**ASSET CLUSTERING**

**temp.py**

1. **establishing connection with database**
2. **saving inspection file from southasia\_cmc\_sandbox.fcg\_inspectionreport to csv called inspection**

Query:

select inspection\_date, fuel, make, year, model, price, mileage, location, odometer, roadtaxpaid,cubiccapacity,insurancetype,numberofowners,registeredcity,roadtaxvalidity,manufacturingdate,caronhypothecation,registrationnumber,chassiscolor,insurancetype,damagessummary,pricewebsitequote,bodyexteriordesign,bodyinteriordesign from southasia\_cmc\_sandbox.fcg\_inspectionreport;

1. **reading inspection data from csv with date-time dtype**

Query:

"select location,createdon,mx\_target\_price ,mx\_km, mx\_price\_offered\_to\_customer , mx\_make ,mx\_model, mx\_final\_buying\_price , mx\_final\_selling\_price, mx\_reg\_no, min\_quote\_price , max\_quote\_price,junkflag from southasia\_cmc\_sandbox.ods\_leads\_inc;

1. **fetching data for last 2.5 years**
2. **Merged the inspection and price on registration number of the car**
3. **removing cars with no make and no model ( as we can’t recommend a car with no make or model) => 779565 rows**
4. **Replacing null values of model from inspection.csv by mx\_model from price.csv. Similar steps were followed for make.**
5. **Mapping Fuel Type**

* 'diesel/hybrid':'diesel/hybrid'
* 'inspection.field.option.petrolElectric':'petrolElectric'
* 'inspection.field.option.dieselElectric':'dieselElectric'
* 'petrol/hybrid':'petrol/hybrid'
* 'Inspection.field.option.petrol':'petrol'
* 'Petrol':'petrol'
* 'inspection.field.fuelType.petrol':'petrol'
* 'PETROL':'petrol'
* 'selfInspection.field.option.petrol':'petrol'
* 'Inspection.field.option.diesel':'diesel'
* 'Diesel':'diesel'
* 'inspection.field.fuelType.diesel':'diesel'
* 'DIESEL':'diesel'
* 'Electric':'elecric'
* 'inspection.field.fuelType.electric':'electric'
* 'electric(bov)':'electric'
* 'CNG':'cng'
* 'selfInspection.field.option.cng':'cng'
* 'inspection.field.option.compressedNaturalGas':'cng'
* 'inspection.field.fuelType.CGN':'cng'
* 'Cngonly':'cng'
* 'inspection.field.option.petrolCompressedNaturalGas':'cng\_petrol'
* 'inspection.field.fuelType.CGNpetrol':'cng\_petrol'
* 'cng petrol':'cng\_petrol'
* 'CNGPETROL':'cng\_petrol'
* 'inspection.field.option.petrolLiquifiedPetroleumGas':'lpg\_petrol'
* 'inspection.field.fuelType.LPGpetrol':'lpg\_petrol'
* 'petrol/lpg':'lpg\_petrol'
* 'LPGPETROL':'lpg\_petrol'
* 'petrol lpg':'lpg\_petrol'
* 'inspection.field.option.liquifiedPetroleumGas':'lpg'
* 'lpg only':'lpg'

1. **Deleting duplicates for registration numbers, make, model, inspection\_year, fuel, etc.**

408787 rows

1. **Mapping**

**Numberofowners (1/2/3/4/5)**

1:1

2:2

3:3

4:4

5:5

'selfInspection.field.option.two':2

'selfInspection.field.option.one':1

**bodyexteriordesign and interiorbodydesign (0/1/2/3/4/5)**

'Inspection.field.option.3':3

'Inspection.field.option.4':4

'Inspection.field.option.2':2

'3':3

'4':4

'Inspection.field.option.5':5

3.0:3

'Inspection.field.option.1':1

4.0:4

'2':2

'5':5

2.0:2

5.0:5

'Inspection.field.option.0':0

'1':1

1.0:1

'0':0

0.0:0

**Insurancetype (5 broad categories)**

'thirdParty':'thirdParty'

