ASTHMA ANALYTICS:

BRIDGING THE GAP IN ASTHMA CARE​



Team 6:  
Otabek Juraev  
Jozil Jacob  
Saloni Tiwari  
Yashveer Shrimal  
Roshan Dubey

Table of Content:

1. Data Set Up
2. Data Cleaning
3. Exploratory Data Analysis using SQL and Tableau
4. Statistical Testing Using Chi-Squared Test
5. K-Means Clustering
6. K-Modes Clustering
7. Cluster Analysis in Tableau
8. Codes

# Data Set Up

We began by outlining the data flow for our project, we dumped the data into the data warehouse and made 4 main tables (Severity, Hospital, GRPS and Core). This provided a clear roadmap for our subsequent analysis, ensuring data integrity and facilitating targeted insights.

# Data Cleaning

Next, we cleaned the raw data to ensure accuracy and consistency. We addressed missing values, outliers, and inconsistencies, creating a refined dataset that formed the foundation for our analysis. For instance, the length of stay column had invalid and inconsistent values like “A” and “B”, We cleaned these values to have data consistency across our entire dataset

As we moved further in our analysis, we created separate tables for each specific disease based on the diagnosis code, having specific table for each disease helped us analyse each disease in detail.

Exploratory Data Analysis using SQL and Tableau

In this phase, we leveraged SQL queries to extract relevant insights from the dataset. Subsequently, we utilized Tableau for visual exploration, creating interactive visuals to uncover patterns, trends, and relationships within the data.

Statistical Testing Using Chi-Squared Test

Chi-Squared Test for statistical analysis, examining the relationships between categorical variables. The test generated a p-value and a Chi-Square value. A low p-value (<0.05) suggested significant associations, enabling us to make informed decisions about the validity of patterns within our dataset and test our initial hypothesis.

K-Means Clustering

Applying K-Means clustering, we segmented the data into distinct groups based on similarities. This unsupervised machine learning technique allowed us to identify inherent patterns and groupings within the dataset.

For better results we created bins for variables like Age and Total Charges.

The clustering algorithm divided the entire data for asthmatic patients into 5 clusters.

K-Modes Clustering

We also employed K-Modes clustering, specifically designed for categorical data. This approach further enriched our understanding by revealing patterns in non-numeric variables, providing a holistic view of the dataset.

The clustering algorithm divided the entire data for asthmatic patients into 3 clusters.

Analysis of Cluster Results in Tableau

To interpret and communicate the cluster results effectively, we integrated the findings into Tableau. By visualizing the clusters and their characteristics, we gained valuable insights into the underlying structure of the data, facilitating informed decision-making.

Code

SQL Queries:

**--Query to create separate table for respiratory diseases based on diagnosis code:   
CREATE TABLE capstone2.RESPIRATORY\_DISEASES AS**

