Development of Computable Phenotyping for congestive heart failure and utilization of ML clustering method for analysis

By Mansi, Soundarya, Ghata, Namita
CHIP 690 BHMI

Introduction

- Develop Computable Phenotype for congestive heart failure patient
- Use of OMOP data model and de-identify MIMIC database
- Detect outlier with probability density statistical method
- Implement machine learning clustering method to analyze comorbidity among clusters
- Implement the Logistic regression ML method to predict hypertension occurrence in Congestive heart failure patients based on other features in datasets.

Methods -Results

Cohort Definition

- Includes all patients with diagnosis of Congestive Heart Failure

- Includes all patients with age 50 years or olders

- Patients must have 2 or more visits on records

- Patient must have systolic and diastolic bp values on records(can not be null)

Other variables for analysis:

- Created pathway1 : patients who takes lisinopril as ingredient noted as 1, otherwise 0

- Gathered patient's age, gender, maximum weight(in Kg)

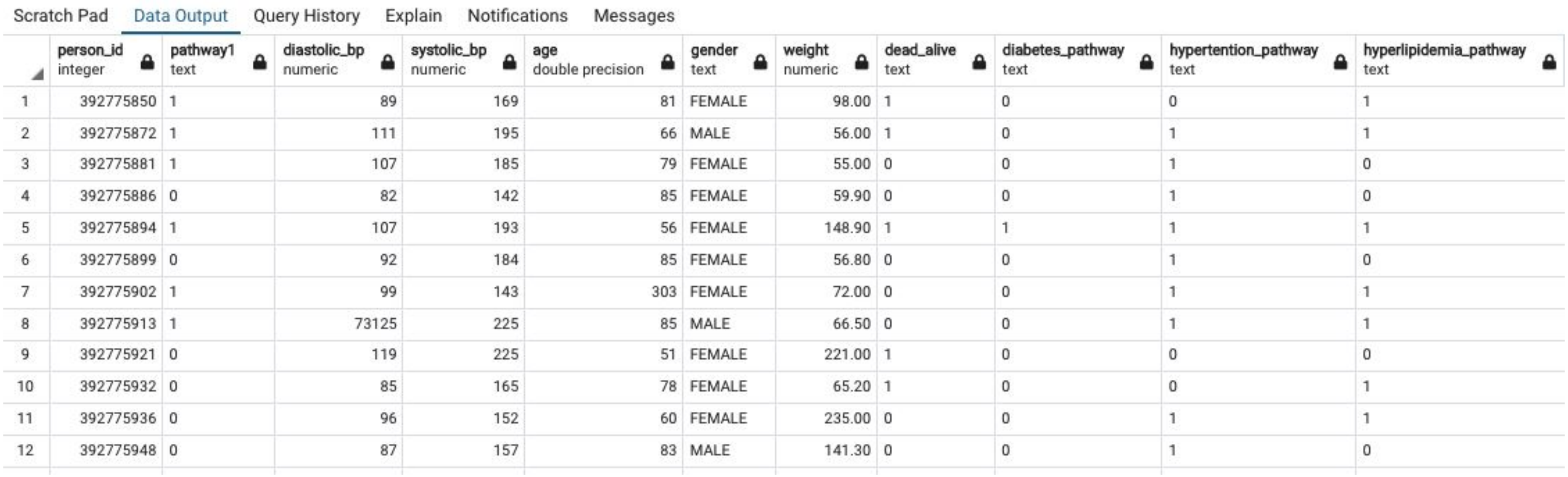
- Created pathway alive_dead:: Patient who is alive denoted as 1, dead denoted as 0

- Created diabetes pathway: Patient who has diabetes as condition denoted as 1, otherwise 0

- Created hypertension pathway: Patient who has hypertension denoted as 1, otherwise 0

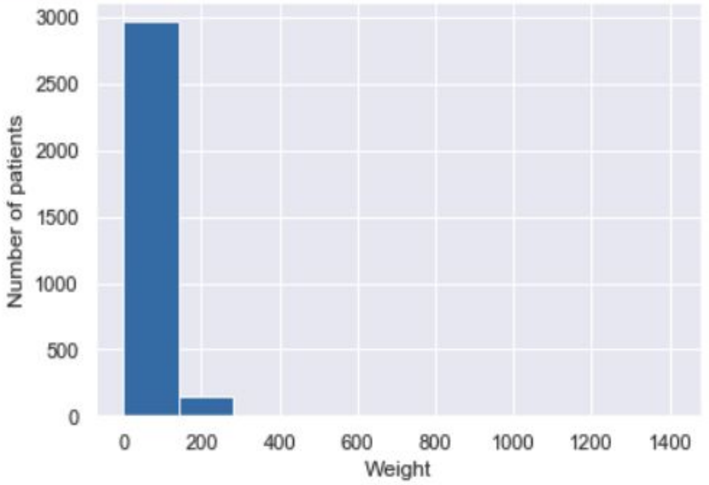
- Created hyperlipidemia pathway: patient who has hyperlipidemia denoted as 1 , otherwise 0

