PREDICT FUTURE SALES USING LINEAR REGRESSION

CS636: DATA ANALYTICS WITH R PROGRAMMING

PROJECT OUTLINE

- In this project we work with a challenging time-series dataset consisting of daily sales data, provided by one of the largest Russian software firms - 1C Company.
- The prediction will be done based on the sales data and the analysis of the sales that were held on the yearly, monthly and daily basis.
- The items will be analyzed from the highest sold to the lowest with the market value associated with it
- Linear Regression Model , a machine learning algorithm will be applied after the data being merged to form the prediction based on the existing old data of the sales and predictions will be made accordingly.

DATA SETS

File descriptions

- **sales_train.csv** - the training set. Daily historical data from January 2013 to October 2015.
- **test.csv** - the test set. You need to forecast the sales for these shops and products for November 2015.
- **sample_submission.csv** - a sample submission file in the correct format.
- **items.csv** - supplemental information about the items/products.
- **item_categories.csv** - supplemental information about the items categories.
- **shops.csv** - supplemental information about the shops.

DATA FIELDS

- **ID** - an Id that represents a (Shop, Item) tuple within the test set
- **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

LIBRARIES

- tidyverse
- tidyr
- tidyselect
- plotly
- dplyr
- reactable
- htmlwidgets
- IRdisplay

Models Used

Linear Regression Model

- Linear regression is one of the easiest and most popular Machine Learning algorithms.
- It is a statistical method that is used for predictive analysis.
- Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.
- A linear regression model means estimating the values of the coefficients used in the representation with the data that we have available.

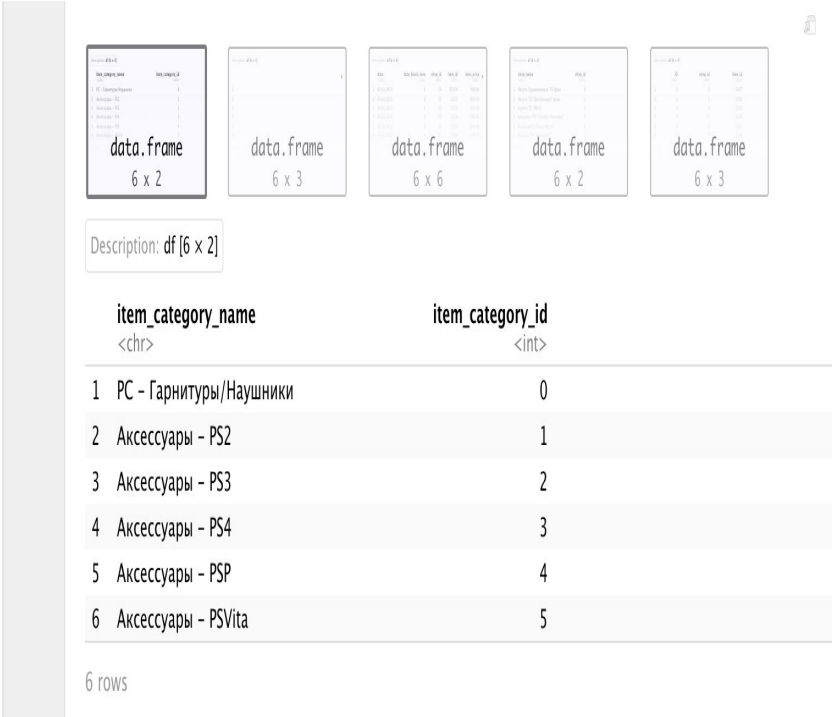
Removing the Missing Values

With the help of the drop na function with the respective data we removed the unnecessary data which can harm the analysis in future.

```
# REMOVE MISSING VALUES
item_categories<-drop_na(item_categories)

items<-drop_na(items)

sales_train<-drop_na(sales_train)

shops<-drop_na(shops)

test<-drop_na(test)
```

The dataset item categories have the name and the id assigned respectively with no NA.

Item categories

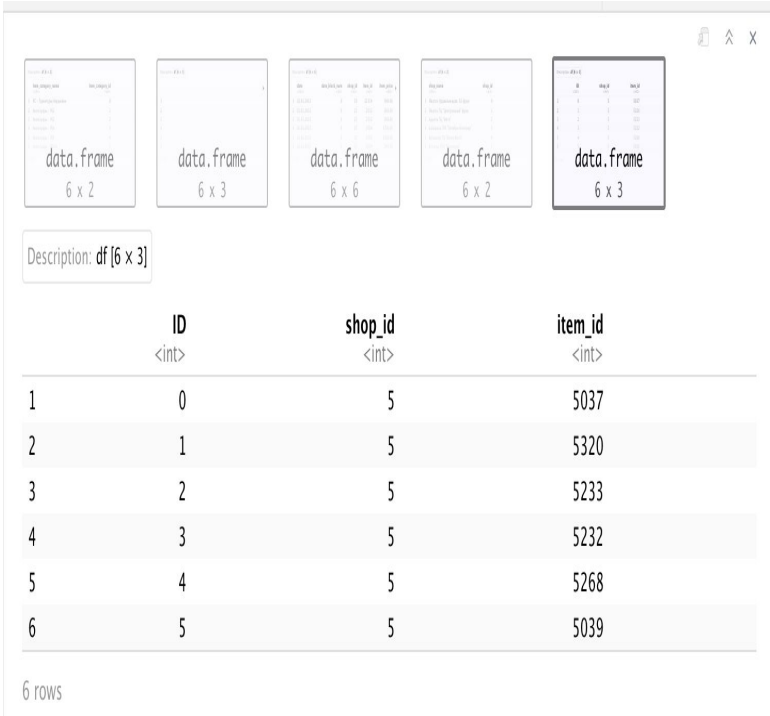


	item_category_name <chr>	item_category_id <int>
1	PC - Гарнитуры/Наушники	0
2	Аксессуары - PS2	1
3	Аксессуары - PS3	2
4	Аксессуары - PS4	3
5	Аксессуары - PSP	4
6	Аксессуары - PSVita	5

6 rows

The test dataset have been shown as above with the ID and the shop id and the item id respectively.

