SAYANTAN RAHA

Roll # : BAI09056

IIMB - BAI09 - Assignment 5

Out[1]: Toggle on/off Code

Out[7]:

	Store_Id	Store_Area	Zone	Net_Sales_CRESCENT	Net_Sales_CRESCENT_MIX_N_MATCH	Net_Sales_CRESCENT_POISE	Net_Sales_CRESCENT_SET	Net_Sales_BLI
0	1879	8154	EAST	0.0	523481.52	0.0	224348.65	88251
1	1885	11032	WEST	0.0	479126.98	0.0	224900.11	151922
2	1903	8374	EAST	0.0	456677.50	0.0	245383.60	109682
3	1911	4001	WEST	0.0	178680.82	0.0	25856.39	3499
4	1916	6603	NORTH	0.0	236793.24	0.0	68630.76	92294

Q-PART A-1

Feature Engineering

Create Dummy variables / One Hot Encoding of Zone

Out[8]:

t_SAHAR	Cost_CRESCENT	Cost_CRESCENT_MIX_N_MATCH	Cost_CRESCENT_POISE	Cost_CRESCENT_SET	Cost_BLINK	Cost_SAHAR	EAST	NORTH	SOUTH	WEST
0.0	0.0	306384.07	0.0	133800.61	54759.22	0.0	1	0	0	0
0.0	0.0	295953.89	0.0	145927.35	99258.18	0.0	0	0	0	1
0.0	0.0	255847.23	0.0	141408.60	65928.58	0.0	1	0	0	0

Profitability / Store Area Unit Calculation

We will now calculate the per unit are profitability of Store. The following formula is used to calculate the same:

- Profitability_Cresent = (Net Sales of Cresent - Cost of Cresent) / Total Area of the Store
- We will create 6 such variables for each product

Out[9]:

	Profit_C_SA	Profit_C_MNM_SA	Profit_C_P_SA	Profit_C_S_SA	Profit_B_SA	Profit_S_SA
0	0.0	26.624657	0.0	11.104739	4.107433	0.0
1	0.0	16.603797	0.0	7.158517	4.773747	0.0
2	0.0	23.982597	0.0	12.416408	5.224940	0.0

Discount / Total Sales (Discount Sensitivity)

- Discount Sensitivity is calculated as Revenue earned by Sales of Product on Discount / Total revue from Product
- We will create 6 such variable for 6 different products

Out[10]:

	Sens_C_SA	Sens_C_MNM_SA	Sens_C_P_SA	Sens_C_S_SA	Sens_B_SA	Sens_S_SA
0	0.0	0.483545	0.0	0.528782	0.533647	0.0
1	0.0	0.559408	0.0	0.616870	0.562860	0.0
2	0.0	0.419930	0.0	0.461731	0.454288	0.0

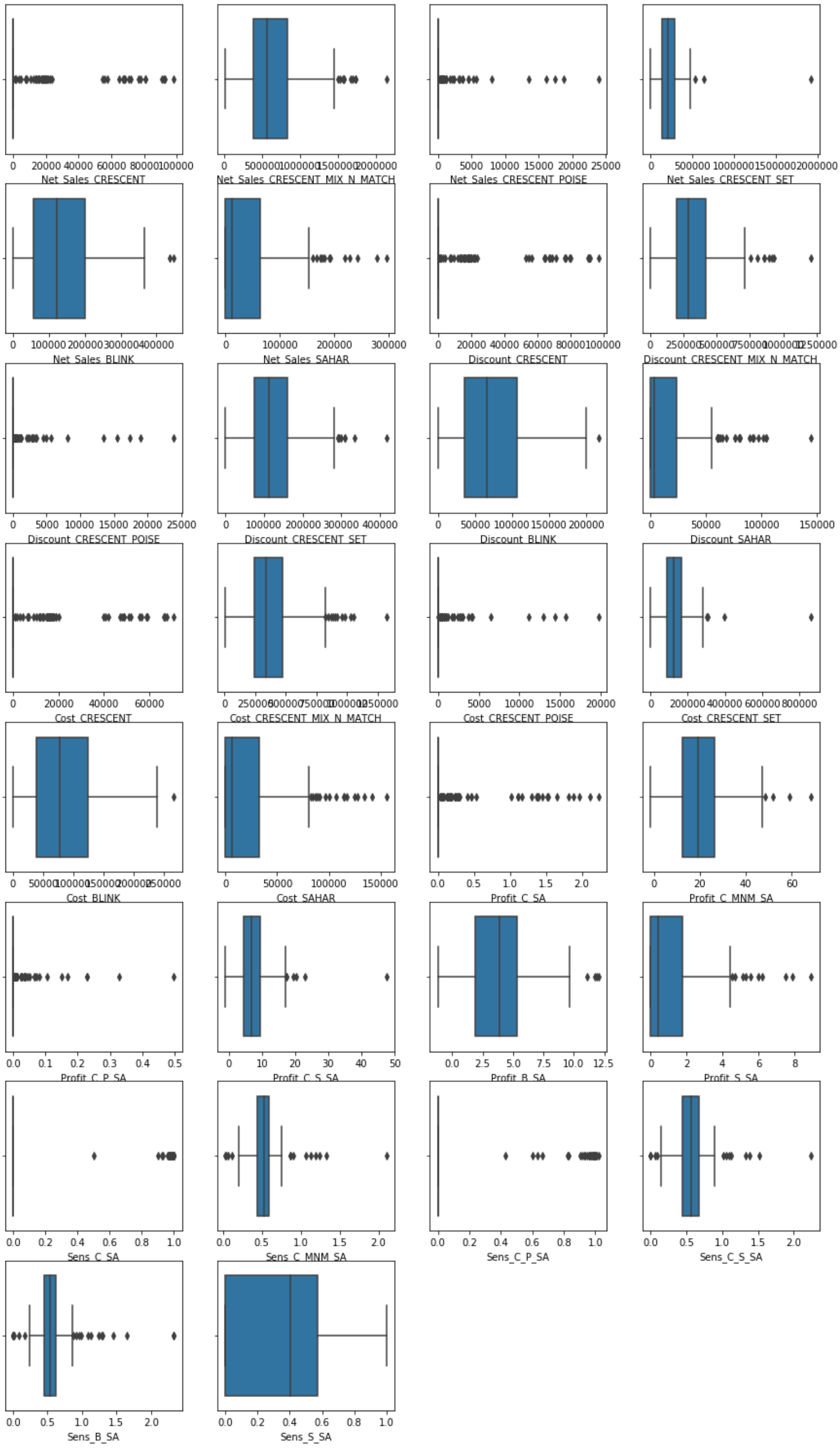
All Fields Used

- Net Revenue
- Discount Revenue
- Cost
- Store Area
- Sore Zones

Q-PART A-2

For the purpose of this analysis we are only considering Outliers from a **Univariate Analysis** perspective. There are ways to perform multivariate outlier analysis using Isolation Forest / Gaussian Mixture models, but such analysis is being considered out of scope of current work

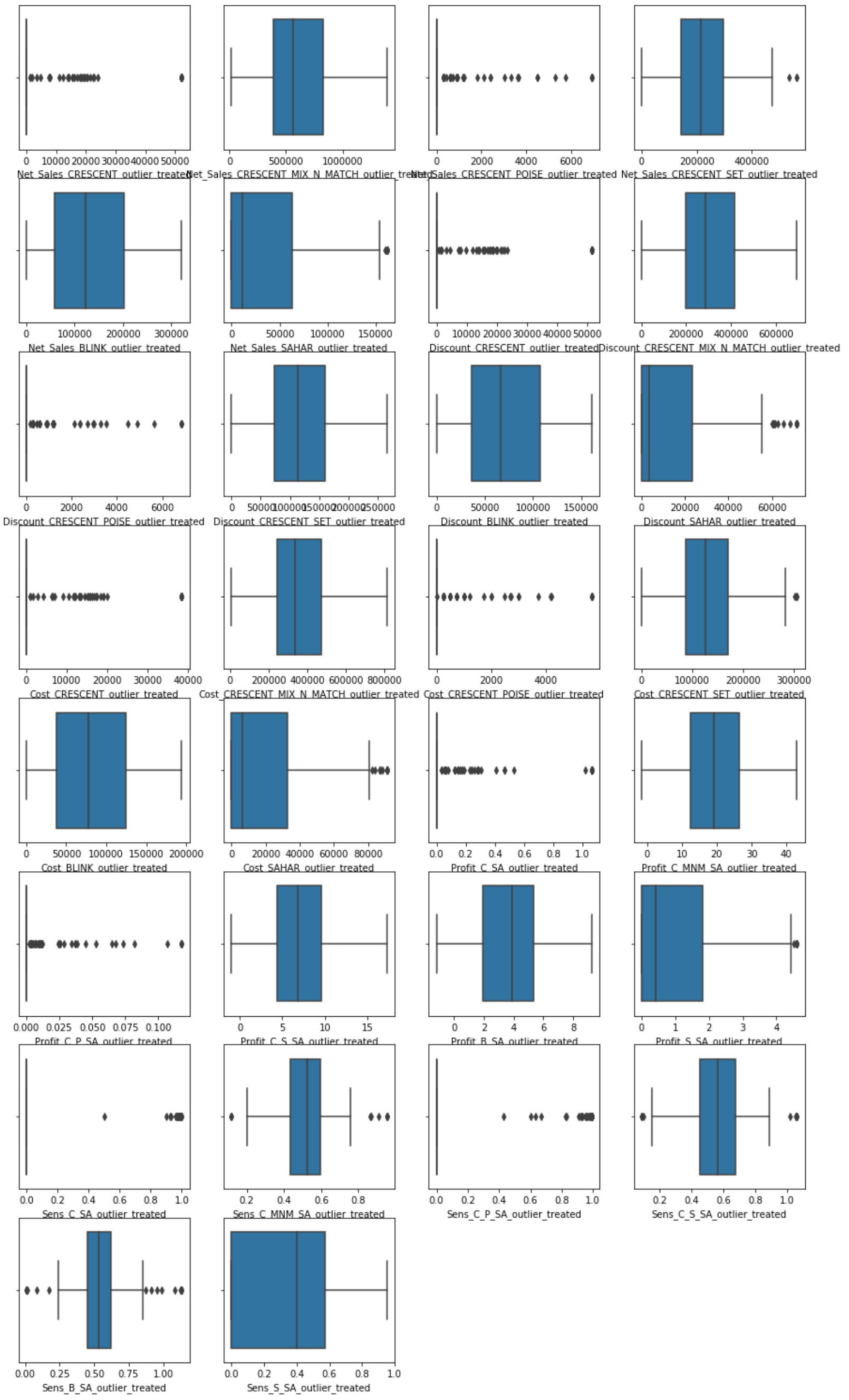
- UNIVARIATE Outlier Analysis



Write and documents Outliers and issues created by them

There are two easy approaches to deal with outliers:

1. Remove the Data points. The problem of such appoaches mean we will be loosing may data points (i.e. Store data) which is unacceptable
2. An alternative and an acceptable option is to clamp the values less than $\mu - 1.96 * \sigma$ to $\mu - 1.96 * \sigma$ and values greater than $\mu + 1.96 * \sigma$ to $\mu + 1.96 * \sigma$

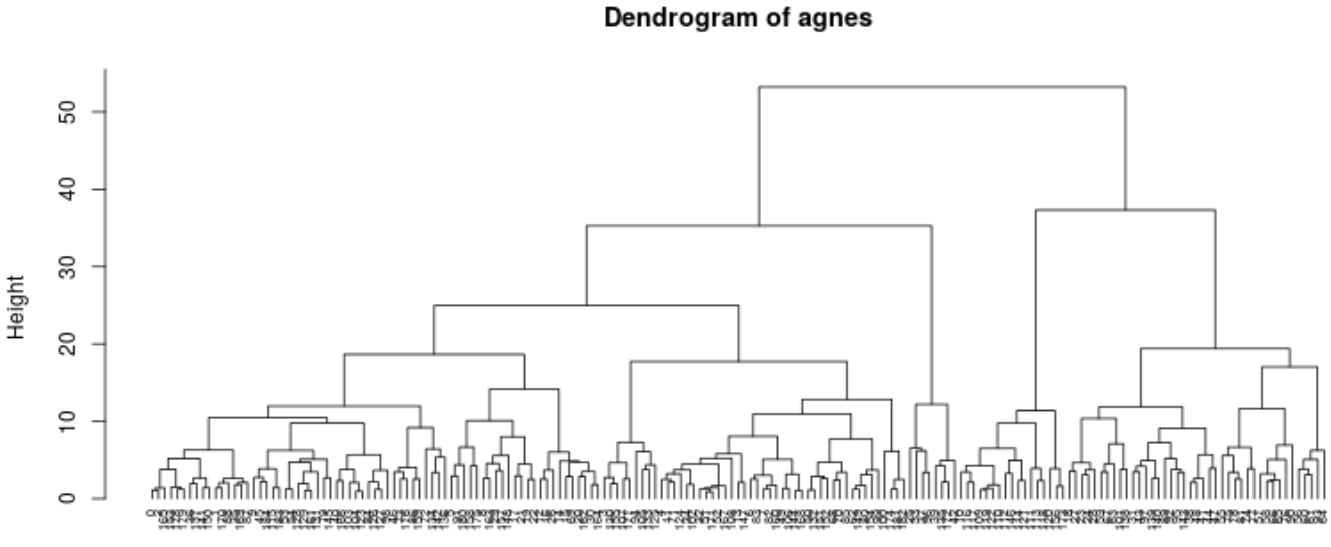


Q-PART A-3 - Hierarchial Clustering

Process of proceeding with Hierarchial Clustering

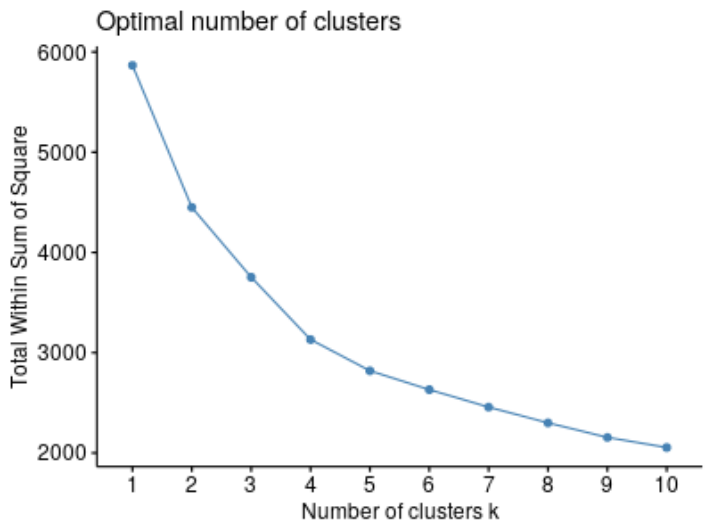
- STEP 1: We will first determine which is the best linkage method
  - Methods to assess:
    - average
    - single
    - complete
    - ward

```
[1] "Loss from each of Linkage Methods: "
  average    single  complete      ward
0.7594538 0.6224832 0.8449897 0.9482476
```

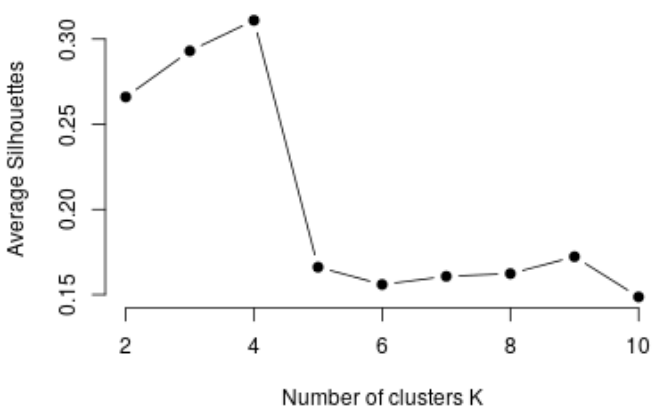


From the Results above it is clear "WARD" linkage is performing better than the other linkage options (Values indicate Clustering Coefficient, the closer it is to 1, the better it is)

- STEP 2: Determine the Optimal number of Clusters / Subgroups
  - Methods to be used:
    - SCREE Plot / Elbow Plot
    - Elbow Method



- As we can see there is no specific Elbow and the curve gently slopes down. The lowest error value is @10, but that will mean too few records in each cluster. **4 / 5 / 6** can be considered as a good number from the above plot, but it is not very clear