**SELECT HOSP\_KID**

**,RECNUM**

**,DXCCSR\_RSP001**

**,DXCCSR\_RSP002**

**,DXCCSR\_RSP003**

**,DXCCSR\_RSP004**

**,DXCCSR\_RSP005**

**,DXCCSR\_RSP006**

**,DXCCSR\_RSP007**

**,DXCCSR\_RSP008**

**,DXCCSR\_RSP009**

**,DXCCSR\_RSP010**

**,DXCCSR\_RSP011**

**,DXCCSR\_RSP012**

**,DXCCSR\_RSP013**

**,DXCCSR\_RSP014**

**,DXCCSR\_RSP015**

**,DXCCSR\_RSP016**

**,DXCCSR\_RSP017**

**FROM `capstone - 400517. capstone2.kid\_GPRS`**

**--QUERY FOR asthma cases by Gender**

**SELECT FEMALE**

**,count(\*)**

**FROM `capstone - 400517. capstone2.table\_core`**

**WHERE Asthma <> '0'**

**AND Asthma IS NOT NULL**

**GROUP BY 1**

**--QUERY FOR asthma cases by INCOME**

**SELECT ZIPINC\_QRTL**

**,count(\*)**

**FROM `capstone - 400517. capstone2.table\_core`**

**WHERE Asthma <> '0'**

**GROUP BY 1**

**--QUERY FOR asthma cases by race**

**SELECT RACE**

**,count(\*)**

**FROM `capstone - 400517. capstone2.table\_core`**

**--WHERE Asthma <> '0'**

**GROUP BY 1**

**--query for asthma and infectious diseases**

**SELECT Asthma**

**,INF\_All**

**,COUNT(\*) AS INFANDASTHMA**

**FROM `capstone - 400517. capstone2.table\_core`**

**WHERE Asthma <> '0'**

**AND INF\_All <> 0**

**GROUP BY 1**

**,2**

**--COMORBIDITY ANALYSIS FOR ASTHMA AND OTHER DISEASES**

**SELECT RECNUM**

**FROM `capstone - 400517. capstone2.table\_core`**

**WHERE (**

**Blood\_All <> 0**

**OR MAL\_All <> 0**

**OR MBD\_ALL <> 0**

**OR END\_ALL <> 0**

**OR INF\_All <> 0**

**OR DIG\_ALL <> 0**

**OR NVS\_ALL <> 0**

**OR PRG\_All <> 0**

**OR CIR\_All <> 0**

**OR SKN\_All <> 0**

**OR MUS\_All <> 0**

**OR Tumor\_All <> 0**

**OR EYE\_All <> 0**

**OR EAR\_All <> 0**

**)**

**AND Asthma <> '0'**

# **--Data cleaning in gender**

**DELETE**

**FROM `capstone - 400517. capstone2.table\_core`**

**WHERE FEMALE IN (**

**'A'**

**,'C'**

**,'nan'**

**);**

**--query to get total charges by payer**

**SELECT**

**--RECNUM**

**Pay1**

**,sum(TotalChargesInteger)**

**FROM `capstone - 400517. capstone2.table\_core`**

**WHERE Asthma <> '0'**

**AND (**

**Blood\_All = 0**

**AND MAL\_All = 0**

**AND MBD\_ALL = 0**

**AND END\_ALL = 0**

**AND INF\_All = 0**

**AND DIG\_ALL = 0**

**AND NVS\_ALL = 0**

**AND PRG\_All = 0**

**AND CIR\_All = 0**

**AND SKN\_All = 0**

**AND MUS\_All = 0**

**AND Tumor\_All = 0**

**AND EYE\_All = 0**

**AND EAR\_All = 0**

**)**

**AND PAY1 IN (**

**'1'**

**,'2'**

**)**

**--and race <> 'nan'**

**--and Race not like '%nan'**

**GROUP BY 1**

**--Query to check count of each endocrine disease**

**SELECT SUM(CASE**

**WHEN CAST(DXCCSR\_END001 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_1**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END002 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_2**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END003 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_3**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END004 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_4**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END006 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_6**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END007 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_7**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END008 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_8**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END009 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_9**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END010 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_10**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END011 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_11**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END012 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_12**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END013 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_13**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END014 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_14**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END015 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_15**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END016 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_16**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_END017 AS INT64) = 1**

**THEN 1**

**ELSE 0**

**END) AS Count\_17**

**,**

**FROM `capstone - 400517. capstone2.ENDOCRINE\_DISEASES`**

**--Query to analyze asthma and length of stay**

**SELECT RECNUM**

**,LOS**

**FROM `capstone - 400517. capstone2.table\_core`**

**WHERE (**

**Blood\_All <> 0**

**OR MAL\_All <> 0**

**OR MBD\_ALL <> 0**

**OR END\_ALL <> 0**

**OR INF\_All <> 0**

**OR DIG\_ALL <> 0**

**OR NVS\_ALL <> 0**

**OR PRG\_All <> 0**

**OR CIR\_All <> 0**

**OR SKN\_All <> 0**

**OR MUS\_All <> 0**

**OR Tumor\_All <> 0**

**OR EYE\_All <> 0**

**OR EAR\_All <> 0**

**)**

**AND Asthma <> '0'**

**AND Asthma IS NOT NULL**

**--Query to get count of individual infectious diseases**

**SELECT SUM(CASE**

**WHEN CAST(DXCCSR\_INF001 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF001**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF002 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF002**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF003 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF003**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF004 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF004**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF005 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF005**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF006 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF006**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF007 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF007**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF008 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF008**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF009 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF009**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF010 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF010**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF011 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF011**

