FINAL PROJECT
DATA MINING

UNDER- **PROFESSOR SENJUTI BASU ROY**
COURSE-**CS634**

BY- SURAJ KUMAR JHA &
BALJINDER KAUR SMAGH

PROJECT PROBLEM DEFINATION

Overarching goal: The overarching goal of the project is to predict total sales for every product and store in the next month. The dataset contains time-series data consisting of daily sales data, by one of the largest Russian software firms .

1. Data Preparation, Exploratory Analysis:

CLEANING DATA :

- Look for any Missing Data
  - Identify observations on Missing Data
  - Graph the representations of Missing Data
  - Decide whether to populate or to remove Missing Data
  - Identify the possible impact of it while Modelling

2. OUTLIER DETECTION:

Find out the months of sale which are considered as outliers for any shop.

3. FEATURE SELECTION/ENGINEERING:

Identify and convert Categorical columns/values to Numerical representation using Dummy Variables if suitable for modelling

4. MODELING:

Build 2 different models that uses all data from the training.csv and other files for all the months and years except October 2015.

5. Validation:

Use October 2015 data as test set and present

a. Mean squared error

b. root mean squared error (RMSE)

6. Compute confidence interval of Model 1 and Model 2 for the following different confidence levels:

80%, 90%, 95%

7. Compare these two models considering

a. error

b. efficiency in training time (scalability)

DATA SET DEFINATION

- **shop_id** - unique identifier of a shop
- **item_id** - unique identifier of a product
- **item_category_id** - unique identifier of item category
- **item_cnt_day** - number of products sold. You are predicting a monthly amount of this measure
- **item_price** - current price of an item
- **date** - date in format dd/mm/yyyy
- **date_block_num** - a consecutive month number, used for convenience. January 2013 is 0, February 2013 is 1,..., October 2015 is 33
- **item_name** - name of item
- **shop_name** - name of shop
- **item_category_name** - name of item category

DATA PREPROCESSING

Reading the data files -

```
#read sales data
sales_data<- read.csv(file = 'sales_train.csv')

#read shops data
shops<- read.csv(file = 'shops.csv')

#read items data
items<- read.csv(file = 'items.csv')

#read item category data
item_categories<- read.csv(file = 'item_categories.csv')
```

Join the tables using left join -

```
library(dplyr)
sales_data<-left_join(sales_data,items)
sales_data<-left_join(sales_data,item_categories)
sales_data<-left_join(sales_data,shops)
```

Get summary of sales data -

```
> summary(sales_data)
         date          date_block_num     shop_id         item_id        item_price        item_cnt_day
 28.12.2013:   9434   Min.   : 0.00   Min.   : 0   Min.   :    0   Min.   :    -1.0   Min.   : -22.000
 29.12.2013:   9335   1st Qu.: 7.00   1st Qu.:22   1st Qu.: 4476   1st Qu.:   249.0   1st Qu.:   1.000
 30.12.2014:   9324   Median :14.00   Median :31   Median : 9343   Median :   399.0   Median :   1.000
 30.12.2013:   9138   Mean   :14.57   Mean   :33   Mean   :10197   Mean   :   890.9   Mean   :   1.243
 31.12.2014:   8347   3rd Qu.:23.00   3rd Qu.:47   3rd Qu.:15684   3rd Qu.:   999.0   3rd Qu.:   1.000
 27.12.2014:   8041   Max.   :33.00   Max.   :59   Max.   :22169   Max.   :307980.0   Max.   :2169.000
 (Other)   :2882230

                                                                                                item_name       item_category_id
 Фирменный пакет майка 1С Интерес белый (34*42) 45 мкм                                               :  31340   Min.   : 0
 Playstation Store пополнение бумажника: Карта оплаты 1000 руб.                                      :   9408   1st Qu.:28
 Прием денежных средств для 1С-Онлайн                                                                :   9067   Median :40
 Diablo III [PC, Jewel, русская версия]                                                              :   7479   Mean   :40
 Kaspersky Internet Security Multi-Device Russian Edition. 2-Device 1 year Renewal Box:              :   6853   3rd Qu.:55
 World of Warcraft. Карта оплаты игрового времени (online) (рус.в.) (60 дней) (Jewel) :             :   6602   Max.   :83
 (Other)                                                                                             :2865100

                 item_category_name                                          shop_name
 Movie-DVD                  : 564652   Moscow shopping center "Semenovsky"        : 235636
 PC games-Standard editions: 351591   Moscow TRK"atrium"                         : 186104
 Music-CD local production : 339585   Khimki shopping center "Mega"              : 143480
 Games-PS3                  : 208219   Moscow shopping center "MEGA Teply Stan "II: 142234
 Cinema-Blu-Ray             : 192674   Yakutsk Ordzhonikidze, 56                  : 117428
 Games-XBOX 360             : 146789   St. Petersburg TC "Nevsky Center"          : 109253
 (Other)                    :1132339   (Other)                                    :2001714
```

Get structure of features in sales data -

```
> str(sales_data)
'data.frame':   2935849 obs. of  10 variables:
 $ date              : Factor w/ 1034 levels "01.01.2013","01.01.2014",..: 35 69 137 171 477 307 35 103 341 69 ...
 $ date_block_num    : int  0 0 0 0 0 0 0 0 0 0 ...
 $ shop_id           : int  59 25 25 25 25 25 25 25 25 25 ...
 $ item_id           : int  22154 2552 2552 2554 2555 2564 2565 2572 2572 2573 ...
 $ item_price        : num  999 899 899 1709 1099 ...
 $ item_cnt_day      : num  1 1 -1 1 1 1 1 1 1 3 ...
 $ item_name         : Factor w/ 22170 levels "! ВО ВЛАСТИ НАВАЖДЕНИЯ (ПЛАСТ.)         D",..: 22155 2651 2651 2653 2654 2663 2664 2676 2676 2677 ...
 $ item_category_id  : int  37 58 58 58 56 59 56 55 55 55 ...
 $ item_category_name: Factor w/ 84 levels "Accessories-PS2",..: 22 65 65 65 60 64 60 61 61 61 ...
 $ shop_name         : Factor w/ 60 levels "!Yakutsk Ordzhonikidze, 56 Fran",..: 58 28 28 28 28 28 28 28 28 28 ...
```

Insights from Data –

(i)Data has 2.9 million observations with 10 variables(columns).
(ii)Data has no missing values but they are negative values present in data as we can see from summary that item price and item count day are having negative values.

Partitioning of sales data into train data and test data—

```
#Partinioning data
train_data<-sales_data[(sales_data$date_block_num<33),]
test_data<-sales_data[!(sales_data$date_block_num<33),]
```

DATA CLEANING

Cleaning Data –

(i)Observations on missing data-

```
> sum(is.na(train_data))
[1] 0
```

As we can see there are no missing values, now we need to do check for negative values.

(ii)Counting the negative values for item price—

```
> sqldf('SELECT count(item_price) FROM train_data where item_price<0')
  count(item_price)
1                 1
```

As we can see, there is only one item price which is negative in 2.9 million data. So, if we remove it, will not affect our modelling.

(iii)Counting the negative values for item count day –

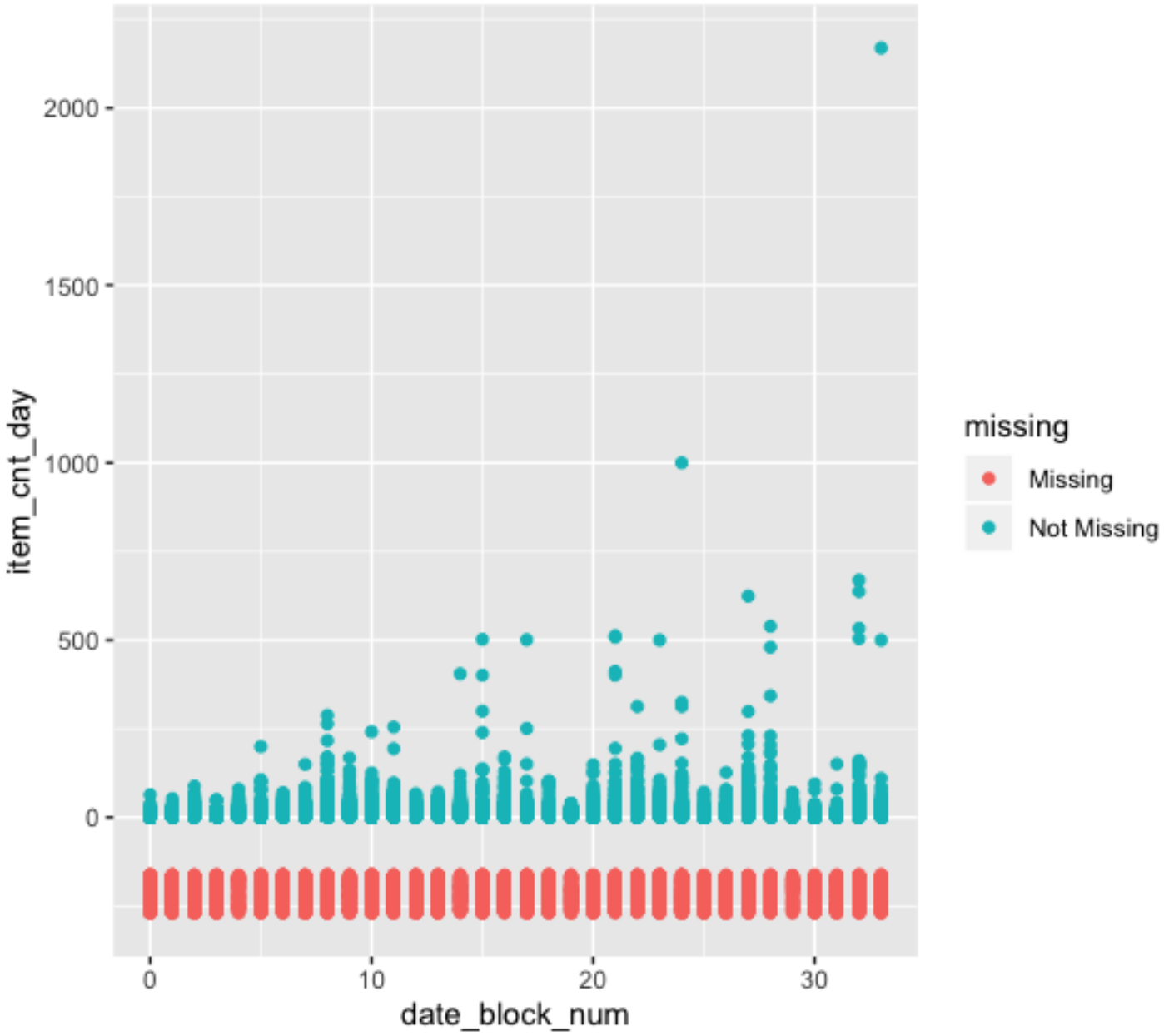
```
> sqldf('SELECT count(item_cnt_day) FROM train_data where item_cnt_day<0')
  count(item_cnt_day)
1                7228
```

