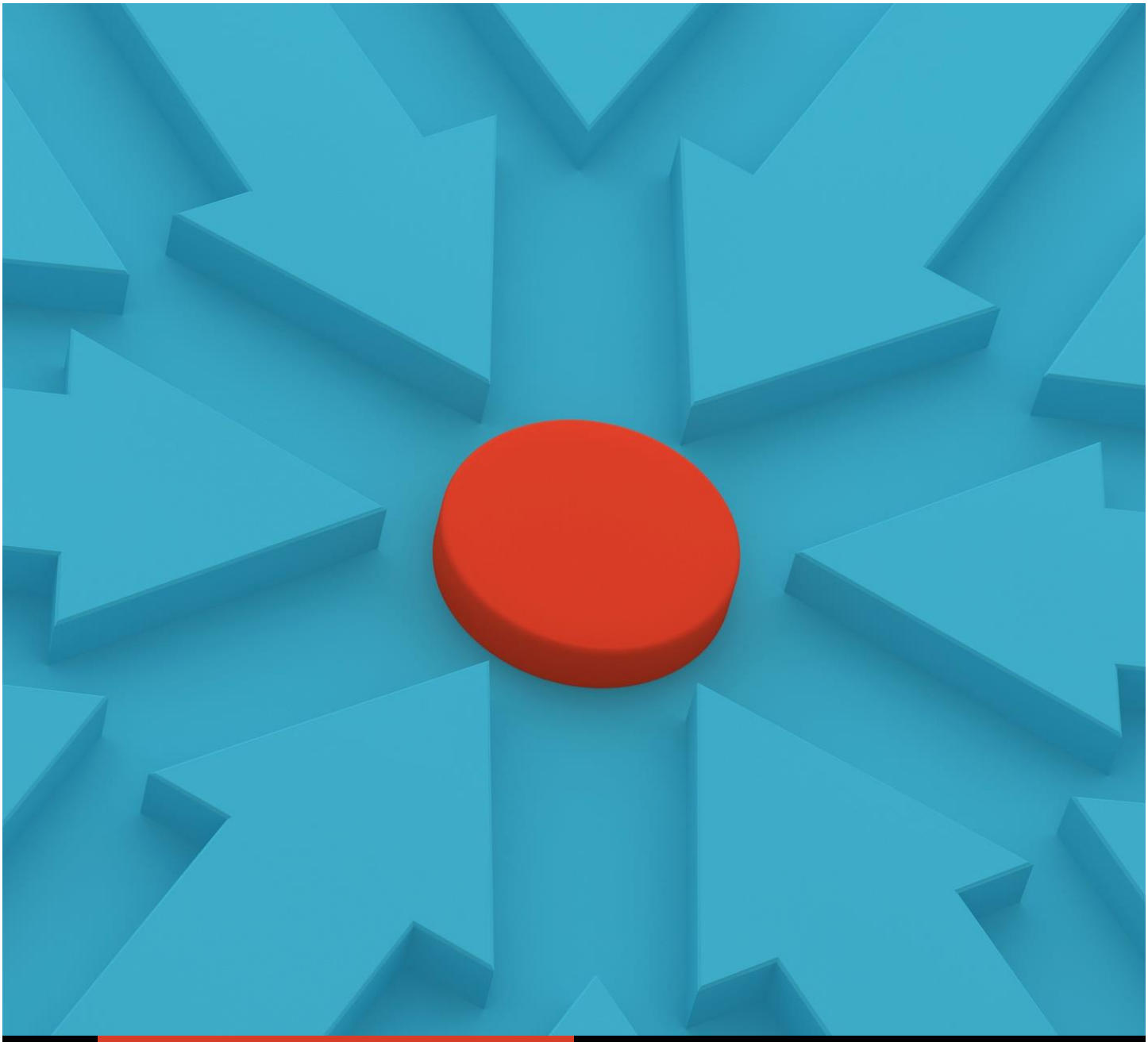


CAPSTONE PROJECT FINAL REPORT

SUPPLY CHAIN

SUNDAR RAM S
PGPDSBA
ONLINE DEC_C 2021
14-DEC-2022

Contents

TABLE OF FIGURES

TABLE OF TABLES

TABLE OF EQUATIONS

INTRODUCTION OF THE BUSINESS PROBLEM

Defining the Problem

A FMCG company has entered into the instant noodles business two years back. Their higher management notices that there is a miss match in the demand and supply. Where the demand is high, supply is pretty low and where the demand is low, supply is pretty high. In both the ways it is an inventory cost loss to the company; hence, the higher management wants to optimize the supply quantity in each and every warehouse in entire country.

The objective of this exercise is to

- Build a model, using historical data that will determine an optimum weight of the product to be shipped each time to the warehouse
- Analyse the demand pattern in different pockets of the country so management can drive the advertisement campaign particular in those pockets.

Need of the project

From the given problem statement, it is understood that there is a miss match in the demand & supply, which is an inventory cost loss to the company. Therefore, the company wants to optimize the supply quantity in each warehouse by determining the optimum weight of the product to be shipped each time to the warehouse. Also, an analysis of demand pattern in different pockets of the country can aid the company in targeted advertisement campaigning, that will help boost sales and hence the increase the bottom line.

Understanding business/social opportunity

Inventory management is an important aspect in managing supply chain. Understanding and the ability to determine demand and supply pattern will aid effective management of the inventory, regulating storage quantity based on demand & supply, thus keeping a check on the total inventory cost from increasing.

As the saying goes – “80% of the cost is accommodated by only 20% of the products”, it becomes need of the hour to classify & manage inventory.

The analysis will also help in targeted campaigning in each region that helps boost the sales and profits.

DATA REPORT, DATA CLEANING, PRE-PROCESSING & EXPLORATORY DATA ANALYSIS

Data Collection Strategy - Understanding how data was collected in terms of time, frequency and methodology

From the problem definition, it is clear that FMCG entered the Noodles business only two years back. Hence, the data collected shall be only from the recent past 2 years. We would be able to understand the data collection strategy, if we have a look at the variables in the dataset.

S.No	Variable Name	Variable Description
1	Ware_house_ID	Product warehouse ID
2	WH_Manager_ID	Employee ID of warehouse manager
3	Location_type	Location of warehouse like in city or village
4	WH_capacity_size	Storage capacity size of the warehouse
5	zone	Zone of the warehouse
6	WH_regional_zone	Regional zone of the warehouse under each zone
7	num_refill_req_l3m	Number of times refilling has been done in last 3 months
8	transport_issue_l1y	Any transport issue like accident or goods stolen reported in last one year
9	Competitor_in_mkt	Number of instant noodles competitor in the market
10	retail_shop_num	Number of retails shop who sell the product under the warehouse area
11	wh_owner_type	Company is owning the warehouse or they have get the warehouse on rent
12	distributor_num	Number of distributer works in between warehouse and retail shops
13	flood_impacted	Warehouse is in the Flood impacted area indicator
14	flood_proof	Warehouse is flood proof indicators. Like storage is at some height not directly on the ground
15	electric_supply	Warehouse have electric back up like generator, so they can run the warehouse in load shedding
16	dist_from_hub	Distance between warehouse to the production hub in Kms
17	workers_num	Number of workers working in the warehouse
18	wh_est_year	Warehouse established year
19	storage_issue_reported_l3m	Warehouse reported storage issue to corporate office in last 3 months. Like rat, fungus because of moisture etc.
20	temp_reg_mach	Warehouse have temperature regulating machine indicator
21	approved_wh_govt_certificate	What kind of standard certificate has been issued to the warehouse from government regulatory body

22	wh_breakdown_l3m	Number of time warehouse face a breakdown in last 3 months. Like strike from worker, flood, or electrical failure
23	govt_check_l3m	Number of time government Officers have been visited the warehouse to check the quality and expire of stored food in last 3 months
24	product_wg_ton	Product has been shipped in last 3 months. Weight is in tons

Table 1 Data Dictionary

From table 1, it can be seen that data is collected for the last 3 months for 5 variables – Product weight, Government Officer Check, Warehouse Breakdown, Warehouse storage issue, Refill frequency. Collected for the last 1 year for transport issue. For the other variables, where in some require update on a daily basis, while some are constant and doesn't change.

The methodology of data collection covers the pockets of demographics/ demographical location, problems it faces, legal alignment, audits, competitors information, etc… thus giving a wider insight on how the ware house is situated.

Visual inspection of data (rows, columns, descriptive details)

	Ware_house_ID	WH_Manager_ID	Location_type	WH_capacity_size	zone	WH_regional_zone	num_refill_req_l3m	transport_issue_l1y
0	WH_100000	EID_50000	Urban	Small	West	Zone 6	3	1
1	WH_100001	EID_50001	Rural	Large	North	Zone 5	0	0
2	WH_100002	EID_50002	Rural	Mid	South	Zone 2	1	0
3	WH_100003	EID_50003	Rural	Mid	North	Zone 3	7	4
4	WH_100004	EID_50004	Rural	Large	North	Zone 5	3	1

Competitor_in_mkt	retail_shop_num	wh_owner_type	distributor_num	flood_impacted	flood_proof	electric_supply	dist_from_hub	workers_num
2	4651	Rented	24	0	1	1	91	29.0
4	6217	Company Owned	47	0	0	1	210	31.0
4	4306	Company Owned	64	0	0	0	161	37.0
2	6000	Rented	50	0	0	0	103	21.0
2	4740	Company Owned	42	1	0	1	112	25.0

wh_est_year	storage_issue_reported_l3m	temp_reg_mach	approved_wh_govt_certificate	wh_breakdown_l3m	govt_check_l3m	product_wg_ton
NaN	13	0	A	5	15	17115
NaN	4	0	A	3	17	5074
NaN	17	0	A	6	22	23137
NaN	17	1	A+	3	27	22115
2009.0	18	0	C	6	24	24071

Figure 1 Sample of the dataset – Shown as 3 Splits

There are a total of 25000 records with 24 variables containing information on each record. While the target variable is Weight in ton, the remaining 23 variables are assumed to be independent and hence called independent variables. The description of each of the variable can be obtained from Table 1.

Understanding of attributes (variable info, renaming if required)

Below table 2 shows the data type of each variable in the dataset.

S.No	Variable Name	Data Type
1	Ware_house_ID	Object
2	WH_Manager_ID	Object
3	Location_type	Object
4	WH_capacity_size	Object
5	zone	Object
6	WH_regional_zone	Object
7	num_refill_req_l3m	int64
8	transport_issue_l1y	int64
9	Competitor_in_mkt	int64
10	retail_shop_num	int64
11	wh_owner_type	Object
12	distributor_num	int64
13	flood_impacted	int64
14	flood_proof	int64
15	electric_supply	int64
16	dist_from_hub	int64
17	workers_num	float64
18	wh_est_year	float64
19	storage_issue_reported_l3m	int64
20	temp_reg_mach	int64
21	approved_wh_govt_certificate	Object
22	wh_breakdown_l3m	int64
23	govt_check_l3m	int64
24	product_wg_ton	int64

Table 2 Variable Data type

There are 14 int64 variables, 8 object type variables & 2 float64 variables. The records in the jupyter notebook are indexed from 0 to 24999, indicating that there are 25000 rows. As already known there are 24 columns/ variables.

Below figure shows the summary of the numerical variables in the dataset.

	count	mean	std	min	25%	50%	75%	max
num_refill_req_l3m	25000.0	4.089040	2.606612	0.0	2.0	4.0	6.0	8.0
transport_issue_l1y	25000.0	0.773680	1.199449	0.0	0.0	0.0	1.0	5.0
Competitor_in_mkt	25000.0	3.104200	1.141663	0.0	2.0	3.0	4.0	12.0
retail_shop_num	25000.0	4985.711560	1052.825252	1821.0	4313.0	4859.0	5500.0	11008.0
distributor_num	25000.0	42.418120	16.064329	15.0	29.0	42.0	56.0	70.0
flood_impacted	25000.0	0.098160	0.297537	0.0	0.0	0.0	0.0	1.0
flood_proof	25000.0	0.054640	0.227281	0.0	0.0	0.0	0.0	1.0
electric_supply	25000.0	0.656880	0.474761	0.0	0.0	1.0	1.0	1.0
dist_from_hub	25000.0	163.537320	62.718609	55.0	109.0	164.0	218.0	271.0
workers_num	24010.0	28.944398	7.872534	10.0	24.0	28.0	33.0	98.0
wh_est_year	13119.0	2009.383185	7.528230	1996.0	2003.0	2009.0	2016.0	2023.0
storage_issue_reported_l3m	25000.0	17.130440	9.161108	0.0	10.0	18.0	24.0	39.0
temp_reg_mach	25000.0	0.303280	0.459684	0.0	0.0	0.0	1.0	1.0
wh_breakdown_l3m	25000.0	3.482040	1.690335	0.0	2.0	3.0	5.0	6.0
govt_check_l3m	25000.0	18.812280	8.632382	1.0	11.0	21.0	26.0	32.0
product_wg_ton	25000.0	22102.632920	11607.755077	2065.0	13059.0	22101.0	30103.0	55151.0

Figure 2 Summary of the numerical variables

Summarizing briefly, the dataset has a total of 25000 records. It can be noticed that the count value in workers_num & wh_est_year are less than 25000, indicating there are null values in the dataset. The 5 point summary (25 percentile, median, 75 percentile, min & max) along with mean & standard deviation of the numerical variables are displayed. It can be seen almost all the variables are skewed. However, skewness is expected to be low comparing the mean and median values.

There are also some binary categorical variables in the above table and are displayed as they are binary & represented by numbers 0 & 1 depicting Yes & No. It seems to be that there are no bad data in the dataset. There is no need to rename any variable in this case.

Duplicated Records/ Bad data/ Anomalies

On evaluating, it can be seen that there are no duplicated records/ Bad data/ anomalies in the dataset.

Null Values and their treatment

On evaluating, it can be seen that there are 13779 null values which is almost 2% of the entire data. On further exploration it is noticed that there are 990 null values in workers_num, 11881 in wh_est_year, & 908 in approved_wh_govt_certificate. Having a total of 25000 records in each of these variables, 3.9%, 47% & 3.6% of the data are null values respectively.

It is important to treat these null values for the below reasons.

- Most of the Machine Learning Algorithms does not work if the dataset contains missing values.
- We may end up building a biased machine learning model which will lead to incorrect results.
- Missing data can lead to a lack of precision in the statistical analysis.

The missing value treatment can be done in many ways. However, we are going to use the forward fill technique of pandas library in python to impute the null values.

However, post using the forward fill, it is noticed that, the data still has 4 missing values in the wh_est_year variable. This is due to the continuous presence of null value in the previous records. Further to treat these 4 null values, we can impute them with the median of that variable which is 2009. Thus, all the null values are treated.

Outliers and their treatment

Outliers are values that are abnormally away from the other values in the dataset. They tend to affect the arithmetic mean of the dataset, abnormally skewing the value to one side (upper or lower), depending on the presence of the outlier. The values below minimum [Q1 - 1.5(Inter Quartile Range)] are the outliers on the lower side of the dataset, while values above maximum [Q3 + 1.5(Inter Quartile Range)] are the outliers on the upper side of the dataset, where Q1 & Q3 are the 25th and 75th percentile respectively.

Boxplot is an excellent plot that gives us the 5 number summary (minimum, Q1, median, Q3, maximum). Q1, median & Q3 are represented by the box and the whiskers denote the values of maximum and minimum

on either side of the box. The outliers are denoted by the points that fall either after the maximum whisker or below the minimum whisker.

Outliers generally affect only the continuous variables and not categorical variables. Hence, our analysis here are restricted to continuous variables. From the below plot, it is noticed that, the variables retail_shop_num & workers_num have outliers present in them and are both in upper and lower side. On further exploration it can be seen that retail_shop_num & workers_num have 948 & 631 outliers respectively that contribute to 3 % & 2 % in the entire count of 25000 records against each variable. Hence, 0.2% (1579 values) of the total data is outlier, which is very low. However, it is important to treat the outliers as they affect the accuracy in model building. One of the methods of treating the outliers is through imputation with the maximum [Q3 + 1.5(Inter Quartile Range)] and minimum [Q1 - 1.5(Inter Quartile Range)] values whichever is applicable.

The lower and upper range i.e. the minimum and maximum values are as in the table below for each of the two variables with which the outliers are imputed accordingly.

