Team Assignment #1 : Forecasting SKU Demand at the Point of Sale

Tianxin Huang, Jinming Li, Yile Zhou, Helena Shi, Shan Qin

1. A detailed description of your forecasting approach, model(s) characteristics, and a brief justification for each of the predictive variables (model features) used.

- Joining Files And Dropping Unnecessary Variables
  In our forecasting approach, we first utilized the DataPreprocessing.R file and left joined multiple files together to get one dataframe with features per UPC. Then, we created the variable DT by first removing/dropping variables Dollars, UPC, and Market_Name, which in our opinion wouldn't contributed to the model's predictive ability. We removed Dollars because we knew it would be collinear with PPU, and UPC because it is a duplicate of the combination of SY, GE, VEND, ITEM. Lastly, we removed Market_Name as the data left in the data frame had already been localized to Chicago only.

- Dummy Coding
  We then converted the remaining variables, including IRI_KEY, SY, GE, VEND, ITEM, F, D, and PR into factors so they could be dummy coded and dummy coded the categorical features with the model.matrix() function.

```
DT$IRI_KEY = as.factor(DT$IRI_KEY)
DT$SY = as.factor(DT$SY)
DT$GE = as.factor(DT$GE)
DT$VEND = as.factor(DT$VEND)
DT$ITEM = as.factor(DT$ITEM)
DT$F = as.factor(DT$F)
DT$D = as.factor(DT$D)
DT$PR = as.factor(DT$PR)
str(DT)
options(na.action = "na.pass")
DTM <-model.matrix(UNITS ~ ., data = DT)[,-1]
```

- Splitting Data and Fitting Model
  We splitted the data into trains, test, and forecasted data.

```
D.TR <- DTM[DTM[,"WEEK"] <= 1663,]
D.TE <- DTM[(DTM[,"WEEK"] >= 1664) & (DTM[,"WEEK"] <= 1673),]
D.H  <- DTM[DTM[,"WEEK"] <= 1673,]
D.FC <- DTM[DTM[,"WEEK"] >= 1674,]

y.tr <- DT %>%
  filter(WEEK <= 1663) %>%
  pull(UNITS)
y.te <- DT %>%
  filter(WEEK >= 1664 & WEEK <= 1673) %>%
  pull(UNITS)
y.h <- DT %>%
  filter(WEEK <= 1673) %>%
  pull(UNITS)
```

We had fit a model with lasso regression to perform feature selection and reduce collinear variables but it wasn't helpful because we had already removed the collinear Dollars variable. Therefore, we performed XGBoost without Lasso, using the following parameters:

```r
xb <- xgboost(D.TR, y.tr,
              learning_rate = 0.05,
              lambda = 1,
              max_depth = 10,
              subsample = 0.8,
              colsample_bytree = 0.9,
              colsample_bylevel = 0.9,
              nround = 20)
```

With those parameters, we obtained a testing MAPE of 47.65, a testing MPE of 7.22.

```
> test_accuracy = accuracy(xb_predict, y.te)
> test_accuracy
               ME     RMSE      MAE      MPE     MAPE
Test set 3.659215 13.66695 4.387343 7.225356 47.65569
```

2. Prepare a forecast for each of the SKU-Store-Week combination (i.e., the next 10 weeks after the end of sales history) in the sales plan file. Part deliverable is an expanded version of the sales plan including an added column with your forecast. Report your team's modeling approach as well as your training and testing metrics. Part of the grade in this question will be based on the MPE and MAPE of the forecast.

   We used the XGBoost model that we figured out in Q1. As we mentioned in the answer of Q1, the model had **a testing MAPE of 47.65 and a testing MPE of 7.22.** Using the model we obtained from Q1, fit it to the entire dataset and forecast the following 10 period:

```r
xb_fc <- xgboost(D.H, y.h,
              learning_rate = 0.05,
              lambda = 1,
              max_depth = 10,
              subsample = 0.8,
              colsample_bytree = 0.9,
              colsample_bylevel = 0.9,
              nround = 20)
```

The training MAPE and MPE turned out 44.68 and 7.54.

```
> fc_train_accuracy
                   ME     RMSE      MAE      MPE     MAPE
Training set 3.158081 9.416961 3.774848 7.536185 44.68257
```

The expanded version of the sales plan including an added column **UNIT_FORECAST** with our forecast is as follows:

forecast_df

IRI_KEY	WEEK	SY	GE	VEND	ITEM	UNITS	F	D	PR	PPU	EST_ACV	UNITS.L1	AVG.UNITS	VOL_EQ	TYPE	TEXTURE	FLAVOR	PPOZ	PBMSF	UNIT_FORECAST
234212	1674	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.018066261	2.037980079650879
234212	1675	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.097291268	2.037980079650879
234212	1676	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.024840186	2.037980079650879
234212	1677	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	0.952394012	2.037980079650879
234212	1678	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	0.949232844	2.037980079650879
234212	1679	0	1	16459	20011	NA	NONE	0	0	6.84	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.4275	0.937694351	2.037980079650879
234212	1680	0	1	16459	20011	NA	NONE	0	0	6.49	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.042191686	2.037980079650879
234212	1681	0	1	16459	20011	NA	NONE	0	0	6.49	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.065193425	2.037980079650879
234212	1682	0	1	16459	20011	NA	NONE	0	0	6.49	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.920966841	2.037980079650879
234212	1683	0	1	16459	20011	NA	NONE	0	1	6.656666667	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.4160416666875	1.012784554	2.8634109497070312
1130089	1674	0	1	16459	20011	NA	NONE	0	0	6.99	30.29099	NA	NaN	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.018066261	0.8275073170661926
1130089	1682	0	1	16459	20011	NA	NONE	0	0	6.49	30.29099	NA	NaN	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.920966841	0.8275073170661926
1130099	1675	0	1	16459	20011	NA	NONE	0	0	6.99	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.097291268	0.8312021493911743
1130099	1676	0	1	16459	20011	NA	NONE	0	0	6.99	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.024840186	0.8312021493911743
1130099	1681	0	1	16459	20011	NA	NONE	0	1	6.49	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.065193425	0.8484846353530884
1130099	1682	0	1	16459	20011	NA	NONE	0	1	6.49	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.920966841	0.8484846353530884
1130099	1683	0	1	16459	20011	NA	NONE	0	1	6.49	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.012784554	0.8484846353530884
1130105	1674	0	1	16459	20011	NA	NONE	0	0	6.99	17.467	0.6931471805599453	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.018066261	1.1646232604980469
1130105	1675	0	1	16459	20011	NA	NONE	0	0	6.99	17.467	0.6931471805599453	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.097291268	1.1646232604980469
1130105	1679	0	1	16459	20011	NA	NONE	0	1	6.49	17.467	0.6931471805599453	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.937694351	1.2139573097229004
1130105	1680	0	1	16459	20011	NA	NONE	0	1	6.49	17.467	0.6931471805599453	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.042191686	1.2139573097229004
1130121	1675	0	1	16459	20011	NA	NONE	0	0	6.99	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.097291268	1.1646232604980469
1130121	1678	0	1	16459	20011	NA	NONE	0	0	6.99	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	0.949232844	1.1646232604980469
1130121	1681	0	1	16459	20011	NA	NONE	0	1	6.49	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.065193425	1.28506927218628
1130121	1682	0	1	16459	20011	NA	NONE	0	1	6.49	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.920966841	1.28506927218628
1130121	1683	0	1	16459	20011	NA	NONE	0	1	6.49	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.012784554	1.28506927218628
1130152	1677	0	1	16459	20011	NA	NONE	0	0	6.99	34.50198	0	0.28768207245178085	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	0.952394012	1.0400134325027466

3. Assume now that you are part of the supply chain analytics group in charge of planning the production for all “Skippy” brand products. You need to prepare an aggregate forecast for the production of each SKU for Chicago market. Prepare a report (CSV file) of sales indicating your forecast for weeks 6 through 10 after the end of the sales history. This will be a total sales forecast for weeks 6-10 for each SKU for the entire Chicago market. Report your team’s modeling approach as well as your training and testing metrics. Part of the grade in this question will be based on the MPE and MAPE of the forecast.

   For data preprocessing, we first created a new data frame “D_SKIPPY”, by extracting the UPC, WEEK, and UNITS columns of all “Skippy” brand products from the original “D” data frame. Then, we added a new column called TOTAL_UNITS, which is the sum of UNITS by each SKU and WEEK combination.Before we started training our model, we removed 3 SKU products which does not have data from 1635 to 1673. It will be impossible to predict the planning sales without previous historical data on each of that three SKU products.

```
############################## Filtering 'SKIPPY' brand products ############################
D_SKIPPY = D %>%
  left_join(select(U, c(L5, UPC))) %>%
  filter(L5 %in% unique(grep('SKIPPY', L5, value = TRUE))) %>%
  select(UPC, WEEK, UNITS) %>%
  group_by(UPC, WEEK) %>%
  mutate(TOTAL_UNITS = sum(UNITS)) %>%
  select(c(UPC, WEEK, TOTAL_UNITS)) %>%
  unique() %>%
  arrange(UPC, WEEK) %>%
  filter((UPC != '00-01-48001-10534') & (UPC != '00-01-48001-12085') & (UPC != '00-01-48001-16016')
        & (UPC != '00-01-48001-27060') & (UPC != '00-02-48001-09430'))%>%
  ungroup()
```

The resulting data frame looks like this:

	UPC	WEEK	TOTAL_UNITS
1	00-01-48001-00641	1635	186
2	00-01-48001-00641	1636	612
3	00-01-48001-00641	1637	314
4	00-01-48001-00641	1638	275
5	00-01-48001-00641	1639	1792
6	00-01-48001-00641	1640	1003

We then tried XGBoost and ARIMA to forecast the production of all “Skippy” brand products for Chicago market. Using MPE and MAPE as the training and testing metrics, the ARIMA model performs better than XGBoost model.

- For the XGBoost model, the best MAPE got was 69% even after adding seasonality to the data.
- For the Arima model, we used auto.arima function to find the best order for each UPC. Then, we used Arima function with best fit order to fit the entire dataset.

```r
result_df = c()
for (sku in unique(D_SKIPPY$UPC)){
  print(sku)
  arima_ts = ts(filter(D_SKIPPY, (UPC == sku) & (WEEK < 1674))[, 'TOTAL_UNITS'], start = 1635)
  arima_train = window(arima_ts, end = 1663)
  arima_test = window(arima_ts, start = 1664, end = 1673)
  arima_model = auto.arima(arima_train)
  arima_predict = forecast(arima_model, h = 10)
  a = accuracy(arima_predict, arima_test)
  arima_forecast_model = Arima(arima_ts, order = c(arima_model$arma[1], arima_model$arma[6], arima_model$arma[2]),
                               seasonal = c(arima_model$arma[3], arima_model$arma[7], arima_model$arma[4]))
  arima_forecast = forecast(arima_forecast_model, h = 10)
  forecast_6to10 = sum(as.data.frame(arima_forecast)[, 'Point Forecast'][6:10])
  result_df = rbind(result_df, c(sku, forecast_6to10, a['Training set', 'MPE'],
                                 a['Training set', 'MAPE'], a['Test set', 'MPE'], a['Test set', 'MAPE']))
}
```

The week 6-10 forecast, the training and testing MPE, and the training and testing MAPE of the ARIMA model are below:

```
> result_df
```

	Forecast_6to10	Train_MPE	Train_MAPE	Test_MPE	Test_MAPE
00-01-48001-00641	2608.2807	-36.955963	57.56468	-1.835524	42.90352
00-01-48001-00643	6983.9902	-27.947633	49.64281	-9.671168	58.06599
00-01-48001-00677	211.0256	-69.574318	91.09990	-20.477472	52.69415
00-01-48001-00678	1030.0000	-33.770641	64.05910	66.193724	66.19372
00-01-48001-00681	2030.0000	-19.846505	44.27368	63.253400	63.77288
00-01-48001-00686	1027.5528	-11.604428	29.24271	-9.056131	30.04816
00-01-48001-00687	2639.4256	-11.853148	29.25184	-2.709700	43.32473
00-01-48001-23202	373.4615	-4.213182	16.80362	-149.388973	151.52768
00-01-48001-27044	502.3033	-5.130470	16.99009	-11.277069	22.63164
00-01-48001-27048	346.4326	-1.439965	17.00461	13.352198	14.96040
00-01-48001-27068	1278.1315	-2.407211	11.31147	2.897769	20.80004
00-01-48001-27072	885.0000	-1.006727	11.90015	-1.984909	23.21183
00-01-48001-27213	228.5897	-11.439223	30.07290	-80.877075	80.87707
00-01-48001-90520	138.4615	-3.726245	15.84235	10.158865	21.75824
00-01-48001-90570	289.7436	-2.659169	13.88168	11.099233	11.81397

4. Now assume the position of the supermarket manager(s), and you must decide how many units (an integer number) of each SKU you must hold at each location for each of the first five weeks of the planning horizon. Prepare a report (CSV file) with the stocking decisions for each store-SKU-week combination. Assume that the supermarket's unit profit margin is 20% of sales price and the cost of holding a unit of a product of leftover inventory at the end of each week (including opportunity costs for the shelf space) is 10% of the sales price. Report your team's decision-making approach as well as your training and testing metrics. Part of the grade will be based on the sum of the costs of over-stocking and under-stocking incurred at the store during the forecasted period.

   In order to set the target inventory, we need to know safety factor k for all SKUs and also we need to know CV for all SKUs. Therefore, we will have to forecast the sales on log-volume since all of these will be done in log-scale.

The unit understocking cost is 20% of the sales prices of all SKUs while the unit overstocking cost is 10% of the sales prices of all SKUs. Using this information, we might be able to get the ratio = 0.667 and hence we can obtain k = 0.42.

Additionally, by transforming the unit sales into log-scale and fit the XGBoost model using data from 1635 to 1663, and testing the model using data from 1664 to 1673, we might be able to capture the test RMSE = 0.7805015 which can be used as the sigma to obtain CV since CV = sqrt(exp(sigma^2) - 1).

```
                     ME      RMSE       MAE  MPE MAPE
Training set 0.3773069 0.7237701 0.6090987 -Inf  Inf
Test set     0.3994000 0.7805015 0.6487655 -Inf  Inf
```

Using CV and k, the target inventory can be computed as "TI = forecast + k * CV * forecast" as such:

```
print(rbind(train_accuracy, test_accuracy))
sigma = test_accuracy['Test set', 'RMSE']
k = 0.42
cv = sqrt(exp(sigma^2) - 1)
forecast_inventory = forecast_df %>%
  mutate(INVENTORY_PLAN = (1 + k * cv) * UNIT_FORECAST)
```

The target inventory is added as a column INVENTORY_PLAN into the planning horizon dataframe:

Q4

IRI_KEY	WEEK	SY	GE	VEND	ITEM	UNITS	F	D	PR	PPU	EST_ACV	UNITS.L1	AVG.UNITS	VOL_EQ	TYPE	TEXTURE	FLAVOR	PPOZ	PBMSF	UNIT_FORECAST	INVENTORY_PLAN
234212	1674	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.018066261	2.037980079650879	2.8219717641929307
234212	1675	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.097291268	2.037980079650879	2.8219717641929307
234212	1676	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.024840186	2.037980079650879	2.8219717641929307
234212	1677	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	0.952394012	2.037980079650879	2.8219717641929307
234212	1678	0	1	16459	20011	NA	NONE	0	0	6.99	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	0.949232844	2.037980079650879	2.8219717641929307
234212	1679	0	1	16459	20011	NA	NONE	0	0	6.84	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.4275	0.937694351	2.037980079650879	2.8219717641929307
234212	1680	0	1	16459	20011	NA	NONE	0	0	6.49	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.042191686	2.037980079650879	2.8219717641929307
234212	1681	0	1	16459	20011	NA	NONE	0	0	6.49	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.065193425	2.037980079650879	2.8219717641929307
234212	1682	0	1	16459	20011	NA	NONE	0	0	6.49	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.920966841	2.037980079650879	2.8219717641929307
234212	1683	0	1	16459	20011	NA	NONE	0	1	6.656666667	36.875	0.6931471805599453	1.2039728043259361	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.4160416666875	1.012784554	2.8634109497070312	3.9649380923970314
1130089	1674	0	1	16459	20011	NA	NONE	0	0	6.99	30.29099	NA	NaN	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.018066261	0.8275073170661926	1.14584156476342
1130089	1682	0	1	16459	20011	NA	NONE	0	0	6.49	30.29099	NA	NaN	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.920966841	0.8275073170661926	1.14584156476342
1130099	1675	0	1	16459	20011	NA	NONE	0	0	6.99	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.097291268	0.8312021493911743	1.150957764180007
1130099	1676	0	1	16459	20011	NA	NONE	0	0	6.99	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.024840186	0.8312021493911743	1.150957764180007
1130099	1681	0	1	16459	20011	NA	NONE	0	1	6.49	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.065193425	0.8484846353530884	1.1748886592297452
1130099	1682	0	1	16459	20011	NA	NONE	0	1	6.49	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.920966841	0.8484846353530884	1.1748886592297452
1130099	1683	0	1	16459	20011	NA	NONE	0	1	6.49	31.96399	0	0	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.012784554	0.8484846353530884	1.1748886592297452
1130105	1674	0	1	16459	20011	NA	NONE	0	0	6.99	17.467	0.6931471805599453	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.018066261	1.1646232604980469	1.6126428270146802
1130105	1675	0	1	16459	20011	NA	NONE	0	0	6.99	17.467	0.6931471805599453	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.097291268	1.1646232604980469	1.6126428270146802
1130105	1679	0	1	16459	20011	NA	NONE	0	1	6.49	17.467	0.6931471805599453	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.937694351	1.2139573097229004	1.6809552189344728
1130105	1680	0	1	16459	20011	NA	NONE	0	1	6.49	17.467	0.6931471805599453	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.042191686	1.2139573097229004	1.6809552189344728
1130121	1675	0	1	16459	20011	NA	NONE	0	0	6.99	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	1.097291268	1.1646232604980469	1.6126428270146802
1130121	1678	0	1	16459	20011	NA	NONE	0	0	6.99	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	0.949232844	1.1646232604980469	1.6126428270146802
1130121	1681	0	1	16459	20011	NA	NONE	0	1	6.49	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.065193425	1.285069227218628	1.7794232193208834
1130121	1682	0	1	16459	20011	NA	NONE	0	1	6.49	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.920966841	1.285069227218628	1.7794232193208834
1130121	1683	0	1	16459	20011	NA	NONE	0	1	6.49	49.01498	0	0.4519851237430572	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.012784554	1.285069227218628	1.7794232193208834
1130152	1677	0	1	16459	20011	NA	NONE	0	0	6.99	34.50198	0	0.28768207245178085	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.436875	0.952394012	1.0400134325027466	1.4400967753359442
1130152	1679	0	1	16459	20011	NA	NONE	0	0	6.49	34.50198	0	0.28768207245178085	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	0.937694351	1.0400134325027466	1.4400967753359442
1130152	1680	0	1	16459	20011	NA	NONE	0	0	6.49	34.50198	0	0.28768207245178085	1	PEANUT BUTTER SPREAD	CREAMY	ORIGINAL	0.405625	1.042191686	1.0400134325027466	1.4400967753359442