There are 7228 negative records in sales data for item count day column. Which is 0.0025% of whole data.
To get more analysis on these negative values, we have aggregated the values of item count day on the basis of shop id and item id, to check whether the negative values is for return of purchased item. And our instinct was correct, negative records reduced to 27 from which maximum are of January 2013 and some are of February 2013, that means the negative count of item was purchased in December 2012 and return in January/February 2013.
So, we kept the negative values, as there is a chance of having negative value in our test data.

```
> sales_grp_data<-sqldf('SELECT shop_id,item_id,avg(item_price) price,SUM(item_cnt_day) as total_count
+                       FROM train_data group by shop_id,item_id
+                      ')
> sales_grp_data<-sqldf('SELECT shop_id,item_id,total_count FROM sales_grp_data where total_count<0')
> str(sales_grp_data)
'data.frame':   27 obs. of  3 variables:
 $ shop_id    : int  4 12 12 12 12 12 12 12 12 12 ...
 $ item_id    : int  12211 1590 1592 1593 8200 9483 9999 11777 15435 19360 ...
 $ total_count: num  -1 -12 -1 -7 -1 -1 -1 -1 -1 -1 ...
```

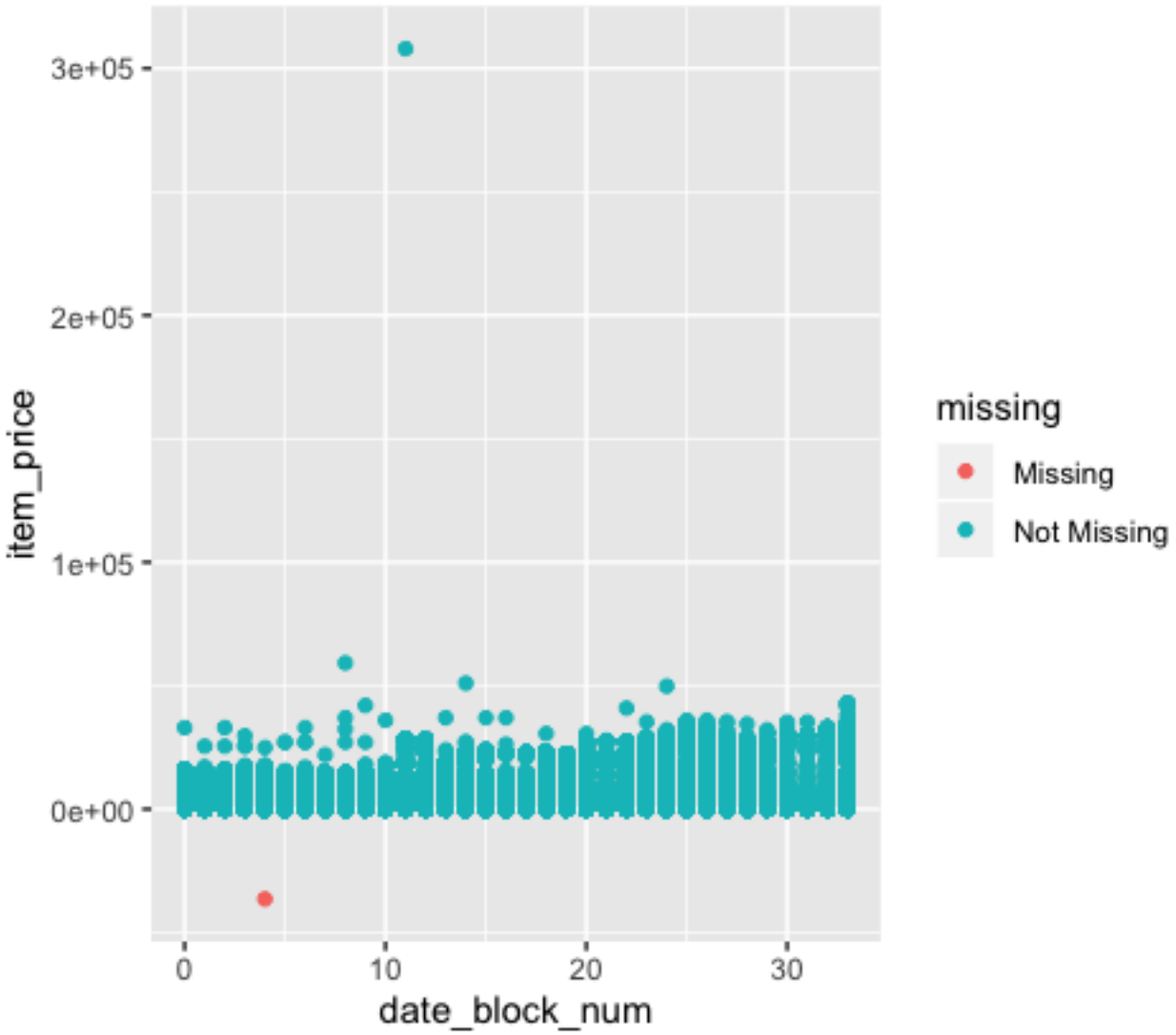
(b)Graphical representation of missing (negative values) data—

(i)item count day negative values



We can see from graph that negative values are present for every month.

(ii)Graph representation for item price—



We can see from graph that there is only one negative value of item price.

(c) Deciding whether to populate the missing data or not—

(i)item count day negative values—After closely viewing data we came to conclusion that some negative values may represent the return of an item. As these values are present for every month (even for our test month October 2015), so removing these values will not give us any benefit.

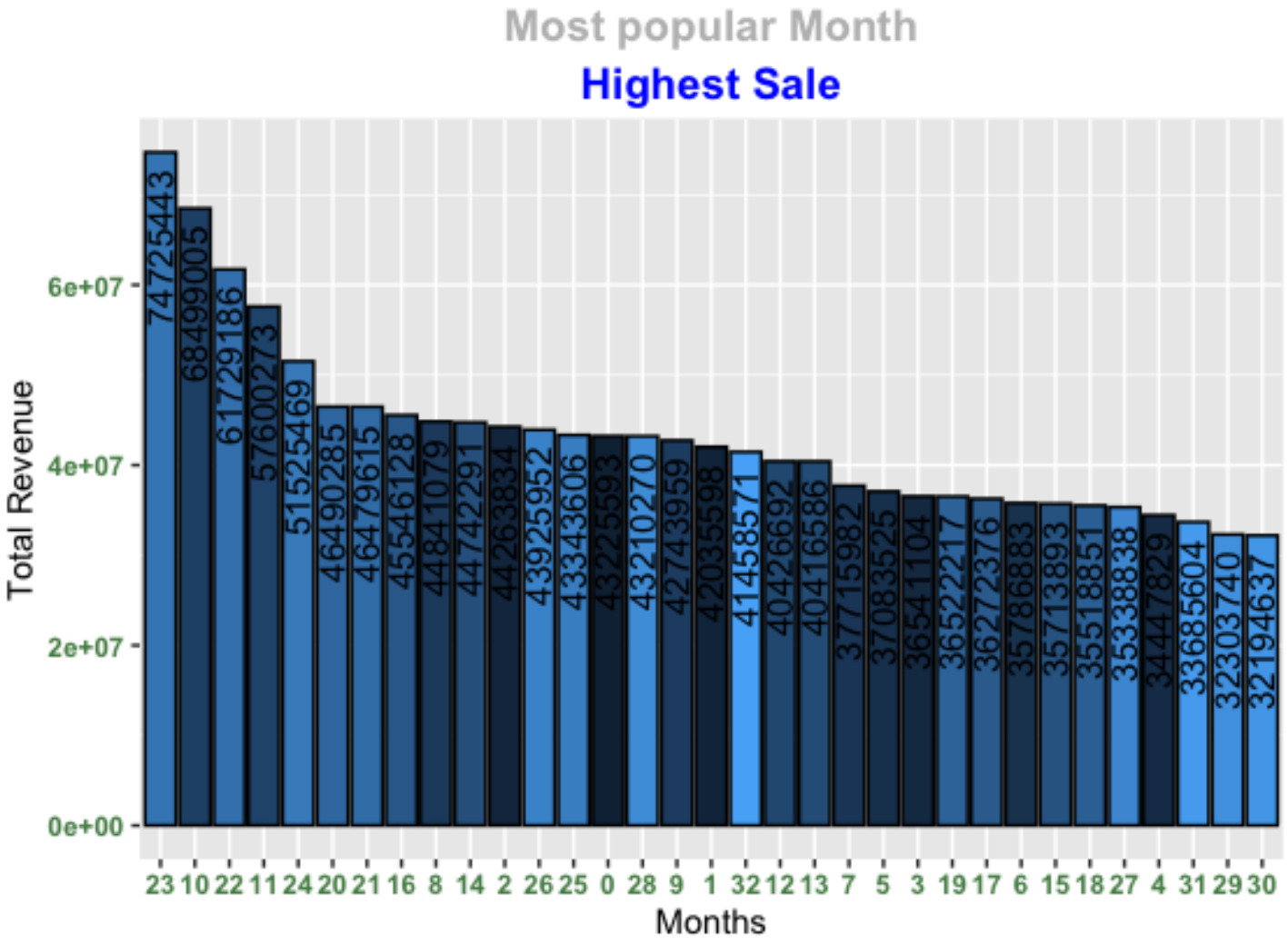
(ii)item price negative -- As item price can't be negative for any item, so we are replacing this negative value with zero.

```
train_data$item_price[train_data$item_price<0]<-0
```

OUTLIER DETECTION

Outlier detection – We need to find out months of sales which can be considered as outlier for any shop.