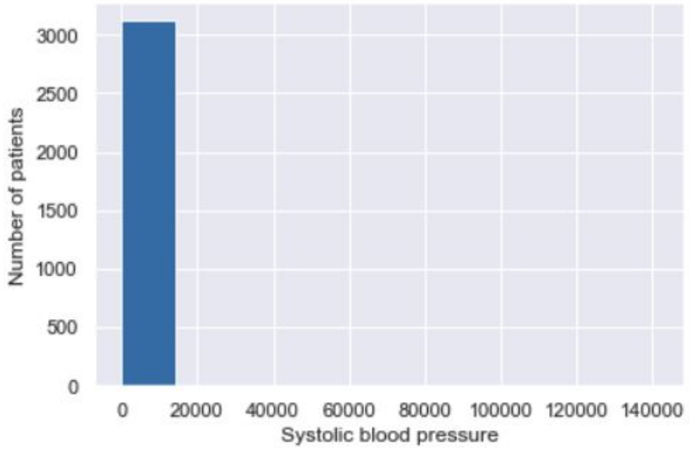
Cohort for analysis

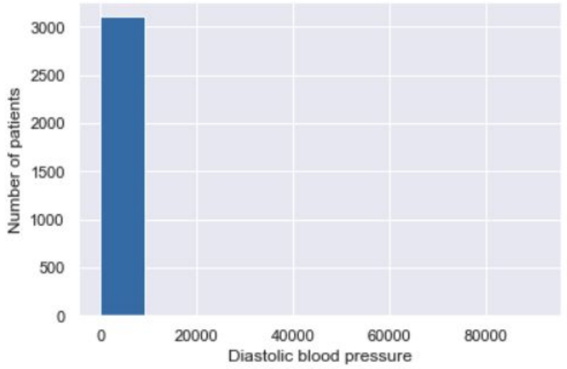


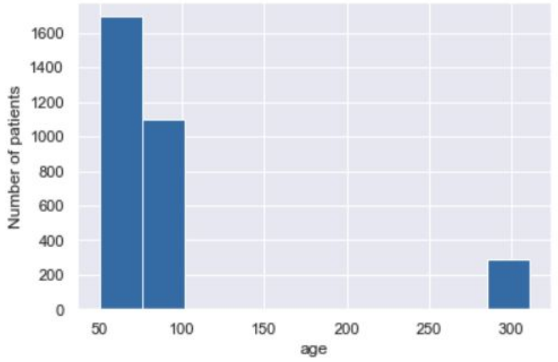
	person_id integer	pathway1 text	diastolic_bp numeric	systolic_bp numeric	age double precision	gender text	weight numeric	dead_alive text	diabetes_pathway text	hypertention_pathway text	hyperlipidemia_pathway text
1	392775850	1	89	169	81	FEMALE	98.00	1	0	0	1
2	392775872	1	111	195	66	MALE	56.00	1	0	1	1
3	392775881	1	107	185	79	FEMALE	55.00	0	0	1	0
4	392775886	0	82	142	85	FEMALE	59.90	0	0	1	0
5	392775894	1	107	193	56	FEMALE	148.90	1	1	1	1
6	392775899	0	92	184	85	FEMALE	56.80	0	0	1	0
7	392775902	1	99	143	303	FEMALE	72.00	0	0	1	1
8	392775913	1	73125	225	85	MALE	66.50	0	0	1	1
9	392775921	0	119	225	51	FEMALE	221.00	1	0	0	0
10	392775932	0	85	165	78	FEMALE	65.20	1	0	0	1
11	392775936	0	96	152	60	FEMALE	235.00	0	0	1	1
12	392775948	0	87	157	83	MALE	141.30	0	0	1	0

