-score



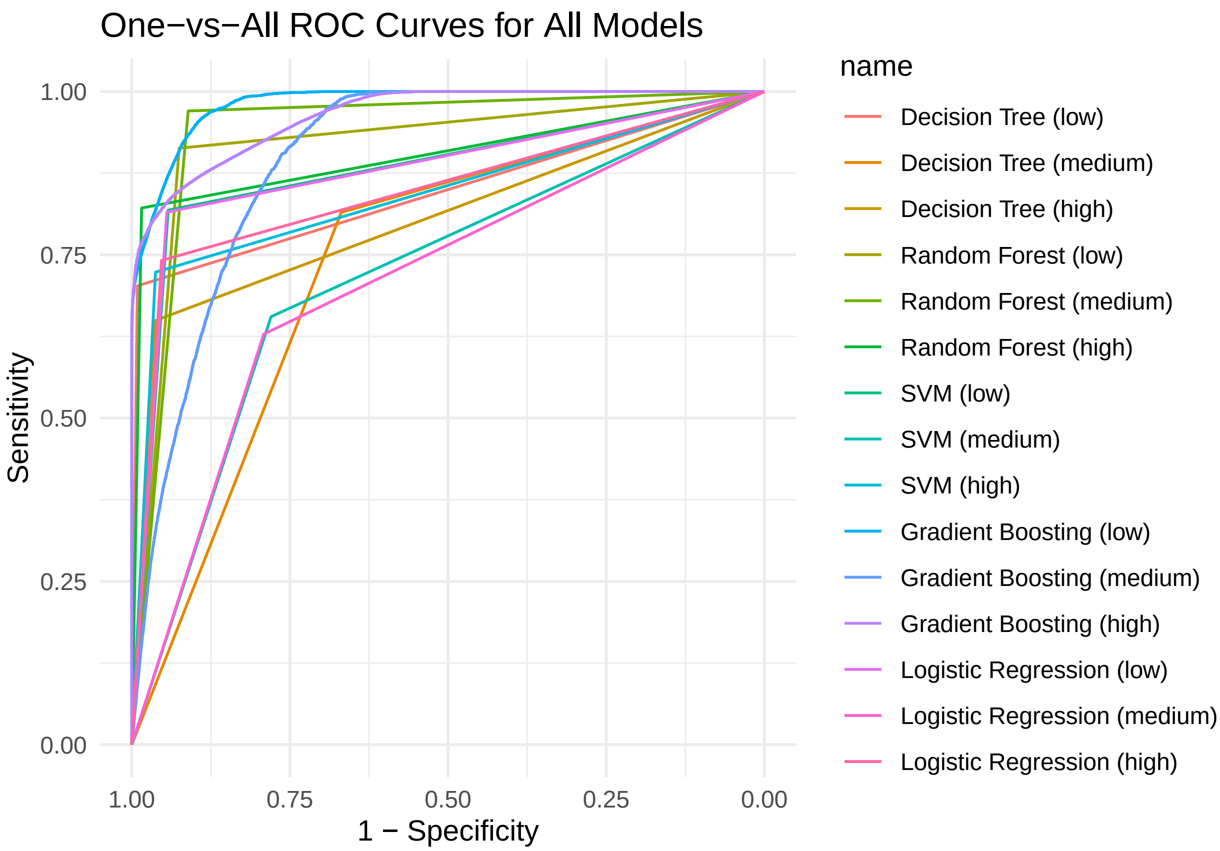



SVM Metrics
1.00
0.75
0.50
0.25
0.00
Score
high
low
medium
Class
Metric
F1
Precision
Recall


Gradient Boosting Metrics
1.00
0.75
0.50
0.25
0.00
Score
high
low
medium
Class
Metric
F1
Precision
Recall

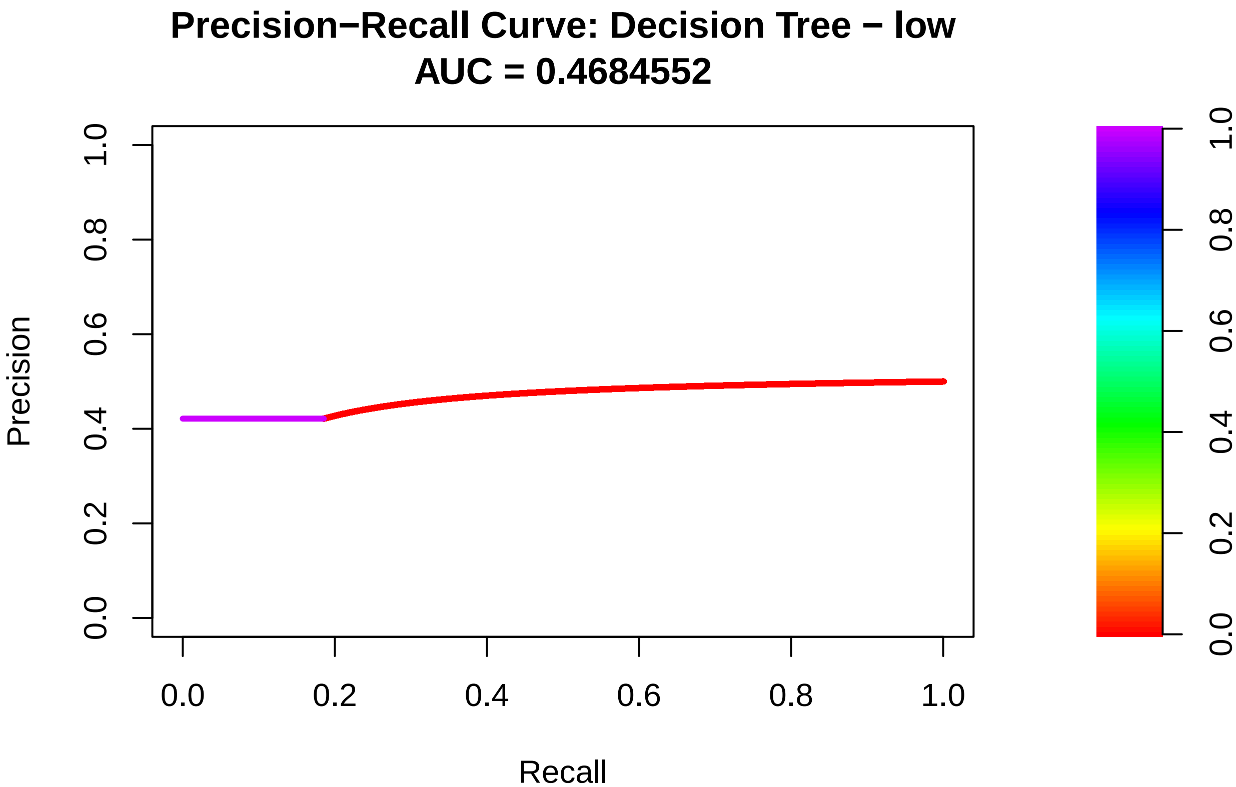

Logistic Regression Metrics
1.00
0.75
0.50
0.25
0.00
Score
high
low
medium
Class
Metric
F1
Precision
Recall