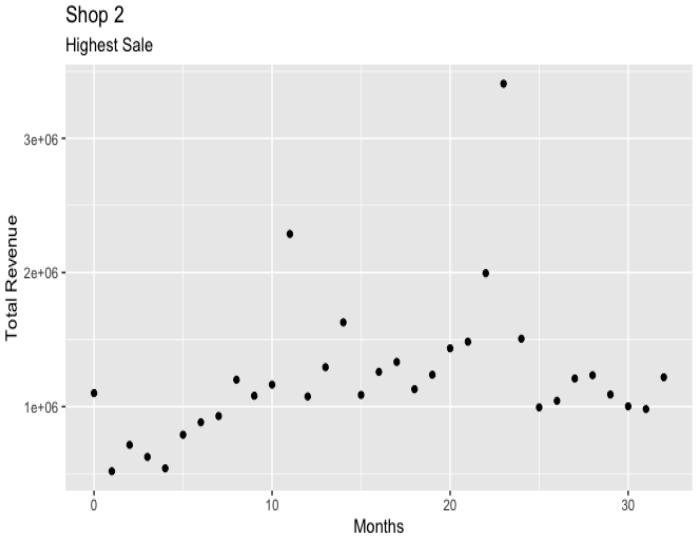
(a) To check this, we need to combine data on the basis of date block month, shop id and item id to get sales for each combination of shop id + item id in each month.

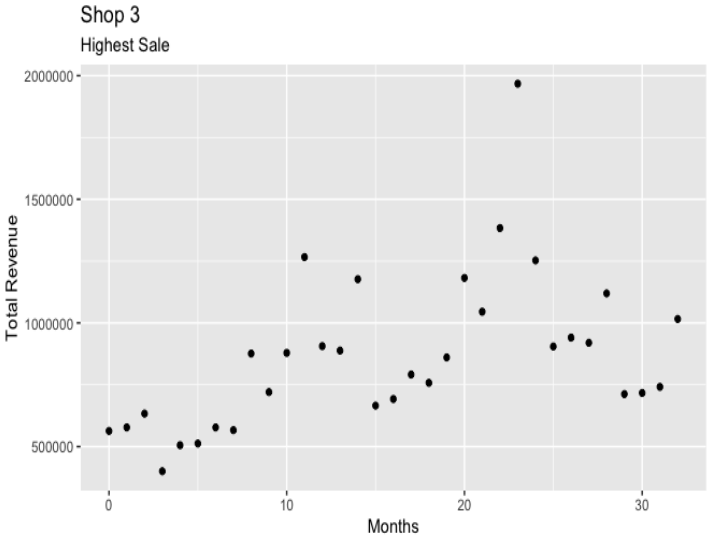


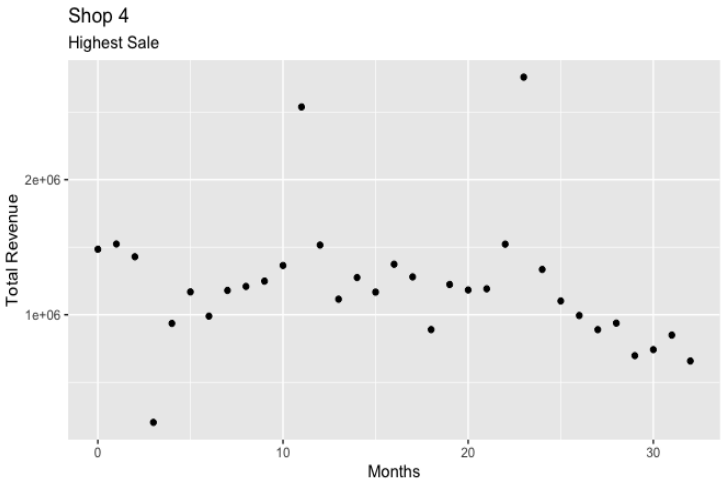
As we can see from graph that highest sale can be found for the months of Nov(2013), Dec(2013),Nov(2014),Dec(2014) are outliers months for any shops due to highest peak of sales as compared to any month.

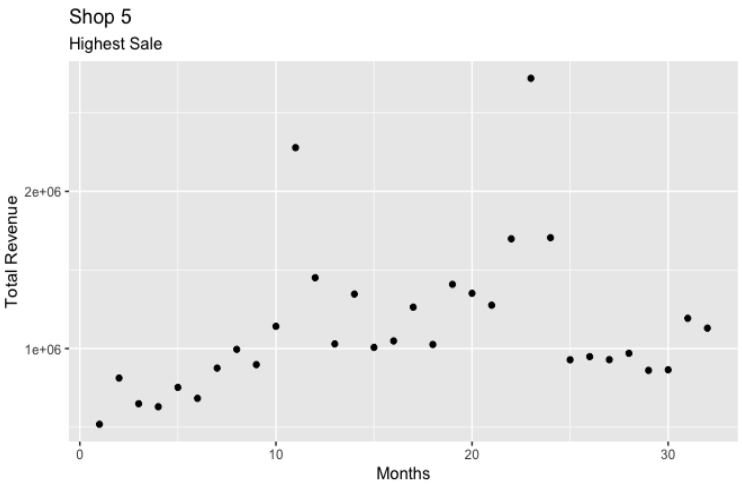
(b)To get insights for each shop id—

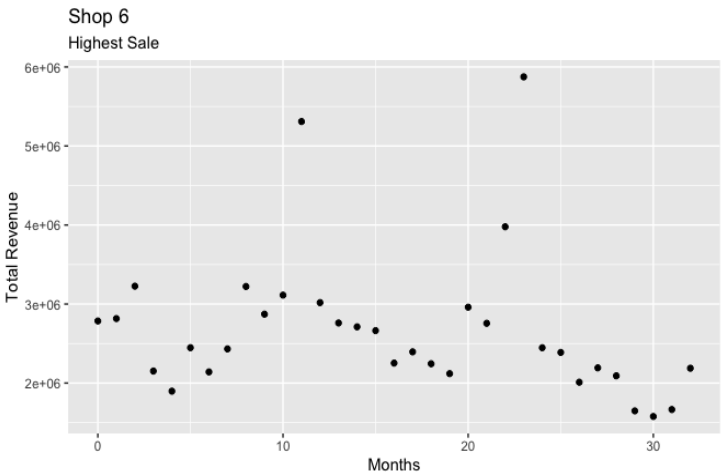
```
query1<-sqldf('SELECT date_block_num,(sum(item_cnt_day) *AVG(item_price)) as Total_revenue
              FROM train_data WHERE shop_id=0 GROUP BY date_block_num ')
ggplot(data=query1,aes(x=date_block_num,y=Total_revenue))+labs(x="Months",y="Total Revenue",
                                                               title="Shop 0", subtitle="Highest Sale",
                                                              fill="Months")+geom_point()
```

