

Data Mining Project 3

Olivia Hofmann, Michael Perkins

2024-12-04

Contents

0.1	Introduction	1
0.2	Deployment: Discussing practical application and maintenance of the models.	1
0.3	1. Data Preparoation	1
0.4	2. Modeling	3
0.5	3. Evaluation	5
0.6	4. Deployment	29
0.7	Appendix	30

0.1 Introduction

The COVID-19 pandemic highlighted the importance of preparedness for infectious disease outbreaks. Anticipating which counties are at higher risk can enable early interventions, potentially saving lives and mitigating economic impacts. This project aims to classify U.S. counties into **high**, **medium**, or **low** risk categories for future pandemics based on historical COVID-19 data and other socioeconomic factors.

Following the CRISP-DM framework, we focus on:

- Data Preparation: Defining classes, cleaning data, feature engineering, and handling missing values.
- Modeling: Training and tuning multiple classification models.
- Evaluation: Assessing model performance and their utility to stakeholders.
-

0.2 Deployment: Discussing practical application and maintenance of the models.

0.3 1. Data Preparoation

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

0.3.1 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.

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

# Convert data from long to wide format
final_merged_dataset <- final_merged_dataset %>%
  pivot_wider(names_from = metric, values_from = count)
```

0.3.2 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(
    confirmed_cases >= 50 ~ "high",
    confirmed_cases >= 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()

```

0.4 2. Modeling

0.4.1 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))

```

0.4.2 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)

```

0.4.3 Model 3: Support Vector Machine (SVM)

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

```

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')

```

0.4.4 Model 4: Gradient Boosting

- Advantages:

```
# 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
)
```

0.4.5 Model 5: Logistic Regression

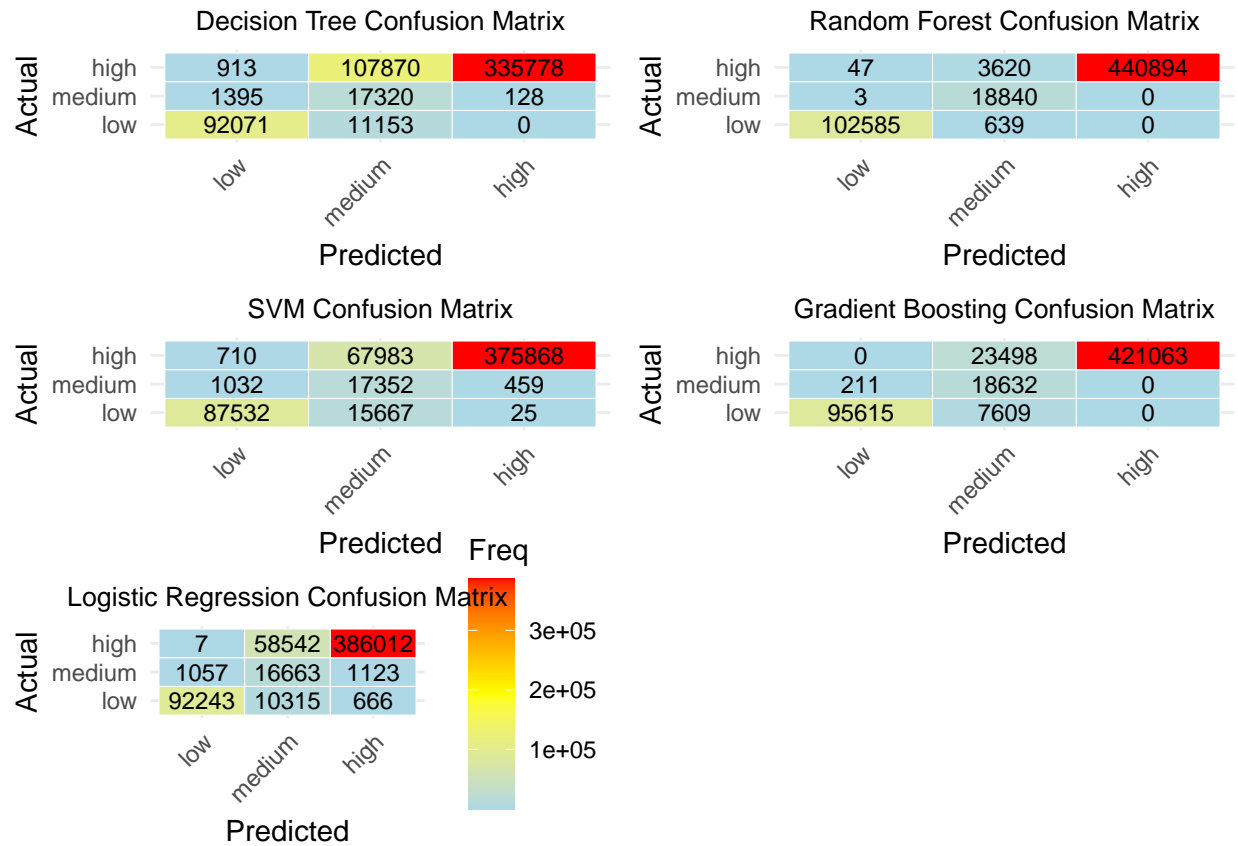
- Advantages:

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

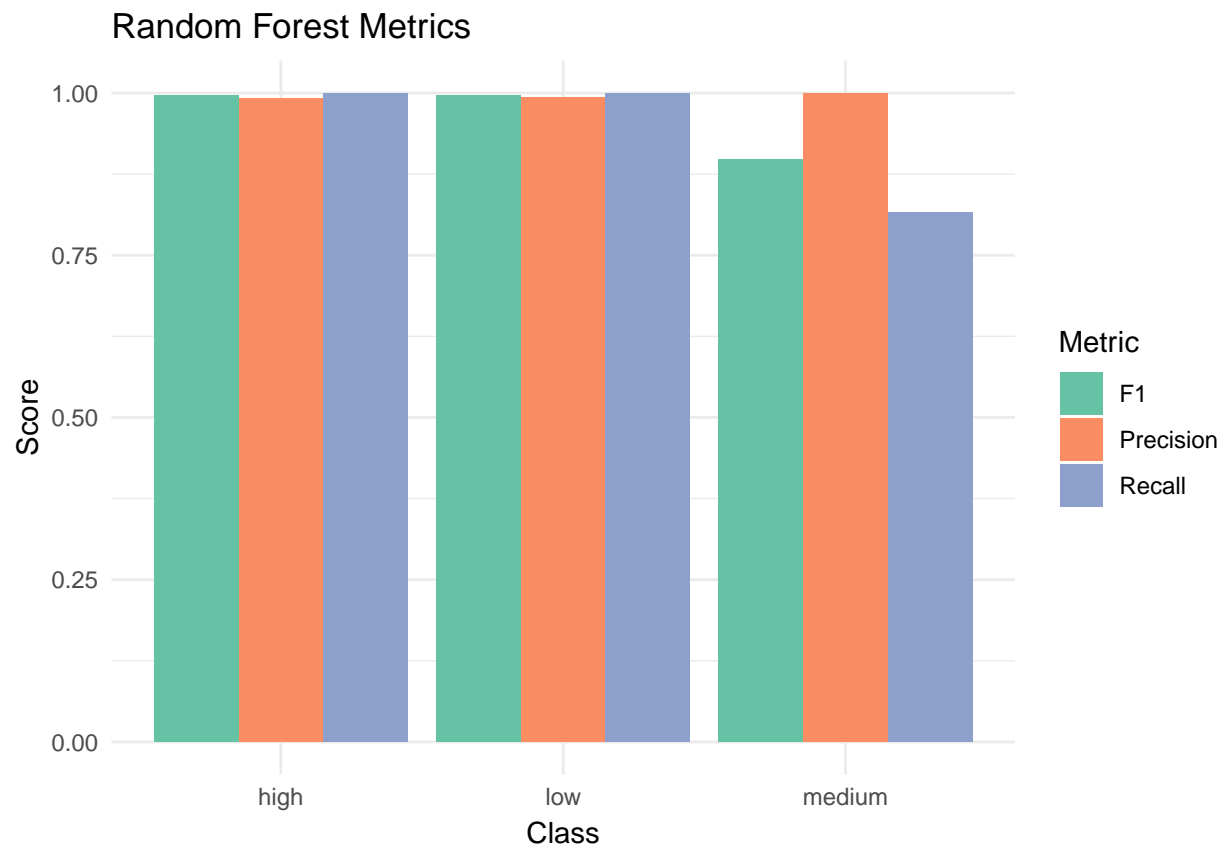
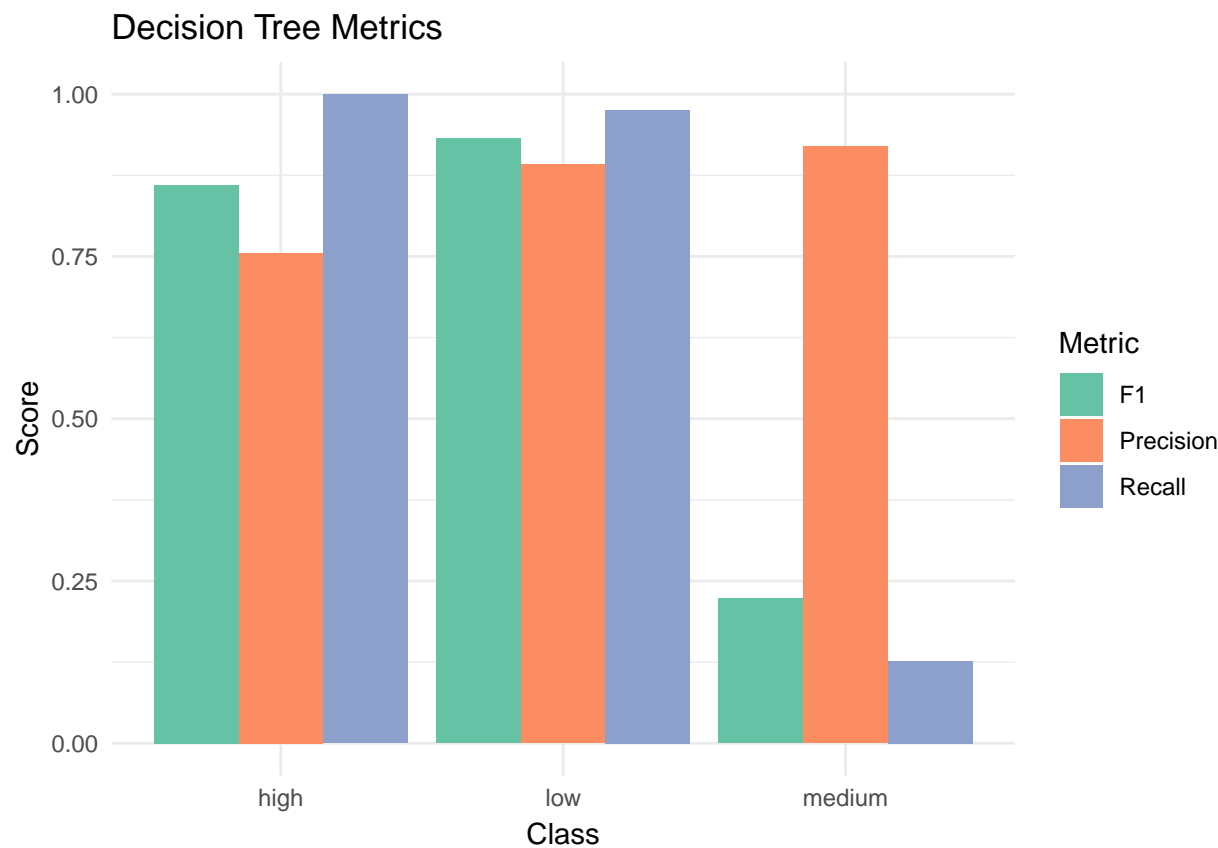
```
## # weights: 24 (14 variable)
## initial value 146546.090410
## iter 10 value 144726.645825
## iter 20 value 49159.376764
## iter 30 value 41823.952812
## iter 40 value 41660.518093
## iter 50 value 41296.754356
## final value 40503.996163
## converged
```

