Jasmine Leblond-Chartrand

COMP 499 Final Project Preliminary Report

Hypertension

The NHANES (National Health and Nutrition Examination Survey) is a survey conducted across United-States to assess the health and nutritional status of individuals living in that country. The dataset includes extensive information about the participants (i.e. demographics, labs, examinations, and questionnaires)

The 2013-2014 edition examines 9,813 participants.

This project will examine determinants and possible outcomes of hypertension, the condition for which medication is the most often prescribed according to the NHANES medications.csv dataset (n = 1595). QUESTIONS Q1: What are the determinants of hypertension? (feature selection) Q2: Can we identify individuals with hypertension from their general health status? (classification) Q3: Which hypertensive patients are at higher risk for complications? (classification) Q4: What is the age of onset of complications for those at higher risk? (regression)

Data Acquisition (via Kaggle API)

```
In [105]:  # Install the Kaggle API to be allowed download the file

           # pip install kaggle
```

```
In [3]:  # Need to download a kaggle.json file from the user account page on kaggle
         .com
         # put it in ~/.kaggle/kaggle.json
```

```
In [4]:  import kaggle

         # dataset from "https://www.kaggle.com/cdc/national-health-and-nutrition-e
         xamination-survey/"

         !kaggle datasets download -d cdc/national-health-and-nutrition-examination
         -survey -p './data/' --unzip

         # -p denotes download path
             # download the dataset as a single zip file to './data'
         # --unzip
             # extract the files ['demographic.csv', 'diet.csv', 'examination.csv',
         'labs.csv', 'medications.csv', 'questionnaire.csv']
```

Warning: Your Kaggle API key is readable by other users on this system! To fix this, you can run 'chmod 600 /Users/macbookpro/.kaggle/kaggle.json'
Warning: Your Kaggle API key is readable by other users on this system! To fix this, you can run 'chmod 600 /Users/macbookpro/.kaggle/kaggle.json'
Downloading national-health-and-nutrition-examination-survey.zip to ./data

100%|██| 6.74M/6.74M [00:02<00:00, 3.09MB/s]

Data Profiling and Cleaning

Step 2: Loading the data

```
In [3]:  import pandas as pd
         import numpy as np


         # data_demographic = pd.read_csv('./data/demographic.csv')
         # data_diet = pd.read_csv('./data/diet.csv')
         # data_examination = pd.read_csv('./data/examination.csv')
         # data_labs = pd.read_csv('./data/labs.csv')
         data_medications = pd.read_csv('./data/medications.csv',  encoding = "ISO-
         8859-1", usecols= ['SEQN', 'RXDRSC1', 'RXDRSD1'])
         data_questionnaire = pd.read_csv('./data/questionnaire.csv', index_col='SE
         QN')

         data_questionnaire.head(20)
```

Out[3]:

SEQN	ACD011A	ACD011B	ACD011C	ACD040	ACD110	ALQ101	ALQ110	ALQ120Q	ALQ120U	ALC
73557	1.0	NaN	NaN	NaN	NaN	1.0	NaN	1.0	3.0	
73558	1.0	NaN	NaN	NaN	NaN	1.0	NaN	7.0	1.0	
73559	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN	
73560	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
73561	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN	
73562	NaN	NaN	NaN	4.0	NaN	1.0	NaN	5.0	3.0	
73563	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
73564	1.0	NaN	NaN	NaN	NaN	2.0	1.0	2.0	3.0	
73565	NaN	NaN	NaN	5.0	NaN	NaN	NaN	NaN	NaN	
73566	1.0	NaN	NaN	NaN	NaN	1.0	NaN	1.0	1.0	
73567	1.0	NaN	NaN	NaN	NaN	1.0	NaN	4.0	1.0	
73568	1.0	NaN	NaN	NaN	NaN	1.0	NaN	2.0	1.0	
73569	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
73570	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
73571	1.0	NaN	NaN	NaN	NaN	2.0	1.0	2.0	3.0	
73572	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

73573	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73574	NaN	NaN	NaN	NaN	1.0	2.0	2.0	NaN	NaN
73575	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73576	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

20 rows × 952 columns

In [4]: `data_medications.iloc[:10]`

Out[4]:

