APRIL 21, 2021

INSURANCE CLAIMS DATA AND ANALYSIS

CASE STUDY 2

Bhargav Bindiganavile, Isaacson Michel, Jianying Zhu, Rezarta Myrtollari

Introduction

The All Payer Claim Data (APCD) database contains data about inpatient discharge, outpatient procedures and services, emergency departments and the revenues for each of the mentioned types. Each dataset gives information about case-specific diagnostic discharge, socio-demographic characteristics of patients, medical issues resulting in admission, treatment and services, duration of patients stay in the health facility, and lastly, total service-specific charges billed by the hospital. Because of the vast amount of information, this database is suitable for applying Machine Learning Algorithms and discovering new insights or addressing current challenges in healthcare.

In this report we are using the APCD database to conduct a cluster analysis of specific cost categories of the inpatient hospital DRGs. In the following sessions we are presenting: (1) overview of cluster analysis based on the K-Means algorithms; (2) description of methodology used: presenting the step by step of modeling in an practical way; (3) results exhibition; (4) interpretation of the results obtained: classifying inpatient Diagnostic Related Groups (DRG) admission, which are reflected in terms of the cost and examine the relationships within three major clusters (or cost categories); and (5) a summary of our findings.

1. A brief overview of cluster analysis

Clustering is the task of identifying similar instances and assigning them to clusters, or groups of similar instances [1]. Clustering is classified as an unsupervised machine learning task, meaning that there are no labels (or targets), so in this case it is the algorithm which tries to find patterns in the data. All the objects which belong to a group or cluster have some common properties which differentiate them from the other objects.

One of many applications of clustering or cluster analysis (CA) is the healthcare domain. Some of the examples that we can mention are: characterizing psychiatric patients on the basis of clusters of symptoms, finding a group of genes that have similar biological functions, or identifying medical patient groups most in need of targeted interventions [2].

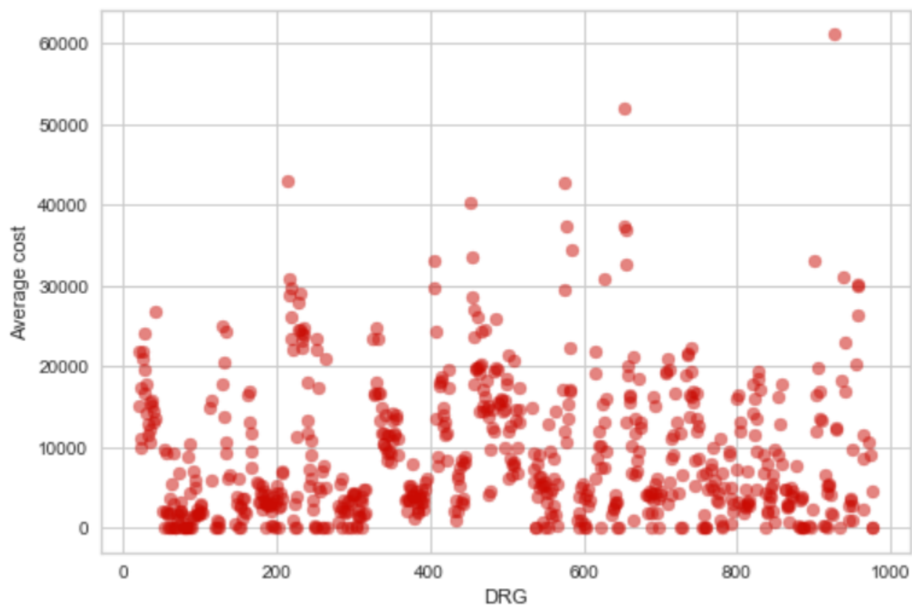
K-means is one of the most popular and clustering algorithms. K is a hyperparameter of the algorithm which determines the number of clusters that we want from our algorithm. Once we specify it, then k new points are assigned randomly in our dataset. These points are called Centroid and during the training phase, the algorithm will try to center them in the middle of the k groups. The initial value of K is very important, thus many times we need to perform a grid search (creating a set of possible values for K, run the algorithm, retrieve a score from a performance metric, compare these scores and then decide on the optimal values). In the following paragraphs, we are going to explain what values of K we are using and what performance metric.

2. Methodology used

For our cluster analysis we are going to use the data from All Payer Claim Data APCD, concretely: Inpatient discharge data and Revenue code file[1]. The initial data processing and the generation of the analytical file is done in MySQL. The analytical file is attached to this submission. It contains a cross tab of DRG and the mean values of PCCR. The total number of DRG selected is 687 and the number of PCCR is 55 including the combined PCCR (PCCR 3700 Operating Room + PCCR 4000 Anesthesiology). The clustering analysis is performed in Python using the Scikit-learn package on the DRG and the single cost combined column named PCCR_OR_and_Anesth_Costs.