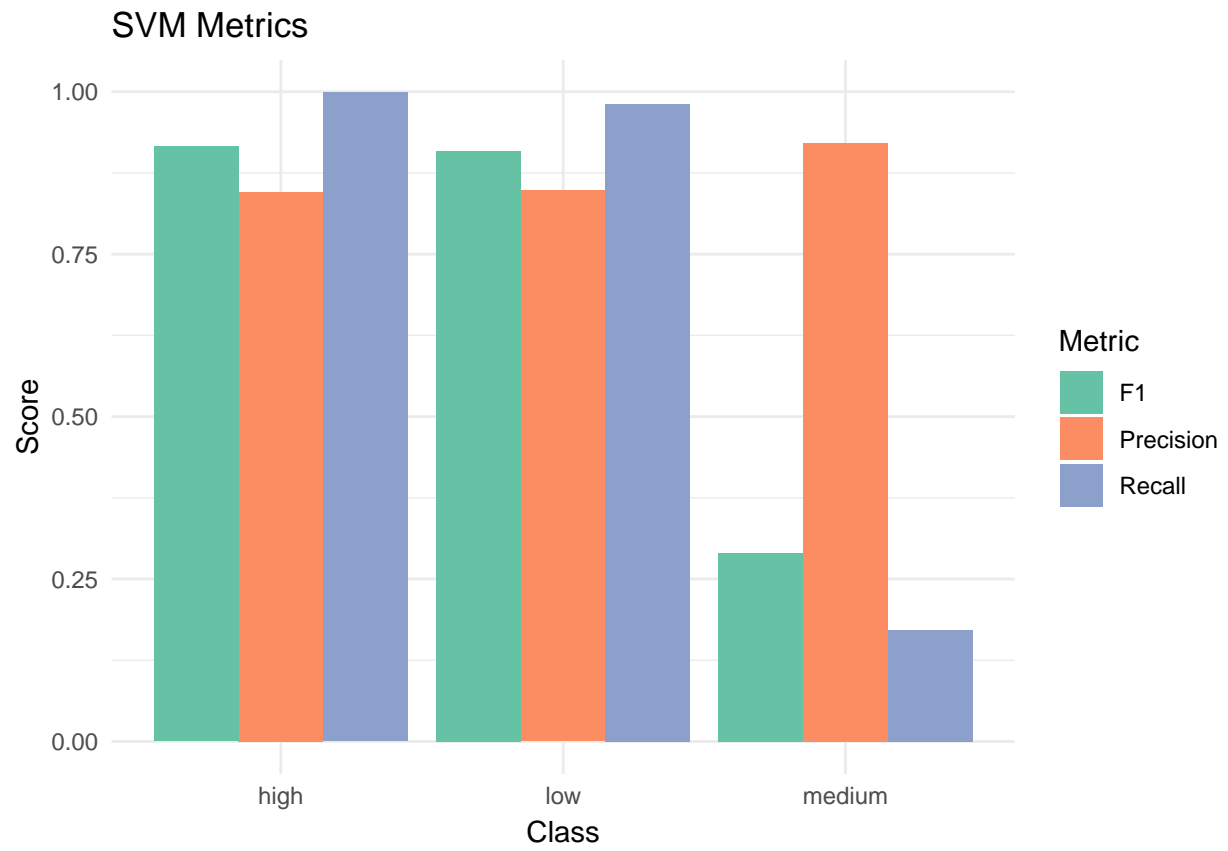
0.5 3. Evaluation

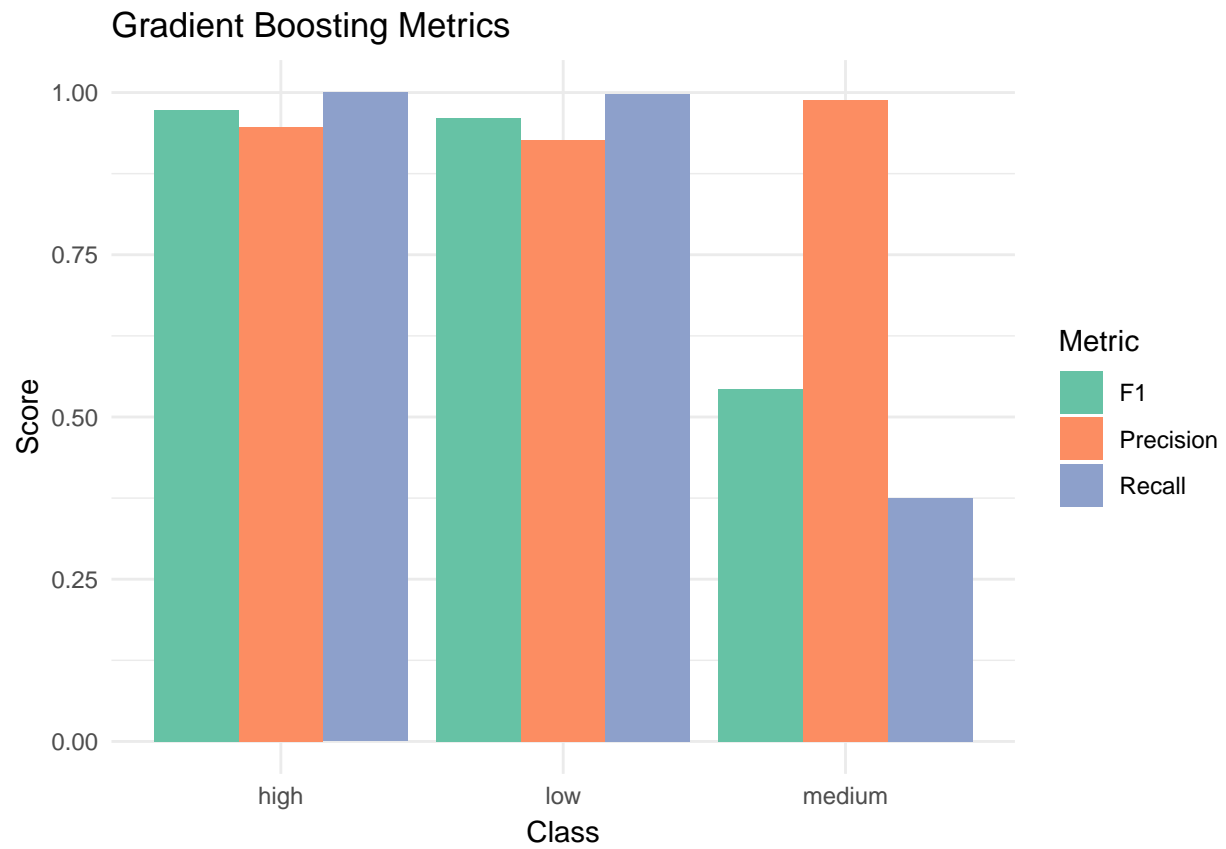
0.5.1 1. Confusion Matrix Heatmaps

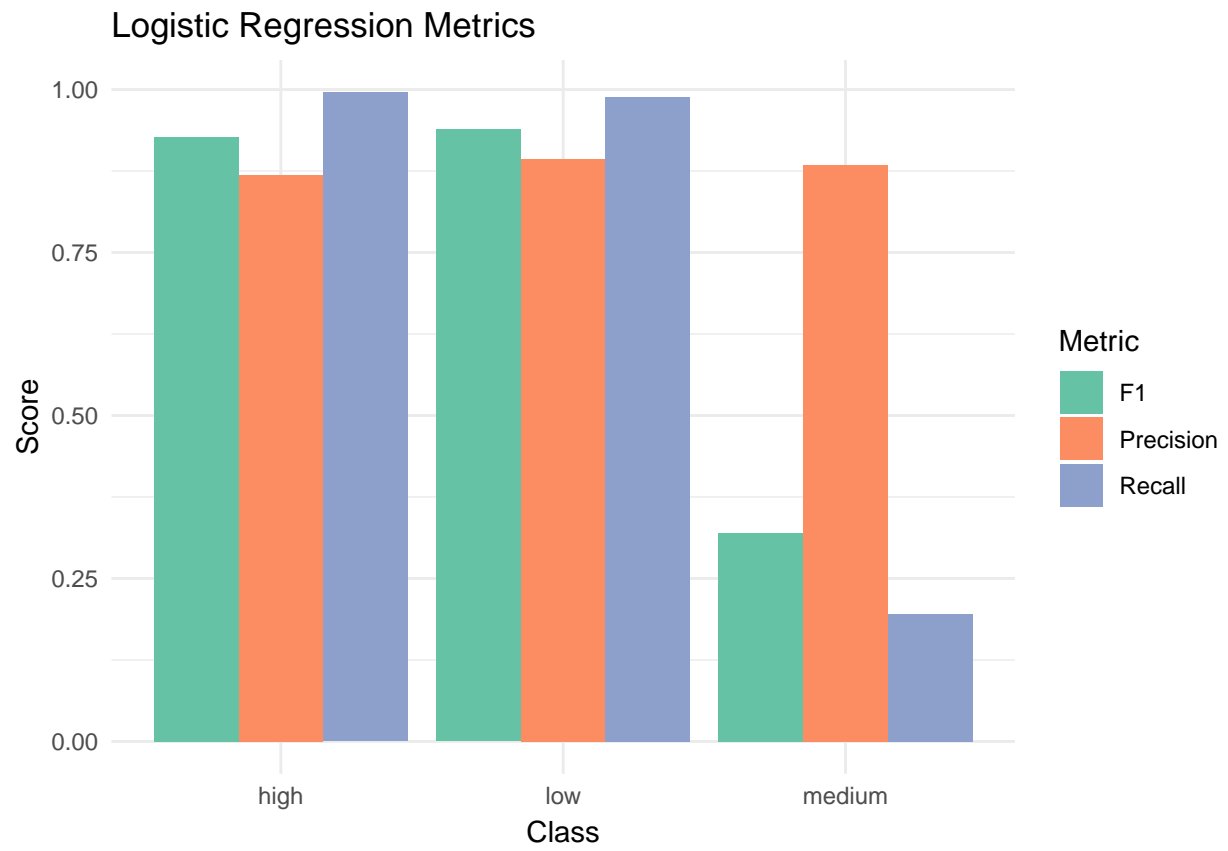


0.5.2 2. Precision, Recall and F1-score