	SEQN	RXDRSC1	RXDRSD1
0	73557	NaN	NaN
1	73557	E11	Type 2 diabetes mellitus
2	73558	G25.81	Restless legs syndrome
3	73558	E11	Type 2 diabetes mellitus
4	73558	E11.2	Type 2 diabetes mellitus with kidney complicat...
5	73558	E78.0	Pure hypercholesterolemia
6	73559	E11	Type 2 diabetes mellitus
7	73559	E11	Type 2 diabetes mellitus
8	73559	K86.9	Disease of pancreas, unspecified
9	73559	E78.0	Pure hypercholesterolemia

In [5]: `data_questionnaire.describe()`

`# Note: descriptive analysis on data_questionnaire will be done at a later step, after merging the 2 datasets.`

Out[5]:

	ACD011A	ACD011B	ACD011C	ACD040	ACD110	ALQ101	ALQ110	ALQ12
count	5759.0	16.0	171.0	2374.000000	1007.000000	5421.000000	1631.000000	4479.000
mean	1.0	8.0	9.0	3.101095	2.956306	1.311197	1.594727	4.709
std	0.0	0.0	0.0	1.511821	1.733794	0.545023	0.615303	34.428
min	1.0	8.0	9.0	1.000000	1.000000	1.000000	1.000000	0.000
25%	1.0	8.0	9.0	2.000000	1.000000	1.000000	1.000000	1.000
50%	1.0	8.0	9.0	3.000000	3.000000	1.000000	2.000000	2.000
75%	1.0	8.0	9.0	4.000000	5.000000	2.000000	2.000000	4.000
max	1.0	8.0	9.0	9.000000	5.000000	9.000000	9.000000	999.000

8 rows × 950 columns

Descriptive statistical analysis is irrelevant for data_medications, since we are only dealing with 'SEQN', which is the respondent sequence number.

In [6]:
```python
data_medications.describe()
```

Out[6]:

	SEQN
count	20194.000000
mean	78545.350946
std	2933.000334
min	73557.000000
25%	75984.250000
50%	78506.000000
75%	81063.000000
max	83731.000000

Defining Questions

In [7]:
```python
# Medication grouped

# rows_with_med = data_medications[(data_medications.RXDCOUNT >= 1)]
# grouped_rows_with_med = rows_with_med.groupby(['SEQN'])
# grouped_rows_with_med['RXDCOUNT'].count()

# grouped_medication = data_medications.groupby(['RXDDRUG'])
# grouped_medication['SEQN'].count().sort_values(ascending=False)
```

In [8]:
```python
pd.set_option('display.max_rows',500)
pd.set_option('display.max_columns',500)


# List of the most frequent medical diagnoses
# Hypertension is the most frequently seen condition in this dataset.
# Therefore, this project will revolve around this topic

patients_conditions = data_medications.drop_duplicates(subset=['SEQN', 'RX
DRSC1'], keep='first').copy()

grouped_conditions = patients_conditions.groupby(['RXDRSC1', 'RXDRSD1'])
print(grouped_conditions['SEQN'].count().sort_values(ascending=False).head
(25))
```

```
RXDRSC1  RXDRSD1
I10      Essential (primary) hypertension                      1595
E78.0    Pure hypercholesterolemia                             1101
E11      Type 2 diabetes mellitus                    575
K21      Gastro-esophageal reflux disease                434
F32.9    Major depressive disorder, single episode, unspecified    399
J45      Asthma                                348
```

```
F41.9    Anxiety disorder, unspecified              346
E03.9    Hypothyroidism, unspecified                334
M54.9    Dorsalgia, unspecified                     214
G47.0    Insomnia                              198
J30.9    Allergic rhinitis, unspecified             184
K30      Functional dyspepsia                  174
F90      Attention-deficit hyperactivity disorders      166
T78.40   Allergy, unspecified                  150
R60.9    Edema, unspecified                    137
M79.2    Neuralgia and neuritis, unspecified        127
I21.P    Prevent heart attack/myocardial infarction     118
D75.9P   Prevent blood clots                   118
Z79.3    Long term (current) use of hormonal contraceptives    115
G47.9    Sleep disorder, unspecified                110
E87.6    Hypokalemia                      109
N40      Enlarged prostate                     94
M79.1    Myalgia                          92
I49.9    Cardiac arrhythmia, unspecified            91
G43      Migraine                         85
Name: SEQN, dtype: int64
```