'Comprehensive':'comprehensive'

'Expired':'expired'

'inspection.field.insuranceType.none':'none'

'zeroDepreciation':'zeroDepreciation’

'Inspection.field.option.comprehensive':'comprehensive'

'inspection.field.insuranceType.comprehensive':'comprehensive'

'inspection.field.option.zeroDepreciation':'zeroDepreciation'

'inspection.field.option.3rdParty':'thirdParty'

'inspection.field.insuranceType.zeroDep':'zeroDepreciation'

'inspection.field.insuranceType.3rdParty':'thirdParty'

'ThirdParty':'thirdParty'

'InsuranceExpired':'expired'

'Comprehensive':'comprehensive'

'ZeroDep':'zeroDepreciation'

'selfInspection.field.option.expired':'expired'

'selfInspection.field.option.thirdParty':'thirdParty'

**City**

**Color**

1. **Defined column age = inspection year - manufacture year**
2. **Identifying mx\_target\_price as the most useful column for price**
3. **Saved df17 to cleaned\_data.csv**

**preprocessing.py**

1. **Reading cleaned\_data.csv**
2. **Replacing garbage values like DibberDobber1234, NA ok, Hdj in cubiccapacity column by np.nan**
3. **Filling null values of fuel by method like vlookup from same make/model of car**

**(Reducing 3059 null values to 335 null values)**

**Filling null values of fuel by mode at last**

1. **filling null values in proportion with past data present**
2. **Filling null values in color by mode of the color of the cars with same make and model**

**(Reducing 11633 null values to 892 null values)**

**Filling null values in color by mode of the color of the cars with same make**

**Lastly filling null values of the mode of the total dataset**

1. **Filling null values of caronhypothecation by mode**
2. **Filling null values of registeredcity by proportion of the values present in the original column**
3. **Mapping of make and model**
4. **Used type\_calculated.csv for type and transmission of the car**
5. **Saved df to half\_cleaned\_data.csv**
6. **Treatment of outliers**
7. **KNN imputation for numerical columns to fill null values**
8. **Standardize numerical columns**
9. **Saved df to outliers\_removed+knn.csv**
10. **Removing spelling mistakes in type, transmission**
11. **Defining zones as a separate columns**
12. **Save to new\_clustering\_data.csv**

Columns

'Make'

'Mx\_target\_price'

'Fuel'

'Type'

'Transmission'

'Km'

'Age'

'Cubiccapacity'

'Insurancetype'

'Numberofowners'

'Zone'

'Caronhypothecation'

'Chassiscolor'

'Damagessummary'

'Bodyexteriordesign'

'Bodyinteriordesign'

**Data\_visualisation.py**

Used matplotlib, seaborn and plotly to visualise different graphs

**Clustering.py**

Clustering using different methods

**final\_clustering.py**

1. **Importing outliers removed+knn.csv**
2. **Defining variable car=make+model**
3. **Removing spelling mistakes in type, transmission**
4. **Defining zones of car**
5. **Choosing most important features for clustering**
6. **Scaling numerical data**
7. **One hot encoding categorical columns**
8. **Performing PCA with 95% confidence interval**
9. **Clusters using kmeans for k=3,4,5,6**
10. **Choosing k=5 for car cluster by analysing cluster summary**
11. **Saving clusters**
12. **Analysing cluster with k=5 and getting graphs for it**

**DEALER CLUSTERS**

**dealer\_cleaning+clustering.py**

1. **Saving dealer\_data from southasia\_cmc\_sandbox.dealerprofile and saving it to dealer\_data.csv**

Query for dealerprofile table:

create table southasia\_cmc\_sandbox.dealerprofile

as (select mx\_reg\_no,mx\_highest\_bid\_dealer\_id ,mx\_make,mx\_model,mx\_km,mx\_target\_price,city,mx\_city,mx\_state,mx\_fuel\_type,createdon

from southasia\_cmc\_sandbox.ods\_leads\_inc

where ((mx\_highest\_bid\_dealer\_id<>'') and (mx\_highest\_bid\_dealer\_id is not null) and (mx\_reg\_no is not null) and (mx\_reg\_no <>'')))