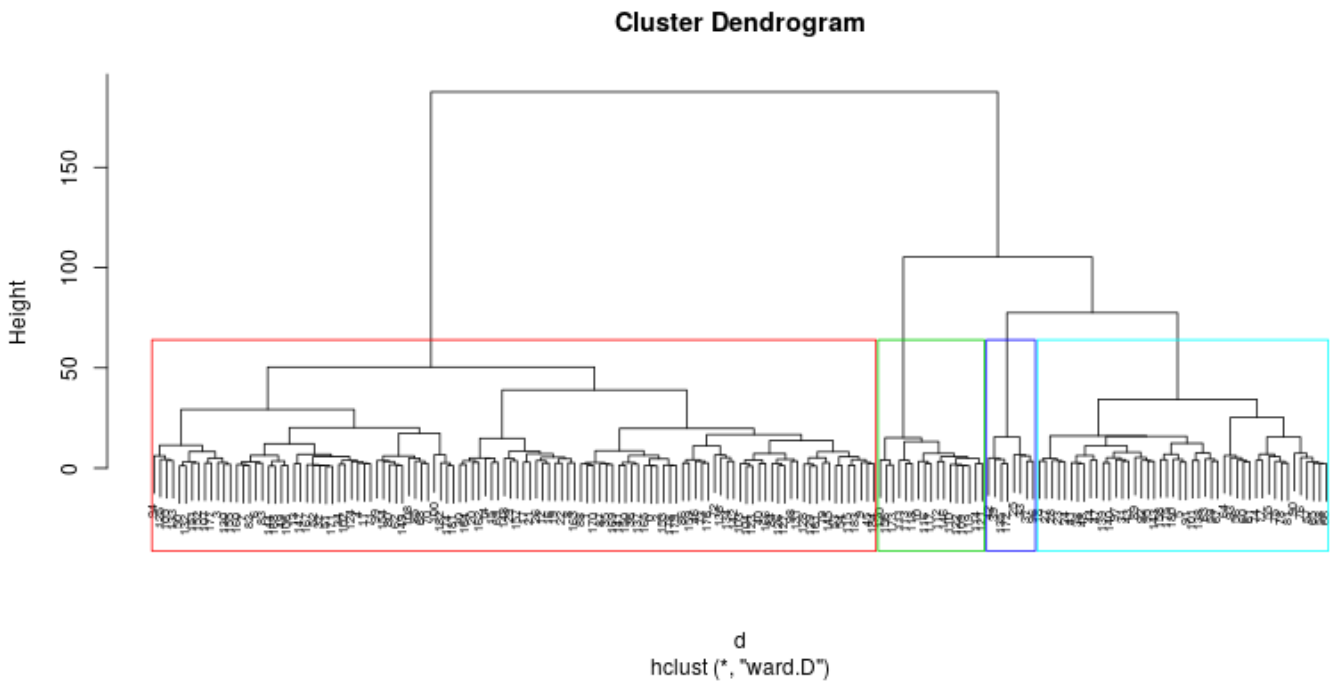


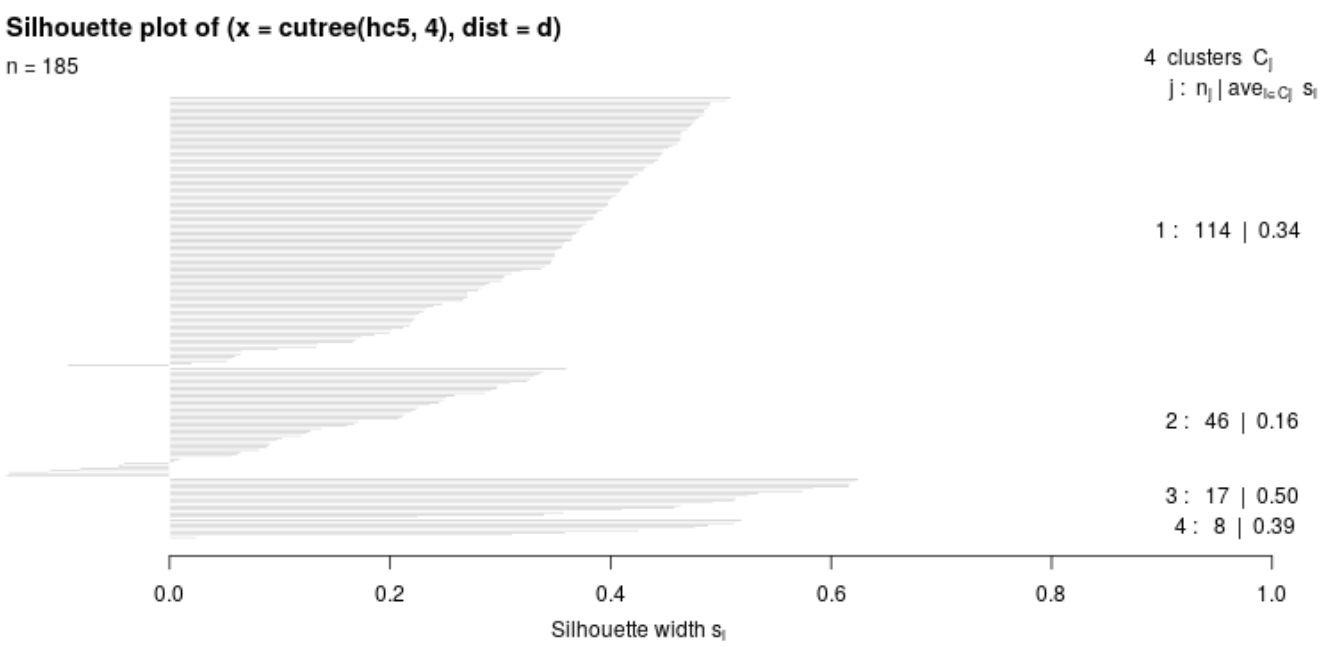
- The Silhouette gives a more clear value of 4 as the optimal number of Clusters.

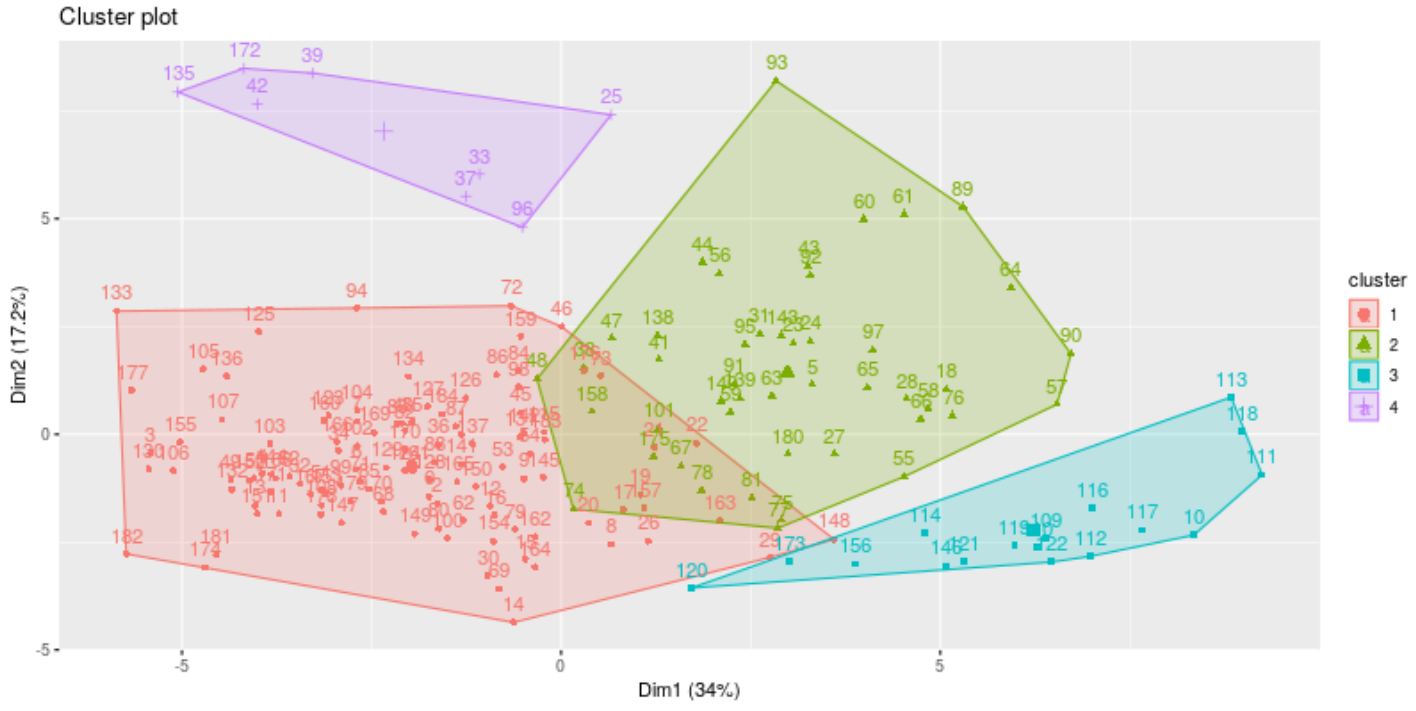
Between the two options, since the Average Silhouette Width gives a more clear solution, we will adopt this as the strategy for selecting optimal cut points

OPTIMAL CUTPOINTS = **4**

```
[1] "Number of members in each cluster"
sub_grp
  1   2   3   4
114  46  17   8
```

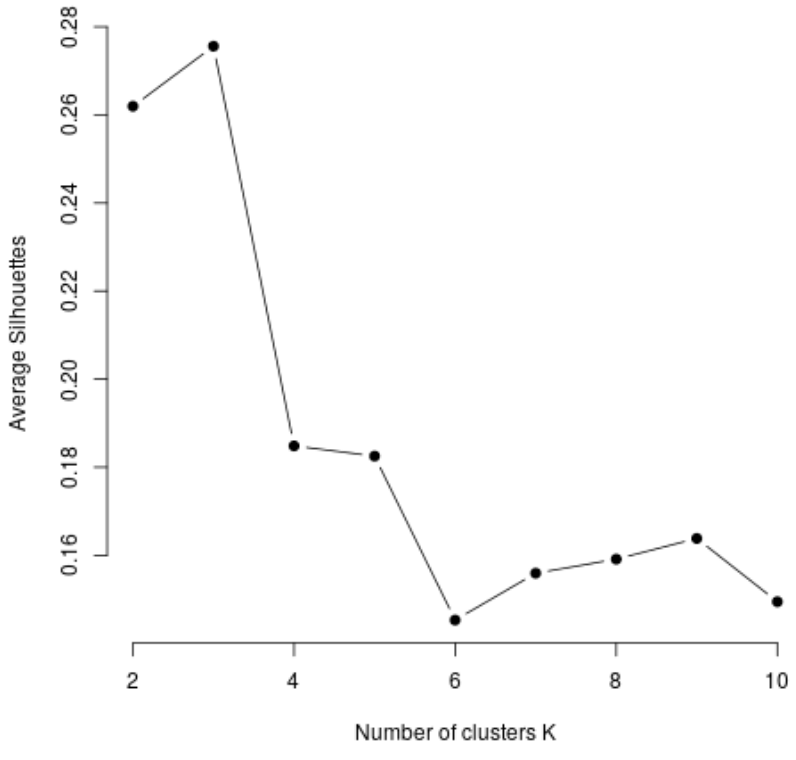






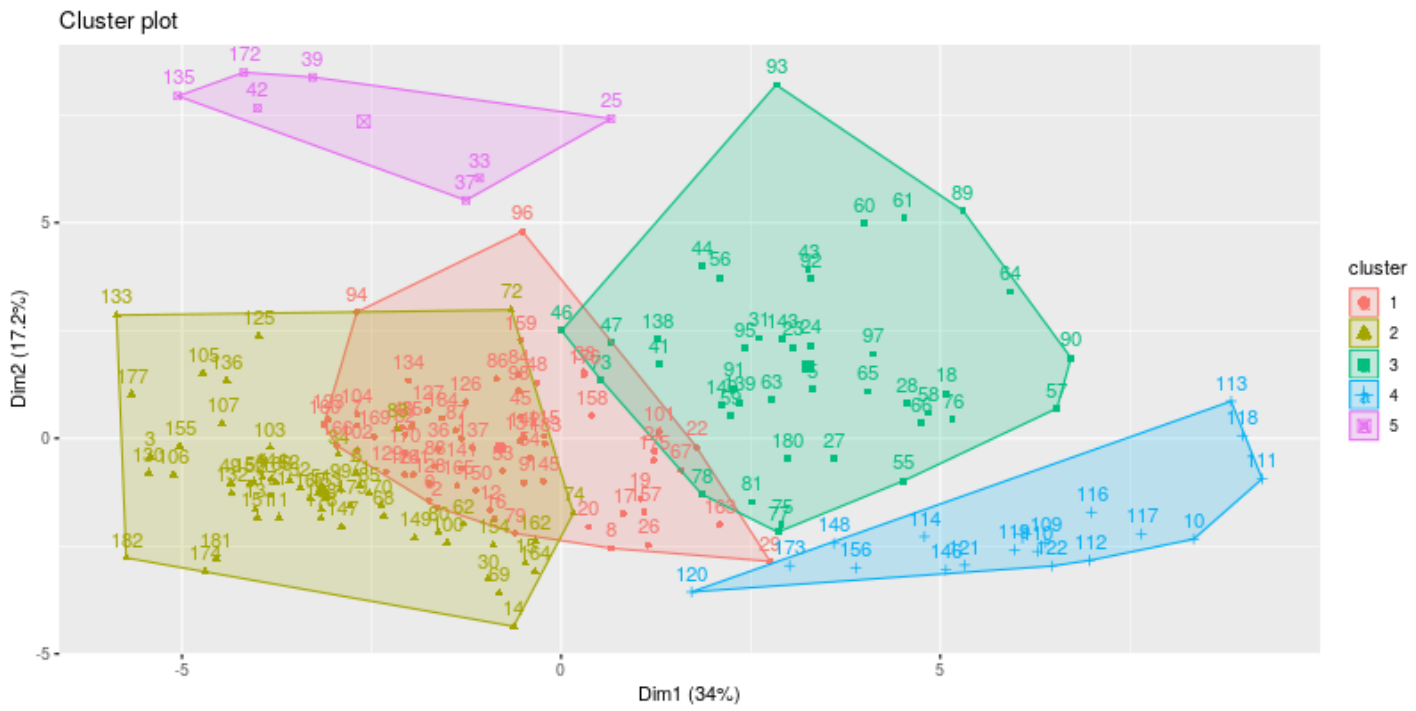
Q-PART A-4

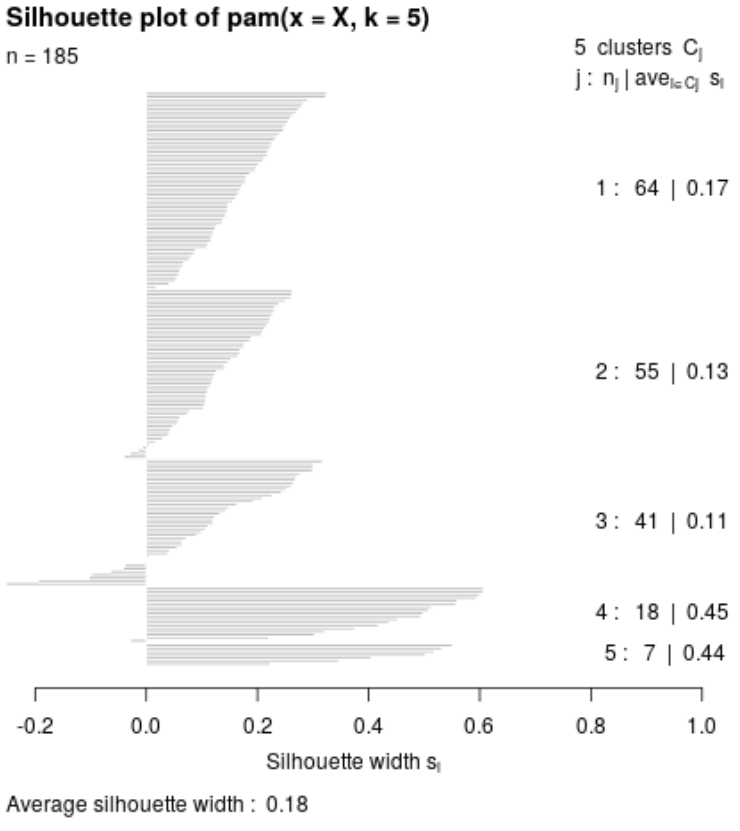
- We will try to determine the number of clusters first using Average Silhouette width metrics



OPTIMAL CUTPOINTS = **5**

```
[1] "Number of members in each cluster"

 1  2  3  4  5
64 55 41 18  7
```





Advantages of using partitioning around medoids (PAM) over K-means

Both k-means and PAM algorithms are breaking the dataset up into k groups. Also, they are both trying to minimize the distance between points of the same cluster and a particular point which is the center of that cluster. In contrast to the k-means algorithm, PAM algorithm chooses points as centers that belong to the dastaset. PAM algorithm uses a greedy search which may not find the global optimum solution. Medoids are more robust to outliers than centroids, but they need more computation for high dimensional data.

