Final Project Code

```r
# ------------------------------
# Vincent Johnson
# Joseph Kang
# Catherine Sung
# ------------------------------

# ------------------------------
# Set working directory
# ------------------------------

setwd("C:/Users/Public/Stats 385/Final Project")

# ------------------------------
# Load necessary libraries
# ------------------------------
library(car)
library(caret)
library(cluster)
library(corrplot)
library(dplyr)
library(e1071)
library(factoextra)
library(gbm)
library(ggplot2)
library(glmnet)
library(gridExtra)
library(ISLR2)
library(kableExtra)
library(knitr)
library(leaps)
library(pls)
library(randomForest)
library(tibble)
library(tidyverse)


# ------------------------------
# Read and Merge Data
# ------------------------------
math <- read.csv("math.csv")
por <- read.csv("portuguese.csv")

join_cols <- c("school","sex","age","address","famsize","Pstatus",
               "Medu","Fedu","Mjob","Fjob","reason","nursery","internet")
merged_data <- merge(math, por, by = join_cols, suffixes = c(".math", ".por"))

# ------------------------------
# Clean and Format Data
# ------------------------------
clean_data <- merged_data %>%
  rename(mathp1 = G1.math,
         mathp2 = G2.math,
         mathfinal = G3.math,
         porp1 = G1.por,
         porp2 = G2.por,
         porfinal = G3.por) %>%
```

```r
  select(starts_with("G"), everything()) %>%
  mutate_if(is.character, as.factor)

clean_data <- clean_data %>%
  select(-ends_with(".math"))
names(clean_data) <- gsub("\\.por$", "", names(clean_data))
colnames(clean_data)
 #we have exactly 36 columns.

# ------------------------------
# Missing Data Check
# ------------------------------
cat("\nMissing data summary:\n")
print(sapply(clean_data, function(x) sum(is.na(x))))
  ##No Missing data!

# ------------------------------
# Exploratory Data Analysis (EDA)
# ------------------------------
##Histograms

hist_math <- ggplot(clean_data, aes(x = mathfinal)) +
  geom_histogram(binwidth = 1, fill = "skyblue", color = "black") +
  theme_minimal() +
  labs(title = "Final Math Grades", x = "Grade", y = "Count")

hist_por <- ggplot(clean_data, aes(x = porfinal)) +
  geom_histogram(binwidth = 1, fill = "salmon", color = "black") +
  theme_minimal() +
  labs(title = "Final Portuguese Grades", x = "Grade", y = "Count")

grid.arrange(hist_math, hist_por, ncol = 2)

##Boxplots
box_math <- ggplot(clean_data, aes(x = sex, y = mathfinal)) +
  geom_boxplot(fill = "lightblue") +
  theme_minimal() +
  labs(title = "Math Grades by Sex", x = "Sex", y = "Math Final Grade")

box_por <- ggplot(clean_data, aes(x = sex, y = porfinal)) +
  geom_boxplot(fill = "lightcoral") +
  theme_minimal() +
  labs(title = "Portuguese Grades by Sex", x = "Sex", y = "Portuguese Final Grade")

grid.arrange(box_math, box_por, ncol = 2)


##QQ Plots
par(mfrow = c(1, 2))
qqnorm(clean_data$mathfinal, main = "QQ Plot - Math")
qqline(clean_data$mathfinal)
qqnorm(clean_data$porfinal, main = "QQ Plot - Portuguese")
qqline(clean_data$porfinal)
par(mfrow = c(1, 1))  # Reset

##Coorelation Heat Map

# Only numeric variables
numeric_data <- clean_data %>% select_if(is.numeric)
```

```r
cor_matrix <- cor(numeric_data, use = "complete.obs")

corrplot(cor_matrix, method = "color", type = "upper",
         title = "Correlation Matrix (Math & Portuguese)", mar = c(0,0,1,0))

summary_table <- tibble(
  Statistic = c("Min", "1st Quartile", "Median", "Mean", "3rd Quartile", "Max", "SD"),
  Math = c(
    min(clean_data$mathfinal, na.rm = TRUE),
    quantile(clean_data$mathfinal, 0.25, na.rm = TRUE),
    median(clean_data$mathfinal, na.rm = TRUE),
    mean(clean_data$mathfinal, na.rm = TRUE),
    quantile(clean_data$mathfinal, 0.75, na.rm = TRUE),
    max(clean_data$mathfinal, na.rm = TRUE),
    sd(clean_data$mathfinal, na.rm = TRUE)
  ),
  Portuguese = c(
    min(clean_data$porfinal, na.rm = TRUE),
    quantile(clean_data$porfinal, 0.25, na.rm = TRUE),
    median(clean_data$porfinal, na.rm = TRUE),
    mean(clean_data$porfinal, na.rm = TRUE),
    quantile(clean_data$porfinal, 0.75, na.rm = TRUE),
    max(clean_data$porfinal, na.rm = TRUE),
    sd(clean_data$porfinal, na.rm = TRUE)
  )
)

print(summary_table)

kable(summary_table, format = "latex", booktabs = TRUE,
      caption = "Model Performance Summary for Math and Portuguese")

# Statistic     Math Portuguese
# 1 Min           0          0
# 2 1st Quartile  8         11
# 3 Median       11         13
# 4 Mean         10.4       12.5
# 5 3rd Quartile 14         14
# 6 Max          20         19
# 7 SD            4.69       2.95


# ------------------------------
# Subset and split Data
# ------------------------------
math_data <- clean_data  %>% mutate_if(is.character, as.factor) %>%
  select(-contains("por"))

por_data <- clean_data  %>% mutate_if(is.character, as.factor) %>%
  select(-contains("math"))

set.seed(38520251)
train_index <- createDataPartition(math_data$mathfinal, p = 0.7, list = FALSE)
train <- math_data[train_index, ]
test <- math_data[-train_index, ]

set.seed(38520251)
train_indexp <- createDataPartition(por_data$porfinal, p = 0.7, list = FALSE)
trainp <- por_data[train_indexp, ]
```

```r
testp <- por_data[-train_indexp, ]



# ------------------------------
# Linear Models + Assumption Checks
# ------------------------------

# ---------- Linear Model ----------
mlr_model <- lm(mathfinal ~ ., data = train)
summary(mlr_model)

plr_model <- lm(porfinal ~ ., data = trainp)
summary(plr_model)

# Assumption Checks - Math
par(mfrow = c(2, 2))  # Diagnostic plots
plot(mlr_model)       # Includes: Residuals vs Fitted, QQ plot, Scale-Location, Residuals vs Leverage

# Assumption Checks - Portuguese
par(mfrow = c(2, 2))
plot(plr_model)


# Convert VIF outputs to data frames with variable names
vif_math <- as.data.frame(vif(mlr_model)) %>%
  rownames_to_column(var = "Variable") %>%
  rename_with(~ paste0("Math_", .), -Variable)

vif_por <- as.data.frame(vif(plr_model)) %>%
  rownames_to_column(var = "Variable") %>%
  rename_with(~ paste0("Portuguese_", .), -Variable)

# Merge the two VIF tables by variable name
vif_comparison <- full_join(vif_math, vif_por, by = "Variable")

# Print the comparison table
print(vif_comparison)

kable(vif_comparison, format = "latex", booktabs = TRUE,
      caption = "Side-by-Side VIF Comparison for Math and Portuguese Linear Models")
# Variable Math_GVIF Math_Df Math_GVIF^(1/(2*Df)) Portuguese_GVIF Portuguese_Df Portuguese_GVIF^(1/(2*Df))
# 1    guardian  1.480193       2             1.103010        1.571115             2                    1.119572
# 2       goout  1.608989       1             1.268459        1.681249             1                    1.296630
# 3      school  1.641374       1             1.281161        1.614344             1                    1.270569
# 4         sex  1.480817       1             1.216888        1.496284             1                    1.223227
# 5         age  1.683393       1             1.297456        1.633526             1                    1.278095
# 6     address  1.434334       1             1.197637        1.520429             1                    1.233057
# 7     famsize  1.155246       1             1.074824        1.248482             1                    1.117355
```

```
# 8      Pstatus  1.189486        1              1.090636         1.220292              1
                          1.104668
# 9         Medu  3.225022        1              1.795835         3.197912              1
                          1.788271
# 10        Fedu  2.256879        1              1.502291         2.371832              1
                          1.540075
# 11        Mjob  4.711666        4              1.213799         4.768852              4
                          1.215631
# 12        Fjob  2.917813        4              1.143226         3.076145              4
                          1.150803
# 13      reason  1.812125        3              1.104158         1.901376              3
                          1.113041
# 14     nursery  1.247515        1              1.116922         1.200685              1
                          1.095758
# 15    internet  1.282259        1              1.132369         1.337381              1
                          1.156452
# 16      mathp1  5.567227        1              2.359497               NA             NA
                               NA
# 17      mathp2  5.443060        1              2.333037               NA             NA
                               NA
# 18 traveltime  1.374866        1              1.172547         1.388562              1
                          1.178373
# 19  studytime  1.453808        1              1.205740         1.601372              1
                          1.265453
# 20   failures  1.475491        1              1.214698         1.495908              1
                          1.223073
# 21  schoolsup  1.395075        1              1.181133         1.401559              1
                          1.183875
# 22     famsup  1.250634        1              1.118317         1.199915              1
                          1.095406
# 23       paid  1.233115        1              1.110457         1.251845              1
                          1.118859
# 24 activities  1.209743        1              1.099883         1.303474              1
                          1.141698
# 25     higher  1.367377        1              1.169349         1.432669              1
                          1.196942
# 26   romantic  1.278935        1              1.130900         1.262127              1
                          1.123444
# 27     famrel  1.220360        1              1.104699         1.226586              1
                          1.107513
# 28   freetime  1.448505        1              1.203539         1.338110              1
                          1.156767
# 29       Dalc  1.941432        1              1.393353         2.008371              1
                          1.417170
# 30       Walc  2.239369        1              1.496452         2.403011              1
                          1.550165
# 31     health  1.295174        1              1.138057         1.351867              1
                          1.162698
# 32   absences  1.260738        1              1.122826         1.312955              1
                          1.145842
# 33      porp1        NA       NA                    NA         5.218640              1
                          2.284434
