ICS Spring 2020
Quiz 2

Note: Please download the quiz materials from Quiz 2 under Assignments on NYU Classes which include the following items:

- answers.txt
- wine_data.txt
- wine_cluster_student.py
- kmeans.py
- mean_std_student.py
- sample.py
- util.py

You have 60 minutes to complete the quiz. You're allowed to use any material while WiFi-OFF. For the non-programming part, please have your answers in **answers.txt**. When finished, compress all your work into a .zip file and submit it to NYU Classes.

Good luck!

Name:_____

NetID:_____

MACHINE LEARNING BASICS

Question 1

Some speech recognition systems require "training" where an individual speaker reads text or isolated vocabulary into the system. The system analyzes the person's specific voice and uses it to fine-tune the recognition of that person's speech, resulting in increased accuracy. (from Wikipedia). What is the performance measure P, experience E, and task T in this setting?

a. Number of words correctly converted into text
b. Recognize a person's speech
c. The speaker's facial expression
d. Speaker's speech and the corresponding text

Measure P: ____ Experience E: ____ Task T: ____

Question 2

You're a marketing analyst of Greyhound (an American intercity bus carrier) and you want to develop learning algorithms to address each of two problems.
Problem 1: You have all the customer information, you want to know whether they will continue subscribing at a higher price or not.
Problem 2: You have the company's annual revenue report of the past 5 years. You want to predict the total revenue for the next year.
Should you treat these as classification or as regression problems?

a. Problem 1 is a regression problem
b. Problem 1 is a classification problem.
c. Problem 2 is a regression problem
d. Problem 2 is a classification problem

Question 3

Please put the following steps of gradient descent in the correct order for a one-parameter estimation:

A. Compute gradient at the current estimated value and use it to update the parameter.
B. If the change of loss function is less than the predetermined tolerance value, i.e.there is barely any change in the loss function, terminate the iteration and report final estimated parameters, otherwise, adjust the parameter again.
C. Set initial value of target parameter and set learning rate.
D. Calculate loss function with parameters at the current value and record it.
E. Calculate the loss function with the updated parameter and compare it with the previous one.

Correct order: ____ -> D -> ____-> ____ -> B

PROBABILITY

Question 4

Now suppose you draw *two* fair dice in a sequence and get two random numbers: X, Y. Please figure out the following probabilities:

1) What is P(X is odd AND Y is odd)?

2) What is P(X is odd OR Y is prime)?

3) **Bonus:** What is P(X is odd | Y is prime)?

Question 5(Conditional probability)

A population is composed of 50% male and 50% female. From a survey we learned that 5% of the population smokes. Of those people who smoke, 60% are male. What is the probability that the person surveyed is a smoker GIVEN that he is a male?

PROGRAMMING

Question 6

The *mean* and *standard deviation* (or simply as std) is defined as

$\mu = \frac{1}{N}\sum_{i=1}^{N} \quad x_i$ and$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \quad (x_i - \mu)^2}$

Complete the ***compute_mean*** and ***compute_std*** function in **mean_std_student.py**. The correct output is two new samples, *data_mean* and *data_std*, each keeps the mean and std of these data. Note that we have learned how to compute the mean in K-means algorithm in the class: the i-th element of *data_mean* feature is the mean of i-th element of features of the complete data. Variance works the same way.

The main testing code is shown on the left; result on the right.

```
30 if __name__ == "__main__":
31     a = sample.Sample('a', [1, 1])
32     b = sample.Sample('b', [5, 3])
33     print(a)
34     print(b)
35
36     data = [a, b]
37     data_mean = compute_mean(data)
38     data_std = compute_std(data, data_mean)
39     print("mean: " + str(data_mean))
40     print("std: " + str(data_std))
```

```
[1, 1]
[5, 3]
mean: [3.0, 2.0]
std: [2.0, 1.0]
```

You should use the appropriate APIs(functions) in the **sample.py**, to add two sample a and b, just call “a + b”,. We have also provided a power(x) method, to raise each feature to power of “x”. You need this to implement variance. Example:

```
In [2]: a = Sample('a', [1, 2, 4])

In [3]: print(a.power(2))
[1, 4, 16]

In [4]: print(a.power(1/2))
[1.0, 1.4142135623730951, 2.0]
```

Question 7

Normalization is a key step to pre-process your data before applying machine learning algorithms to it. It makes different features have equal contributions, instead of biased. Features can be, for instance, body weight and front teeth size -- not only they are in different metrics space, their values differ dramatically.