Variables	Minimum	Maximum
workers_num	10.5	46.5
retail_shop_num	2532.5	7280.5

Table 3 Minimum & Maximum values - Outlier Treatment

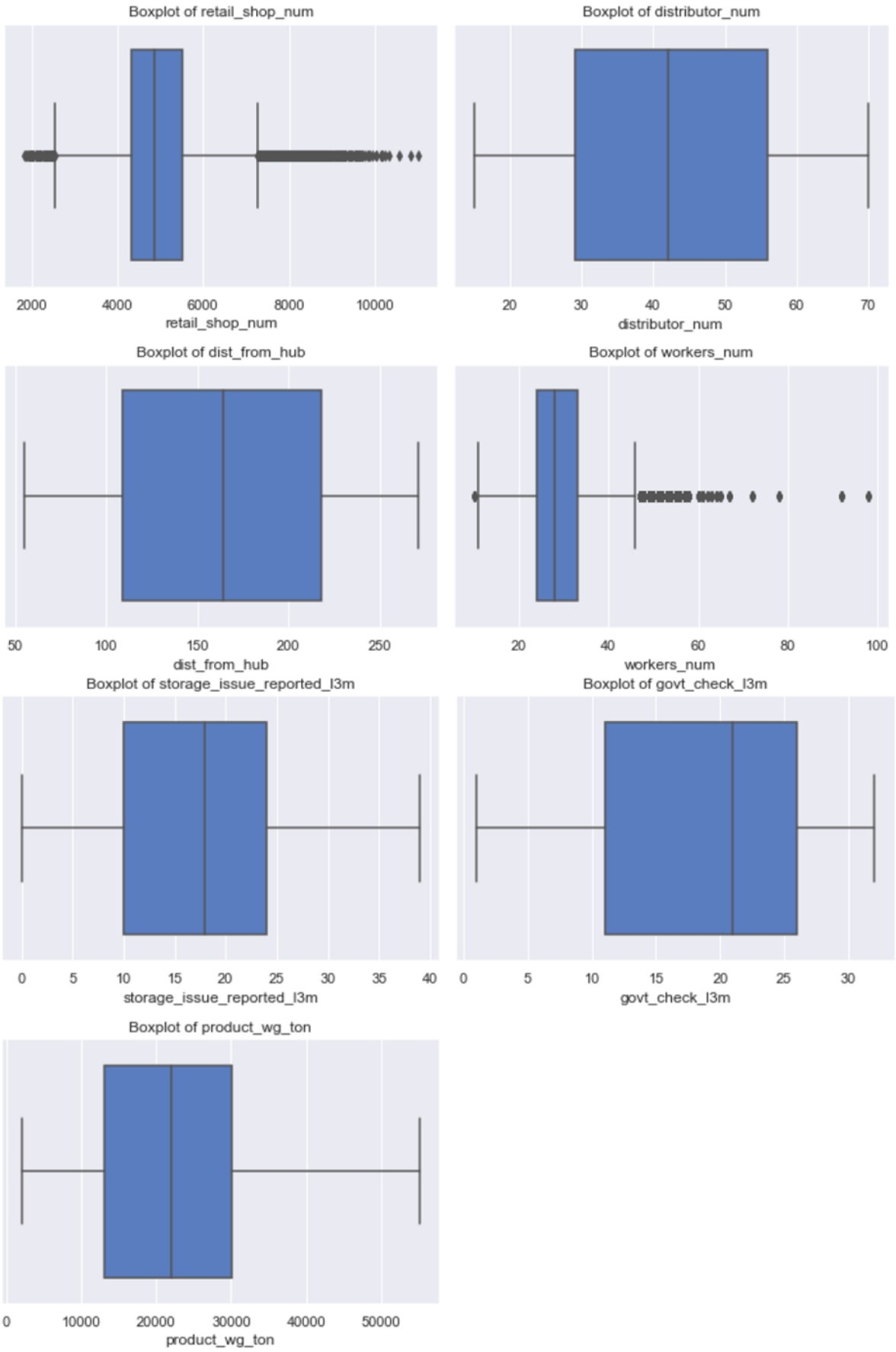


Figure 3 Box Plot - Outlier Check

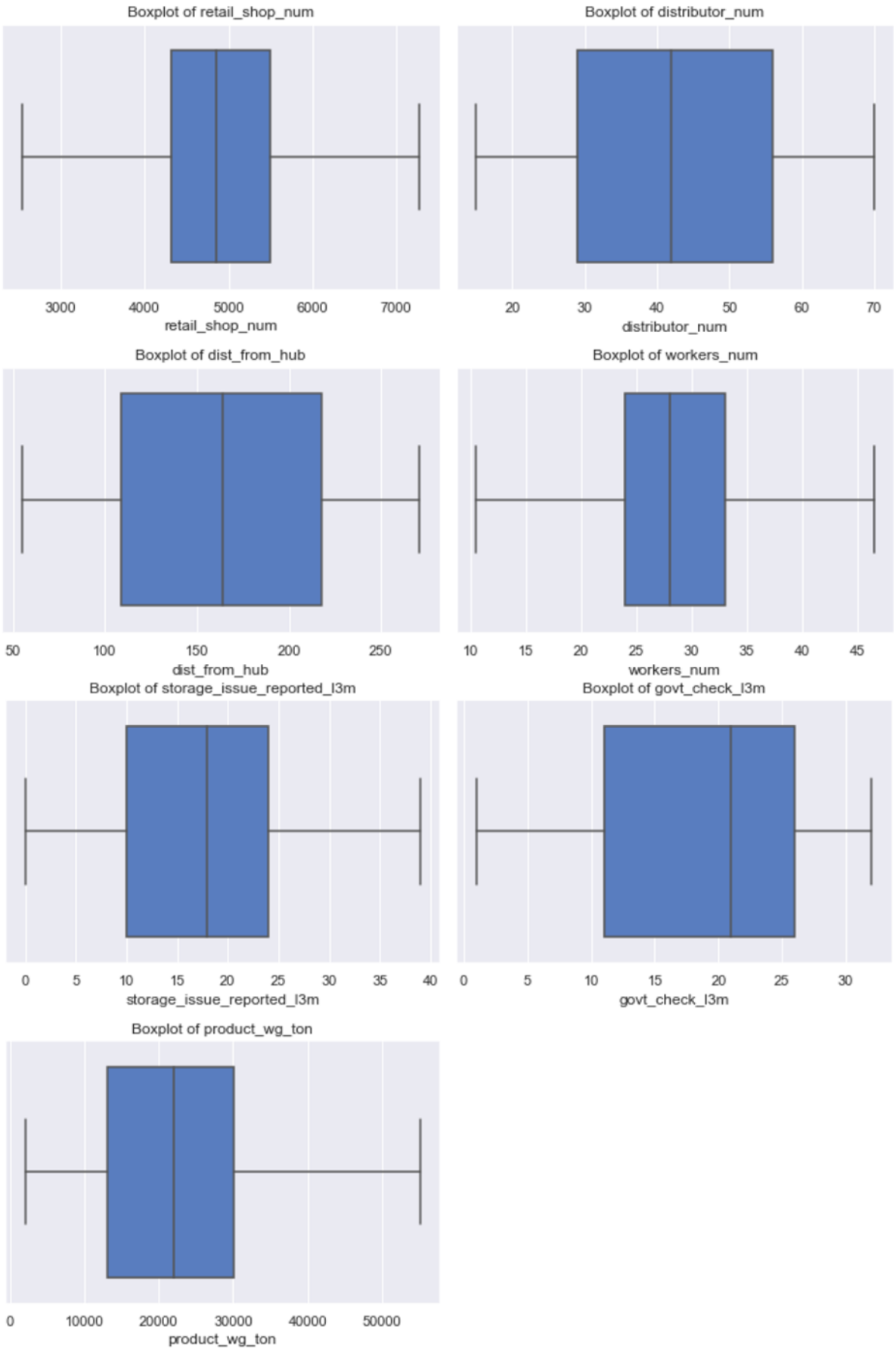


Figure 4 Box Plot - Post Outlier Treatment

Addition of New Variables

In the given data set, we can see that there is an independent variable having details of the Year of Establishment. This variable can be transformed in to age of the ware house, which may give us some untapped insights. In order to transform this variable, we first introduce an additional variable that contains the details of current year and further create an 'age' variable that contains the difference (Current Year – Year of Establishment). Thus the new variable containing the age of the Ware house is created.

We can see that the age of ware house starts from -1 and ends with 26. Age -1 seems to be a bad data. We have the below three options,

- Get back to the customer for clarification
- Assume the age -1 as the ware houses that would be established in the following year
- Consider -1 as bad data and impute them with a suitable imputation technique

Here, Option 3 would not be apt, as imputation might sometimes lead to biased approach and option 1 seems to be non-feasible in the current stage. Hence it would be apt to go ahead with option 2.

On the other hand, age 0 can be interpreted as ware houses that are yet to complete one year after establishment i.e., they are established in the current year only.

Binning - Variable Transformation (1)

Further to the addition & interpretation of a new variable, to make them more usable for our analysis, we can bin the age in the categories as listed below.

Age Bins	Description
-1	Yet to Establish
0	Less than 1 year
1 to 9	New
10 to 17	Mediocre
18 to 26	Expert

Table 4 Age Bins

Similar binning can be done for weight as below for getting better insights.

Weight Bins	Description
2065 to 17695.33	Low
17695.34 to 35390.67	Medium
35390.68 to 55151	High

Table 5 Weight Bins

Univariate Analysis

HISTOGRAMS

Histogram helps us understand the distribution, while box plot helps us understand the five-point summary.

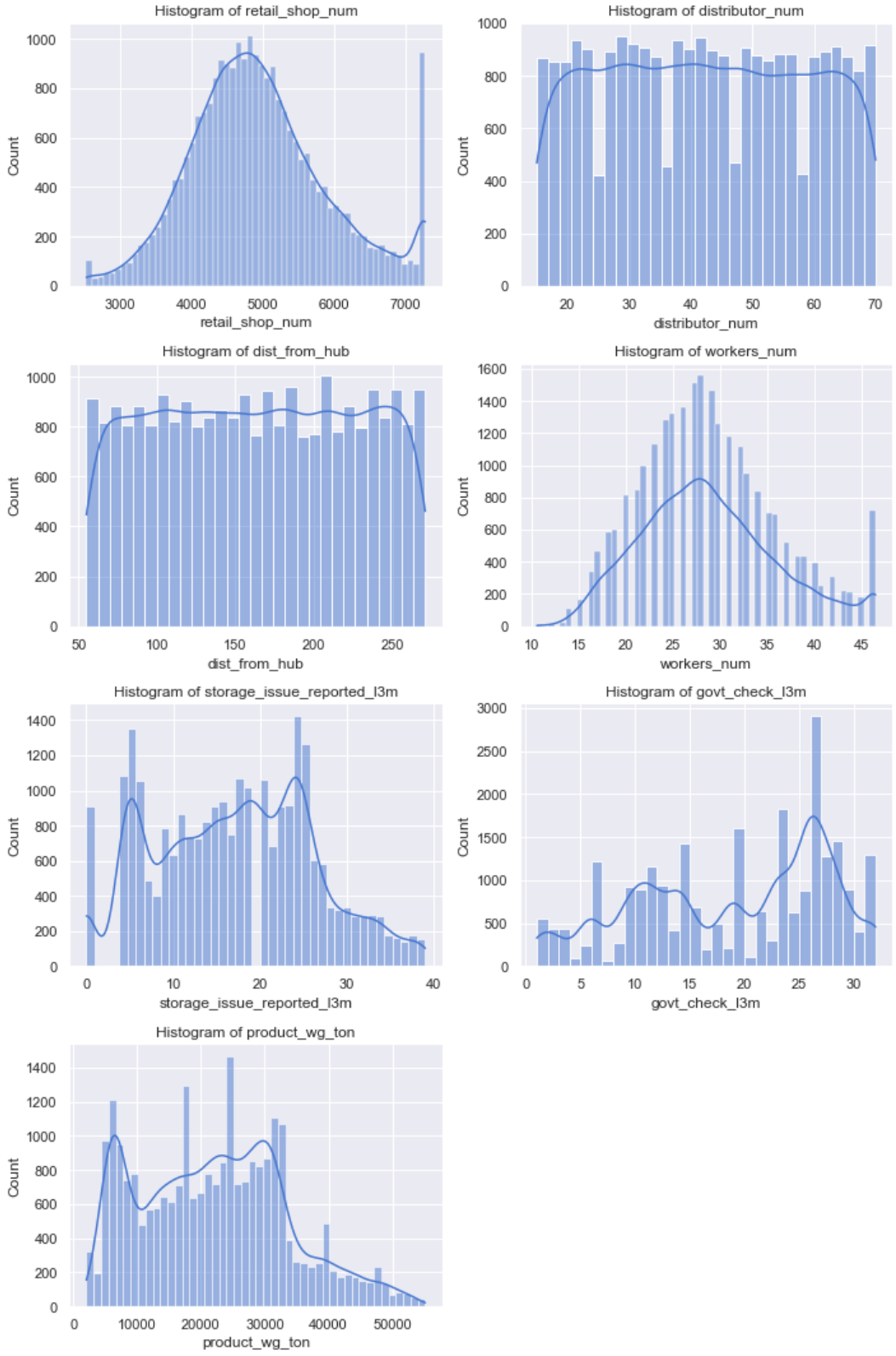
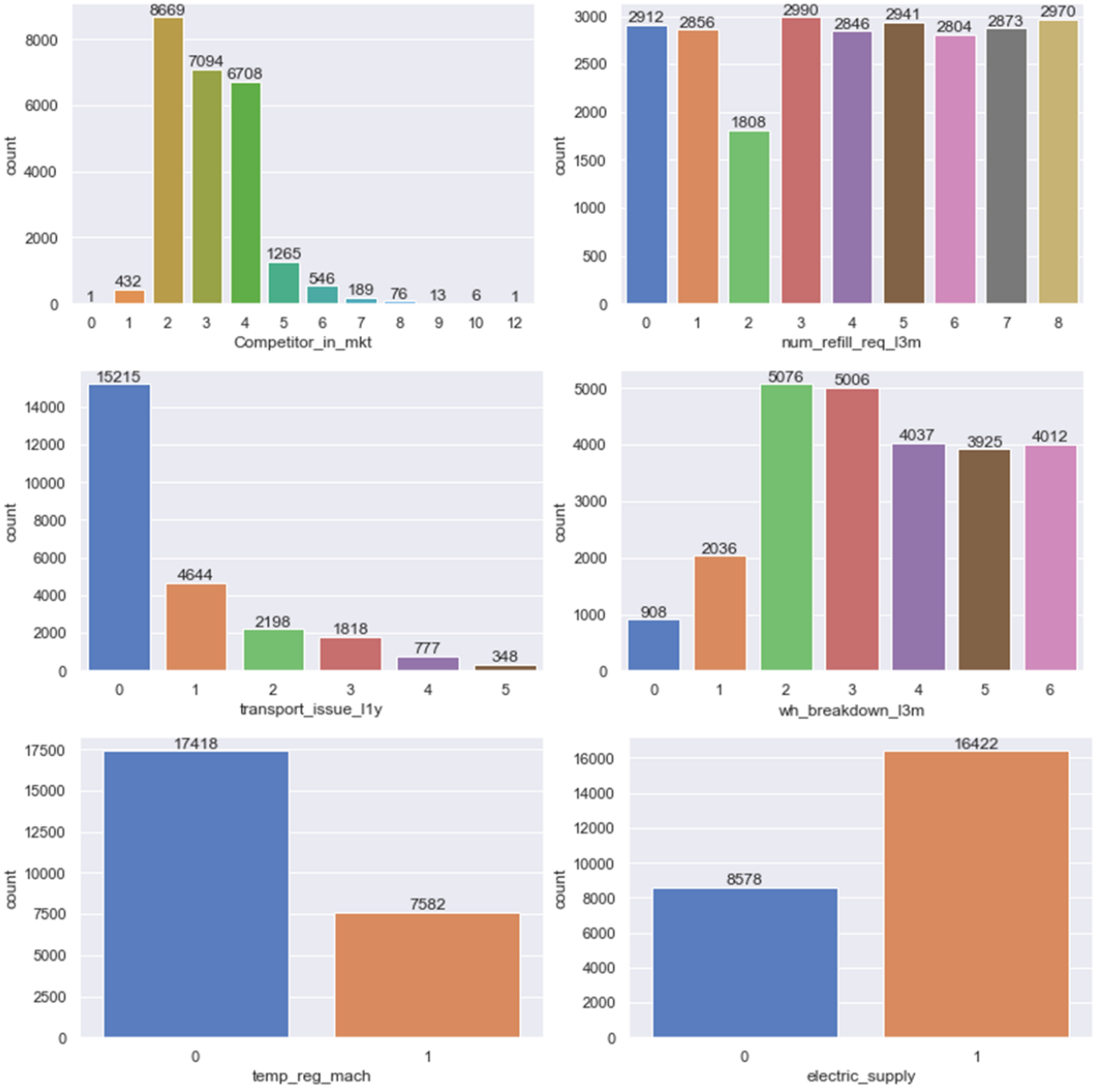
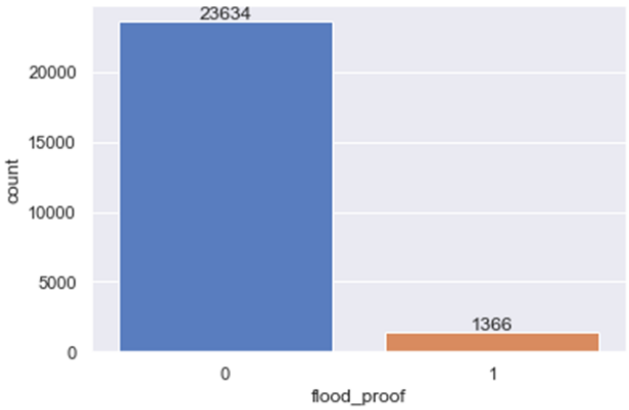


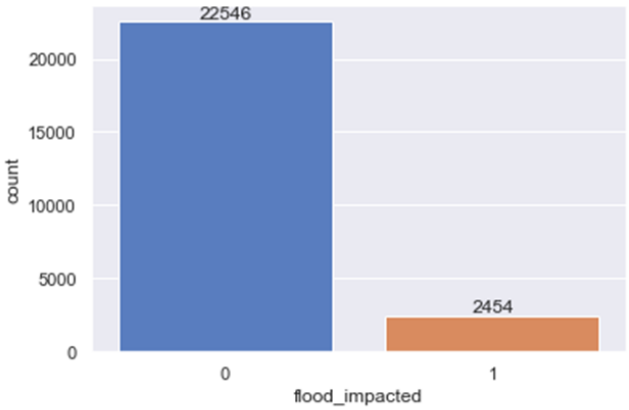
Figure 5 Histogram Plot – Continuous Variables

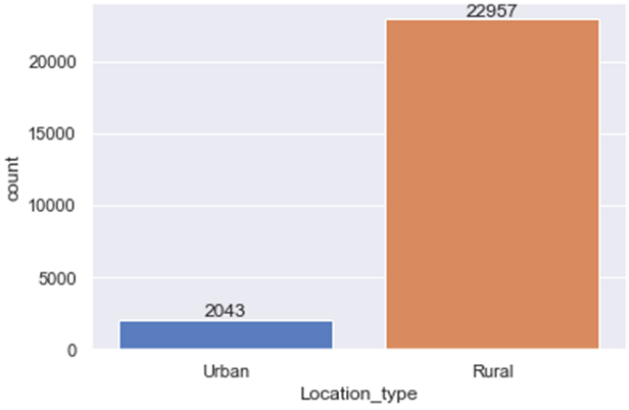
From the above Histogram plots it can be seen that retail_num_shops & workers_num are more or less normally distributed, but the peak at the end of the curve suggest that they are not normally distributed. Multiple irregular peaks can be seen in storage_issue_reports_l3m, govt_check_l3m & product_wg_ton. These shows certain clustering of data. However, we will not be exploring clusters in this particular case, as clustering doesn't seem to have significance considering the nature of the variable these 3 are. Distributor_num and dist_from_hub are found to have flat top.