Q-PART A-5

From the solution in Q3 and Q4 we see that the Average Silhouette Value for the hierarchial clustering is around .31, while that from partition around medoids is .2.

Range of Silhouette Interpretation can be considered as follows:

- 0.71-1.0 ==> A strong structure has been found
- 0.51-0.70 ==> A reasonable structure has been found
- 0.26-0.50 ==> The structure is weak and could be artificial
- < 0.25 ==> No substantial structure has been found

From the chart above we can conclude that Hierarchial Clustering is able to identify some sort of weak structures, however PAM is failing to do so. Hence Hierarchial Clustering is the better option for clustering for this dataset.

Q-PART A-6

158	-0.5358517	0.666293014
159	1.7520007	0.288483066
160	-0.5358517	0.430674099
161	-0.5358517	-0.462243550
162	-0.5358517	-0.301498842
163	-0.5358517	0.284633361
164	-0.5358517	0.293694607
165	-0.5358517	-0.346136385
166	-0.5358517	1.394197770
167	-0.5358517	-0.911809279
168	-0.5358517	-1.160189860
169	-0.5358517	0.601965322
170	-0.5358517	0.480861701
171	-0.5358517	-0.096324949
172	1.8950194	2.486796631
173	-0.5358517	-0.461242124
174	-0.5358517	-2.415445903
175	-0.5358517	-1.075304933
176	1.9716698	0.608849119

Out[45]:

	cluster	1	2	3	4
Store_Area	**mean**	10937.526316	17155.021739	1.381159e+04	13147.125000
	count	114.000000	46.000000	1.700000e+01	8.000000
EAST	**mean**	0.219298	0.108696	0.000000e+00	0.250000
	count	114.000000	46.000000	1.700000e+01	8.000000
NORTH	**mean**	0.245614	0.086957	0.000000e+00	0.125000
	count	114.000000	46.000000	1.700000e+01	8.000000
SOUTH	**mean**	0.324561	0.521739	1.000000e+00	0.125000
	count	114.000000	46.000000	1.700000e+01	8.000000
WEST	**mean**	0.210526	0.282609	0.000000e+00	0.500000
	count	114.000000	46.000000	1.700000e+01	8.000000
Net_Sales_CRESCENT	**mean**	2782.873333	1845.039783	7.470936e+04	1006.575000
	count	114.000000	46.000000	1.700000e+01	8.000000
Net_Sales_CRESCENT_MIX_N_MATCH	**mean**	446292.892719	923379.151087	1.350839e+06	429651.393750
	count	114.000000	46.000000	1.700000e+01	8.000000
Net_Sales_CRESCENT_POISE	**mean**	137.643246	679.847826	1.127529e+02	13292.001250
	count	114.000000	46.000000	1.700000e+01	8.000000
Net_Sales_CRESCENT_SET	**mean**	172786.258772	353120.086739	3.528298e+05	150450.335000
	count	114.000000	46.000000	1.700000e+01	8.000000
Net_Sales_BLINK	**mean**	90317.010088	237694.458913	1.767445e+05	116221.736250
	count	114.000000	46.000000	1.700000e+01	8.000000
Net_Sales_SAHAR	**mean**	9995.033070	116662.322174	7.198671e+04	6636.432500
	count	114.000000	46.000000	1.700000e+01	8.000000
Discount_CRESCENT	**mean**	2729.661754	1802.460217	7.364964e+04	976.675000
	count	114.000000	46.000000	1.700000e+01	8.000000
Discount_CRESCENT_MIX_N_MATCH	**mean**	229744.426140	415795.102609	7.184721e+05	348723.265000
	count	114.000000	46.000000	1.700000e+01	8.000000
Discount_CRESCENT_POISE	**mean**	130.330439	609.304348	9.865882e+01	13213.748750
	count	114.000000	46.000000	1.700000e+01	8.000000
Discount_CRESCENT_SET	**mean**	94555.596491	167707.156739	2.083284e+05	116231.340000
	count	114.000000	46.000000	1.700000e+01	8.000000
Discount_BLINK	**mean**	48477.355702	111309.243261	9.221618e+04	91426.826250
	count	114.000000	46.000000	1.700000e+01	8.000000
Discount_SAHAR	**mean**	4096.756404	49634.351739	3.270740e+04	2837.192500
	count	114.000000	46.000000	1.700000e+01	8.000000
Cost_CRESCENT	**mean**	2341.904912	1549.296087	5.421362e+04	866.588750
	count	114.000000	46.000000	1.700000e+01	8.000000
Cost_CRESCENT_MIX_N_MATCH	**mean**	267148.182544	528303.654783	8.257514e+05	289942.931250
	count	114.000000	46.000000	1.700000e+01	8.000000
Cost_CRESCENT_POISE	**mean**	108.533333	520.335000	8.763529e+01	10899.857500
	count	114.000000	46.000000	1.700000e+01	8.000000
Cost_CRESCENT_SET	**mean**	104752.794825	201441.456304	2.144232e+05	99184.520000
	count	114.000000	46.000000	1.700000e+01	8.000000
Cost_BLINK	**mean**	56863.698158	141591.625435	1.099316e+05	81383.395000
	count	114.000000	46.000000	1.700000e+01	8.000000
Cost_SAHAR	**mean**	5610.929737	66242.799348	4.124455e+04	3618.476250
	count	114.000000	46.000000	1.700000e+01	8.000000
Profit_C_SA	**mean**	0.045156	0.014974	1.530833e+00	0.007520
	count	114.000000	46.000000	1.700000e+01	8.000000
Profit_C_MNM_SA	**mean**	17.116570	24.601617	3.938966e+01	10.447682
	count	114.000000	46.000000	1.700000e+01	8.000000
Profit_C_P_SA	**mean**	0.002808	0.009468	1.427989e-03	0.213671
	count	114.000000	46.000000	1.700000e+01	8.000000
Profit_C_S_SA	**mean**	6.505019	9.401245	1.029311e+01	3.505118
	count	114.000000	46.000000	1.700000e+01	8.000000
Profit_B_SA	**mean**	3.124455	5.993844	4.868295e+00	2.323015
	count	114.000000	46.000000	1.700000e+01	8.000000
Profit_S_SA	**mean**	0.398252	3.101811	2.247203e+00	0.219095
	count	114.000000	46.000000	1.700000e+01	8.000000
Sens_C_SA	**mean**	0.184998	0.126865	9.849703e-01	0.121287
	count	114.000000	46.000000	1.700000e+01	8.000000
Sens_C_MNM_SA	**mean**	0.534988	0.477417	5.254490e-01	0.892149
	count	114.000000	46.000000	1.700000e+01	8.000000
Sens_C_P_SA	**mean**	0.119510	0.336408	1.428571e-01	0.989744

	cluster	1	2	3	4
	count	114.000000	46.000000	1.700000e+01	8.000000
Sens_C_S_SA	**mean**	0.570164	0.525943	5.742711e-01	0.928619
	count	114.000000	46.000000	1.700000e+01	8.000000
Sens_B_SA	**mean**	0.588127	0.474161	5.323417e-01	0.963368
	count	114.000000	46.000000	1.700000e+01	8.000000
Sens_S_SA	**mean**	0.291924	0.413423	4.574955e-01	0.478513
	count	114.000000	46.000000	1.700000e+01	8.000000

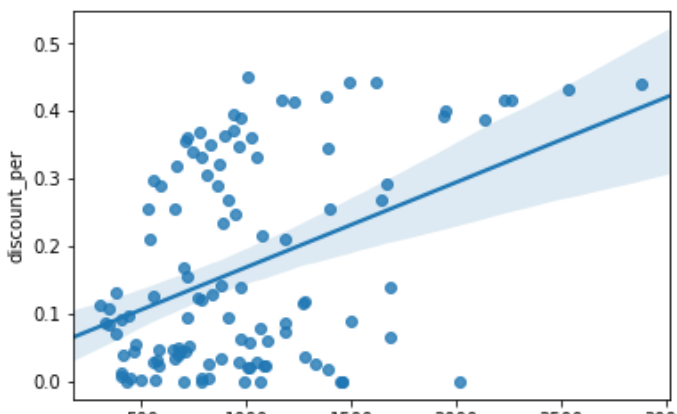
Cluster Characteristics

- Cluster 1:
  - Smaller Stores (Around 1000 Area on average)
  - Equally distributed across all zone. Slightly higher density in the Southern Zone
  - Third in Profitability / Unit of Store Area
  - Majority of Revenue comes from Sales of Cresent Mix & match, Cresent Set and Blink (in the order as specified)
  - All the above items are also sensitive to Discounts being offered, i.e. > 50% are sold on discount
- Cluster 2:
  - Largest Stores (Around 1700 Area on average)
  - More than 50% of Stores are in the Southern Zone, and another 25% in Western Zone.
  - First in Profitability / Unit of Store Area
  - Majority of Revenue comes from Sales of Cresent Mix & match, Cresent Set and Blink (in the order as specified)
  - Its sells more than 50% of its products without Discount (except for Cresent Set)
- Cluster 3:
  - Mid Sized Stores (Around 1380 Area on average)
  - All the stores are in Southern Zone
  - Second in Profitability / Unit of Store Area
  - Highest revenue grosser across all stores on average
  - Majority of Revenue comes from Sales of Cresent Mix & match, Cresent Set and Blink (in the order as specified)
  - Its sells more than 50% of its products on Discount
- Cluster 4:
  - Mid Sized Stores (Around 1300 Area on average)
  - More than 50% of Stores are in the Western Zone, and another 25% in Eastern Zone.
  - Least in Profitability / Unit of Store Area
  - Lowest revenue grosser across all stores on average across most products
  - Its sells approx 90% of its products on Discount

Q-PART B-7

a. Relationship between Sales units(sales_units) & Discount % (discount_per)

Out[47]: `<matplotlib.axes._subplots.AxesSubplot at 0x7fc1d2c1c748>`

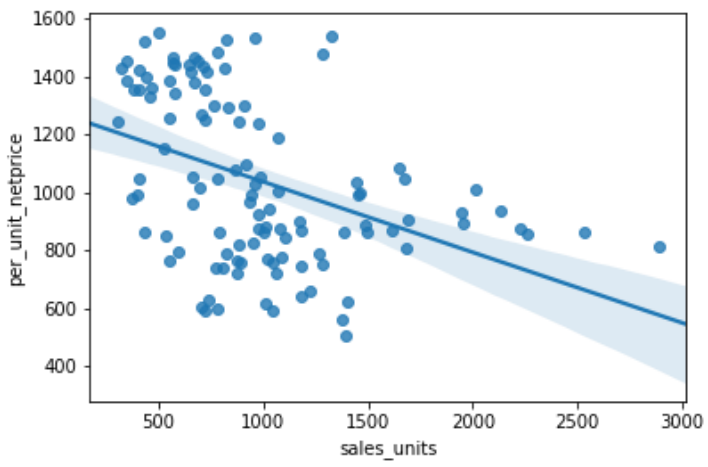



```
Correlation between Sales Units and Disount % is : 0.40931870139375237
```

- Between Discount % 0- 10%: Sales units are more or less flat. It seems there is little effect of Discount on Sales Units
- Between Discount % 10- 40%: Sales units increase almost linearly with increase in discount
- Beyond 40%: Sales units taper off and becomes almost flat
- There is a correlation between Sales Unit and Discount Percentage

b. Relationship between Sales units & Net Price (per_unit_netprice)

Out[49]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc1bfe42eb8>

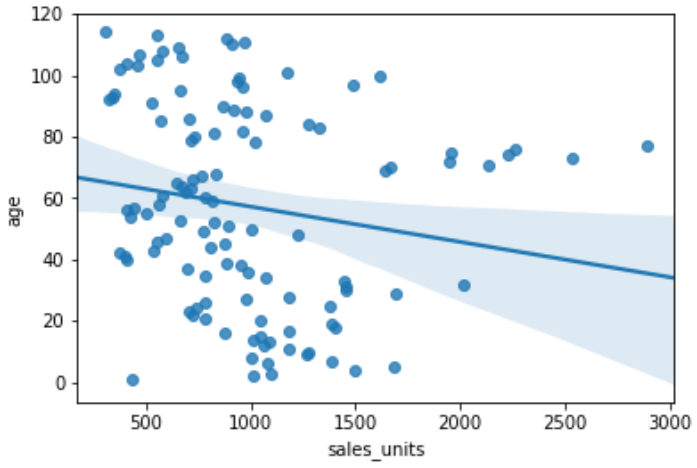



```
Correlation between Sales Units and Net Price is : -0.41528964093754234
```

- There is a negative correlation between Sales Unit and Net Price
- As Per Unit Net Price reduces the number of sales units purchased increases
- There are some outliers / leverage points around 800-900 price-band

c. Relationship between Sales units & Age (age)

Out[51]: <matplotlib.axes._subplots.AxesSubplot at 0x7fc1bbb662e8>




```
Correlation between Sales Units and Age is : -0.17082320167216053
```

- There is a very weak but negative correlation between Age and Sales Units, with increasing age Sales Units reduce
- In general Sales Unit falls in a band between (400-900) for most ages
- There are breakouts from the above pattern leading to increased Sales Units
  - Period 10-23
  - Period 28-34
  - Period 69-78

Q-PART B-8

Overfitting in Machine Learning

Overfitting refers to a model that fits / models the training data too well almost perfectly.

Consequences of Overfitting

Overfitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model. The problem is that these concepts do not apply to new data and negatively impact the models ability to generalize.

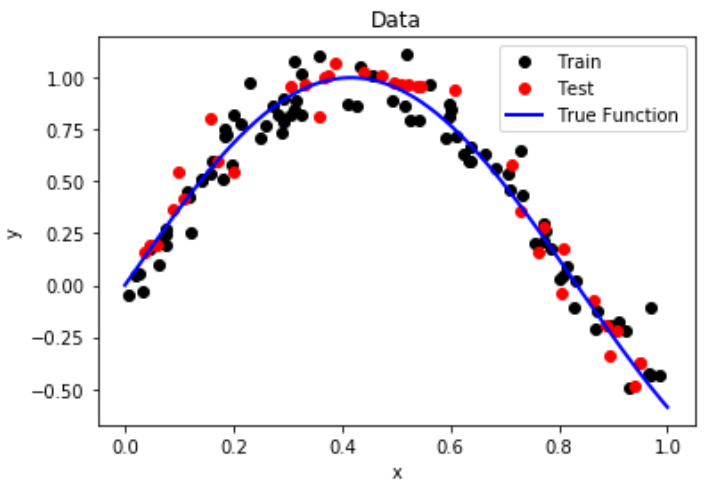
Underfitting in Machine Learning

Underfitting refers to a model that can neither model the training data nor generalize to new data. An underfit machine learning model is not a suitable model and will be obvious as it will have poor performance on the training data.

Example below

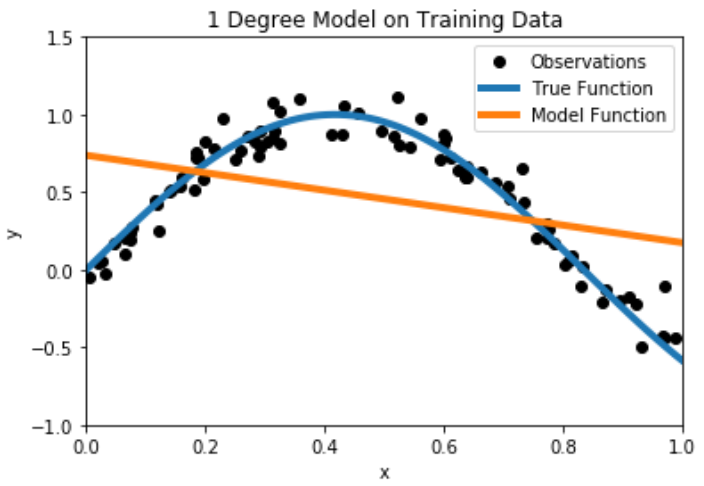
First, we need a "true" relationship. We define a curve, in this case a sine curve to serve as our process that generates the data. As the real-world is never perfectly clean however, we also need to add some noise into the observations. This is done by adding a small random number to each value.