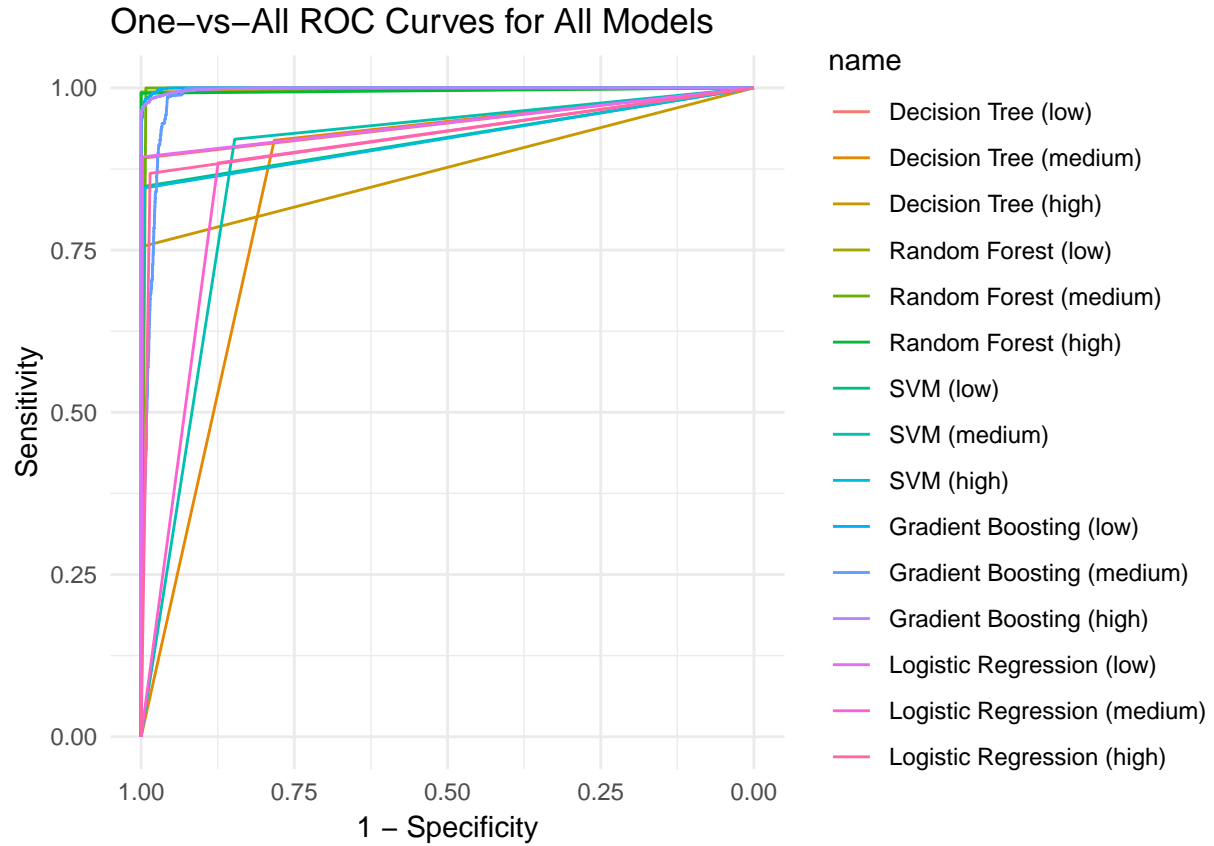






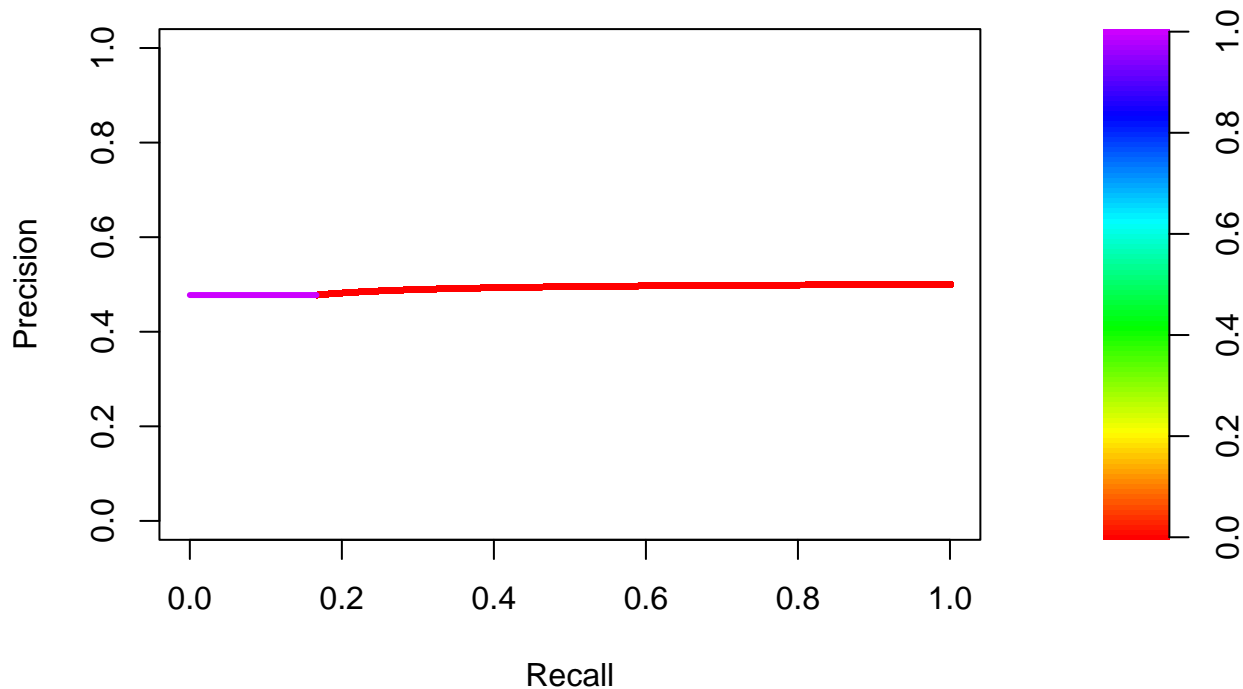


0.5.3 2. ROC Curves

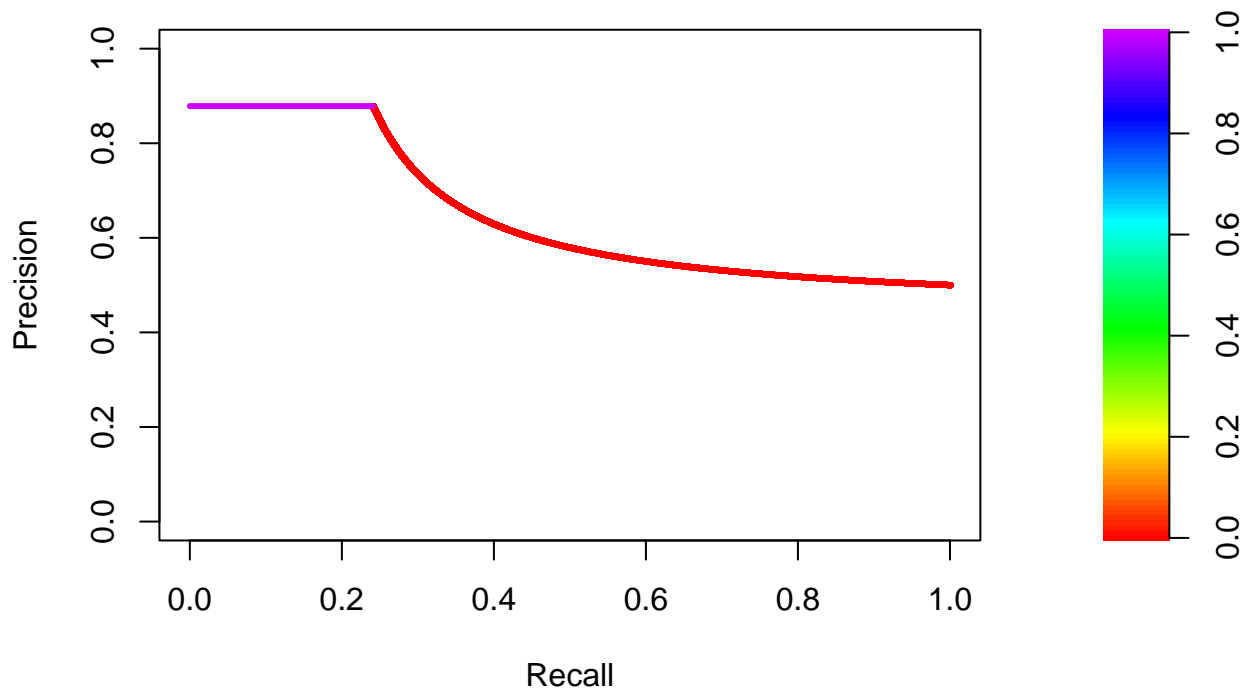


0.5.4 3. Precision-Recall Curves

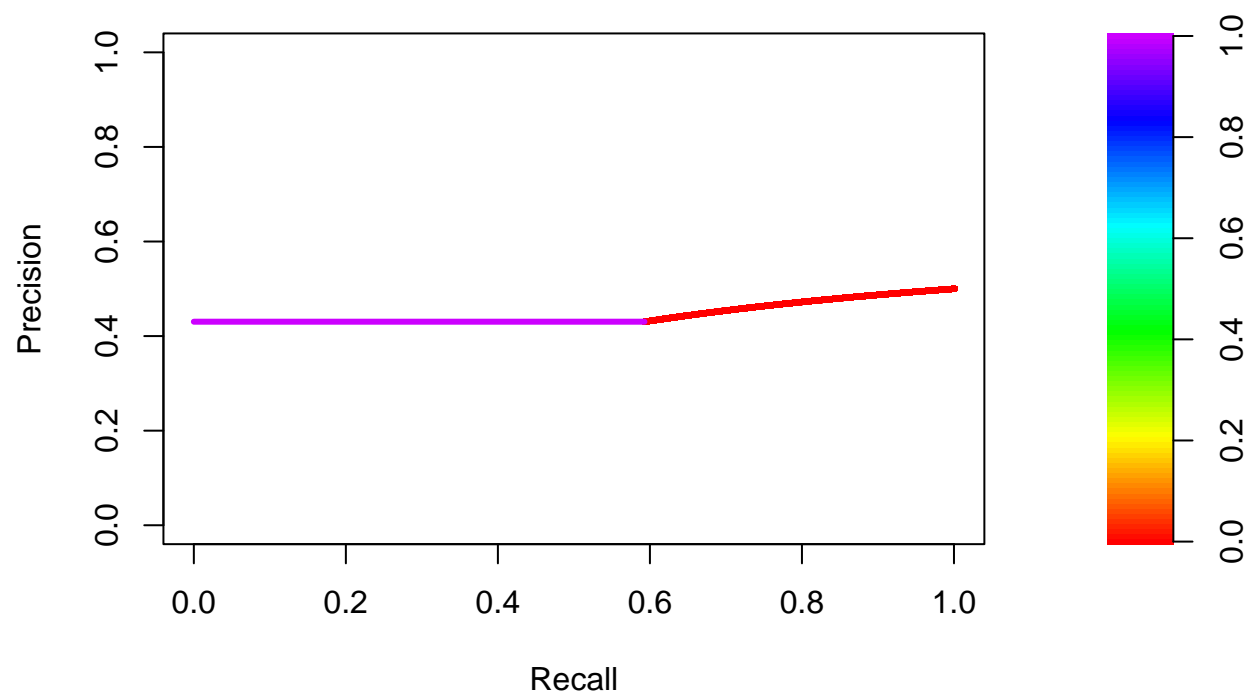
Precision-Recall Curve: Decision Tree – low