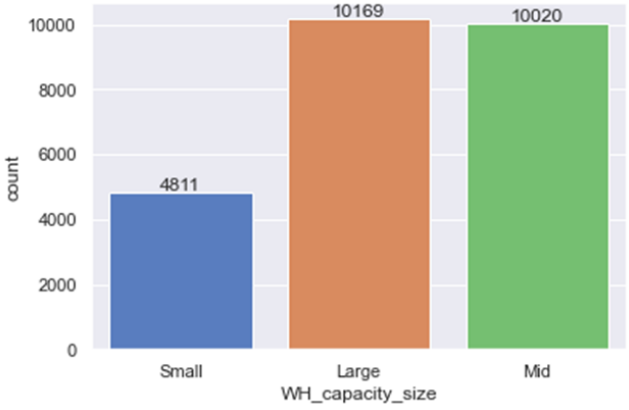
COUNT PLOTS

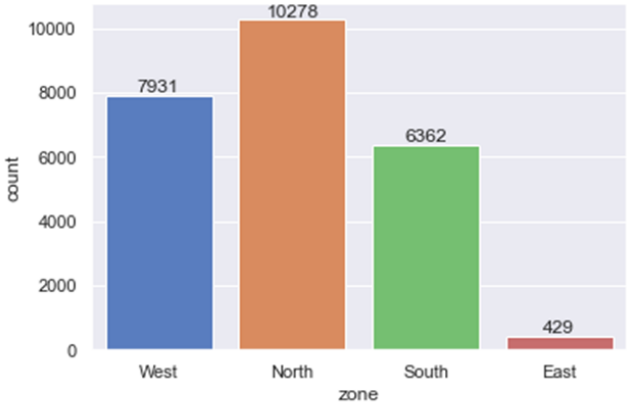


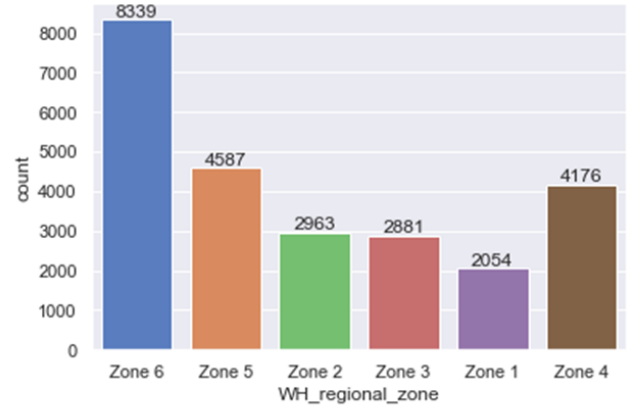

23634
1366
count
flood_proof
0
1

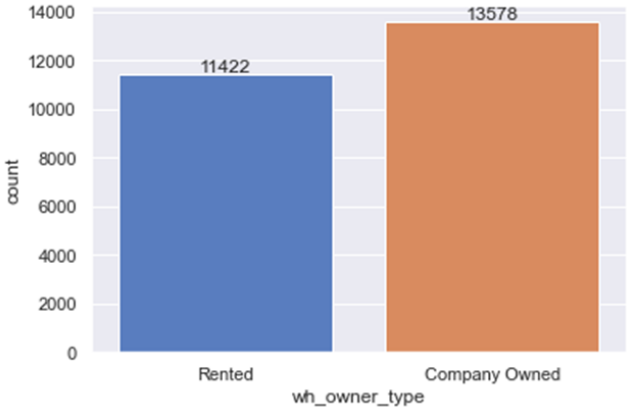

22546
2454
count
flood_impacted
0
1

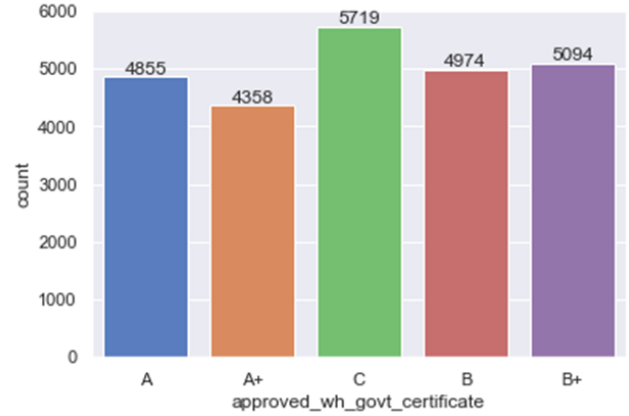

22957
2043
count
Location_type
Urban
Rural


10169
10020
4811
count
WH_capacity_size
Small
Large
Mid


10278
7931
6362
429
count
zone
West
North
South
East


8339
4587
2963
2881
2054
4176
count
WH_regional_zone
Zone 6
Zone 5
Zone 2
Zone 3
Zone 1
Zone 4


13578
11422
count
wh_owner_type
Rented
Company Owned


5719
4855
4358
4974
5094
count
approved_wh_govt_certificate
A
A+
C
B
B+

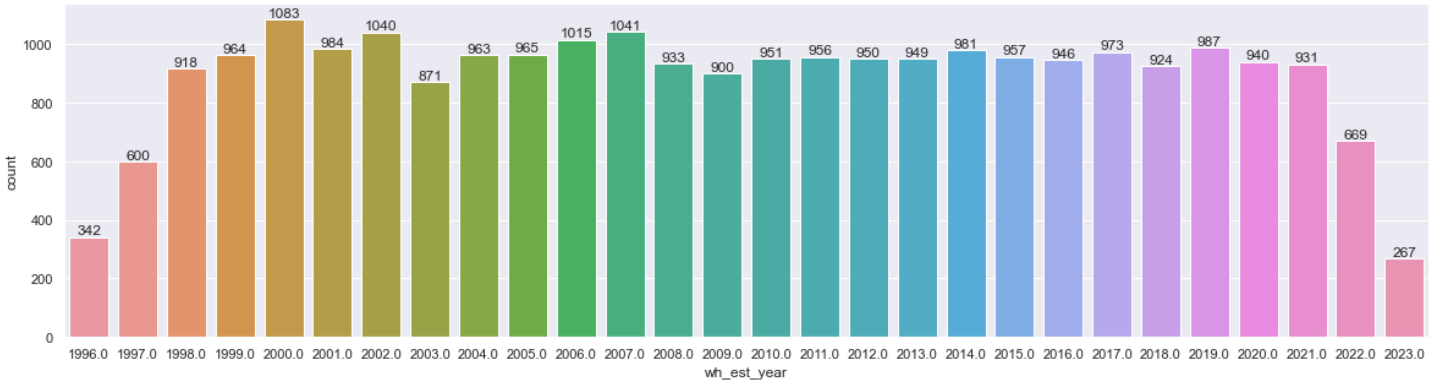


Figure 6 Count Plots - Categorical variables

From the above count plots it can be seen that

- Most of the ware houses had 2 competitors in market, while 1 ware house (WH_106813) did not have any competitor and 1 warehouse (WH_101568) had 12 competitors.
- Most of the ware houses required 3 refills in the last 3 months
- Most of the ware houses did not face any transport related accidents in the last 1 year
- Most of the ware houses had 2 break downs in the last 3 months
- Most of the ware houses does not have temperature regulating machine indicator
- Most of the ware houses have electric backup like generator.
- Most of the ware houses have electric backup like generator.
- Most of the ware houses are not flood proof and most of the ware houses are not located in flood impacted area.
- Most of the ware houses are in rural areas.
- Most of the ware houses are large and mid-sized.
- Most of the ware houses are in the northern region and are in Zone 6.
- More than half of the ware houses are company owned
- Most of the ware houses are established in the year 2000.
- Most of the ware houses have received the C certification form government followed by B+, B, A & A+.

Bi-Variate & Multi-Variate Analysis

HEAT MAP – CORRELATION PLOT

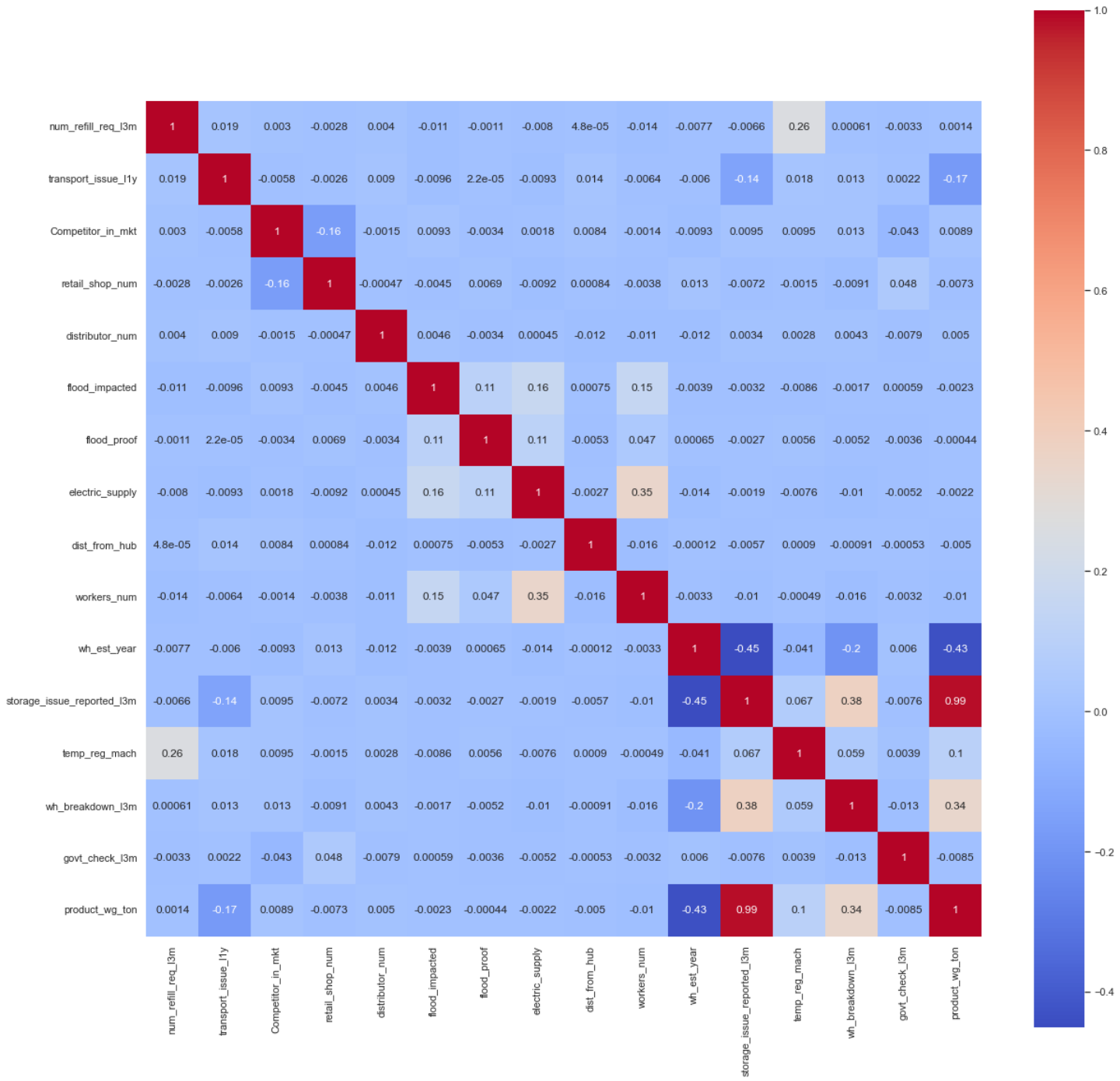


Figure 7 Correlation Plot

It can be seen that some of the variables are correlated with each other. Hence, we have the problem of multi-collinearity. This need to be treated appropriately, or it will affect the accuracy of the models that are used for prediction.

NUMBER OF REFILL IN LAST 3 MONTHS – REGION WISE, ZONE WISE & LOCATION TYPE WISE

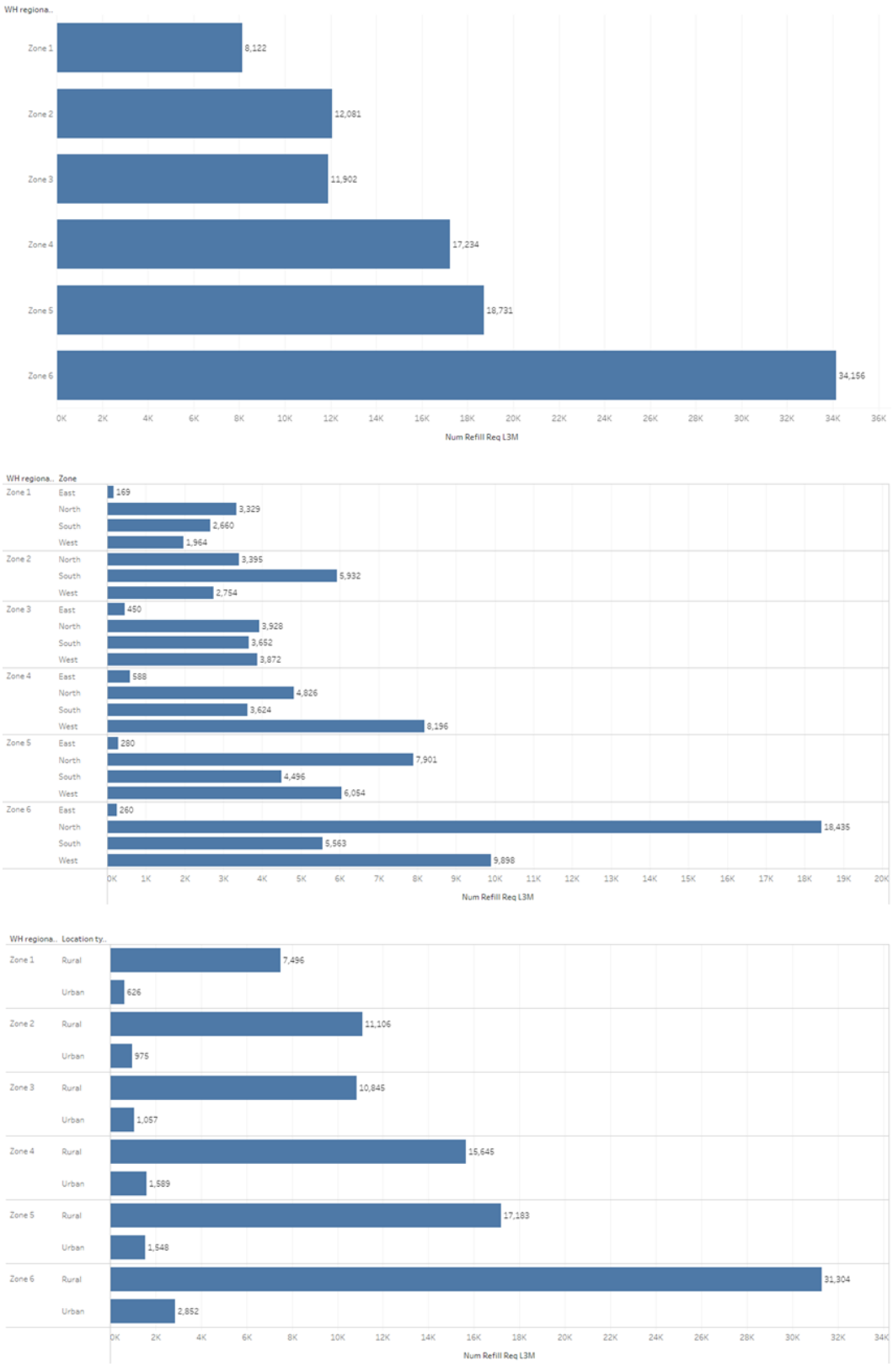


Figure 8 Number of Refill in last 3 months – Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of refills in the last 3 months.

NUMBER OF TRANSPORT ISSUES IN LAST 1 YEAR – REGION WISE, ZONE WISE & LOCATION TYPE WISE

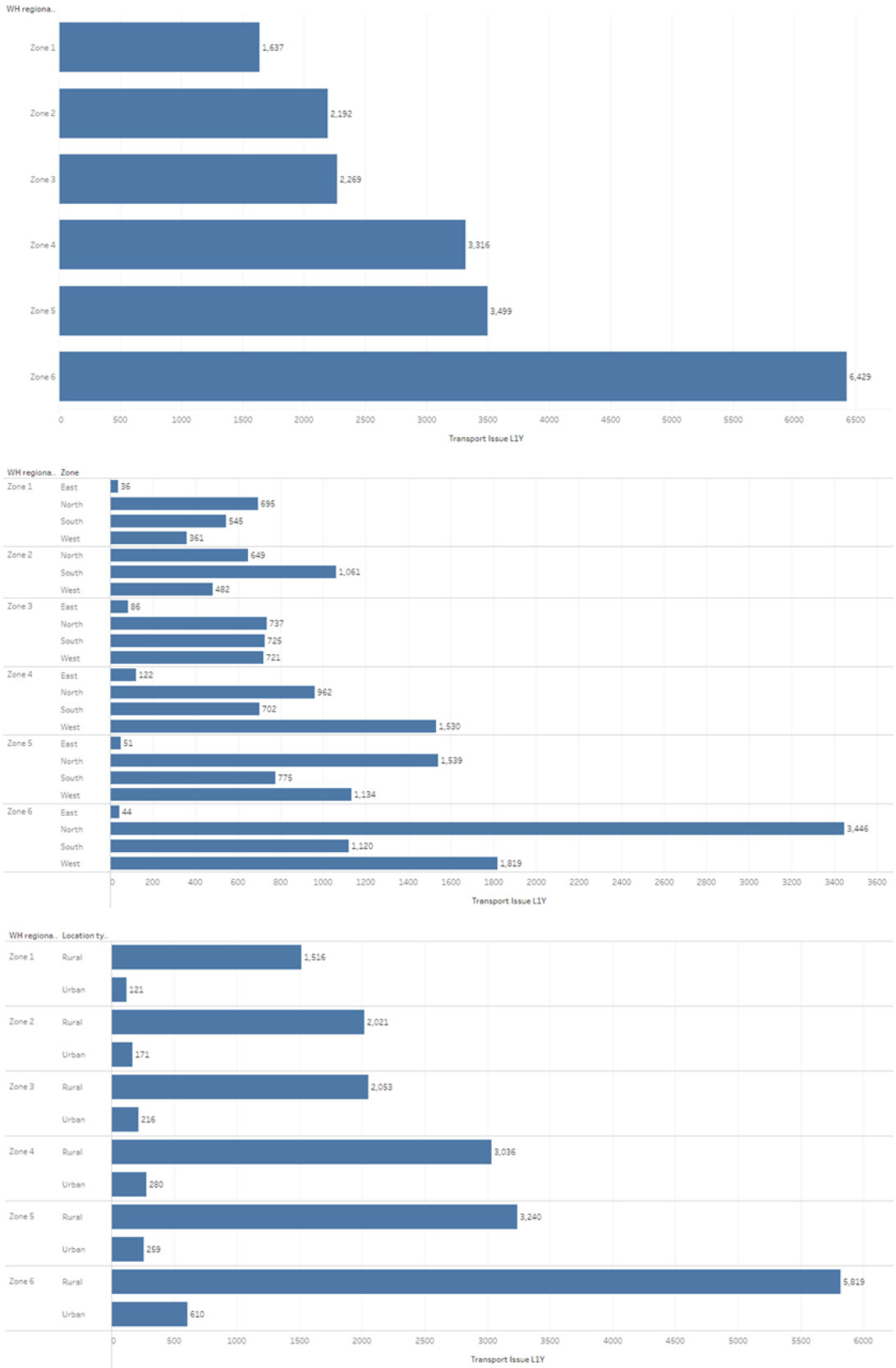


Figure 9 Number of Transport issues in last 1 year - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of transport issues in the last 1 year.

NUMBER OF COMPETITORS IN MARKET – REGION WISE, ZONE WISE & LOCATION TYPE WISE

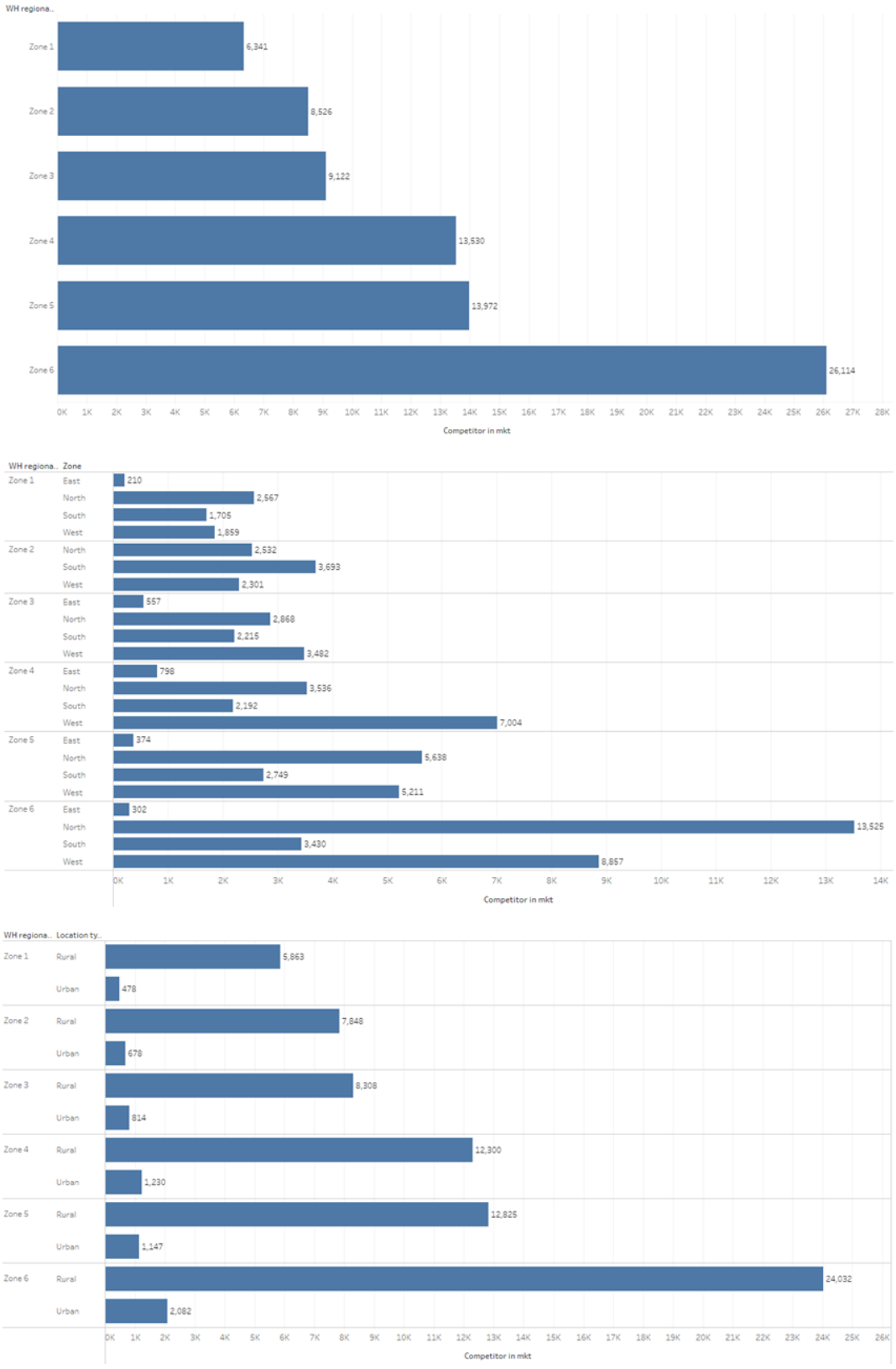


Figure 10 Number of competitors in market - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of competitors in market.