Outlier Detection with Exploratory Data analysis and three standard deviation

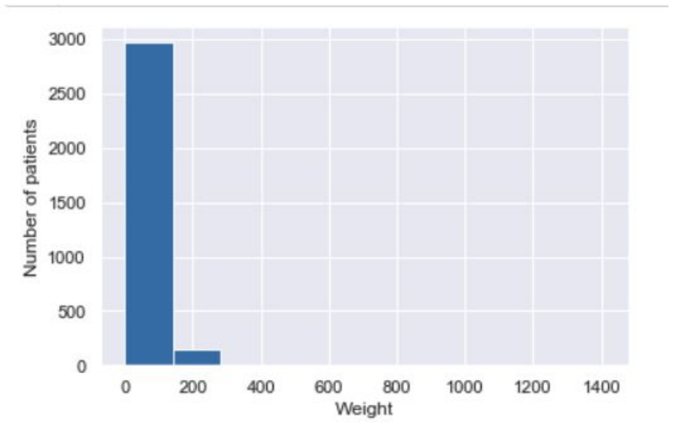


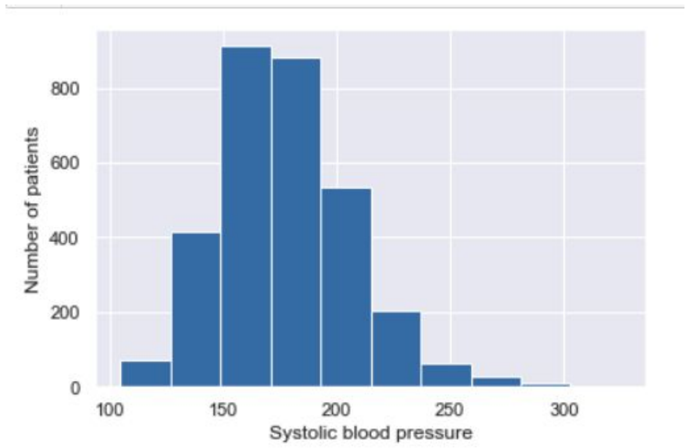


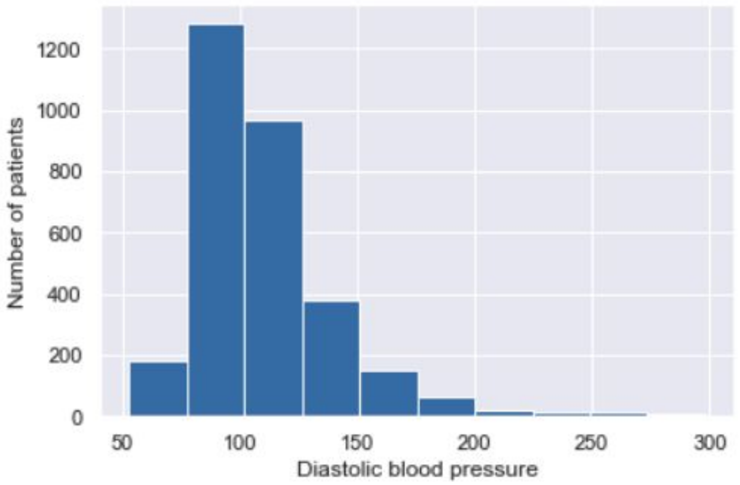


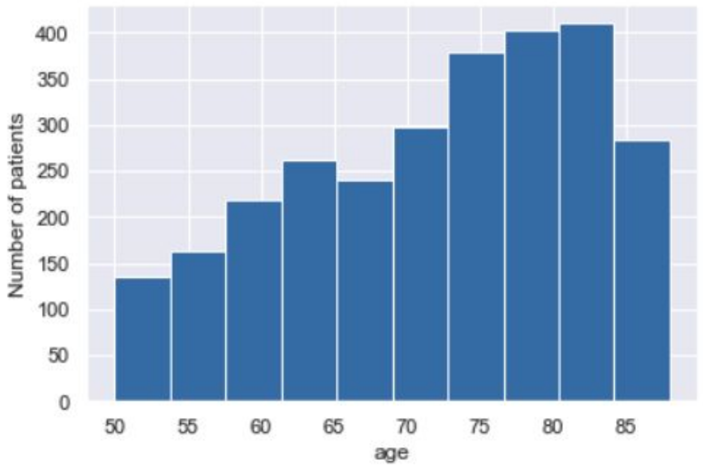


Removing manual error by setting the limits

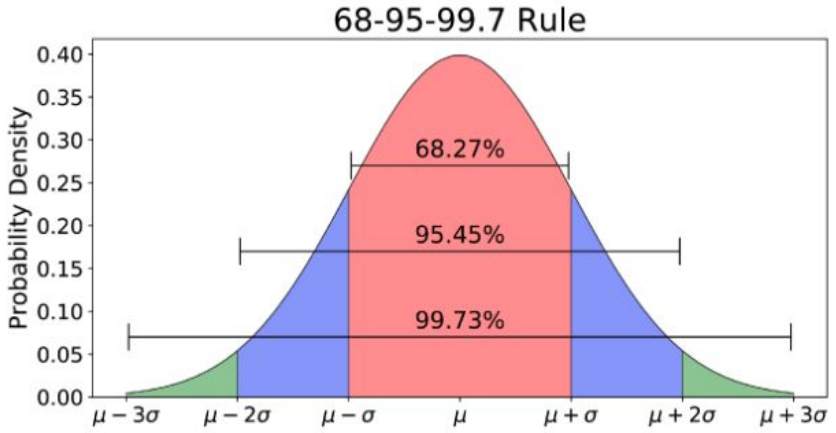




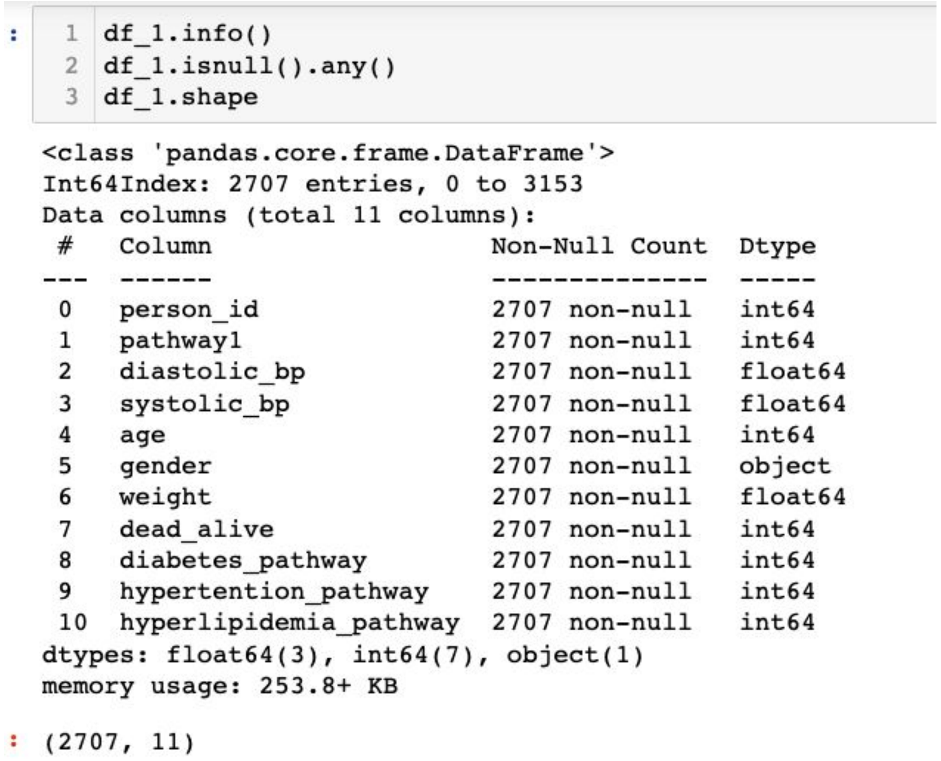




Application of Outlier removal with probability density function



68% of the data is within 1 standard deviation, 95% is within 2 standard deviation, 99.7% is within 3 standard deviations



```
df_1.info()
df_1.isnull().any()
df_1.shape
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2707 entries, 0 to 3153
Data columns (total 11 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   person_id              2707 non-null   int64
 1   pathway1               2707 non-null   int64
 2   diastolic_bp           2707 non-null   float64
 3   systolic_bp            2707 non-null   float64
 4   age                    2707 non-null   int64
 5   gender                 2707 non-null   object