Polynomial Model We want to try and capture the data using a polynomial function. A polynomial is defined by the degree, or the highest power to for the x-values. A line has a degree of 1 because it is of the form $y = b_1 * x + b_0$ where b_1 is the slope and b_0 is the intercept. A third degree polynomial would have the form $y = b_3 * x^3 + b_2 * x^2 + b_1 * x + b_0$ and so on. The higher the degree of the polynomial, the more flexible the model. A more flexible model is prone to overfitting because it can can "bend" to follow the training data. The following function creates a polynomial with the specified number of degrees and plots the results. We can use these results to determine the optimal degrees to achieve the right balance between over and underfitting.



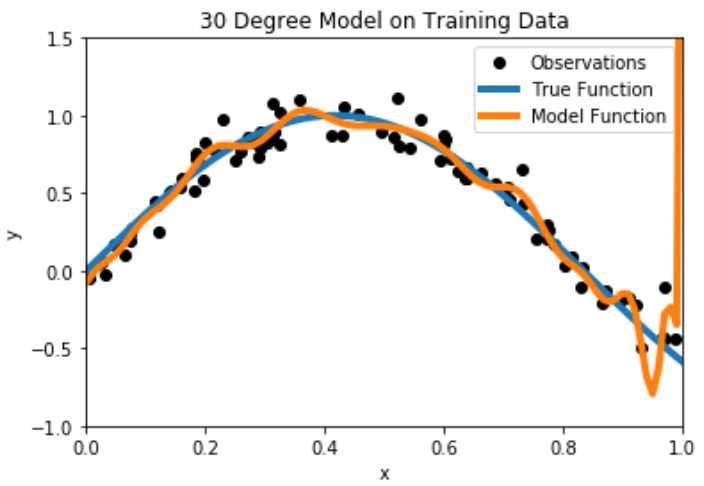
Try Model with Different Degrees

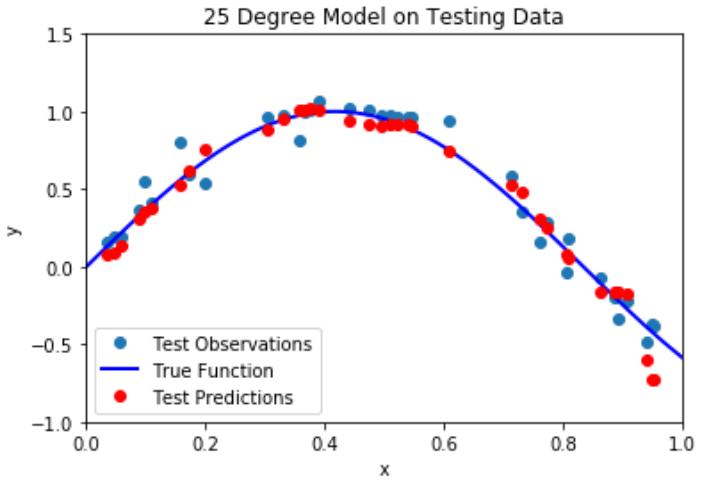
- Degrees = 1 -> Underfitting from sklearn.preprocessing import PolynomialFeatures from sklearn.linear_model import LinearRegression from sklearn.model_selection import cross_val_score from sklearn.metrics import mean_squared_error



Degrees = 25 -> Overfitting

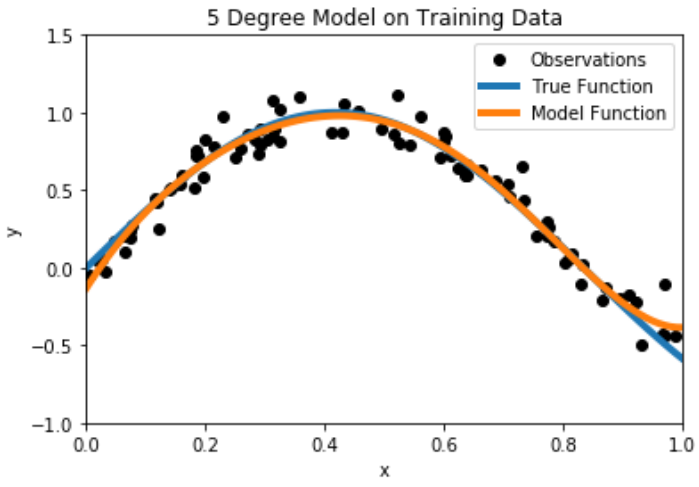
- We can go in the completely opposite direction and create a model that overfits the data. This model has too much flexibility and learns the training data too closely. As the training data has some amount of noise, it will end up capturing that noise and will be misled by that noise when it tries to make predictions on the test data.

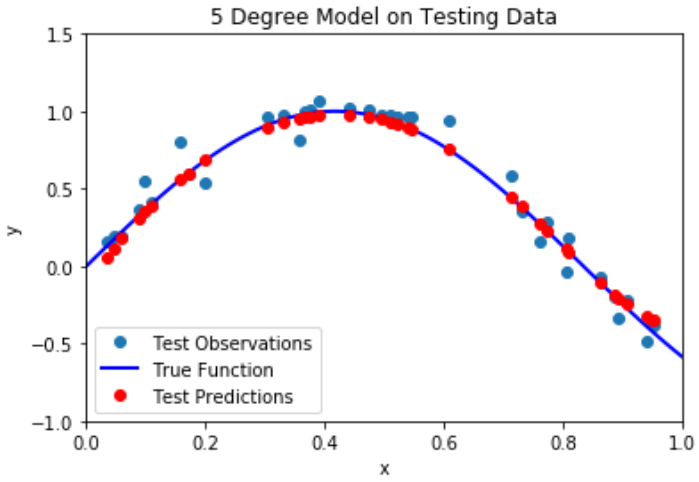




Degrees = 5 -> Balanced Model

Now that we have seen the two extremes, we can take a look at a model that does a good job of both accounting for the data while not following it too closely.





Q-PART B-9

Separating data into training, validation and testing sets is an important part of evaluating analytics models. Typically, when we separate a data set into a training set, validation set and testing set, most of the data is used for training, and a smaller portion of the data is used for validation and testing.

After a model has been built by using the training set, we fine tune the model hyperparameters by making predictions against the validation set. This done to prevent the model from overfitting on the test set.

The final prediction is done on the test set which acts as an unseen future set, and can be considered as a true measurement of the model performance on future unseen data.

For time series forcasting, we typically do not perform cross validation as it is important to capture the time component of the data. Data split is performed based on time.

Q-PART B-10

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 114 entries, 0 to 113
Data columns (total 13 columns):
week                      114 non-null float64
cluster_id                114 non-null int64
brand                     114 non-null object
prod_brick                114 non-null object
sales_units               114 non-null int64
per_unit_grossprice       114 non-null float64
per_unit_netprice         114 non-null float64
per_unit_discountprice    114 non-null float64
discount_per              114 non-null float64
promo_week_flg            114 non-null int64
age                       114 non-null int64
week_no                   114 non-null int64
year_no                   114 non-null int64
dtypes: float64(5), int64(6), object(2)
memory usage: 11.7+ KB
```

Out[58]:

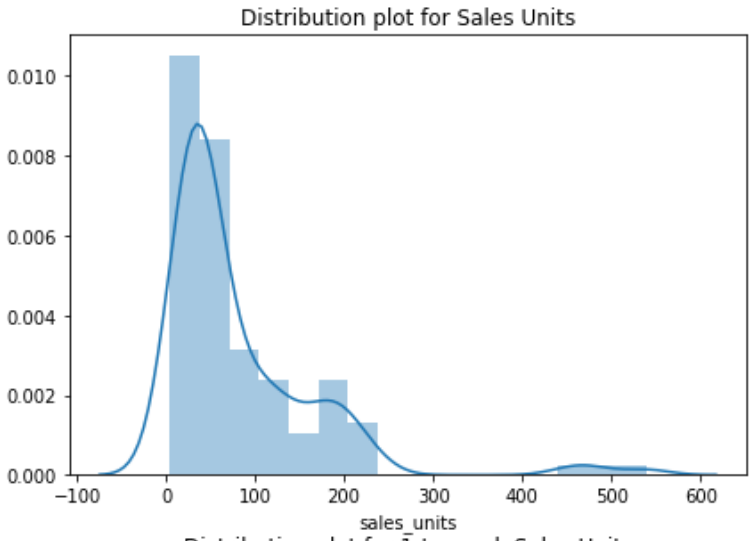
	week	cluster_id	brand	prod_brick	sales_units	per_unit_grossprice	per_unit_netprice	per_unit_discountprice	discount_per	promo_week_flg	age	week_no	ye
0	9.2013	2	BLINK	HAREMS	38	596.368421	536.596053	59.772368	0.100227	0	1	9	
1	10.2013	2	BLINK	HAREMS	93	597.924731	566.334516	31.590215	0.052833	0	2	10	
2	11.2013	2	BLINK	HAREMS	122	600.639344	519.528115	81.111230	0.135041	0	3	11	
3	12.2013	2	BLINK	HAREMS	133	598.248120	541.940301	56.307820	0.094121	0	4	12	
4	13.2013	2	BLINK	HAREMS	130	591.307692	476.174077	115.133615	0.194710	0	5	13	

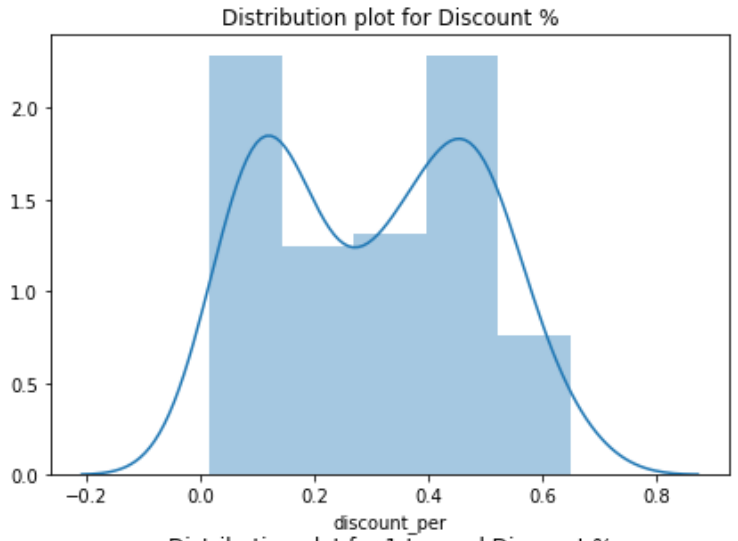
Out[59]:

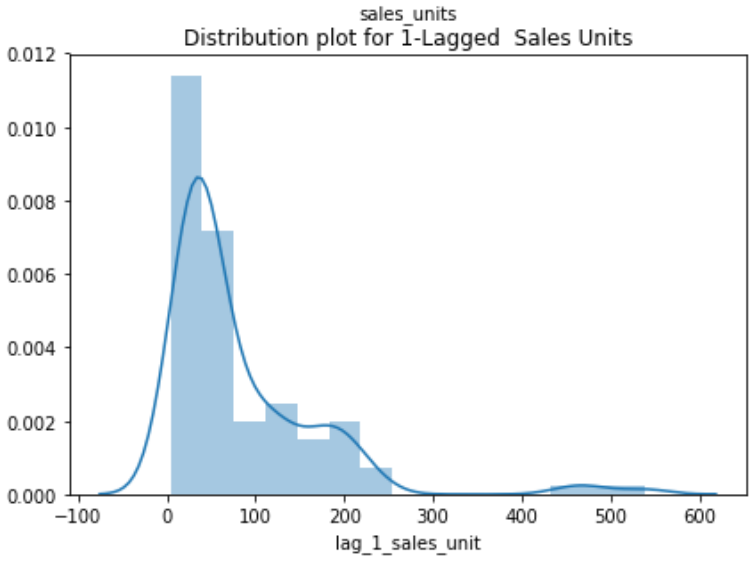
sales_units	per_unit_grossprice	per_unit_netprice	per_unit_discountprice	discount_per	promo_week_flg	age	week_no	year_no	lag_1_sales_unit	lag_1_discount_per
38	596.368421	536.596053	59.772368	0.100227	0	1	9	2013	NaN	NaN
93	597.924731	566.334516	31.590215	0.052833	0	2	10	2013	38.0	0.100227
122	600.639344	519.528115	81.111230	0.135041	0	3	11	2013	93.0	0.052833
133	598.248120	541.940301	56.307820	0.094121	0	4	12	2013	122.0	0.135041
130	591.307692	476.174077	115.133615	0.194710	0	5	13	2013	133.0	0.094121

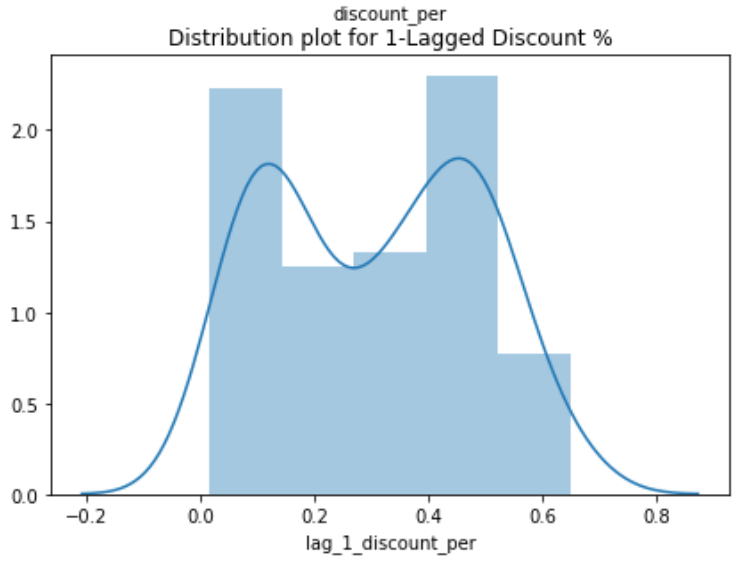
Out[60]:

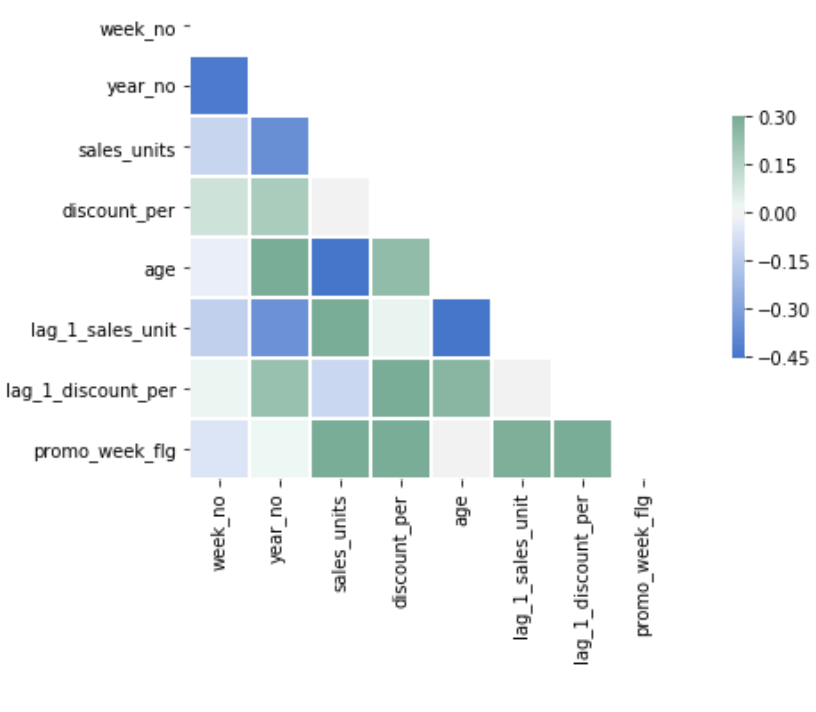
	sales_units	discount_per	age	week_no	year_no	lag_1_sales_unit	lag_1_discount_per	promo_week_flg
0	38	0.100227	1	9	2013	NaN	NaN	0
1	93	0.052833	2	10	2013	38.0	0.100227	0
2	122	0.135041	3	11	2013	93.0	0.052833	0

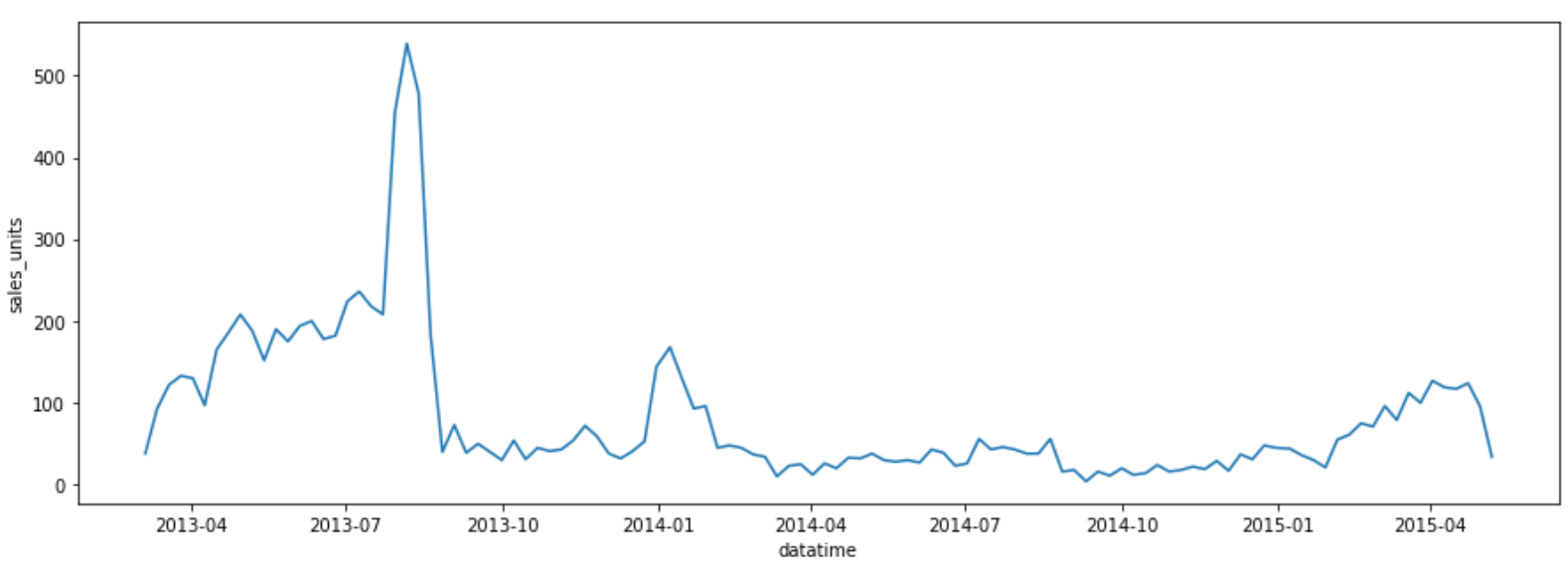


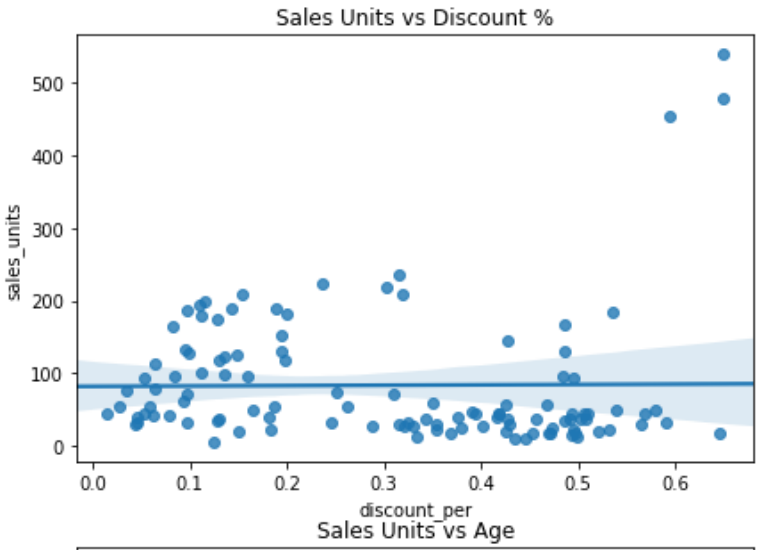


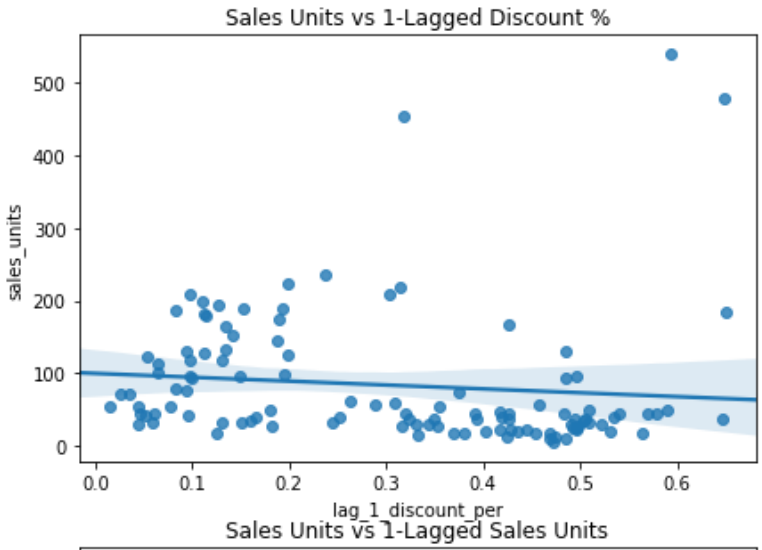


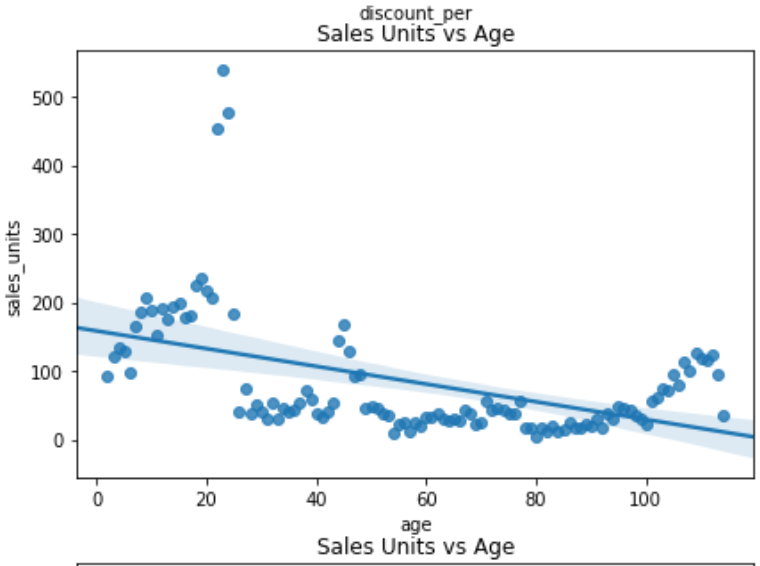


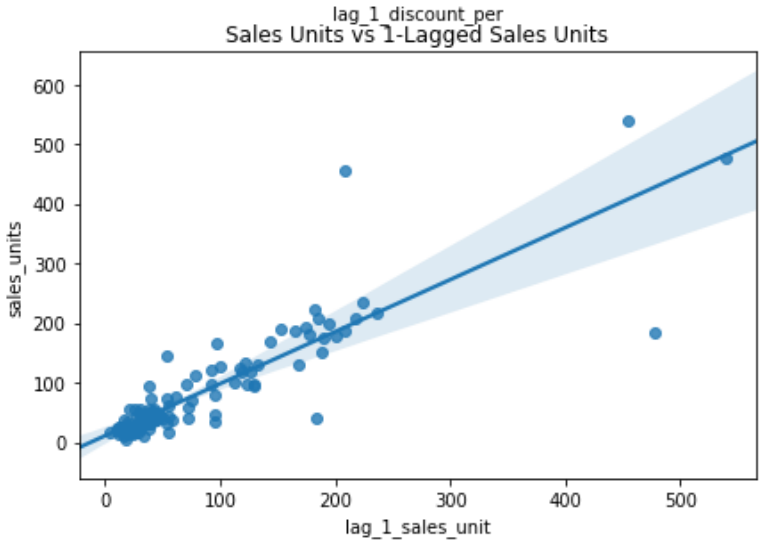


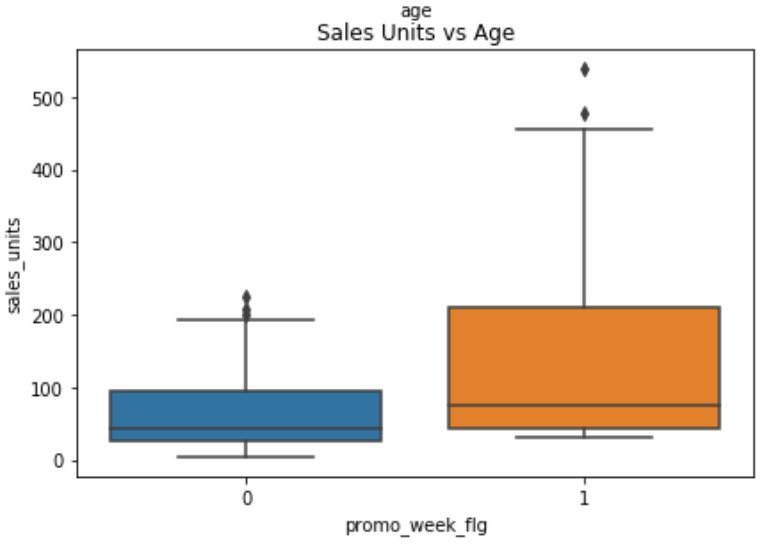


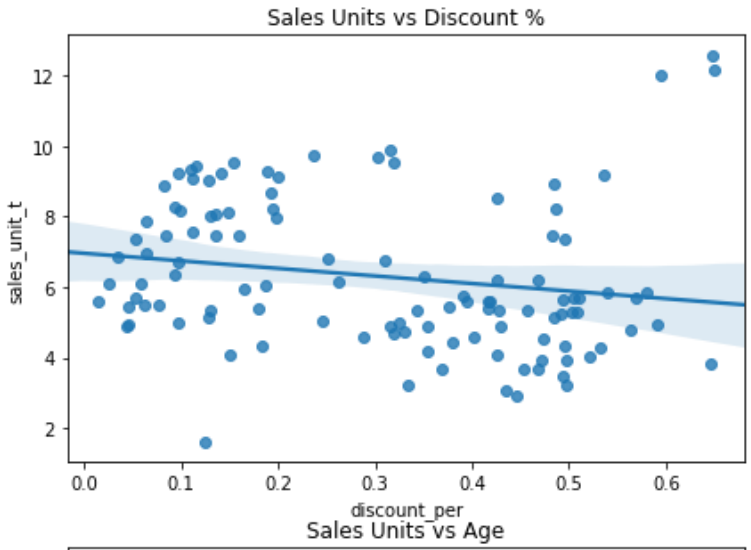

Sales Units vs Discount %
sales_units
discount_per

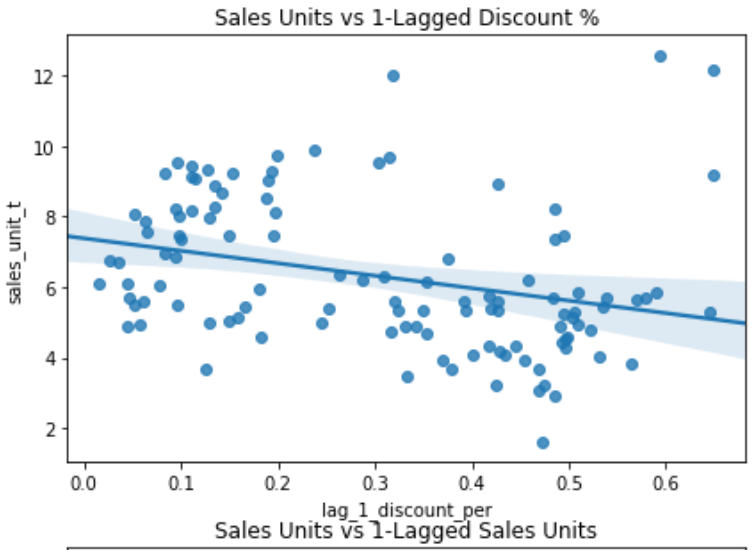

Sales Units vs 1-Lagged Discount %
sales_units
lag_1_discount_per

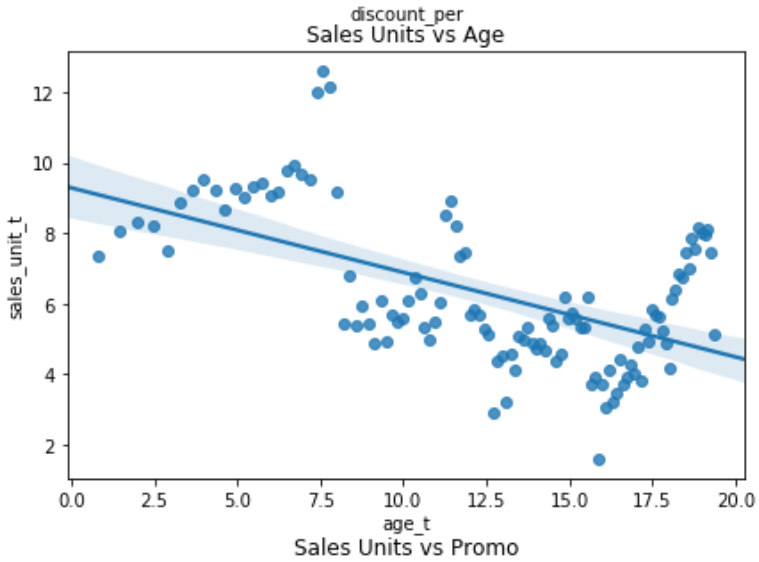

Sales Units vs Age
sales_units
age

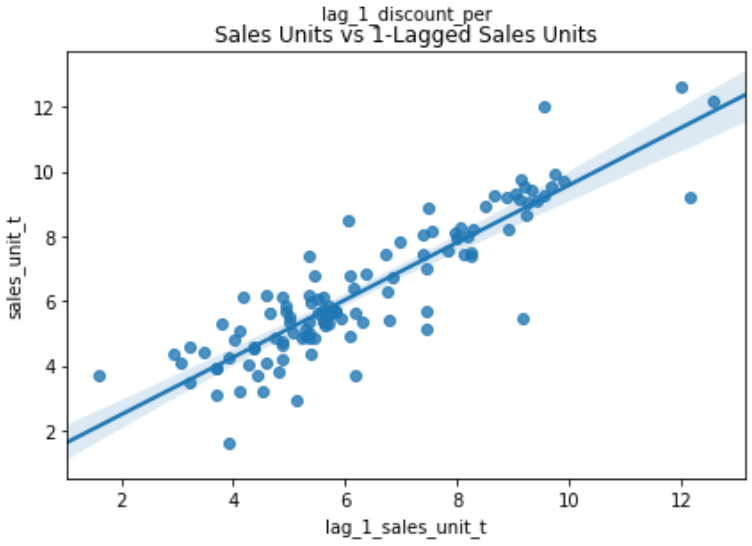

Sales Units vs 1-Lagged Sales Units
sales_units
lag_1_sales_unit

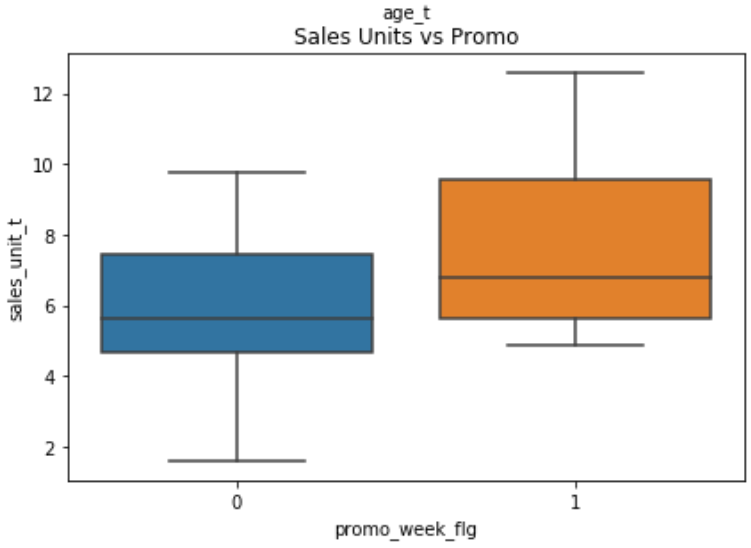

Sales Units vs Age
sales_units
promo_week_flg

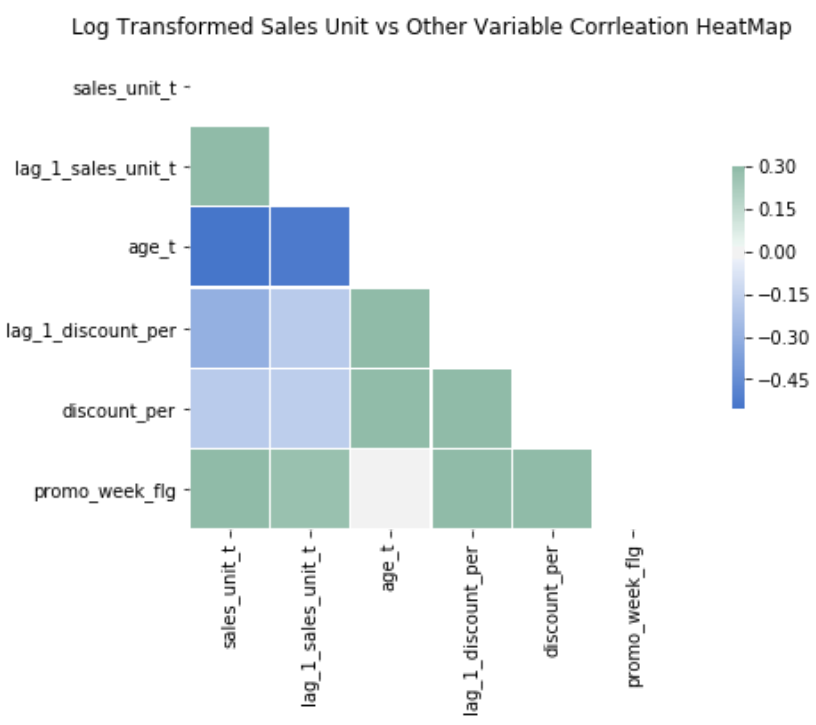

Sales Units vs Discount %
sales_unit_t
discount_per

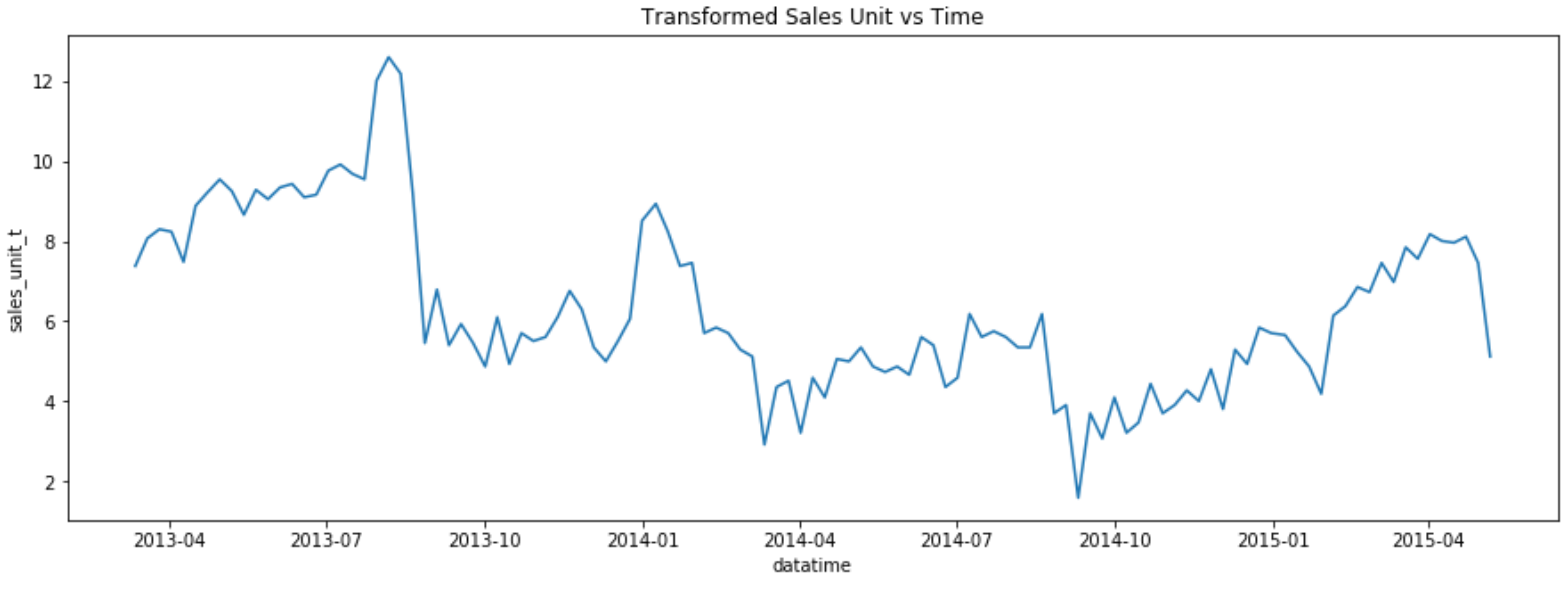

Sales Units vs 1-Lagged Discount %
sales_unit_t
lag_1_discount_per


Sales Units vs Age
sales_unit_t
age_t


Sales Units vs 1-Lagged Sales Units
sales_unit_t
lag_1_sales_unit_t


Sales Units vs Promo
sales_unit_t
promo_week_flg


Log Transformed Sales Unit vs Other Variable Corrleation HeatMap
sales_unit_t
lag_1_sales_unit_t
age_t
lag_1_discount_per
discount_per
promo_week_flg
0.30
0.15
0.00
−0.15
−0.30
−0.45


Transformed Sales Unit vs Time
sales_unit_t
datatime

```
X_train.head(3)
X_train.drop(['sales_units', 'lag_1_sales_unit'], inplace=True, axis='columns')
```

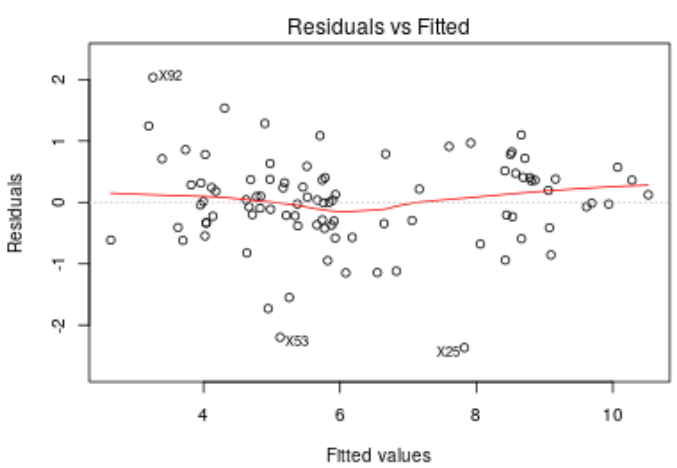
Split and then do predictions

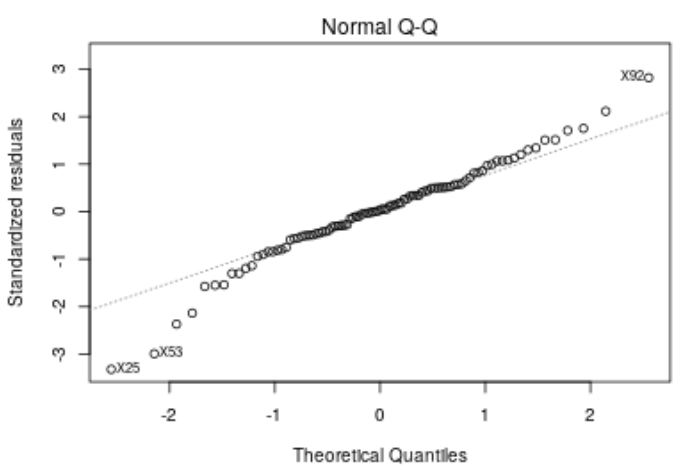
Should we normalize and see if it works better - this will create problems for Optimization though

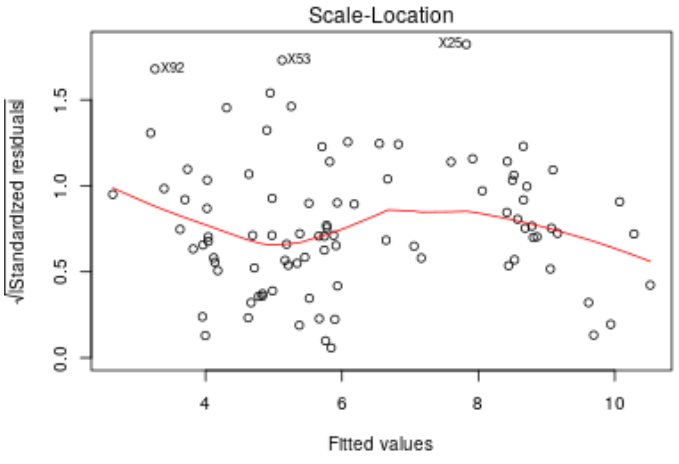
Out[74]:

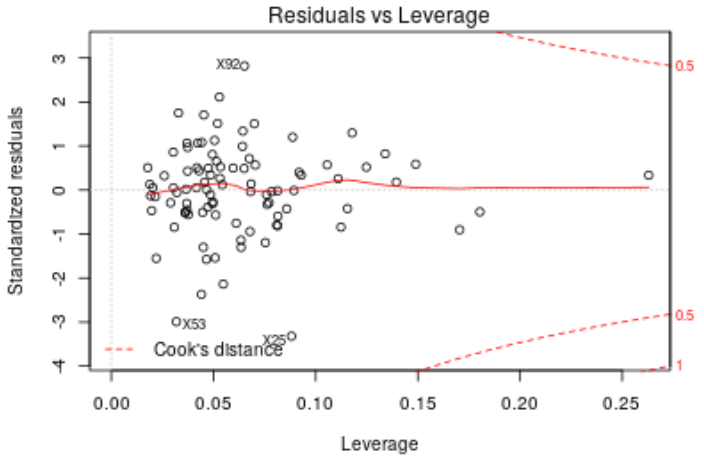
	week_no	year_no	sales_units	discount_per	age	lag_1_sales_unit	lag_1_discount_per	promo_week_flg	sales_unit_t	lag_1_sales_unit_t	age_t	datatime
95	52	2014	45	0.569003	96	48.0	0.579305	1	5.705637	5.844718	17.595918	2014-12-31
96	1	2015	44	0.493558	97	45.0	0.569003	1	5.657628	5.705637	17.697716	2015-01-08
97	2	2015	36	0.491280	98	44.0	0.493558	1	5.238363	5.657628	17.798990	2015-01-15
98	3	2015	30	0.428839	99	36.0	0.491280	1	4.871752	5.238363	17.899749	2015-01-22

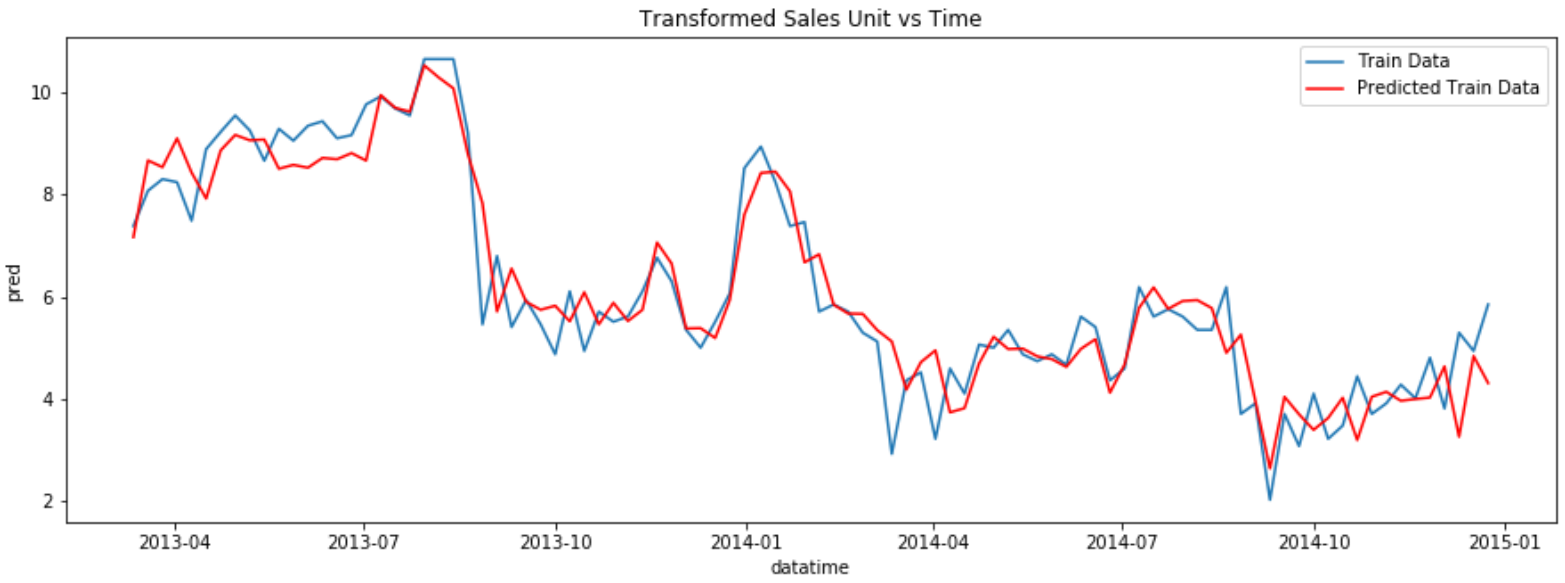
```
Call:
lm(formula = .outcome ~ ., data = dat, verbose = FALSE)

Residuals:
     Min       1Q   Median       3Q      Max
-2.36746 -0.36168  0.02453  0.37913  2.03476

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)        4.41125    0.75806   5.819 9.42e-08 ***
discount_per       3.89091    0.87471   4.448 2.52e-05 ***
age_t             -0.18602    0.04093  -4.545 1.74e-05 ***
lag_1_discount_per -3.19977    0.85670  -3.735 0.000333 ***
lag_1_sales_unit_t  0.56468    0.07503   7.526 4.25e-11 ***
promo_week_flg      0.80126    0.25198   3.180 0.002035 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7466 on 88 degrees of freedom
Multiple R-squared:  0.8818,    Adjusted R-squared:  0.8751
F-statistic: 131.4 on 5 and 88 DF,  p-value: < 2.2e-16

NULL
```











Transformations Used:

Used the following transformations to variables before performing model fit:

- Transform target variable with following Rule $\text{Sales Units} = (\text{Sales Unit}^{.2} - 1)/.2$. This performed best in comparision will all other transformation like log, square root etc.
- Removed Outliers in Sales Units. Any value beyond $mean \pm 1.96\sigma$ was replaced with the upper capped value $mean \pm 1.96\sigma$
- Transform Lagged Sales with following Rule $\text{Sales Units Lagged} = (\text{Sales Unit Lagged}^{.2} - 1)/.2$. Did not remove outlier from this value.
- Transform Age variable with following Rule $\text{Age} = (\text{Age}^{.5} - 1)/.5$

Performing Model Fit

- From the plots above and the Adjusted R-square score (.8751) we can conclude that this is a pretty good model to describe the behavior of the sales units.
- All variables are significant as seen from the p-values
- Dicussion on the validity of the model will be done on the next section

Q-PART B-11

By analysing the models plots we can conclude the following:

- The residual plot is an approximate normal plot. There are deviations from normality for the extreme values, but this can be considered approximately normal
- The mean residual analysis is around zero only. Though there is a very slight curvature, showing there might be a better functional fit, but the curvature is very small
- Residuals are homoscedastic
- The functional form is a linear function of coefficients
- The R-Squared is vand adjested R-Squared is very high (.875) which signifies a good fit

There is correlation among dependent variable, but removing them leads to significant performance degradation. Also the p-values are low for each variable which implies they are **significant** hence we will conclude multi-collinearity is not de-stabilizing the model here.

This is a timeseries data. But we are skipping the Durbin Watson Test for auto-correlation

Q-PART B-12

a - Trend, Seasonality

1. Trend: The variable Age and its relationship with Sales Units is a good proxy for trend. In general the data shows negative trend with time. The correlation between Age and Sales Units is negatively correlated and that captures the trend line
2. Seasonality: this is captured:
    - Partly by regressing on the lagged last week sales units
    - Partly by the Promotion indicator

The above helps capture the seasonal trens in data

b - Price Elasticity

The price elasticity of demand is simply a number; it is not a monetary value. What the number tells you is a 1 % decrease in price causes a y % increase in quantity demanded. In other words, quantity demanded's percentage increase is greater than the percentage decrease in price.

The formula used to calculate the price elasticity of demand is:

$$\eta = \frac{(Q_1 - Q_0)/Q_1 + Q_0)}{(P_1 - P_0)/(P_1 + P_0)}$$

The symbol η represents the price elasticity of demand. The symbol Q_0 represents the initial quantity demanded that exists when the price equals P_0. The symbol Q_1 represents the new quantity demanded that exists when the price changes to P_1.

Considering Week 2, 3 Sales we will derive Price Elasticity:

- Week 2 Price = 566.3 (Discounted @.05%) = P0
- Week 3 Price = 519.5 (Discounted @.19%) = P1
- Week 2 Units Demand / Sold = 93 = Q0
- Week 3 Units Demand / Sold = 122 (Discounted @.19%) = Q1

```
Price Elasticity::-3.1294176108129625
```

Implies 1% drop in price increases demand by 3.1%. Please note this is derived based on two point estimates and may not generalise to entire dataset

c - Promo Indicator

The Promotion indicator indicates, that when there is Promotion on offer on an average **0.8 extra units are sold** than when there is no Promotion

Q-PART B-13

Power of a Model::

Thiel's Coefficient is a good indicator to determine the effectiveness of a forecating model. It calculates the effectiveness of a model prediction against a naive model. We will consider a naive model as one that predicts the prior period value (there are other options like mean model etc. for our example we will consider the naive model forecasts the prior period value as the future value).

$$\text{Theil's Coefficient (U)} = \frac{\sum_{t=1}^{n}(Actual_{t+1} - Model_{t+1})^2}{\sum_{t=1}^{n}(Actual_{t+1} - Naive_{t+1})^2}$$

if U < 1, then the model is considered to be a good forecasting model and better than Naive method.

We will compute the value U over the test set (Periods from 2014 - 52 week till 3rd week of 2015). The exact values are shown in the next section. But we will use the predicted, actual and lagged values for this set.

```
Theil's Coefficient (U) = 0.6363636363636364
```

Out[79]:

	week_no	year_no	sales_units	lag_1_sales_unit	pred_ac	Actual_ModelPred_sq	Actual_NaivePred_sq
95	52	2014	45	48.0	43.0	4.0	9.0
96	1	2015	44	45.0	36.0	64.0	1.0
97	2	2015	36	44.0	35.0	1.0	64.0
98	3	2015	30	36.0	31.0	1.0	36.0

The value is U < 1. Hence this is an more effective model than a Naive Model.

WHITE Noise wrt to Forecasting Models:

An additive forecasting model can be represented by:

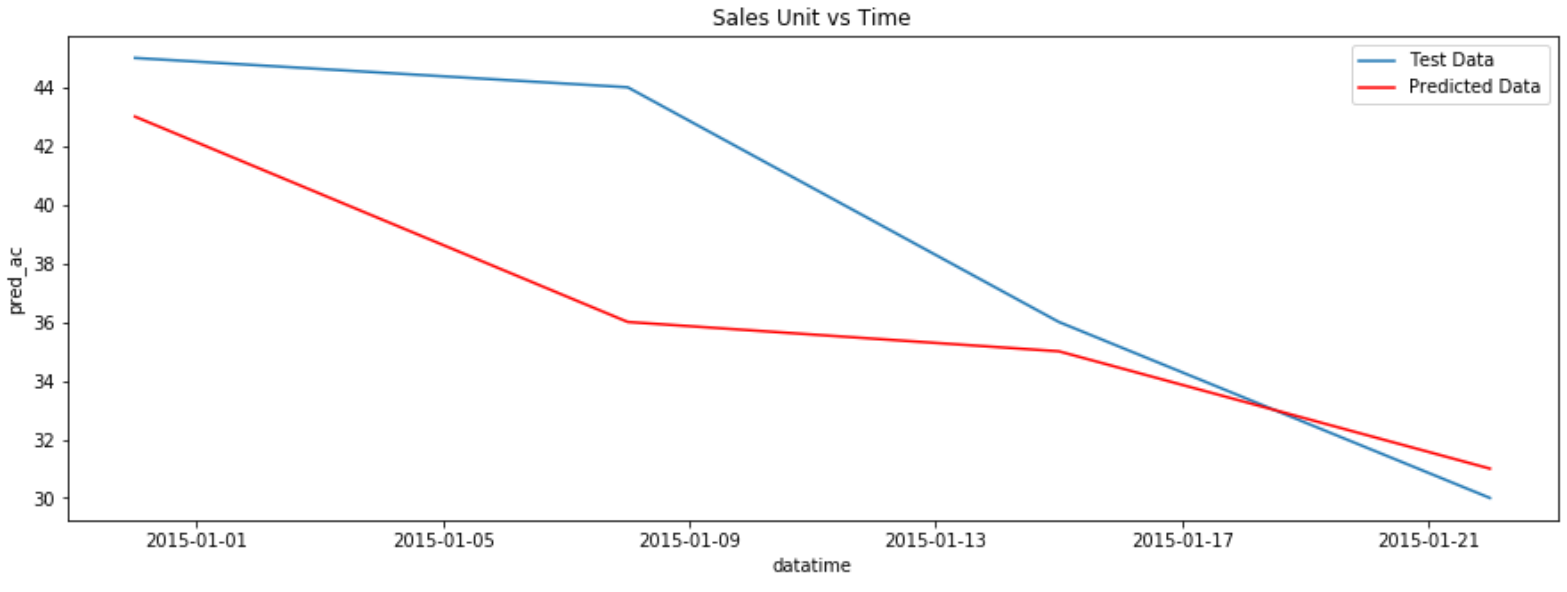
$$Y_t = Trend_t + Seasonal_t + Cycle_t + \epsilon_t$$

where ϵ_t is the **random un-correlated changes (white noise)** with mean zero and constant variance. These are random fluctuations and cannot be explained by any factors.

Q-PART B-14

- Since the last week sales is one of the more important factors for prediction, and it is missing for week 2-4 of test data, I will be predicting weekly sales values and use predicted value of week 1 (2014 - 52 week) as the Lageed week value for predicting week 1 of 2015, (we cannot have nulls in that column).
- This is not optimal as we will accumulate the errors in prediction, but given the situation where we are using last week sales (Lagged 1 week) as one of the key predictors, this is the only option we have.
- The following are the prediction values using this approach

	week_no	year_no	sales_units	discount_per	age	lag_1_sales_unit	lag_1_discount_per	promo_week_flg	sales_unit_t	lag_1_sales_unit_t	age_t	datatime	
95	52	2014	45	0.569003	96	48.0	0.579305	1	5.705637	5.844718	17.595918	2014-12-31	5.
96	1	2015	44	0.493558	97	45.0	0.569003	1	5.657628	5.705637	17.697716	2015-01-08	5.
97	2	2015	36	0.491280	98	44.0	0.493558	1	5.238363	5.657628	17.798990	2015-01-15	5.
98	3	2015	30	0.428839	99	36.0	0.491280	1	4.871752	5.238363	17.899749	2015-01-22	4.



MAPE

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\frac{|Y_t - F_t|}{Y_t}$$

	week_no	year_no	sales_units	lag_1_sales_unit	pred_ac	Actual_Model_abs	Actual_Model_abs_Actual
95	52	2014	45	48.0	43.0	2.0	0.044444
96	1	2015	44	45.0	36.0	8.0	0.181818
97	2	2015	36	44.0	35.0	1.0	0.027778
98	3	2015	30	36.0	31.0	1.0	0.033333

```
MAPE = 7.184343434343435
Naive Model MAPE = 12.790404040404042
We can conclude from MAPE values that our model is better than the Naive model predicting last week sales.
```

Q-PART C-15

```
Status = optimal
demand = 16.659688
demand = 22.480693
demand = 33.425632
demand = 41.424099
disc = 0.100000
disc = 0.150475
disc = 0.315890
disc = 0.458748
Objective = 48103.861923
```

```
2 Set Declarations
    a : Dim=0, Dimen=1, Size=4, Domain=None, Ordered=False, Bounds=(1, 4)
        [1, 2, 3, 4]
    age_index : Dim=0, Dimen=1, Size=4, Domain=None, Ordered=False, Bounds=(1, 4)
        [1, 2, 3, 4]

1 Param Declarations
    age : Size=4, Index=age_index, Domain=Any, Default=None, Mutable=False
        Key : Value
          1 :    96
          2 :    97
          3 :    98
          4 :    99

4 Var Declarations
    demand : Size=4, Index=a
        Key : Lower : Value              : Upper : Fixed : Stale : Domain
          1 : 1e-20 : 16.659688185673737 :  2500 : False : False :  Reals
          2 : 1e-20 :  22.48069308164154 :  2500 : False : False :  Reals
          3 : 1e-20 :  33.42563177633806 :  2500 : False : False :  Reals
          4 : 1e-20 : 41.424098774161884 :  2500 : False : False :  Reals
    disc : Size=4, Index=a
        Key : Lower : Value               : Upper : Fixed : Stale : Domain
          1 :   0.1 :                 0.1 :   0.6 : False : False :  Reals
          2 :   0.1 : 0.15047517522615667 :   0.6 : False : False :  Reals
          3 :   0.1 :  0.3158899089210391 :   0.6 : False : False :  Reals
          4 :   0.1 :  0.4587481067486615 :   0.6 : False : False :  Reals
    inventory : Size=4, Index=a
        Key : Lower : Value              : Upper : Fixed : Stale : Domain
          1 : 1e-20 : 1930.0598360084086 :  2500 : False : False :  Reals
          2 : 1e-20 :  1577.510981033917 :  2500 : False : False :  Reals
          3 : 1e-20 : 1212.9070728749755 :  2500 : False : False :  Reals
          4 : 1e-20 :  518.1459703006967 :  2500 : False : False :  Reals
    saleVal : Size=4, Index=a
        Key : Lower             : Value : Upper : Fixed : Stale : Domain
          1 : 363.59999999999997 :  None :   606 : False :  True :  Reals
          2 : 363.59999999999997 :  None :   606 : False :  True :  Reals
          3 : 363.59999999999997 :  None :   606 : False :  True :  Reals
          4 : 363.59999999999997 :  None :   606 : False :  True :  Reals

1 Objective Declarations
    o : Size=1, Index=None, Active=True
        Key  : Active : Sense    : Expression
        None :   True : maximize : 606 * demand[1] * ( 1 - disc[1] ) + 606 * demand[2] * ( 1 - disc[2] ) + 606 * demand[3]
* ( 1 - disc[3] ) + 606 * demand[4] * ( 1 - disc[4] )

5 Constraint Declarations
    c1 : Size=4, Index=a, Active=True
        Key : Lower : Body                                        : Upper  : Active
          1 :  -Inf :                inventory[1] - demand[1] : 2476.0 :   True
          2 :  -Inf : inventory[2] - demand[2] - inventory[1] :    0.0 :   True
          3 :  -Inf : inventory[3] - demand[3] - inventory[2] :    0.0 :   True
          4 :  -Inf : inventory[4] - demand[4] - inventory[3] :    0.0 :   True
    c2 : Size=4, Index=a, Active=True
        Key : Lower : Body
                                : Upper : Active
          1 :  -Inf :                                                                  demand[1] - ( 1 + 0.2*( 3.387045711
97895 + 3.89091*disc[1] ) )**5.0 :   0.0 :   True
          2 :  -Inf : demand[2] - ( 1 + 0.2*( 1.9203809434197776 + 3.89091*disc[2] - 3.19977*disc[1] + 2.8233999999999995*
( -1 + demand[1]**0.2 ) ) )**5.0 :   0.0 :   True
          3 :  -Inf : demand[3] - ( 1 + 0.2*( 1.9015419037829964 + 3.89091*disc[3] - 3.19977*disc[2] + 2.8233999999999995*
( -1 + demand[2]**0.2 ) ) )**5.0 :   0.0 :   True
          4 :  -Inf : demand[4] - ( 1 + 0.2*( 1.8827987389885315 + 3.89091*disc[4] - 3.19977*disc[3] + 2.8233999999999995*
( -1 + demand[3]**0.2 ) ) )**5.0 :   0.0 :   True
    c3 : Size=4, Index=a, Active=True
        Key : Lower : Body              : Upper : Active
          1 :   0.1 :           disc[1] :  +Inf :   True
          2 :  -Inf : disc[1] - disc[2] :   0.0 :   True
          3 :  -Inf : disc[2] - disc[3] :   0.0 :   True
          4 :  -Inf : disc[3] - disc[4] :   0.0 :   True
    c4 : Size=4, Index=a, Active=True
        Key : Lower : Body                      : Upper  : Active
          1 :  -Inf :                demand[1] : 2476.0 :   True
          2 :  -Inf : demand[2] - inventory[1] :    0.0 :   True
          3 :  -Inf : demand[3] - inventory[2] :    0.0 :   True
          4 :  -Inf : demand[4] - inventory[3] :    0.0 :   True
    c5 : Size=1, Index=None, Active=True
        Key  : Lower : Body                                                : Upper  : Active
        None :  -Inf : demand[1] + demand[2] + demand[3] + demand[4] : 2476.0 :   True