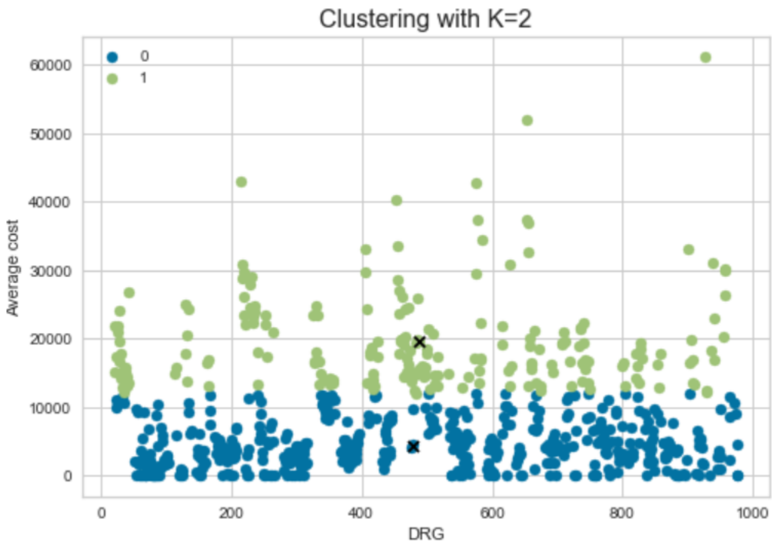
3. CA Results

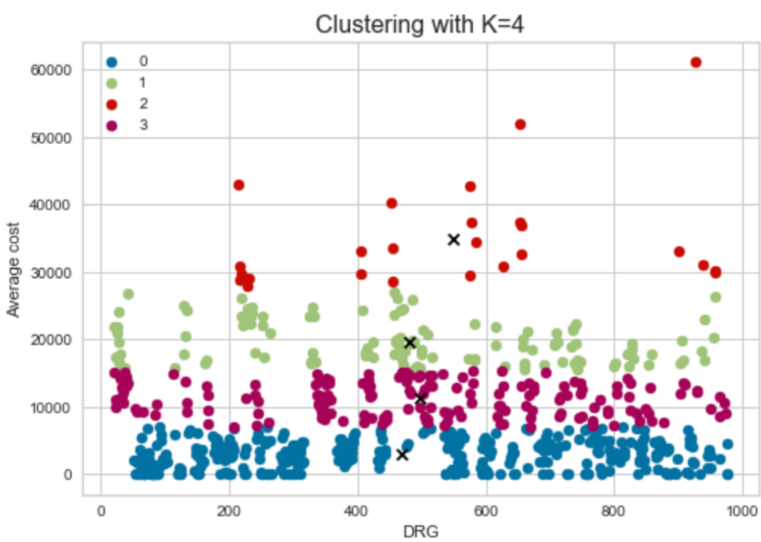
First let us visualize the data we are going to work with in order to understand the clustering. The x axis represents the DRGs and the y axis represents the average cost in USD for the PCC PCCR_OR_and_Anesth_Costs.

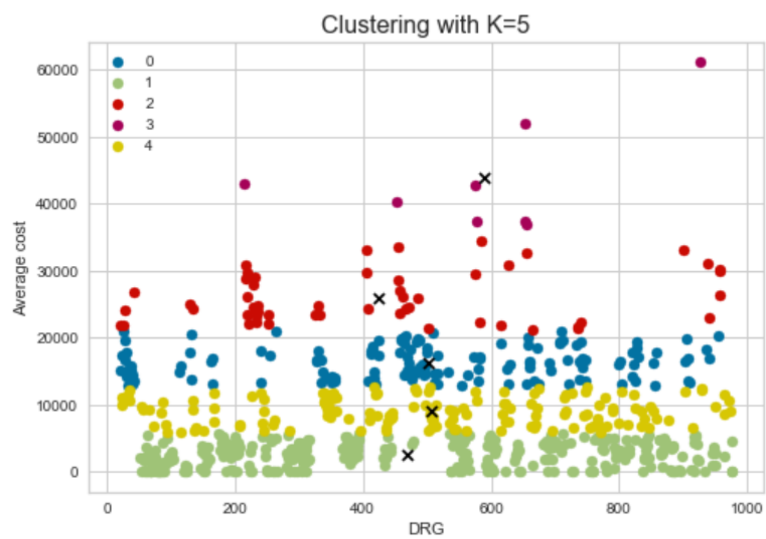


The graphs below show the results of the clustering analysis for different values of the hyperparameter K, specifically 2,3,4, and 5. We are using Calinski and Harabasz score[2] as the performance metric to derive the conclusions about the optimal number of clusters.