NUMBER OF RETAIL SHOPS – REGION WISE, ZONE WISE & LOCATION TYPE WISE

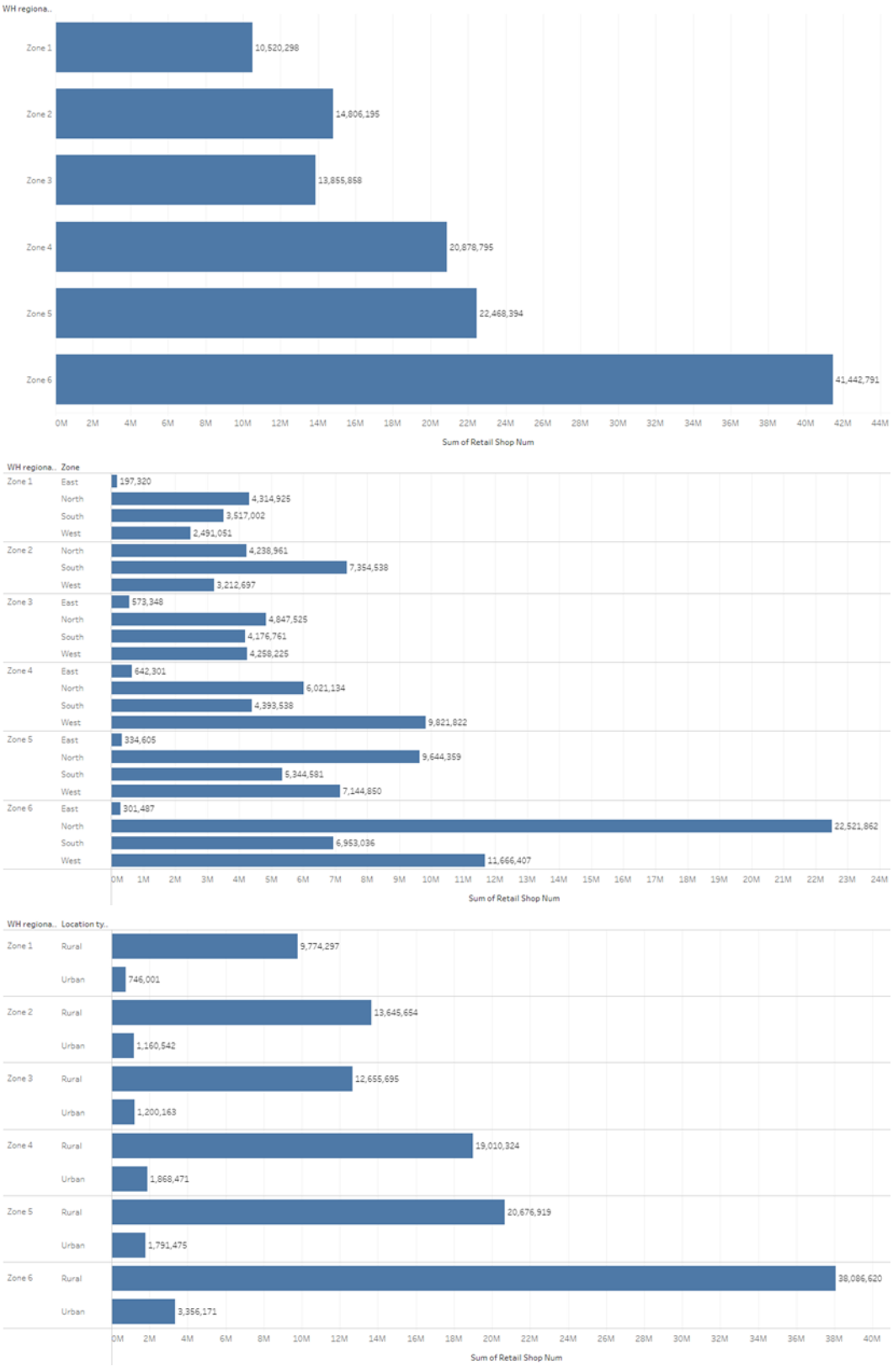


Figure 11 Number of Retail shops - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of Retail Shops

NUMBER OF DISTRIBUTORS – REGION WISE, ZONE WISE & LOCATION TYPE WISE

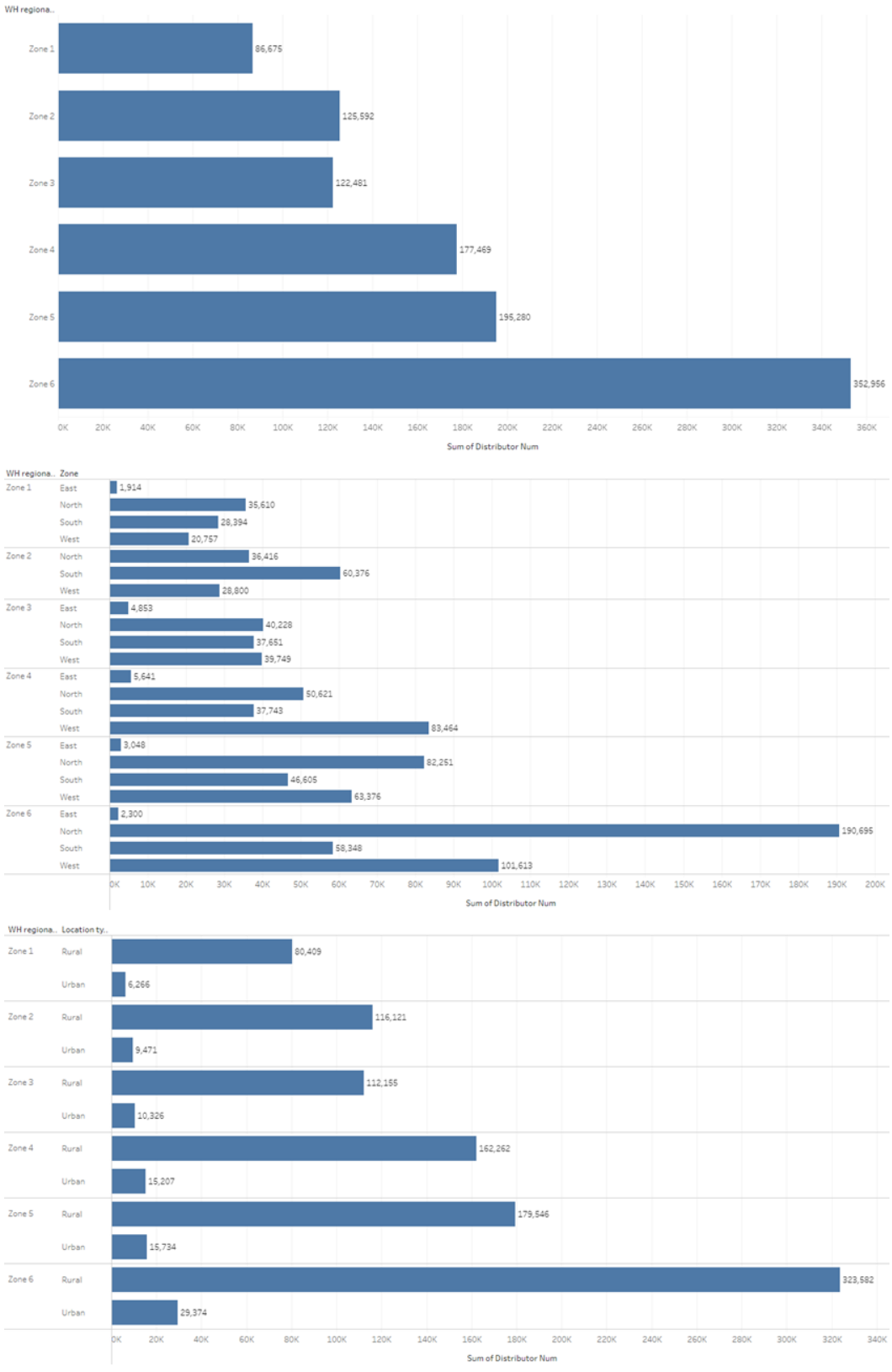


Figure 12 Number of Distributors - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of Distributors

NUMBER OF WAREHOUSE IN FLOOD IMPACTED AREAS – REGION WISE, ZONE WISE & LOCATION TYPE WISE

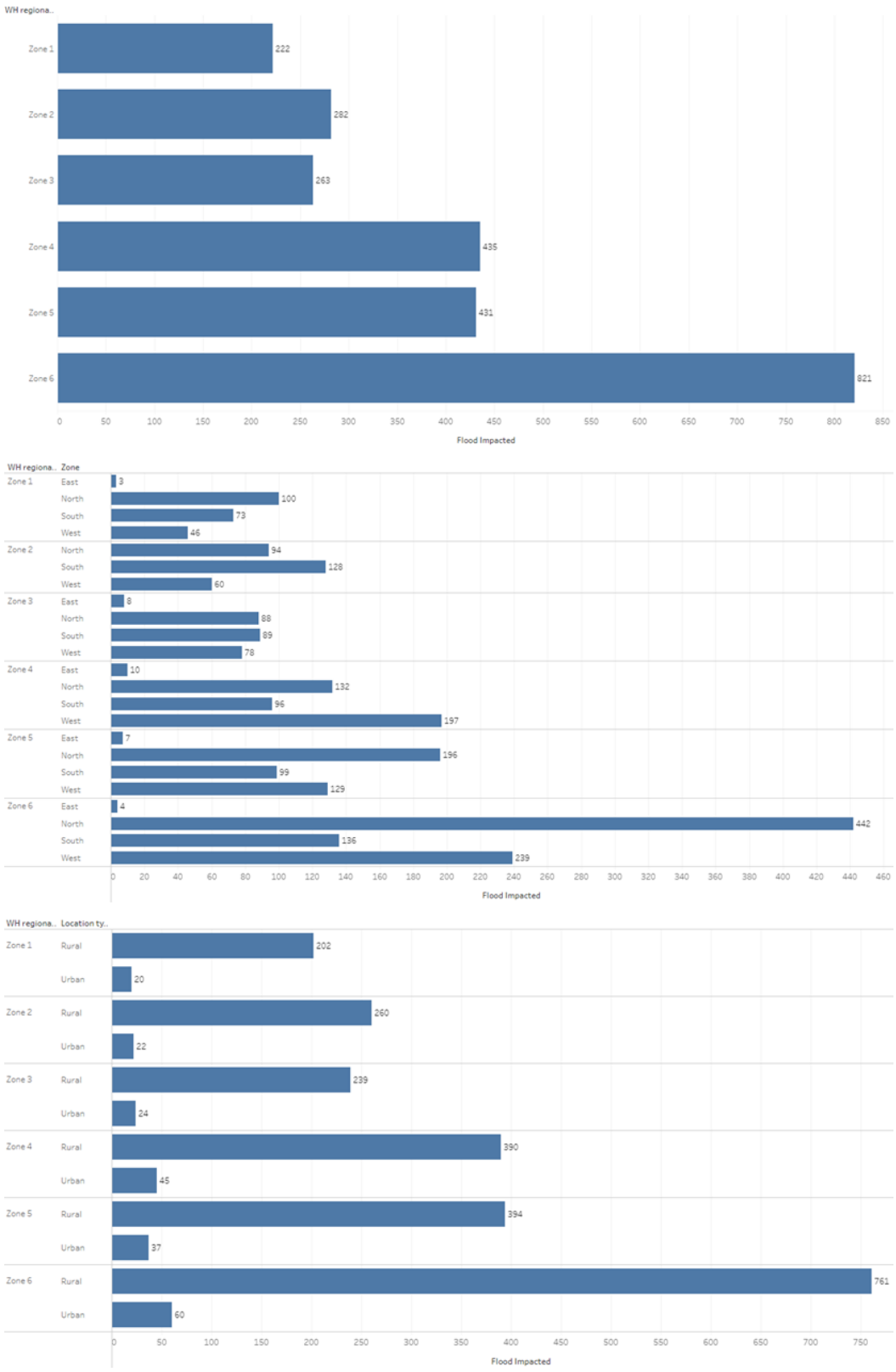


Figure 13 Number of Warehouse in Flood Impacted areas - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of ware house in flood impacted areas.

NUMBER OF WAREHOUSE IN FLOOD PROOF AREAS – REGION WISE, ZONE WISE & LOCATION TYPE WISE

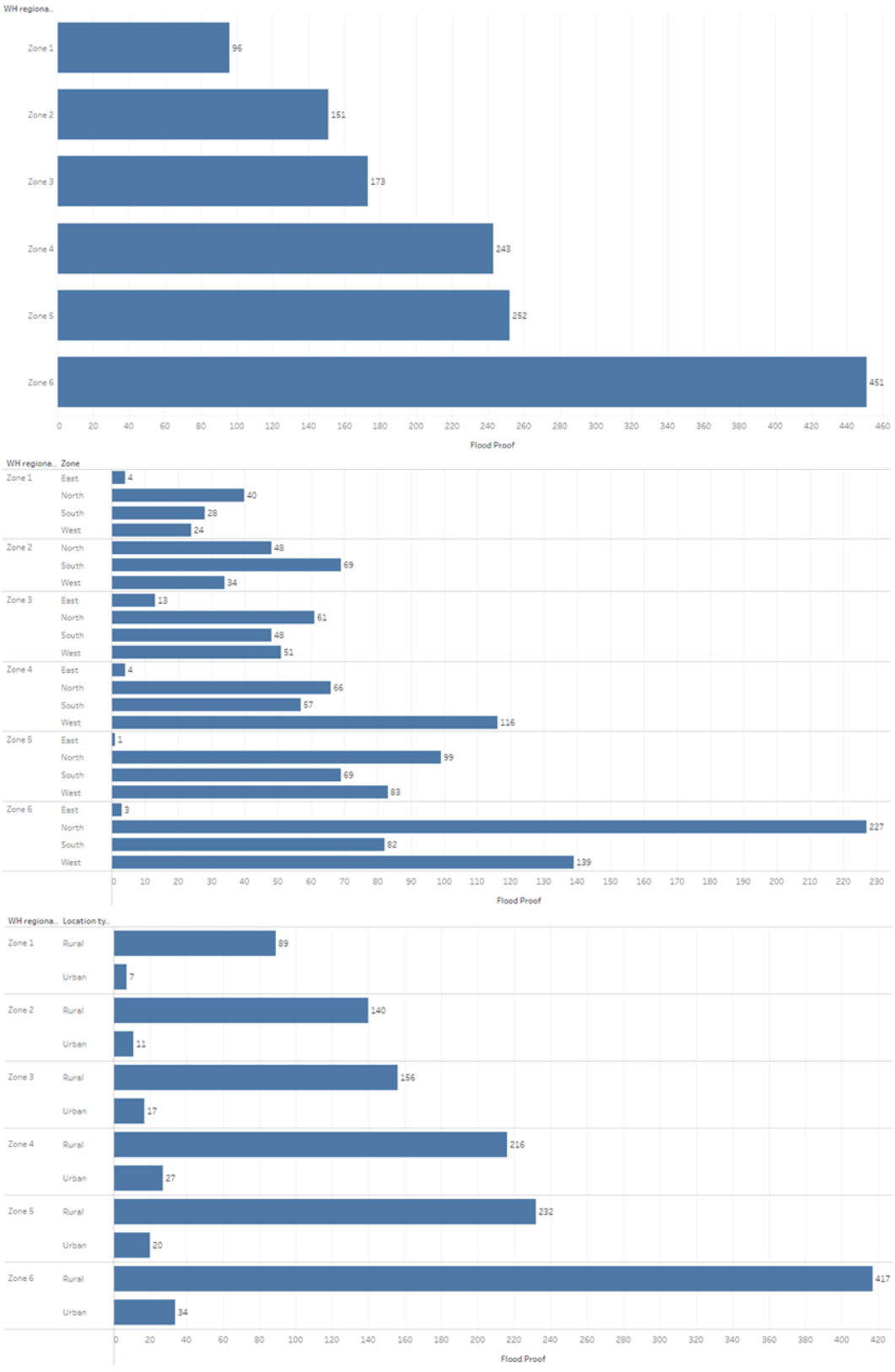


Figure 14 Number of Warehouse in Flood Proof areas - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of ware house in flood proof areas.

