**Assignment #5**

**Using R Functions to Compare Forecasting Models**

Due: Monday, Oct 9, by 11:59pm

(40 pts. Total)

In Assignment #4, we used The Lasso and Backward Selection to select variables according to their predictive value and create parsimonious (small) forecasting models with high degree of efficiency. In this assignment we start with two candidate models; we can think of them as the two best models in a previous selection process, and we want to further explore their performance.

In the selection process in Assignment #4 we would use a single model (model form and parameter values) to forecast demand for two different products (e.g., two SKUs) across seven different stores. Although in our model search we controlled for store differences using a store dummy variable and a real estate index, in the end these variables were eliminated in the model selection process. Now we want to put our models to the test on each store-SKU combination and explore their forecasting performance.

In this assignment we start with two models created in a previous step. These models are saved as “model mSD1.rda” and “model mSD2.rda.” To get started include in your script the following lines of code to initialize the libraries and read the two models:

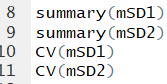
library(fpp)

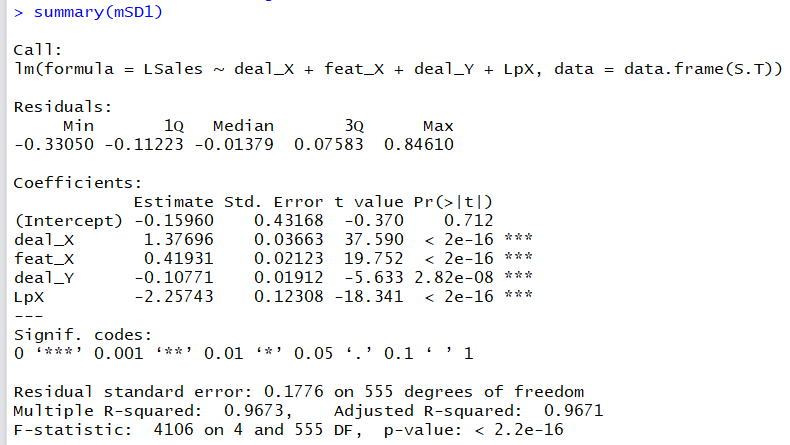
library(dplyr)

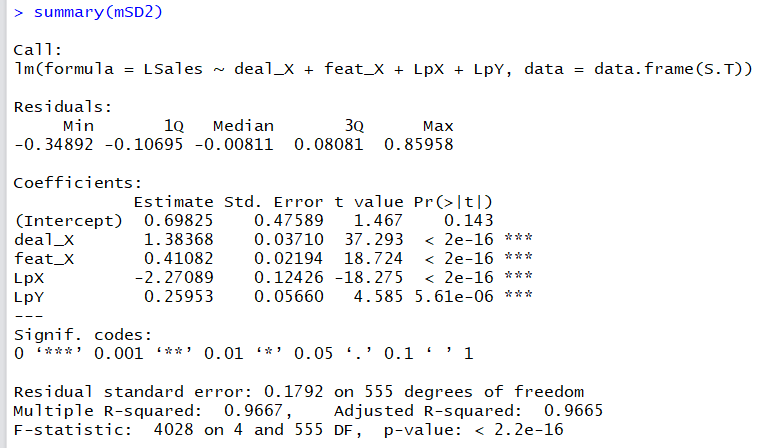
load("model mSD1.rda")  
load("model mSD2.rda")

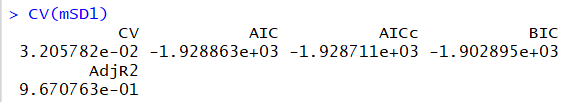
After you have executed the above lines of code, verify in your Global Environment (top right window) to make sure that you have two model objects named “mSD1” and “mSD2”.

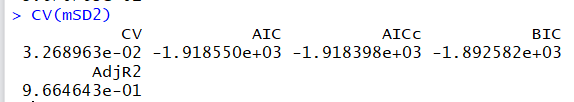
1. (5 pts) Use the summary(…) function to examine the variables and fit of these two models. Then run the cross-validation function, CV(…), on each of them and compare them in terms of their respective CV-index (), their , and their . Is there a better model across the board? Explain.











Both models use the variables deal\_X, feat\_X, and LpX (log of price of X). Model 1 uses the 4th variable of deal\_Y. Model 2 uses the 4th variable of LpY (log of price of Y). Deal\_Y has a lower p-value than LpY, suggesting that Deal\_Y is a better predictor variable in the model. This difference in variables contributes to the differences we see in the R^2 values, CV (MSE), AICc, and BIC values.

In terms of the R^2 value, the first model has a slightly higher R^2 of 0.9673 compared to model 2’s 0.9667 R^2 value. This means that model 1 explains slightly more variation in the Sales of Product X.

Model 1 has a lower CV (MSE) of 0.0321 compared to Model 2’s CV of 0.03269. This means there is less mean standard error in Model 1. When comparing the AICc, Model 1 has a slightly lower AICc of -1928.7 compared to Model 2’s AICc of -1918.4. A lower AICc suggests that Model 1 may be the better fit. In comparing their BIC values, Model 1 also shows a slightly smaller value of -1902 compared to Model 2’s value of -1892.

Model 1 had the higher R^2 value and the lower CV, AICc, and BIC values. This means that Model 1 is a better model across the board, in all the categories we examined. However, the differences in all of the variables examined are very small. Model 1 and Model 2 are not significantly different in their R^2 values or the p-values of their variables. The CV values differed by a thousandth of a place. The AICc and BIC values were close together as well. Even though Model 1 has the “better” numbers, there may be underlying reasons in obtaining variables to use Model 2 instead. Model 2 may be an easier model to use, and, since their summary statistics are similar, the models may show similarly valid predictions.

1. (5 pts) After examining the predictive variables, you must be prepared to obtain in advance (before preparing a forecast) all the predictive variables. Which model is easier to use in terms of the required information? Explain.

Model 1 and Model 2 both use the variables deal\_X, feat\_X, and LpX.

Model 1 uses the variable deal\_Y, while Model 2 uses the variable LpY.

I think that Model 2 is easier to use in terms of obtaining the required information. The price of Y is probably easier to obtain than whether there was a deal on product Y. It can be difficult to say how much of a deal there was on Y, and this model simply uses 1 for “Yes” and 0 for “No,” not quantifying the deal. On the other hand, the log of the price of Y just requires a mathematical calculation on a specific number. In this way, I think that the price of Y is easier to obtain, and the log(price(Y)) is easier to compute than finding the deals on Y and attempting to quantify and compare those deals. Model 2 is easier to use in terms of obtaining and using the required information.

As you recall, from the previous assignment, the models were fitted using the combined data for all the stores and products combined, and you must explore the performance of each of these models for each store-SKU combination. To this end first load and preprocess the dataset (i.e., “Soft Drink Sales.csv” file) so we can get the corresponding store-SKU slices and examine the model performance. The pre-processing script is included below:

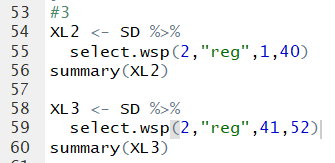
SD <- **read.csv**("Soft Drink Sales.csv")  
*#*  
*#Preprocess data*  
*#*  
SD <- SD %>%   
 **mutate**(STORE=**as.factor**(STORE),  
 LpX = **log**(pX),  
 LpY = **log**(pY),  
 DLpX = deal\_X\*LpX,  
 DLpY = deal\_Y\*LpY,  
 LSales = **log**(Sales\_oz\_X))

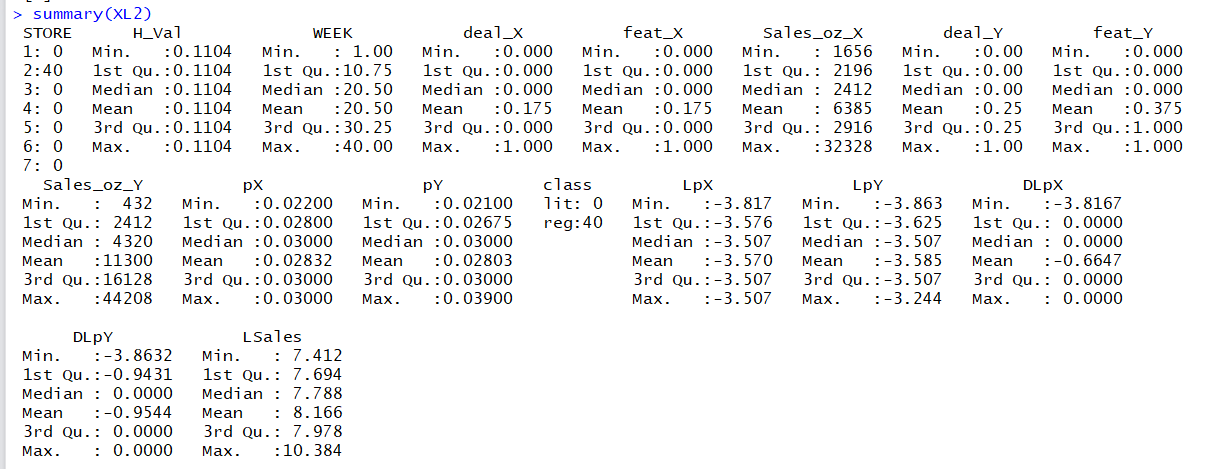
Notice that the new columns (variables) created are the same as in Assignment #4. As there are store-product-model combinations, this appears to be a complicated task. To simplify it we will use user-defined R-functions.

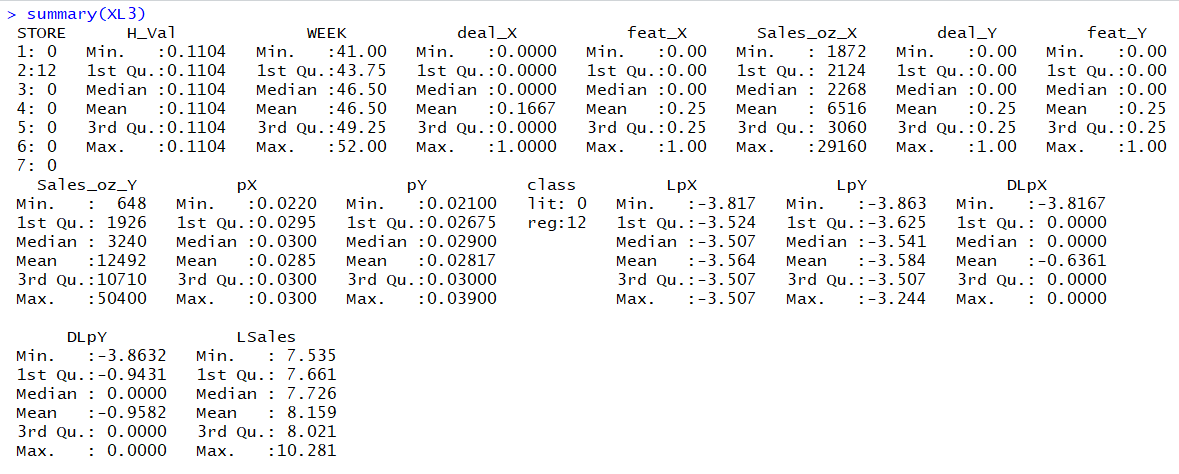
1. (5 pts) First we create a function to select the data corresponding to a particular store-product combination over a range of weeks. The R code is included below for your examination.

*#*  
*# Function to extract a range of weeks of demand*  
*# for a given product and store*  
*#*  
*# Inputs:*  
*# x - data frame with combined data*  
*# store - store number(1 to 7) to extract*  
*# prod - product to extract: "lit" or "reg"*  
*# sw - first week to extract*   
*# ex - last week to extract*  
*#*  
*# Output:*  
*# reduced data.frame with only the store-week-product needed*  
*#*  
select.wsp <- function(x, store, prod, sw, ew){  
 xr <- x %>%   
 **filter**(STORE == store,  
 class == prod,  
 WEEK >= sw,  
 WEEK <= ew)  
 **return**(xr)  
}

Run the above code, and then call to extract the data corresponding to store 2 and “reg” product for weeks 1 through 40 (i.e., use the XL2 <- select.wsp(SD,”reg”,1,40) command) and print the summary(…) of the resulting data set (i.e., XL2). Repeat the process for the out-of-sample range (weeks 41 through 52) for the same store-product combination, and print the in-sample and out-of-sample mean values of the relevant variables (i.e., the predicted and explanatory variables)







These summaries include all of the variables, but I only recorded the values for the relevant explanatory and predictor variables from the 1st two models in Q1 and Q2. In-sample is the 1st summary. Out-of-sample is the 2nd summary.

Predicted Variable:

LSales In-Sample: 8.166 Out-of-Sample: 8.159

Explanatory Variables:

Deal\_X In-Sample: .175 Out-of-Sample: .1667

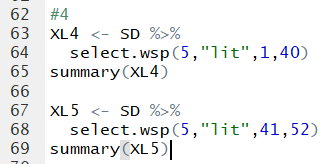
Feat\_X In-Sample: .175 Out-of-Sample: .25

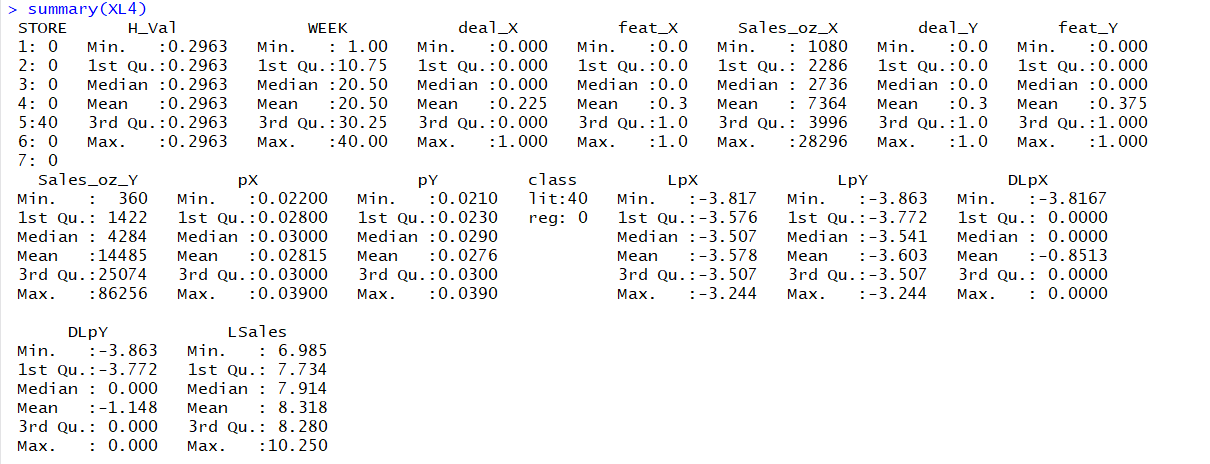
Deal\_Y In-Sample: .25 Out-of-Sample: .25

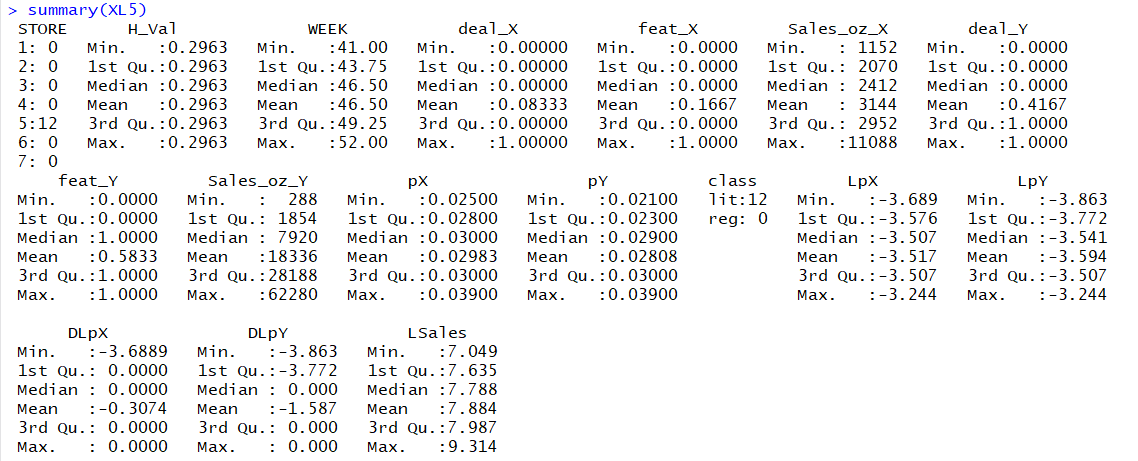
LpX In-Sample: -3.57 Out-of-Sample: -3.564

LpY In-Sample: -3.585 Out-of-Sample: -3.584

1. (5 pts) repeat the process in question (3), but now use the newly created function to extract the data slice for store 5 and product “lit”. Report the in-sample and out-of-sample mean values for all the relevant variables.







These summaries include all of the variables, but I only recorded the values for the relevant explanatory and predictor variables from the 1st two models in Q1 and Q2. In-sample is the 1st summary. Out-of-sample is the 2nd summary.

Predicted Variable:

LSales In-Sample: 8.318 Out-of-Sample: 7.884

Explanatory Variables:

Deal\_X In-Sample: .225 Out-of-Sample: .0833

Feat\_X In-Sample: .3 Out-of-Sample: .1667

Deal\_Y In-Sample: .3 Out-of-Sample: .4167

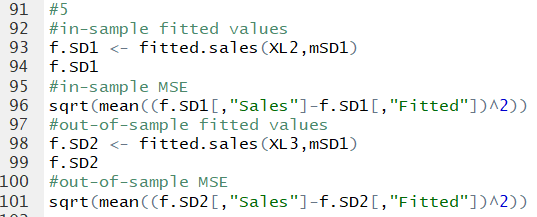
LpX In-Sample: -3.578 Out-of-Sample: -3.517

LpY In-Sample: -3.603 Out-of-Sample: -3.594

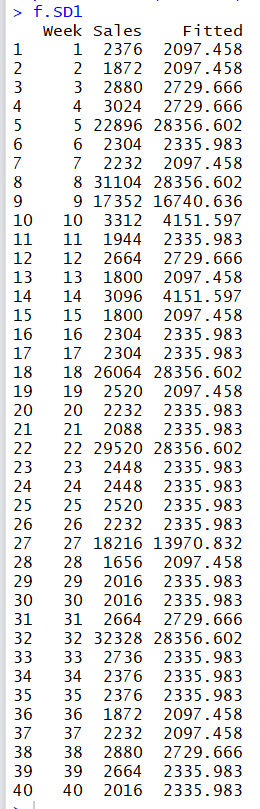
Next we would like to extract the in-sample fitted values and out-of-sample forecasts for each product-store-model combination. Normally we can extract the fitted values of a regression model using the fitted(model) function, but in this case it is not straight forward as the models were fitted using all the combined data and is not clear which fitted values correspond to which stores and products. A simple way to deal with this is to use each model to calculate the in-sample and out-of-sample fit as required. The function included below accomplished this task.

*#*  
*# Evaluate fitted values on the reduced store-product data set*  
*#*  
*# Inputs:*  
*# x - reduced data set (for a store-product over a week-range)*  
*# m - regression model object to obtain the fit*  
*#*  
*# Output:*  
*# data.frame with three columns.*  
*# Week - Week number*  
*# Sales - sales of product*  
*# Fit - fitted value of sales*  
*#*   
fitted.sales <- function(x,m){  
 Fit <- **exp**(**forecast**(m,newdata=x)$mean)  
 Sales <- x[,"Sales\_oz\_X"]  
 **return**(**data.frame**(Week=x[,"WEEK"], Sales=Sales, Fitted=Fit))  
}  
Notice that this function takes a data set, and a model-object as arguments.

1. (5 pts) Examine the above function, and then use it to obtain the in-sample (i.e., weeks 1 through 40) fitted values for store 2 regular product combination using model “mSD1” (i.e. call the function as fitted.sales(XL2,mSD1) using the data set you select in question 3) Now use the function to calculate the out-of-sample fitted values and report the in-sample and out-of-sample MSE for this store-product-model combination.



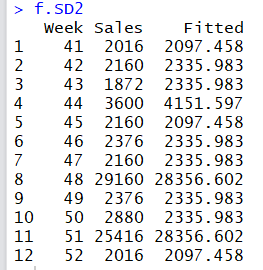
Here are the in-sample fitted values:



In-Sample MSE:



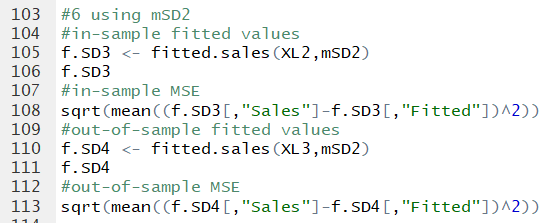
Out-of-Sample Fitted Values:



Out-of-Sample MSE:

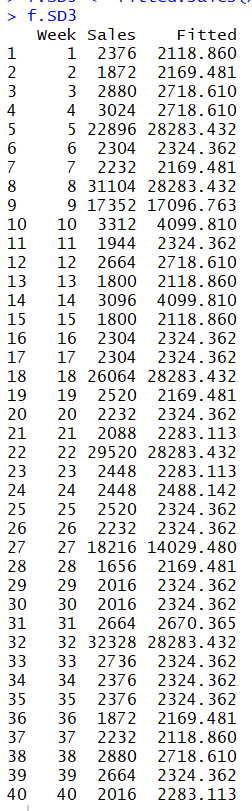


1. (5 pts) Repeat the calculations in question (5) using model “mSD2”.



Using the 2nd model:

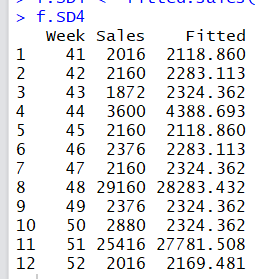
In-Sample Fitted Values:



In-Sample MSE:



Out-of-Sample fitted values:



Out-of-Sample MSE:



The next step in our analysis is to create plots for each store-product-model combination so we compare visually the in-sample and out-of-sample fit. Below I include R-code to create one such plot for Store 1, product “reg” using model “mSD1”.

*#*  
*# Code to create forecasting plot for*  
*# store #1 Regular ("reg") product*  
*# using model mSD1*  
*#*  
*# First we select the data for store 1*  
*# product "reg" over weeks 1-40*  
X <- **select.wsp**(SD,1,"reg",1,40)  
*#*   
*# Next we calculate the fitted sales*  
*# obtained by model mSD1 over weeks 1-40*  
FS <- **fitted.sales**(X,mSD1)  
Week <- FS[,"Week"]  
Sales <- FS[,"Sales"]  
Fitted <- FS[,"Fitted"]  
*#*  
*# Next select the OUT-OF SAMPLE data*  
*# for the same store-product*  
newX <- **select.wsp**(SD,1,"reg",41,52)  
*#*  
*# Calculate the out-of-sample forecasts*  
Fcst <- **fitted.sales**(newX,mSD1)  
FWeek <- Fcst[,"Week"]  
N.Sales <- Fcst[,"Sales"]  
F.Sales <- Fcst[,"Fitted"]  
*#*  
*# Render the Plot*  
**plot**(Week,Sales, type="l", xlim=**c**(0,52), ylim=**c**(2000,45000),  
 main=**paste**("Store",1,"reg",**substitute**(mSD1)),  
 xlab="Weeks", ylab="Sales (oz)")  
**points**(Week,Fitted, col="red", pch=20)  
**lines**(FWeek,N.Sales, col="grey")  
**points**(FWeek,F.Sales, col="blue", pch=20)

Run the code above and examine the plot it produces. Now we would like to create a similar plot for each store-product-model combination. Even though the code above relies on the functions we created in earlier parts of the assignment, this is still a very cumbersome task. To make this task simple we would like to encapsulate the above code in a function, such that we could call this function with the store-product-model combination and the function will respond with the corresponding plot. Questions 7 and 8 ask you to do this

1. (5 pts) Create a function named spm.fplot(…) with the specification below, that will produce the forecasting plot for a required store-product-model combination.

*#*

*# This function creates a forecasting plot*

*# for a store-product-model combination*

*#*

*# Inputs:*

*# sn - store number (1,2,...,7)*

*# pname - product name ("lit" or "reg")*

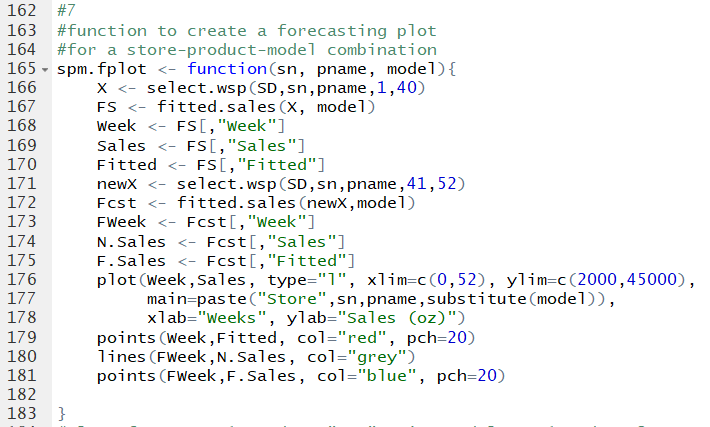
*# model - forecasting model (mSD1 or mSD2)*