Test



	ID <int>	shop_id <int>	item_id <int>
1	0	5	5037
2	1	5	5320
3	2	5	5233
4	3	5	5232
5	4	5	5268
6	5	5	5039

6 rows

Data Visualizations

A. Total Product Sold Per Year : Using the group by function on the year features and summarizing the data using the plotly function on the year and the product sold per year we get the plot. According to the plot the 2013 have the maximum sales as compared to the 2014 and the 2015 .

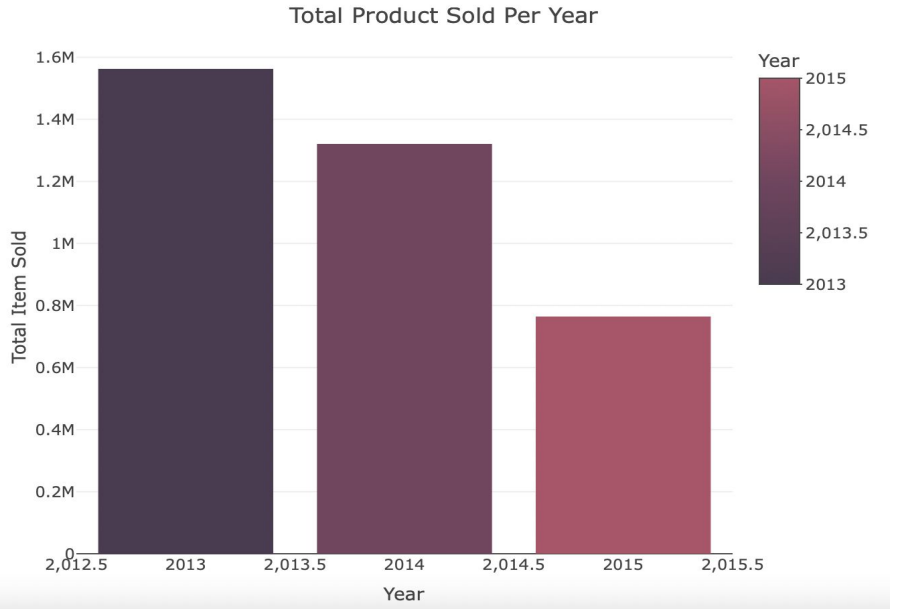
```r
# TOTAL PRODUCT SOLD PER YEAR
product_sold_Year<-sales_data%>%group_by(Year)%>%summarise(Product_sold_year=sum(item_cnt_day))

product_sold_Year

plot_ly(product_sold_Year, x=~Year, y=~Product_sold_year, color = ~Year, colors =
c("#4C3A51","#774360","#B25068"), type='bar')%>%layout(title='Total Product Sold Per Year',
yaxis=list(title='Total Item Sold'))
```

tbl_df 3 x 2 | R Console | plotly htmlwidget

A tibble: 3 × 2

Year <dbl>	Product_sold_year <dbl>
2013	1562733
2014	1320889
2015	764584

3 rows



B. Total Product Sold Per Month : The total product sold per month is being formed by extracting the data by month wise. For the sum of the item sold we form the graph using the plotly function where the Product sold per month and the month is being considered. According to the graph the month 1 have the highest which goes on decreasing by month 10 and then increasing afterwards.

Source Visual Outline

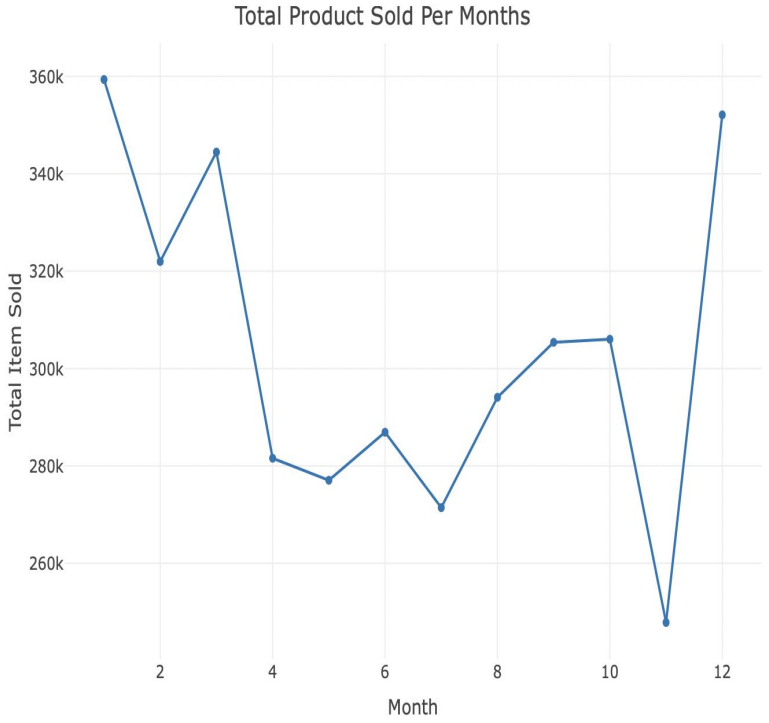
```r
# TOTAL PRODUCT SOLD PER MONTH
product_sold_Month<-sales_data%>%group_by(Month)%>%summarise(Product_sold_Month=sum(item_cnt_day))

head(product_sold_Month)
plot_ly(product_sold_Month, y=~Product_sold_Month, x=~Month,  type='scatter',
mode='lines+markers')%>%layout(title='Total Product Sold Per Months', yaxis=list(title='Total Item Sold'))

```

tbl_df 6 x 2 | plotly htmlwidget

A tibble: 6 × 2

Month <dbl>	Product_sold_Month <dbl>
1	359349
2	321975
3	344453
4	281573
5	277055
6	286924

6 rows



C.Total Product Sold Per Day : The total product sold per day is being analysed by grouping the day and then sum of the product per day and with the help of the plot ly function the product sold per day and the day is being plotted which states that the when the day increases to 20 the total sales of the product increases till 30 day after that it decreases.

```r
# TOTAL PRODUCT SOLD PER DAY
product_sold_Day<-sales_data%>%group_by(Day)%>%summarise(Product_sold_day=sum(Day))

head(product_sold_Day)

plot_ly(product_sold_Day, x=~Day, y=~Product_sold_day, type='scatter',
mode='lines+markers')%>%layout(title='Total Product Sold Per Day',yaxis=list(title='Total Item Sold'))
```