NUMBER OF WAREHOUSE WITH ELECTRICAL BACKUP – REGION WISE, ZONE WISE & LOCATION TYPE WISE

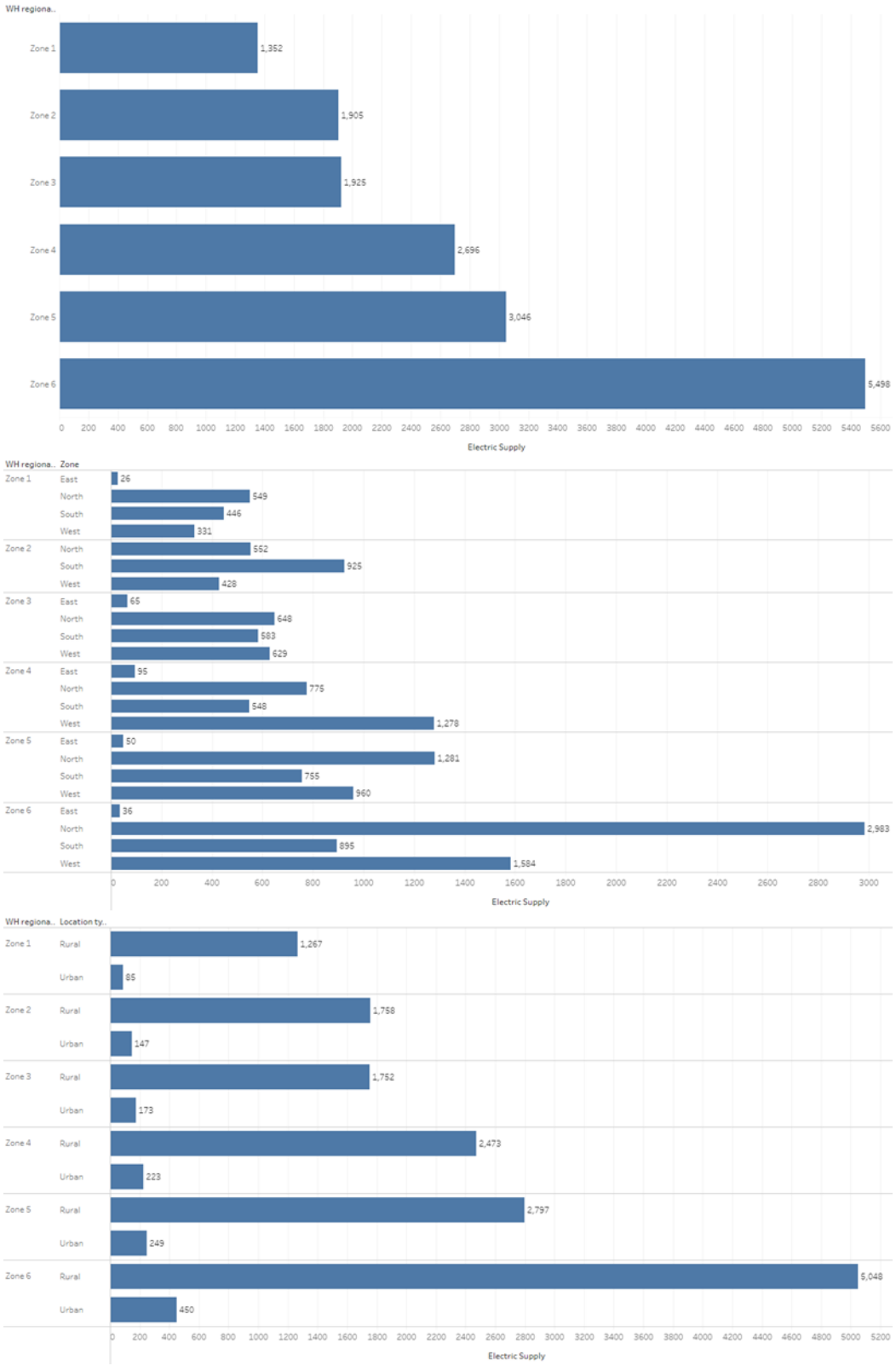


Figure 15 Number of Warehouse with Electrical Backup - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of ware house with electrical back up.

NUMBER OF WORKERS WORKING – REGION WISE, ZONE WISE & LOCATION TYPE WISE

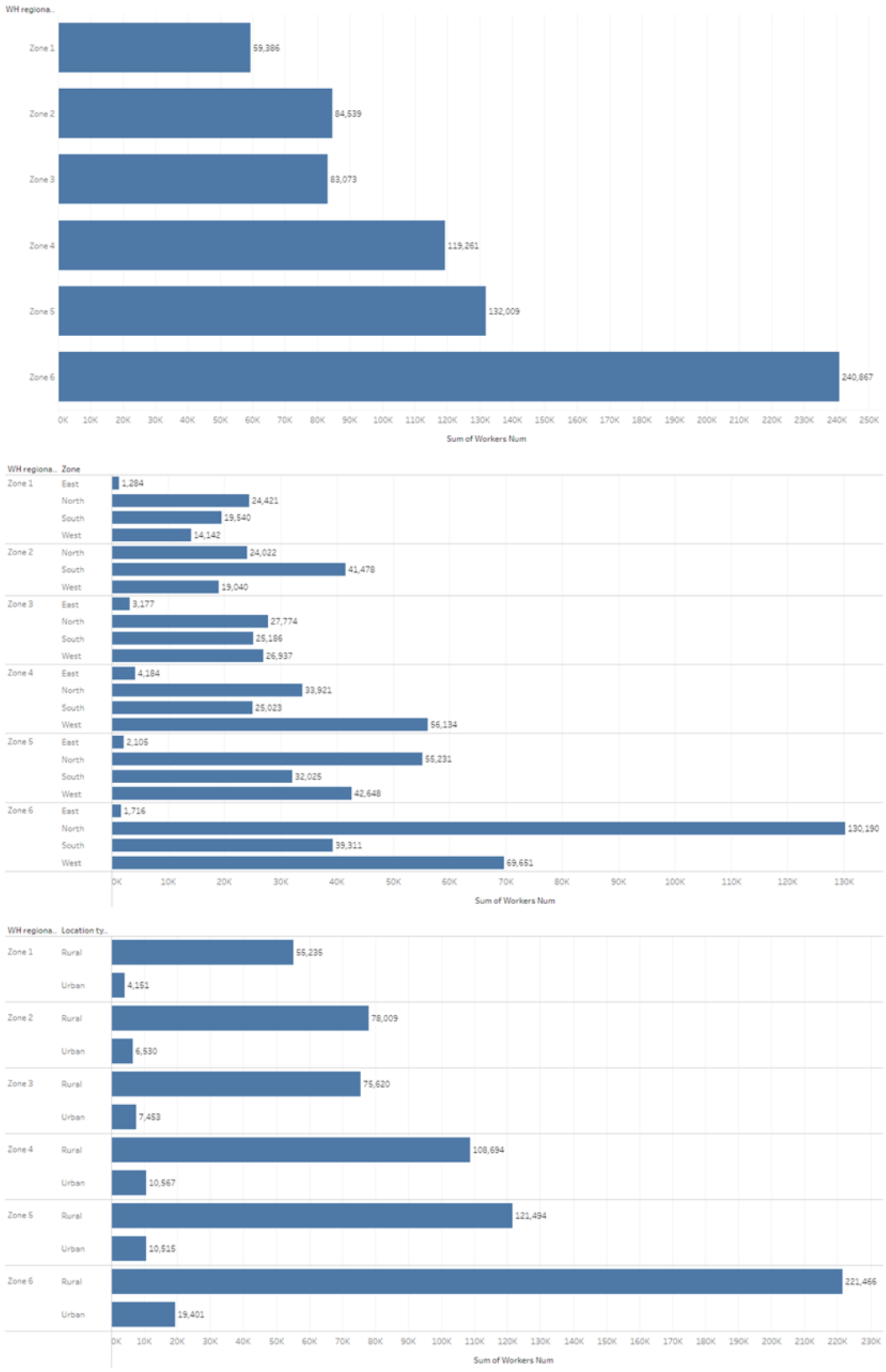


Figure 16 Number of Workers working - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of workers working.

NUMBER OF STORAGE ISSUES REPORTED IN LAST 3 MOS – REGION WISE, ZONE WISE & LOCATION TYPE WISE

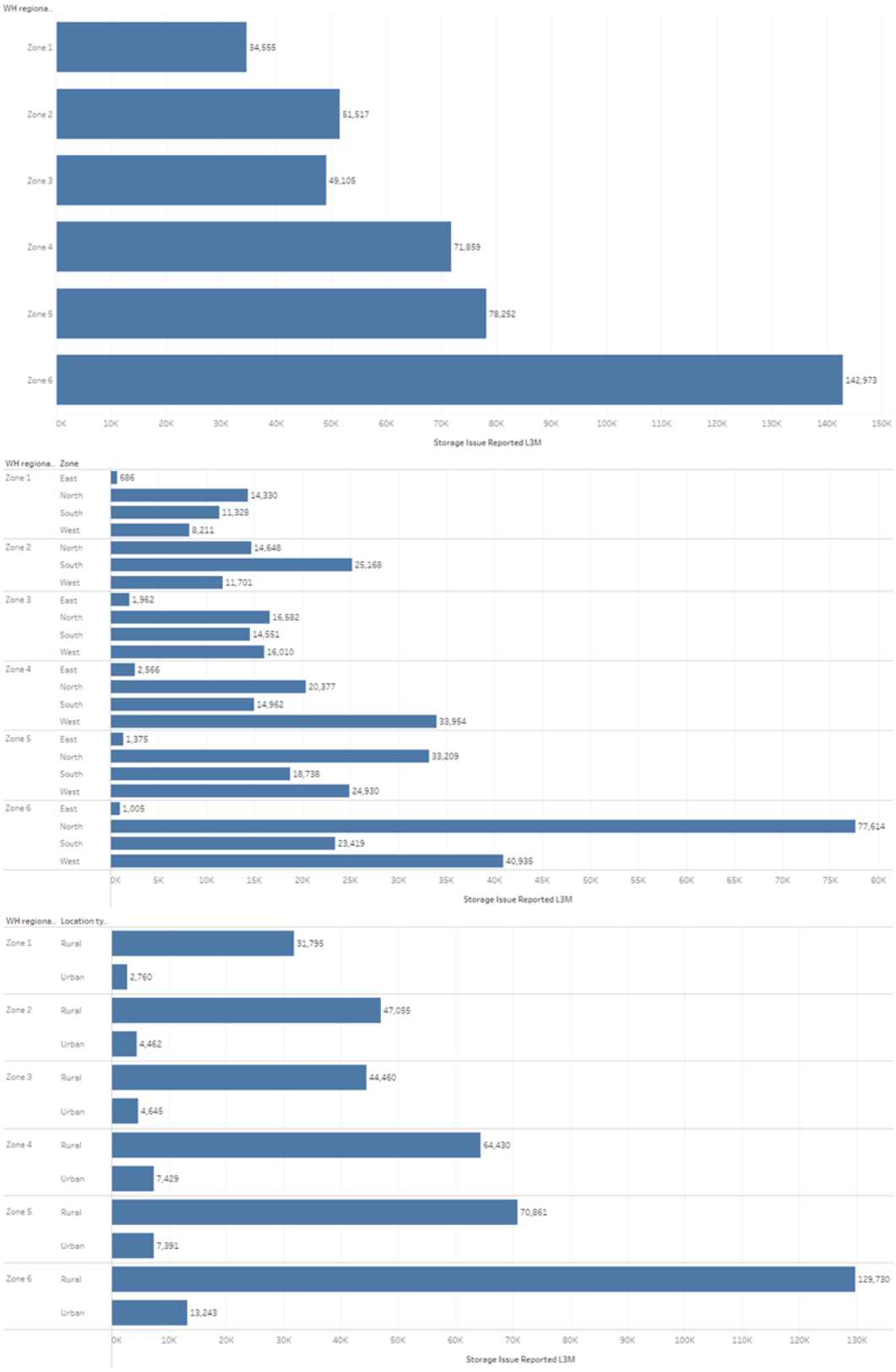


Figure 17 Number of storage issues reported in Last 3 Months - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of ware houses where storage issues reported in Last 3 Months.

NUMBER OF WAREHOUSES CATEGORIZED WITH CERTIFICATION-REGION, ZONE WISE & LOCATION TYPE WISE

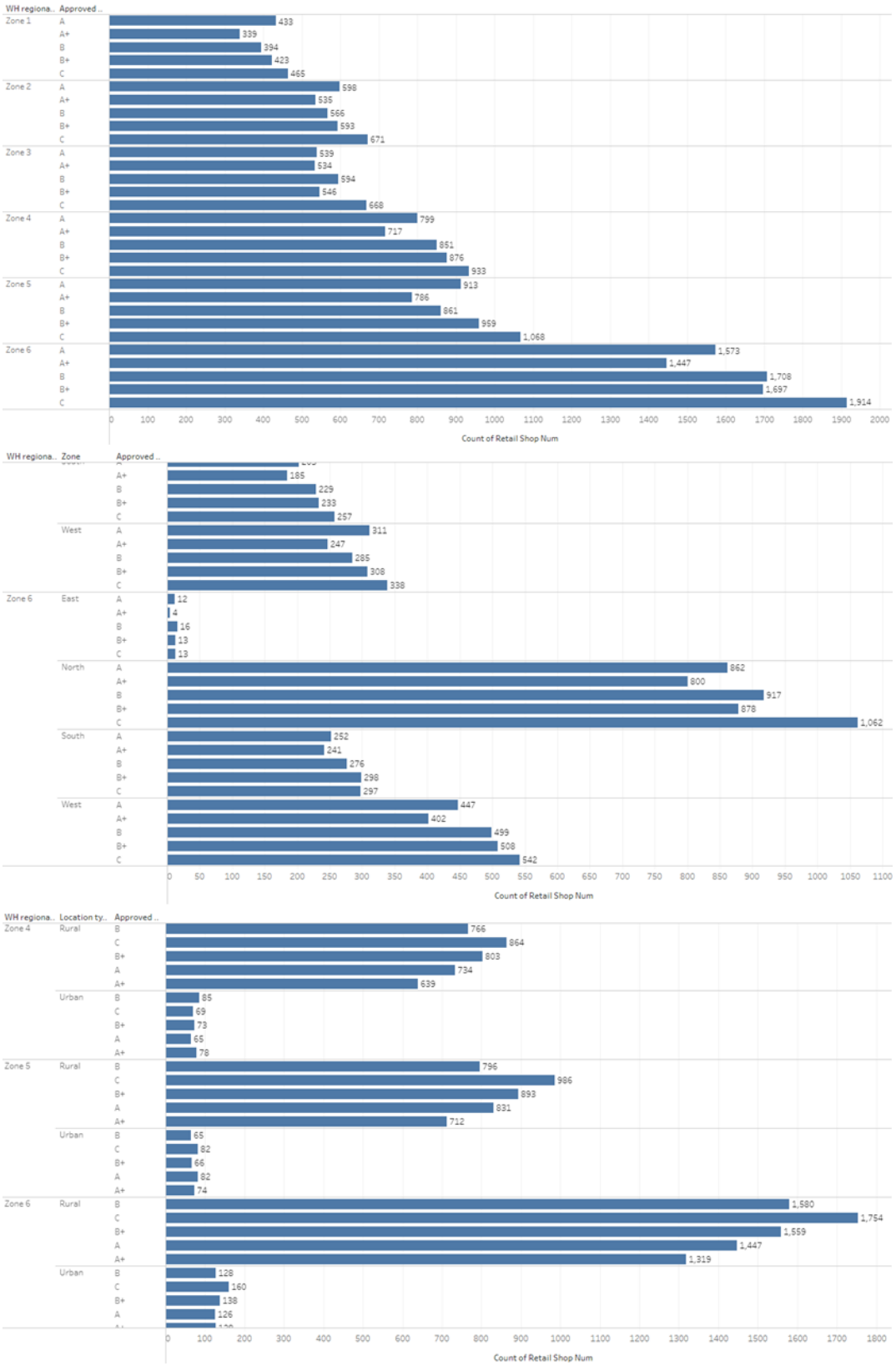


Figure 18 Number of Warehouses categorized with certification - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of ware houses with 'C' certified warehouses.

NUMBER OF BREAK DOWN IN LAST 3 MOS – REGION WISE, ZONE WISE & LOCATION TYPE WISE

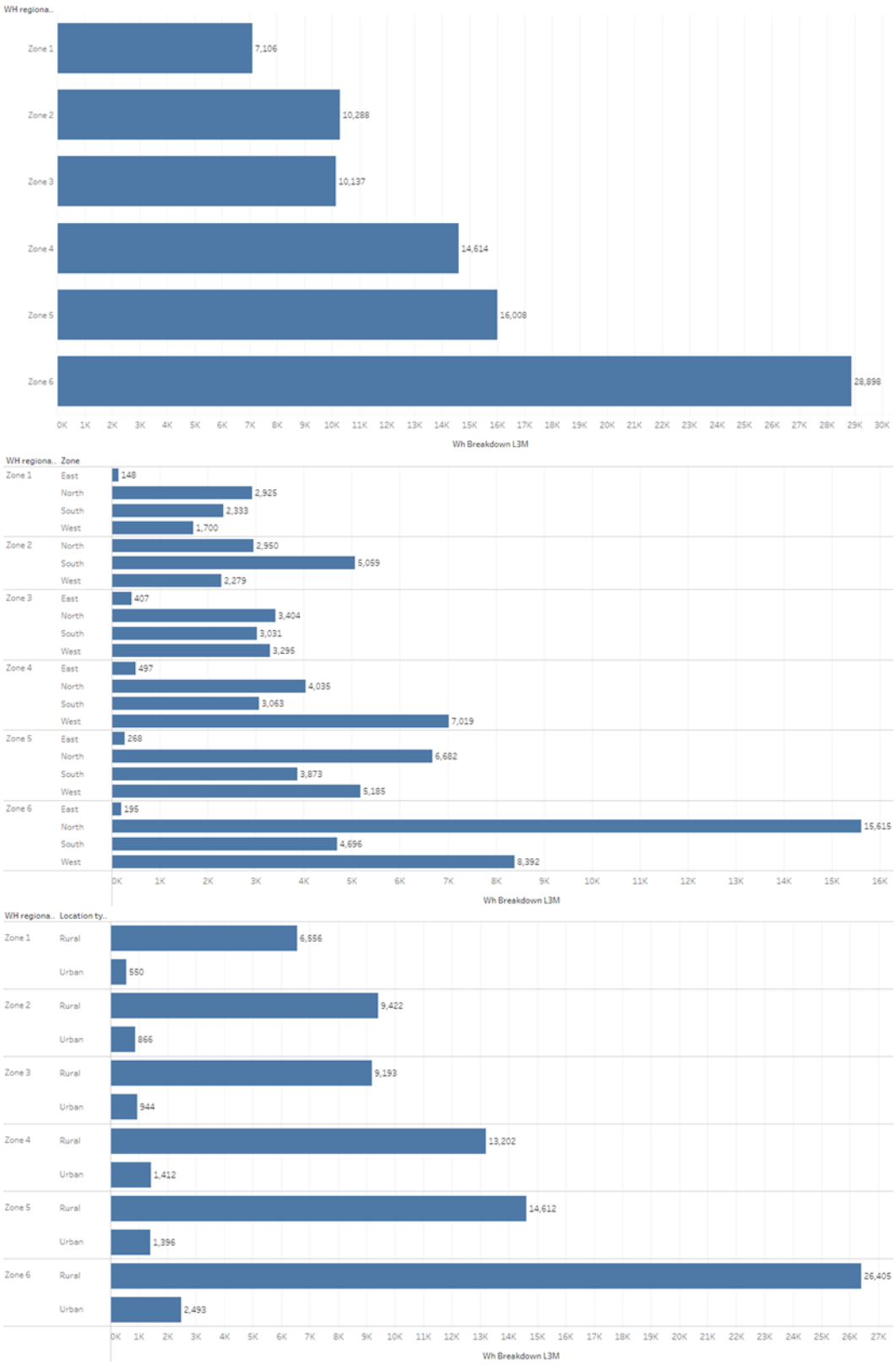


Figure 19 Number of Break down in Last 3 Months - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of ware houses break downs in Last 3 Months.