 6   weight                 2707 non-null   float64
 7   dead_alive             2707 non-null   int64
 8   diabetes_pathway       2707 non-null   int64
 9   hypertention_pathway   2707 non-null   int64
 10  hyperlipidemia_pathway 2707 non-null   int64
dtypes: float64(3), int64(7), object(1)
memory usage: 253.8+ KB
```

```
(2707, 11)
```

K-Medoids Clustering Methods

- This method will produce k clusters from a given data set based on similarity among the data points within a cluster.
- Before proceeding with clustering methods, first we scale our dataset. Scaling the data set will reduce the variability among different variables in the dataset.
- Each datapoint acts as medoids and with each iteration of the algorithm, medoids will be a minimal total distance to other members of clusters. We also used the elbow method to determine the optimal number of clusters.
- Even Though we can not visually see the kink in the graph, we determine k=6

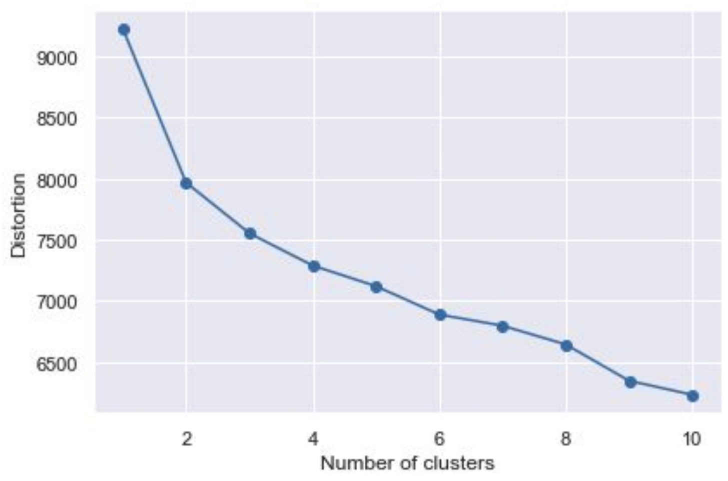
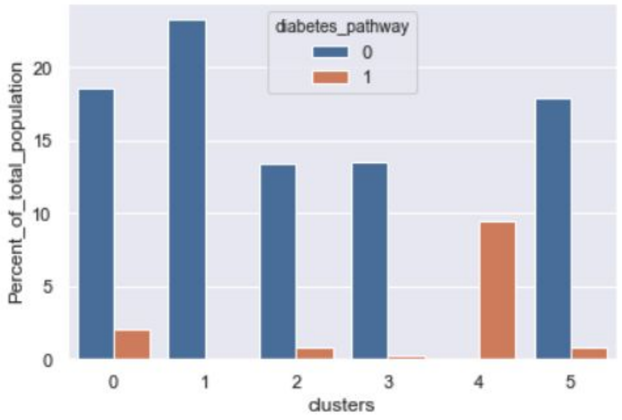
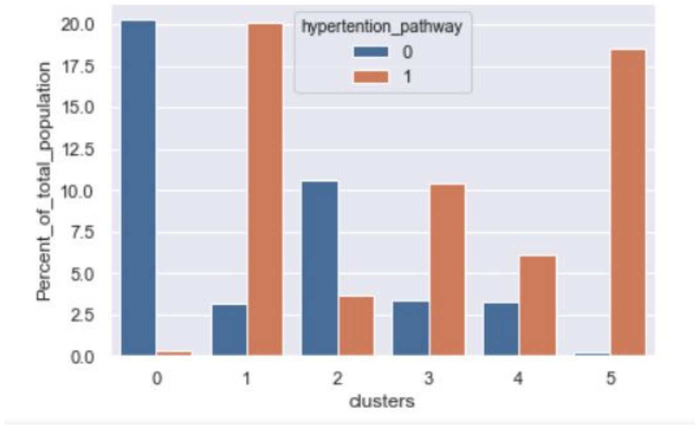


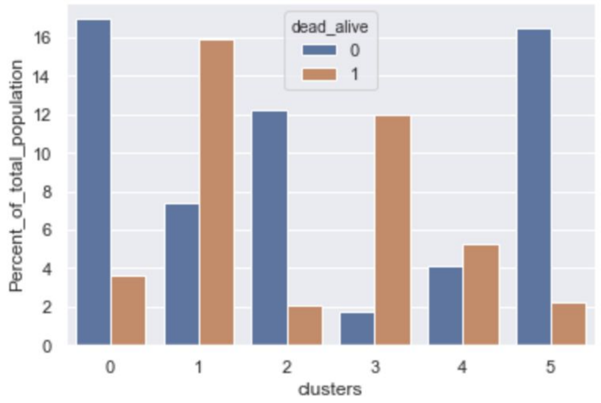
Table # 1 Summary

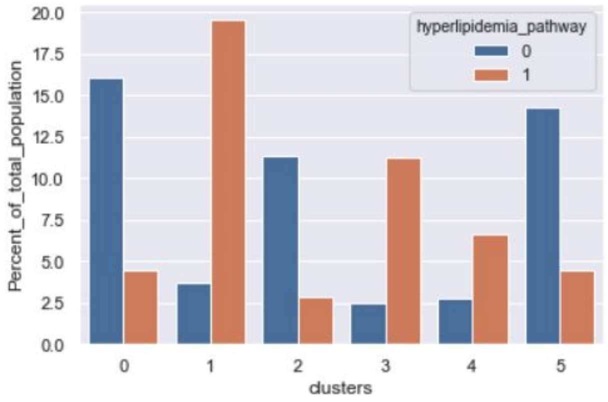
Cluster Characteristic	Cluster 0 n=749	Cluster 1 n=360	Cluster 2 n=567	Cluster 3 n=339	Cluster 4 n=451	Cluster 5 n=241
Sex(n)						
Female	322	178	261	158	197	101
Male	427	182	306	181	254	140
Systolic BP (Mean, SD)	170.48 (26.2)	182.52 (27.5)	176.63 (26.4)	173.24 (24.3)	172.4 (27.3)	182.7 (25.5)
Diastolic BP (Mean,SD)	103.8 (22.8)	117.5 (28.3)	108.5 (25.2)	104.2 (23.1)	108 (24.4)	111 (26.0)
Weight (Mean, SD)	86.1 (25.8)	100.6 (32.6)	86.7 (22.6)	93.6 (25.5)	91.2 (27.3)	92.4 (26.6)
Pathway1(n) (Lisinopril)	155	93	462	0	451	104
Alive (n)	59	249	0	289	447	69
Diabetes as comorbidity (N)	0	11	28	28	51	241
Hypertension as comorbidity (N)	148	277	491	251	349	85
Hyperlipidemia as comorbidity (N)	113	0	417	339	369	96

Comorbidity among the Clusters



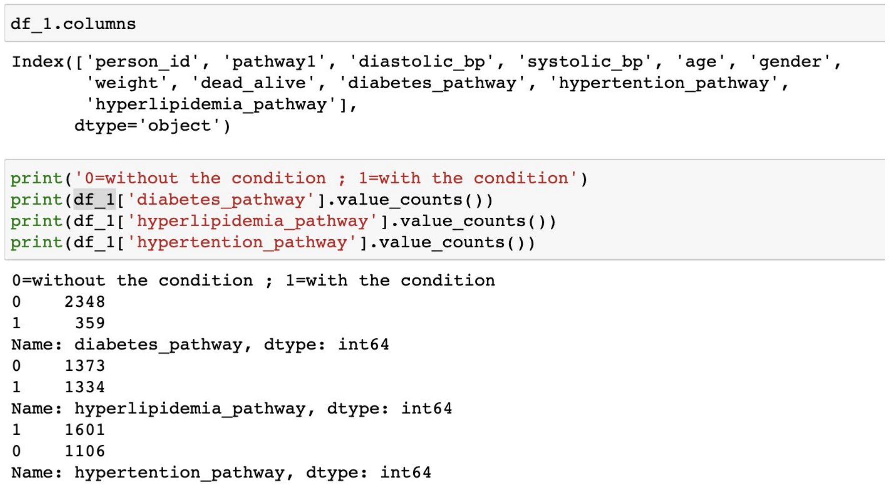






Logistic Regression Model to predict hypertension

Data exploration

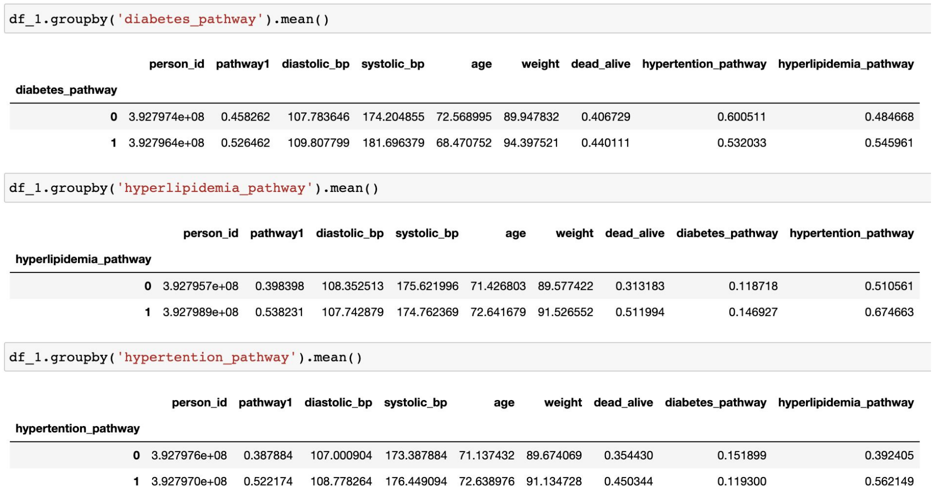


```
df_1.columns
```

```
Index(['person_id', 'pathway1', 'diastolic_bp', 'systolic_bp', 'age', 'gender',
       'weight', 'dead_alive', 'diabetes_pathway', 'hypertention_pathway',
       'hyperlipidemia_pathway'],
      dtype='object')
```

```
print('0=without the condition ; 1=with the condition')
print(df_1['diabetes_pathway'].value_counts())
print(df_1['hyperlipidemia_pathway'].value_counts())
print(df_1['hypertention_pathway'].value_counts())
```

```
0=without the condition ; 1=with the condition
0    2348
1     359
Name: diabetes_pathway, dtype: int64
0    1373
1    1334
Name: hyperlipidemia_pathway, dtype: int64
1    1601
0    1106
Name: hypertention_pathway, dtype: int64
```



```
df_1.groupby('diabetes_pathway').mean()
```

	person_id	pathway1	diastolic_bp	systolic_bp	age	weight	dead_alive	hypertention_pathway	hyperlipidemia_pathway
diabetes_pathway									
0	3.927974e+08	0.458262	107.783646	174.204855	72.568995	89.947832	0.406729	0.600511	0.484668
1	3.927964e+08	0.526462	109.807799	181.696379	68.470752	94.397521	0.440111	0.532033	0.545961

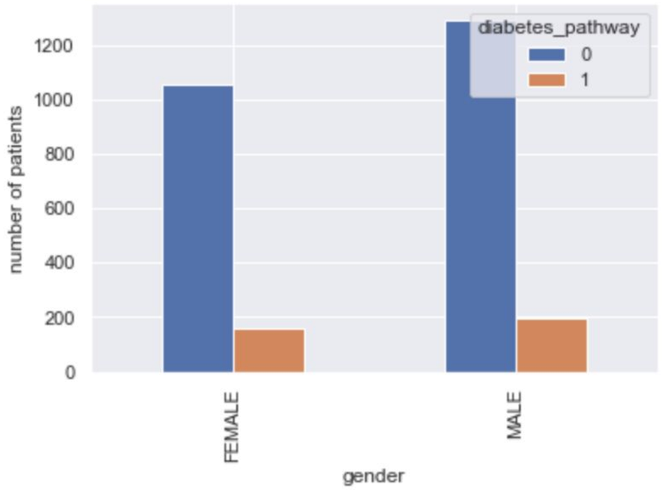
```
df_1.groupby('hyperlipidemia_pathway').mean()
```

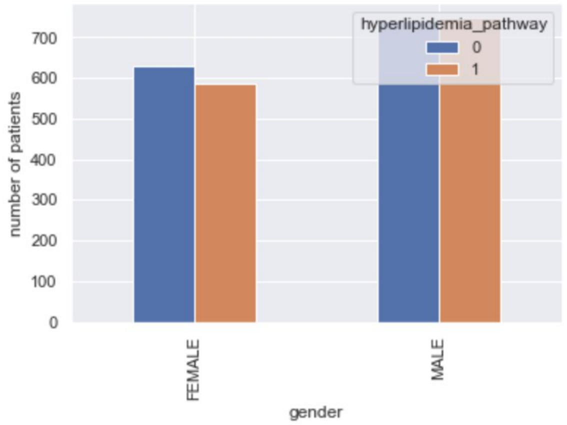
	person_id	pathway1	diastolic_bp	systolic_bp	age	weight	dead_alive	diabetes_pathway	hypertention_pathway
hyperlipidemia_pathway									
0	3.927957e+08	0.398398	108.352513	175.621996	71.426803	89.577422	0.313183	0.118718	0.510561
1	3.927989e+08	0.538231	107.742879	174.762369	72.641679	91.526552	0.511994	0.146927	0.674663

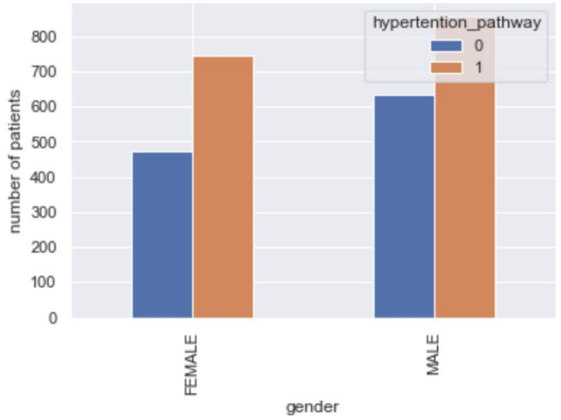
```
df_1.groupby('hypertention_pathway').mean()
```

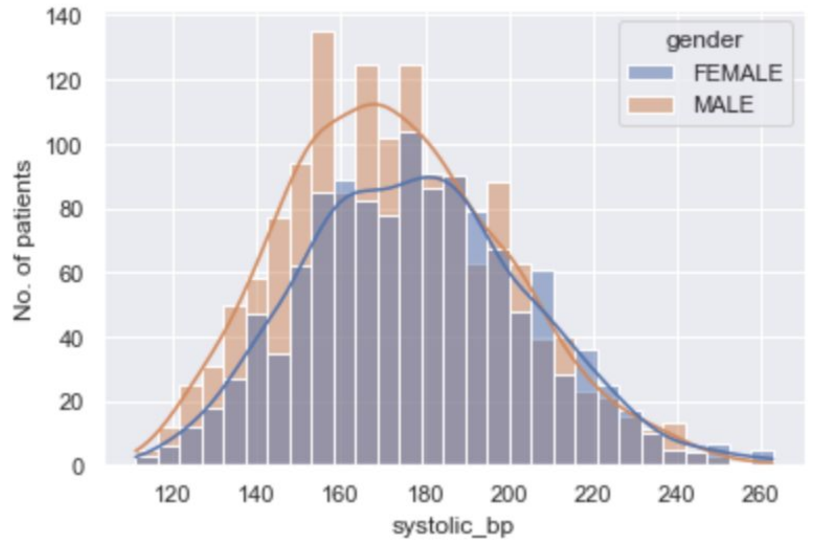
	person_id	pathway1	diastolic_bp	systolic_bp	age	weight	dead_alive	diabetes_pathway	hyperlipidemia_pathway
hypertention_pathway									
0	3.927976e+08	0.387884	107.000904	173.387884	71.137432	89.674069	0.354430	0.151899	0.392405
1	3.927970e+08	0.522174	108.778264	176.449094	72.638976	91.134728	0.450344	0.119300	0.562149