AUC = 0.491878



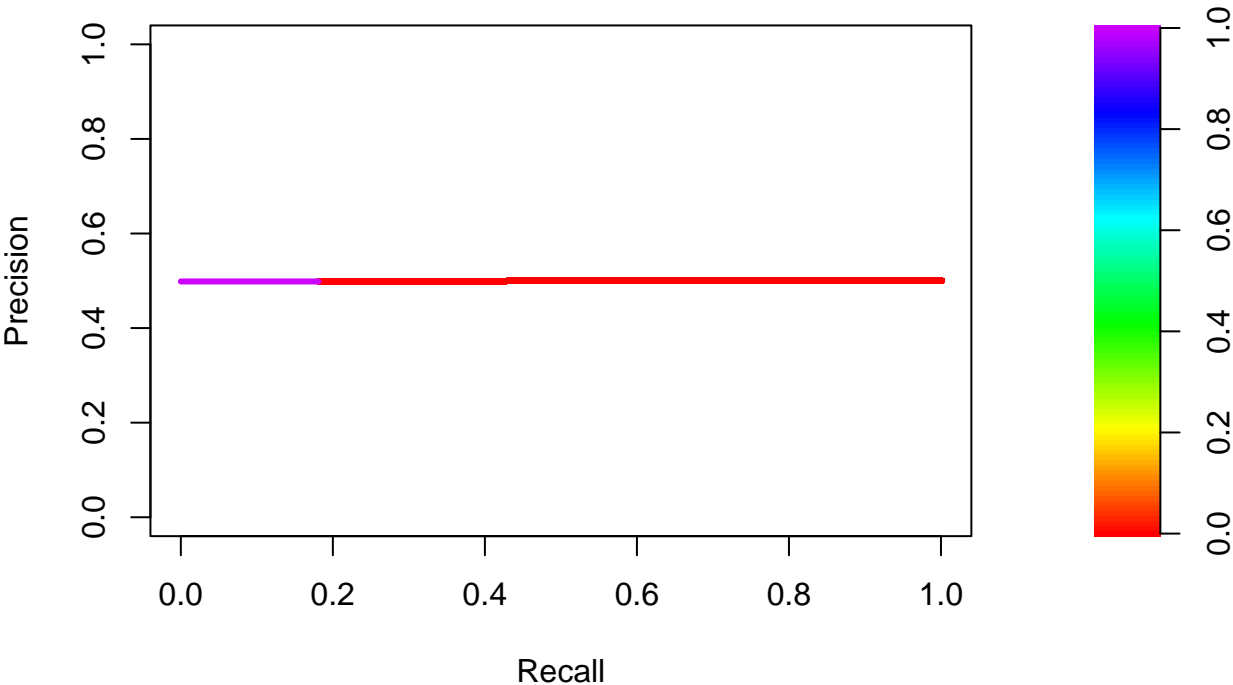
Precision-Recall Curve: Decision Tree – medium
AUC = 0.6505603



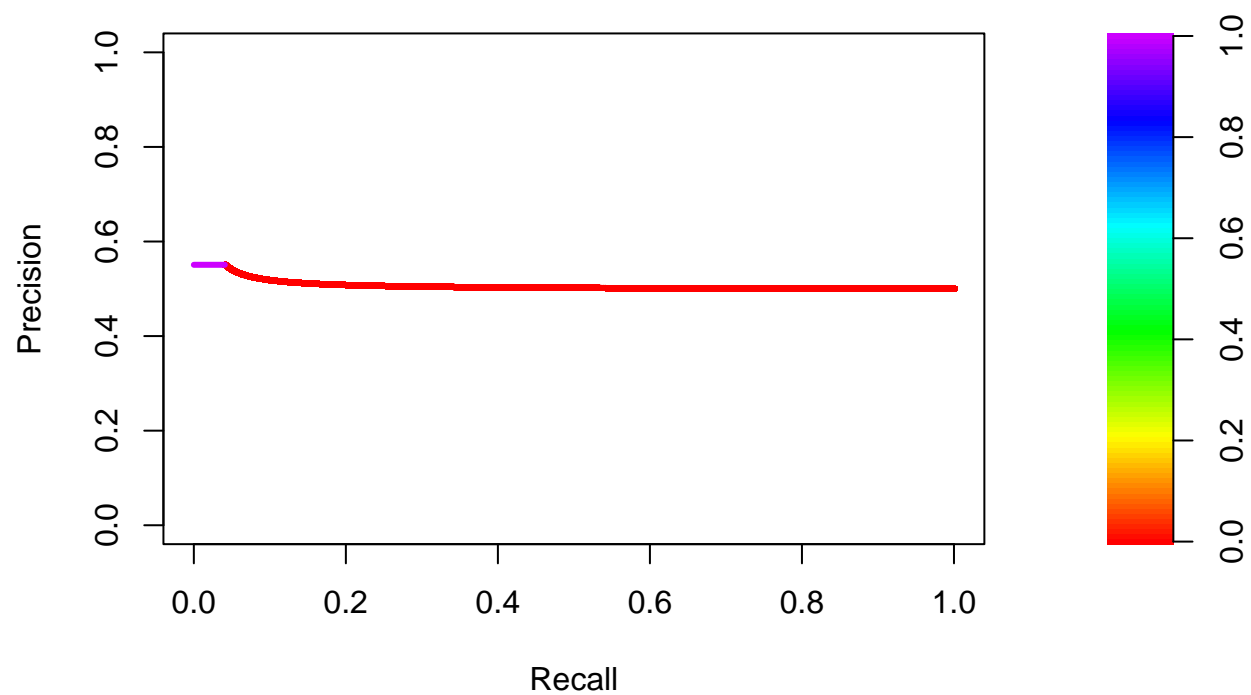
**Precision–Recall Curve: Decision Tree – high
AUC = 0.4463166**