NUMBER OF GOVERNMENT CHECKS IN LAST 3 MOS – REGION WISE, ZONE WISE & LOCATION TYPE WISE

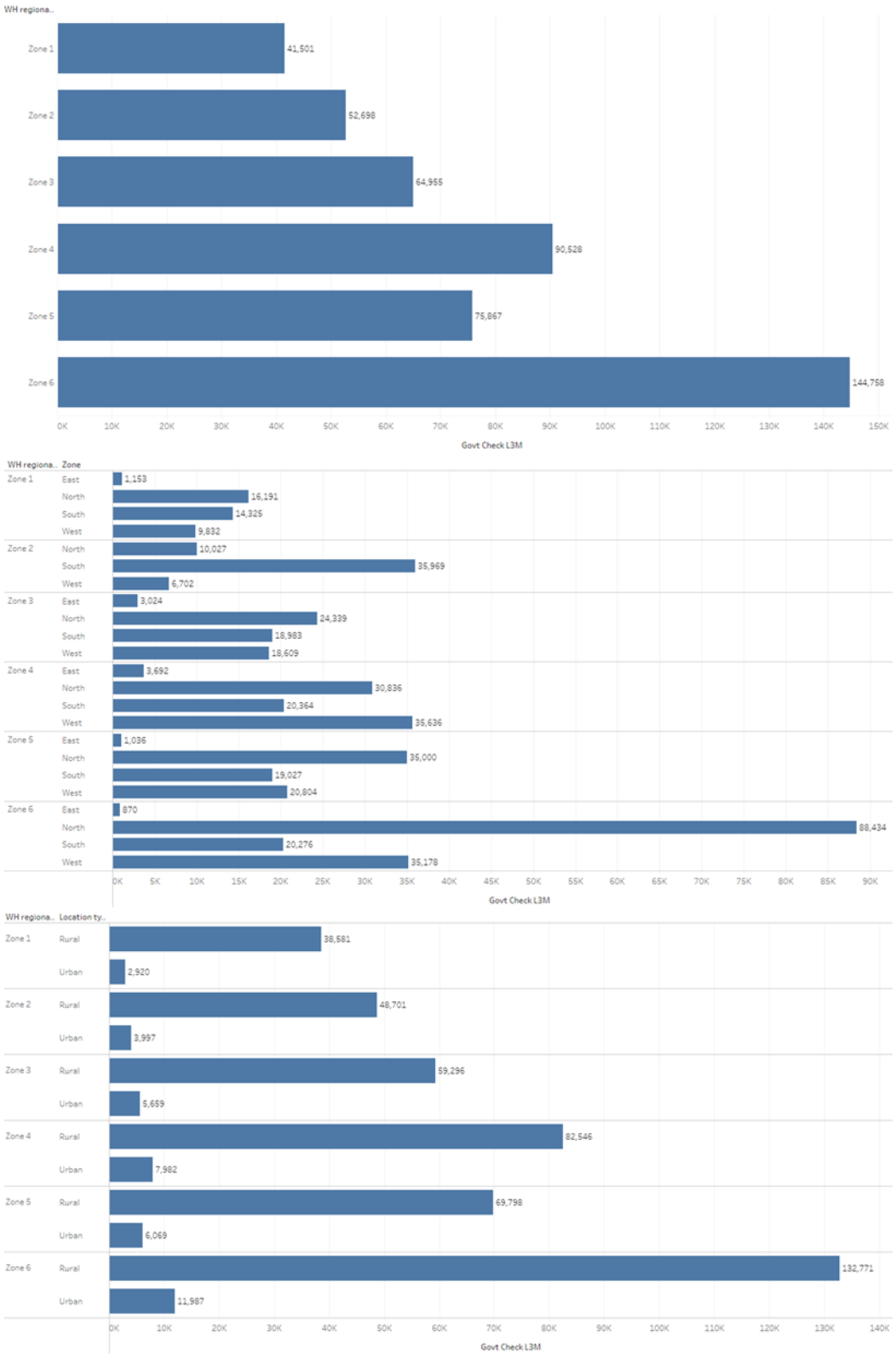


Figure 20 Number of Government checks in Last 3 Months - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 have the highest number of government checks in Last 3 Months.

WAREHOUSES CATEGORISED BY TOTAL WEIGHT– REGION, ZONE WISE & LOCATION TYPE WISE

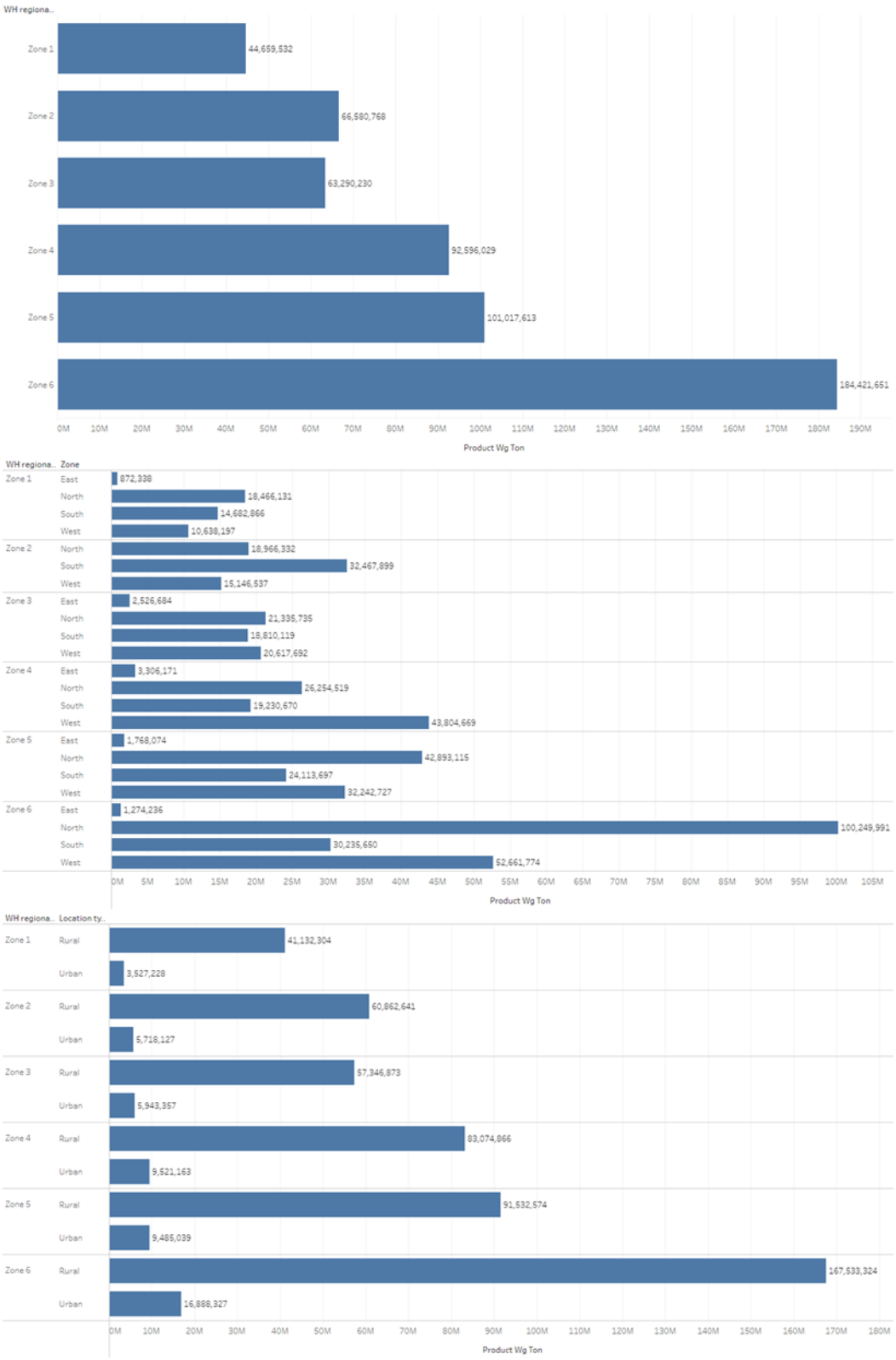


Figure 21 Warehouses categorized by total weight - Region wise, Zone wise & Location type wise

From the above plot, we can see that Northern region & rural location type of Zone 6 warehouses have the highest total product weight.

WAREHOUSES CATEGORISED ON AGE

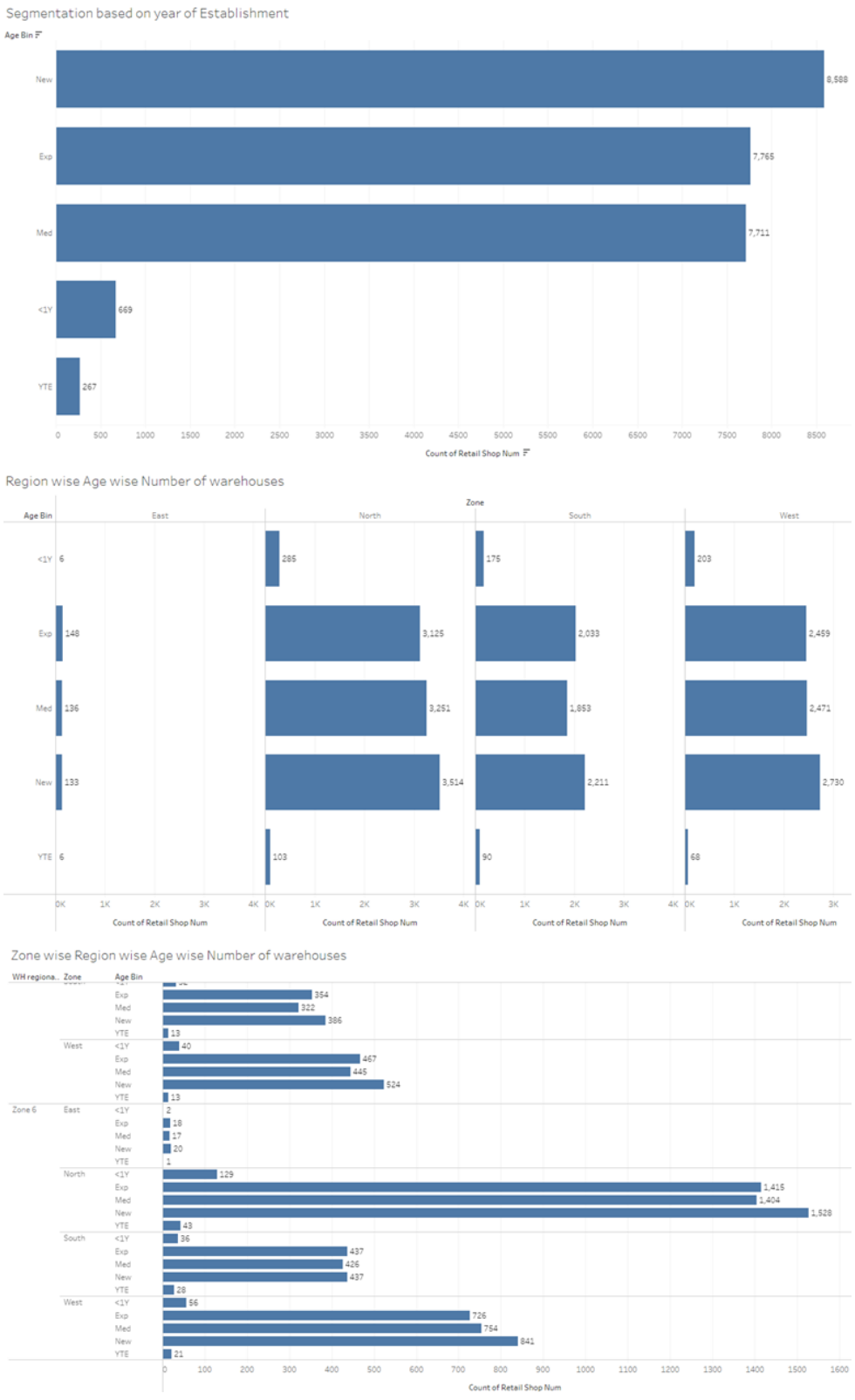


Figure 22 Number of warehouses against each bin - Over all, Zone wise & Region wise

From the above plot, we can see that most of the Ware houses are in the New age bracket (1 to 9 years) and are mostly concentrated in the Northern region of Zone 6.

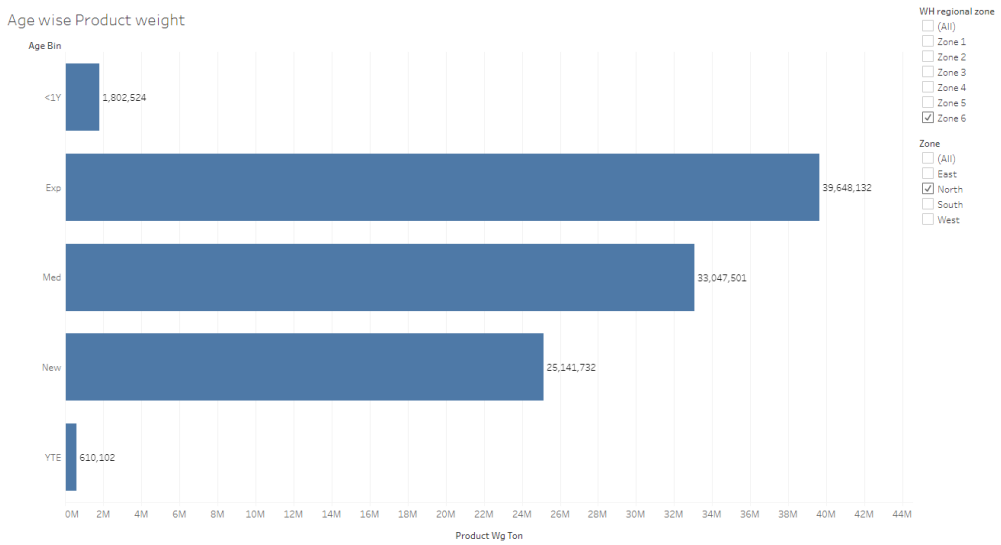


Figure 23 Age Bin Wise Product Weight

From the above plot, we can see that in spite of more number of ware houses in the new category of Zone 6, northern region, it is Expert category ware houses that has the capacity to hold more number of weight and in turn indicates that it has developed a huge customer base that more weight is stored followed by Med & New.

WAREHOUSES CATEGORISED ON AGE

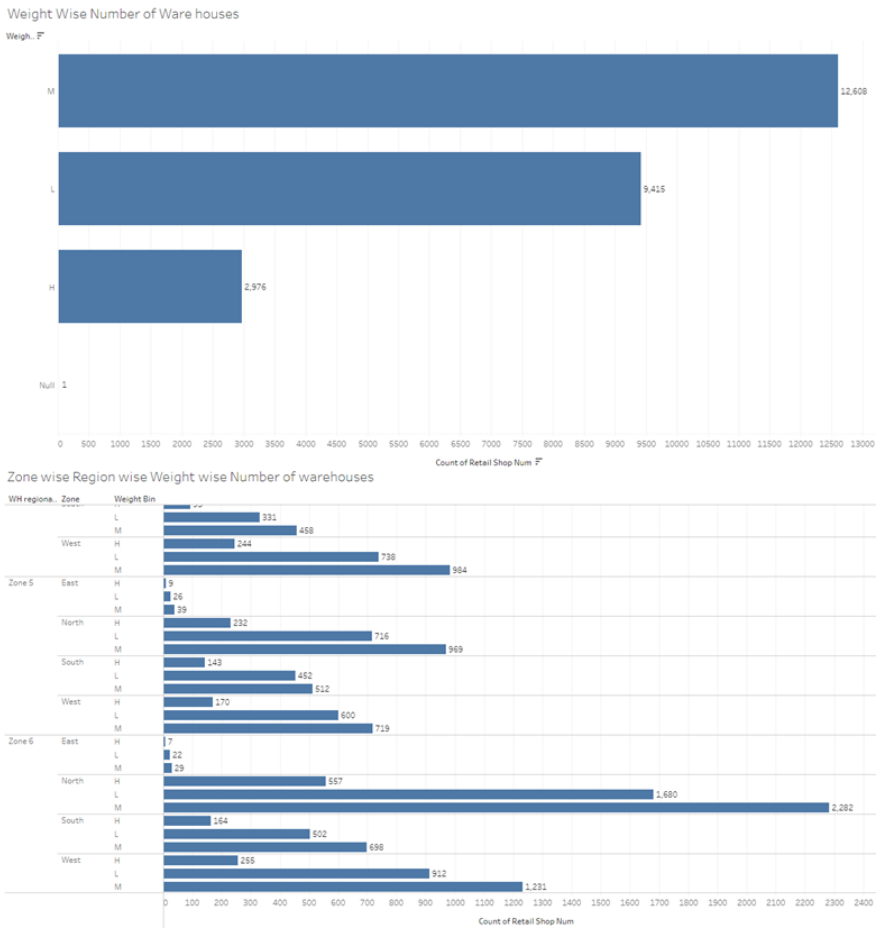


Figure 24 Number of Ware houses - Zone wise, Region wise, and weight wise

From the above plot, we can see that, most of the most of the ware houses carry medium category weight. On further analyzing it zone wise & region wise, we can see that Northern region of zone 6 has the most number of ware houses with Medium category weight followed by Low category weight.

Encoding – Variable Transformation (2)

Before going with finding the most significant variables, we need to transform the data in some of the variables to enable the model to find the significant variables. In line with this, we need to convert the object type variables to numerical variables, so that the Variable inflation factor technique can be utilized to find the most significant variables. One hot encoding technique is used for the variables 'Location_type', 'wh_owner_type', 'WH_regional_zone', 'zone', 'WH_capacity_size. However,as 'approved_wh_govt_certificate' has hierarchical categories, we encode them as in the below table.

Certification	Ordinal
A+	1
A	2
B+	3
B	4
C	5

Table 6 Encoding

Removal of non-significant variables

Firstly, we know that Ware house ID & Manager ID are merely for representation purpose and doesn't have any role in Analysis or model building. Hence are dropped directly from the dataset.

As already discussed in the Variable Transformation section, there are a large number of variables and also multi collinearity as seen in the correlation plot. Hence, both these points need to be addressed. Variance Inflation Factor technique is used to address this. It measures how much the variance of an independent variable is influenced, or inflated, by its interaction/correlation with the other independent variables. It is a measure of how much the variable is contributing to the standard error. Here the dependent variable is kept out as the VIF is used to check the redundancy in the Independent variable. Typically, a value of 5 for VIF is allowed. If it is more than 5 for an independent variable, then it is more or less compensated by other variables. A for loop and

variance inflation factor from stats models library is used to calculate the vif, that lists the variables and their VIFs in the form of a data frame.

The procedure is repeated by removing variables one by one, as vif changes with removal of each variable. 'WH_regional_zone_Zone2', 'WH_regional_zone_Zone3', 'WH_regional_zone_Zone4', 'WH_capacity_size_Mid', 'wh_est_year', 'retail_shop_num', 'zone_North', 'zone_West', 'workers_num', 'Competitor_in_mkt', 'distributor_num', 'dist_from_hub', 'wh_breakdown_l3m' are dropped as their VIF were greater than 5.

	variables	VIF
18	WH_regional_zone_Zone 2	inf
19	WH_regional_zone_Zone 3	inf
20	WH_regional_zone_Zone 4	inf
26	WH_capacity_size_Mid	inf
10	wh_est_year	209.607485
3	retail_shop_num	28.410337
23	zone_North	27.459556
25	zone_West	20.627975
9	workers_num	19.409699
24	zone_South	17.843253
2	Competitor_in_mkt	10.752832
4	distributor_num	7.976431
8	dist_from_hub	7.806882
15	govt_check_l3m	7.693713
22	WH_regional_zone_Zone 6	7.686700
21	WH_regional_zone_Zone 5	6.473106
13	approved_wh_govt_certificate	6.420291
14	wh_breakdown_l3m	6.159909
11	storage_issue_reported_l3m	5.451688
0	num_refill_req_l3m	3.735117
7	electric_supply	3.494898
27	WH_capacity_size_Small	3.156064
17	wh_owner_type_Rented	1.982570
12	temp_reg_mach	1.670258
1	transport_issue_l1y	1.456439
5	flood_impacted	1.166891
16	Location_type_Urban	1.097803
6	flood_proof	1.081342

Figure 25 VIF Values - All Variables

As can be seen from the above figure 22, we can see that govt_check_l3m, WH_regional_zone_Zone6, WH_regional_zone_Zone5, approved_wh_govt_certificate, wh_breakdown_l3m, storage_issue_reported_l3m have VIF higher than 5, but as the variables were dropped one by one, we can see that the above variables VIF were less than 5 and not dropped. Refer figure 23.

	variables	VIF
8	govt_check_l3m	4.899737
7	approved_wh_govt_certificate	4.807978
5	storage_issue_reported_l3m	3.800284
0	num_refill_req_l3m	3.485383
4	electric_supply	2.877205
10	wh_owner_type_Rented	1.848336
12	WH_regional_zone_Zone 6	1.735769
6	temp_reg_mach	1.629826
14	WH_capacity_size_Small	1.417972
1	transport_issue_l1y	1.402753
13	zone_South	1.397069
11	WH_regional_zone_Zone 5	1.383800
2	flood_impacted	1.153500
9	Location_type_Urban	1.096589
3	flood_proof	1.080488

Figure 26 Significant Variables with VIF

BUSINESS INSIGHTS & RECOMMENDATIONS FROM EDA

We can ignore the data imbalance as it is a supervised Regression problem.