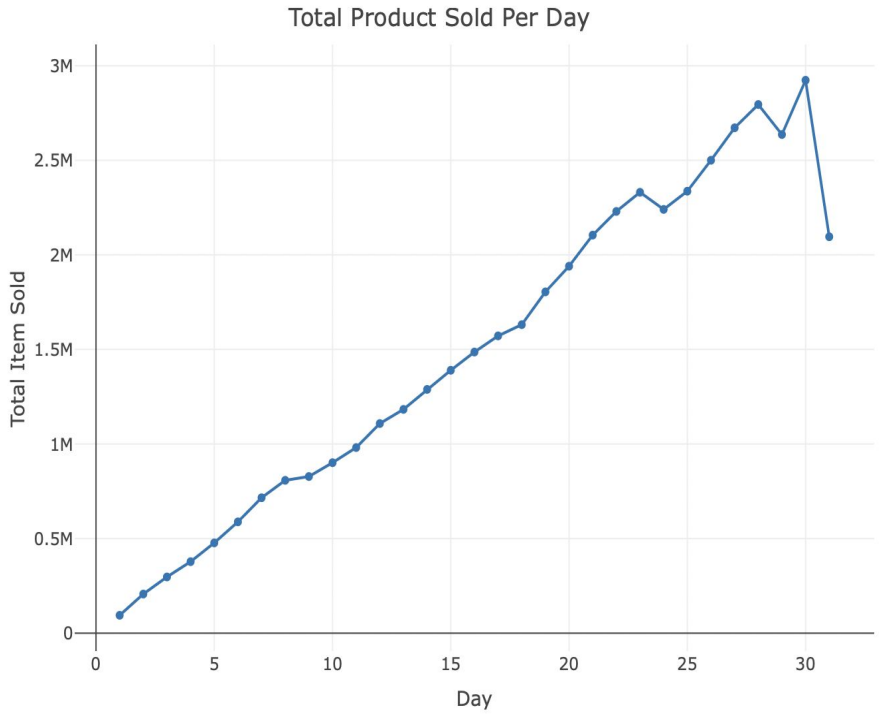
tbl_df
6 x 2

plotly
htmlwidget

A tibble: 6 x 2

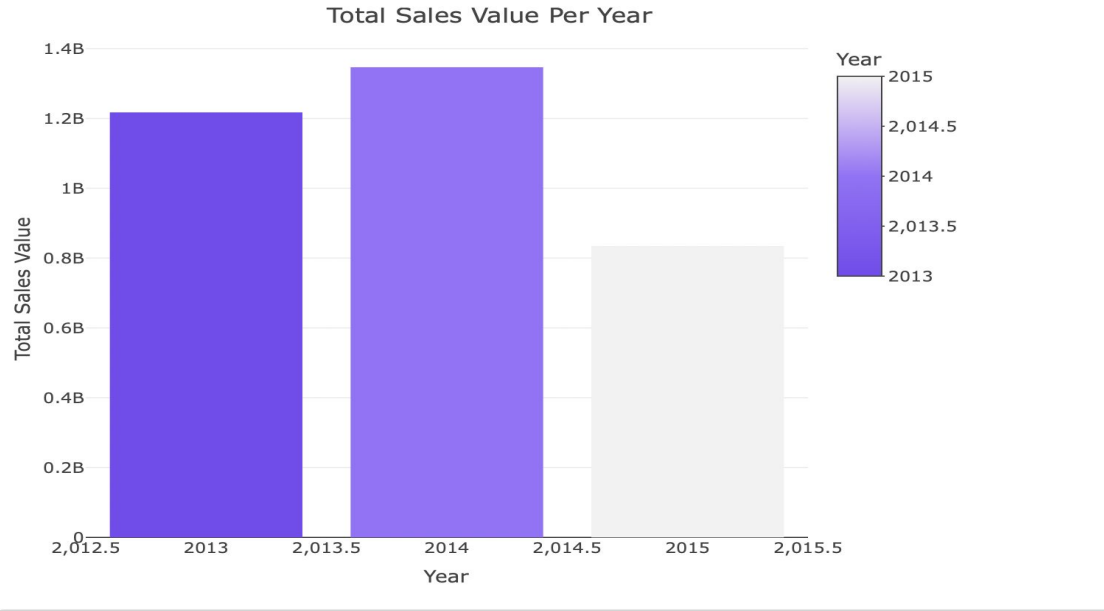
Day <dbl>	Product_sold_day <dbl>
1	94421
2	206744
3	297081
4	377876
5	477180
6	588348

6 rows



Total Sales Per Year

According to the Total sales per year that have been made by grouping the year and then forming the plot depicts the total sales of the price was maximum in 2014 and minimum in 2015.



