MCIT 590 Final Project, TEAM 27

- SANTOSH NAZARE
- KEN HUANG
- NICK MORISI

Summary

Our project finds clinically similar patients from a given set of patient population. It calculates the similarity between the *index* patient and all other `n` patients.

The program reads in the data from a CSV file and creates a list of clinical encounters. Using the list of clinical encounters, the software builds a clinical utilization for each patient profile (`HashMap`). The unique patient identifier (`patient_number`) is the key and the patient's clincial profile is stored as value. The patient level clinical profile is created by averaging a patient's inpatient visits, outpatient visits, procedures, medications, lab procedures, and emergency visits. These patient clinical utilization profiles are used to compute *similarity* between patients. We used two methods to calculate the similiarity between patient's prfile, *Euclidean distance* and *Cosine distance*.

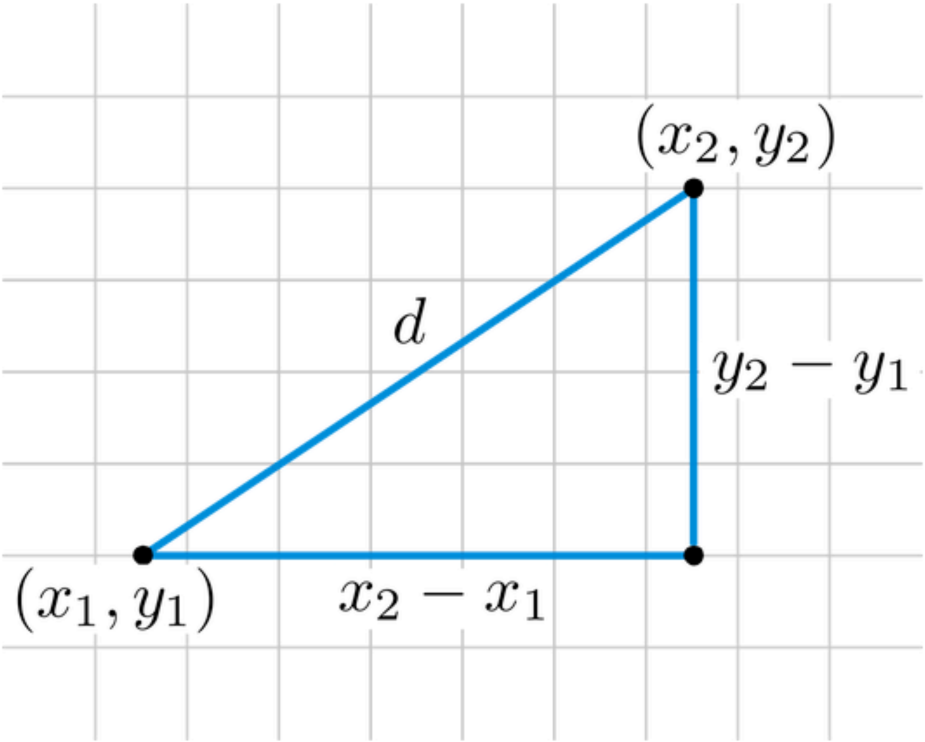
The clinical profile is a vector of patient level measures. We compare the *index* patient’s clinical profile with all other patients’ clinical profiles to calculate similarity metric and return list of patient's which are within user provided threshold of similarity from *index* patient. The distance measure value ranges from 0 (*most similar*) to 1 (*least similar*)

Distance measures

Euclidean distance

Euclidean distance is the distance between two points and is calculated using the Pythagorean Theorem

$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.



(Rosalind 2020).

For the k-dimentional data this formula is generalizd to following form

Given, $p_1 = \{p_{1i}, p_{1j}, \ldots, p_{1k}\}$ and $p_2 = \{p_{2i}, p_{2j}, \ldots, p_{2k}\}$ *educlidean distance* $d(p_1, p_2)$ is

$$d(p_1, p_2) = \sqrt{\sum_{i=i}^{k}(p_{1i} - p_{2i})^2}$$

We used scaled version of *euclidean distance* for our software, to force the range of values between 0 and 1 for easy interpretability.

we calcualted a maximum possible squared discrepency for each variable. If m are the number of patients in the given sample, we caculated maximum squared discrepancy for variable i is given by

$$md_i = (\max(p_{1i}, p_{2i}, p_{3i}, \ldots, p_{mi}) - \min(p_{1i}, p_{2i}, p_{3i}, \ldots, p_{mi}))^2$$

then *scaled euclidean distance* is calculated as,

$$d_s(p1, p2) = \sqrt{\sum_{i=1}^{k}(\frac{(p_{1i} - p_{1i})^2}{md_i})}$$

Cosine distance

The Cosine similarity is a measure of similarity is inner product space of two non-zero vectors, which measures cosine of angle between two vectors.

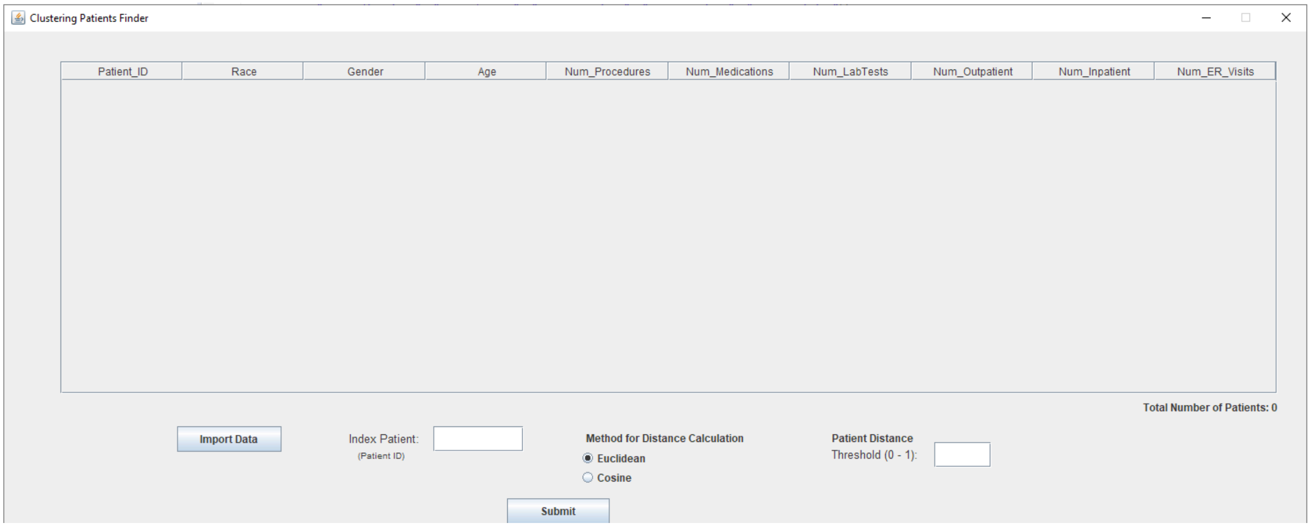
Given, $p_1 = \{p_{1i}, p_{1j}, \ldots, p_{1k}\}$ and $p_2 = \{p_{2i}, p_{2j}, \ldots, p_{2k}\}$, the *cosine* similarity is calculated as,

$$\cos(\theta) = \frac{\sum_{i=1}^{k} p_{1i}p_{2i}}{\sqrt{\sum_{i=1}^{k} p_{1i}^2}\sqrt{\sum_{i=1}^{k} p_{2i}^2}}$$

cosine distance is calcualted as $1 - cos(\theta)$

Project Flow

The user runs the MainFrame.java file and the window below is displayed.



Patient_ID	Race	Gender	Age	Num_Procedures	Num_Medications	Num_LabTests	Num_Outpatient	Num_Inpatient	Num_ER_Visits

