```r
# load necessary packages
if (!require("lubridate")) install.packages("lubridate")
if (!require("fpp2")) install.packages("fpp2")
if (!require("reshape")) install.packages("reshape")
if (!require("plyr")) install.packages("plyr")
if (!require("tidyverse")) install.packages("tidyverse")
library(lubridate)
library(fpp2)
library(reshape)
library(plyr)
library(tidyverse)
mypredict = function(){
        fold_file <- paste0('fold_', t, '.csv')
        new_test <- readr::read_csv(fold_file)
        all.stores = unique(test$Store)
        num.stores = length(all.stores)
        train.dates = unique(train$Date)
        num.train.dates = length(train.dates)
        train.frame = data.frame(Date=rep(train.dates, num.stores),
                                 Store=rep(all.stores, each=num.train.dates))
        preprocess.svd = function(train, n.comp){
                train[is.na(train)] = 0
                z = svd(train[, 2:ncol(train)], nu=n.comp, nv=n.comp)
                s = diag(z$d[1:n.comp])
                train[, 2:ncol(train)] = z$u %*% s %*% t(z$v)
                train
        }
        n.comp = 12 # keep first 12 components
        all.dept = unique(train$Dept)
        for (d in all.dept){
                tr.d = train.frame
                tr.d = join(tr.d, train[train$Dept==d, c('Store','Date','Weekly_Sales')])
                tr.d = cast(tr.d, Date ~ Store)
                tr.d[is.na(tr.d)]=0
                test.dates = unique(new_test$Date)
                num.test.dates = length(test.dates)
                forecast.frame = data.frame(Date=rep(test.dates, num.stores),
                                            Store=rep(all.stores, each=num.test.dates))

                fc.d = forecast.frame
                fc.d$Weekly_Sales = 0
                fc.d = cast(fc.d, Date ~ Store)  # similar as tr.d
                fc.d1 = fc.d
                fc.d2 = fc.d
                horizon = nrow(fc.d)  # number of steps ahead to forecast
                if (t<=6){
                        for(j in 2:ncol(tr.d)){ # loop over stores
                                s = ts(tr.d[,j],frequency = 52)
                                fit = tslm(s~trend + season)
                                fc.d[,j] = as.numeric(forecast(fit,h=horizon)$mean)
                                fc.d1[,j] = as.numeric(naive(s,h=horizon)$mean)
                                fc.d2[,j] = as.numeric(meanf(s,h=horizon)$mean)
                                cd = melt(fc.d, id = c('Date','Store'))
                                ce = join(test[which(test$Dept==d & test$Date %in% test.dates), c('Store'
,'Date','Weekly_Pred1')],cd)
                                test[which(test$Dept==d & test$Date %in% cd$Date & test$Store %in% cd$Sto
re), 'Weekly_Pred1']<<-ce$value

                                cd1 = melt(fc.d1, id = c('Date','Store'))
                                ce1 = join(test[which(test$Dept==d & test$Date %in% test.dates), c('Store
','Date','Weekly_Pred2')],cd1)
                                test[which(test$Dept==d & test$Date %in% cd1$Date & test$Store %in% cd1$S
tore), 'Weekly_Pred2']<<-ce1$value

                                cd2 = melt(fc.d2, id = c('Date','Store'))
                                ce2 = join(test[which(test$Dept==d & test$Date %in% test.dates), c('Store
','Date','Weekly_Pred3')],cd2)
                                test[which(test$Dept==d & test$Date %in% cd2$Date & test$Store %in% cd2$S
tore), 'Weekly_Pred3']<<-ce2$value

                        }
                } else{
```

```
                        for(j in 2:ncol(tr.d)){
                                s = ts(tr.d[, j], frequency = 52)
                                fc.d1[,j] = as.numeric(naive(s,h=horizon)$mean)
                                fc.d2[,j] = as.numeric(meanf(s,h=horizon)$mean)
                                tr.d = preprocess.svd(tr.d, n.comp)
                                s = ts(tr.d[, j], frequency = 52)
                                fc = stlf(s, h=horizon,method='arima')
                                pred = as.numeric(fc$mean)
                                fc.d[, j] = pred
                                cd = melt(fc.d, id = c('Date','Store'))
                                ce = join(test[which(test$Dept==d & test$Date %in% test.dates), c('Store'
,'Date','Weekly_Pred1')],cd)
                                test[which(test$Dept==d & test$Date %in% cd$Date & test$Store %in% cd$Sto
re), 'Weekly_Pred1']<<-ce$value

                                cd1 = melt(fc.d1, id = c('Date','Store'))
                                ce1 = join(test[which(test$Dept==d & test$Date %in% test.dates), c('Store
','Date','Weekly_Pred2')],cd1)
                                test[which(test$Dept==d & test$Date %in% cd1$Date & test$Store %in% cd1$S
tore), 'Weekly_Pred2']<<-ce1$value

                                cd2 = melt(fc.d2, id = c('Date','Store'))
                                ce2 = join(test[which(test$Dept==d & test$Date %in% test.dates), c('Store
','Date','Weekly_Pred3')],cd2)
                                test[which(test$Dept==d & test$Date %in% cd2$Date & test$Store %in% cd2$S
tore), 'Weekly_Pred3']<<-ce2$value

                        }
                }
        }
        train <<-rbind(train,new_test)
        train[is.na(train)] <- 0
}
```

For this project, I build models for each store and department combination. If there is missing values for week and store combination, I imputed with zeros.

For model 1, I used tslm function for the first 6 folds. This will get seasonality and trends components without 2 periods of data. For the last 4 folds, I applied svd and keep the first 12 components to the original time series. As I have more than two years data, stlf function is used by specifying method equals to arima. This function divided the time series into error, trend and seasonality and combine with arima for seasonality adjusted data prediction.

For model 2, I used naive function as the last value of the time series will be used as the predictions in test data.For model 3, I used smean function as the the mean of the time series will be used to predict in the testing period.

The overall running time is 2 hours with a 2.40GHZ computer. The final model performance is shown below.

model 1	model 2	model 3
2042.401	2078.726	2136.906
1440.083	2589.338	2526.458
1434.716	2253.936	2535.618
1596.988	2823.098	2364.819
2327.638	5156.012	5470.421
1674.185	4218.348	2798.065
1594.886	2226.376	2341.859
1330.824	2103.689	2643.144
1267.275	2196.452	2625.443
1236.867	2321.425	2422.752

Overal Average is shown below:

model_one	model_two	model_three
1594.586	2796.740	2786.549