Datasets and indicators that will be used in this project:

data_medications

```
   * SEQN: Respondent sequence number
   * RXDRSC1:  ICD-10-CM code 1
   * RXDRSD1: ICD-10-CM code 1 description
```

data_questionnaire

```
   * All indicators
```

QUESTIONS

Q1: What are the determinants of hypertension? (feature selection)

Q2: Can we identify individuals with hypertension from their general health status? (classification)

Q3: Which hypertensive patients are at higher risk for complications? (classification)

Q4: What is the age of onset of complications for those at higher risk? (regression)

Step 2: Basic Statistics (Profiling)

```
In [9]: print ('Number of participants in data_questionnaire: {}'.format(data_ques
        tionnaire.shape[0]))
```

Number of participants in data_questionnaire: 10175

```
In [10]: print ('Number of rows in data_medications: {}'.format(data_medications.sh
```

```
ape[0]))
```

Number of rows in data_medications: 20194

```
In [11]: print ('Number of variables in data_questionnaire: {}'.format(data_questio
         nnaire.shape[1]))
```

Number of variables in data_questionnaire: 952

Step 3: Dealing with duplicates (Cleaning)

data_questionnaire

```
In [12]: # No duplicate participants in data_questionnaire

         nb_duplicates = sum(data_questionnaire.duplicated(keep=False))
         print('Number of duplicate participants in data_questionnaire: ' + str(nb_
         duplicates))
```

Number of duplicate participants in data_questionnaire: 0

data_medications

This is a long-format table with duplicate participant rows, where each rows represents a unique combination of ['SEQN', 'RXDDRUG'] columns (=> a prescribed drug to a patient)

There will certainly be duplicates 'SEQN', or duplicates 'RXDRSC1'(if a patient takes more than one medication for the same diagnosed condition)

```
In [13]: print ('Number of rows in data_medications: {}'.format(data_medications.sh
         ape[0]))

         nb_duplicates = sum(data_medications.duplicated(subset=['SEQN'], keep=Fals
         e))
         print('Number of duplicate participants in data_medications: ' + str(nb_du
         plicates))
```

Number of rows in data_medications: 20194
Number of duplicate participants in data_medications: 12777

- We have to filter participants based on hypertension diagnosis (ICD10 code: 'I10')

```
In [14]: print ('Number of rows in data_medications: {}'.format(data_medications.sh
         ape[0]))

         unique_participants = sum(data_medications.duplicated(subset=['SEQN'], kee
         p=False))
         print('Number of unique participants in data_medications: ' + str(unique_p
         articipants))
```

Number of rows in data_medications: 20194

Number of unique participants in data_medications: 12777

In [15]:
```python
rows_with_hypertension = data_medications[(data_medications.RXDRSC1=='I10'
)]

nb_rows_with_hypertension = len(rows_with_hypertension)
print('Number of rows with hypertension in data_medications: ' + str(nb_ro
ws_with_hypertension))


nb_duplicate_patients_hypertension = sum(rows_with_hypertension.duplicated
(subset=['SEQN'], keep=False))
print('Number of duplicate patients with hypertension in data_medications:
 ' + str(nb_duplicate_patients_hypertension))
```

Number of rows with hypertension in data_medications: 2421
Number of duplicate patients with hypertension in data_medications: 1431

In [16]:
```python
# Remove duplicate patient entries
unique_patients = rows_with_hypertension.drop_duplicates(subset=['SEQN'],
keep='first').copy()
# set 'SEQN' as table index
unique_patients = unique_patients.set_index('SEQN')

nb_unique_patients = len(unique_patients)
print('Number of unique patients with hypertension: ' + str(nb_unique_pati
ents))
```

Number of unique patients with hypertension: 1595

In [ ]:

Data Matching & Merging

In [61]:
```python
data = pd.merge(data_questionnaire, unique_patients, how='inner', left_ind
ex=True, right_index=True)

# Preview of data
data.head(10)
```