[1] More about the files can be found in the description of the study case.
[2] The score is defined as ratio between the within-cluster dispersion and the between-cluster dispersion https://scikit-learn.org/stable/modules/generated/sklearn.metrics.calinski_harabasz_score.html



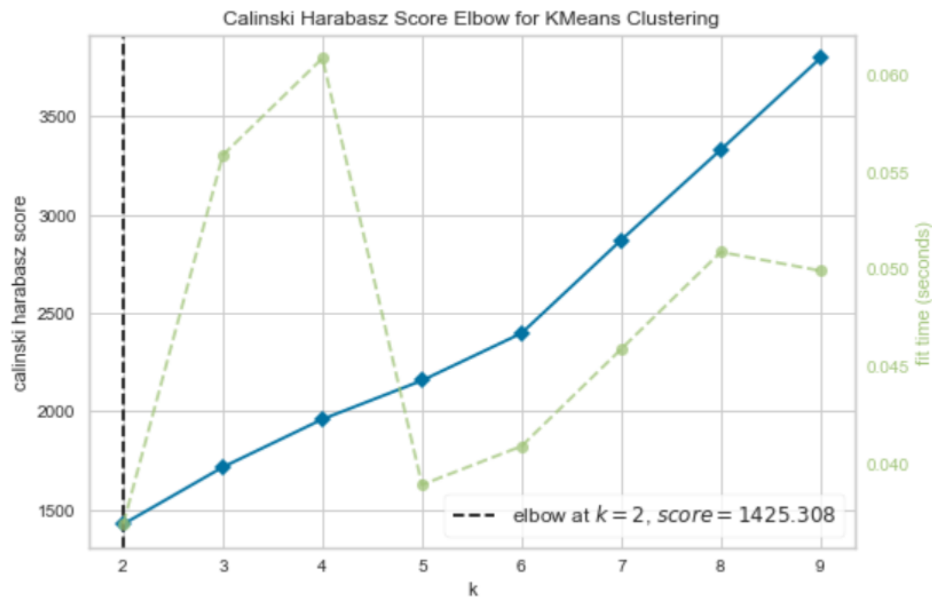




The following table shows the Calinski and Harabasz score for each of the clusters. The highest score is achieved for K=5 (five clusters).

No. of clusters	2	3	4	**5**
Calinski -Harabasz score	1425.30	1715.35	1958.52	**2161.61**

It seems that the highest the number of clusters the larger the score. Let's visualize it for even larger Ks.



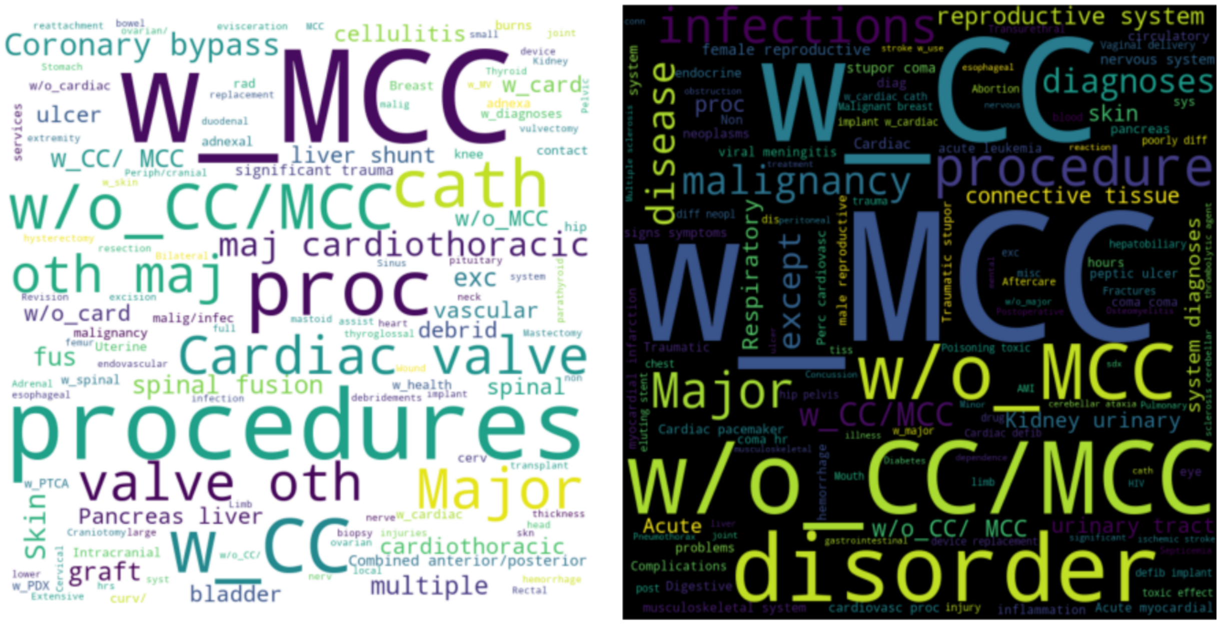
Apparently we would get a higher score for K larger than 5, but the focus in the report will be K=3.