13 Declarations: a demand disc saleVal inventory age_index age c1 c2 c3 c4 c5 o
```

Analysis

We have used non-linear optimization to derive the outcome.

- The discounts offerred as per the plan:
  - Week 1 - 10%
  - Week 2 - 15%
  - Week 3 - 31%
  - Week 4 - 41%
- Units sold would be:
  - Week 1 - 16
  - Week 2 - 22
  - Week 3 - 33
  - Week 4 - 41

- Revenue from Sales: 48103, which is higher than 41302 which he made in reality.

Store would have sold less units 112, compared to the 159 the shop sold, but the revenue generated would have been close to 7000 more (without taking into consideration the extra profit he would make by selling the additional units for 60% discount approximately 11000). So in total, he would have made an extra 18000 by using the above optimization strategy.

Q-2-1

Issues

- Data Collection: If a digital feedback form is not unavailable, collecting data can become a challenging task. Individuals may be motivated to provide feedback on digital forums / platforms than providing feedback on forms (hardcopy / portals). In such cases the following processes can be used to collect data:
  - Data can be manually collected from sources / sites

    This can be automated using the following options:
  - Purchase data from data providers
  - Scrape data (automated) from sites / internet (if privacy policies are not violated)
  - Digitally extracting information from scanned / handwritten documents
- Information Understanding: Documents can be labelled by individuals if forms / digital media does not provide direct ratings. Manually these need to be interpreted and labelled
  - Challenges: Interpretation of sentiment of a comment is subjective to an individual based on his choices and biases. This may lead to inconsistent results
  - Other challenges: Deriving actionble insights from mixed class reviews
- Language challenges:
  - Text data may have mixed language. Translation may not always provide best solutions
  - Using colloquial / vernacular languages for comments may pose challenges to data extraction
  - Spelling mistakes create challenges to context understanding
- Other Challenges:
  - Cumbersome to do analysis manually
  - Many key insights like patterns in negative sentiment over time, its trend analysis, effect of seasonality is difficult to study manually

Text Preprocessing Steps:

- Duplicate removal
- Spelling correction
- Part of Speech Tagging can be performed
- Names Entity Recognition
- Lemmatization / Stemming
- Synonym / Antonym replacements
- Special characters removal
- Stop word removal
- Tokenization (n-gram)
- Create document term matrix (using either counting methods of TFIDF)

Q-2-2

The following EDA can be performed on text data:

- Frequency distributions:
  - Word count frequencies
  - Character count frequencies
  - Punctuation count frequencies
- Time / Trendline
  - Word Count Distribution over the year / month / weeks / days
  - Character Count Distribution over the year / month / weeks / days
  - Date distribution - Comments pulished by Month / Date / Year
  - Publish Date Parameters distribution - Week-Day, Hour
  - Week day vs Week end distribution
  - Sentiment over Time
- Bag of Words / Work Cloud
  - Top Positive vs Top Negative Words Used in the headlines
  - Top Words Usage
  - N-gram Analysis
  - Topic Models Analysis

Q-2-3

Feature Extraction Steps:

- Duplicate removal
- Spelling correction
- Part of Speech Tagging can be performed
- Names Entity Recognition
- Lemmatization / Stemming
- Synonym / Antonym replacements
- Special characters removal
- Stop word removal
- Tokenization (n-gram)
- Create document term matrix (using either counting methods of TFIDF)
- For more advanced processing, GLOVE embeddings can be used to embed words / sentences
- Other Statistical features:
  - Word Counts
  - Character Counts

    - Parts Of Speech Counts
    - Spelling error counts
    - Punctuation count
    - Special Characters Counts
- Depending on business context and data available many other interesting features can be extracted from text data

Q-2-4 ## Add the steps

Out[121]:

	SOURCE	REVIEW BY	REVIEW DATE	REVIEW SUBJECT	text	REVIEW RATING	REVIEW TYPE
0	Trip Advisor	3612	2014-04-30	To commercial	This reativly new temple was a big hindu versi...	2	NEGATIVE
1	Trip Advisor	9573519851	2015-07-24	?Amazing temple in Bangalore?	Me and my friends enjoyed a lot in ISKCON temp...	5	POSITIVE
2	Trip Advisor	???? ?	2016-07-28	A well maintained temple	Otherworldly vibrations throuout the sanctuary...	5	POSITIVE

Out[122]:

	REVIEW ID	SOURCE	REVIEW BY	REVIEW DATE	REVIEW SUBJECT	text	REVIEW RATING	REVIEW TYPE
0	814	Trip Advisor	maryd1928	2013-01-02	Don't miss this Bangalore Temp	Don't miss this Gaudiya Vaishnava Temple locat...	5	POSITIVE
1	1181	Trip Advisor	maryd1928	2013-01-02	Don't miss this Bangalore Temple	Don't miss this Gaudiya Vaishnava Temple locat...	5	POSITIVE
2	813	Trip Advisor	shoubhik	2013-01-05	peacefull and devotional	This was my 2nd visit to the temple.. last tim...	4	POSITIVE

Out[123]:

REVIEW TYPE	% Distribution
MIXED	5.968541
NEGATIVE	5.817712
NEUTRAL	1.982331
POSITIVE	86.231416

From the data distribution above we can see that the data distribution is skewed / imbalanced with 86% belonging to Positive class and merely 6% belonging to Negative and Mixed class with only 2% belonging to Neutral class.

Such imbalance at times can create problems for Models to train properly. There are several options to handle imbalanced data.

Sampling based methods

- We can under-sample the majority class (remove samples randomly leading to loss of information)
    - There can be many variations of the above method, like Clustering based under-sampling, Tomek based under-sampling to name a few
- We can oversample the minority class (random duplication of records, which leads to overfitting issues)
- SMOTE (Synthetic Minority Oversampling TEchnique) consists of synthesizing elements for the minority class, based on those that already exist. It works randomly picking a point from the minority class and computing the k-nearest neighbors for this point. The synthetic points are added between the chosen point and its neighbors.

Other Advanced techniques:

- Algorithms like XGBOOST can handle class imbalance by weighing each class or row of an example to the loss function.
- Neural Networks (Keras) can do the same
- Generative Adverserial Models have been used to synthesize data and can be particularly used in cases with low minority classes (but this is at present particularly complicated with text generation and is an active area of research)

For this exercise we will be **using SMOTE to handle class imbalance problem**

Q-2-5

- Features Engineering: We will perform the following processing on the texts:
    - Remove special characters
    - Remove puctuations
    - Remove Digits
    - Remove html if present
    - Change case to lower case
- Feature Extraction
    - We will use TFIDF Vectorizer to extract features
- Model Building & prediction
    - Split the data into train & test sets (7:3)
    - STEP 1:
        - We will use Naive Bayes model to build model and predict
    - STEP 2:
        - First we will use SMOTE to balance the classes
        - We will use Naive Bayes model to build model and predict

We will compare the Classification Reports of STEP 1 & STEP 2 to compare model outputs to see if SMOTE improved performance or not.

The results will be based on the Test set only

Out[126]:
```
0    reativly new temple big hindu version disney l...
1    friends enjoyed lot iskcon temple one biggest ...
2    otherworldly vibrations throuout sanctuary exc...
Name: text_clean, dtype: object
```

- Typical issues with miss-spelling can be seen below

Out[129]:

	aarati	aarthi	aarti	aarti even	aarti time	abl	abod	absolut	ac	accept	access	accessori	accommod	accord	accordingli	across	across india	activ	activ go	actual	ad
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.138921	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.303034	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Out[130]: (4641, 3692)

Out[132]: Index(['NEGATIVE', 'POSITIVE', 'MIXED', 'NEUTRAL'], dtype='object')

Out[134]: BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)

Out[136]:

	NEGATIVE	POSITIVE	MIXED	NEUTRAL
NEGATIVE	14	47	11	0
POSITIVE	18	1107	85	6
MIXED	13	45	23	0
NEUTRAL	2	17	5	0

```
              precision    recall  f1-score   support

           0       0.30      0.19      0.24        72
           1       0.91      0.91      0.91      1216
           2       0.19      0.28      0.22        81
           3       0.00      0.00      0.00        24

   micro avg       0.82      0.82      0.82      1393
   macro avg       0.35      0.35      0.34      1393
weighted avg       0.82      0.82      0.82      1393
```

```
After OverSampling, the shape of train_X: (11144, 3692)
After OverSampling, the shape of train_y: (11144,)

After OverSampling, counts of label '1': 2786
After OverSampling, counts of label '0': 2786
After OverSampling, counts of label '2': 2786
After OverSampling, counts of label '3': 2786
```

Out[140]:

	NEGATIVE	POSITIVE	MIXED	NEUTRAL
NEGATIVE	14	47	11	0
POSITIVE	18	1107	85	6
MIXED	13	45	23	0
NEUTRAL	2	17	5	0

```
              precision    recall  f1-score   support

           0       0.37      0.21      0.27        72
           1       0.91      0.95      0.93      1216
           2       0.23      0.22      0.23        81
           3       0.00      0.00      0.00        24

   micro avg       0.85      0.85      0.85      1393
   macro avg       0.38      0.34      0.35      1393
weighted avg       0.82      0.85      0.84      1393
```

As we can clearly see above using SMOTE we can improve the overall model performace. Few things of note:

- This is not the best possible model. The assumption here is accuracy is not important and hence I will refrain from tuning this model any further
- This data can be used to generate further improved predictions by performing the following:
  - We can clearly see the same word 'AARTI' is written in quite a few ways. This is a typical example of mis-spelling. Spelling have to be corrected. Some corrections can be automated, while some have to be manually editted in case of Vernacular words being used etc.
  - Adding Statistical features like count of words, characters, Whole UPPER Case letter, use of capitalization, use of puctuation, special characters as features in many situtaions improve the performances.
  - Moreover GLOVE embeddings and more advanced models like Neural Networks or GBTs will surely improve further performance

Q-2-6

There are two major approaches to sentiment analysis.

- Supervised machine learning or deep learning approaches

- Unsupervised lexicon-based approaches

For the first approach we typically need pre-labeled data. (exactly what we did for above solution).

In the second scenario, we do not have the convenience of a well-labeled training dataset. Hence, we will need to use unsupervised techniques for predicting the sentiment by using knowledgebases, ontologies, databases, and lexicons that have detailed information, specially curated and prepared just for sentiment analysis. A lexicon is a dictionary, vocabulary, or a book of words. In this case, lexicons are special dictionaries or vocabularies that have been created for analyzing sentiments. Most of these lexicons have a list of positive and negative polar words with some score associated with them, and using various techniques like the position of words, surrounding words, context, parts of speech, phrases, and so on, scores are assigned to the text documents for which we want to compute the sentiment. After aggregating these scores, we get the final sentiment.

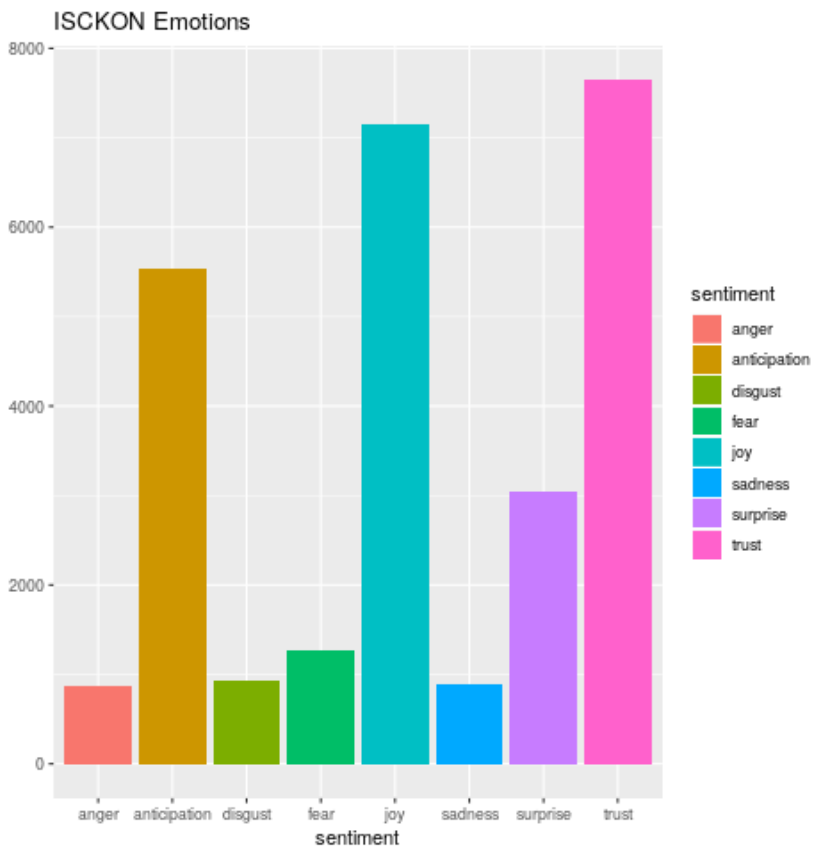
NLTK VADER is an example of an Lexicon based sentiment analyser

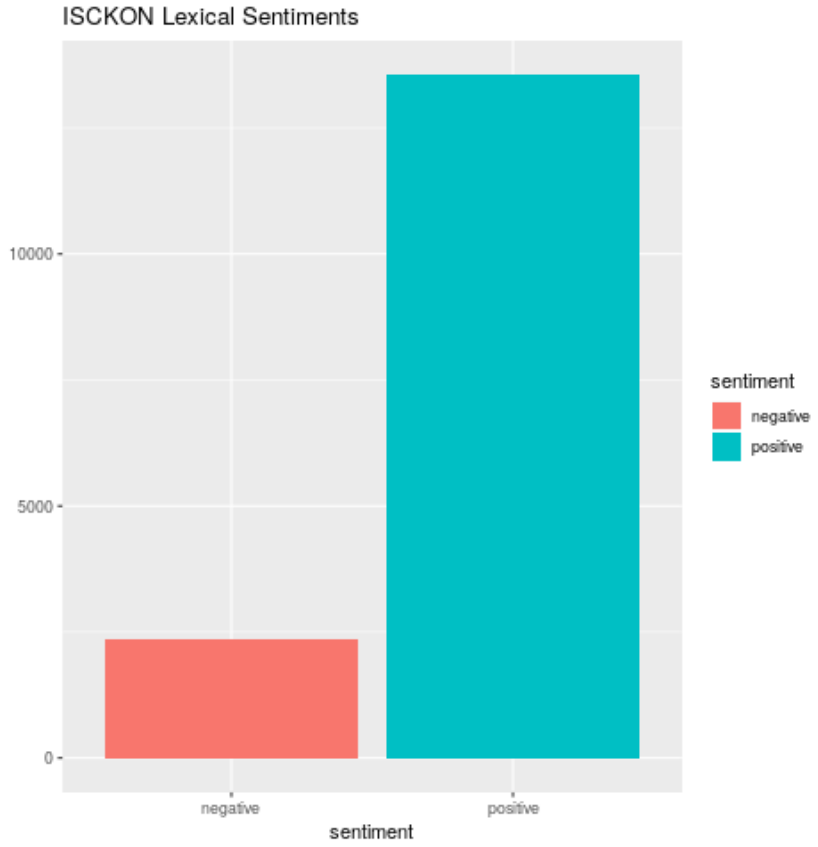
Q-2-7

Since there is no labeled dataset for emotions, we will be using a Lexical based NRC emotion vocabulary to solve this problem

- Features Engineering: We will perform the following processing on the texts:
  - Remove special characters
  - Remove puctuations
  - Remove Digits
  - Remove html if present
  - Change case to lower case
- Emotion Extraction: The we will use NRC lexical analysis to extract the emotion / words from the corpus
- Plots
  - Emotion chart
  - Sentiment Chart as derived from Lexical analysis
  - Work vs Emotion mapping

Each comment Emotion can be identified by consider the maximum presence of any category of emotion. This is not the optimal solution, but since Lexical solutions are based on words, hence the outputs are based on word associations. Sentences / Paragraphs can be associated by summing up the individual word emotions.





```
Error in comparison.cloud(tdm1, random.order = FALSE, colors = c("#00B2FF",  :
  could not find function "comparison.cloud"
```

Q-2-8

Overall Insights

- More visitors have positive sentiments about their visits
- Feeling of JOY and TRUST far outweighs negative emotions
- There is a sizable community of visitors who have expressed ANTICIPATION and SURPRISE
- FEAR, ANGER, SADNESS are relatively less evident but present in many visitors

Q-2-9

Strategy

IT Strategy

- The data collection process needs to be automated
- Sample check of labeled data for validation needs to be performed
- Deployed solution outputs should be made aaccessible / available to public relationship management team

Action Strategy based on Models output

- Identify negative and passive individuals / reviews
- Identify the context for negativity
- Charter a course of action to counter the negativity
  - If services need improvents, devise a strategy for same
  - If its a perception based comment (like too much commercialization), then explain the reason for need for money (the purpose of ISCKON and the various charitable activities and the contribution of the stores to such activities should be explained to people).
- Public relationship management team should be posting approprite replies to negative / passive comments (with envisoned improvement strategy / explaining the need for the stalls etc.)

Q-3

Steps:

- We will create a DF with all possible words
- We will only keep the necessary words as stated in the question
- Build the Model and predict

```
Data with all possible 1,2 Grams.
```

Out[196]:

	ac	ac room	bad	bad experience	beautiful	beautiful restaurant	beds	beds really	behaviour	behaviour room	bites	breakfast	breakfast beautiful	care	care guests	clean	clean beds	cleanliness	cleanlines fir
0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0	
1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1	0	
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
3	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	
4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	

Out[197]:
```
['beautiful',
 'good service',
 'good location',
 'superb',
 'cleanliness',
 'mosquitoes',
 'unfriendly',
 'bad experience']
```

Out[199]:

	unfriendly	cleanliness	superb	good location	beautiful	bad experience	mosquitoes
0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0
2	0	1	1	1	0	0	0
3	0	1	0	0	0	0	0
4	0	0	0	0	0	0	1
5	1	0	0	0	0	0	0
6	0	0	0	0	0	1	0
7	1	0	0	1	0	0	0

Test Data

Out[192]:

	unfriendly	cleanliness	superb	good location	beautiful	bad experience	mosquitoes
7	1	0	0	1	0	0	0

As we can see from above 'good service' is not available in the training set / test set and hence it will not be considered for model building
building Naive Bayes Model..

Out[200]: BernoulliNB(alpha=1.0, binarize=0.0, class_prior=None, fit_prior=True)

Probability of Negative Sentiment is : 0.5633083058452802
Probability of Positive Sentiment is : 0.43669169415472003