Total Number of Patients: 0

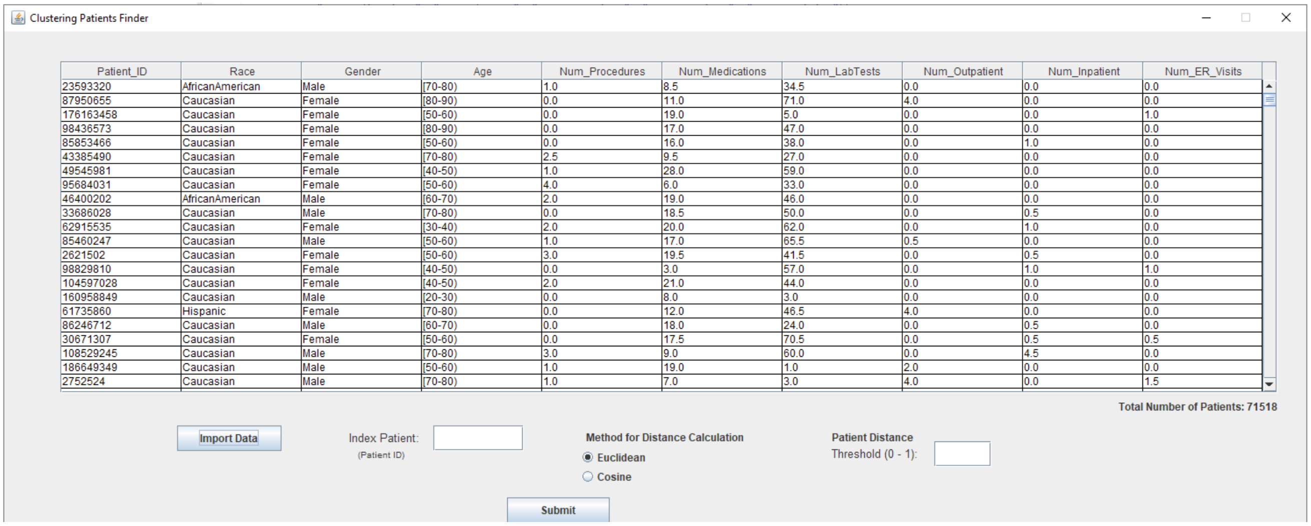
Index Patient: (Patient ID)

Method for Distance Calculation

● Euclidean
○ Cosine

Patient Distance Threshold (0 - 1):

The "Import Data button" allows the user to select the "diabetic_data.csv" provided patient data. The `MainFrame` then executes the `DataReader` class to read data into `ClinicalEncounters` class. Then methods in `PatientProcessor` class are executed to build `HashMap` of `Patient` class. Data from the `Patient` class is displayed in the window as below.



Patient_ID	Race	Gender	Age	Num_Procedures	Num_Medications	Num_LabTests	Num_Outpatient	Num_Inpatient	Num_ER_Visits
23593320	AfricanAmerican	Male	[70-80)	1.0	8.5	34.5	0.0	0.0	0.0
87950655	Caucasian	Female	[80-90)	0.0	11.0	71.0	4.0	0.0	0.0
176163458	Caucasian	Female	[50-60)	0.0	19.0	5.0	0.0	0.0	1.0
98436573	Caucasian	Female	[80-90)	0.0	17.0	47.0	0.0	0.0	0.0
85853466	Caucasian	Female	[50-60)	0.0	16.0	38.0	0.0	1.0	0.0
43385490	Caucasian	Female	[70-80)	2.5	9.5	27.0	0.0	0.0	0.0
49545981	Caucasian	Female	[40-50)	1.0	28.0	59.0	0.0	0.0	0.0
95684031	Caucasian	Female	[50-60)	4.0	6.0	33.0	0.0	0.0	0.0
46400202	AfricanAmerican	Male	[60-70)	2.0	19.0	46.0	0.0	0.0	0.0
33686028	Caucasian	Male	[70-80)	0.0	18.5	50.0	0.0	0.5	0.0
62915535	Caucasian	Female	[30-40)	2.0	20.0	62.0	0.0	1.0	0.0
85460247	Caucasian	Male	[50-60)	1.0	17.0	65.5	0.5	0.0	0.0
2621502	Caucasian	Female	[50-60)	3.0	19.5	41.5	0.0	0.5	0.0
98829810	Caucasian	Female	[40-50)	0.0	3.0	57.0	0.0	1.0	1.0
104597028	Caucasian	Female	[40-50)	2.0	21.0	44.0	0.0	0.0	0.0
160958849	Caucasian	Male	[20-30)	0.0	8.0	3.0	0.0	0.0	0.0
61735860	Hispanic	Female	[70-80)	0.0	12.0	46.5	4.0	0.0	0.0
86246712	Caucasian	Male	[60-70)	0.0	18.0	24.0	0.0	0.5	0.0
30671307	Caucasian	Female	[50-60)	0.0	17.5	70.5	0.0	0.5	0.5
108529245	Caucasian	Male	[70-80)	3.0	9.0	60.0	0.0	4.5	0.0
186649349	Caucasian	Male	[50-60)	1.0	19.0	1.0	2.0	0.0	0.0
2752524	Caucasian	Male	[70-80)	1.0	7.0	3.0	4.0	0.0	1.5