As we can see from the above EDA & the analysis of clusters on Zone wise, region wise & Location type wise, warehouses in the Northern region & rural location type of Zone 6 have been prominent in refill and total weight of products. Hence it becomes utmost necessity to properly analyze the market requirement in terms of what type of Noodles is preferred w.r.t raw substance, taste preferences, etc…, plan warehouse inventory and introduction of products accordingly in these regions/ Zones/ Localities. Based on this plan, targeted marketing

can be done in these regions & zones to attract customers and boost the sales. Strategic pricing of these product helps increase the bottom line.

Further, we need to work towards migrating from C Certifications to A+ certifications as this might lead to loss of certain customers and also impact the Brand value.

MODEL BUILDING, VALIDATION, TUNING & INTERPRETATION

As we have already discussed, this is a supervised regression problem, where in we need to find a real number that is the value of the target variable. This is unlike binary and can be any value between -/ + infinity. However, in this problem we are in need to build a model that predicts a real number, the weight of products in ton in an inventory at a given point of time. Hence, the values might range from 0 to infinity.

There are many models that are used for predicting a regression variable. Some of the models that are most commonly used are as below.

1. Linear Regression
2. Decision Tree Regressor
3. Ensemble models - Random Forest Regressor, Bagging, Adaboost & Gradient boost Regressor.

While some ensemble models themselves have in built model tuning techniques/ algorithms, we will be discussing on the Grid search cross validation for model tuning of Decision trees & Random forest models.

Train Test Split

Before moving on with any type of model building, it is essential to split the data in to training and test datasets, so that we will be able to first train the model and then perform our model validations in the test data. To split the data in to training & testing data, we need to first create separate data frames for the independent & dependent variables. Once the data set is separated, the split of training & testing data can be achieved through train_test_split function in sklearn.model_selection library. This function requires the independent & dependent variables, percentage data to be considered for testing and the random state. Usually, to obtain good results, we need to train the model with a larger number of data points, so that it would be able to capture

most of the characteristics of the dataset. Hence, we select the train: test samples in the ratio 70:30. The input for the testing percentage to be considered will be 0.3. The random state considered here is 1. Random state is to ensure that the set of codes when run in a different jupyter notebook will give the same values for the defined random state.

Linear Regression

MODEL BUILDING APPROACH

Linear Regression or any regression in that case deal only with numbers. The term "regression" refers to predicting a real number. The term "linear" in the name “linear regression” refers to the fact that the method models data with linear combination of the explanatory variables. A linear combination is an expression where one or more variables are scaled by a constant factor and added together. In the case of simplest linear regression with a single explanatory variable, the linear combination used in linear regression can be expressed as:

Dependent variable value = (weight ∗independent variable) + constant

Equation 1 Simple Linear Regression

It is the straight line in the scatter plot of the variables. For a linear model to be built, there must be correlation amongst the dependent variables, which we saw in the Correlation & pair plot. Based on the correlation a scatter plot can be obtained and there can be infinite straight lines that can fit in the scatter plot as a linear model. The best fit line can be found out using the Gradient Descent method. Gradient descent methods use partial derivatives on the parameters (slope and intercept) to minimize sum of squared errors. The line that represents the model may not touch all the points in the scatter plot. The vertical distance between a point and the line is the error in prediction of the model. The line which gives least sum of squared errors across all the data points put together is considered as the best fit line.

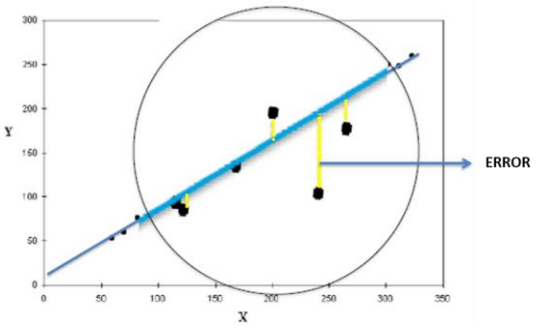


Figure 27 Best Fit Line & Error - Linear Regression

The best fit line will always go through that point in the features space where the Xbar and ybar meet.

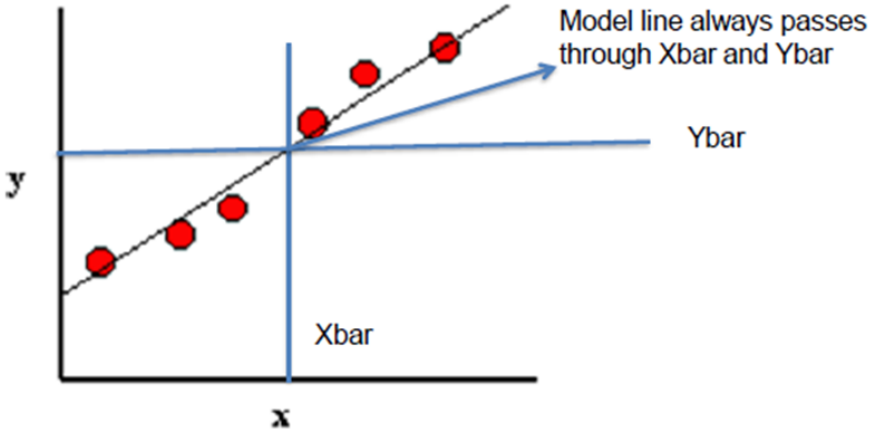


Figure 28 Best Fit Line, Xbar & ybar

ERRORS & METRICS IN LINEAR REGRESSION MODEL

The figure below shows the different type of errors & the table provides description of the same.

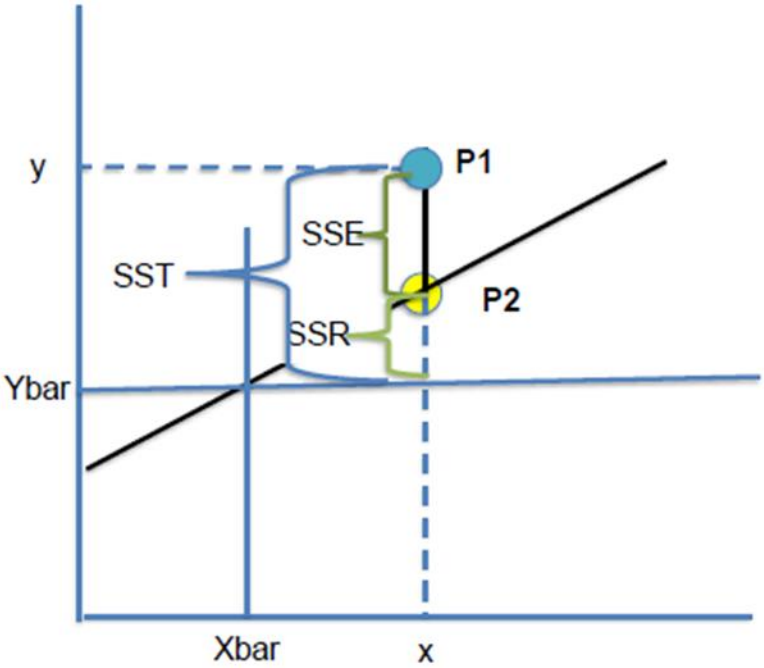


Figure 29 Errors in Linear Regression Model

P1	Original y data point for given x
P2	Estimated y value for given x
Ybar	Average of all Y values in data set
SST	Sum of Square error Total (SST), Variance of P1 from Ybar (Y Ybar)^2
SSR	Regression error (p2 ybar) 2 (portion SST captured by regression model)
SSE	Residual error (p1 p2)^2

Table 7 Errors in Linear Regression Model

That model is the best fit where every data point lies on the line. i.e. SSE = 0 for all data points. Hence SSR should be equal to SST i.e. SSR/SST should be 1. Poor fit will mean large SSE. SSR/SST will be close to 0. SSR / SST is called as R^2 (r square) or coefficient of determination. R^2 is always between 0 and 1 and is a measure of utility of the regression model and determines the fitness of a linear model. The closer the points get to the line, the R^2 (coefficient of determinant) tends to 1, the better the model is. Hence, R^2 is one of the metrics to determine the model fitness. Similarly, the root of mean squared error also is one of the metrics, the lower the value, better the model is.

STRUCTURE OF A LINEAR REGRESSION MODEL

$$Y = m1X1 + m2X2 + \ldots\ldots + mnXn + C + e$$

Equation 2 Linear Regression structure

Where,

Y = Dependent / target / predicted variable

Xi = Independent/ predictor variable

mi = coefficients for the independent / predictor variable

C = constant / intercept / bias

e = residual error / unexplained variance / difference between actual and prediction

Linear Regression models can be built using two ways

- SKLEARN
- SCIPY STATS

LINEAR REGRESSION MODEL USING SKLEARN

LinearRegression model library from sklearn.linear_model in python is used to build the Linear Regression model. The train data is fit in to the model. By fitting the data, the model gets trained using the training set. The independent attributes have different units and scales of measurements. Hence, It is always the best to scale all the dimensions using z scores or some other method to address the problem of different scales.

We know that the coefficients of the predictor variables (weights), determine how much influence it has in predicting the target variable. The coefficients of different variables are as below.

S. No	Variable Name	Iteration 1 - W/o scaling	Iteration 2 - Scaled Data
1	num_refill_req_l3m	5.1136	0.0011
2	transport_issue_l1y	-331.3939	-0.0342
3	flood_impacted	13.1833	0.0003
4	flood_proof	68.7404	0.0013
5	electric_supply	15.6510	0.0006
6	storage_issue_reported_l3m	1236.4648	0.9760
7	temp_reg_mach	699.5693	0.0277
8	approved_wh_govt_certificate	-264.8158	-0.0320
9	govt_check_l3m	0.9160	0.0007
10	Location_type_Urban	-130.3932	-0.0031
11	wh_owner_type_Rented	2.7573	0.0001
12	WH_regional_zone_Zone_5	-30.1892	-0.0010
13	WH_regional_zone_Zone_6	-16.1058	-0.0007
14	zone_South	-32.9799	-0.0012
15	WH_capacity_size_Small	28.8261	0.0010
16	Intercept	1768.5190	-1.9680
17	R^2 - Train Data	0.9766	0.9766
18	R^2 - Test Data	0.9777	0.9777
19	RMSE - Train Data	1780.4493	0.1528
20	RMSE - Test Data	1716.9875	0.1490

Table 8 Coefficients, R^2, RMSE - Linear Regression using SKLearn

It can be seen that, there are no major difference due to scaling of dataset. However, the intercept, which is of no meaning in the linear model is found to be reduced. On seeing the coefficients, it can be interpreted as storage issue reported in last 3 months has the highest weightage in prediction of the product

weight and is directly proportional .i.e. increase in this parameter indicates greater the weight. Similarly, transport issue in last 1 year is inversely proportional to the product weight in warehouse. The model is also performing well in both train and test data which is evident from the R^2 and RMSE values. This indicates that the model is a Good fit. Also the R2 value tends to be closer to 1. However, we need to further evaluate with different models to get an understanding on the RMSE values.

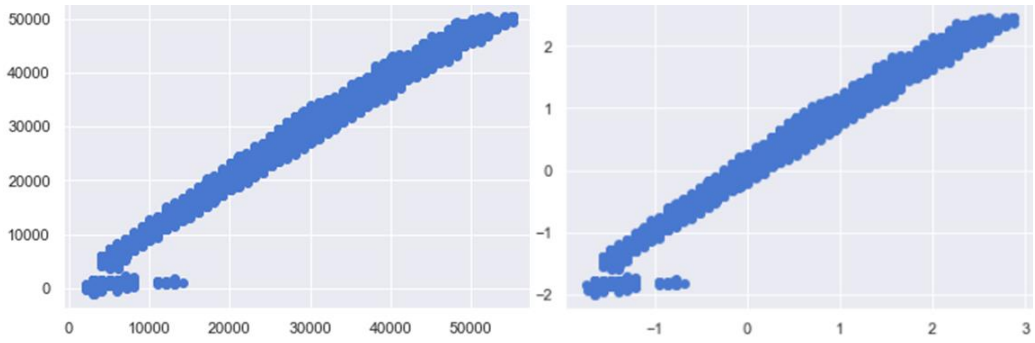


Figure 30 Linear Regression plot – Un scaled & Scaled Data SK learn

LINEAR REGRESSION MODEL USING STATSMODEL

Hypothesis Testing

To establish the reliability of the coefficients, we need hypothesis testing. The Null hypothesis (H0) claims there is no relation between predictor & predicted variables. That means the coefficient is 0. At 95% confidence level, if the p value is < .05, we reject the H0 i.e., probability of finding these coefficients in sample is very low. If p value is >=.05, we do not have sufficient evidence in the data to reject the H0 and hence we do not reject H0. We believe H0 is likely to be true in the universe.

Model Building

R^2 is not a reliable metric as it always increases with addition of more attributes even if the attributes have no influence on the predicted variable. Instead, we use adjusted R^2 which removes the statistical chance that improves R^2. SKlearn does not provide a facility for adjusted R^2. So we use statsmodel, a library that gives similar results. This library expects the X and Y to be given in one single data frame.

The predictor and predicted attributes are again merged as a single data frame & split in to train (70%) and test (30%) data. Then the training Xs & Ys and Testing Xs & Ys are merged separately to meet the statsmodel requirement.

The scipy stats uses the Ordinary Least Squares method to build the model. The formula is built and fit in the linear model. Formula is a below:

expr= 'product_wg_ton ~ num_refill_req_l3m + transport_issue_l1y + flood_impacted + flood_proof + electric_supply + storage_issue_reported_l3m + temp_reg_mach + approved_wh_govt_certificate + govt_check_l3m + Location_type_Urban + wh_owner_type_Rented + WH_regional_zone_Zone_5 + WH_regional_zone_Zone_6 + zone_South + WH_capacity_size_Small'

Equation 3 OLS - Formula - Iteration 1

The linear model summary is as below.

OLS Regression Results

Dep. Variable:	product_wg_ton	R-squared:	0.977
Model:	OLS	Adj. R-squared:	0.977
Method:	Least Squares	F-statistic:	4.870e+04
Date:	Sun, 27 Nov 2022	Prob (F-statistic):	0.00
Time:	05:14:43	Log-Likelihood:	-1.5581e+05
No. Observations:	17500	AIC:	3.117e+05
Df Residuals:	17484	BIC:	3.118e+05
Df Model:	15		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1768.5190	70.025	25.256	0.000	1631.263	1905.775
num_refill_req_l3m	5.1136	5.365	0.953	0.341	-5.402	15.629
transport_issue_l1y	-331.3939	11.343	-29.215	0.000	-353.628	-309.160
flood_impacted	13.1833	46.688	0.282	0.778	-78.331	104.697
flood_proof	68.7404	60.509	1.136	0.256	-49.863	187.344
electric_supply	15.6510	29.614	0.528	0.597	-42.395	73.697
storage_issue_reported_l3m	1236.4648	1.499	824.716	0.000	1233.526	1239.403
temp_reg_mach	699.5693	31.520	22.194	0.000	637.786	761.352
approved_wh_govt_certificate	-264.8158	10.006	-26.466	0.000	-284.428	-245.203
govt_check_l3m	0.9160	1.674	0.547	0.584	-2.364	4.196
Location_type_Urban	-130.3932	49.484	-2.635	0.008	-227.386	-33.401
wh_owner_type_Rented	2.7573	27.884	0.099	0.921	-51.898	57.413
WH_regional_zone_Zone_5	-30.1892	38.499	-0.784	0.433	-105.652	45.274
WH_regional_zone_Zone_6	-16.1058	31.560	-0.510	0.610	-77.966	45.755
zone_South	-32.9799	31.640	-1.042	0.297	-94.997	29.037
WH_capacity_size_Small	28.8261	37.570	0.767	0.443	-44.814	102.467

Omnibus:	6775.261	Durbin-Watson:	2.002
Prob(Omnibus):	0.000	Jarque-Bera (JB):	47579.688
Skew:	1.696	Prob(JB):	0.00
Kurtosis:	10.331	Cond. No.	150.

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Figure 31 OLS Summary - Iteration 1

In the above summary, we can see that, the intercept & coefficient values remain the same as in that of the model built using SK learn. Hence, the same interpretations are valid. However, there are some variables where the p value is > 0.05. Hence, we will reiterate the entire model building activity after removing the non-

significant variables. This is a part of tuning the linear regression model that is built using the stats model library.

Formula and summary as below.

expr_temp= 'product_wg_ton ~ transport_issue_l1y + storage_issue_reported_l3m + temp_reg_mach + approved_wh_govt_certificate + Location_type_Urban'

Equation 4 OLS - Formula - Iteration 2

OLS Regression Results

Dep. Variable:	product_wg_ton	R-squared:	0.977
Model:	OLS	Adj. R-squared:	0.977
Method:	Least Squares	F-statistic:	1.461e+05
Date:	Sun, 27 Nov 2022	Prob (F-statistic):	0.00
Time:	05:14:45	Log-Likelihood:	-1.5581e+05
No. Observations:	17500	AIC:	3.116e+05
Df Residuals:	17494	BIC:	3.117e+05
Df Model:	5		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1806.2924	48.499	37.244	0.000	1711.230	1901.355
transport_issue_l1y	-331.4113	11.339	-29.227	0.000	-353.637	-309.185
storage_issue_reported_l3m	1236.4157	1.498	825.180	0.000	1233.479	1239.353
temp_reg_mach	707.3640	30.357	23.301	0.000	647.861	766.867
approved_wh_govt_certificate	-264.3267	9.984	-26.474	0.000	-283.897	-244.756
Location_type_Urban	-129.4820	49.454	-2.618	0.009	-226.418	-32.547

Omnibus:	6791.563	Durbin-Watson:	2.002
Prob(Omnibus):	0.000	Jarque-Bera (JB):	47859.115
Skew:	1.700	Prob(JB):	0.00
Kurtosis:	10.354	Cond. No.	74.4

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Figure 32 OLS Summary - Iteration 2

Summary of Iterations 1 & 2 for Linear regression using stats model as in the below table.

S.No	Variable Name	Iteration 1	Iteration 2
1	num_refill_req_l3m	5.1136	NA
2	transport_issue_l1y	-331.3939	-331.4113
3	flood_impacted	13.1833	NA
4	flood_proof	68.7404	NA
5	electric_supply	15.6510	NA
6	storage_issue_reported_l3m	1236.4648	1236.4157
7	temp_reg_mach	699.5693	707.3640
8	approved_wh_govt_certificate	-264.8158	-264.3267
9	govt_check_l3m	0.9160	NA
10	Location_type_Urban	-130.3932	-129.4820
11	wh_owner_type_Rented	2.7573	NA
12	WH_regional_zone_Zone_5	-30.1892	NA
13	WH_regional_zone_Zone_6	-16.1058	NA
14	zone_South	-32.9799	NA
15	WH_capacity_size_Small	28.8261	NA
16	Intercept	1768.5190	1806.2924
17	R^2	0.977	0.977
18	Adj. R^2	0.977	0.977

19	RMSE - Train Data	1780.4493	1780.7200
20	RMSE - Test Data	1716.9875	1716.7562

Table 9 Coefficients, R^2, adj. R^2, RMSE - Linear Regression using stats model

We can infer that the predictive power of the variables are changing when the non-significant variables are dropped. However, the same interpretations are still valid. We can also see that R2 and adj. R2 values remain the same. Hence, these models have performance similar to that of the SKlearn. RMSE values of Iteration 1 are same as that of the SK learn. However, RMSE values of iteration 2 are also almost the same, with negligible differences. The linear model plot also remains the same and is as below.