1. **Importing outliers\_removed+knn.csv**
2. **Merging the two dataframes by registration number**
3. **Making corrections in spelling in type and transmission**
4. **Save csv to cleaned\_dealer\_data.csv**
5. **getting numerical attributes and treating outliers introduced**
6. **Creating dealer profile:**

* Getting total car count, age median, tp median, km median, cc median and damagessummary median
* Calculating the proportion of the type of the cars bought by the dealer
* Getting median of the target price of each type of the car
* Getting the number of each type of cars bought by dealer

1. **Joining data**
2. **Standardizing data**
3. **PCA**
4. **Clustering using cosine similarity**
5. **Exporting clusters to csv files**

**graphs,mapping \_for\_final\_dealer\_cluster.py**

1. **Reading dealer clusters CSVs**
2. **Plotted 3D graphs for dealer clusters**
3. **Plotted graphs for statistics of each dealer cluster**
4. **Mapping of Asset clusters with dealer cluster**

Reducing variance between features of both cluster sets

Features used:

* Target price
* Age
* KM
* Damages summary
* Proportion of type of cars

**UPDATIONS**

**dealers\_l6m.py**

1. **Importing dealer\_l6m.csv**

Query for dealer\_l6m.csv

1. Filling null values
2. Getting number of damages by the list of damages
3. Mapping some make and model
4. Creating dealer profiles for l6m
5. Importing gmv.xlsx which contains l6m transacting dealers summary
6. Creating activity flag

0 : L3M GMV > 0 Active

1 : L3M GMV == 0 Inactive

2 : 0 < L3M ASP <50000 Scrap

1. **Defining type of cars**
2. **Making corrections in spelling in type**
3. **Dropping irrelevant columns**
4. **Saving CSV to dealer\_l6m\_final\_cleaned.csv**

**delhi\_dealers\_final.py**

1. **Importing Asset Clusters**
2. **Getting value Counts in Each cluster**
3. **Creating cluster profiles**
4. **Importing dealer\_l6m\_final\_cleaned.csv**
5. **Creating dealer profiles**

(numerical : medians of important features )

1. **Alloting new cluster to each dealer on 6 last 6 months behavior**
2. **Getting Best products list for each dealer (priority\_excel dataframe)**

It consists of top 10 cars bought by the dealer on the basis of their count

1. **Creating activity flag**

0 : L3M GMV > 0 Active

1 : L3M GMV == 0 Inactive

2 : 0 < L3M ASP <50000 Scrap

1. **Getting details of dealers**

**such as dealer name, dealer\_ship name, dealer\_number, drm\_name, allotted cluster**

1. **Combining numerical, priority\_excel and details**
2. **Extracting Dehi dealers for testing**
3. **Saving the csv to delhi\_final.csv**
4. **Count of all the cars bought by delhi dealers in l6m => Delhi\_dealers\_car\_count\_l3m\_inactivity.csv**

**R code**

1. **Establishing connection with google sheets**
2. **Taking age, km, damages, number of owners as input along with make and model**
3. **Reading CSVs**

delhi\_final.csv

Delhi\_dealers\_car\_count\_l3m\_inactivity.csv

1. **Searching for make-model in top 10 cars of delhi Dealers**

Cae 1 : If such dealers exist, this subset of dealers is chosen.

Case 2 : Else, the dealers belonging to that cluster with the maximum number of such cars are chosen.

1. **After getting the dealers, the variance between the input features and features from dealer profiles is reduced.**
2. Now, for case 1, a **final score** is calculated taking the score calculated previously, proportion of this make-model bought and the absolute count of this make-model bought into account

While for case 2, the final score is same as score

1. **The list of dealers is then first sorted by the activity flag and then the final score calculated.**
2. **This list is then displayed in the output sheet of the same google sheet**