Data Visualization

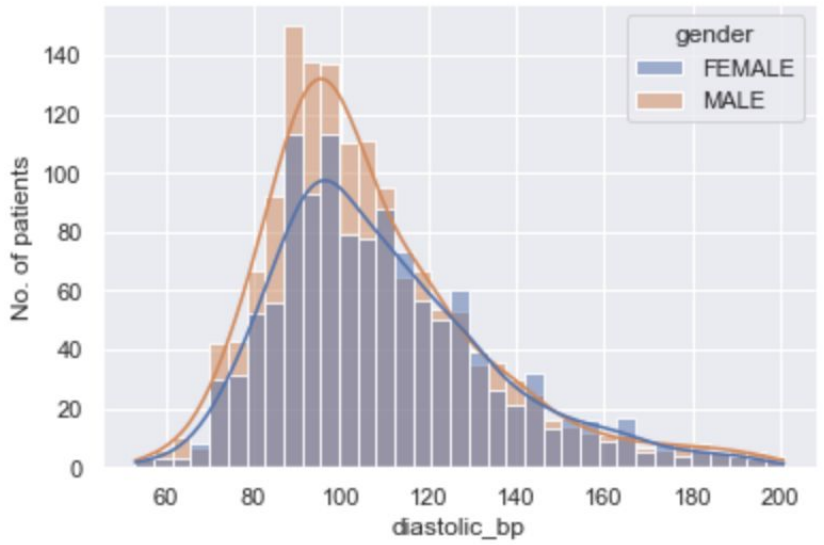






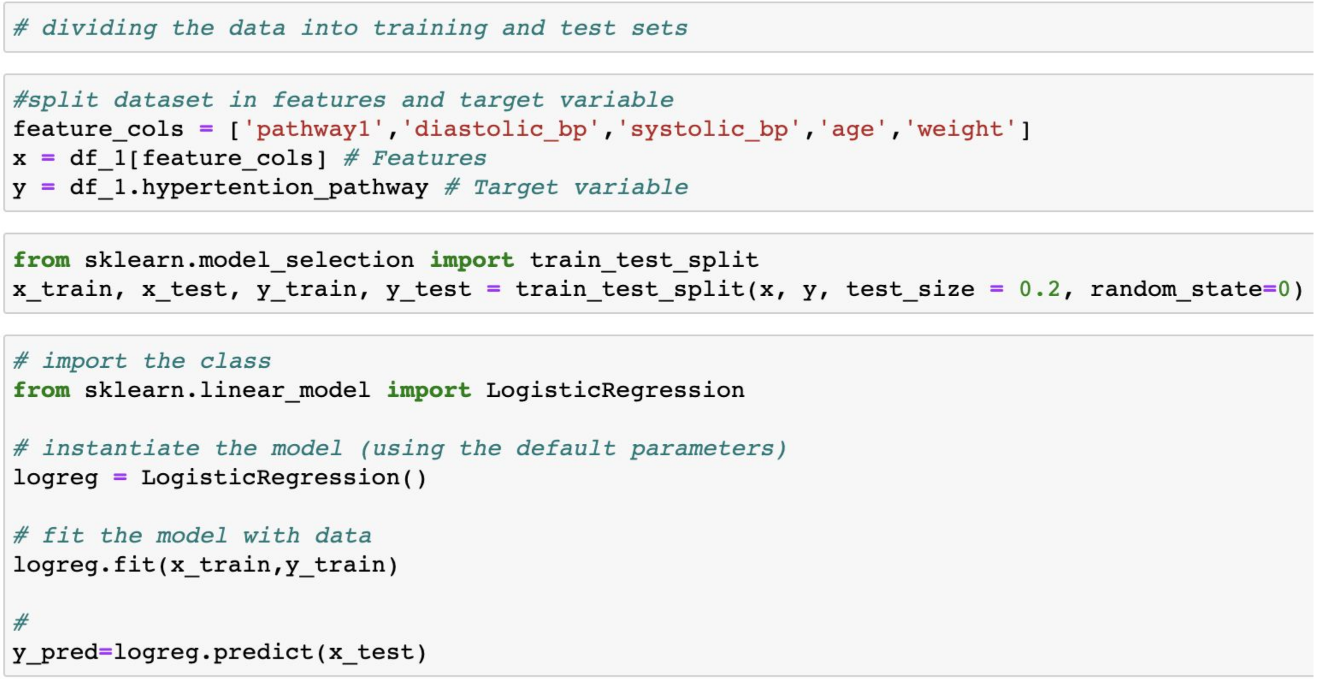


gender
FEMALE
MALE
No. of patients
systolic_bp



gender
FEMALE
MALE
No. of patients
diastolic_bp

Selecting Feature & Splitting Data



```
# dividing the data into training and test sets
```

```
#split dataset in features and target variable
feature_cols = ['pathway1','diastolic_bp','systolic_bp','age','weight']
x = df_1[feature_cols] # Features
y = df_1.hypertention_pathway # Target variable
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state=0)
```

```
# import the class
from sklearn.linear_model import LogisticRegression

# instantiate the model (using the default parameters)
logreg = LogisticRegression()

# fit the model with data
logreg.fit(x_train,y_train)

#
y_pred=logreg.predict(x_test)
```