Out[61]:

	ACD011A	ACD011B	ACD011C	ACD040	ACD110	ALQ101	ALQ110	ALQ120Q	ALQ120U	AL(
SEQN										
73559	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN	
73561	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN	
73562	NaN	NaN	NaN	4.0	NaN	1.0	NaN	5.0	3.0	

73613	1.0	NaN	NaN	NaN	NaN	1.0	NaN	2.0	1.0
73626	1.0	NaN	NaN	NaN	NaN	2.0	1.0	0.0	NaN
73633	1.0	NaN	NaN	NaN	NaN	2.0	2.0	NaN	NaN
73638	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN
73640	1.0	NaN	NaN	NaN	NaN	2.0	1.0	0.0	NaN
73641	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73643	NaN	NaN	NaN	NaN	1.0	1.0	NaN	3.0	3.0

10 rows × 954 columns

Basic Stats

```
In [62]: rows_count = data.shape[0] # Number of rows

columns_count = data.shape[1] # Number of columns

rows_count
```

Out[62]: 1595

Cleaning

There are no rows with all NaN column values.

```
In [63]: # Drops rows where all column values are NaN
data.dropna(axis=0, how='all')
data.shape
```

Out[63]: (1595, 954)

Replacing missing values

```
In [68]: dont_know_values = [99.0, 999.0]
refused_to_answer_values = [77.0, 777.0]
```

```
values_to_replace = dont_know_values + refused_to_answer_values

for value in values_to_replace:
    data.replace(float(value), np.nan, inplace=True)

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

data.head(50)
```

Out[68]:

SEQN	ACD011A	ACD011B	ACD011C	ACD040	ACD110	ALQ101	ALQ110	ALQ120Q	ALQ120U
73559	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN
73561	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN
73562	NaN	NaN	NaN	4.0	NaN	1.0	NaN	5.0	3.0
73613	1.0	NaN	NaN	NaN	NaN	1.0	NaN	2.0	1.0
73626	1.0	NaN	NaN	NaN	NaN	2.0	1.0	0.0	NaN
73633	1.0	NaN	NaN	NaN	NaN	2.0	2.0	NaN	NaN
73638	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN
73640	1.0	NaN	NaN	NaN	NaN	2.0	1.0	0.0	NaN
73641	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73643	NaN	NaN	NaN	NaN	1.0	1.0	NaN	3.0	3.0
73645	1.0	NaN	NaN	NaN	NaN	1.0	NaN	10.0	3.0
73647	NaN	NaN	NaN	4.0	NaN	1.0	NaN	0.0	NaN

73652	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73654	NaN	NaN	9.0	NaN	NaN	2.0	2.0	NaN	NaN
73659	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN
73662	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73666	NaN	NaN	NaN	NaN	5.0	1.0	NaN	0.0	NaN
73672	1.0	NaN	NaN	NaN	NaN	1.0	NaN	7.0	1.0
73677	1.0	NaN	NaN	NaN	NaN	2.0	1.0	0.0	NaN
73689	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73690	1.0	NaN	NaN	NaN	NaN	2.0	1.0	0.0	NaN
73694	1.0	NaN	NaN	NaN	NaN	2.0	1.0	3.0	3.0
73695	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73702	1.0	NaN	NaN	NaN	NaN	1.0	NaN	4.0	3.0
73706	1.0	NaN	NaN	NaN	NaN	1.0	NaN	2.0	2.0
73708	NaN	NaN	NaN	5.0	NaN	1.0	NaN	5.0	1.0
73709	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
73717	1.0	NaN	NaN	NaN	NaN	2.0	2.0	NaN	NaN