For ease of interpretation, we are using the terms High-cost for clusters labeled with 0, Mid-cost for the one labeled with 1 and Low-cost for those labeled with 2. Note here that labeling from the algorithm is random. The output file for K=3 has the following format:

	DRG	PCCR_OR_and_Anesth_Costs	Clusters
0	Intracranial vascular procedures w_PDX hemorrh...	21805.857143	High-cost
1	Intracranial vascular procedures w_PDX hemorrh...	15172.533333	Mid-cost
2	Intracranial vascular procedures w_PDX hemorrh...	9857.000000	Mid-cost
3	Cranio w_major dev impl/acute complex CNS PDX ...	17395.568182	Mid-cost
4	Cranio w_major dev impl/acute complex CNS PDX ...	11151.166667	Mid-cost

Interpreting the clusters

In order to find out common properties of the hospital inpatient DRGs of each cluster, we compared the word cloud of the DRG for the two groups: High-cost vs Low-cost. Word clouds display the frequency of the words: the larger the size of the word, the more frequent it appears.

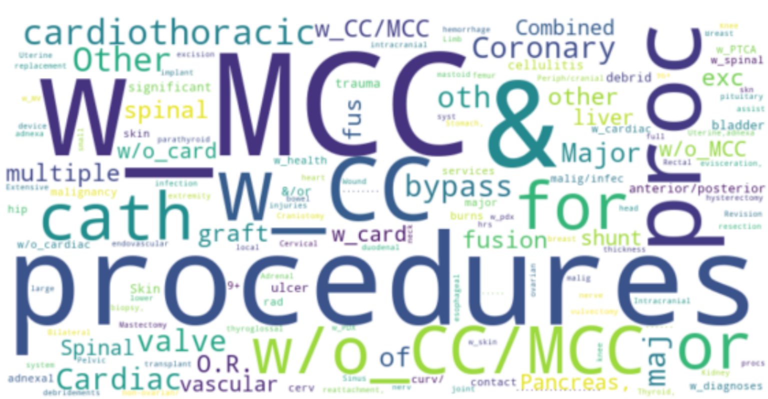


Word cloud for High-cost vs. Low-cost DRGs

While the conclusion for the Low-cost cluster is not straightforward, we can see some kind of pattern in the High-cost cluster, which is the distinction of the frequency for the word **W_MCC** compared to other words. Second in terms of frequency comes the word **W_CC** and **procedures**. It means that in this group, the DRGs accompanied with the phrase "w MCC" and "w CC" and "procedures" are much more in number. The same conclusions derive even from a simple word frequency count like in the picture below:

```
Counter({'w_MCC': 26, 'procedures': 23, '&': 21, 'proc': 13, 'w_CC': 13, 'or': 10, 'cath': 10, 'for':
10, 'w/o_CC/MCC': 9, 'cardiothoracic': 9, 'Other': 8, 'Cardiac': 6, 'valve': 6, 'oth': 6, 'maj': 6,
'Coronary': 6, 'bypass': 6, 'Major': 5, 'MCC': 5, 'of': 5, 'other': 4, 'w_CC/': 4, 'spinal': 4, 'fusi
on': 4, 'fus': 4, 'exc': 4, 'multiple': 4, 'graft': 4, 'O.R.': 4, 'vascular': 3, 'Spinal': 3, 'w_car
```

Word frequency count for the High-cost DRGs

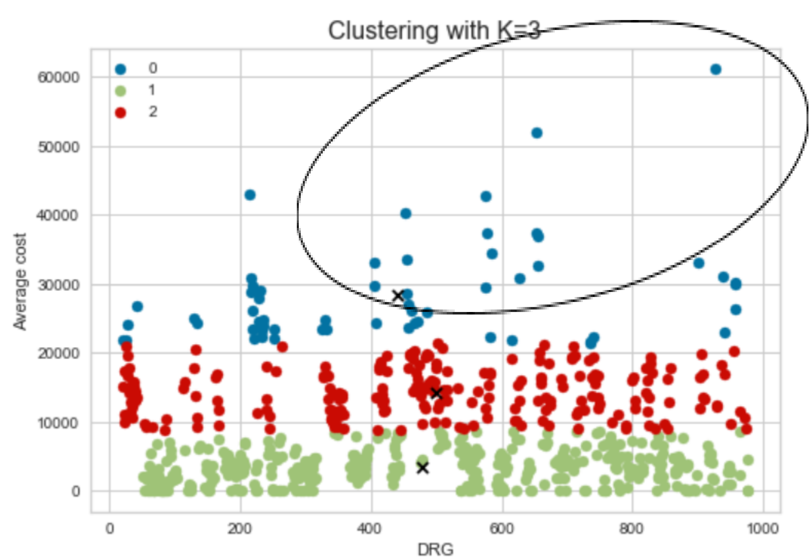


Word cloud for High-cost DRGs based on frequency count

There is also another pattern visible for the high-cost DRGs. It seems that heart-related conditions are typical for this cluster. For example, note the frequency of the words *cardiothoracic, cardiac, valve, coronary, bypass*, *vascular* etc. It might imply that if a DRG is related to the circulatory system, it is classified as a High-cost DRG in the majority of the cases. This could be relative to the amount of risk associated with high-cost DRG and the resources required to work on.