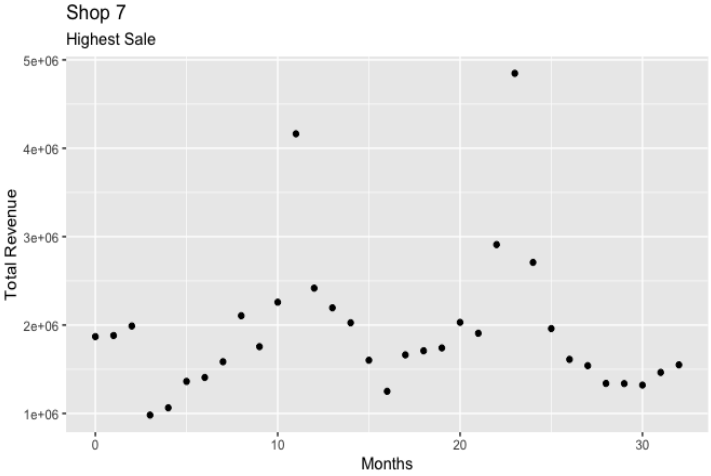






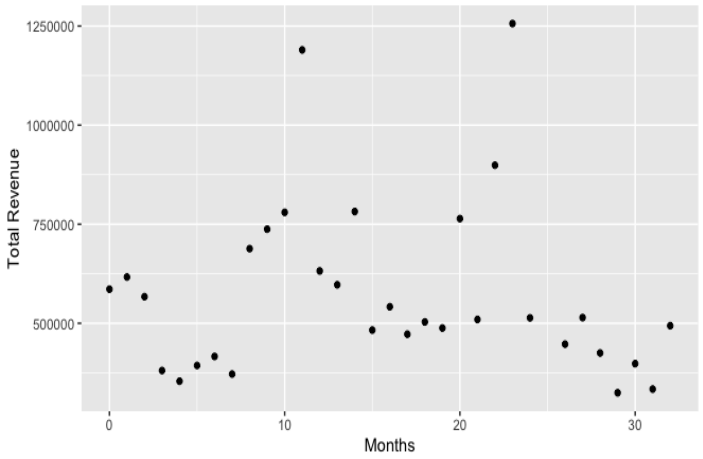






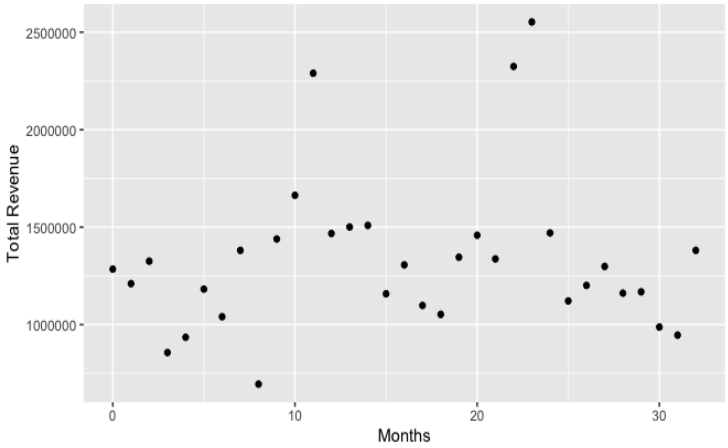
Shop 10

Highest Sale



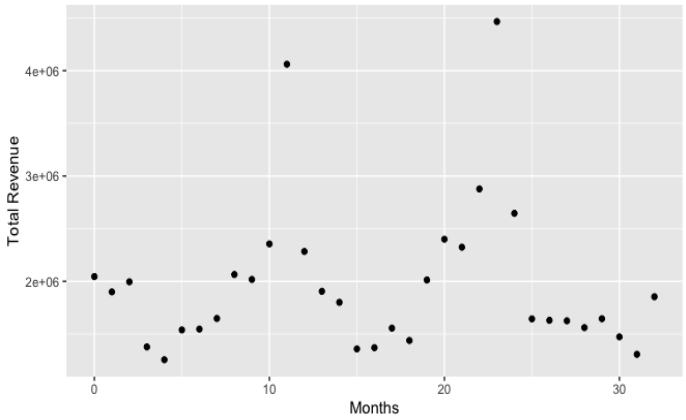
Shop 14

Highest Sale



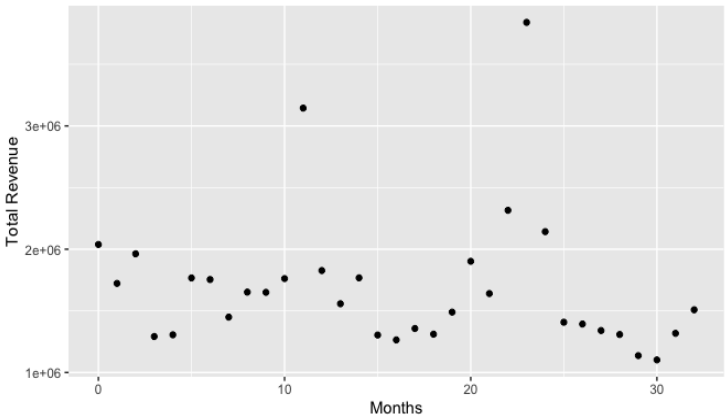
Shop 15

Highest Sale



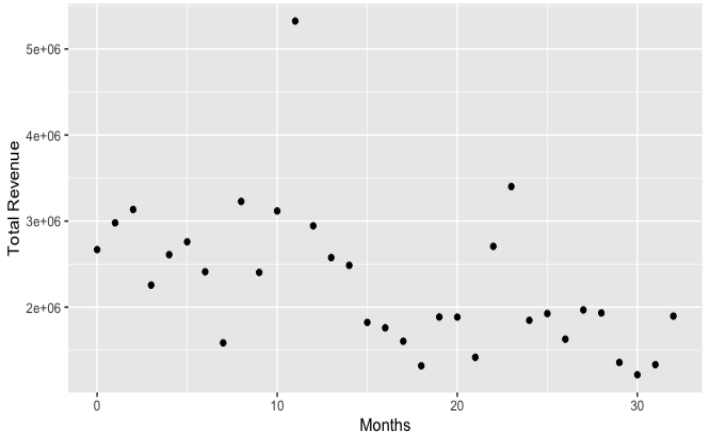
Shop 16

Highest Sale



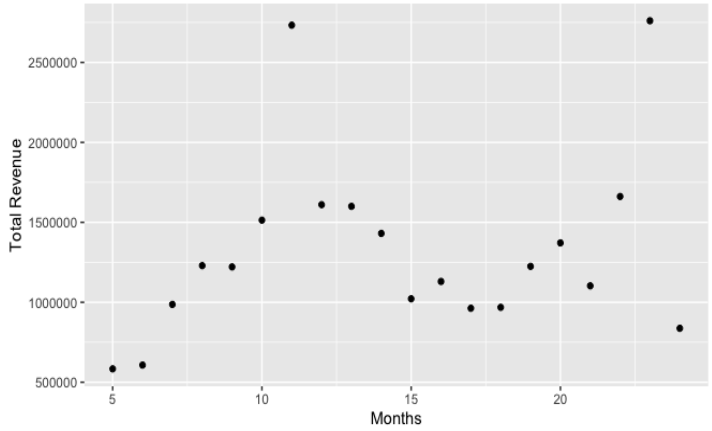
Shop 18

Highest Sale



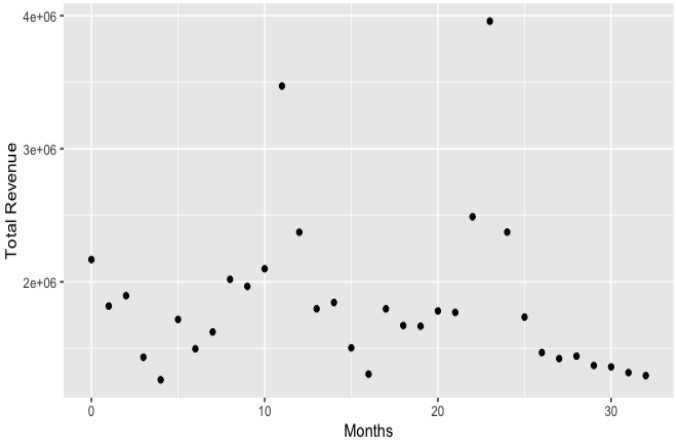
Shop 17

Highest Sale



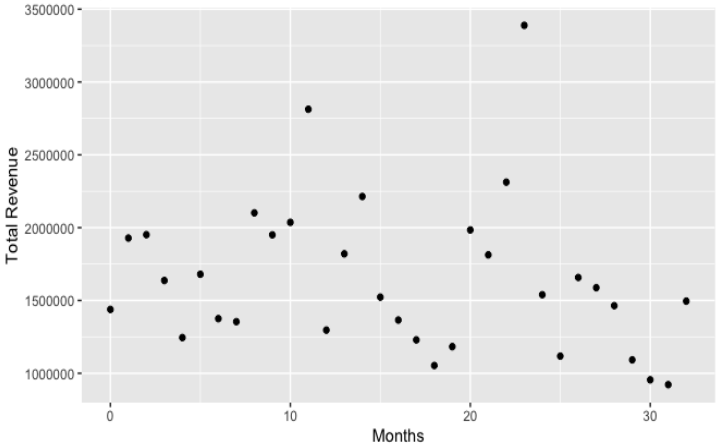
Shop 19

Highest Sale



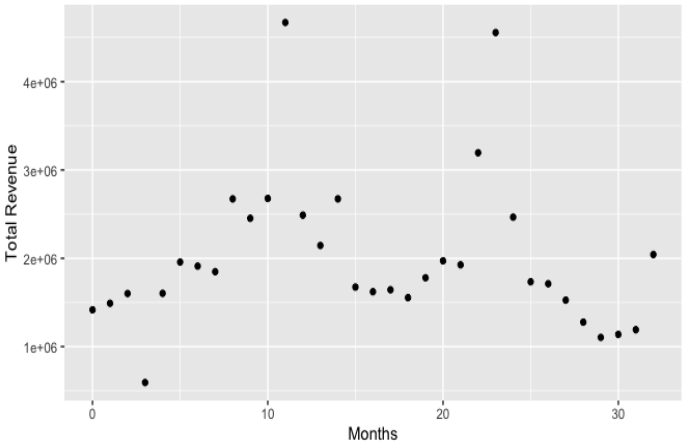
Shop 22

Highest Sale



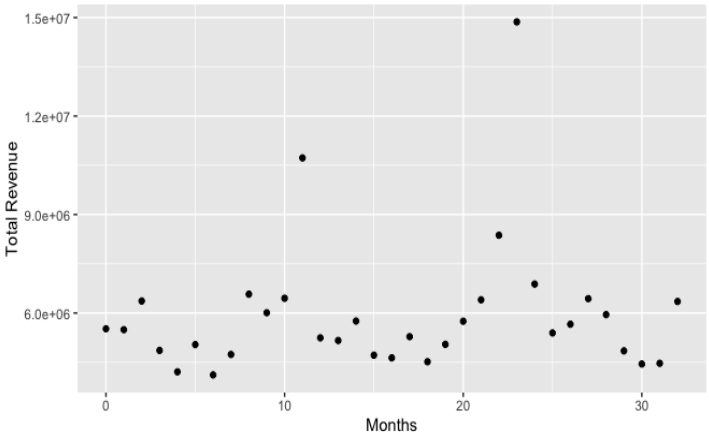
Shop 24

Highest Sale



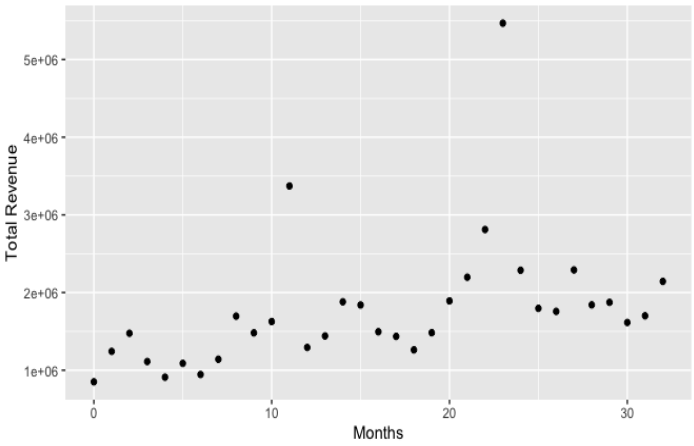
Shop 25

Highest Sale



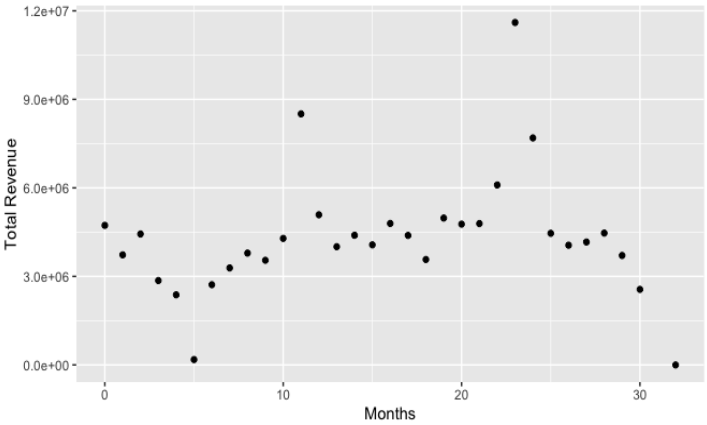
Shop 21

Highest Sale



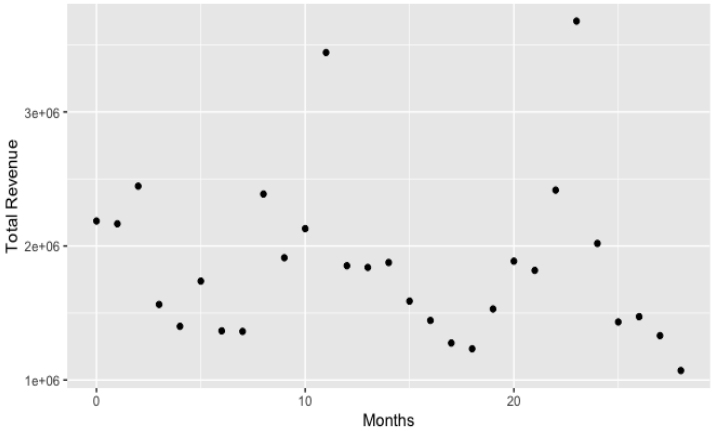
Shop 27

Highest Sale



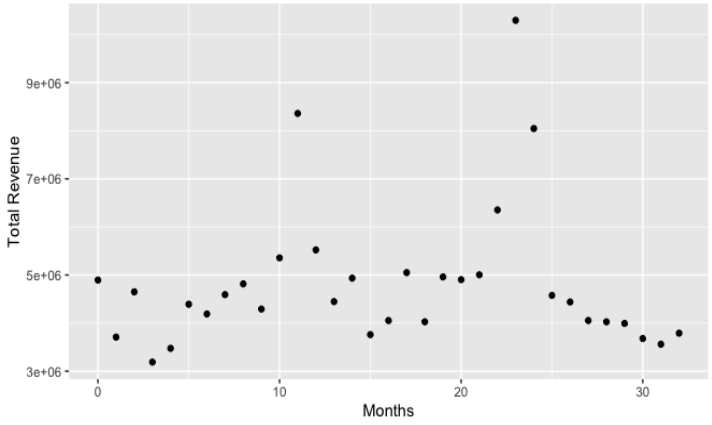
Shop 29

Highest Sale



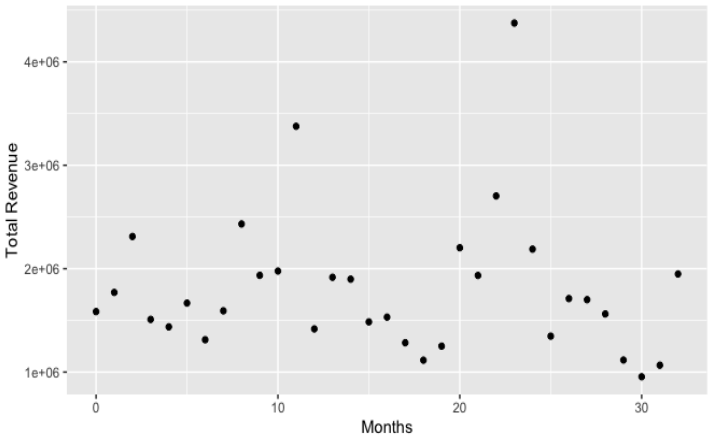
Shop 28

Highest Sale



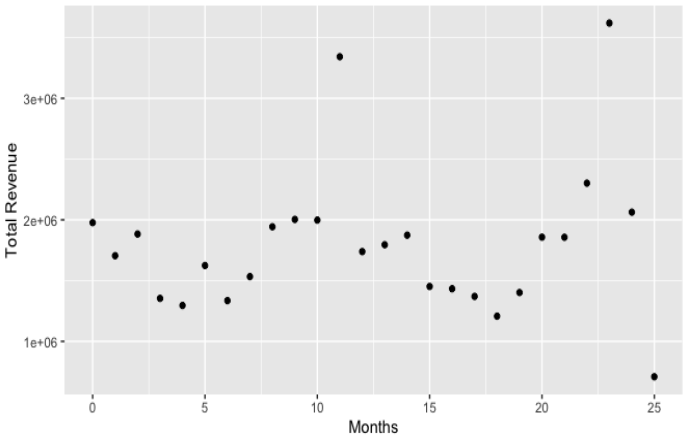
Shop 26

Highest Sale



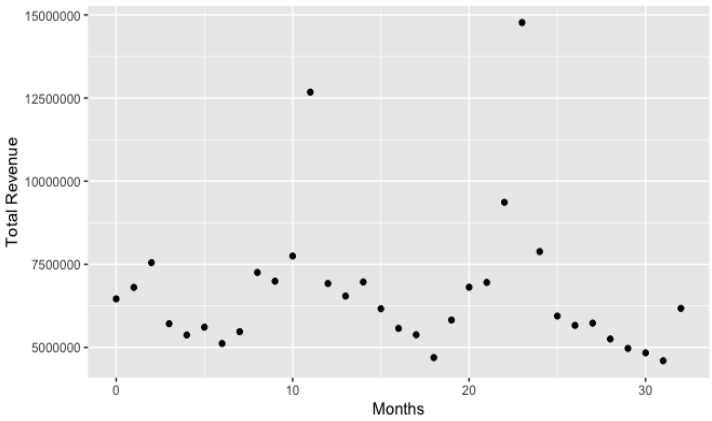
Shop 30

Highest Sale



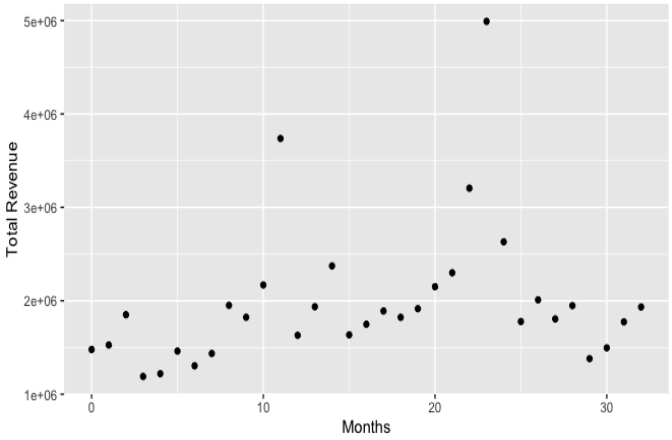
Shop 31

Highest Sale



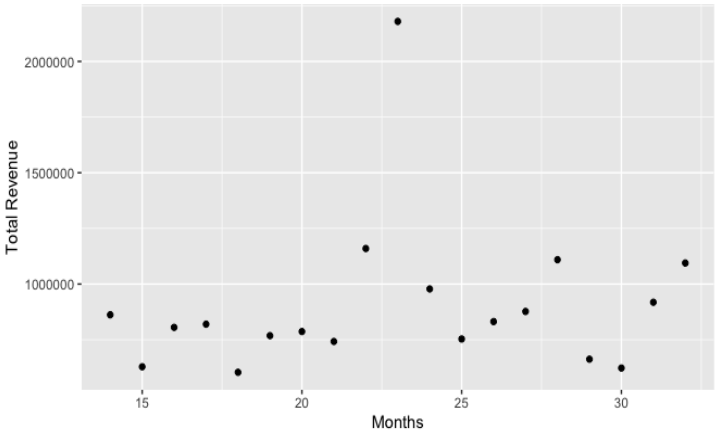
Shop 35

Highest Sale



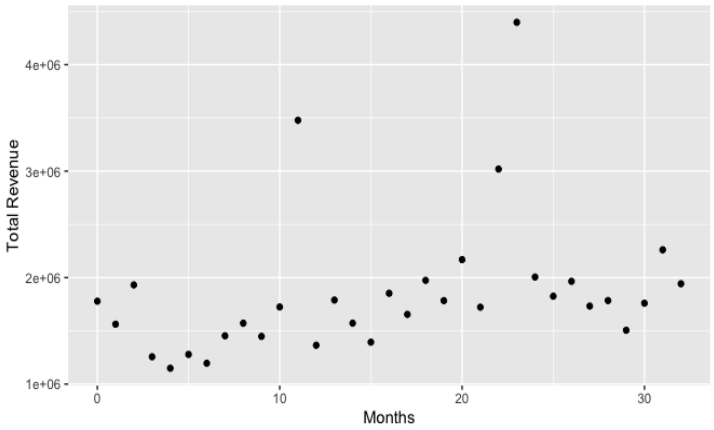
Shop 39

Highest Sale



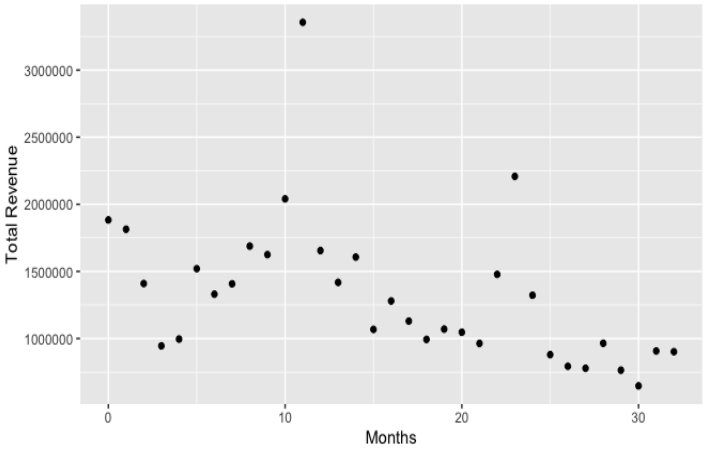
Shop 38

Highest Sale



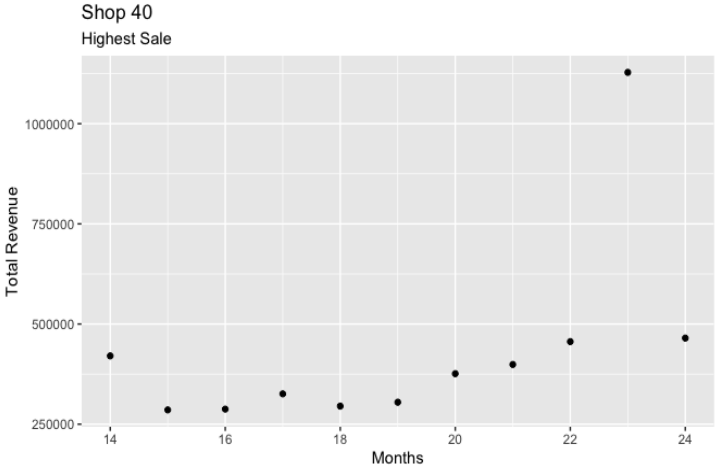
Shop 41

Highest Sale



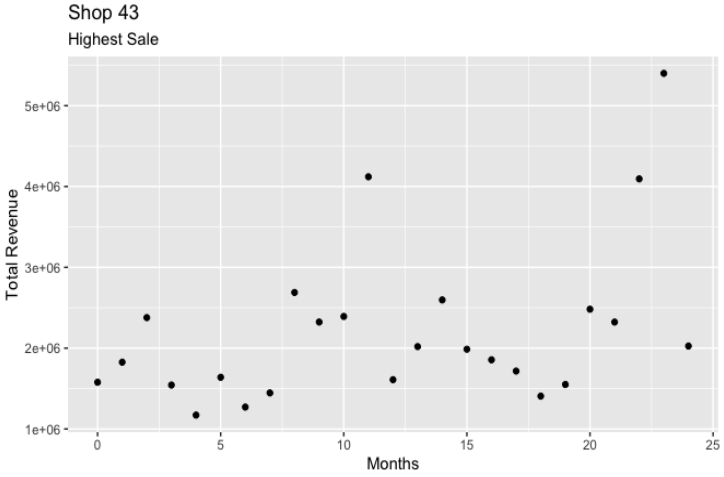
Shop 40

Highest Sale



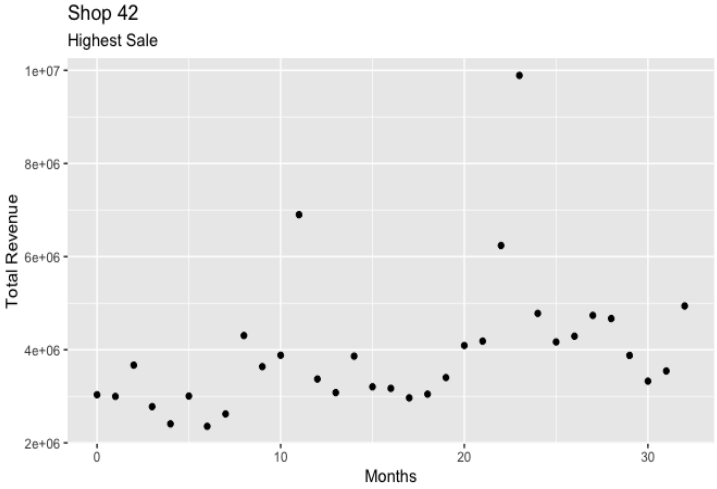
Shop 43

Highest Sale



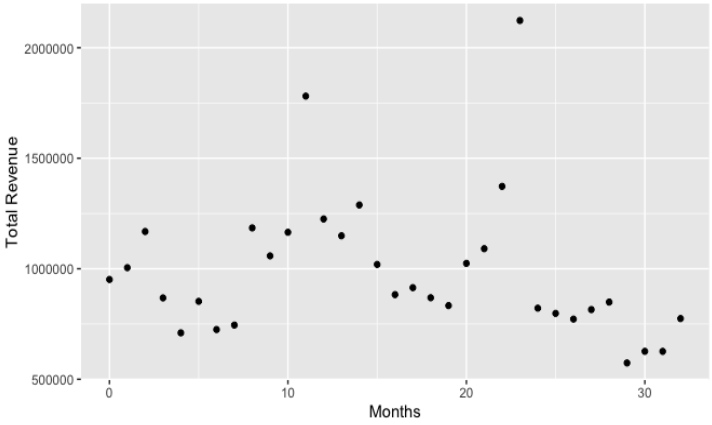
Shop 42

Highest Sale



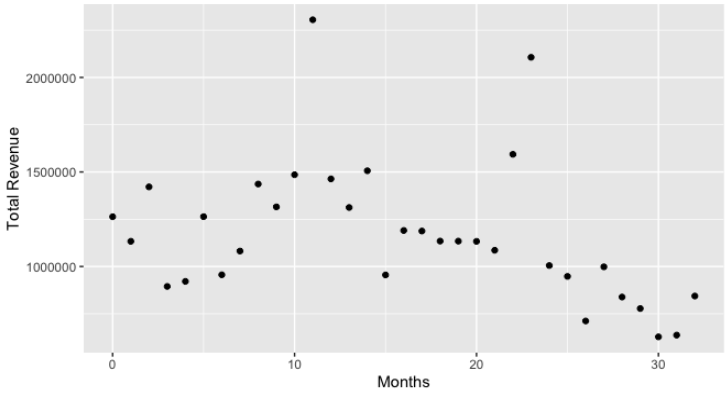
Shop 44

Highest Sale



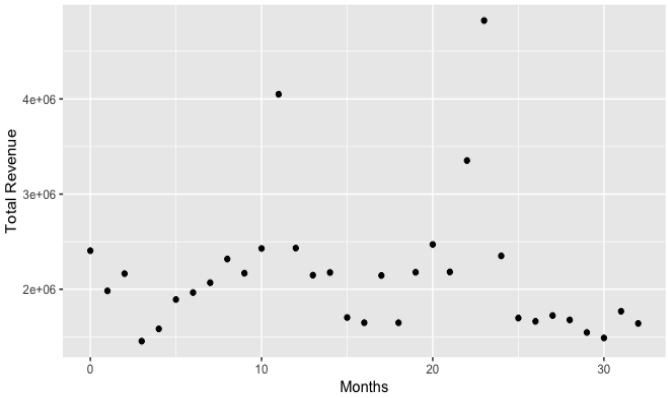
Shop 45

Highest Sale



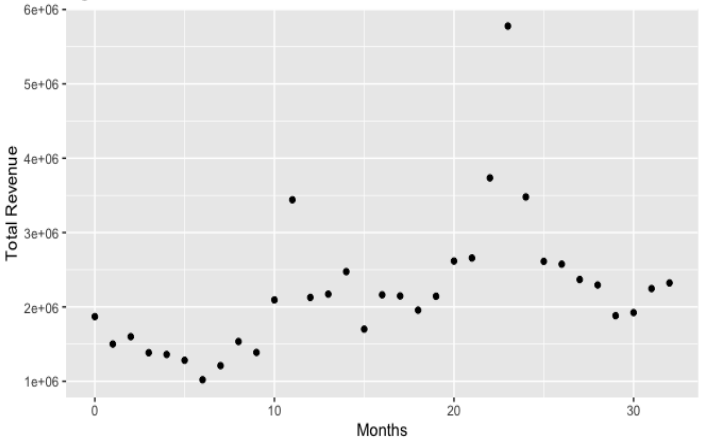
Shop 46

Highest Sale



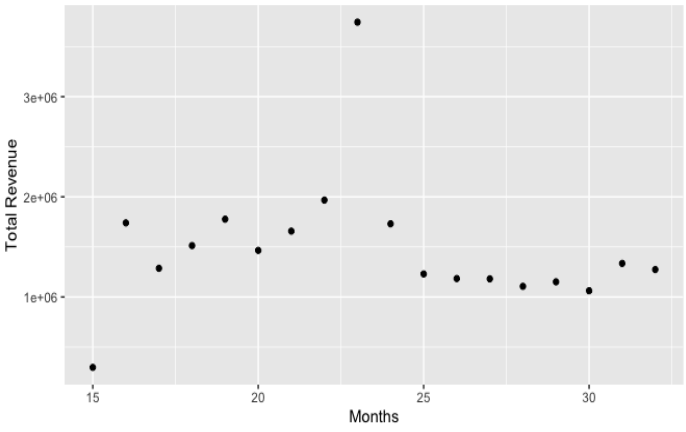
Shop 47

Highest Sale



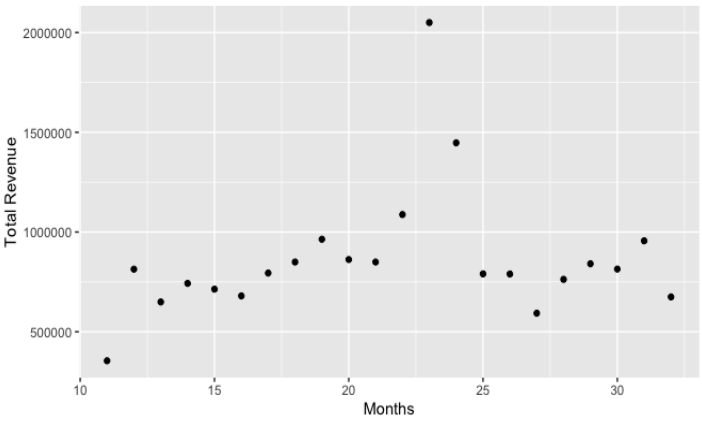
Shop 48

Highest Sale



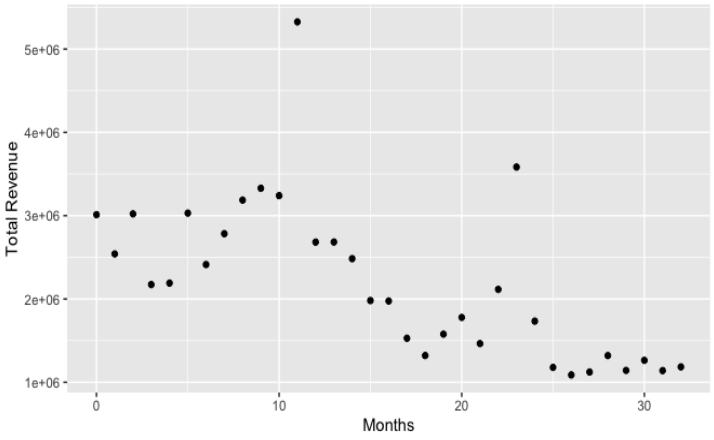
Shop 49

Highest Sale



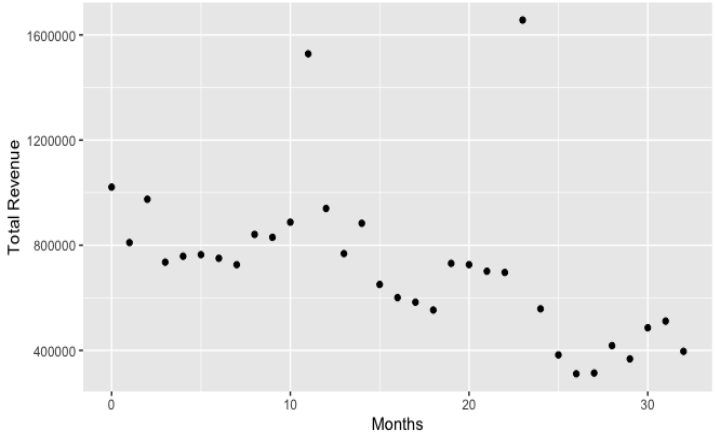
Shop 50

Highest Sale



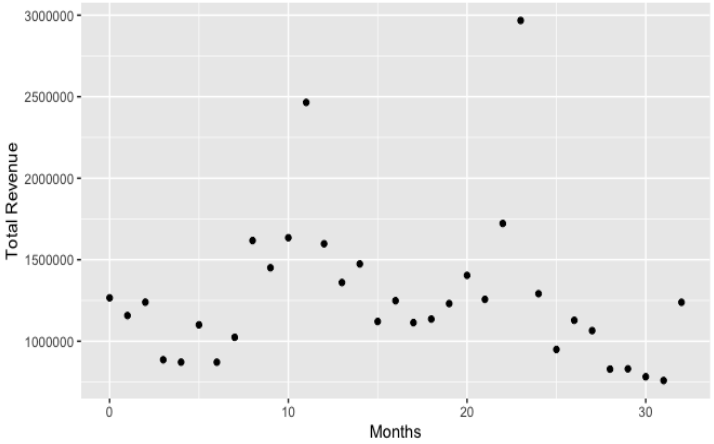
Shop 51

Highest Sale



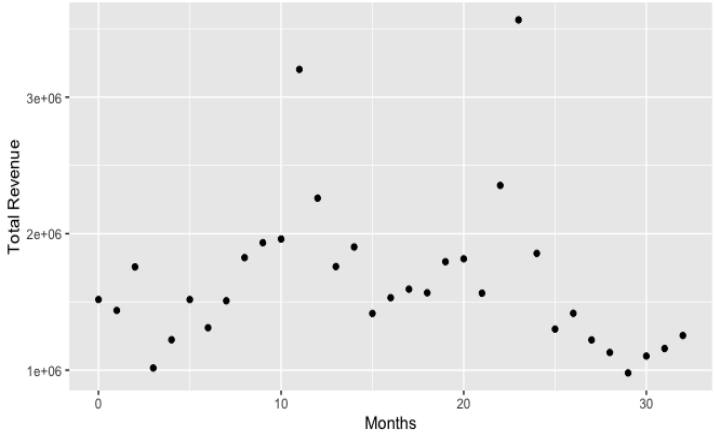
Shop 52

Highest Sale



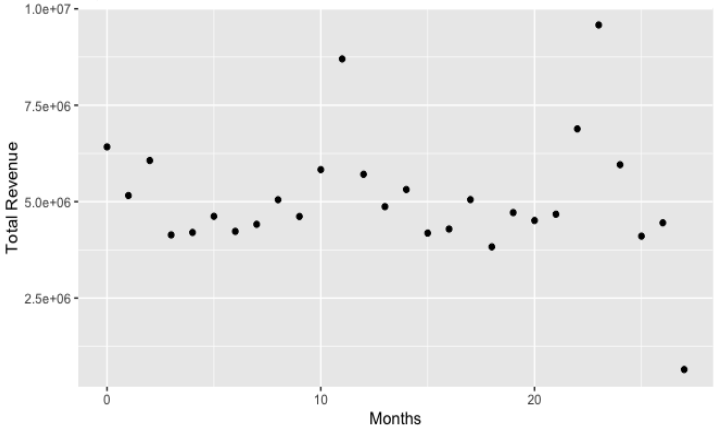
Shop 53

Highest Sale



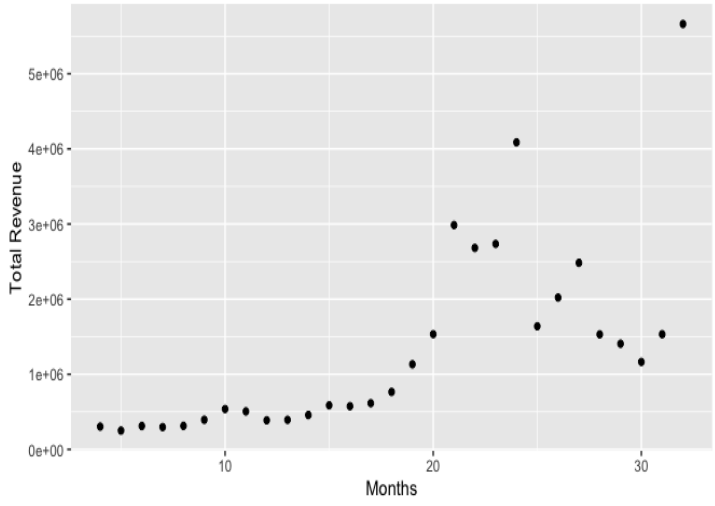
Shop 54

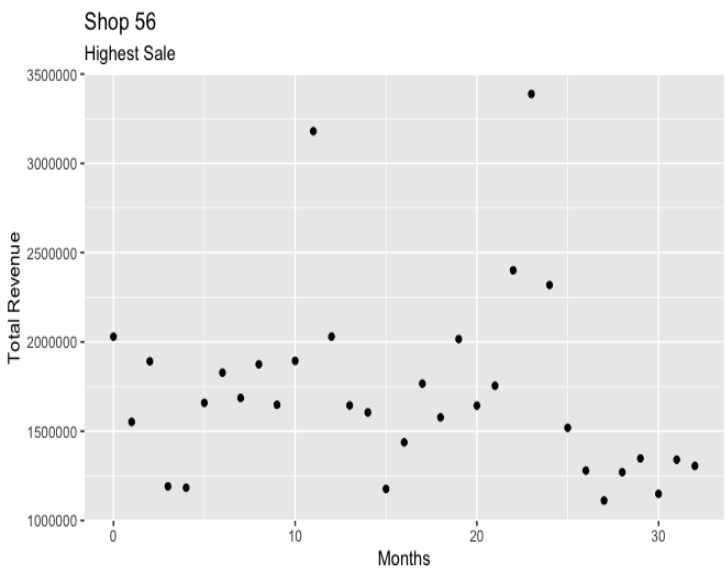
Highest Sale

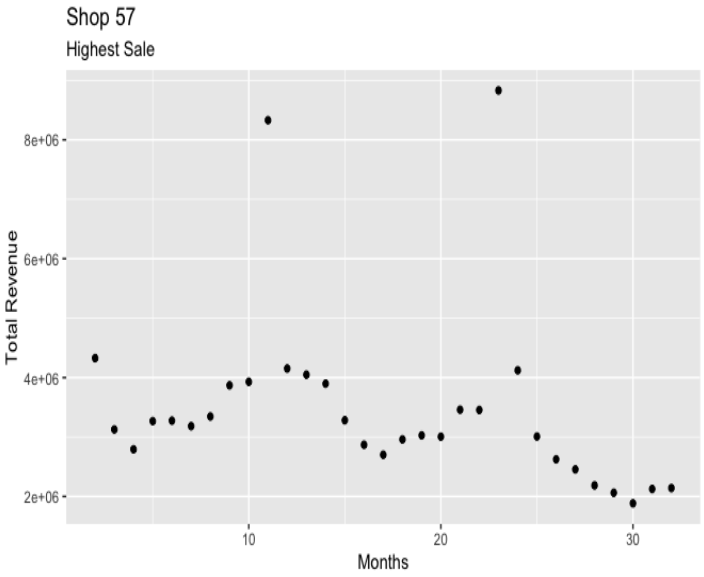


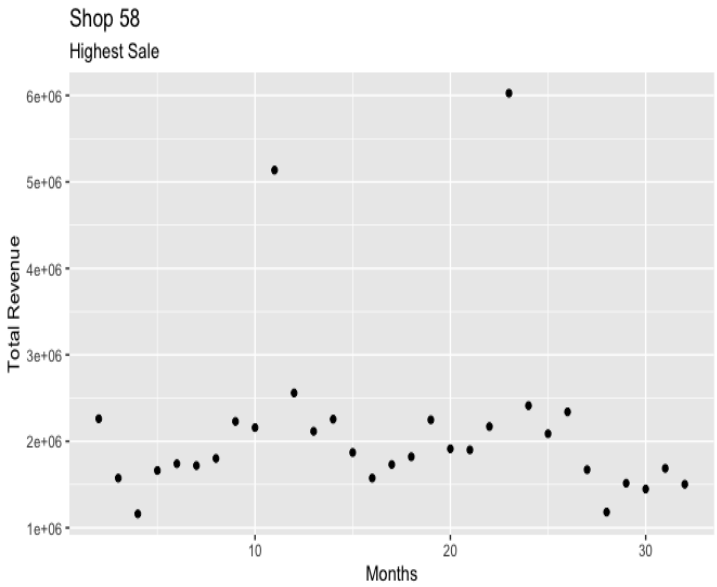
Shop 55

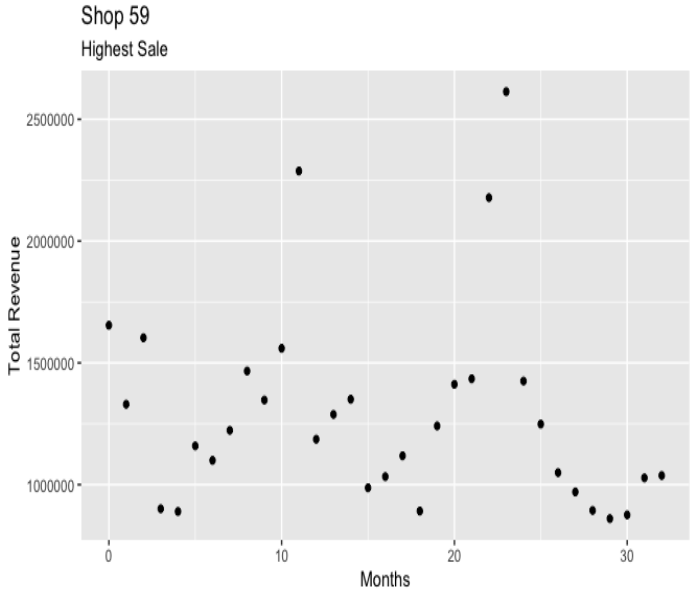
Highest Sale











As we can see from all the plots, highest sales for all shops can be observed for Dec(2013) and Dec(2014).

FEATURE ENGINEERING

(9) Feature Engineering-In this we are adding columns for year, day and month—

```
#Feature engineering adding day,month,year in data
library(lubridate)
#fromatting date
train_data$date<-dmy(train_data$date)
#adding columns
train_data$year<-year(train_data$date)
train_data$month<-month(train_data$date)
train_data$day<-day(train_data$date)
train_data$weekday<-weekdays(train_data$date)
```

(10) Removing unnecessary columns –

```
#removing unnecssary column
train_data$item_name<-NULL
train_data$item_category_name<-NULL
train_data$shop_name<-NULL
train_data$date<-NULL
```

MODELLING

Modelling first on Linear regression model—

Linear regression-it is used to predict the value of an outcome variable Y based on one or more input predictor variables X. The aim is to establish a linear relationship between the predictor variable and response variable.

(

a)linear model—

```
linear_model<-lm( formula=item_cnt_day ~ shop_id + item_id,
                  data=train_data)
```

(b)prediction—

```
pred<-predict(linear_model,test_data[,c("shop_id","item_id")])
```

(c)Adding result into data frame from prediction—

```