*#*

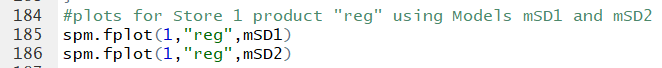
spm.fplot <- function(sn, pname, model){

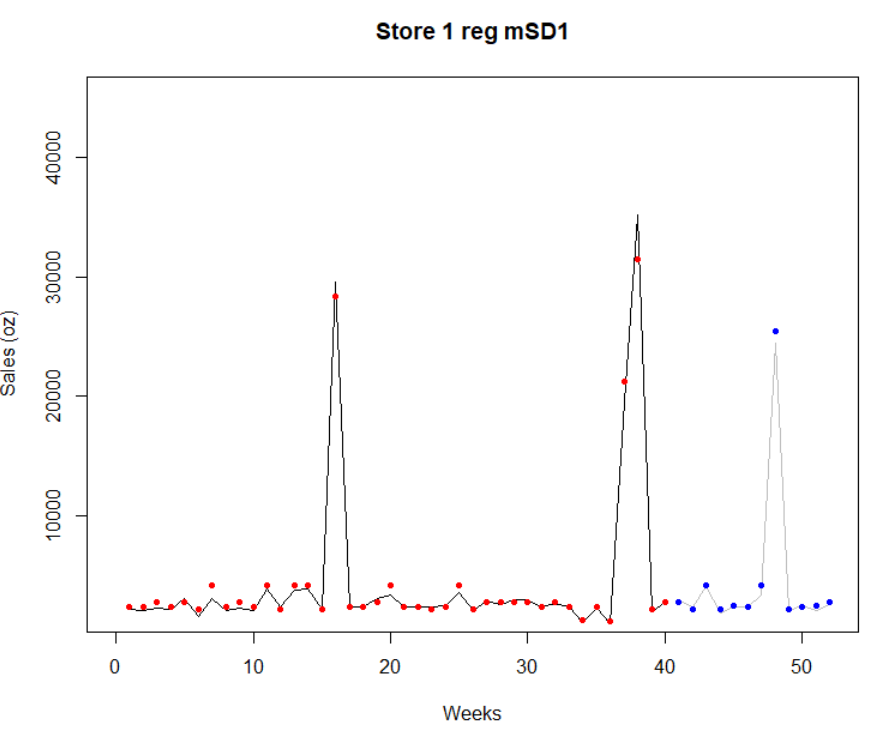
(INSERT FUNCTION CODE HERE)

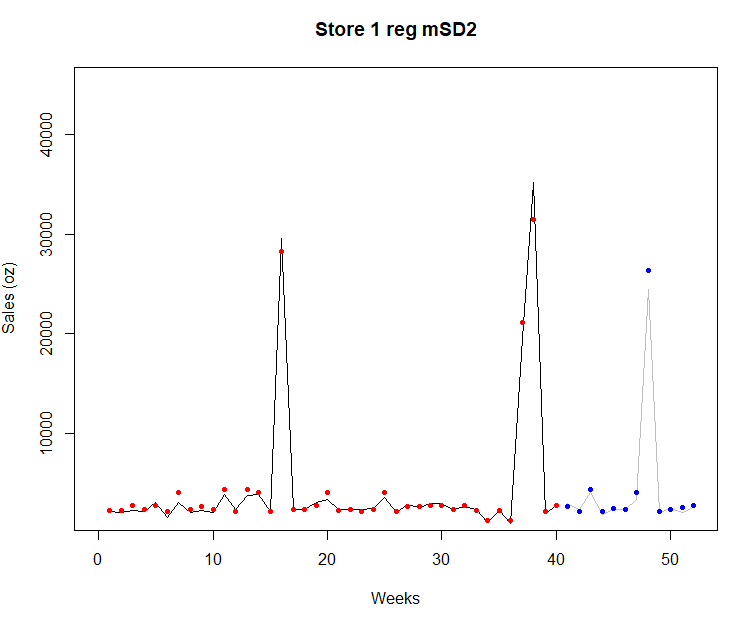
}



After you have create plots for store 1 product “reg” using models mSD1 and mSD2. Include the plots created in your assignment paper.