73718	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN
73720	1.0	NaN	NaN	NaN	NaN	1.0	NaN	1.0	2.0
73722	1.0	NaN	NaN	NaN	NaN	1.0	NaN	7.0	1.0
73725	1.0	NaN	NaN	NaN	NaN	2.0	2.0	NaN	NaN
73730	NaN	NaN	NaN	4.0	NaN	1.0	NaN	5.0	3.0
73740	NaN	NaN	NaN	NaN	2.0	2.0	2.0	NaN	NaN
73741	NaN	NaN	NaN	1.0	NaN	2.0	2.0	NaN	NaN
73747	NaN	NaN	NaN	5.0	NaN	2.0	2.0	NaN	NaN
73774	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN
73778	NaN	NaN	NaN	2.0	NaN	2.0	1.0	0.0	NaN
73782	1.0	NaN	NaN	NaN	NaN	1.0	NaN	2.0	3.0
73787	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN
73792	1.0	NaN	NaN	NaN	NaN	1.0	NaN	3.0	1.0
73797	1.0	NaN	NaN	NaN	NaN	2.0	1.0	0.0	NaN
73800	1.0	NaN	NaN	NaN	NaN	1.0	NaN	0.0	NaN
73811	1.0	NaN	NaN	NaN	NaN	1.0	NaN	7.0	1.0

73816	NaN	NaN	NaN	1.0	NaN	1.0	NaN	3.0	2.0
73820	1.0	NaN	NaN	NaN	NaN	1.0	NaN	1.0	2.0
73825	NaN	NaN	NaN	NaN	1.0	2.0	2.0	NaN	NaN
73831	1.0	NaN	NaN	NaN	NaN	1.0	NaN	1.0	1.0
73840	1.0	NaN	NaN	NaN	NaN	1.0	NaN	4.0	1.0
73845	1.0	NaN	NaN	NaN	NaN	2.0	1.0	0.0	NaN

50 rows × 954 columns

Descriptive Analysis

The count, for each column, denotes how many patients have answered this column.

There are many columns for which the number of patients (n) is too small to draw conclusions.

In [69]: `data.describe()`

Out[69]:

	ACD011A	ACD011B	ACD011C	ACD040	ACD110	ALQ101	ALQ110	ALQ120Q
count	1179.0	2.0	24.0	269.000000	130.000000	1430.000000	461.000000	1192.00000
mean	1.0	8.0	9.0	2.776952	2.476923	1.341958	1.557484	3.12500
std	0.0	0.0	0.0	1.612240	1.699135	0.618007	0.782591	11.07339
min	1.0	8.0	9.0	1.000000	1.000000	1.000000	1.000000	0.00000
25%	1.0	8.0	9.0	1.000000	1.000000	1.000000	1.000000	0.00000
50%	1.0	8.0	9.0	3.000000	2.000000	1.000000	2.000000	1.00000
75%	1.0	8.0	9.0	4.000000	4.750000	2.000000	2.000000	3.00000
max	1.0	8.0	9.0	5.000000	5.000000	9.000000	9.000000	305.00000

8 rows × 950 columns

Count

In [70]: `nb_samples_required = 310`

```
# based on https://select-statistics.co.uk/calculators/sample-size-calcula
tor-population-proportion/
# confidence=0.95, n=1595

print('The number of attributes went')
print('from ' + str(columns_count))
data = data.loc[:, data.count() >= nb_samples_required]

print('to ' + str(data.shape[1]))
```

The number of attributes went
from 954
to 399

In [71]:
```
# Preview of data table with only statistically significant columns
data.head(10)
```

Out[71]:

SEQN	ACD011A	ALQ101	ALQ110	ALQ120Q	ALQ120U	ALQ130	ALQ141Q	ALQ151	ALQ160	BPQ0
73559	1.0	1.0	NaN	0.0	NaN	NaN	NaN	2.0	NaN	
73561	1.0	1.0	NaN	0.0	NaN	NaN	NaN	2.0	NaN	
73562	NaN	1.0	NaN	5.0	3.0	1.0	0.0	2.0	0.0	
73613	1.0	1.0	NaN	2.0	1.0	1.0	0.0	2.0	NaN	
73626	1.0	2.0	1.0	0.0	NaN	NaN	NaN	2.0	NaN	
73633	1.0	2.0	2.0	NaN	NaN	NaN	NaN	NaN	NaN	
73638	1.0	1.0	NaN	0.0	NaN	NaN	NaN	2.0	0.0	
73640	1.0	2.0	1.0	0.0	NaN	NaN	NaN	2.0	NaN	
73641	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
73643	NaN	1.0	NaN	3.0	3.0	1.0	0.0	2.0	NaN	

project_preliminary_report

In [ ]:

In [ ]: