**~~Chapter/section summary or major learning points~~**

**Data acquisition**

**Breast Cancer (Diagnostic) Dataset**

**Origin**: From Kaggle and UCI machine learning repository

<https://www.kaggle.com/uciml/breast-cancer-wisconsin-data>

**Reasons:**

* Interested in Bioinformatics and health-related area
* High accuracy in prediction is important for early cancer diagnosis
* Interested in how similar tumors are and if the similarity is closely related to cancer diagnosis

**Subset**:

I did feature selection and removed some columns, including the information about the standard error of tumor’s attributes and patients’ id. (I think the mean and the standard error are kind of redundant, so removing one of the measures should be fine.) After feature selection, we have 11 columns with 1 response variable and 10 explanatory variables about the mean of each attribute.

**Dimension**:

569 rows and 11 columns

**Features:**

| Features | Data type | Description | Non-Null Count |
| --- | --- | --- | --- |
| Diagnosis | object | (M = malignant, B = benign) | 569 Non-Null |
| radius\_mean | float64 | mean of distances from the center to points on the perimeter | 569 Non-Null |
| texture\_mean | float64 |  | 569 Non-Null |
| perimeter\_mean | float64 |  | 569 Non-Null |
| area\_mean | float64 |  | 569 Non-Null |
| smoothness\_mean | float64 | local variation in radius lengths | 569 Non-Null |
| compactness\_mean | float64 | perimeter^2 / area - 1.0 | 569 Non-Null |
| concavity\_mean | float64 | The severity of concave portions of the contour | 569 Non-Null |
| concave points\_mean | float64 | number of concave portions of the contour | 569 Non-Null |
| symmetry\_mean | float64 |  | 569 Non-Null |
| fractal\_dimension\_mean | float64 | "coastline approximation" - 1 | 569 Non-Null |

**Tasks:**

* Scale each feature before analysis
* Analyze correlations between each observation(tumor) by constructing a similarity/dissimilarity matrix
* Understand the relationship between pairs of tumor’s attribute
* Determine which attribute contributes most to cancer
* Prediction of diagnosis with breast cancer (Future goal)

**Data analysis and package use**

Packages: NumPy, Pandas, Matplotlib, seaborn

**Step 1: Scale each feature so that all values are ranging from 0 to 1.**

d′=(d−min\_d)/(max\_d−min\_d)

This is an example of a linear transformation, which preserves the relative distances between points. After the scaling, the values from each feature are comparable.

I used two ways to scale the data. Method one is to use for loop and apply the above formula to each data point. The second method is to use a function called *scaler.fit\_transform* from package *sklearn.preprocessing (MinMaxScaler)*. These two methods get the same result.

Code:

For method 1, I did not use any canned function from packages. I first drop the response variable from my dataset since we only need to scale the numeric data. Then I used a for loop to loop through each column and do the scaling for each value from each column by using simple math functions such as min() and max(). Then, I stored the scaled data in a new data frame called data\_scale for further use.

**Step 2: Construct a Euclidean distance matrix between tumors and scale the values between 0 and 1. I also save the matrix as a CSV file in the current directory.** I found it useful to open the .csv file in **excel** for more data exploration. If you want to know which tumor is the most similar to tumor 1, you can select column 1 and then sort the corresponding row in excel. Then the lowest value indicates the highest similarity between two tumors.