Total Number of Patients: 71518

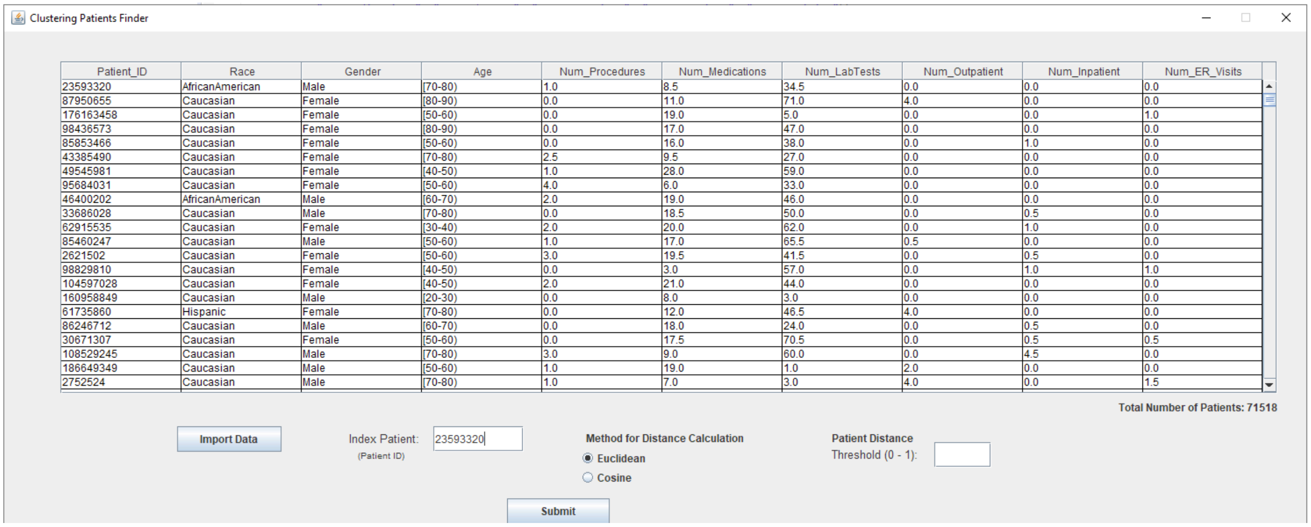
Index Patient: (Patient ID)

Method for Distance Calculation

● Euclidean
○ Cosine

Patient Distance Threshold (0 - 1):

Once the data is imported the user can view and select an index patient for comparison against all other patients.



Clustering Patients Finder

Patient_ID	Race	Gender	Age	Num_Procedures	Num_Medications	Num_LabTests	Num_Outpatient	Num_Inpatient	Num_ER_Visits
23593320	AfricanAmerican	Male	[70-80)	1.0	8.5	34.5	0.0	0.0	0.0
87950655	Caucasian	Female	[80-90)	0.0	11.0	71.0	4.0	0.0	0.0
176163458	Caucasian	Female	[50-60)	0.0	19.0	5.0	0.0	0.0	1.0
98436573	Caucasian	Female	[80-90)	0.0	17.0	47.0	0.0	0.0	0.0
85853466	Caucasian	Female	[50-60)	0.0	16.0	38.0	0.0	1.0	0.0
43385490	Caucasian	Female	[70-80)	2.5	9.5	27.0	0.0	0.0	0.0
49545981	Caucasian	Female	[40-50)	1.0	28.0	59.0	0.0	0.0	0.0
95684031	Caucasian	Female	[50-60)	4.0	6.0	33.0	0.0	0.0	0.0
46400202	AfricanAmerican	Male	[60-70)	2.0	19.0	46.0	0.0	0.0	0.0
33686028	Caucasian	Male	[70-80)	0.0	18.5	50.0	0.0	0.5	0.0
62915535	Caucasian	Female	[30-40)	2.0	20.0	62.0	0.0	1.0	0.0
85460247	Caucasian	Male	[50-60)	1.0	17.0	65.5	0.5	0.0	0.0
2621502	Caucasian	Female	[50-60)	3.0	19.5	41.5	0.0	0.5	0.0
98829810	Caucasian	Female	[40-50)	0.0	3.0	57.0	0.0	1.0	1.0
104597028	Caucasian	Female	[40-50)	2.0	21.0	44.0	0.0	0.0	0.0
160958849	Caucasian	Male	[20-30)	0.0	8.0	3.0	0.0	0.0	0.0
61735860	Hispanic	Female	[70-80)	0.0	12.0	46.5	4.0	0.0	0.0
86246712	Caucasian	Male	[60-70)	0.0	18.0	24.0	0.0	0.5	0.0
30671307	Caucasian	Female	[50-60)	0.0	17.5	70.5	0.0	0.5	0.5
108529245	Caucasian	Male	[70-80)	3.0	9.0	60.0	0.0	4.5	0.0
186649349	Caucasian	Male	[50-60)	1.0	19.0	1.0	2.0	0.0	0.0
2752524	Caucasian	Male	[70-80)	1.0	7.0	3.0	4.0	0.0	1.5