But let's go into more details for the conditions with MCC and CC. **MCC** stands for Major Complications and Comorbidity and **CC** for Complications and Comorbidity. Complications are conditions that appear during the hospital stay, while Comorbidities are pre-existing conditions. "Diagnoses with Complication/Comorbidity (CC) increase the resources used to care for the patient. These diagnoses may increase a patient's length of stay, too. While diagnoses with Major Complications/Comorbidities (MCC) have a larger impact on a patient's stay and always require additional interventions" [3]. Practically, it means that the hospital charges for these patients will be higher compared to patients that do not have any complications or comorbidity. Probably they will require O.R (Operating Room) Procedures, hence explaining the weight in the PCCR_OR_and_Anesth_Costs variable.

Let's see the characteristics of the instances classified as High-cost that are shown in the extremities (upper part of the graph) assumed to have clear similarities between them and clear distinctions from the rest of the clusters. The table below showing top 15 DRG, identifies diseases related to skin or cardiovascular system and organ transplants classified as w CC, w MCC or Major Procedures. Major CC or MDCs increase the costs of these diseases even further. For example, not only organs transplants are extremely resource-intensive procedures, involving high-paid doctors, transportation, expensive drugs that keep both the organs healthy and help the body accept the new organ. We observe that hospitals are trying to make money through the transplants center. They charge high service fees, which is another important factor into the overall cost of related diseases [4] and if it is classified as with MCC or CC the cost is even larger.

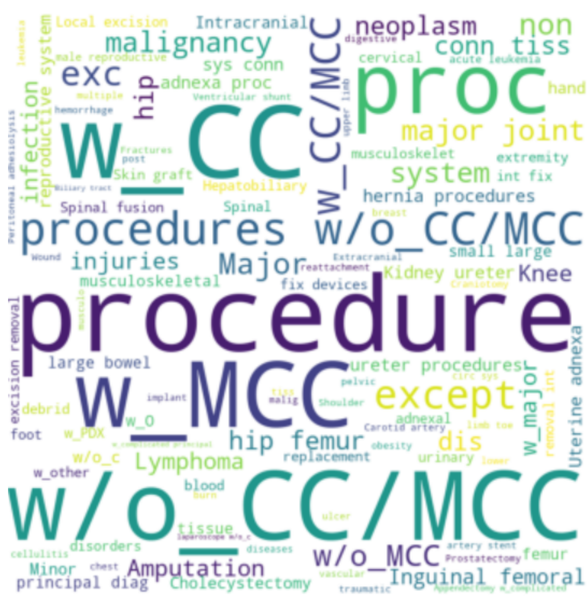


DRG description	PCCR_OR_and_Anesth_Costs	Clusters
Extensive burns or full thickness burns W MV 96+ hrs W skin graft	61064	High-cost
Major bladder procedures W MCC	51883	High-cost
Other heart assist system implant	43013	High-cost
Skin graft &/or debrid exc for skin ulcer or cellulitis W MCC	42625	High-cost
Combined anterior/posterior spinal fusion W MCC	40307	High-cost
Skin graft &/or debrid exc for skin ulcer or cellulitis W CC	37365	High-cost
Kidney transplant	37310	High-cost
Major bladder procedures w/o CC/MCC	36897	High-cost
Breast biopsy, local excision & other breast procedures w/o CC/MCC	34390	High-cost
Combined anterior/posterior spinal fusion W CC	33569	High-cost
Pancreas, liver & shunt procedures W CC	33084	High-cost
Wound debridements for injuries W MCC	33058	High-cost
Major bladder procedures W CC	32706	High-cost
O.R. proc W diagnoses of other contact W health services W MCC	31027	High-cost
Thyroid, parathyroid & thyroglossal procedures w CC	30841	High-cost

Top 15 DRGs in the High-cost cluster

This distinction is less visible for the clusters identified as Mid-cost. In our case we have almost equal numbers of DRGs that are W CC or W MCC and those W/o CC/MCC as shown in the word frequency and word cloud below:

```
Counter({'&': 96, 'procedures': 86, 'w/o_CC/MCC': 77, 'w_CC': 57, 'w_MCC': 47, 'proc': 41, 'O.R.': 3
8, 'for': 37, 'Other': 30, 'or': 29, 'w_CC/MCC': 20, 'of': 20, 'except': 19, 'system': 15, 'Major': 1
2, 'exc': 12, 'procedure': 11, 'w/o_MCC': 10, 'joint': 10, 'tiss': 10, 'sys': 9, 'w/o_c.d.e.': 8, 'co
```



So, what about the DRGs with MCC and CC which are classified as Low-Cost or DRG w/o MCC CC which are classified as Low-Cost?