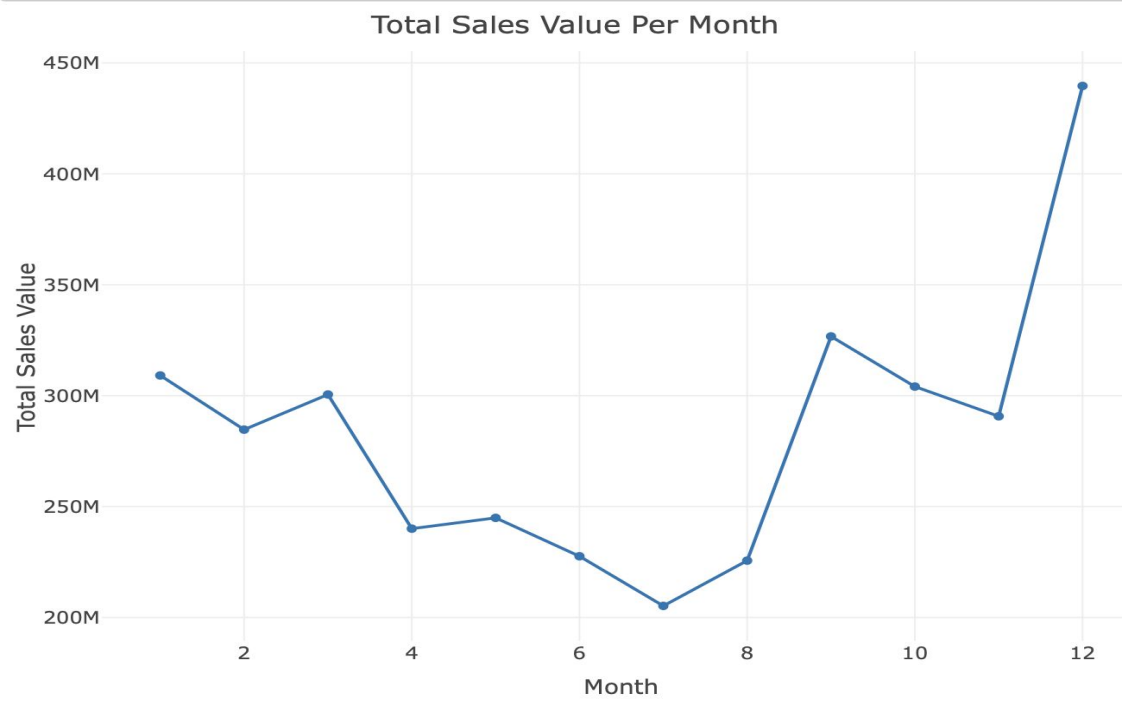
A tibble: 3 x 2

Year <dbl>	Sales_value_year <dbl>
2013	1217524734
2014	1346778479
2015	834623132

3 rows

Total Sales Per Month

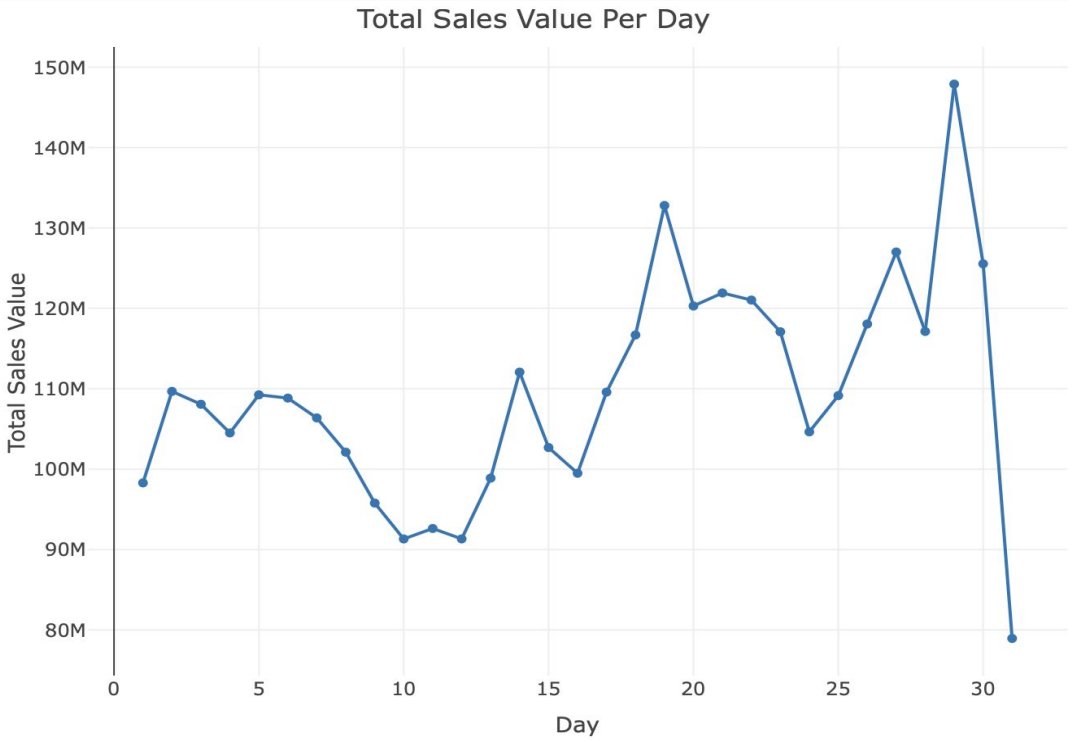
The total sales per month depicts the month 2 and 4 and the 6 months have the highest sales as compared to the other months.



Month <dbl>	Sales_value_month <dbl>
1	309100814
2	284690714
3	300524359
4	240058855
5	244924485
6	227616940

Total Sales Per Day

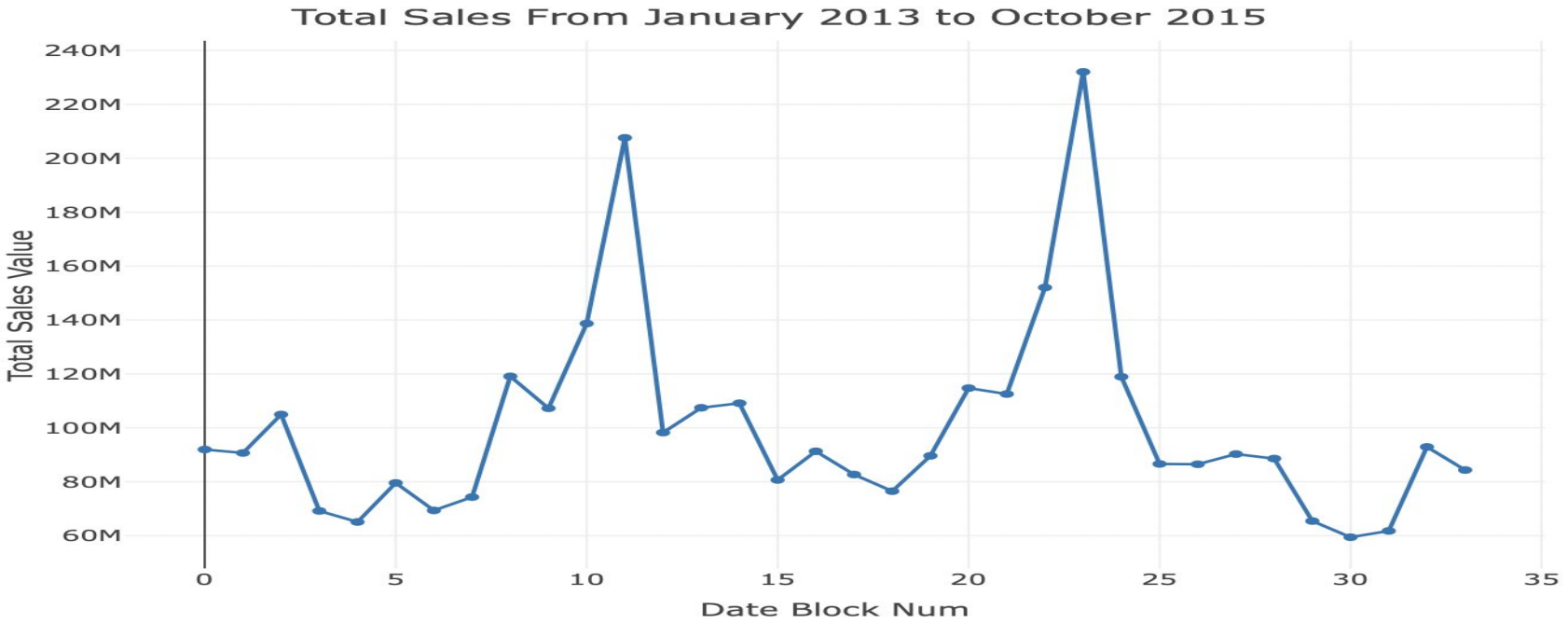
The Total sales per day is being said to highest in the days 2, 4 and 6 and the after 30 days it's set to decreasing order.