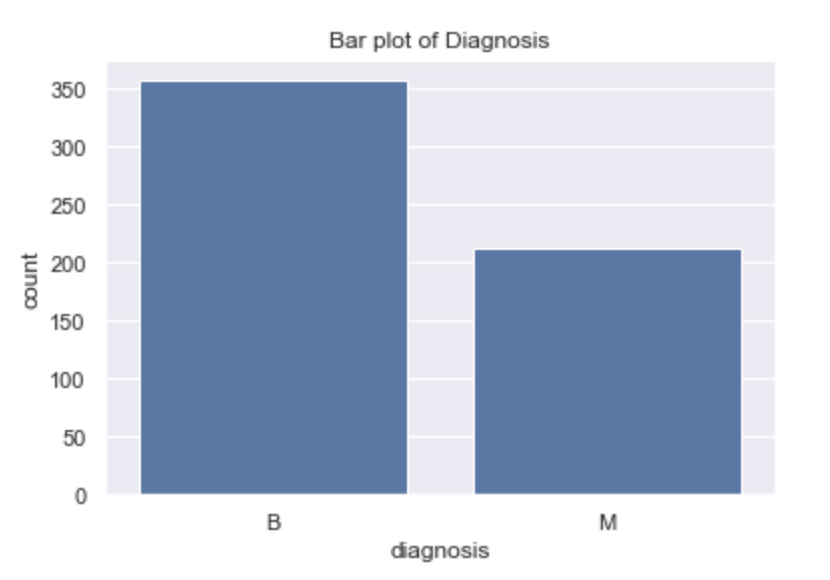
Code: I used a double for loop for looping through each pair of observations and calculate their corresponding Euclidean distance. Within the for loop, I used my linear algebra knowledge to compute. I first do the subtraction and use np.dot(from numpy package) to calculate the dot product(aka. The sum of the square from the formula). Then, I take the square root and get the distance. I also make this as a function for calling.

The Euclidean distance is for measuring dissimilarity, and we can get the equivalent similarity matrix by 1-values in the matrix after scaling.

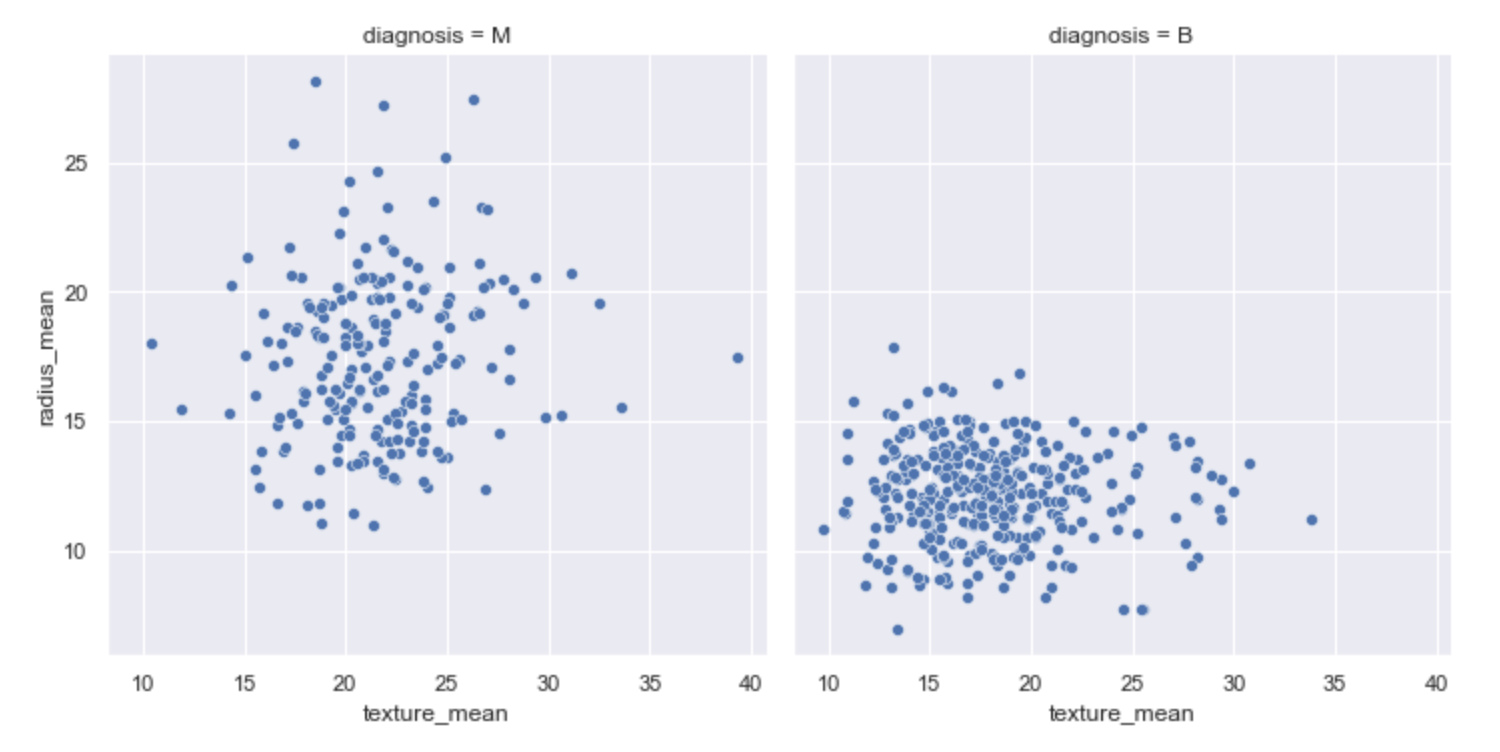
**Step 3: Construct a Cosine similarity matrix between tumors. Save the matrix as a CSV file in the current directory.** You can do the same thing in excel and explore the similarity between any pair of tumors.

Data visualization:

Step 4: Plot a barplot for the response variable indicating the number of observations in each category. (B for benign and M for maglinant)



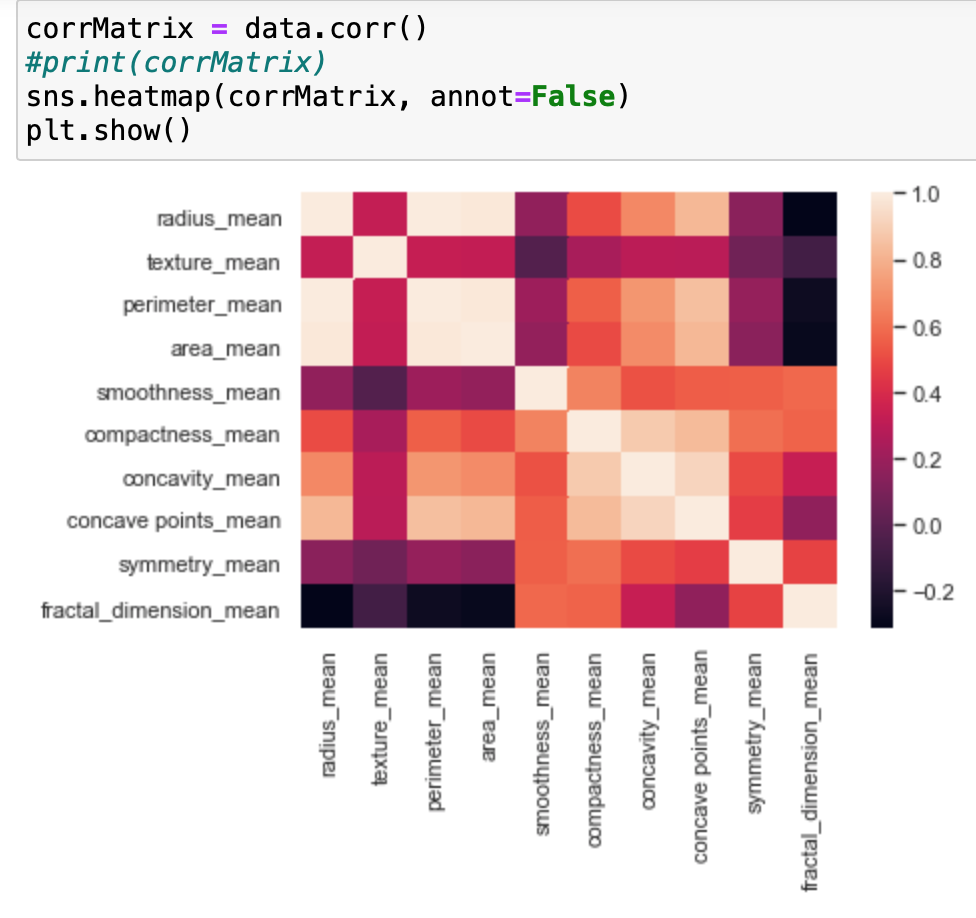
Step 5: **We made several scatter plots to explore the relationship between two features.** From the below plot, we can conclude that there is a relatively strong correlation between texture\_mean and radius mean. Also, benign tumors tend to have smaller texture\_mean and radius mean. This is an interesting finding and might be helpful for cancer diagnosis.



Step 5: We then try to explore the relationship between explanatory variables and we calculate the Pearson correlation and made a correlation matrix and visualize it.

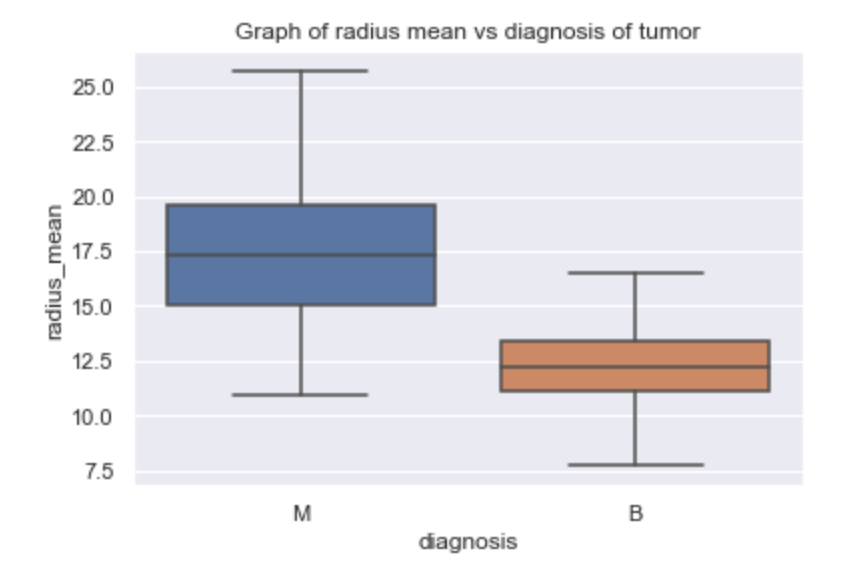
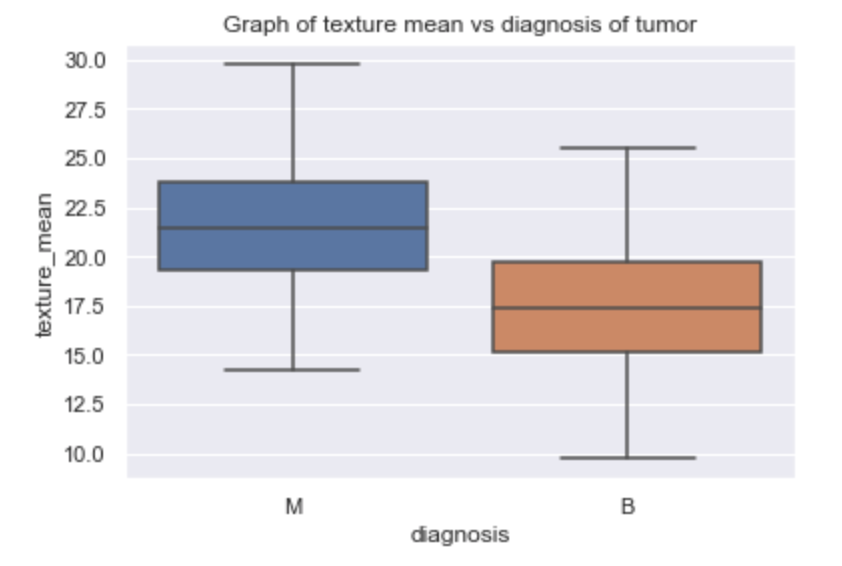
The relationship between radius\_mean and perimeter\_mean and area\_mean seems to be very strong and this makes sense since perimeter = 2\*radius and area = π\*radius^2.

According to the correlation heatmap, I think the relationship between texture mean and radius mean is relatively strong and the correlation factor is around 0.4.