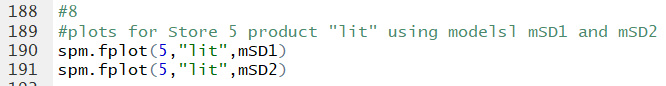


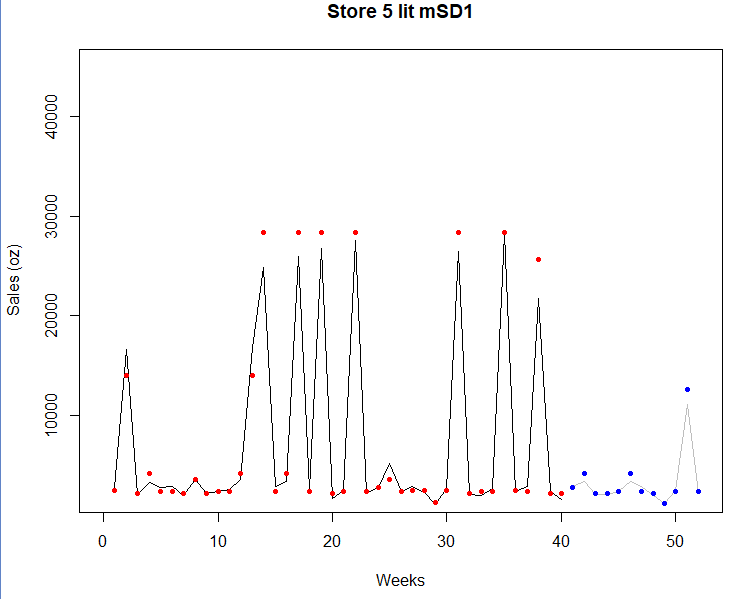


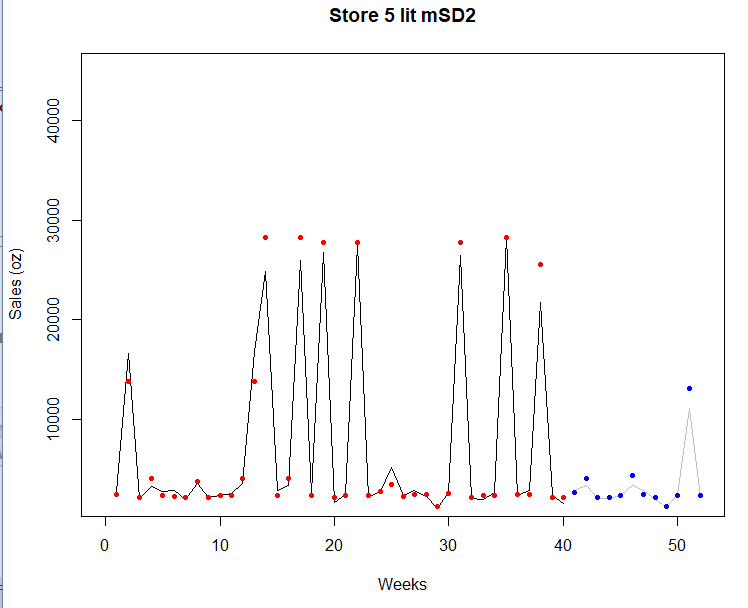


1. (5 pts) Now create plots for store 5 product “lit” using models mSD1 and mSD2. Include the plots created in your assignment paper.

Changed the arguments of the function call for store 5 and product “lit.”







**Question for further thought:**

After you examine visually the forecasting plots, how are you going to select the right model for each store-product? To be continued in Assignment #6 …

After we visually examine the forecasting plots, we would have to quantify the errors for each store-product model. The model with the least amount of error when we combine both the in-sample and out-of-sample errors will be the best fit. In assignment 6, we will have to quantify and compare the errors for both the in-sample and out-of-sample error.