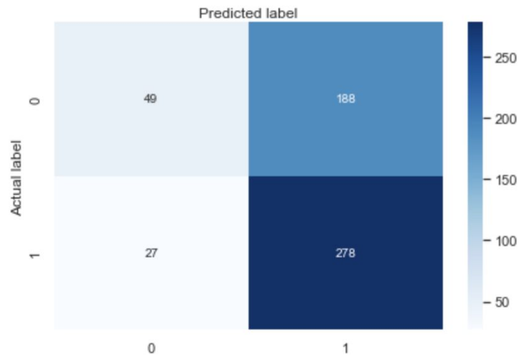
Model Evaluation

```
# import the metrics class
from sklearn import metrics
cnf_matrix = metrics.confusion_matrix(y_test, y_pred)
cnf_matrix
```

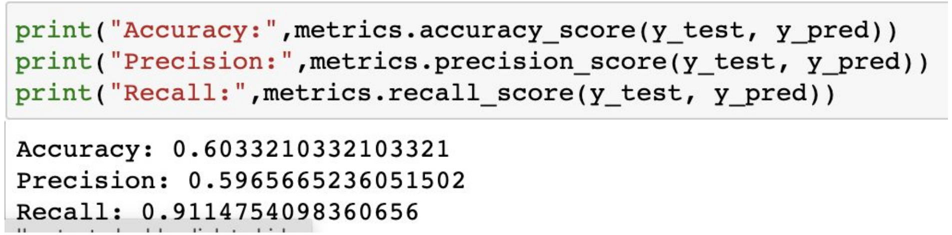
```
array([[ 49, 188],
       [ 27, 278]])
```

```
class_names=[0,1] # name  of classes
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)
# create heatmap
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="Blues" ,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.ylabel('Actual label')
plt.xlabel('Predicted label')
```

```
Text(0.5, 257.44, 'Predicted label')
```

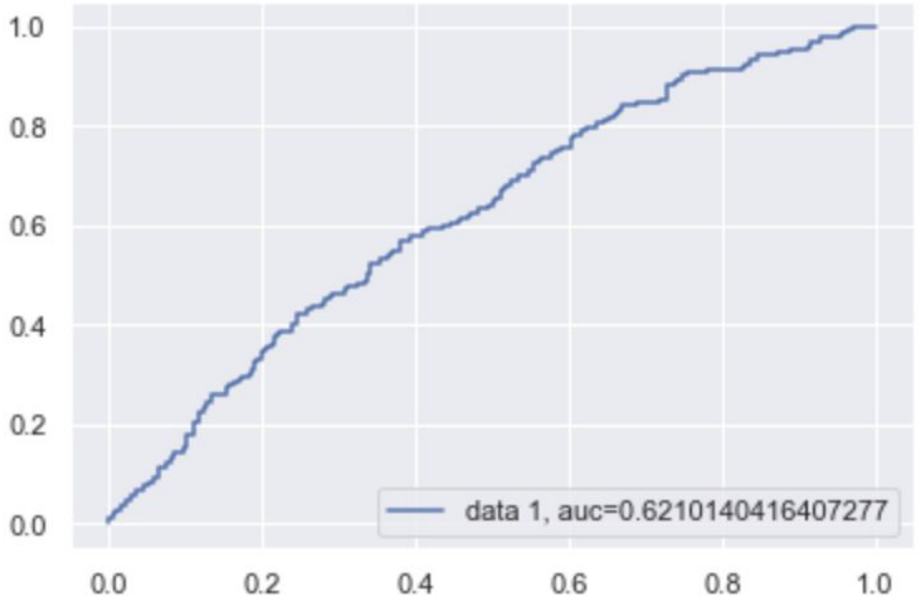


ROC Curve



```
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
print("Precision:",metrics.precision_score(y_test, y_pred))
print("Recall:",metrics.recall_score(y_test, y_pred))
```

```
Accuracy: 0.6033210332103321
Precision: 0.5965665236051502
Recall: 0.9114754098360656
```



Discussion

- In our logistics classification we try to predict the hypertension in patients depending on other factors, after prediction our model has an accuracy of 0.6(good accuracy average is 0.7), indicating that the model is unsuitable for this Dataset. One reason for the low classification accuracy could be that our classes are not well separable using the current features. Finding more features would be a solution to this. Also, we originally intended to use this model to predict CHF in patients, but we only created a dataset for patients with CHF and not for patients without, so including that information could improve the accuracy.
- Our cohort can not be the ideal sample of a population and thus analysis results can not be interpreted for the whole population.
- Our cluster classifications do not show any substantial difference in hypertension or in hyperlipidemia patients.
- The overall occurrence of diabetes with congestive heart failure is relatively less as compared to hypertension and hyperlipidemia.