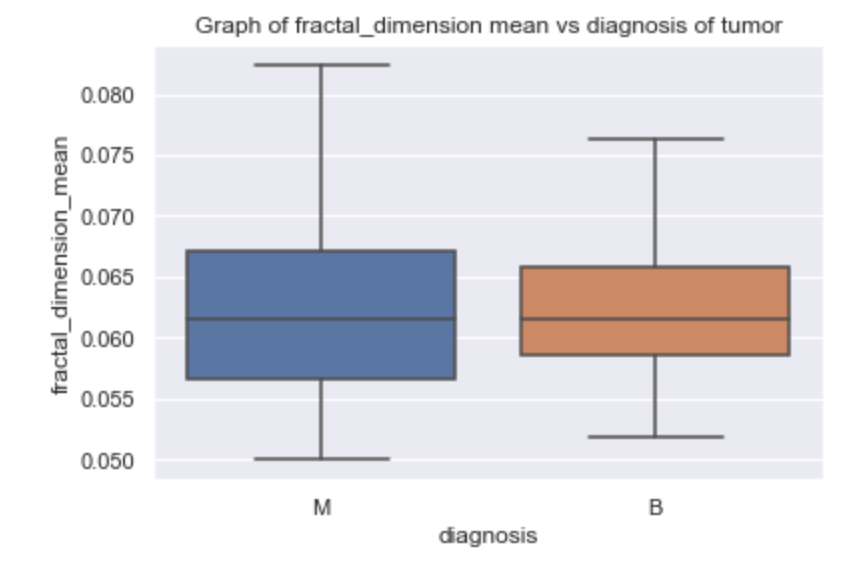


Step 5: Try to find which factors have an influence on the results of cancer. Since the response variable is a binary variable, we made boxplots to explore our data.

This side-by-side boxplot shows that there is a big difference of texture mean between malignant and benign. The median, interquartile range, and the range of Malignant tumors are bigger than those of benign tumors. So does radius mean. Therefore, we believe these two attributes are good predictors of cancer.



However, there is not a huge difference between malignant and benign tumors’ fractal dimension mean even though the range of malignant tumor’s fractal dimension mean is larger. We might conclude that the fractal dimension is not a determinant factor for predicting cancer.



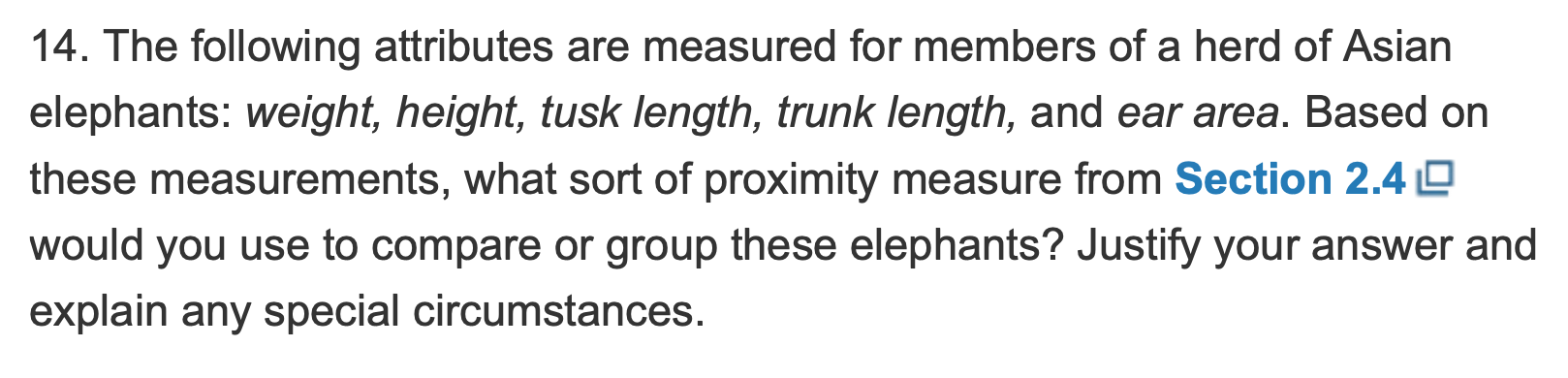
**Program development**

I implement several algorithms from chapter 2, including the Euclidean distance matrix(dissimilarity matrix) and cosine similarity matrix. I also implement a normalization algorithm to scale my values to [0,1].