result=data.frame(shop_id=test_data$shop_id,item_id=test_data$item_id,item_cnt_pred=pred)
```

Validation of Model— To check validation of model we need to calculate RMSE and MSE values for the predicted values

(a)Calculating RMSE value— RMSE value is the root mean squared error. It is the standard deviation of the residuals (prediction errors). RMSE is the measure of how spread out these residuals are, or it tells you how concentrated the data is around the line of best fit.

```
> library(Metrics)
> rmse(result$item_cnt_pred, test_data$item_cnt_day)
[1] 9.770833
```

(b)MSE— it is mean squared error or mean squared deviation of an estimator measures the average squared difference between the estimated values and the actual value. MSE is risk Function, corresponding to the expected value of the squared error loss.

```
> mse(result$item_cnt_pred, test_data$item_cnt_day)
[1] 95.46918
```

Modelling Second on GBM Model—

GBM(Generalized Boosted Regression Modelling)-

```r
library(tictoc)
library(gbm)
tic("Time Taken to Run GBM Model ")
gbm_model  =  gbm(item_cnt_day ~ shop_id + item_id,
                  data = train_data,
                  shrinkage = 0.01,
                  distribution = "gaussian",
                  n.trees = 1000,
                  interaction.depth = 5,
                  bag.fraction = 0.5,
                  train.fraction = 0.8,
                  # cv.folds = 5,
                  n.cores = -1,
                  verbose = T)

toc()
```

Training on GBM model with 1000 trees:

Iter	TrainDeviance	ValidDeviance	StepSize	Improve
1	3.8312	10.6997	0.0100	0.0083
2	3.8229	10.6925	0.0100	0.0079
3	3.8148	10.6854	0.0100	0.0079
4	3.8069	10.6786	0.0100	0.0078
5	3.7992	10.6719	0.0100	0.0078
6	3.7917	10.6654	0.0100	0.0074
7	3.7844	10.6592	0.0100	0.0074
8	3.7773	10.6532	0.0100	0.0075
9	3.7702	10.6472	0.0100	0.0071
10	3.7631	10.6411	0.0100	0.0070
20	3.7005	10.5898	0.0100	0.0056
40	3.6073	10.5200	0.0100	0.0038
60	3.5455	10.4798	0.0100	0.0025
80	3.5015	10.4545	0.0100	0.0021
100	3.4689	10.4393	0.0100	0.0014
120	3.4447	10.4296	0.0100	0.0012
140	3.4258	10.4230	0.0100	0.0009
160	3.4125	10.4185	0.0100	0.0007
180	3.4029	10.4149	0.0100	0.0005
200	3.3964	10.4121	0.0100	0.0002
220	3.3896	10.4104	0.0100	0.0001
240	3.3830	10.4065	0.0100	0.0002
260	3.3759	10.4026	0.0100	0.0004
280	3.3686	10.4013	0.0100	0.0004
300	3.3633	10.3980	0.0100	0.0003
320	3.3573	10.3966	0.0100	0.0001
340	3.3515	10.3948	0.0100	0.0001
360	3.3459	10.3926	0.0100	0.0003
380	3.3402	10.3908	0.0100	0.0004
400	3.3345	10.3890	0.0100	0.0001
420	3.3298	10.3849	0.0100	0.0004
440	3.3260	10.3802	0.0100	0.0001
460	3.3217	10.3773	0.0100	0.0002
480	3.3178	10.3728	0.0100	0.0001
500	3.3133	10.3668	0.0100	0.0001
520	3.3101	10.3646	0.0100	0.0001
540	3.3059	10.3614	0.0100	0.0003

Time taken to train on GBM Model for 1000 Trees:

```
> toc()
Time Taken to Run GBM Model : 734.506 sec elapsed
```

RMSE Value

```
> pred2 <- predict(gbm_model,newdata = test_data[,c("shop_id","item_id")], n.trees = 1000)
> result2=data.frame(shop_id=test_data$shop_id,item_id=test_data$item_id,item_cnt_day_pred2=pred2)
> rmse(result2$item_cnt_day_pred2, test_data$item_cnt_day)
[1] 9.745973
```

MSE value

```
> mse(result2$item_cnt_day_pred2, test_data$item_cnt_day)
[1] 94.98399
>
```

While modelling we also checked increasing the number of tress from 1000 to 1500 didn't affected the result much so we are going with 1000 trees as this will take less time compared to 1500 trees.

Value of RMSE with 1500 trees:

```
> rmse(result2$item_cnt_day_pred2, test_data$item_cnt_day)
[1] 9.744217
> mse(result2$item_cnt_day_pred2, test_data$item_cnt_day)
[1] 94.94976
```

Confidence interval—

We used K-fold method :

- Split the training data in 10 folds
- Use one fold as test dataset and other 9 as training dataset to train the model
- We did this for every possible test dataset (which would be 10)