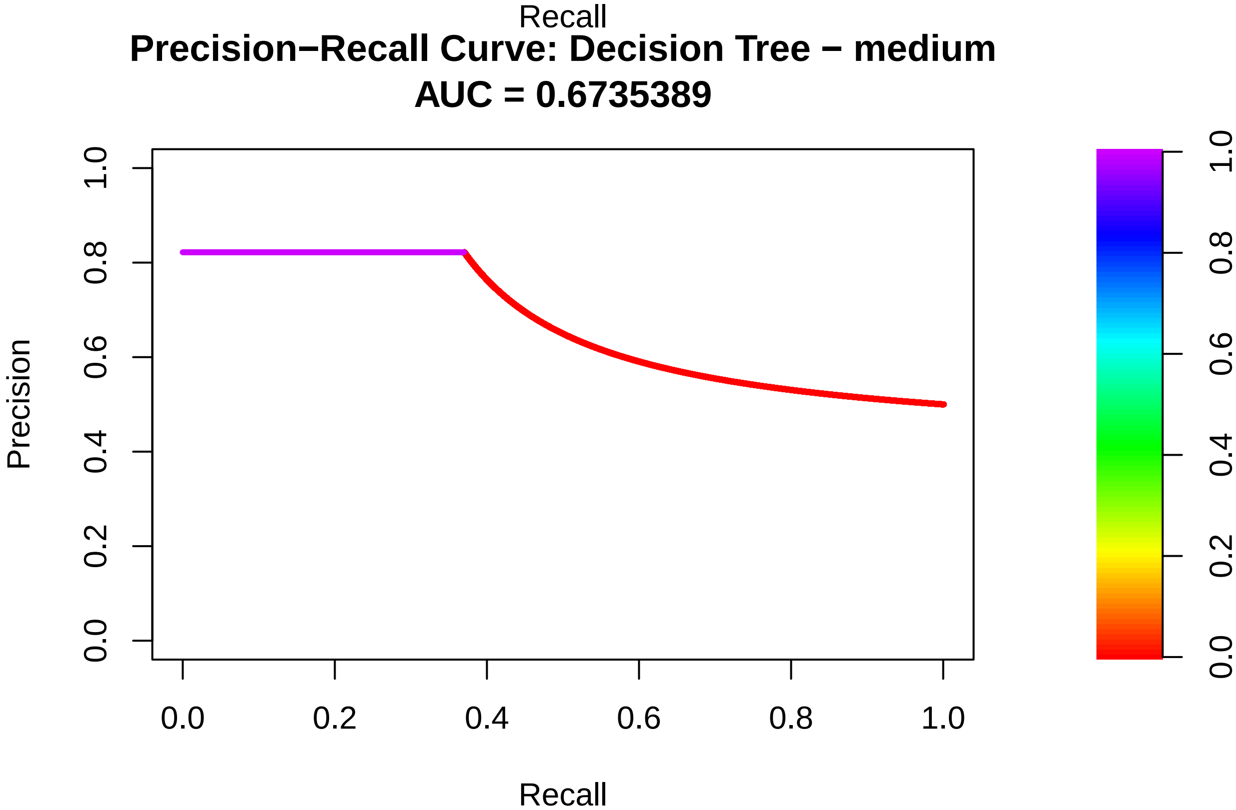
2. ROC Curves

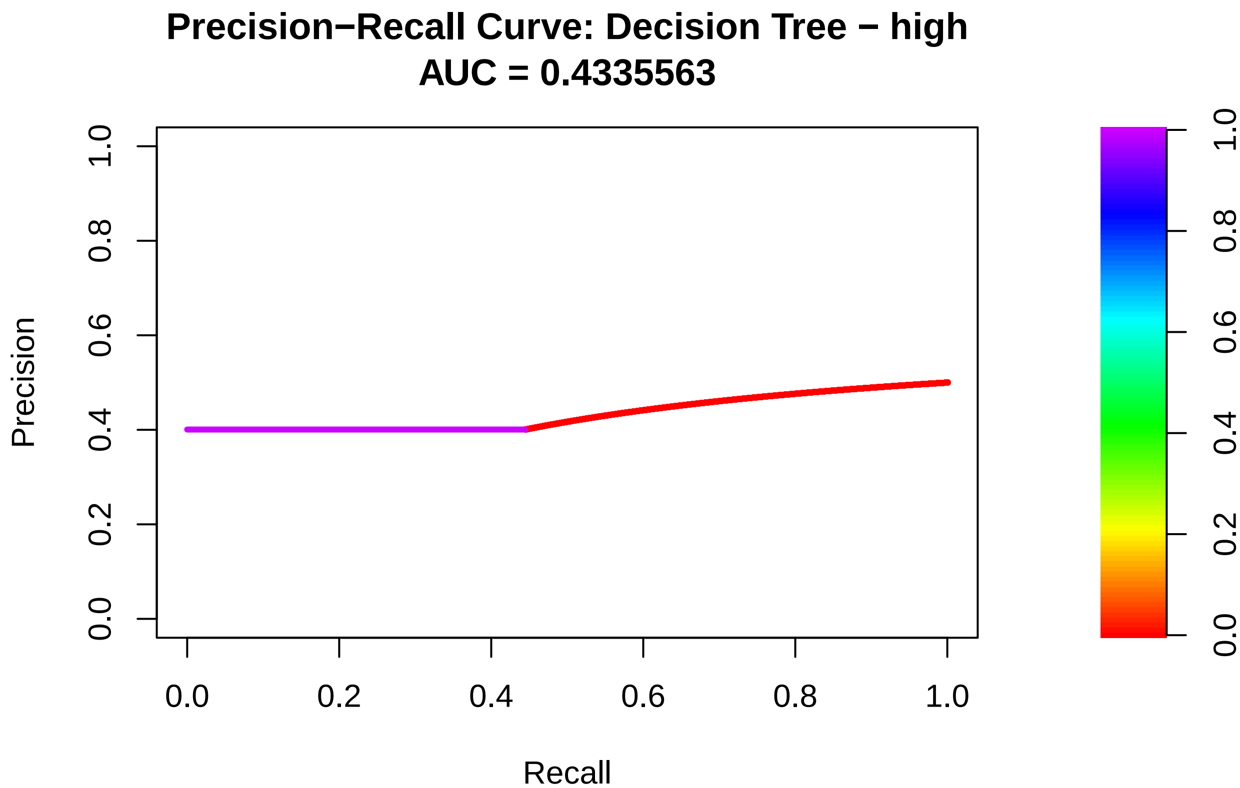

One−vs−All ROC Curves for All Models
Sensitivity
1.00
0.75
0.50
0.25
0.00
1.00
0.75
0.50
0.25
0.00
1 − Specificity
name
Decision Tree (low)
Decision Tree (medium)
Decision Tree (high)
Random Forest (low)
Random Forest (medium)
Random Forest (high)
SVM (low)
SVM (medium)
SVM (high)
Gradient Boosting (low)
Gradient Boosting (medium)
Gradient Boosting (high)
Logistic Regression (low)
Logistic Regression (medium)
Logistic Regression (high)

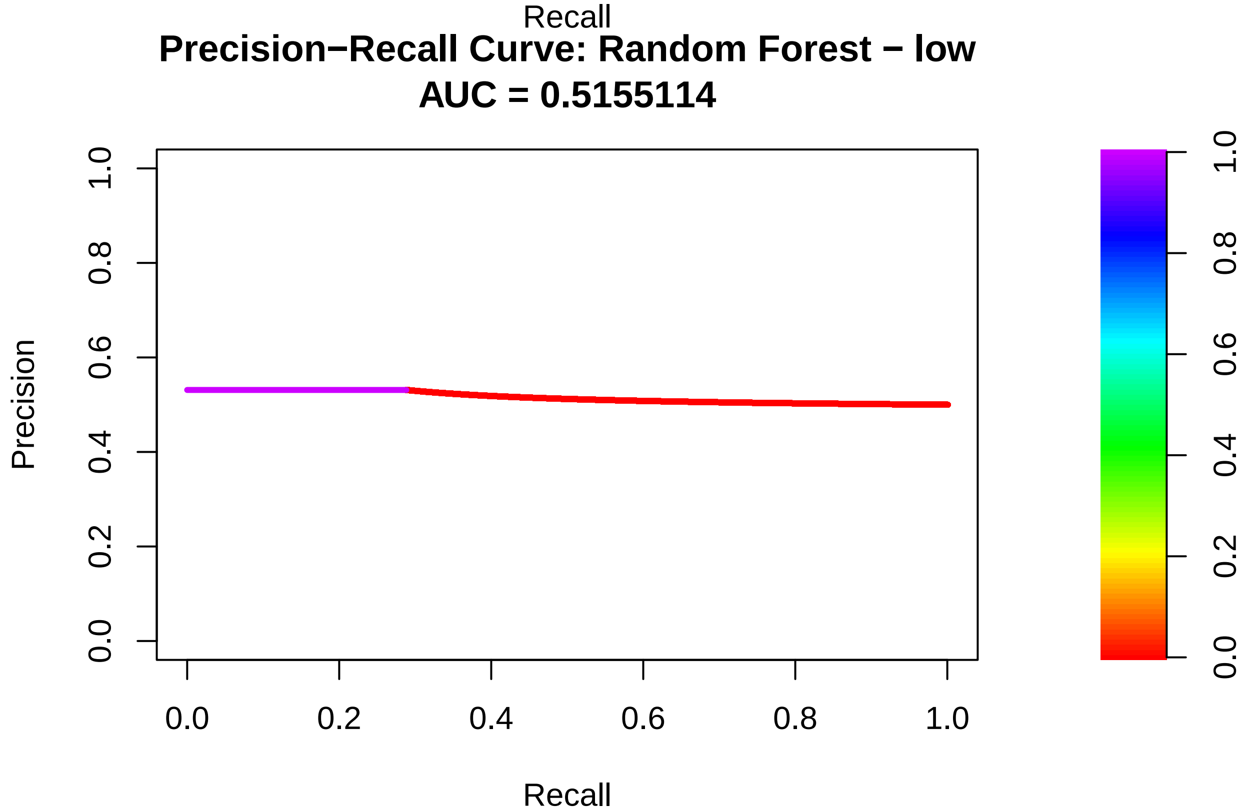
---

3. Precision-Recall Curves

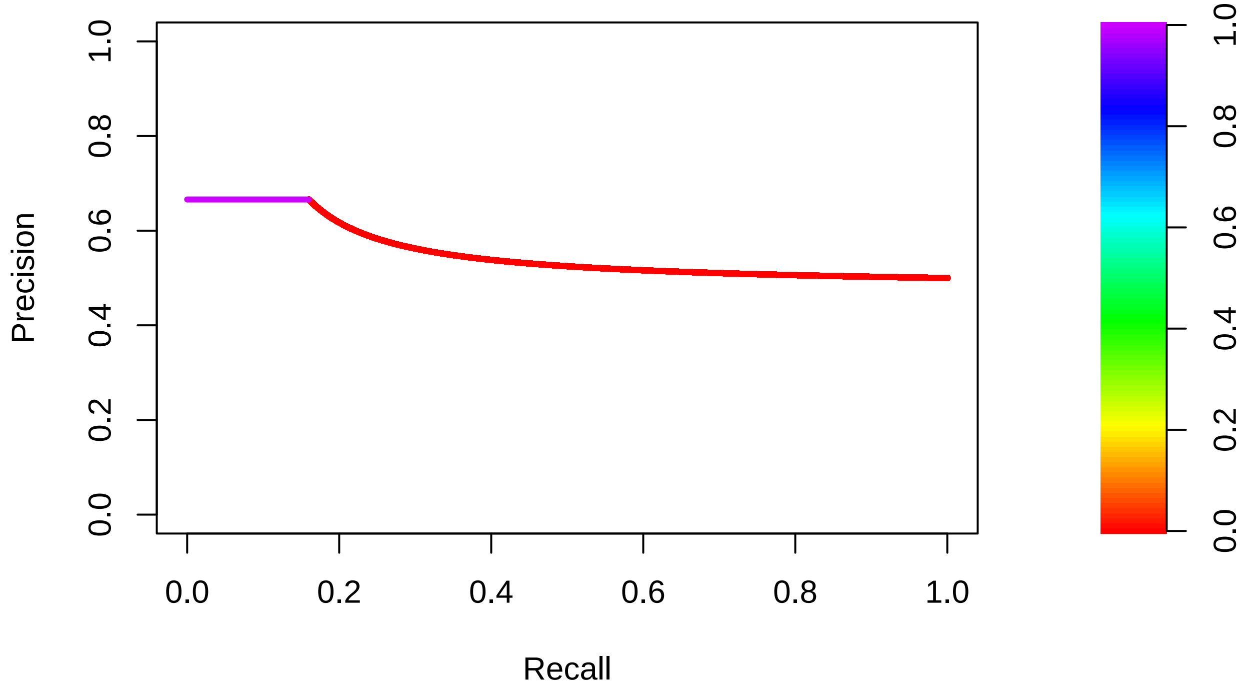




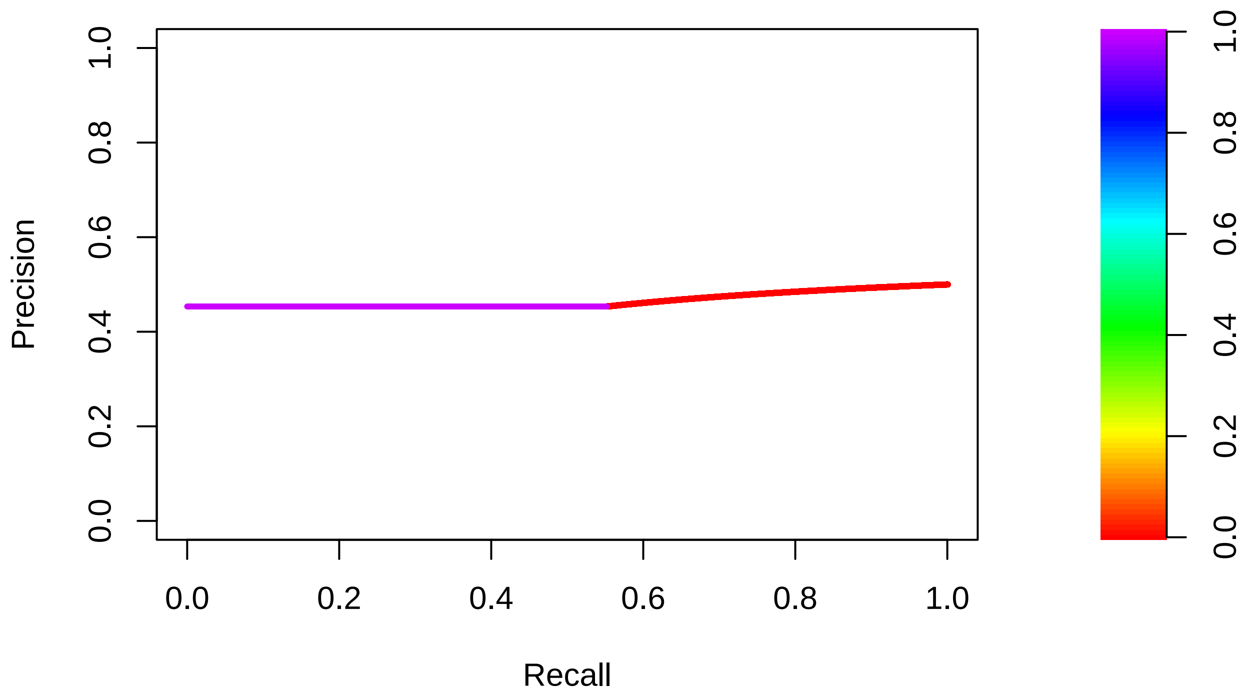

Precision−Recall Curve: Decision Tree − high
AUC = 0.4335563
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

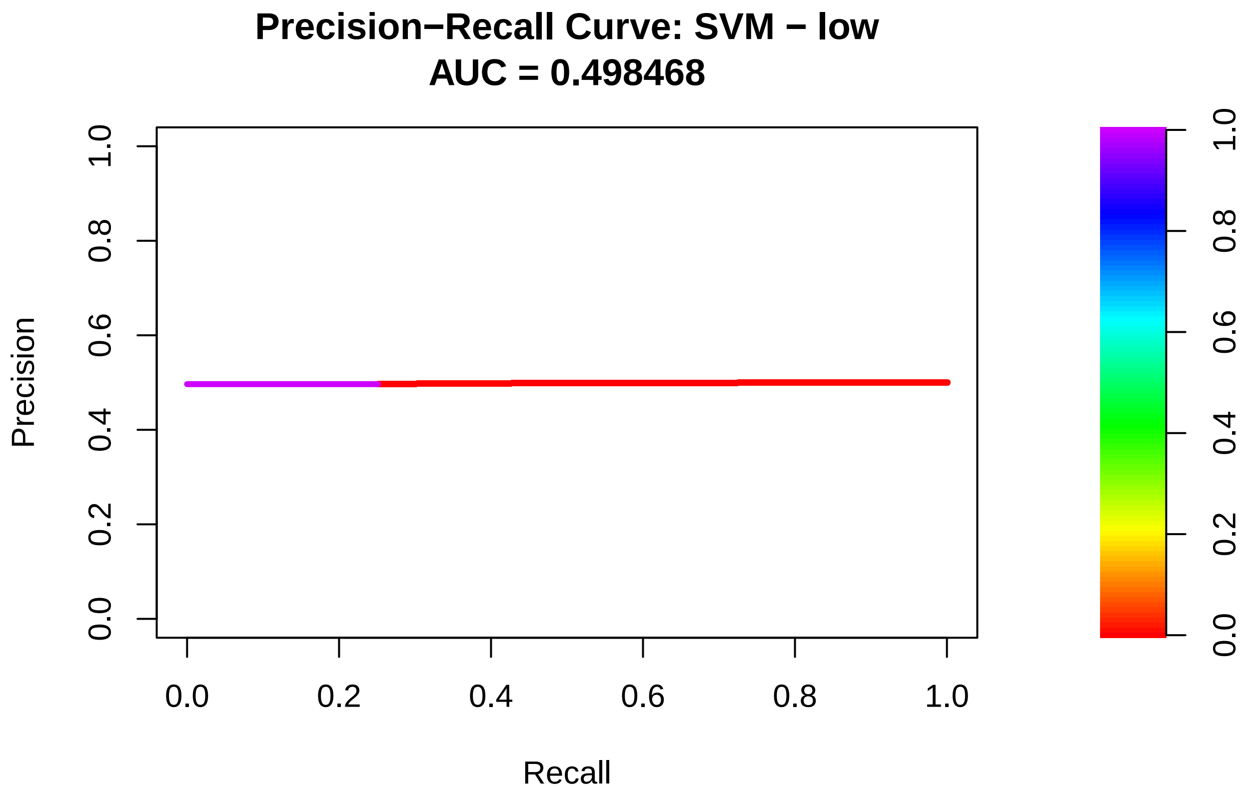

Precision−Recall Curve: Random Forest − low
AUC = 0.5155114
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

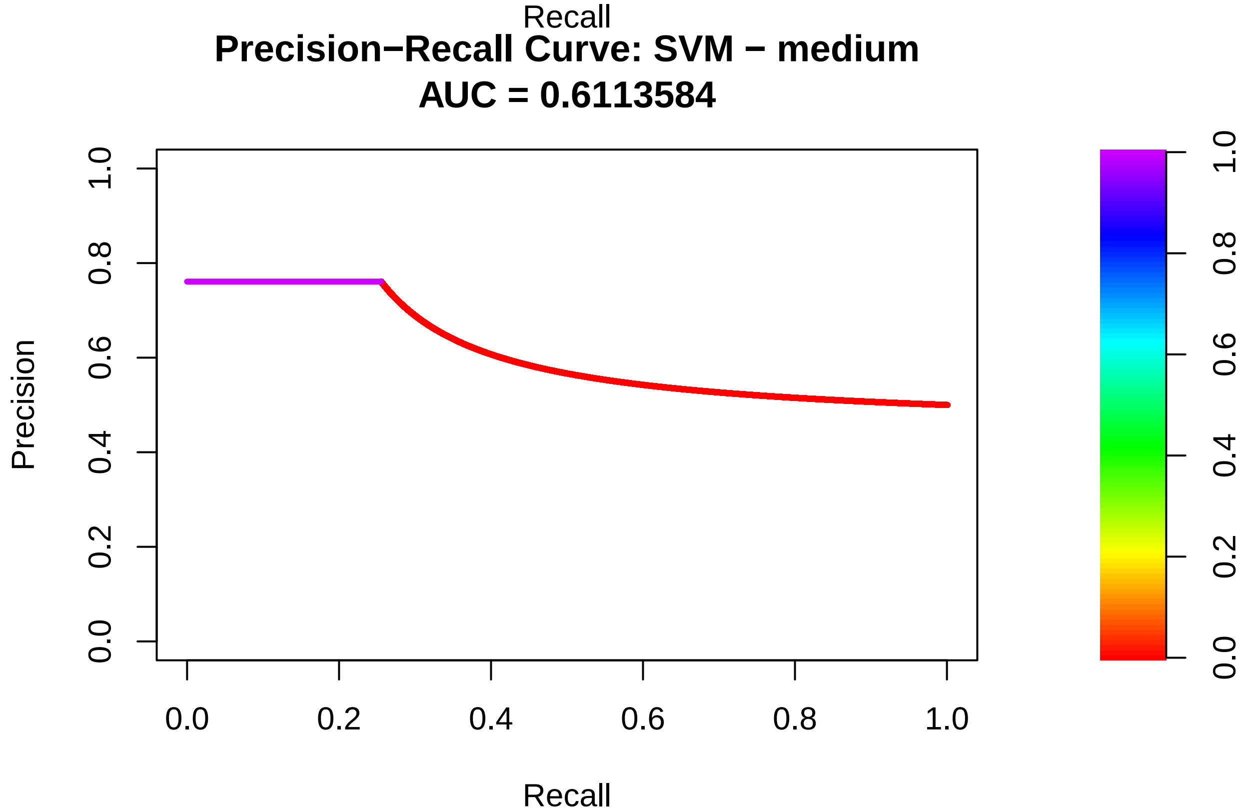
Precision−Recall Curve: Random Forest − medium
AUC = 0.5533086



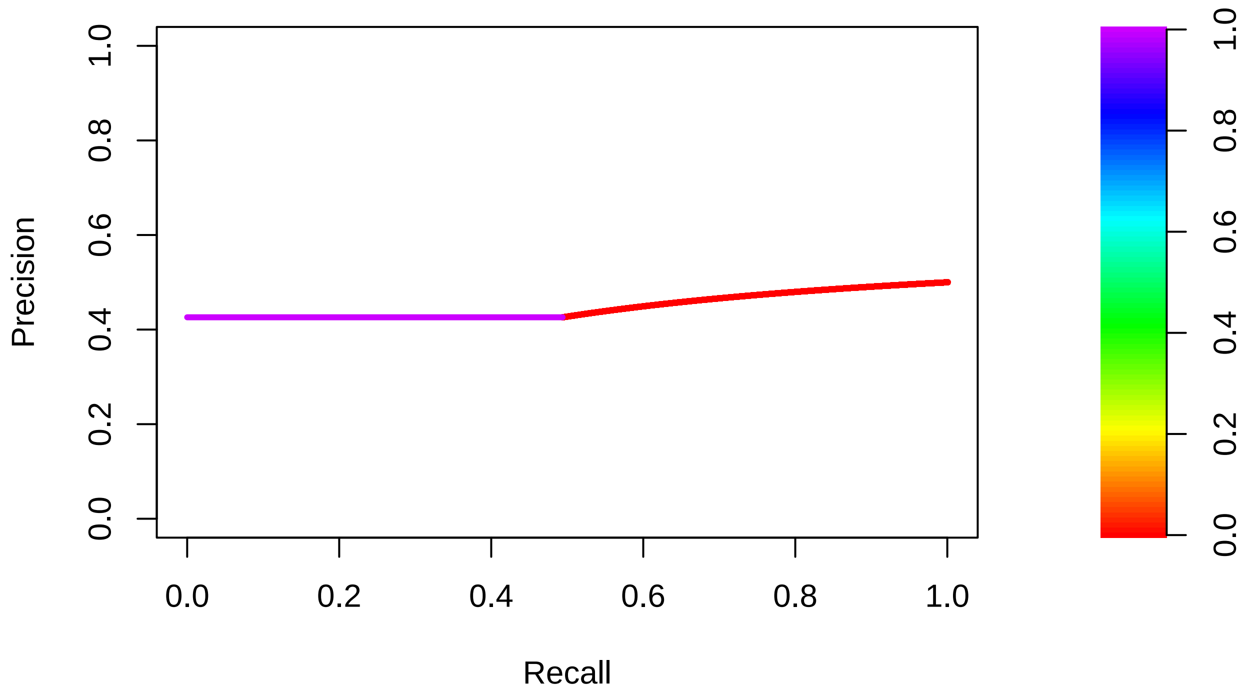
Precision−Recall Curve: Random Forest − high
AUC = 0.4655508



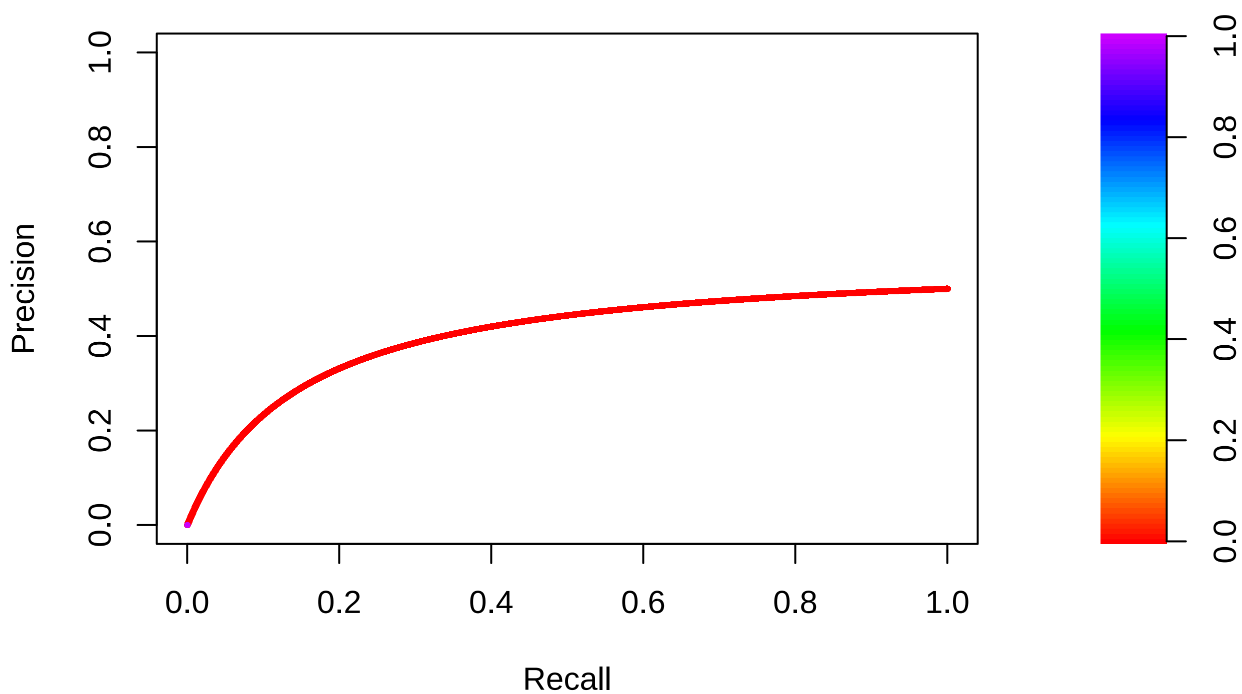

Precision−Recall Curve: SVM − low
AUC = 0.498468
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

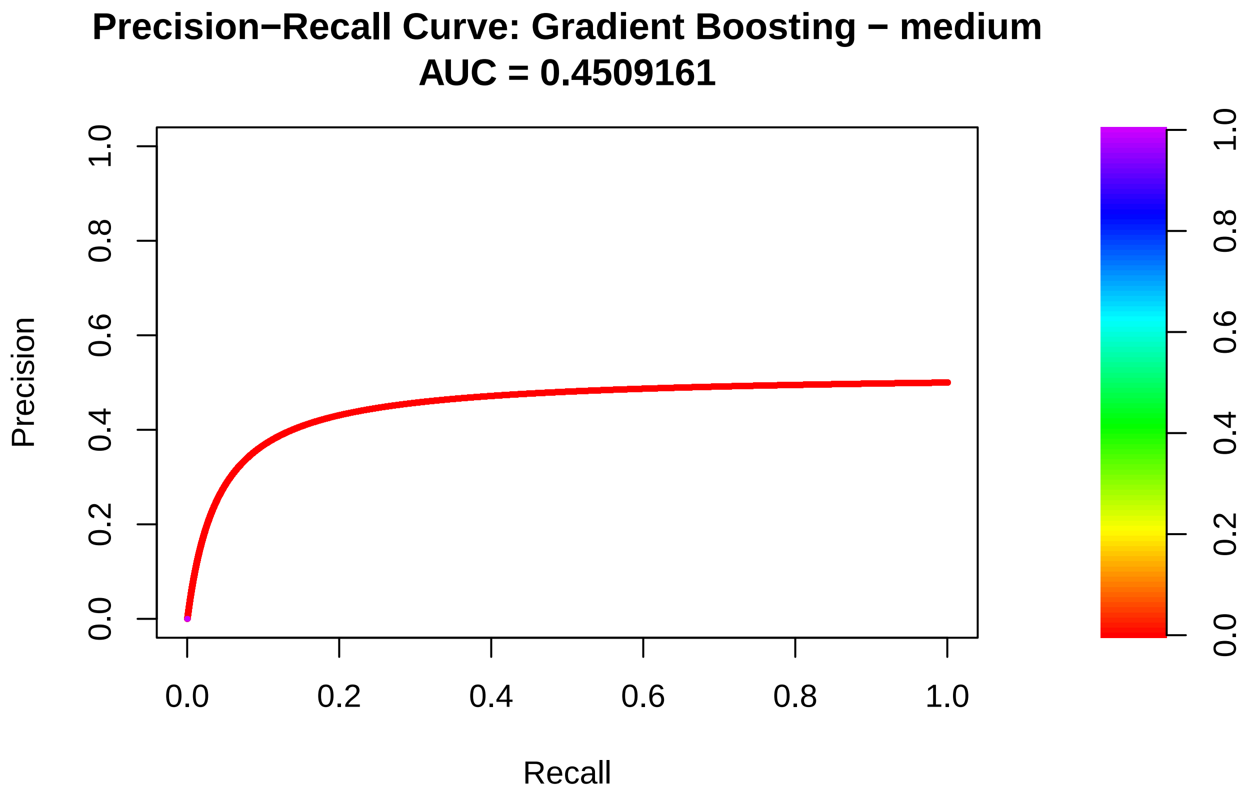

Precision−Recall Curve: SVM − medium
AUC = 0.6113584
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

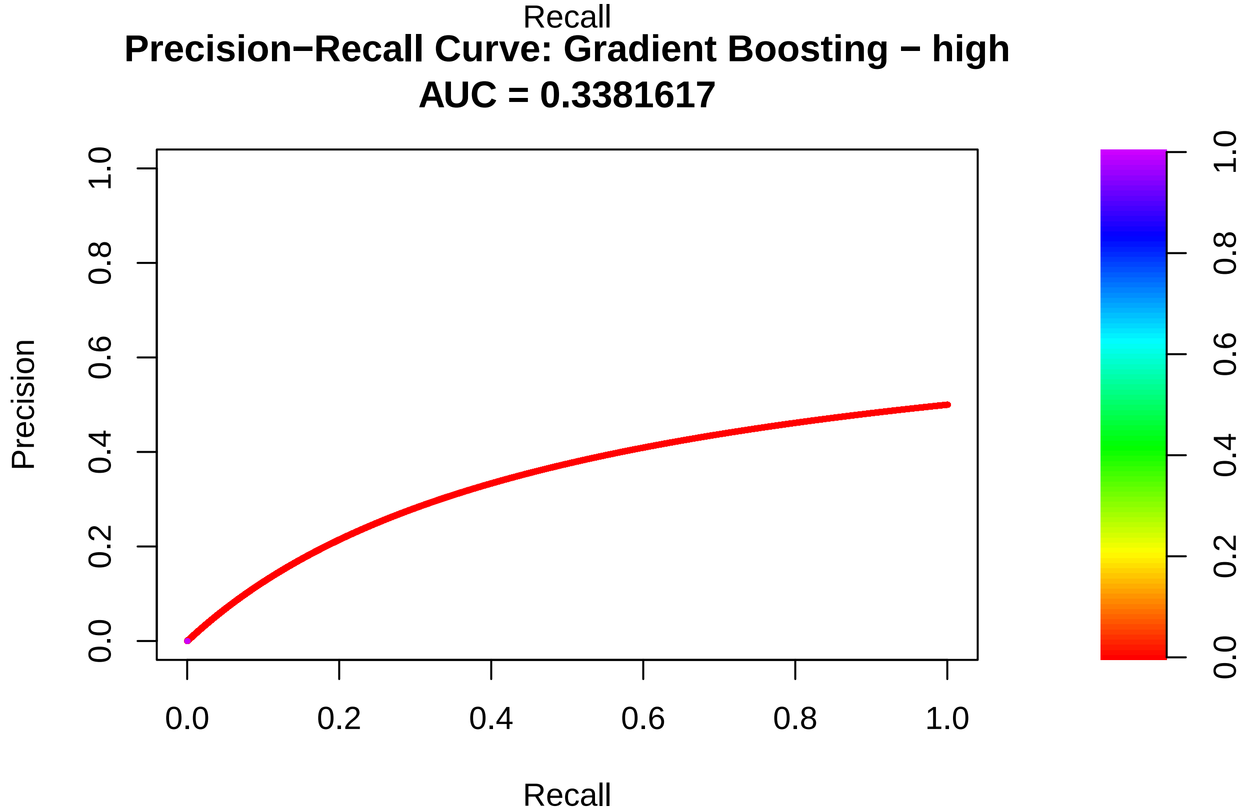
Precision−Recall Curve: SVM − high
AUC = 0.4481092

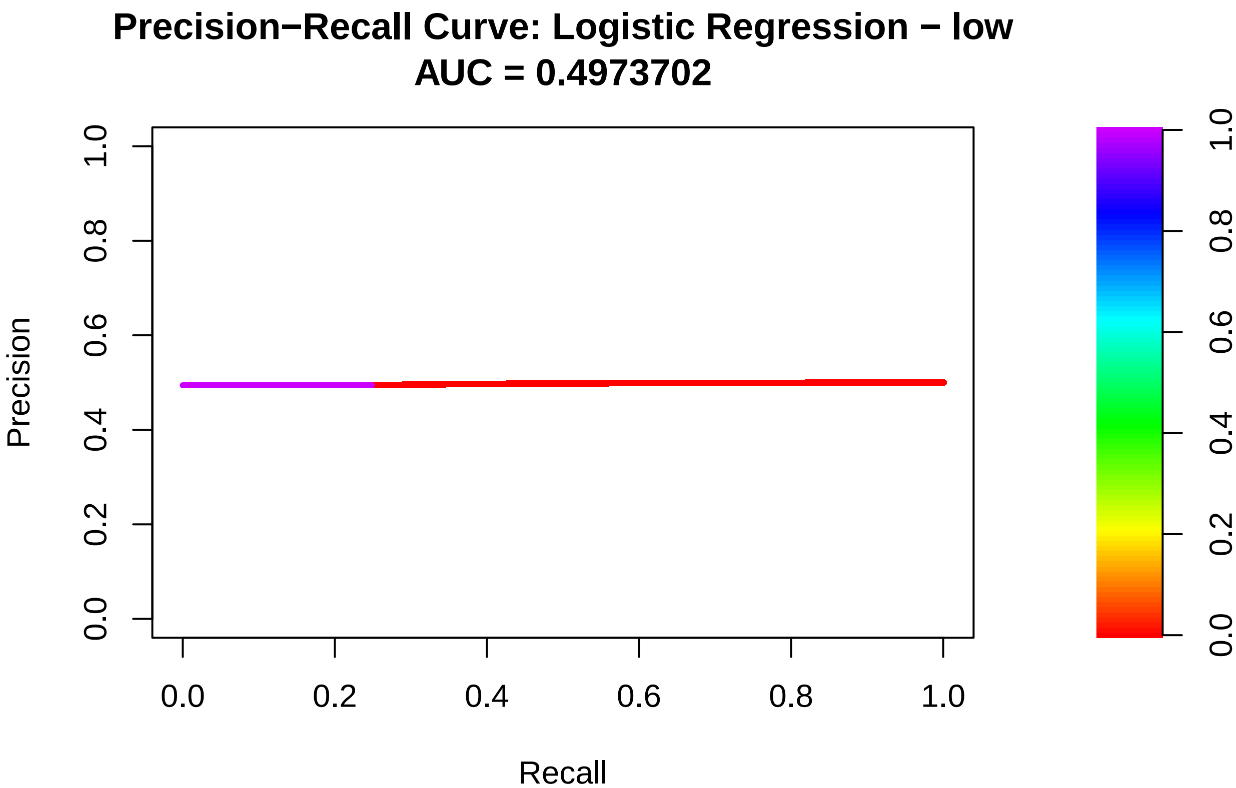


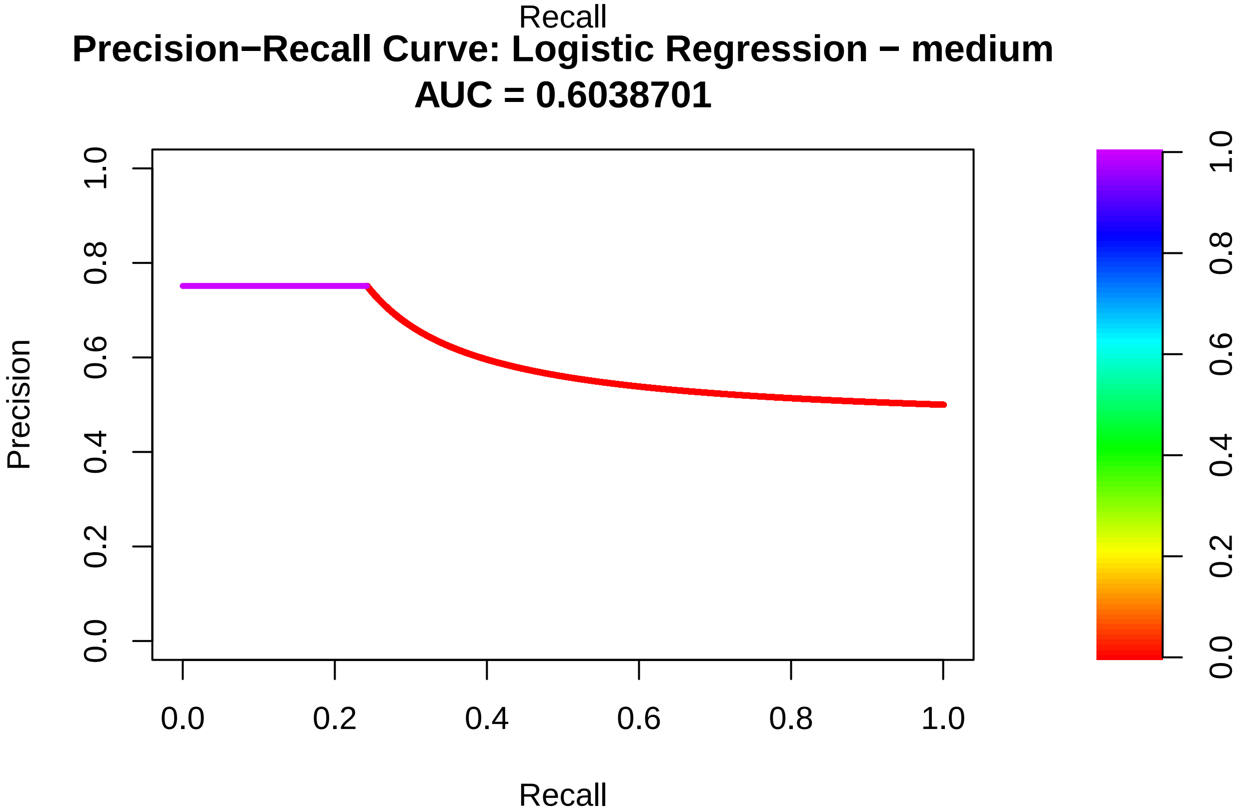
Precision−Recall Curve: Gradient Boosting − low
AUC = 0.400705

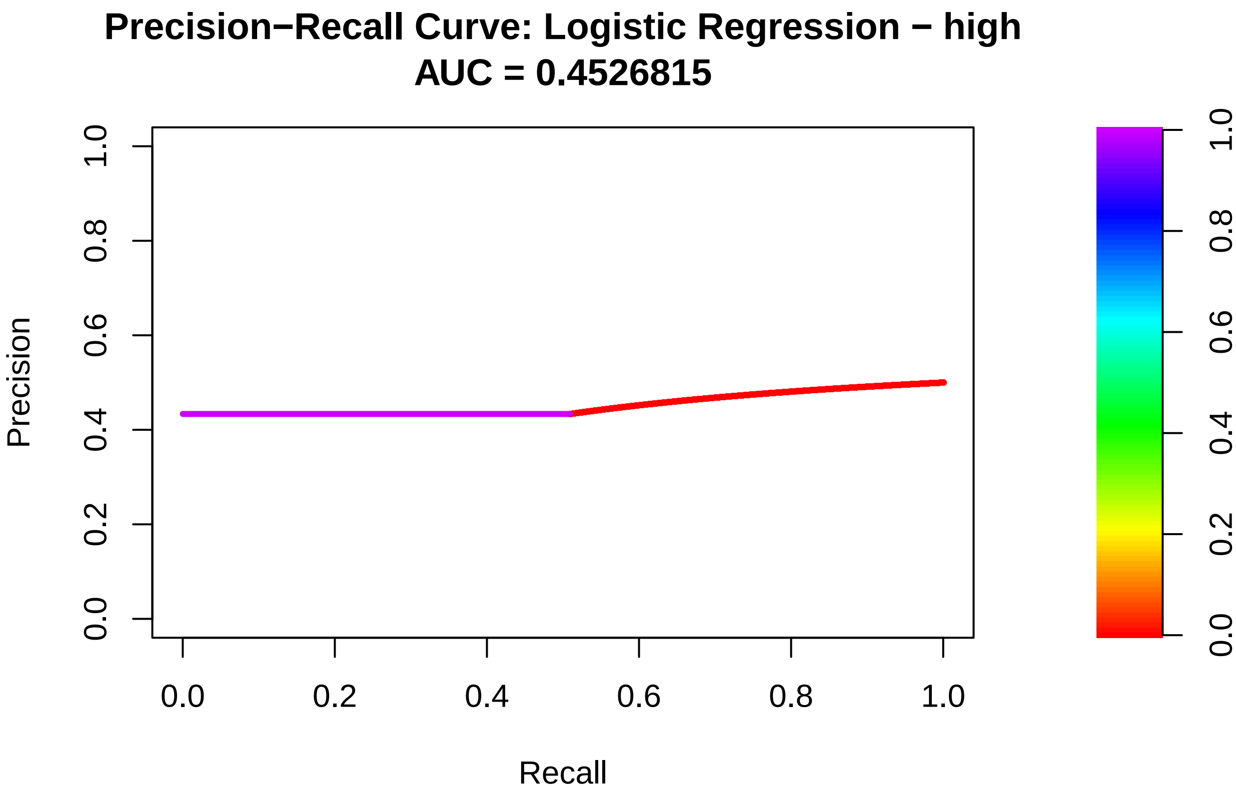



Precision−Recall Curve: Gradient Boosting − medium
AUC = 0.4509161
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

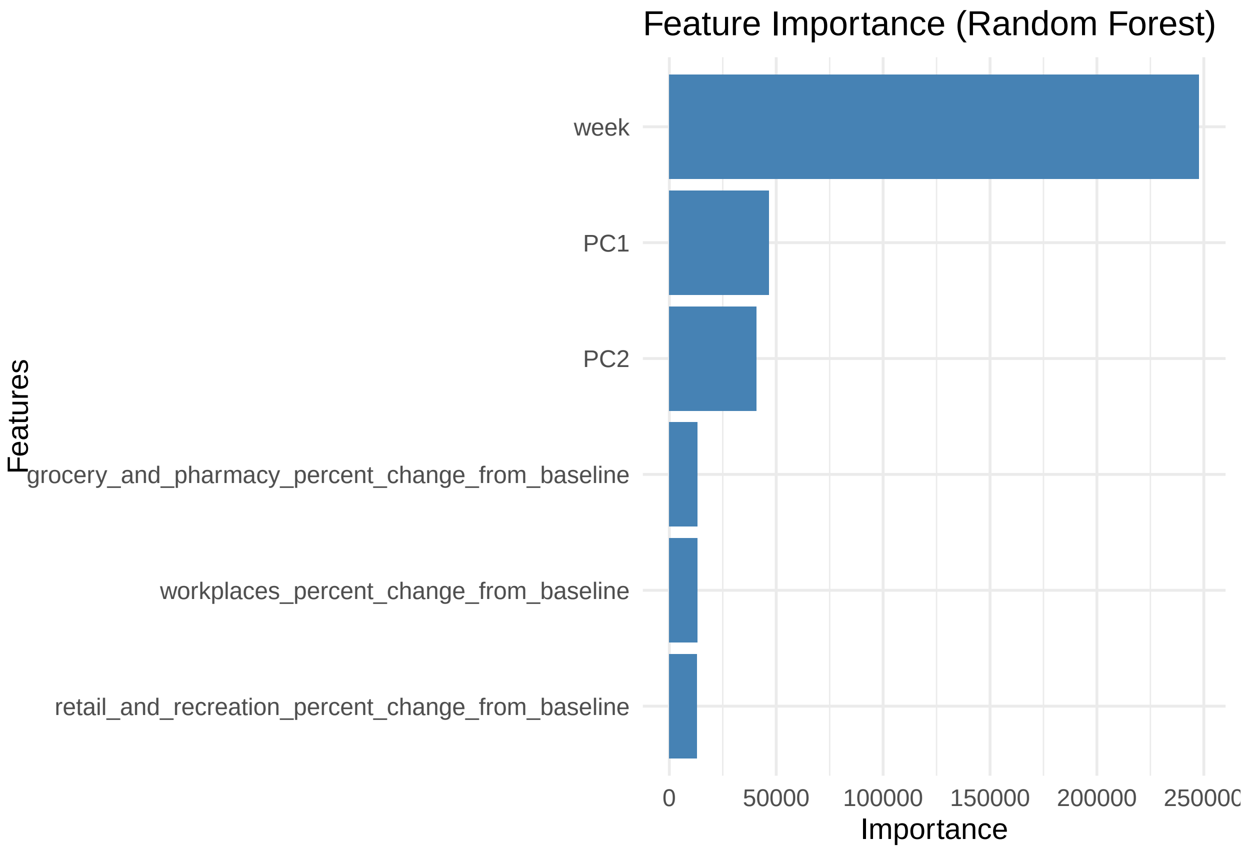

Precision−Recall Curve: Gradient Boosting − high
AUC = 0.3381617
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

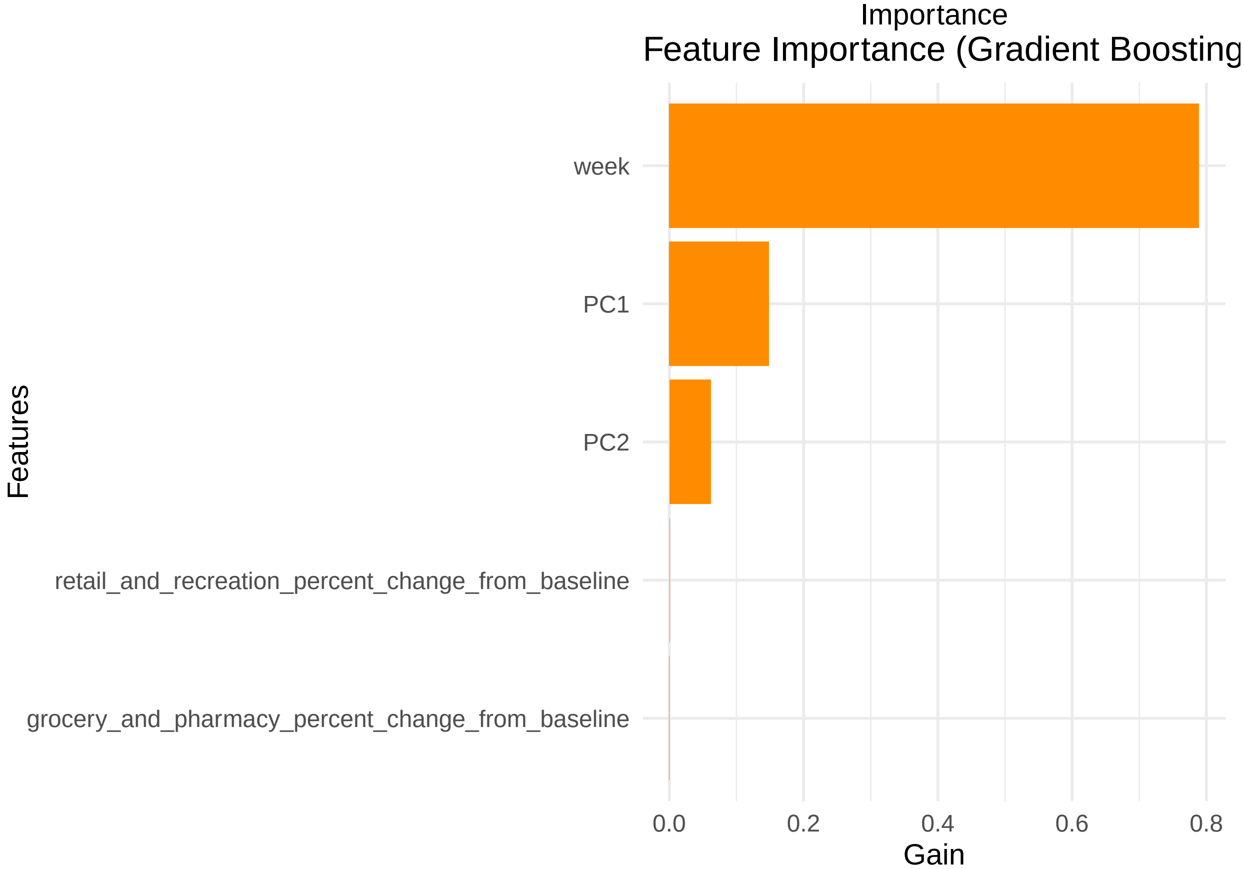

Precision−Recall Curve: Logistic Regression − low
AUC = 0.4973702
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0


Precision−Recall Curve: Logistic Regression − medium
AUC = 0.6038701
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0


Precision−Recall Curve: Logistic Regression − high
AUC = 0.4526815
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0

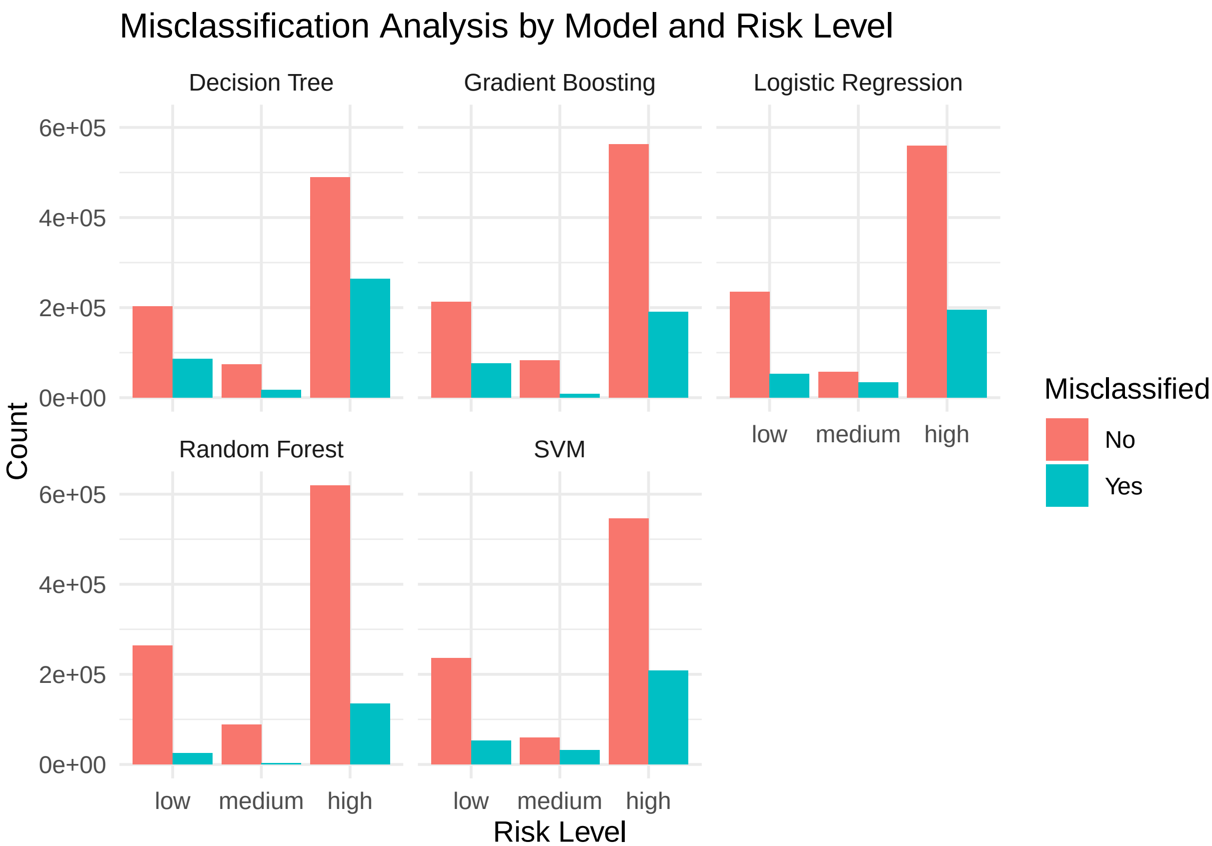
4. Feature Importance





---

5. Misclassification Analysis



Conclusion

These visuals provide insights into the models' performance and help stakeholders understand the trade-offs between accuracy, precision, and recall for each classification method. They also highlight areas for improvement, such as addressing misclassifications.

---

4. Deployment

- **Practical Use**: The model can guide early interventions (e.g., mask mandates, closures).
- **Update Frequency**: Weekly updates based on new data.
- **Integration**: Stakeholders can incorporate model predictions into decision-making frameworks.

```r
# Save all models
save_model(dt_model, "dt_model_balanced.rds")
```

```
## Model saved to: dt_model_balanced.rds
```

```r
save_model(rf_model, "rf_model_balanced.rds")
```

```
## Model saved to: rf_model_balanced.rds
```

```r
save_model(svm_model, "svm_model_balanced.rds")
```

```
## Model saved to: svm_model_balanced.rds
```

```r
save_model(xgb_model, "xgb_model_balanced.rds")
```

```
## Model saved to: xgb_model_balanced.rds
```

```r
save_model(logistic_model, "logistic_model_balanced.rds")
```

```
## Model saved to: logistic_model_balanced.rds
```

```r
# Load all models
loaded_dt_model <- load_model("dt_model_balanced.rds")
```

```
## Model loaded from: dt_model_balanced.rds
```

```r
loaded_rf_model <- load_model("rf_model_balanced.rds")
```

```
## Model loaded from: rf_model_balanced.rds
```

```r
loaded_svm_model <- load_model("svm_model_balanced.rds")
```

```
## Model loaded from: svm_model_balanced.rds
```

```r
loaded_xgb_model <- load_model("xgb_model_balanced.rds")
```

```
## Model loaded from: xgb_model_balanced.rds
```

```r
loaded_logistic_model <- load_model("logistic_model_balanced.rds")
```

```
## Model loaded from: logistic_model_balanced.rds
```

Appendix

- **Team Contributions**:
  - Olivia Hofmann: Lead on data preparation and feature engineering.
  - Michael Perkins: Lead on modeling and evaluation.
- **Graduate Work**:
  - Additional models: Gradient Boosting and k-Nearest Neighbors (to be implemented).