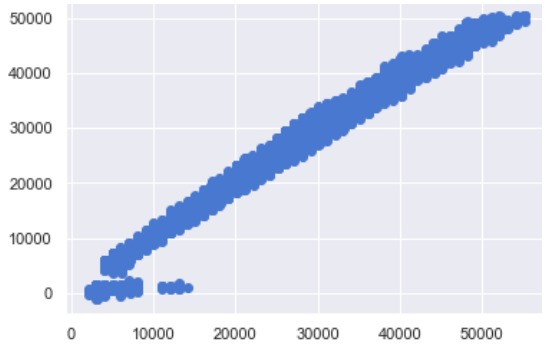


Figure 33 Linear Regression plot - Stats model

Decision Tree & Ensemble Models

Decision Tree is a decision-making tool that uses a flowchart-like tree structure or is a model of decisions and all of their possible results, including outcomes, input costs, and utility. Decision tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables.

The branches/edges represent the result of the node and the nodes have either:

1. Conditions [Decision Nodes]
2. Result [End Nodes]

The branches/edges represent the truth/falsity of the statement and take makes a decision based on that. In this project we will carry out Regression models in the Decision trees considering the nature of the problem.

METRICS IN DECISION TREE REGRESSION MODELS

The metrics in decision tree Regression models are same as that of the Linear Regression model i.e. the R^2 and RMSE values. R^2 should tend towards 1 while RMSE should be as low as possible for a model be fit.

CART MODEL - MODEL BUILDING APPROACH

CART – Classification And Regression Tree is a binary decision tree that can give both categorical & continuous output variable. The most commonly used splitting criteria for a Regression problem is mean squared which is equal to variance reduction as feature selection criterion and minimizes the L2 loss using the mean of each terminal node. The nodes are split based on the least mean squared error.

CART is a predictive algorithm used in Machine learning and it explains how the target variable's values can be predicted based on other matters. It is a decision tree where each fork is split into a predictor variable and each node has a prediction for the target variable at the end.

In the decision tree, nodes are split into sub-nodes on the basis of a threshold value of an attribute. The root node is taken as the training set and is split into two by considering the best attribute and threshold value. Further, the subsets are also split using the same logic. This continues till the last pure sub-set is found in the tree or the maximum number of leaves possible in that growing tree.

CART MODEL USING SKLEARN

In order to build a CART model, we first need to import the DecisionTreeRegressor from sklearn.tree. Post importing, we need to build the model using the squared error criterion and fit the training data (both dependent & independent variables). The model is initially built with the default values of max_depth, min_samples_leaf &min_samples_split. These are also called the hyper parameters. Maximum depth refers to the number of branches that the tree can be split along vertically, min sample leaf refers to the minimum number of records that a node must contain after splitting and the min sample split refers to the minimum number of records that a node must have so that it can be split. The default values are None, 2 &2 respectively, which indicate that the tree is fully grown. The feature importance can also be understood using the

feature_importance_property available in the CART model of SKlearn. It is a measure of how important each independent variable is in predicting the dependent variable.

The feature importance of the CART model with default values is as below.

	Imp
num_refill_req_l3m	0.001466
transport_issue_l1y	0.001255
flood_impacted	0.000206
flood_proof	0.000111
electric_supply	0.000413
storage_issue_reported_l3m	0.982813
temp_reg_mach	0.000867
approved_wh_govt_certificate	0.009550
govt_check_l3m	0.001712
Location_type_Urban	0.000141
wh_owner_type_Rented	0.000461
WH_regional_zone_Zone_5	0.000228
WH_regional_zone_Zone_6	0.000277
zone_South	0.000327
WH_capacity_size_Small	0.000171

Figure 34 Feature Importance - CART default values

It can be seen that storage issue reported l3m has the highest predictive power while all the other variables have an equal share on the prediction power when the tree is fully grown. This is in line with the interpretation in the Linear Regression. Y values are predicted for both train and test data and are evaluated. The model evaluation details as in the below table. Here, we can see that the model performs well on train data set, while the model is not up to the mark in the test dataset, which is evident from the large difference in the RMSE of train & test data sets. Hence, the data need to be tuned/ pruned.

Model	Train Data		Test Data	
	R2	RMSE	R2	RMSE
Decision Tree (DT) - Default parameters	0.9998	154.7	0.9878	1270.2599

Table 10 Metrics - CART - Default Hyper Parameters

However, re-building the model (regularizing) after initializing the hyper parameters with certain values (max_depth =10, min_samples_leaf=10, min_samples_split=30), certain improvement was noticed. The details of the feature importance & model evaluation as below. The Feature importance shows the same trend but with different values now. However, the RMSE values are closer than the ones noticed in the CART with default parameters. Yet, Model pruning/ tuning still need to be done.

	Imp
num_refill_req_l3m	0.000255
transport_issue_l1y	0.001049
flood_impacted	0.000015
flood_proof	0.000000
electric_supply	0.000022
storage_issue_reported_l3m	0.988384
temp_reg_mach	0.000788
approved_wh_govt_certificate	0.009256
govt_check_l3m	0.000148
Location_type_Urban	0.000005
wh_owner_type_Rented	0.000017
WH_regional_zone_Zone_5	0.000011
WH_regional_zone_Zone_6	0.000020
zone_South	0.000020
WH_capacity_size_Small	0.000010

Figure 35 Feature Importance - CART (max_depth =10, min_samples_leaf=10, min_samples_split=30)

Model	Train Data		Test Data	
	R2	RMSE	R2	RMSE
Reg. DT (max_depth =10, min_samples_leaf=10, min_samples_split=30)	0.9941	891.524	0.9935	926.8318

Table 11 CART (max_depth =10, min_samples_leaf=10, min_samples_split=30)

CART MODEL TUNING – GRID SEARCH CROSS VALIDATION

The optimum parameters can be obtained by the Grid search Cross validation function of model selection library in the SKlearn module. Grid Search cross-validation is a technique to select the best of the machine learning model, parameterized by a grid of hyper parameters. Cross validation works by splitting our dataset into random groups, holding one group out as the test, and training the model on the remaining groups. This process is repeated for each group being held as the test group, then the average of the models is used for the resulting model. One of the most common types of cross validation is k-fold cross validation, where 'k' is the number of folds within the dataset.

The hyper parameters of CART model are initialized in a grid (Grid values as in the below figure) and the Grid Search cross validation is run for the initialized hyper parameters in the DecisiontreeRegressor model.

We select a value of 5 for Cross validation. Greater the CV number, higher the time to execute the python codes.

```
param_grid = {
    'max_depth': [7, 9, 10, 12],
    'min_samples_leaf': [10, 15, 20, 25],
    'min_samples_split': [30, 45, 60, 75]
}
```

Figure 36 CART - Hyper parameters initialization - Grid search 1

Once the Grid search CV is run, we need to fit our training dataset in to the Grid Search CV model and the best parameters obtained are max_depth =10, min_samples_leaf=10, min_samples_split=45. Using these best hyper parameters, we shall predict the values for both training & testing dataset. However, considering the values obtained for min_samples_leaf to be at the right end, we can further rerun the cross validation with a fresh grid of hyper parameters further extending the min_samples_leaf downward (ref figure below). Usually the min_samples_split will be 3 times the min_samples_leaf. Hence, those parameters are also adjusted accordingly.

```
param_grid = {
    'max_depth': [7, 9, 10, 12],
    'min_samples_leaf': [5, 10, 15, 20],
    'min_samples_split': [15, 30, 45, 60]
}
```

Figure 37 CART - Hyper parameters initialization - Grid search 2

Once the Grid search CV is re run, we need to fit our training dataset in to the Grid Search CV model and the best parameters obtained are max_depth =9, min_samples_leaf=5, min_samples_split=15. Similar findings as in the first case of Cross validation is noticed here also. Hence the hyper parameters are re initialized again (ref figure below) and the Grid search CV is run again.

```
param_grid = {
    'max_depth': [7, 9, 10, 12],
    'min_samples_leaf': [3, 4, 5, 10],
    'min_samples_split': [9, 12, 15, 30]
}
```

Figure 38 CART - Hyper parameters initialization - Grid search 3

The hyper parameters thus obtained are max_depth =9, min_samples_leaf=5, min_samples_split=12. The model is evaluated for each iteration of the grid search CV and the evaluation results are as below.

Model	Train Data		Test Data	
	R2	RMSE	R2	RMSE
Reg. DT Grid Search CV (max_depth =10, min_samples_leaf=10, min_samples_split=45)	0.9939	904.7152	0.9935	925.018
Reg. DT Grid Search CV (max_depth =9, min_samples_leaf=5, min_samples_split=15)	0.9939	908.675	0.9935	922.737
Reg. DT Grid Search CV (max_depth =9, min_samples_leaf=5, min_samples_split=12)	0.9939	908.1311	0.9935	923.33

Table 12 CART Model Evaluation - Best Hyper Parameters

We can see that the model best performs for the hyper parameters (max_depth =9, min_samples_leaf=5, min_samples_split=15) i.e the lowest RMSE value.

RANDOM FOREST - MODEL BUILDING APPROACH, METRICS & MODEL TUNING

Random Forest is an ensemble technique, where in multiple CART models are built to get the accuracy in the predicted value. The model building is same except that we need to build a random forest Regressor model instead of decision tree Regressor. In addition to the hyper parameters of Decision tree classifier, we have max_features and n_estimators. Max_features refer to the number of variables to be considered and the n_estimators define the number of decision trees to be built.

The Random forest Regressor model is first built with the n_estimators = 501 & the remaining hyper parameters are set to the default value. The hypothesis is that more the decision trees built, greater is the performance of the model.

The values of R^2 & RMSE values are as in the below table. The model is over fit and is evident from the wide gap between the RMSEs of train and test data.

Model	Train Data		Test Data	
	R2	RMSE	R2	RMSE
Random Forest Model	0.9988	390.449	0.993	963.791

Table 13 Metrics - RF - Default Hyper Parameters

The optimum hyper parameters here are also obtained by grid search cross validation. The parameter Grid for for Grid Search CV is as in the figure below.

```
param_grid = {
    'max_depth': [9,10],
    'max_features': [6, 7],
    'min_samples_leaf': [5, 10, 15],
    'min_samples_split': [15, 30, 45],
    'n_estimators': [101, 301, 501]
}
```

Figure 39 RF – Grid search CV Parameter -1

Using the best hyper parameters that are obtained by fitting the train data (max_depth = 10, max_features=7, min_samples_leaf=5, min_samples_split=15, n_estimators=301), we shall predict the values for both training & testing dataset. The metrics for this model are as in the table below.

Model	Train Data		Test Data	
	R2	RMSE	R2	RMSE
Random Forest Model with Grid Search CV (max_depth = 10, max_features=7, min_samples_leaf=5, min_samples_split=15, n_estimators=301)	0.9927	991.32	0.9919	1031.87

Table 14 Metrics - RF - Grid Search CV – 1

The grid search, though the consistency of the model seems to be improved, the RMSE is large in the test dataset, which is not viable as it may lead to over stocking or under stocking of products in the warehouse which may lead to increase in inventory costs. We further try to optimize the model by changing the parameter grid as below.

```
param_grid = {
    'max_depth': [5,7,10],
    'max_features': [7,8,9],
    'min_samples_leaf': [10,20,30],
    'min_samples_split': [30,60,90],
    'n_estimators': [50,100,150]
}
```

Figure 40 RF – Grid search CV Parameter -2

Using the best hyper parameters that are obtained by fitting the train data, we shall predict the values for both training & testing dataset. The metrics & best hyper parameters for this model are as in the table below.

Model	Train Data		Test Data	
	R2	RMSE	R2	RMSE
Random Forest Model with Grid Search CV (max_depth = 10, max_features=7, min_samples_leaf=5, min_samples_split=15, n_estimators=301)	0.9927	991.3	0.992	1031.873

Table 15 Metrics - RF - Grid Search CV – 1

Though the models obtained through Grid search validation has some consistency in the train and test data, we will try the bootstrap aggregating (Bagging) & find if that improves the RMSE scores of the Random forest model.

BAGGING

MODEL BUILDING APPROACH & METRICS

Bootstrapping is a technique of sampling by which we can create sub sample of observation with the actual dataset, with replacement. Bagging is also called as bootstrap aggregating. There will be reduced chances of over fitting by training each model only with a randomly chosen subset of the training data. Training can be done in parallel. Essentially trains a large number of “strong” learners in parallel (each model is an over fit for that subset of the data). Combines (averaging or voting) these learners together to "smooth out" predictions. By using the BaggingRegressor model library from sklearn.ensemble, we will be able to fit the train data in to our model. In defining the model, we need to define a base estimator model for which the bagging needs to be done. We will use the Random forest as base model with 100 Decision trees, which must be first created before creating the Bagging classifier model. The train data is then fit to the bagging model and predictions are made for both the train and test data. The model validation results are as in the table below. However, the bagging model is also found to be over fit. Which is evident from the gap in the train and test RMSE values.

Model	Train Data		Test Data	
	R2	RMSE	R2	RMSE
Bagging	0.9974	587.0825	0.9934	929.7687

Table 16 Metrics – Bagging

BOOSTING MODELS

Boosting is a linear sequential process, where next or upcoming model tries to minimize the errors made by previous model in prediction. This method is different from bagging in the sense where each succeeding model is dependent on the previous model. Trains a large number of "weak" learners in sequence. A weak learner is a simple model that is only slightly better than random (E.g., One depth decision tree). Miss-classified data weights are increased for training the next model. So, training has to be done in sequence. Boosting then combines all the weak learners into a single strong learner. Bagging uses complex models and tries to "smooth out" their predictions, while Boosting uses simple models and tries to "boost" their aggregate complexity.

There are 2 boosting methods

- Adaptive boosting
- Gradient boosting.

ADABOOST MODEL BUILDING APPROACH & METRICS

In AdaBoost (adaptive boosting), the successive learners are created with a focus on the ill fitted data of the previous learner. Each successive learner focuses more and more on the harder to fit data i.e., their residuals in the previous tree. Ada Boost Model Building & fitting. By using the AdaBoostRegressor model library from sklearn.ensemble, we will be able to fit the train data in to our model. In defining the model, we need to define number of estimators i.e., no. of decision trees, which we consider as 100. Prediction are then made on both train and test data and the model evaluation results are as in the table below.

Model	Train Data		Test Data	
	R2	RMSE	R2	RMSE
AdaBoost	0.9986	427.6	0.993	958.0611

Table 17 Metrics - Adaboost model

It can be seen that this model is also overfit, which means it performs well in the train data and fails in the test data.

GRADIENT BOOSTING MODEL BUILDING APPROACH & METRICS

In Gradient boosting, each learner is fit on a modified version of original data (original data is replaced with the x values and residuals from previous learner). By fitting new models to the residuals, the overall learner gradually improves in areas where residuals are initially high. By using the GradientBoostingRegressor model library from sklearn.ensemble, we will be able to fit the train data in to our model. In defining the model, we need to define the hyper parameters. We will be going with the default parameters in this case. Prediction are then made on both train and test data and the model evaluation results are as in the table below.

Model	Train Data		Test Data	
	R2	RMSE	R2	RMSE
Gradient Boosting	0.9935	936.7	0.994	902.504

Table 18 Metrics - Gradient Bosting Model

It can be seen that the model is consistent but has a higher RMSE value in both train & test data. However, the RMSE in test data is lesser than that of the train data.

MODELS COMPARISON, INTERPRETATION & BUSINESS IMPLICATIONS

Model	Metrics				Model Fitness	Business Implication
	Train Data		Test Data			
	R2	RMSE	R2	RMSE		
Linear Regression using SKLEARN	0.9766	1780.4493	0.9777	1716.9875	Good	Larger value of RMSE shows that variation in prediction with rest to actual weight is high and predictions might sometime lead to Inventory cost loss in the ware house.
Linear Regression using SKLEARN - Scaled	0.9766	0.1528	0.9777	0.149	Good	
Linear Regression using Statsmodel	0.977	1780.4493	0.977	1716.9875	Good	
Linear Regression using Statsmodel after removing non significant variables	0.977	1780.72	0.977	1716.7562	Good	
CART Model (DT)	0.9998	154.7	0.9878	1270.2599	Poor	Model performs exceptionally well in the train data with a low RMSE amongst the models built. However, the model is overfit that is evident from the wide gap between the RMSEs of train and test data. Deploying this model for production would incur huge losses to the business
Reg. Cart DT (max_depth =10, min_samples_leaf=10, min_samples_split=30)	0.9941	891.524	0.9935	926.8318	Good	Larger value of RMSE shows that variation in prediction with rest to actual weight is high and predictions might sometime lead to Inventory cost loss in the ware house.
Reg. Cart DT Grid Search CV (max_depth =10, min_samples_leaf=10, min_samples_split=45)	0.9939	904.7152	0.9935	925.018	Good	
Reg. Cart DT Grid Search CV (max_depth =9, min_samples_leaf=5, min_samples_split=15)	0.9939	908.675	0.9935	922.737	Good	
Reg. Cart DT Grid Search CV (max_depth =9, min_samples_leaf=5, min_samples_split=12)	0.9939	908.1311	0.9935	923.33	Good	
Random Forest Model	0.9988	390.449	0.993	963.7913	Poor	Model performs exceptionally well in the train data with a low RMSE amongst the models built. However, the model is overfit that is evident from the wide gap between the RMSEs of train and test data. Deploying this model for production would incur huge losses to the business
Random Forest Model with Grid Search CV (max_depth = 10, max_features=7, min_samples_leaf=5, min_samples_split=15, n_estimators=301)	0.9927	991.32	0.9919	1031.8729	Good	Larger value of RMSE shows that variation in prediction with rest to actual weight is high and predictions might sometime lead to Inventory cost loss in the ware house.
Random Forest Model with Grid Search CV (max_depth = 10, max_features=9, min_samples_leaf=10, min_samples_split=30, n_estimators=150)	0.9938	913.4476	0.9935	924.6953	Good	
Bagging	0.9974	587.0825	0.9934	929.7687	Poor	Model performs exceptionally well in the train data with a low RMSE amongst the models built. However, the model is overfit that is evident from the wide gap between the RMSEs of train and test data. Deploying this model for production would incur huge losses to the business
AdaBoost	0.9986	427.6	0.993	958.0611	Poor	
Gradient Boosting	0.9935	936.6518	0.9938	902.504	Good	Larger value of RMSE shows that variation in prediction with rest to actual weight is high and predictions might sometime lead to Inventory cost loss in the ware house.

Table 19 Summary of Models - Interpretations & Business Implications

The table above has the summary of the models, their fitness/ interpretations & business implications. We can conclude that the Gradient boosting model can be deployed for prediction of weights as it has the least RMSE in test data. Also, it is the only model where the RMSE has decreased for testing data in comparison to

the training data. This implies the least possible inventory cost loss to the company. These predictions also help in pre planning and specific campaigns that has the potential to boost sales which aids to further boosts the bottom line of company.

In addition to the interpretation & Business implications from Predictive Model building approach, the company also needs to focus on the implications that were shared after the Exploratory Data Analysis.

END