Here we present the example of a patient with DRG 40 which is classified as High-cost even though it is w/o MCC/C. The patient has 18 out of 20 diagnoses present (the DX1-DX18 variables). Probably some of them impact the severity of its main condition which then is reflected in the hospital charges and as a result the classification in the High-cost cluster.

...	...
DX1	G40409
DX2	S01111A
DX3	S0990XA
DX4	Y9269
DX5	Y990
DX6	I10
DX7	S01552A
DX8	H2103
DX9	M25511
DX10	M25512
DX11	E876
DX12	E1165
DX13	E7800
DX14	R21
DX15	R1013
DX16	Z79899
DX17	Z7901
DX18	Z7984
DX19	
DX20	
CHRGS_HCIA	32321.85
DRG	42

In addition, it is important to stress here the importance of pre-processing the data before applying a model. Sometimes databases, especially those from the healthcare domain, are skewed, contain outliers, human errors, etc. Furthermore, we are dealing with a small number of instances and variables which impacts the training of algorithms. In our case, we had less than 700 instances and were focused solely on one variable.

At the end, to the question " In what cluster would COVID-19 patients be classified?" , the answer is "It depends". If the patients have any major comorbidities like obesity, heart issues, diabetes; or if the patients during the admission in the hospital show other major complications, it will be classified as a High-cost cluster. Otherwise, if the comorbidities or complications are less severe, or do not exist, it will be classified as Mid-cost or Low-cost.

Conclusions

This report has examined insurance claims data for inpatients, which provided many useful insights. We have researched and performed a cluster analysis to interpret the reasons that caused certain types of DRGs to be clustered together. The cluster analysis can identify groups of patients that have similar symptoms and diseases. Although 5 clusters can give a higher performance score, we focus on this problem with 3 clusters. We interpret the issue from three perspectives, namely, features description and frequency of DRGs, the main treatments needed by DRGs, and conditions with MCC and CC, and these factors all more or less contribute to the formation of clusters. The main conclusion is that DRG with MCC and major CC subtypes will be clustered together.

By applying cluster analysis, hospitals could better position themselves, explore new markets of DRG, and develop products that specific clusters of costs find relevant medication and valuable services. Hospitals can also determine the amount of resources to allocate the different areas of their practice of medicine. They could choose to invest more in the areas that attack problems that fall in the cluster with the highest cost as that means that those problems are most likely the most fatal or require the most amount of care.

References

[1] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, Sebastopol: O’Reilly Media, Inc., 2019.

[2] Liao M, Li Y, Kianifard F, Obi E, Arcona S. Cluster analysis and its application to healthcare claims data: a study of end-stage renal disease patients who initiated hemodialysis. *BMC Nephrol*. 2016;17:25. Published 2016 Mar 2. doi:10.1186/s12882-016-0238-2

[3] Brennan D. Clinical Documentation Improvement At UIHC, University of Iowa uhic.org

[4] Houston, J. Why organ transplants are so expensive in the US. Published 2019 September 12. https://www.businessinsider.com/why-organ-transplants-so-expensive-united-states-2019-9

Relevant Coding

Annex 1: SQL Code

```
CREATE DATABASE ICDA;
USE ICDA;
CREATE TABLE `vtinp16_upd` (
  `hnum2` text DEFAULT NULL,
  `ATYPE` INT DEFAULT NULL,
  `asour` INT DEFAULT NULL,
  `intage` INT DEFAULT NULL,
  `TXTZIP` TEXT,
  `sex` INT DEFAULT NULL,
  `dstat` INT DEFAULT NULL,
  `PPAY` INT DEFAULT NULL,
  `CHRGS` DOUBLE DEFAULT NULL,
  `DX1` TEXT,
  `DX2` TEXT,
  `DX3` TEXT,
  `DX4` TEXT,
  `DX5` TEXT,
  `DX6` TEXT,
  `DX7` TEXT,
  `DX8` TEXT,
  `DX9` TEXT,
  `DX10` TEXT,
  `DX11` TEXT,
  `DX12` TEXT,
  `DX13` TEXT,
  `DX14` TEXT,
  `DX15` TEXT,
  `DX16` TEXT,
  `DX17` TEXT,
  `DX18` TEXT,
  `DX19` TEXT,
  `DX20` TEXT,
  `PX1` TEXT,
  `PX2` TEXT,
  `PX3` TEXT,
  `PX4` TEXT,
  `PX5` TEXT,
  `PX6` TEXT,
  `PX7` TEXT,
```

```
`PX8` TEXT,
`PX9` TEXT,
`PX10` TEXT,
`PX11` TEXT,
`PX12` TEXT,
`PX13` TEXT,
`PX14` TEXT,
`PX15` TEXT,
`PX16` TEXT,
`PX17` TEXT,
`PX18` TEXT,
`PX19` TEXT,
`PX20` TEXT,
`ECODE1` TEXT,
`ECODE2` TEXT,
`ECODE3` TEXT,
`hsa` text DEFAULT NULL,
`pdays` INT DEFAULT NULL,
`ccsdx` INT DEFAULT NULL,
`ccsdxgrp` INT DEFAULT NULL,
`CCSPX` TEXT,
`CCSPXGRP` TEXT,
`ccsppx` TEXT,
`ccsppxgrp` TEXT,
`ccsproc` TEXT,
`ccsprocgrp` TEXT,
`DY` INT DEFAULT NULL,
`RECNO` INT DEFAULT NULL,
`BTYPE` INT DEFAULT NULL,
`ERFLAG` INT DEFAULT NULL,
`cah` INT DEFAULT NULL,
`vtres` INT DEFAULT NULL,
`OBSFLAG` INT DEFAULT NULL,
`AFLAG` INT DEFAULT NULL,
`UNIQ` INT DEFAULT NULL,
`ADMID_QTR` INT DEFAULT NULL,
`DISCD_QTR` INT DEFAULT NULL,
`CHRGS_HCIA` DOUBLE DEFAULT NULL,
`SCUD` TEXT,
`DRG` INT DEFAULT NULL,
`MDC` INT DEFAULT NULL,
`sdf` INT DEFAULT NULL,
```

```
  `GROUPER` INT DEFAULT NULL
) ENGINE=MYISAM DEFAULT CHARSET=UTF8MB4 COLLATE = UTF8MB4_0900_AI_CI;

# Importing data from the CVS file
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\vtinp16_upd.csv'
INTO TABLE    vtinp16_upd
CHARACTER SET utf8mb4
FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n' ignore 1 lines;

CREATE TABLE `vtrevcode16` (
  `dy` INT DEFAULT NULL,
  `hnum2` INT DEFAULT NULL,
  `DISCD_QTR` INT DEFAULT NULL,
  `BTYPE` INT DEFAULT NULL,
  `Uniq` INT DEFAULT NULL,
  `REVCODE` INT DEFAULT NULL,
  `REVCHRGS` INT DEFAULT NULL,
  `REVUNITS` INT DEFAULT NULL,
  `CPT` TEXT,
  `PCCR` INT DEFAULT NULL,
  `PRIMARY_CPT` INT DEFAULT NULL,
  `SFLAG` INT DEFAULT NULL,
  `ccsproc` TEXT,
  `ccsprocgrp` TEXT
) ENGINE=MYISAM DEFAULT CHARSET=UTF8MB4 COLLATE = UTF8MB4_0900_AI_CI;

# Importing data from the CVS file
LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\vtrevcode16.txt'
INTO TABLE    vtrevcode16
CHARACTER SET utf8mb4
FIELDS TERMINATED BY ',' ENCLOSED BY '"' LINES TERMINATED BY '\r\n' ignore 1 lines;

# Selecting only important DRG from 20 to 977 (inclusive)
drop temporary table vtinp_16_s;
Create temporary table vtinp_16_s
Select * from vtinp16_upd where drg between 20 and 977;
Select count(distinct(DRG)) from vtinp_16_s; #702 DRG

# filtering the revcodes with charges less than 100 usd
create temporary table vtrevcode16_filtered;
select * from vtrevcode16
where REVCHRGS>100;
```

```
select count(*) from vtrevcode16_filtered; #3,828,913

# linking the two filtered tables
drop temporary table if exists vtinp_rev;
Create temporary table vtinp_rev
select a.REVCHRGS,a.PCCR,a.UNIQ ,b.DRG
from vtrevcode16_filtered as a join vtinp_16_s as b
on a.UNIQ=b.UNIQ; #553,125

# Summing all the charges by the PCCR categories
Create temporary table PCCR_Revcharge
Select UNIQ, DRG, PCCR, Sum(REVCHRGS) as REVCHRGS
From vtinp_rev
Group By UNIQ, DRG, PCCR; #420,390 rows

select * from PCCR_Revcharge;

# Transfering data to excel for further processing
SELECT * FROM PCCR_Revcharge
INTO OUTFILE 'C:\\ProgramData\\MySQL\\MySQL Server 8.0\\Uploads\\PCCR_Revcharge.csv'
FIELDS ENCLOSED BY '"'
TERMINATED BY ';'
ESCAPED BY '"'
LINES TERMINATED BY '\r\n';
```