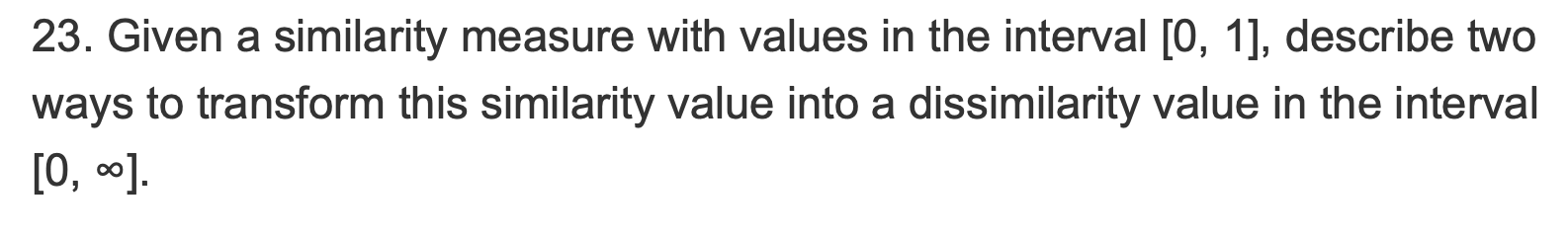
**Theory (Some ideas below, including text exercises. Not all would have to be done. In future chapters, there are some good theory-type exercises at the end of the chapter.)**

I also implement an interactive function to get the similarity by asking the user to input object 1 and object 2. This function locates and corresponding similarity value and prints it out.

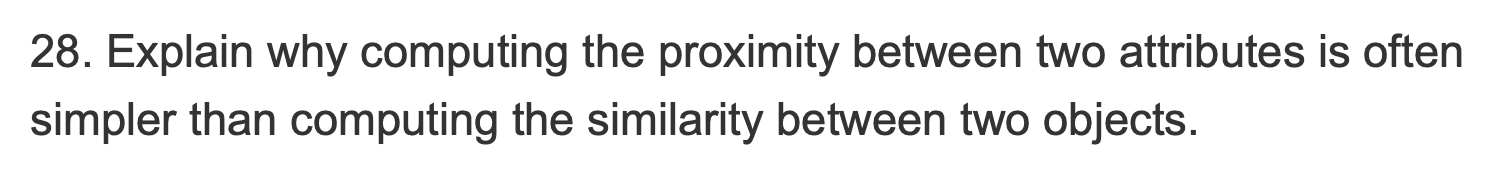
I did some textbook exercises:



**In my opinion, Euclidean distance might be the best choice since all the elephants’ features are numerical. Also, I suggest the programmer normalize each feature before applying the Euclidean calculation since the scale of weight and height must be different.**



**I think we can try to get (1-similarity)/similarity. This is a good way to map [0,1] to [0,infinity]. Another way could be doing a log transform.**



**I think this is because for comparing two objects, we need to consider all the attributes they have. In this way, the similarity between two objects is actually the combination of all similarity values from each attribute the object has.**

**Student learning summary and self-assessment**

I think I did a good job in this portfolio. I have a good understanding of the algorithms described in the textbook and I implement some of them in python. I also did many self-study in python syntax including how to manipulate data frames(how to drop one column, how to write the new dataframe as a csv file, how to define a function and etc). When implementing the Euclidean distance matrix, I used my linear algebra knowledge from MST121 to convert the formula to an algebra computation(dot product). I also taught myself about how to do scatterplot, barplot, and boxplot in python by using matplotlib and seaborn package. Those visualizations are good and clear.

My Jupyter notebook file is in the same data folder. Feel free to check it!