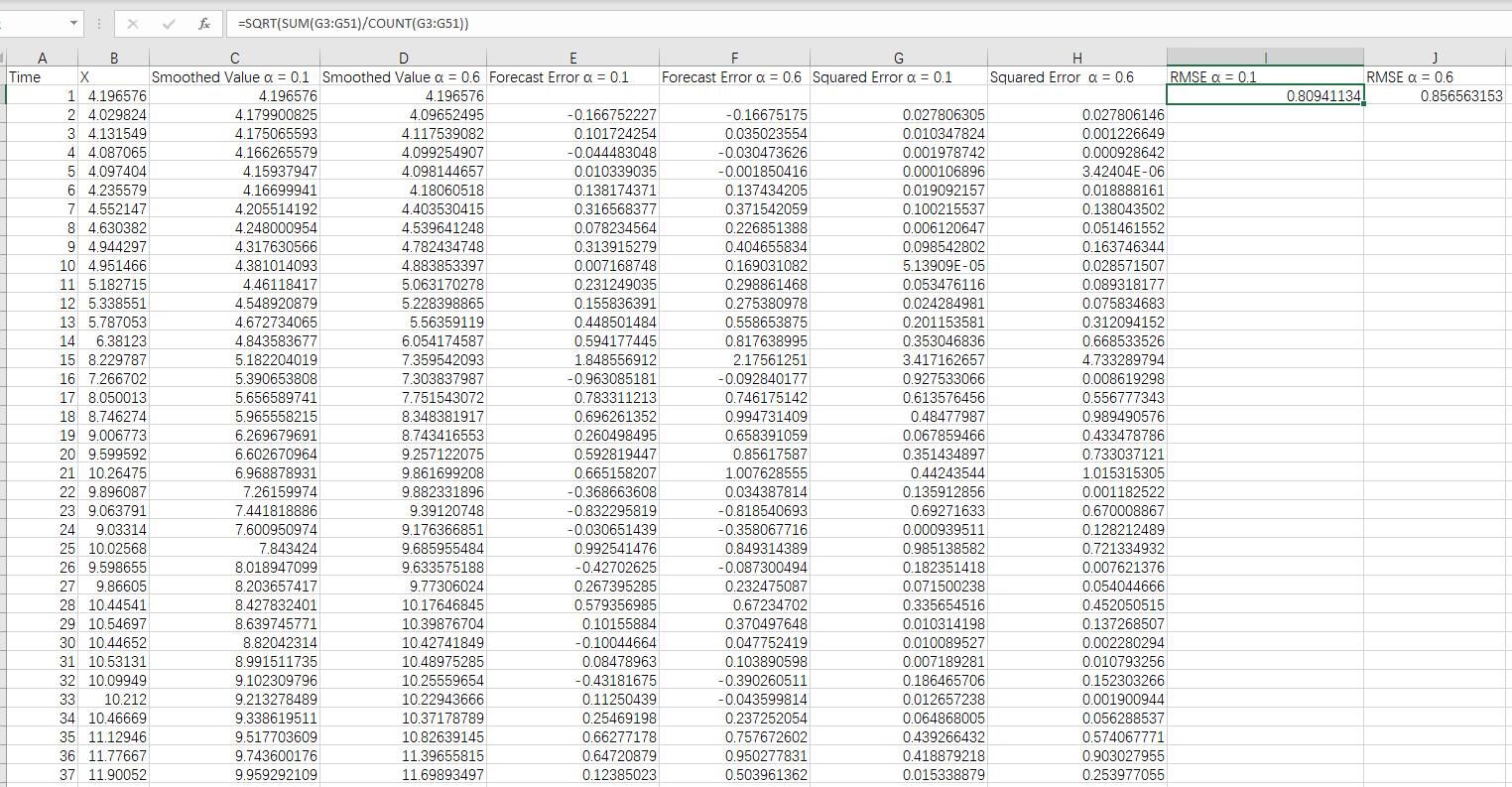
**Problem 1 (10 points). Use the hw9\_p1.csv dataset for this problem. Suppose that you forecast the time series using the single exponential smoothing method that we discussed in the class.**

**(1). Calculate the RMSE for α = 0.1**

**(2). Calculate the RMSE for α = 0.6**

**(3). Which one is better?**



RMSE α = 0.1: 0.80941134

RMSE α = 0.6: 0.856563153

0.80941134 < 0.856563153

So α = 0.1 is better.

**Problem 2 (10 points). For this problem, use hw9\_p2.csv dataset.**

**(1). Plot the time plot.**

**(2). Split the time series into a training set and a validation set with the validation set size = 10.**

**(3). Build a linear trend model from the training set and test it on the validation set. Plot the actual vs. predicted graph and calculate the RMSE.**

**(4). Build an exponential trend model from the training set and test it on the validation set. Plot the actual vs. predicted graph and calculate the RMSE.**

**(5). Build a quadratic trend model from the training set and test it on the validation set. Plot the actual vs. predicted graph and calculate the RMSE.**

**(6). Which model is the best?**

> data <- read.csv("hw9\_p2.csv")

>

> #Question1: Plot the time series

> plot(data$Year, data$Exports, type='o', col='blue', main='Time Series Plot', xlab='Year', ylab='Exports')

图表, 折线图

描述已自动生成

>

> #Question2: Split the dataset

> train\_size <- length(data$Year) - 10

> train <- data[1:train\_size,]

> validation <- data[(train\_size+1):length(data$Year),]

图表, 折线图

描述已自动生成

>

> #Question3: Linear Trend Model

> linear\_model <- lm(Exports ~ Year, data=train)

> linear\_predictions <- predict(linear\_model, newdata=validation)

> plot(validation$Year, validation$Exports, type='o', col='blue', main='Linear Model: Actual vs Predicted', xlab='Year', ylab='Exports')

> points(validation$Year, linear\_predictions, type='o', col='red')

> linear\_rmse <- sqrt(mean((validation$Exports - linear\_predictions)^2))

> print(paste("Linear Model RMSE:", linear\_rmse))

[1] "Linear Model RMSE: 1.47300251267703"

图表, 折线图

描述已自动生成

>

> #Question4: Exponential Trend Model

> exp\_model <- lm(log(Exports) ~ Year, data=train)

> exp\_predictions <- exp(predict(exp\_model, newdata=validation))

> plot(validation$Year, validation$Exports, type='o', col='blue', main='Exponential Model: Actual vs Predicted', xlab='Year', ylab='Exports')

> points(validation$Year, exp\_predictions, type='o', col='red')

> exp\_rmse <- sqrt(mean((validation$Exports - exp\_predictions)^2))

> print(paste("Exponential Model RMSE:", exp\_rmse))

[1] "Exponential Model RMSE: 2.48811103338535"

图表, 折线图

描述已自动生成

>

> #Question5: Quadratic Trend Model

> quad\_model <- lm(Exports ~ Year + I(Year^2), data=train)

> quad\_predictions <- predict(quad\_model, newdata=validation)

> plot(validation$Year, validation$Exports, type='o', col='blue', main='Quadratic Model: Actual vs Predicted', xlab='Year', ylab='Exports')

> points(validation$Year, quad\_predictions, type='o', col='red')

> quad\_rmse <- sqrt(mean((validation$Exports - quad\_predictions)^2))

> print(paste("Quadratic Model RMSE:", quad\_rmse))

[1] "Quadratic Model RMSE: 1.27784982244155"

>

> #Question6: Compare the models

> min\_rmse <- min(linear\_rmse, exp\_rmse, quad\_rmse)

> if(min\_rmse == linear\_rmse){

+ print("The best model is Linear.")

+ } else if(min\_rmse == exp\_rmse){

+ print("The best model is Exponential.")

+ } else {

+ print("The best model is Quadratic.")

+ }

[1] "The best model is Quadratic."

**Problem 3 (10 points). For this problem, use hw9\_p3.csv dataset.**

**(1). Plot the time plot.**

**(2). Decompose the time series and plot the components.**

**(3). Split the time series into a training set and a validation set with the validation set size = 12.**

**(4). Build a model with quadratic trend and seasonality (refer to Slide 43) from the training set and test it on the validation set. Plot the actual vs. predicted graph and calculate the RMSE.**

**(5). Build a Holt-Winter's triple exponential smoothing model (refer to Slide 57) from the training set and test it on the validation set. Plot the actual vs. predicted graph and calculate the RMSE.**

**(6). Run the auto.arima function to find "best" parameters. Show the parameters in your answer.**

**(7). Build an arima model using the above "best" parameters. Plot the actual vs. predicted graph and calculate the RMSE**

> library(forecast)

> library(zoo)

>

> data <- read.csv("hw9\_p3.csv", header = TRUE)

> data$Quarter <- as.yearqtr(data$Quarter, format = "%Y Q%q")

>

> #Convert quarters to a numeric time index for regression analysis

> data$TimeIndex <- as.numeric(format(data$Quarter, "%Y")) + (as.numeric(format(data$Quarter, "%q")) - 1) / 4

>

> #Question1: Plot the time series

> plot(data$Quarter, data$Trips, type = "l", xlab = "Quarter", ylab = "Number of Trips", main = "Time Series Plot")

图形用户界面

描述已自动生成

>

> #Question2: Decompose the time series

> ts\_data <- ts(data$Trips, frequency = 4, start = c(1998, 1))

> decomposed <- stl(ts\_data, s.window = "periodic")

> plot(decomposed)

直方图

描述已自动生成

>

> #Prepare the regression model data with seasonal dummy variables

> data$Trend <- seq\_along(data$Trips)

> data$TrendSquared <- data$Trend^2

>

> #Create dummy variables for each season

> for(i in 1:4) {

+ data[paste("Season", i, sep = "")] <- ifelse(cycle(ts\_data) == i, 1, 0)

+ }

>

> #Question3: Split into training and validation sets

> train\_set <- head(ts\_data, -12)

> validation\_set <- tail(ts\_data, 12)

> train\_time\_index <- seq\_along(train\_set)

> validation\_time\_index <- (length(train\_set) + 1):length(data$Trips)

>

> #Create newdata for training and validation sets

> train\_newdata <- data[train\_time\_index, c("Trend", "TrendSquared", "Season1", "Season2", "Season3", "Season4")]

> validation\_newdata <- data[validation\_time\_index, c("Trend", "TrendSquared", "Season1", "Season2", "Season3", "Season4")]

>

> #Question4: Build a regression model with seasonal and quadratic trend

> lm\_model <- lm(train\_set ~ Trend + TrendSquared + Season1 + Season2 + Season3 + Season4, data = train\_newdata)

>

> #Predict for the validation set

> predicted <- predict(lm\_model, newdata = validation\_newdata)

>

> #Plot the actual vs predicted values

> plot(data$Quarter[validation\_time\_index], validation\_set, type = "l", col = "blue", main = "Actual vs Predicted Plot")

图表, 折线图

描述已自动生成

> lines(data$Quarter[validation\_time\_index], predicted, col = "red")

> legend("topright", legend = c("Actual", "Predicted"), col = c("blue", "red"), lty = 1, cex = 0.8)

>

> #Calculate RMSE

> rmse\_quad <- sqrt(mean((validation\_set - predicted)^2))

> print(paste("RMSE for the regression model with seasonal and quadratic trend:", rmse\_quad))

[1] "RMSE for the regression model with seasonal and quadratic trend: 1.02342086920325"

>

> #Question5: Build a Holt-Winters model

> hw\_model <- HoltWinters(train\_set)

> hw\_forecast <- forecast(hw\_model, h=12)

>

> #Plot the Holt-Winters model's forecast results

> plot(hw\_forecast)

图形用户界面

描述已自动生成

> lines(validation\_set, col="red")

> legend("topright", legend = c("HW Forecast", "Actual"), col = c("blue", "red"), lty = 1, cex = 0.8)

>

> #Calculate RMSE for the Holt-Winters model

> hw\_rmse <- sqrt(mean((validation\_set - hw\_forecast$mean)^2))

> print(paste("RMSE for the Holt-Winters model:", hw\_rmse))

[1] "RMSE for the Holt-Winters model: 1.05578133348919"

>

> #Question6: Run auto.arima to find the best parameters

> best\_arima <- auto.arima(train\_set)

> print(best\_arima)

Series: train\_set

ARIMA(0,0,0)(0,1,1)[4]

Coefficients:

sma1

-0.6149

s.e. 0.1439

sigma^2 = 0.2138: log likelihood = -39.3

AIC=82.59 AICc=82.81 BIC=86.78

>

> #Question7: Build an ARIMA model with the best parameters

> arima\_model <- Arima(train\_set, model=best\_arima)

>

> #Make predictions

> arima\_forecast <- forecast(arima\_model, h=12)

>

> #Plot the actual vs predicted values from the ARIMA model

> plot(arima\_forecast)

图形用户界面

描述已自动生成

> lines(data$Quarter[length(train\_set) + (1:12)], validation\_set, col="red", type="o")

> legend("topright", legend = c("ARIMA Forecast", "Actual"), col = c("blue", "red"), lty = c(1,1), pch = c(NA, 1), cex = 0.8)

>

> #Calculate RMSE for the ARIMA model

> arima\_rmse <- sqrt(mean((validation\_set - arima\_forecast$mean)^2))

> print(paste("RMSE for the ARIMA model:", arima\_rmse))

[1] "RMSE for the ARIMA model: 1.18329810414482"