Annex 2: Python Code

```
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.metrics import calinski_harabasz_score
from sklearn.preprocessing import StandardScaler
from yellowbrick.cluster import KElbowVisualizer
import pandas as pd
import numpy as np
import seaborn as sns

# reading the dataframe
df=pd.read_excel('Clustering.xlsx')
```

```
df.head(5)

# visualzing the data
fig=plt.scatter(df.iloc[:, 0] , df.iloc[:, 1], color='r',alpha=0.5 )
plt.xlabel('DRG')
plt.ylabel('Average cost')
plt.ylabel('')

# analysis with two clusters
scaler = StandardScaler()
scaled_features = scaler.fit_transform(df)
kmeans = KMeans(n_clusters=2).fit(scaled_features)
label = kmeans.fit_predict(df)
kmeans.cluster_centers_
print(calinski_harabasz_score(df, label))

#Getting unique labels
u_labels = np.unique(label)

#plotting the results:
for i in u_labels:
    plt.scatter(df.iloc[label == i , 0] , df.iloc[label == i , 1] , label = i)
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], c='black', marker='x')
plt.xlabel('DRG')
plt.ylabel('Average cost')
plt.title("Clustering with K=2",fontsize=16)
plt.legend()
plt.show()
# analysis with three clusters
scaler = StandardScaler()
scaled_features = scaler.fit_transform(df)
kmeans = KMeans(n_clusters=3).fit(scaled_features)
label = kmeans.fit_predict(df)
kmeans.cluster_centers_
print(calinski_harabasz_score(df, label))

#Getting unique labels
u_labels = np.unique(label)

#plotting the results:
for i in u_labels:
    plt.scatter(df.iloc[label == i , 0] , df.iloc[label == i , 1] , label = i)
```

```
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], c='black', marker='x')
plt.legend()
plt.xlabel('DRG')
plt.ylabel('Average cost')
plt.title("Clustering with K=3",fontsize=16)
plt.show()

# exporting the predictions
df['Clusters']=label
df.head(5)
df.to_excel("Output.xlsx")

# analysis with four clusters
scaler = StandardScaler()
scaled_features = scaler.fit_transform(df)
kmeans = KMeans(n_clusters=4).fit(scaled_features)
label = kmeans.fit_predict(df)
kmeans.cluster_centers_
print(calinski_harabasz_score(df, label))

#Getting unique labels
u_labels = np.unique(label)

#plotting the results:
for i in u_labels:
    plt.scatter(df.iloc[label == i , 0] , df.iloc[label == i , 1] , label = i)
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], c='black', marker='x')
plt.xlabel('DRG')
plt.ylabel('Average cost')
plt.title("Clustering with K=4",fontsize=16)
plt.legend()
plt.show()

# analysis with five clusters
scaler = StandardScaler()
scaled_features = scaler.fit_transform(df)
kmeans = KMeans(n_clusters=5).fit(scaled_features)
label = kmeans.fit_predict(df)
print(calinski_harabasz_score(df, label))
kmeans.cluster_centers_

#Getting unique labels
```

```
u_labels = np.unique(label)

#plotting the results
for i in u_labels:
    plt.scatter(df.iloc[label == i , 0] , df.iloc[label == i , 1] , label = i)
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], c='black', marker='x')
plt.xlabel('DRG')
plt.ylabel('Average cost')
plt.title("Clustering with K=5",fontsize=16)
plt.legend()
plt.show()

# visualizing the 'calinski_harabasz' score
model = KMeans()
# k is range of number of clusters.
visualizer = KElbowVisualizer(model, k=(2,10),metric='calinski_harabasz', timings= True)
visualizer.fit(df)        # Fit the data to the visualizer
visualizer.show()

# generation of the word clouds for the low-cost DRG
# repeat the same for High-cost and Mid-cost DRGs

lowcost=df1.loc[df1['Clusters'] == 'Low-cost']
lowcost.head()

# creating the word cloud
wordcloud = WordCloud(width = 800, height = 800,
                min_font_size = 10, regexp=r"\w[\w/]+").generate(' '.join(lowcost['DRG']))

# plot the WordCloud image
plt.figure(figsize = (6, 6), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)
plt.show()

# word frequencies as dictionaries
from collections import Counter
results = Counter()
lowcost['DRG'].str.split().apply(results.update)
print(results)
```

```
d={1: 'High-cost', 2: 'Low-cost', 0: 'Mid-cost'}
df.Clusters=df.Clusters.map(d)
df.head()

# Top 15 High cost DRGs
sns.catplot( x="Clusters", y="PCCR_OR_and_Anesth_Costs",
data=df,kind='strip',jitter='0.5',order=['Low-cost','Mid-cost','High-cost'])

clusters1 = df.sort_values('PCCR_OR_and_Anesth_Costs', ascending=False)
clusters1.head()

filecode = pd.read_excel('C:\\Users\\14830\\Downloads\\Group2HW2_AnalyticalFile.xlsx')
new_header = filecode.iloc[0]
filecode = filecode[1:]
filecode.columns = new_header
filecode.columns.values[0] = 'DRG_Descriptions'
filecode.head()

clusters_filecode_table = pd.merge(clusters1, filecode, how='left',on=['PCCR_OR_and_Anesth_Costs'])
clusters_filecode_table.rename(columns = {" ": "Prediction", "PCCR_OR_and_Anesth_Costs":
"PCCR_OR_and_Anesth_Costs($)"}, inplace = True)
clusters_filecode_table.head()

top15 =pd.DataFrame(clusters_filecode_table, columns=['DRG', 'DRG_Descriptions',
'PCCR_OR_and_Anesth_Costs($)', 'Clusters'])
top15 .index += 1
print(top15.head(15))
```