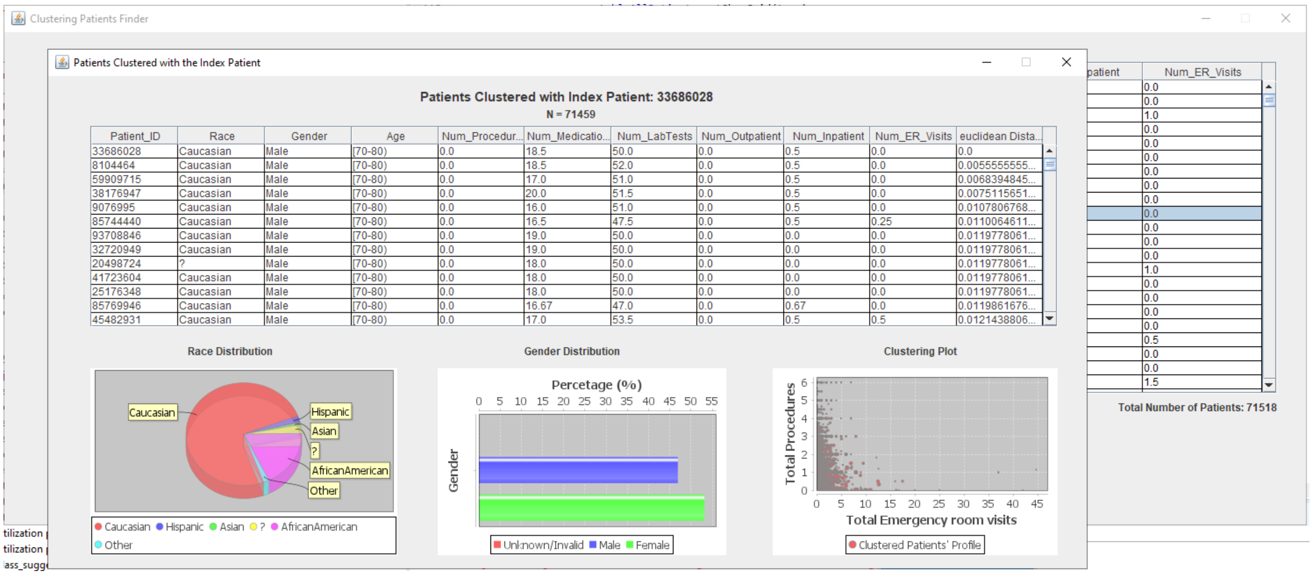
Total Number of Patients: 71518

Index Patient: 23593320
(Patient ID)

Method for Distance Calculation
● Euclidean
○ Cosine

Patient Distance
Threshold (0 - 1):

Once the user has selected the index patient, the user can either copy and paste or type in the patient ID into the Index Patient box. Additionally the user can choose Euclidean or Cosine distance calculations. Finally the user must enter a threshold value of 0 - 1 and then click submit. The MainFrame then executes PatientProcessor and Patient classes to develop profile HashMap. Finally the methods in `DataAnaylsis`, `ClusteringFrame`, `Plots`, `DataDisplay`, `mapValueCompatator` are called to calculate similarity values and display those values into three JCharts seen below.