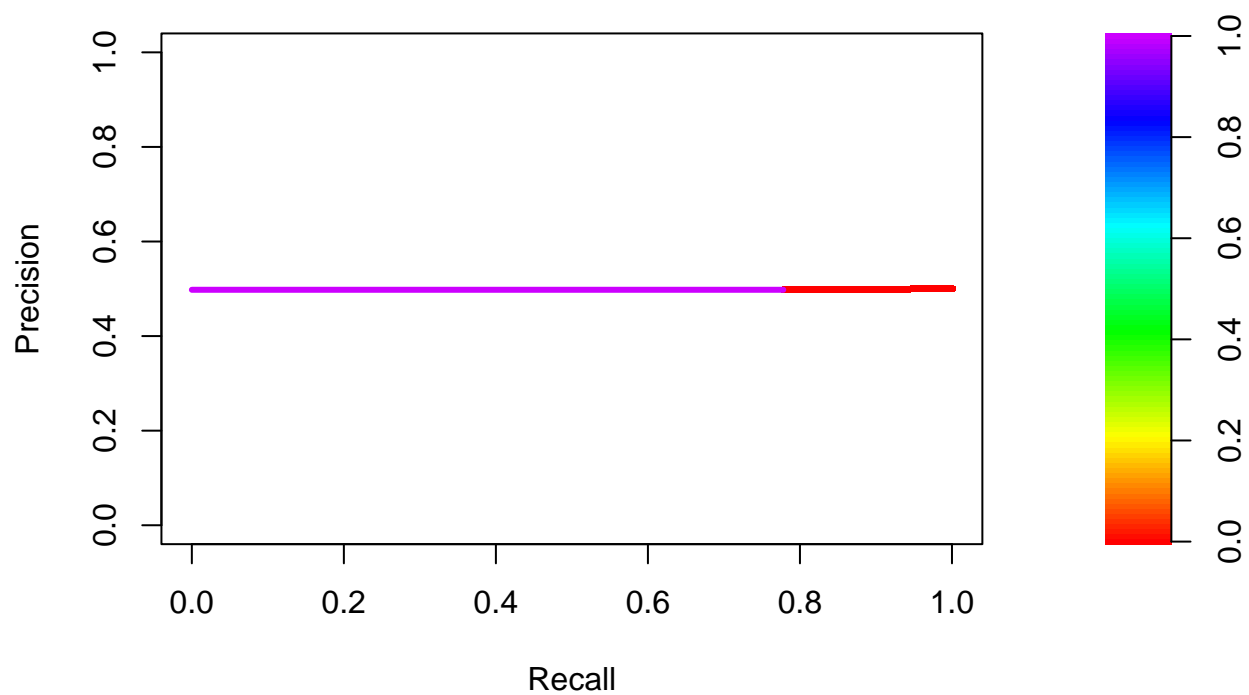
Precision–Recall Curve: Random Forest – low
AUC = 0.4994589



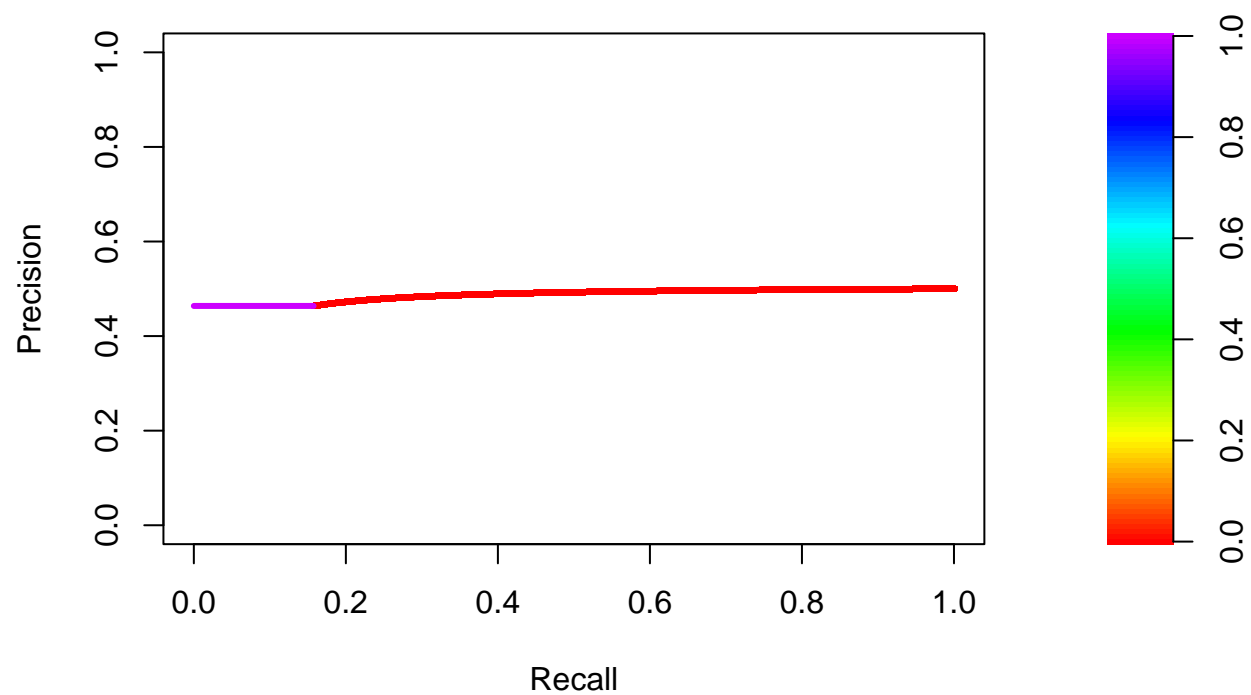
Precision–Recall Curve: Random Forest – medium
AUC = 0.5066009



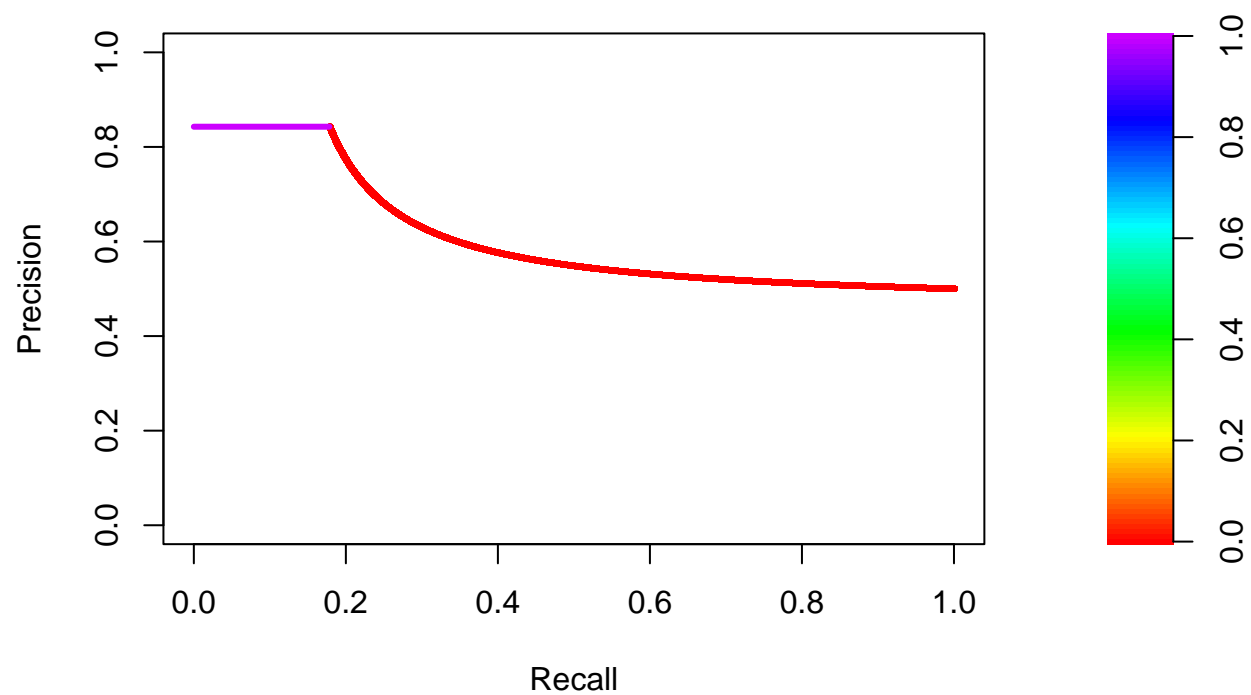
Precision–Recall Curve: Random Forest – high
AUC = 0.4981779