Day <dbl>	Sales_value_day <dbl>
1	98287746
2	109669776
3	108069701
4	104503536
5	109241616
6	108834005

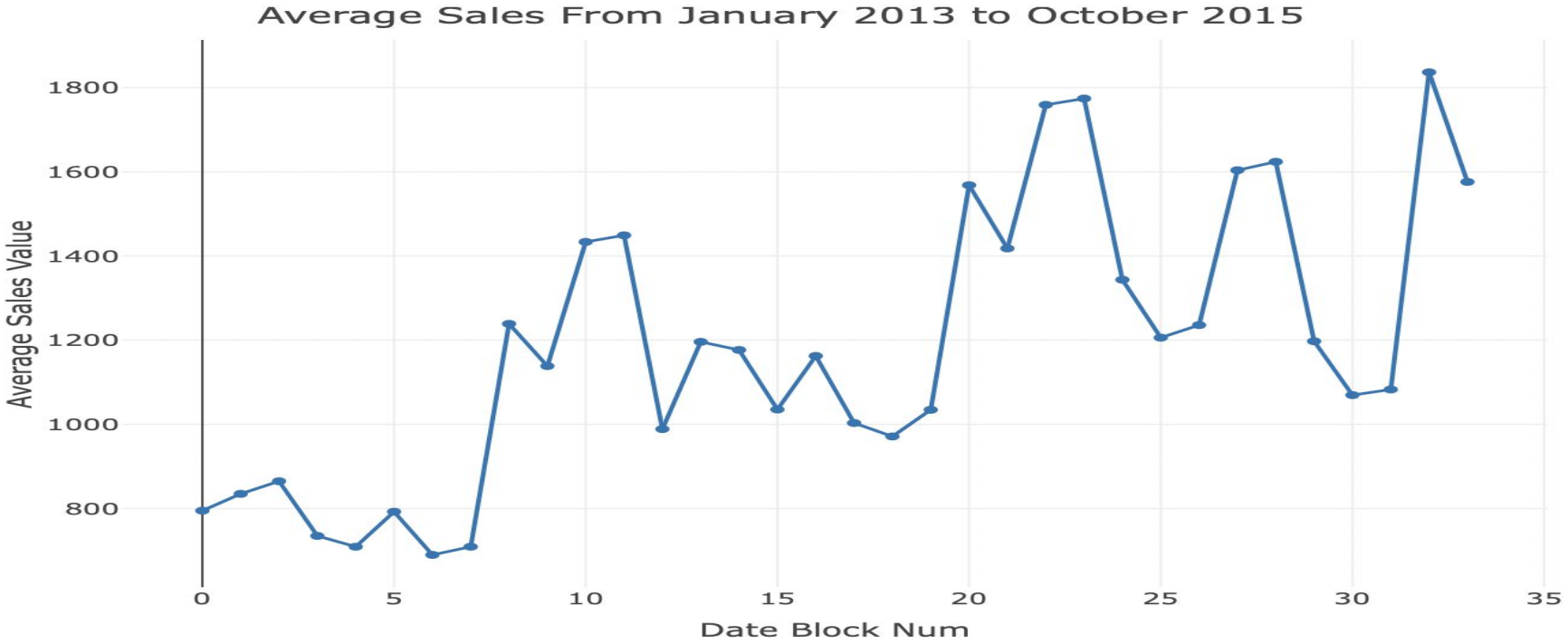
Total Sales Value From January 2013 to October 2015

To analyze the data from the January 2013 to October 2015 which is being formed in the block 1,3 and 5 and set to maximum at the block 10 and 20 at total sales value above 200M.



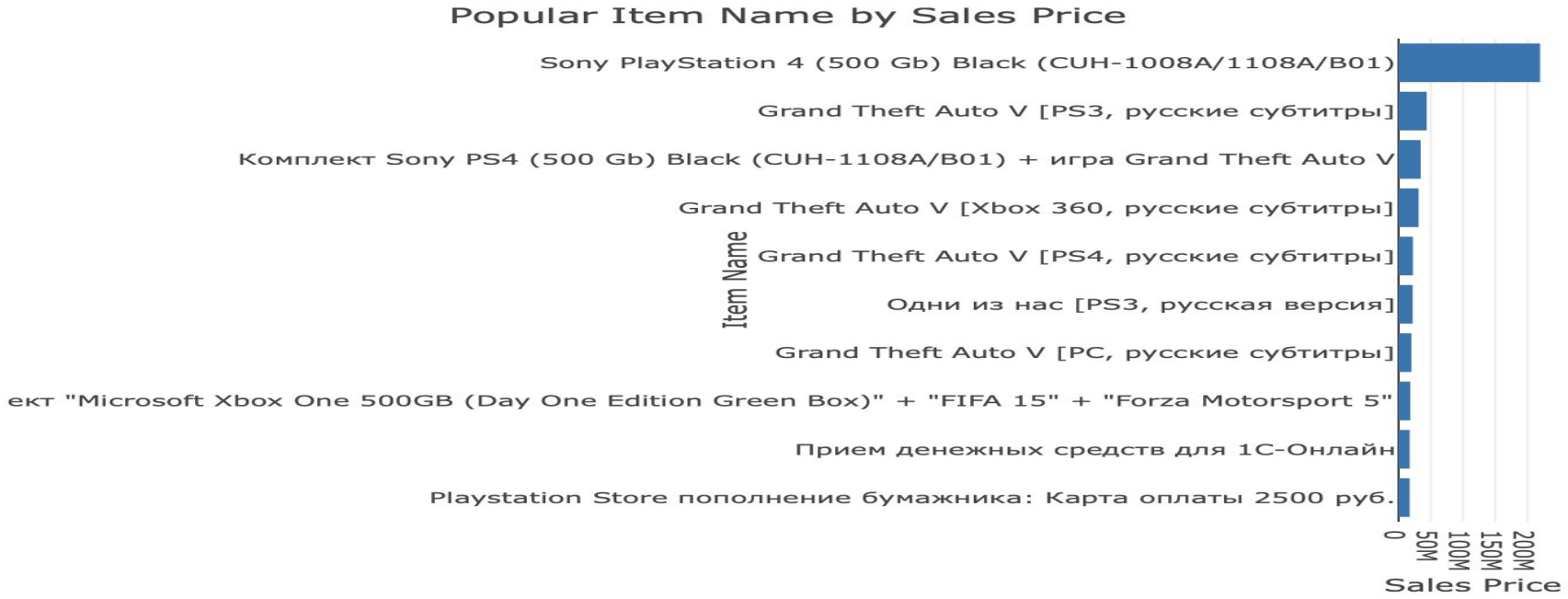
Average Sales Value From January 2013 to October 2015

The average Sales value from the January 2013 to the October 2015 have the maximum date block value 1,3 and 5 and the highest at the block 10 and 20 with the average sales value above 1400.



Sales Price of Popular Items

The sales price of the popular items that have been listed have the highest sales price of the Sony Play station 4 of the 200M to 0.



Training Model

The model have been trained, which is grouped by the year and the month and delete of the columns sold quantities and arrange in the decreasing order of the sold quantities which is ungrouped function.

Source Visual Outline

```
198  # TRAINING MODEL
199  train <-sales_data%>%group_by(Year,Month)%>%mutate(sold_qties=mean(item_cnt_day,na.rm=TRUE))%>%arrange
     (desc(sold_qties))%>%ungroup()
200
201  train <-train%>%select(date_block_num,shop_id,item_id,item_price,item_category_id,sold_qties)
202
203  item_data <-sales_data%>%group_by(item_id)%>%summarise(item_price,shop_id,item_category_id)
204
205  item_data <-distinct(item_data)
206  train
207  ```
```

R Console

tbl_df
2935849 x 6

A tibble: 2,935,849 × 6

date_block_num <int>	shop_id <int>	item_id <int>	item_price <dbl>	item_category_id <int>	sold_qties <dbl>
32	42	11170	58.0000	37	1.439926
32	42	11215	349.0000	40	1.439926
32	42	11215	349.0000	40	1.439926
32	42	11215	349.0000	40	1.439926
32	42	11232	299.0000	37	1.439926
32	42	11252	449.0000	40	1.439926
32	42	11253	649.0000	37	1.439926
32	42	11354	3999.0000	20	1.439926
32	42	11259	49.0000	40	1.439926
32	42	11404	399.0000	40	1.439926

1–10 of 2,935,849 rows Previous 1 2 3 4 5 6 ... 100 Next

Test Data

The test have been formed with the merge of the items by the item id .

```r
```{r}
test = merge(test, items[,c("item_id", "item_category_id")], by = "item_id", all.x = T)

test = merge(test, item_data[,c("shop_id","item_id", "item_price")], by.x = c("shop_id",
"item_id"),by.y = c("shop_id", "item_id"), all.x = T)

head(test)
```
```

Description: df [6 × 5]

	shop_id <int>	item_id <int>	ID <int>	item_category_id <int>	item_price <dbl>
1	2	30	22987	40	169.0
2	2	30	22987	40	359.0
3	2	30	22987	40	399.0
4	2	31	20994	37	698.5
5	2	31	20994	37	699.0
6	2	31	20994	37	399.0