**,SUM(CASE**

**WHEN CAST(DXCCSR\_INF012 AS INT64) <> 0**

**THEN 1**

**ELSE 0**

**END) AS Count\_INF012**

**FROM `capstone - 400517. capstone2.Infectious\_DISEASES`**

**--Query to clean total charges col.**

**--ALTER TABLE `capstone-400517.capstone2.table\_core`**

**--ADD COLUMN TotalChargesInteger INT64;**

**-- Copy data from the existing column to the new column**

**UPDATE `capstone - 400517. capstone2.table\_core`**

**SET TotalChargesInteger = CASE**

**WHEN TOTCHG = 'nan'**

**THEN NULL -- or set to a default value**

**WHEN TOTCHG = 'A'**

**THEN NULL**

**WHEN TOTCHG = 'C'**

**THEN NULL**

**WHEN TOTCHG = 'TOTCHG'**

**THEN NULL**

**ELSE CAST(TOTCHG AS INT64)**

**END**

**WHERE 1 = 1**

**--Query to get LOS AND TOTAL CHARGE FOR PATIENTS WITH ASTHMA AND OTHER DISEASES**

**SELECT t1.DXCCSR\_RSP009**

**,COUNT(\*)**

**FROM (**

**SELECT RECNUM**

**,DXCCSR\_Default\_DX1**

**,DXCCSR\_RSP009**

**FROM `capstone - 400517. capstone2.kid\_GPRS` A11**

**WHERE A11.DXCCSR\_Default\_DX1 = ‘RSP009’**

**) t1**

**GROUP BY 1**

**--query to create separate table for CongenitalMalfunction**

**CREATE TABLE capstone2.CongenitalMalfunction\_DISEASES AS**

**SELECT HOSP\_KID**

**,RECNUM**

**,DXCCSR\_MAL001**

**,DXCCSR\_MAL002**

**,DXCCSR\_MAL003**

**,DXCCSR\_MAL004**

**,DXCCSR\_MAL005**

**,DXCCSR\_MAL006**

**,DXCCSR\_MAL007**

**,DXCCSR\_MAL008**

**,DXCCSR\_MAL009**

**,DXCCSR\_MAL010**

**FROM `capstone - 400517. capstone2.kid\_GPRS`**

**--query to make new table for infectious diseases**

**CREATE TABLE capstone2.Infectious \_DISEASES AS**

**SELECT HOSP\_KID**

**,RECNUM**

**,DXCCSR\_INF001**

**,DXCCSR\_INF002**

**,DXCCSR\_INF003**

**,DXCCSR\_INF004**

**,DXCCSR\_INF005**

**,DXCCSR\_INF006**

**,DXCCSR\_INF007**

**,DXCCSR\_INF008**

**,DXCCSR\_INF009**

**,DXCCSR\_INF010**

**,DXCCSR\_INF011**

**,DXCCSR\_INF012**

**FROM `capstone - 400517. capstone2.kid\_GPRS`**

**--Query to make range for LOS**

**UPDATE capstone2.table\_core**

**SET LOS\_Range = CASE**

**WHEN LOS BETWEEN '0'**

**AND '1'**

**THEN '0-1'**

**WHEN LOS BETWEEN '2'**

**AND '3'**

**THEN '2-3'**

**WHEN LOS BETWEEN '4'**

**AND '9'**

**THEN '4-9'**

**WHEN LOS > '10'**

**THEN '10+'**

**END**

**WHERE 1 = 1**

**--query to make range for total charges**

**UPDATE capstone2.table\_core**

**SET TotalChargeRange = CASE**

**WHEN TotalChargesInteger BETWEEN 0**

**AND 12000**

**THEN 1**

**WHEN TotalChargesInteger BETWEEN 12001**

**AND 20000**

**THEN 2**

**WHEN TotalChargesInteger BETWEEN 20001**

**AND 40000**

**THEN 3**

**WHEN TotalChargesInteger BETWEEN 40001**

**AND 80000**

**THEN 4**

**WHEN TotalChargesInteger > 80001**

**THEN 5**

**END**

**WHERE 1 = 1**

**# Python script for calculating Chi-Squared value and p value**

**from scipy.stats import chi2\_contingency**

**import matplotlib.pyplot as plt**

**import numpy as np**

**observed\_data = [[101427, 1333965], [103168, 1248246]]**

**# Perform chi-square test**

**chi2, p, \_, \_ = chi2\_contingency(observed\_data)**

**print(f"P-value: {p}")**

**print(f"Chi-squared value: {chi2}")**

Code for clustering:

# Filtering only for RSP009

results = job.to\_dataframe()

results = results[results['DXCCSR\_Default\_DX1']== 'RSP009']

# Import numpy to handle NaN values

import numpy as np

# Replace all instances of 'nan' with NaN in the results DataFrame

results.replace('nan', np.nan, inplace=True)

# Drop NA values from table

results.dropna(inplace=True)

results.replace('A', np.nan, inplace=True)

# Drop NA values form table

results.dropna(inplace=True)

# check if we have NA values among unique values

cols = results.columns

for i in cols:

  print(i, results[i].unique())