The most common process is the following: loop through all the samples, for every sample:

- Subtract *data_mean*
- Divide by *data_std*

The first operation *center* data, and second standardize the range (contribution) of different features.

We have also provided a new API vec_div(other) method in the Sample class, to do element-wise division of a over b, just call a.vec_div(b). Example shown below:

```
In [3]: a = sample.Sample('a', [5, 4])

In [4]: b = sample.Sample('b', [2, 4])

In [5]: print(a.vec_div(b))
[2.5, 1.0]
```

Complete the ***normalization*** function in the **mean_std_student.py** code. Again, testing code on the left, and result on the right.

```
35 if __name__ == "__main__":
36     a = sample.Sample('a', [1, 1])
37     b = sample.Sample('b', [5, 3])
38     print(a)
39     print(b)
40
41     data = [a, b]
42     data_mean = compute_mean(data)
43     data_std = compute_std(data, data_mean)
44     print("mean: " + str(data_mean))
45     print("std: " + str(data_std))
46
47     normalized_data = normalization(data)
48     print("after normalization...")
49     for d in normalized_data:
50         print(d)
```

```
[1, 1]
[5, 3]
mean: [3.0, 2.0]
std: [2.0, 1.0]
after normalization...
[-1.0, -1.0]
[1.0, 1.0]
```

Question 8

We are given the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars (please see the data file wine_data.txt). The analysis determined the quantities of 13 constituents found in each of the three types of wines.

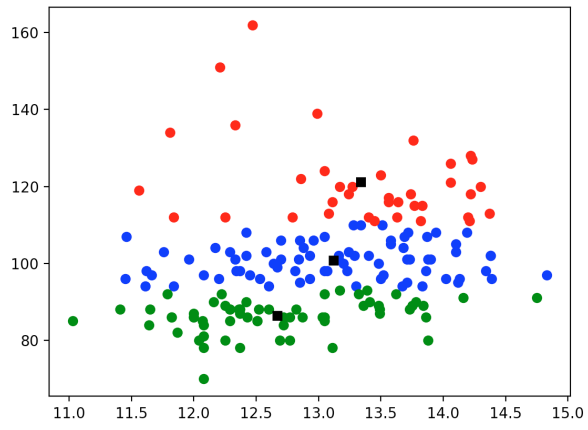
As listed in the data, there are 13 features/attributes:1) Alcohol, 2) Malic acid, 3) Ash, 4) Alcalinity of ash, 5) Magnesium, 6) Total phenols, 7) Flavanoids, 8) Nonflavanoid phenols, 9) Proanthocyanins, 10)Color intensity, 11)Hue, 12)OD280 of diluted wines, 13)Proline.

We can cluster the 178 wines based on the 13 features. However, to make things simple, pick the ***first and fifth*** attributes (**Alcohol and Magnesium**) as features, and cluster the 178 samples from **wine_data.txt** into **three** clusters.

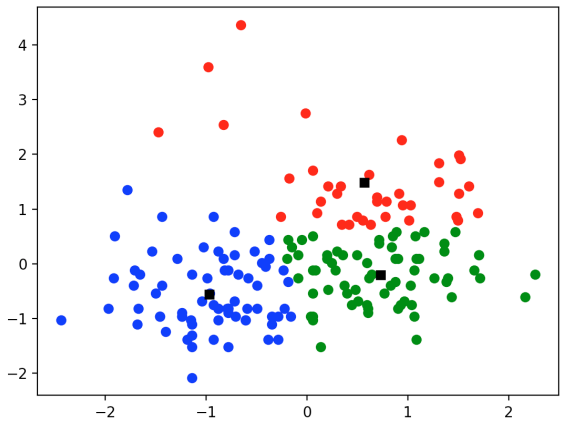
For this to work, you need to read the data into Sample class, and then call k-means function. Complete your work in **wine_cluster_student.py**.

Bonus: ***normalize*** your selected data which were read into Sample class, and then call k-means to cluster them. Save your work in **wine_cluster_student.py** as well.

Sample output:



The 3 cluster w/o normalizing the data first



The 3 cluster after normalizing the data first