- We get 10 RMSE value each for both the models, On which we find confidence interval.

```
> CI(rmse_1$rmse,ci=.95)
   upper     mean    lower
2.507181 2.258884 2.010587
> CI(rmse_1$rmse,ci=.90)
   upper     mean    lower
2.460088 2.258884 2.057680
>
> CI(rmse_1$rmse,ci=.80)
   upper     mean    lower
2.410687 2.258884 2.107081
>
> #MODEL-2
>
> CI(rmse_2$rmse,ci=.95)
   upper     mean    lower
2.438250 2.182684 1.927118
>
> CI(rmse_2$rmse,ci=.90)
   upper     mean    lower
2.389779 2.182684 1.975589
>
> CI(rmse_2$rmse,ci=.80)
   upper     mean    lower
2.338931 2.182684 2.026437
```

Comparing two models—

(a)Error—comparing the RMSE Error both the models, the error for GBM model is slightly less than Linear regression Model

RMSE value for GBM Model—

```
> pred2 <- predict(gbm_model,newdata = test_data[,c("shop_id","item_id")], n.trees = 1000)
> result2=data.frame(shop_id=test_data$shop_id,item_id=test_data$item_id,item_cnt_day_pred2=pred2)
> rmse(result2$item_cnt_day_pred2, test_data$item_cnt_day)
[1] 9.745973
```

RMSE value for Linear regression Model—

```
> library(Metrics)
> rmse(result$item_cnt_pred, test_data$item_cnt_day)
[1] 9.770833
```

(b)Efficiency in training time

Time taken in Training by Linear regression Model—

```
> toc()
time taken by linear regression: 3.324 sec elapsed
```

Time taken in Training by GBM Model—

```
> toc()
Time Taken to Run GBM Model : 734.506 sec elapsed
```

As we can see the efficiency of training time for Linear regression Model is better. Although, if we decrease the number of trees in GBM model, the training time decreases but there is slight increase in RMSE value.

References:
1) https://towardsdatascience.com/
2) https://rpubs.com/
3) https://github.com/
4) https://stattrek.com/regression/slope-confidence-interval.aspx
5)http://sphweb.bumc.bu.edu/otlt/MPHModules/BS/BS704_Confidence_Intervals/BS704_Confidence_Intervals_print.html
6) https://www.kaggle.com/
7) https://medium.com/