#Converting all values into numeric so that we can run ML algorithms

import pandas as pd

# Convert all columns to numeric, coercing errors to NaN

results = results.apply(pd.to\_numeric, errors='coerce')

# Convert to Int64

results = results.astype('Int64')

#Checking if the values converted into numeric

cols = results.columns

for i in cols:

  print(i, results[i].unique())

#Converting all values into numeric so that we can run ML algorithms

import pandas as pd

# Convert all columns to numeric, coercing errors to NaN

results = results.apply(pd.to\_numeric, errors='coerce')

# Convert to Int64

results = results.astype('Int64')

#Checking if the values converted into numeric

cols = results.columns

for i in cols:

  print(i, results[i].unique())

# Separating the features that  we need for K-Mode algorthm

df\_kmode = results[['AGE','FEMALE', 'RACE', 'HOSP\_REGION', 'ZIPINC\_QRTL', 'LOS', 'TOTCHG', 'PAY1']]

# Creating bins for Length of stay column

## labels = ['0-1', '2-3', '4-9', '10 and over']

bins = [-1, 1, 3, 9, float('inf')]  # Using -1 as the lower bound to include 0

labels = [1, 2, 3, 4]

# Create a new column with the binned categories

df\_kmode.loc[:, 'LOS'] = pd.cut(df\_kmode['LOS'], bins=bins, labels=labels)

# Creating bins for "Total charge" feature

## labels = ['0-12000', '12,000-20,000' '20,000-40,000', '40,000-80,000', '80,000+']

bins = [-1, 12000, 20000, 40000, 80000, float('inf')]  # Using -1 as the lower bound to include 0

labels = [1, 2, 3, 4, 5]

# Create a new column with the binned categories

df\_kmode.loc[:, 'TOTCHG'] = pd.cut(df\_kmode['TOTCHG'], bins=bins, labels=labels)

# Creating bins for "AGE" feature of the model

## labels = ['0-1', '1-10' '11-20']

bins = [-1, 1, 10, 21]  # Using -1 as the lower bound to include 0

labels = [1, 2, 3]

# Create a new column with the binned categories

df\_kmode.loc[:, 'AGE'] = pd.cut(df\_kmode['AGE'], bins=bins, labels=labels)

# Install K-Modes ML library

!pip install kmodes

# Import necessary libraries from KModes and create clusters

from kmodes.kmodes import KModes

import pandas as pd

from sklearn.preprocessing import OneHotEncoder

encoder = OneHotEncoder(sparse=False)

df\_encoded = encoder.fit\_transform(df\_kmode)

km = KModes(n\_clusters=3, init='Huang', n\_init=5, verbose=1)

#Check of the clusters created in vector format

df\_encoded

# Fit the cluster

clusters = km.fit\_predict(df\_encoded)

# Checking the optimal number of clusters using Elbow method

import matplotlib.pyplot as plt

from kmodes.kmodes import KModes

costs = []

for num\_clusters in range(1, 10):

    km = KModes(n\_clusters=num\_clusters, init='Huang', n\_init=5, verbose=0)

    km.fit(df\_encoded)

    costs.append(km.cost\_)

plt.plot(range(1, 10), costs, marker='o')

plt.xlabel('Number of clusters')

plt.ylabel('Cost')

plt.title('Elbow Method For Optimal Number of Clusters')

plt.show()

# Join the cluster results to the original table

df\_kmode['Cluster'] = clusters

# Using dataframe df\_kmode: save to csv file for future analysis and visualization

df\_kmode.to\_csv('df\_kmode\_Asthma\_primary\_3.csv', index=False)

#Importing Library for kMeans

import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

#Scaling the dataset

scaler = StandardScaler()

scaled\_features = scaler.fit\_transform(df\_asthma)

# Elbow Method

inertia = []

for i in range(1, 11):

    kmeans = KMeans(n\_clusters=i, random\_state=0)

    kmeans.fit(scaled\_features)

    inertia.append(kmeans.inertia\_)

plt.plot(range(1, 11), inertia)

plt.title('Elbow Method')

plt.xlabel('Number of clusters')

plt.ylabel('Inertia')

plt.show()

# Initialize KMeans with k clusters

kmeans = KMeans(n\_clusters=5, random\_state=0)

# Fit the model

kmeans.fit(scaled\_features)

# Get the cluster labels for each data point

labels = kmeans.labels\_

# Add the cluster labels as a new column in the original DataFrame

df\_asthma['Cluster'] = labels

#Export result to CSV

df\_asthma.to\_csv("/df\_cluster\_new.csv")