Clustering Patients Finder

Patients Clustered with the Index Patient

Patients Clustered with Index Patient: 33686028
N = 71459

Patient_ID	Race	Gender	Age	Num_Procedur...	Num_Medicatio...	Num_LabTests	Num_Outpatient	Num_Inpatient	Num_ER_Visits	euclidean Dista...
33686028	Caucasian	Male	[70-80)	0.0	18.5	50.0	0.0	0.5	0.0	0.0
8104464	Caucasian	Male	[70-80)	0.0	18.5	52.0	0.0	0.5	0.0	0.0055555555...
59909715	Caucasian	Male	[70-80)	0.0	17.0	51.0	0.0	0.5	0.0	0.0068394845...
38176947	Caucasian	Male	[70-80)	0.0	20.0	51.5	0.0	0.5	0.0	0.0075115651...
9076995	Caucasian	Male	[70-80)	0.0	16.0	51.0	0.0	0.5	0.0	0.0107806768...
85744440	Caucasian	Male	[70-80)	0.0	16.5	47.5	0.0	0.5	0.25	0.0110064611...
93708846	Caucasian	Male	[70-80)	0.0	19.0	50.0	0.0	0.0	0.0	0.0119778061...
32720949	Caucasian	Male	[70-80)	0.0	19.0	50.0	0.0	0.0	0.0	0.0119778061...
20498724	?	Male	[70-80)	0.0	18.0	50.0	0.0	0.0	0.0	0.0119778061...
41723604	Caucasian	Male	[70-80)	0.0	18.0	50.0	0.0	0.0	0.0	0.0119778061...
25176348	Caucasian	Male	[70-80)	0.0	18.0	50.0	0.0	0.0	0.0	0.0119778061...
85769946	Caucasian	Male	[70-80)	0.0	16.67	47.0	0.0	0.67	0.0	0.0119861676...
45482931	Caucasian	Male	[70-80)	0.0	17.0	53.5	0.0	0.5	0.5	0.0121438806...

Total Number of Patients: 71518

The window then displays Race Distribution of patients, Gender distribution, and the Clustering plot.

Project Setup and Component Installation:

Step 1:

Follow video installing Window Builder SWING:

https://www.youtube.com/watch?v=K7R68JFd024

Alternative:

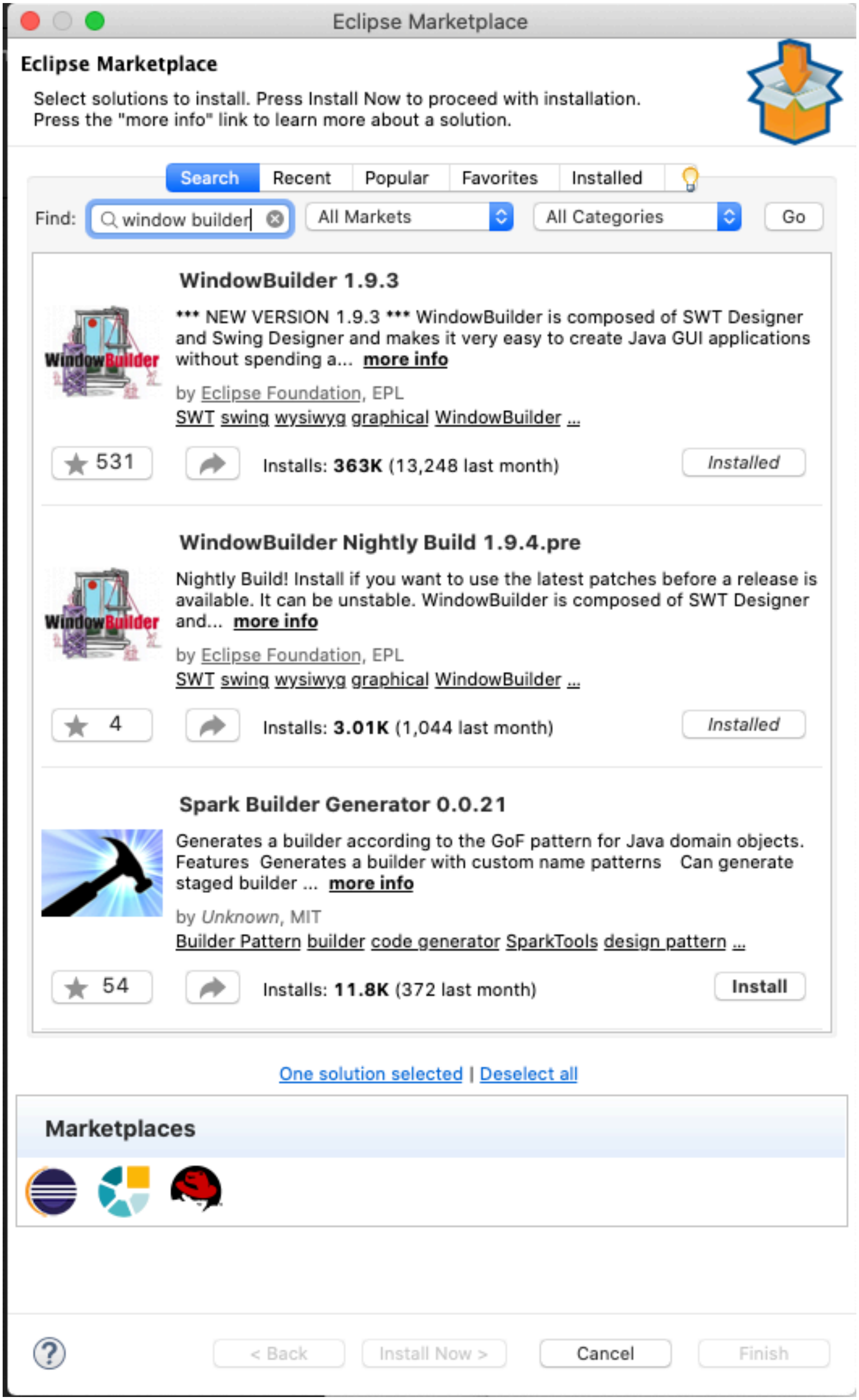
(1) Browse to: https://www.eclipse.org/windowbuilder/