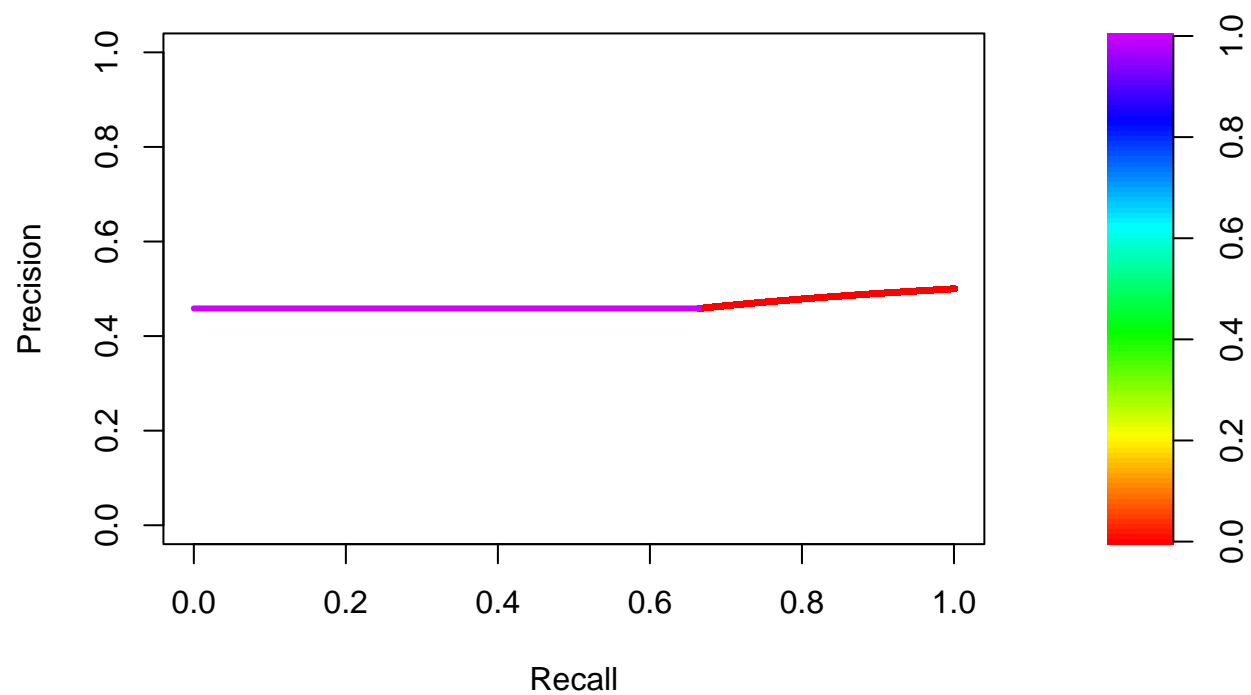
Precision–Recall Curve: SVM – low
AUC = 0.4871987



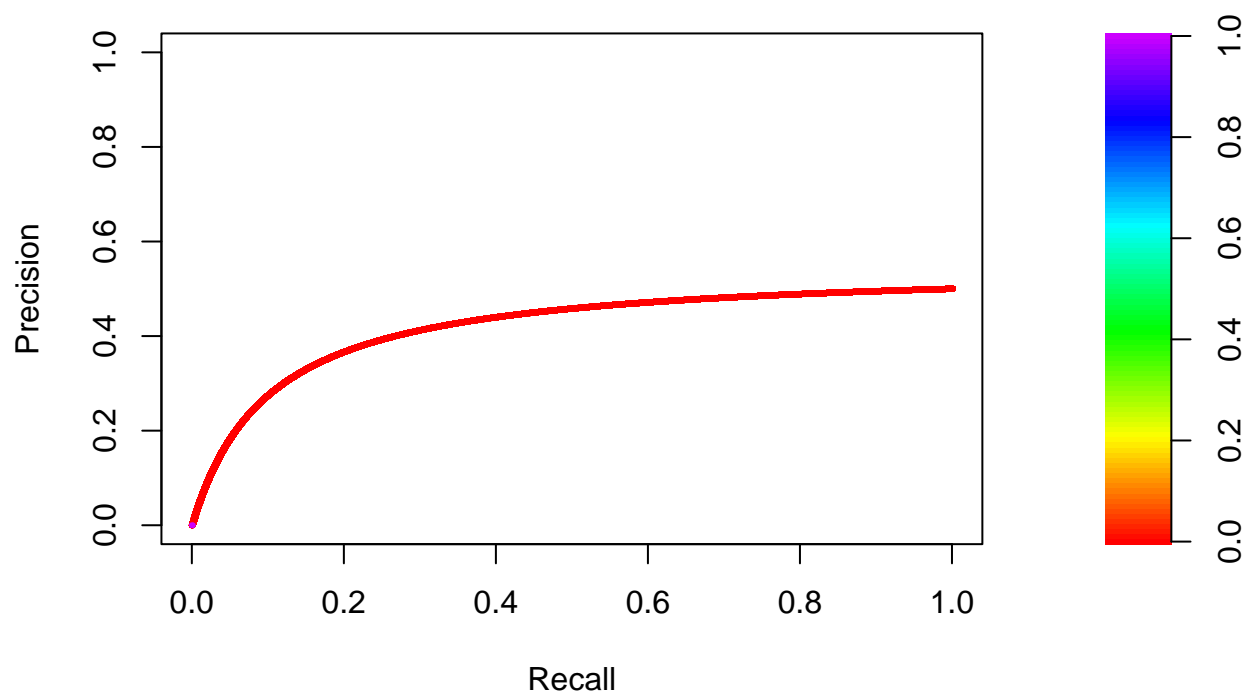
Precision-Recall Curve: SVM – medium
AUC = 0.6114759



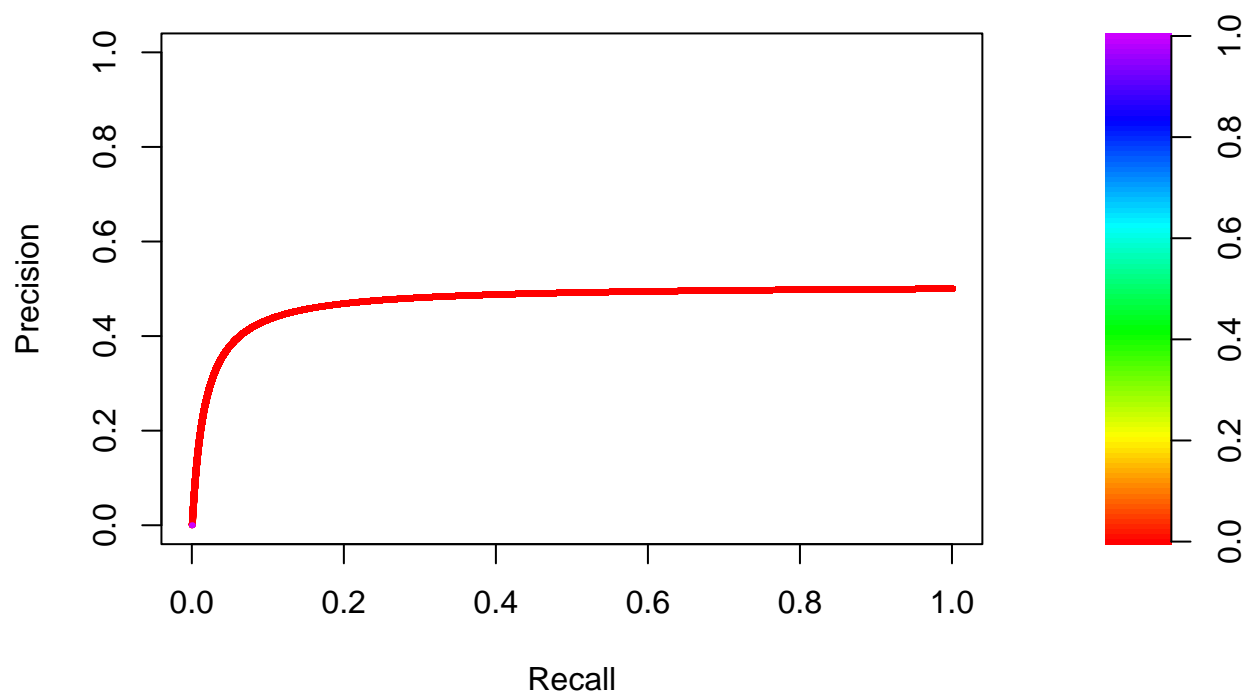
Precision–Recall Curve: SVM – high
AUC = 0.4661779



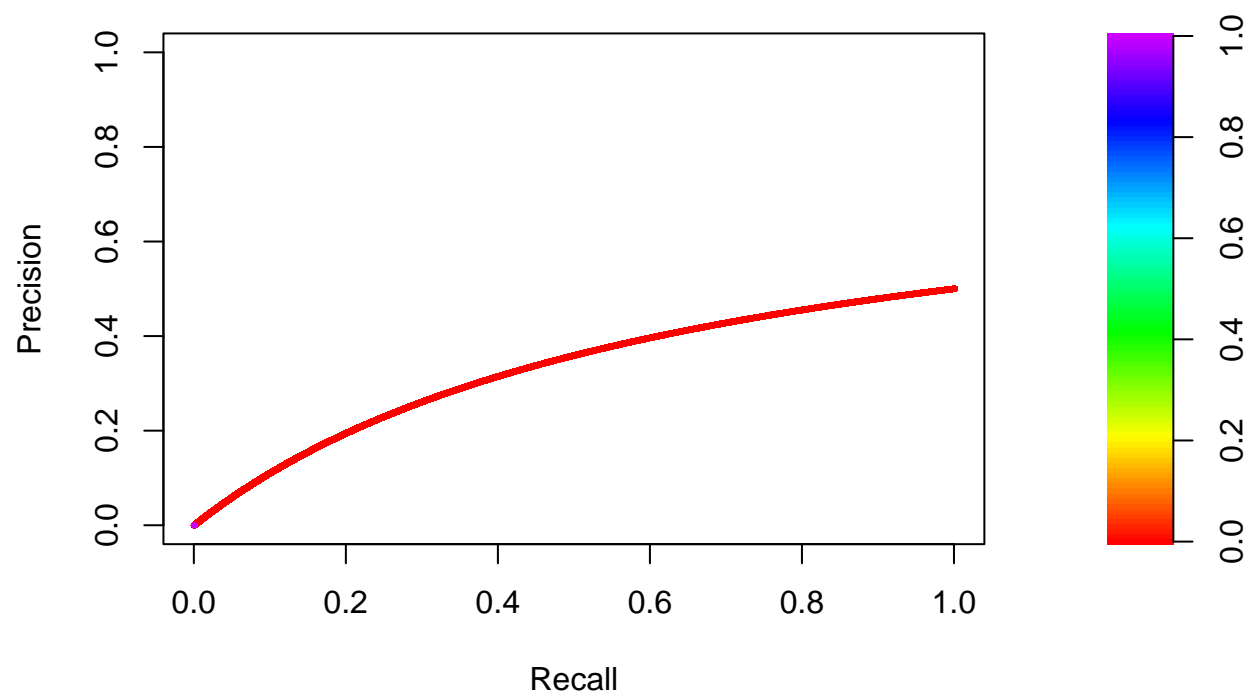
Precision-Recall Curve: Gradient Boosting – low
AUC = 0.4180218



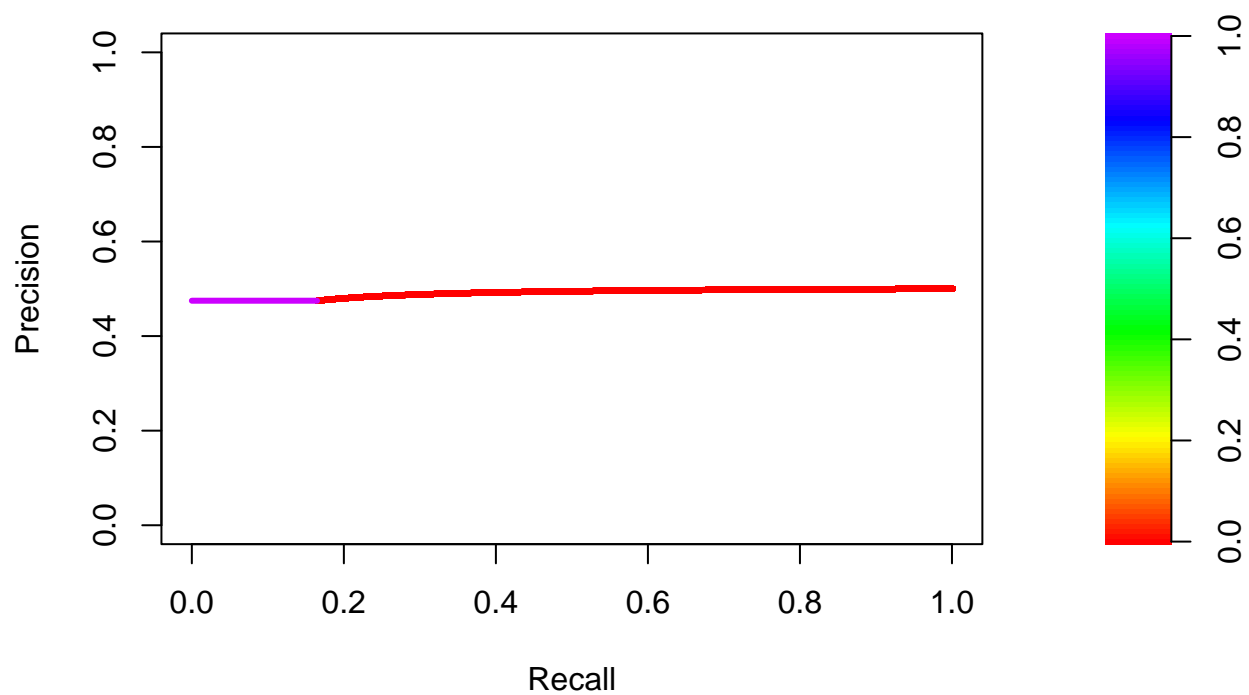
Precision–Recall Curve: Gradient Boosting – medium
AUC = 0.4732341



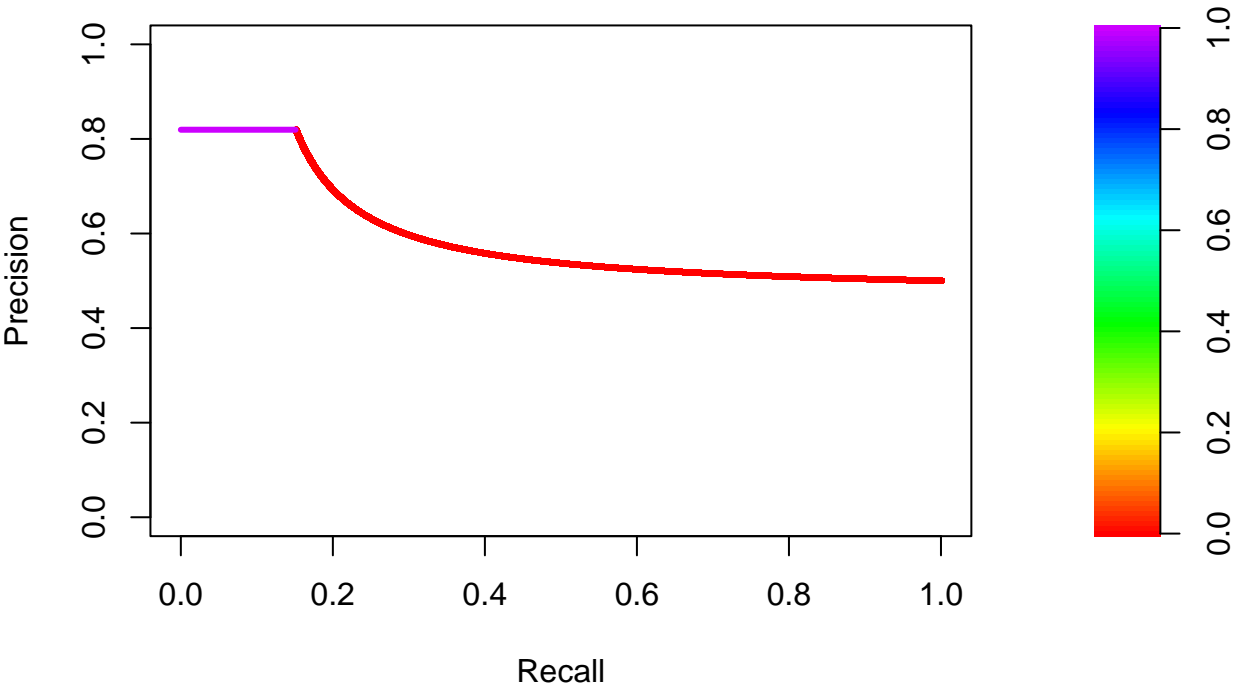
**Precision–Recall Curve: Gradient Boosting – high
AUC = 0.3257733**



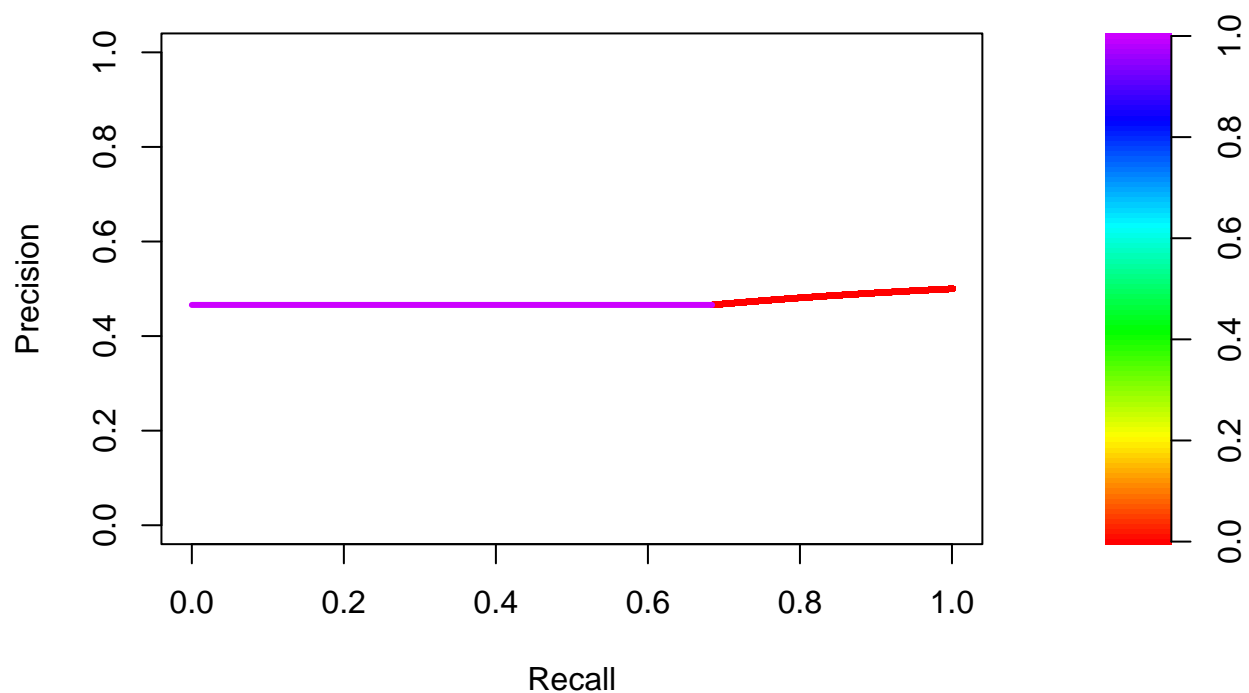
Precision–Recall Curve: Logistic Regression – low
AUC = 0.4908947



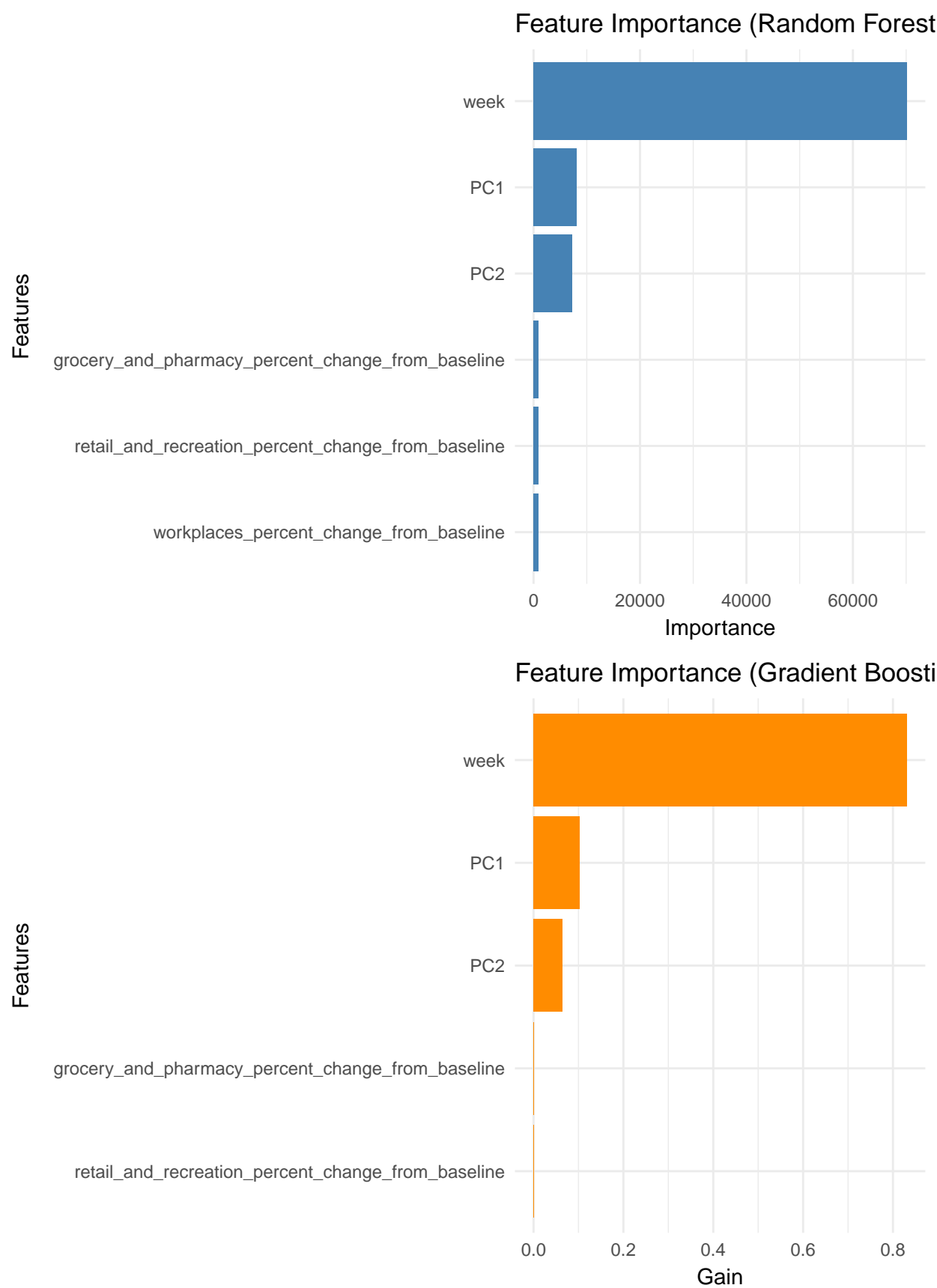
Precision–Recall Curve: Logistic Regression – medium
AUC = 0.5929723



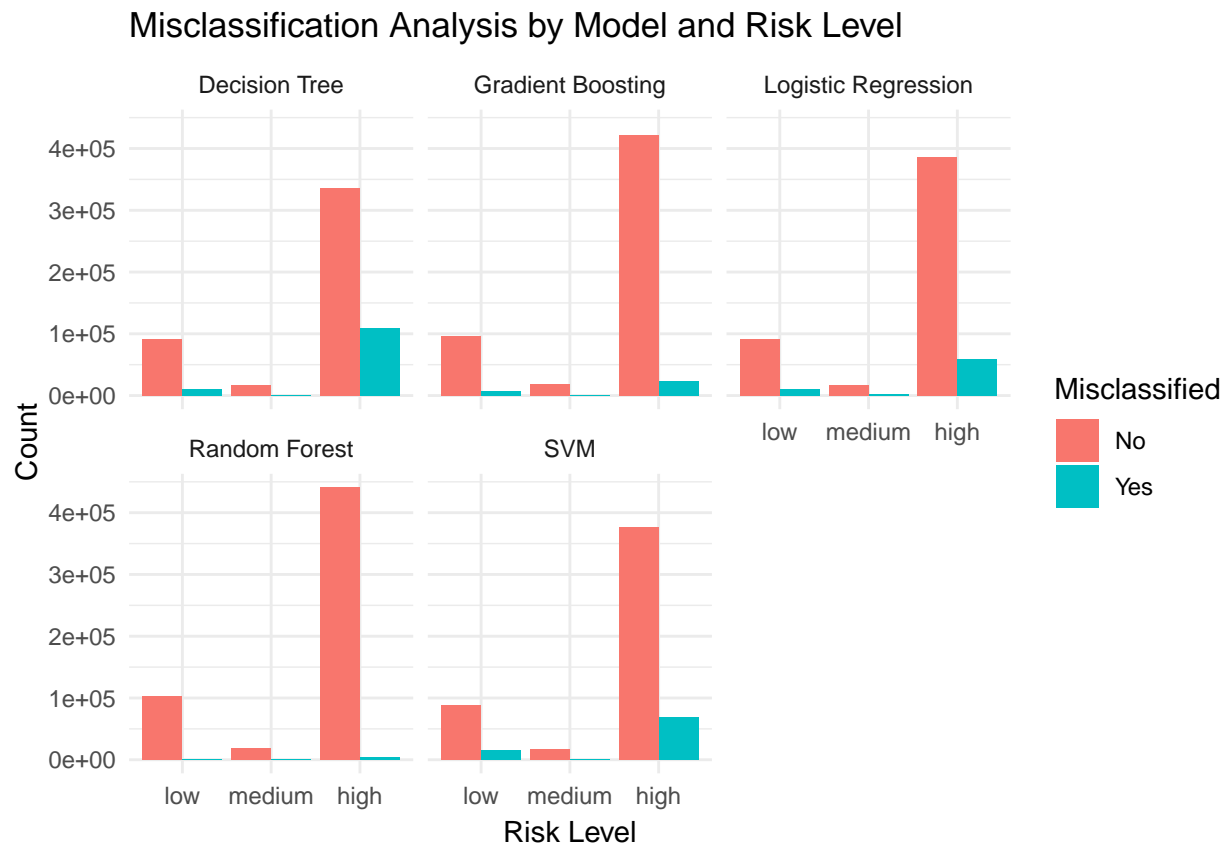
**Precision–Recall Curve: Logistic Regression – high
AUC = 0.4718362**



0.5.5 4. Feature Importance



0.5.6 5. Misclassification Analysis



0.5.7 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.

0.6 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.

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

```

## Model saved to: dt_model_balanced.rds

save_model(rf_model, "rf_model_balanced.rds")

## Model saved to: rf_model_balanced.rds

save_model(svm_model, "svm_model_balanced.rds")

## Model saved to: svm_model_balanced.rds

save_model(xgb_model, "xgb_model_balanced.rds")

## Model saved to: xgb_model_balanced.rds

save_model(logistic_model, "logistic_model_balanced.rds")

## Model saved to: logistic_model_balanced.rds

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

## Model loaded from: dt_model_balanced.rds

loaded_rf_model <- load_model("rf_model_balanced.rds")

## Model loaded from: rf_model_balanced.rds

loaded_svm_model <- load_model("svm_model_balanced.rds")

## Model loaded from: svm_model_balanced.rds

loaded_xgb_model <- load_model("xgb_model_balanced.rds")

## Model loaded from: xgb_model_balanced.rds

loaded_logistic_model <- load_model("logistic_model_balanced.rds")

## Model loaded from: logistic_model_balanced.rds

```

0.7 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).