6 rows

Linear Regression Model

The Linear Regression Model have been created using the variables sold quantities as the dependent variable and the independent variable as the shop id, item id, item price, item category id using the data as train. The p value is less than 2.2 and R square is 0.003 and the Adjusted R square is 0.003.

```
Call:
lm(formula = sold_qties ~ shop_id + item_id + item_price + item_category_id,
    data = train)

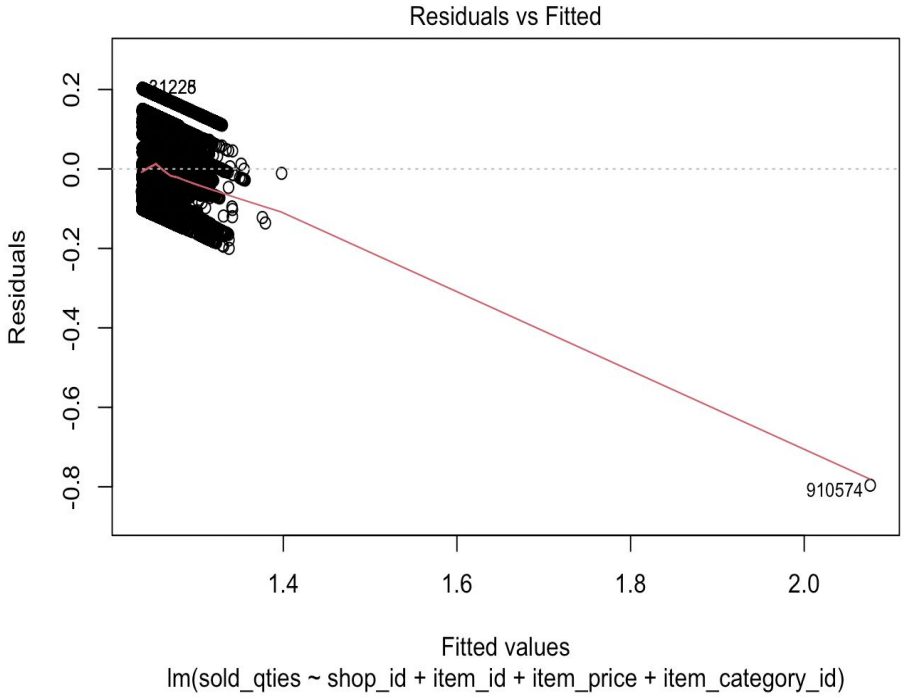
Residuals:
     Min       1Q   Median       3Q      Max
-0.79611 -0.06875 -0.01949  0.07645  0.20226

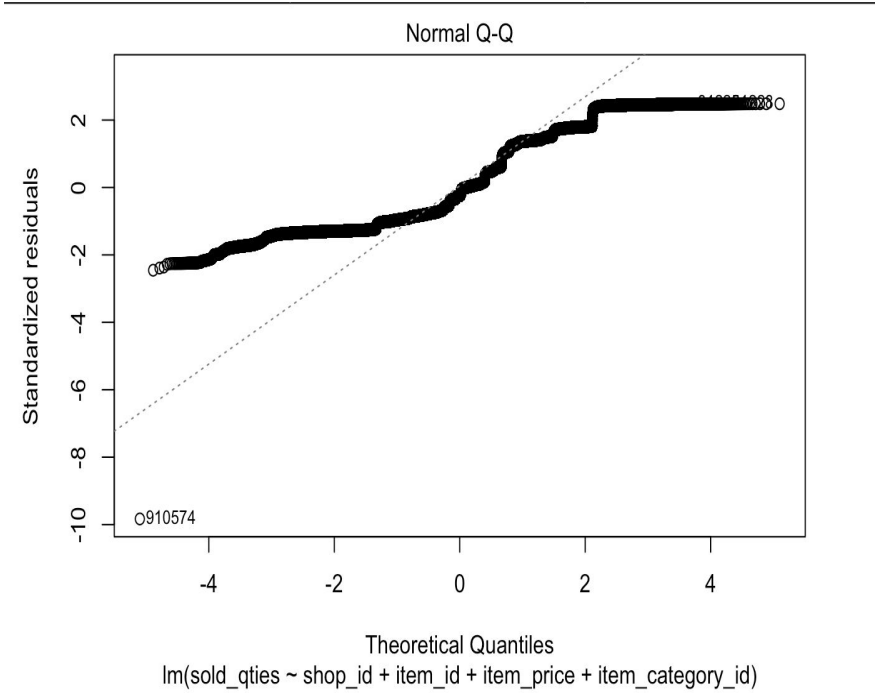
Coefficients:
                  Estimate Std. Error  t value Pr(>|t|)
(Intercept)       1.237e+00  1.680e-04 7363.944  < 2e-16 ***
shop_id           5.370e-05  2.931e-06   18.321  < 2e-16 ***
item_id          -5.001e-08  8.098e-09   -6.176 6.58e-10 ***
item_price        2.711e-06  2.844e-08   95.331  < 2e-16 ***
item_category_id  4.837e-05  3.067e-06   15.770  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

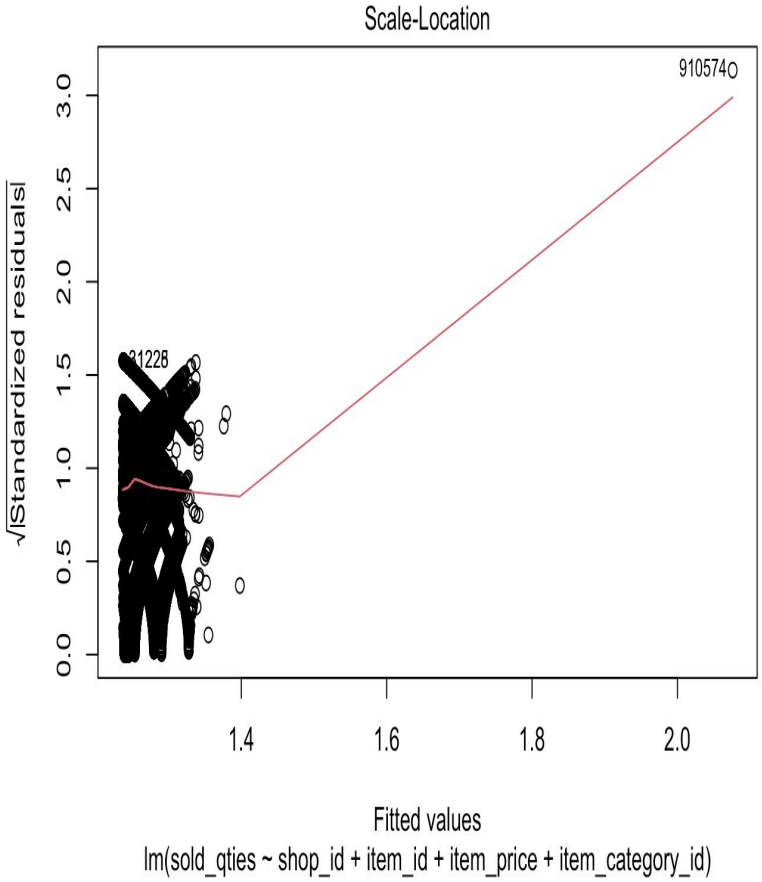
Residual standard error: 0.08144 on 2935844 degrees of freedom
Multiple R-squared:  0.003245,  Adjusted R-squared:  0.003243
F-statistic:  2389 on 4 and 2935844 DF,  p-value: < 2.2e-16
```