(2) Drag install icon in an open Eclipse window.

(3) If installation does not start automatically the Eclipse Marketplace Window will open:

(4) In search box type “Window Builder”

(5) Follow instructions for installation:



Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Find: window builder | All Markets | All Categories

WindowBuilder 1.9.3

*** NEW VERSION 1.9.3 *** WindowBuilder is composed of SWT Designer and Swing Designer and makes it very easy to create Java GUI applications without spending a... **more info**

by Eclipse Foundation, EPL

SWT swing wysiwyg graphical WindowBuilder ...

★ 531 Installs: **363K** (13,248 last month) *Installed*

WindowBuilder Nightly Build 1.9.4.pre

Nightly Build! Install if you want to use the latest patches before a release is available. It can be unstable. WindowBuilder is composed of SWT Designer and... **more info**

by Eclipse Foundation, EPL

SWT swing wysiwyg graphical WindowBuilder ...

★ 4 Installs: **3.01K** (1,044 last month) *Installed*

Spark Builder Generator 0.0.21

Generates a builder according to the GoF pattern for Java domain objects. Features Generates a builder with custom name patterns Can generate staged builder ... **more info**

by *Unknown*, MIT

Builder Pattern builder code generator SparkTools design pattern ...

★ 54 Installs: **11.8K** (372 last month) **Install**

One solution selected | Deselect all

Step 2:

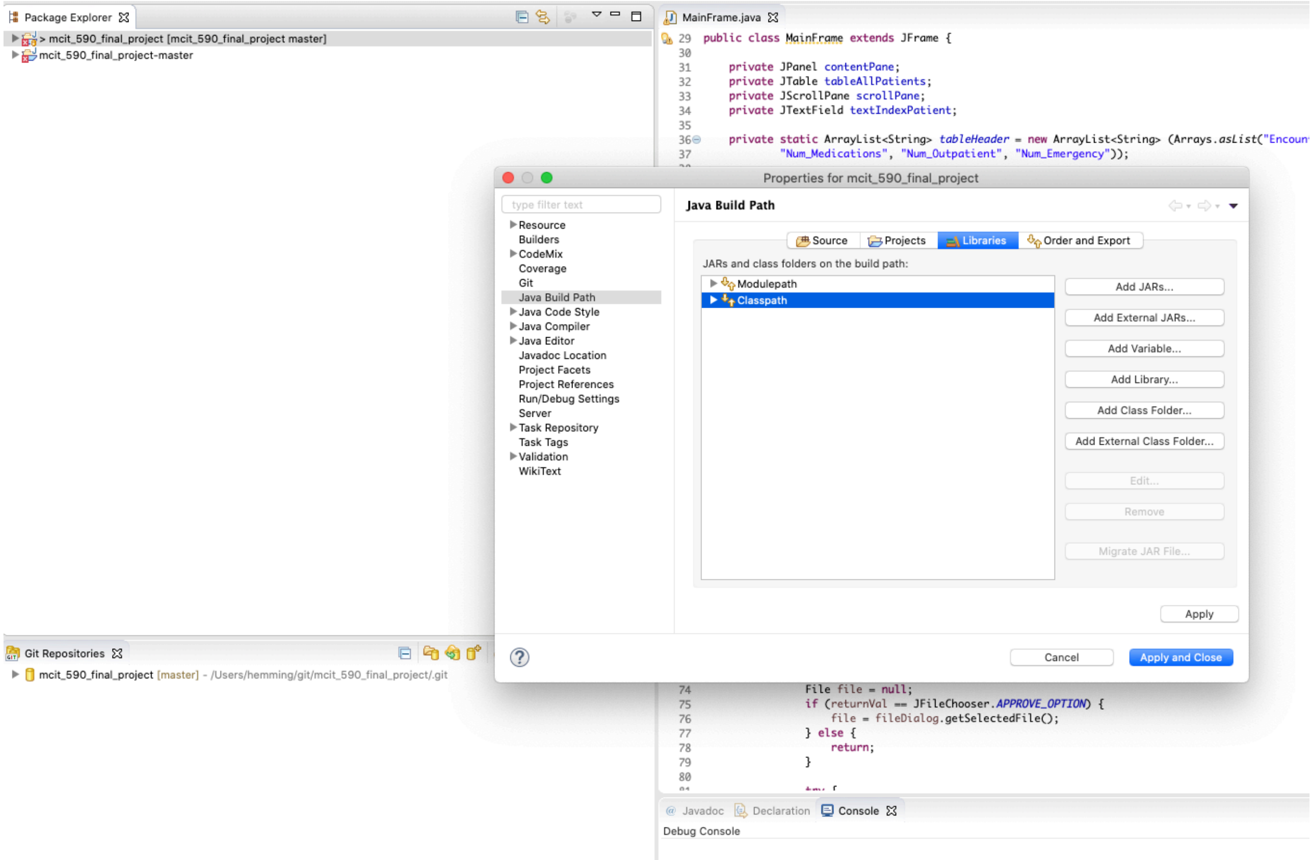
J Common and J Chart Installation

(1) Navigate to jfreechart-1.0.19-install.pdf

(2) Follow instructions on page 31, A3 for Eclipse installation.

(3) Jcommon-1.0.23.zip and jfreechar-1.0.19.zip included for installation.

(4) Once you add J Chart and J Common to your eclipse libraries, you may need to add the Library path into your project. See instructions below.



```
public class MainFrame extends JFrame {

    private JPanel contentPane;
    private JTable tableAllPatients;
    private JScrollPane scrollPane;
    private JTextField textIndexPatient;

    private static ArrayList<String> tableHeader = new ArrayList<String> (Arrays.asList("Encoun
            "Num_Medications", "Num_Outpatient", "Num_Emergency"));
```

```
                File file = null;
                if (returnVal == JFileChooser.APPROVE_OPTION) {
                    file = fileDialog.getSelectedFile();
                } else {
                    return;
                }
```

(1) Right Click project, Select properties

(2) Navigate to build path

(3) Select Libraries

(4) Add Library from Eclipse Libraries, this was competed during Jchart and JCommon installation.

References:

Hu, J., Wang, F., Sun, J., Sorrentino, R., & Ebadollahi, S. (2012). A healthcare utilization analysis framework for hot spotting and contextual anomaly detection. *AMIA ... Annual Symposium proceedings. AMIA Symposium*, *2012*, 360–369.

Rosalind 2020. http://rosalind.info/glossary/euclidean-distance/

Machine Learning Plus 2020. https://www.machinelearningplus.com/nlp/cosine-similarity/

Chris Emmery 2020. https://cmry.github.io/notes/euclidean-v-cosine