# 34      porp2        NA       NA                    NA         5.504888              1
                          2.346250


# ---------- Additional Residual Analysis - Math ----------

par(mfrow = c(1, 2))
```

```r

residuals_math <- resid(mlr_model)
shapiro.test(residuals_math)  # Normality test for Math residuals
plot(fitted(mlr_model), residuals_math,
     main = "Residuals vs Fitted - Math",
     xlab = "Fitted values", ylab = "Residuals")
abline(h = 0, col = "red")  # Red line at 0 for reference

# ---------- Additional Residual Analysis - Portuguese ----------
residuals_por <- resid(plr_model)
shapiro.test(residuals_por)  # Normality test for Portuguese residuals
plot(fitted(plr_model), residuals_por,
     main = "Residuals vs Fitted - Portuguese",
     xlab = "Fitted values", ylab = "Residuals")
abline(h = 0, col = "red")  # Red line at 0 for reference

# Reset layout to 1 plot per row (optional, for future plotting)
par(mfrow = c(1, 1))



# Reset plot layout
par(mfrow = c(1,1))

# ------------------------------
# Stepwise Model Selection (AIC, BIC, Forward, Backward)
# ------------------------------
step_aic <- step(mlr_model, direction = "both", trace = 0)
summary(step_aic)
step_aicp <- step(plr_model, direction = "both", trace = 0)
summary(step_aicp)

step_bic <- step(mlr_model, direction = "both", k = log(nrow(train)), trace = 0)
summary(step_bic)
step_bicp <- step(plr_model, direction = "both", k = log(nrow(train)), trace = 0)
summary(step_bicp)

forward <- step(lm(mathfinal ~ 1, data = train), scope = formula(mlr_model), direction
     = "forward")
summary(forward)
forwardp <- step(lm(porfinal ~ 1, data = trainp), scope = formula(plr_model),
    direction = "forward")
summary(forwardp)

backward <- step(mlr_model, direction = "backward")
summary(backward)
backwardp <- step(plr_model, direction = "backward")
summary(backwardp)


##Comparisons

# Helper to extract model summary info
get_model_info <- function(model) {
  data.frame(
    Variables = paste(names(coef(model))[-1], collapse = ", "),
    R2 = summary(model)$r.squared,
    Adj_R2 = summary(model)$adj.r.squared,
    AIC = AIC(model),
```

```
    BIC = BIC(model)
  )
}

# Create comparison table
model_comparison <- tibble::tibble(
  Method = c("AIC", "BIC", "Forward", "Backward"),

  Math_Vars = c(
    get_model_info(step_aic)$Variables,
    get_model_info(step_bic)$Variables,
    get_model_info(forward)$Variables,
    get_model_info(backward)$Variables
  ),
  Math_R2 = c(
    get_model_info(step_aic)$R2,
    get_model_info(step_bic)$R2,
    get_model_info(forward)$R2,
    get_model_info(backward)$R2
  ),
  Math_AdjR2 = c(
    get_model_info(step_aic)$Adj_R2,
    get_model_info(step_bic)$Adj_R2,
    get_model_info(forward)$Adj_R2,
    get_model_info(backward)$Adj_R2
  ),
  Math_AIC = c(
    get_model_info(step_aic)$AIC,
    get_model_info(step_bic)$AIC,
    get_model_info(forward)$AIC,
    get_model_info(backward)$AIC
  ),
  Math_BIC = c(
    get_model_info(step_aic)$BIC,
    get_model_info(step_bic)$BIC,
    get_model_info(forward)$BIC,
    get_model_info(backward)$BIC
  ),

  Port_Vars = c(
    get_model_info(step_aicp)$Variables,
    get_model_info(step_bicp)$Variables,
    get_model_info(forwardp)$Variables,
    get_model_info(backwardp)$Variables
  ),
  Port_R2 = c(
    get_model_info(step_aicp)$R2,
    get_model_info(step_bicp)$R2,
    get_model_info(forwardp)$R2,
    get_model_info(backwardp)$R2
  ),
  Port_AdjR2 = c(
    get_model_info(step_aicp)$Adj_R2,
    get_model_info(step_bicp)$Adj_R2,
    get_model_info(forwardp)$Adj_R2,
    get_model_info(backwardp)$Adj_R2
  ),
  Port_AIC = c(
    get_model_info(step_aicp)$AIC,
```

```
    get_model_info(step_bicp)$AIC,
    get_model_info(forwardp)$AIC,
    get_model_info(backwardp)$AIC
  ),
  Port_BIC = c(
    get_model_info(step_aicp)$BIC,
    get_model_info(step_bicp)$BIC,
    get_model_info(forwardp)$BIC,
    get_model_info(backwardp)$BIC
  )
)

print(model_comparison)



# Save LaTeX table as a string
latex_table <- kable(model_comparison, format = "latex", booktabs = TRUE,
                     caption = "Stepwise Model Comparison for Math and Portuguese")

# Write to .tex file
writeLines(latex_table, "model_comparison_table.tex")

# ------------------------------
# Lasso Regression - Math
# ------------------------------
x <- model.matrix(mathfinal ~ ., train)[, -1]
y <- train$mathfinal
lasso_mod <- cv.glmnet(x, y, alpha = 1)
best_lambda <- lasso_mod$lambda.min
lasso_final <- glmnet(x, y, alpha = 1, lambda = best_lambda)
coef(lasso_final)

# ------------------------------
# Lasso Regression - Portuguese
# ------------------------------
xp <- model.matrix(porfinal ~ ., trainp)[, -1]
yp <- trainp$porfinal
lasso_modp <- cv.glmnet(xp, yp, alpha = 1)
best_lambdap <- lasso_modp$lambda.min
lasso_finalp <- glmnet(xp, yp, alpha = 1, lambda = best_lambdap)
coef(lasso_finalp)


# Get non-zero coefficients
coef_math <- coef(lasso_final)
coef_por <- coef(lasso_finalp)

# Convert to data frames
coef_math_df <- as.data.frame(as.matrix(coef_math))
coef_math_df <- tibble::rownames_to_column(coef_math_df, var = "Variable")
colnames(coef_math_df)[2] <- "Math_Coefficient"

coef_por_df <- as.data.frame(as.matrix(coef_por))
coef_por_df <- tibble::rownames_to_column(coef_por_df, var = "Variable")
colnames(coef_por_df)[2] <- "Portuguese_Coefficient"

# Merge by variable name
lasso_compare <- full_join(coef_math_df, coef_por_df, by = "Variable")
```

```r

# Keep only coefficients that are non-zero in at least one model
lasso_compare_filtered <- lasso_compare %>%
  filter(Math_Coefficient != 0 | Portuguese_Coefficient != 0)



kable(lasso_compare_filtered, format = "latex", booktabs = TRUE,
      caption = "Lasso Coefficients for Math and Portuguese Final Grades") %>%
  kable_styling(latex_options = c("striped", "hold_position"))



# ------------------------------
# Bagging and Boosting Math
# ------------------------------
set.seed(38520251)
bag_mod <- randomForest(mathfinal ~ ., data = train, mtry = ncol(train) - 1)
boost_mod <- gbm(mathfinal ~ ., data = train, distribution = "gaussian", n.trees =
    1000, interaction.depth = 4)

# ------------------------------
# Bagging and Boosting Portugese
# ------------------------------
set.seed(38520251)
bag_modp <- randomForest(porfinal ~ ., data = trainp, mtry = ncol(trainp) - 1)
boost_modp <- gbm(porfinal ~ ., data = trainp, distribution = "gaussian", n.trees =
    1000, interaction.depth = 4)

# Math Model Predictions
pred_bag <- predict(bag_mod, newdata = test)
pred_boost <- predict(boost_mod, newdata = test, n.trees = 1000)

# Portuguese Model Predictions
pred_bagp <- predict(bag_modp, newdata = testp)
pred_boostp <- predict(boost_modp, newdata = testp, n.trees = 1000)

# RMSE Calculation Function
rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted)^2))
}

# RMSE for Math Models
rmse_bag_math <- rmse(test$mathfinal, pred_bag)
rmse_boost_math <- rmse(test$mathfinal, pred_boost)

# RMSE for Portuguese Models
rmse_bag_por <- rmse(testp$porfinal, pred_bagp)
rmse_boost_por <- rmse(testp$porfinal, pred_boostp)

# Collecting RMSE into a summary table
bagging_boosting_summary <- data.frame(
  Model = c("Bagging (Math)", "Boosting (Math)", "Bagging (Portuguese)", "Boosting (
      Portuguese)"),
  RMSE = c(rmse_bag_math, rmse_boost_math, rmse_bag_por, rmse_boost_por)
)

# Print the summary table
print(bagging_boosting_summary)
```

```r

kable(bagging_boosting_summary, format = "latex", booktabs = TRUE,
      caption = "Model Comparison: Bagging and Boosting (Math and Portuguese)")


# ------------------------------
# Principal Component Regression (PCR) Math
# ------------------------------
pcr_mod <- pcr(mathfinal ~ ., data = train, scale = TRUE, validation = "CV")
summary(pcr_mod)
validationplot(pcr_mod, val.type = "MSEP")

# ------------------------------
# Principal Component Regression (PCR) Portugese
# ------------------------------
pcr_modp <- pcr(porfinal ~ ., data = trainp, scale = TRUE, validation = "CV")
summary(pcr_modp)
validationplot(pcr_modp, val.type = "MSEP")

# ------------------------------
# Partial Least Squares Regression (PLS) Math
# ------------------------------
pls_mod <- plsr(mathfinal ~ ., data = train, scale = TRUE, validation = "CV")
summary(pls_mod)
validationplot(pls_mod, val.type = "MSEP")

# ------------------------------
# Partial Least Squares Regression (PLS) Portugese
# ------------------------------
pls_modp <- plsr(porfinal ~ ., data = trainp, scale = TRUE, validation = "CV")
summary(pls_modp)
validationplot(pls_modp, val.type = "MSEP")


# Optimal number of components for PCR and PLS (Math)
opt_comp_pcr_math <- which.min(MSEP(pcr_mod)$val[1, , ])
opt_comp_pls_math <- which.min(MSEP(pls_mod)$val[1, , ])

# Optimal number of components for PCR and PLS (Portuguese)
opt_comp_pcr_por <- which.min(MSEP(pcr_modp)$val[1, , ])
opt_comp_pls_por <- which.min(MSEP(pls_modp)$val[1, , ])

#Predict using Optimal Components

# Math
pred_pcr <- predict(pcr_mod, newdata = test, ncomp = opt_comp_pcr_math)
pred_pls <- predict(pls_mod, newdata = test, ncomp = opt_comp_pls_math)

# Portuguese
pred_pcrp <- predict(pcr_modp, newdata = testp, ncomp = opt_comp_pcr_por)
pred_plsp <- predict(pls_modp, newdata = testp, ncomp = opt_comp_pls_por)

# RMSE
rmse_pcr_math <- rmse(test$mathfinal, pred_pcr)
rmse_pcr_por <- rmse(testp$porfinal, pred_pcrp)
rmse_pls_math <- rmse(test$mathfinal, pred_pls)
rmse_pls_por <- rmse(testp$porfinal, pred_plsp)

# Summary table
```

```r
pcr_pls_summary <- data.frame(
  Model = c("PCR (Math)", "PCR (Portuguese)", "PLS (Math)", "PLS (Portuguese)"),
  Optimal_Components = c(opt_comp_pcr_math, opt_comp_pcr_por, opt_comp_pls_math, opt_
      comp_pls_por),
  RMSE = c(rmse_pcr_math, rmse_pcr_por, rmse_pls_math, rmse_pls_por)
)

kable(pcr_pls_summary, format = "latex", booktabs = TRUE,
      caption = "PCR and PLS Model Comparison Using Optimal Number of Components")

# ------------------------------
# Random Forest - Math
# ------------------------------
rf_mod <- randomForest(mathfinal ~ ., data = train, importance = TRUE)
print(rf_mod)
varImpPlot(rf_mod)

# ------------------------------
# Random Forest - Portuguese
# ------------------------------
rf_modp <- randomForest(porfinal ~ ., data = trainp, importance = TRUE)
print(rf_modp)
varImpPlot(rf_modp)

# ------------------------------
# Support Vector Machine (SVM) math
# ------------------------------
svm_mod <- svm(mathfinal ~ ., data = train)
pred_svm <- predict(svm_mod, newdata = test)