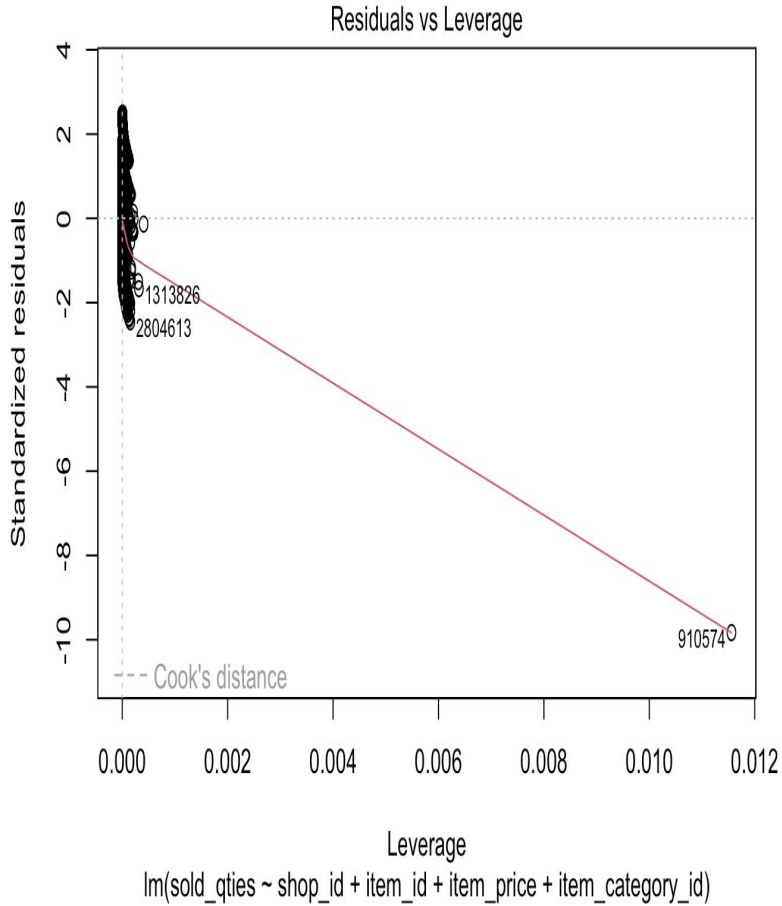
Model Graphs

The Residual fitted graph depicts the decrease in order from 0 to 2 with the scale locations rising from the 1.4 to 2.0 of the fitted values and the Residual have been formed from the rate in decreasing order from the -2 to 0.012.






Scale-Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
3.0
1.4
1.6
1.8
2.0
910574
Fitted values
lm(sold_qties ~ shop_id + item_id + item_price + item_category_id)


Residuals vs Leverage
Standardized residuals
4
2
0
-2
-4
-6
-8
-10
1313826
2804613
910574
Cook's distance
0.000
0.002
0.004
0.006
0.008
0.010
0.012
Leverage
lm(sold_qties ~ shop_id + item_id + item_price + item_category_id)

Residuals

The residuals that have been formed using the model have the order from 0.199.

```
240
241 ```{r}
242 res<-residuals(model)
243
244 res_data<-as.data.frame(res)
245
246 head(res_data)
247 ```
```

Description: df [6 × 1]

	res <dbl>
1	0.1992543
2	0.1983225
3	0.1983225
4	0.1983225
5	0.1986040
6	0.1980532

Prediction Model

The prediction model have been formed using the model with which the linear regression model have been created using the test ids of the variables forming the above results.

Source Visual Outline

```
249 ```{r}
250 predict<-predict(model, test[,c("ID","shop_id","item_id","item_category_id", "item_price")])
251
252 predict
253 ```
```

1	2	3	4	5	6	7	8	9	10
1.241983	1.242345	1.240392	1.239772	1.240655	1.242574	1.248177	1.240583	1.240502	1.240668
11	12	13	14	15	16	17	18	19	20
1.238912	1.265132	1.240538	1.241775	1.240902	1.240800	1.243190	1.240602	1.241211	1.246147
21	22	23	24	25	26	27	28	29	30
1.240532	1.240620	1.246016	1.240363	1.240455	1.240623	1.240768	1.244405	1.251637	1.237833
31	32	33	34	35	36	37	38	39	40
1.246698	1.239813	1.246295	1.245385	1.239326	1.239324	1.238511	1.245798	1.243942	1.243942
41	42	43	44	45	46	47	48	49	50
1.241773	1.245420	1.241068	1.241573	1.240825	1.242804	1.240960	1.241638	1.241475	1.241231
51	52	53	54	55	56	57	58	59	60
1.241501	1.241773	1.240563	1.241638	1.244009	1.241220	1.241944	1.241977	1.245596	1.248056
61	62	63	64	65	66	67	68	69	70
1.245560	1.240167	1.239998	1.240329	1.241225	1.242433	1.240134	1.239580	1.245320	1.240165
71	72	73	74	75	76	77	78	79	80
1.242431	1.248455	1.240312	1.240660	1.241135	1.240748	1.241419	1.241232	1.241263	1.241895
81	82	83	84	85	86	87	88	89	90
1.241480	1.241811	1.241705	1.243705	1.240143	1.241306	1.240541	1.240563	1.239612	1.239539
91	92	93	94	95	96	97	98	99	100
1.241352	1.241791	1.240153	1.245237	1.243731	1.240546	1.245015	1.241089	1.241671	1.245161
101	102	103	104	105	106	107	108	109	110
1.247729	1.258680	1.241768	1.248024	1.239842	1.242420	1.242416	1.244640	1.240570	1.242412
111	112	113	114	115	116	117	118	119	120
1.243030	1.240861	1.240667	1.240716	1.240511	1.241074	1.241469	1.240603	1.241161	1.241023
121	122	123	124	125	126	127	128	129	130
1.241024	1.241024	1.240888	1.240888	1.241159	1.241159	1.242867	1.239387	1.242159	1.239710
131	132	133	134	135	136	137	138	139	140
1.246743	1.241411	1.239223	1.240275	1.241023	1.242247	1.241989	1.241555	1.241608	1.241555
141	142	143	144	145	146	147	148	149	150

Prediction Data

The new data have been prepared using the name as the submission where the id and the predicted item count month have been defined.

```r
submission <- data.frame(ID = test$ID,item_cnt_month = predict)

head(submission)
```

Description: df [6 × 2]

	ID <int>	item_cnt_month <dbl>
1	0	1.241983
2	1	1.242345
3	2	1.240392
4	3	1.239772
5	4	1.240655
6	5	1.242574

Conclusion

The Total products sold per year were highest in year 2013 around 1562733 while the products sold per month is 359349 and day is by the 1, 2 and 4. The total Sales per year is highest in the year 2014 of 1346778479 and with the alternate month of the 1, 2 and 4. The highest quantity and the item have been listed in the year 2013 to 2015. The Maximum sales price in year 2013 is 1829990 and the in 2014 is 1044450. The Linear regression we performed to predict the model is significant and the prediction can be more attained with the other machine learning models.

THANK YOU

Presented By -

- Jyothsna Kaamala (jk734)
- Jashwaannth Sai Kilari (jk696)
- Nikhilesh Cherukuri (nc472)
- Manoj Ravipati (mr862)