# ------------------------------
# Support Vector Machine (SVM) Portugese
# ------------------------------
svm_modp <- svm(porfinal ~ ., data = trainp)
pred_svmp <- predict(svm_modp, newdata = testp)

# Define RMSE function (if not already)
rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted)^2))
}


# Predictions for Random Forest
pred_rf <- predict(rf_mod, newdata = test)
pred_rfp <- predict(rf_modp, newdata = testp)

# RMSE for Random Forest
rmse_rf_math <- rmse(test$mathfinal, pred_rf)
rmse_rf_por <- rmse(testp$porfinal, pred_rfp)

# RMSE for SVM
rmse_svm_math <- rmse(test$mathfinal, pred_svm)
rmse_svm_por <- rmse(testp$porfinal, pred_svmp)

rf_svm_summary <- data.frame(
  Model = c("Random Forest (Math)", "Random Forest (Portuguese)",
            "SVM (Math)", "SVM (Portuguese)"),
  RMSE = c(rmse_rf_math, rmse_rf_por, rmse_svm_math, rmse_svm_por)
)
```

```r

kable(rf_svm_summary, format = "latex", booktabs = TRUE,
      caption = "Model Comparison: Random Forest and SVM for Math and Portuguese")
# ------------------------------
# Model Evaluation - Math
# ------------------------------
pred_lm <- predict(mlr_model, newdata = test)
pred_rf <- predict(rf_mod, newdata = test)
pred_pcr <- predict(pcr_mod, newdata = test, ncomp = 5)
pred_pls <- predict(pls_mod, newdata = test, ncomp = 5)
pred_bag <- predict(bag_mod, newdata = test)
pred_boost <- predict(boost_mod, newdata = test, n.trees = 1000)

rmse <- function(actual, predicted) {
  sqrt(mean((actual - predicted)^2))
}
r2 <- function(actual, predicted) {
  cor(actual, predicted)^2
}

# RMSE & R   for Math
math_metrics <- data.frame(
  Model = names(math_rmse),
  RMSE = round(math_rmse, 3),
  R2 = round(c(
    r2(test$mathfinal, pred_lm),
    r2(test$mathfinal, predict(step_aic, newdata = test)),
    r2(test$mathfinal, predict(step_bic, newdata = test)),
    r2(test$mathfinal, predict(forward, newdata = test)),
    r2(test$mathfinal, predict(backward, newdata = test)),
    r2(test$mathfinal, predict(lasso_final, s = best_lambda, newx = model.matrix(test$
        mathfinal ~ ., test)[, -1])),
    r2(test$mathfinal, pred_pcr),
    r2(test$mathfinal, pred_pls),
    r2(test$mathfinal, pred_bag),
    r2(test$mathfinal, pred_boost),
    r2(test$mathfinal, pred_rf),
    r2(test$mathfinal, pred_svm)
  ), 3)
)

# RMSE & R   for Portuguese
portugese_metrics <- data.frame(
  Model = names(portugese_rmse),
  RMSE = round(portugese_rmse, 3),
  R2 = round(c(
    r2(testp$porfinal, pred_lmp),
    r2(testp$porfinal, predict(step_aicp, newdata = testp)),
    r2(testp$porfinal, predict(step_bicp, newdata = testp)),
    r2(testp$porfinal, predict(forwardp, newdata = testp)),
    r2(testp$porfinal, predict(backwardp, newdata = testp)),
    r2(testp$porfinal, predict(lasso_finalp, s = best_lambdap, newx = model.matrix(
        testp$porfinal ~ ., testp)[, -1])),
    r2(testp$porfinal, pred_pcrp),
    r2(testp$porfinal, pred_plsp),
    r2(testp$porfinal, pred_bagp),
    r2(testp$porfinal, pred_boostp),
    r2(testp$porfinal, pred_rfp),
    r2(testp$porfinal, pred_svmp)
```

```
  ), 3)
)

##Combine into final table

final_metrics <- data.frame(
  Model = math_metrics$Model,
  RMSE_Math = math_metrics$RMSE,
  R2_Math = math_metrics$R2,
  RMSE_Portuguese = portugese_metrics$RMSE,
  R2_Portuguese = portugese_metrics$R2
)

# Latex
kable(final_metrics, format = "latex", booktabs = TRUE,
      caption = "Model Performance Comparison: RMSE and R   for Math and Portuguese
          Final Grades")
```