Lauren Gilfillan
Lag179
4466713

Machine Learning : ps1

Instructions for running code

The code should be an executable python file, you should be able to run it in terminal

1. Propose a new regression problem

Problem : Predict the height plant x will grow to

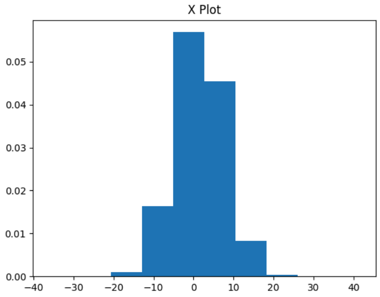
a. Features : Age of plant, history of plant height, amount of sunlight given, amoun of water given
b. height (in cm)
c. You could collect data through discrete measurement of height, time since germination, UV index of Sunlight, and liters of water given
d. Height of a plant can always vary and is never static based on the DNA of the plant itself, how it was germinated etc. There are a lot of features (perhaps too many) to quantify how tall the plant will grow

2. Propose a new classification problem

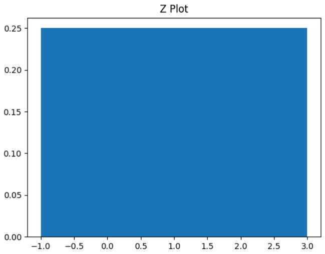
Problem : What day of the week is an exam most likely to be on for a certain class?

a. Features : history of exam dates, length of semester, number of exams
b. Day of the Week {Monday Tuesday Wednesday Thursday Friday
c. Evaluate previous exam dates, evaluate previous start/end dates of a semester, the average number exams given in one semester, average spacing between exams
d. Exams are often at the whim of the professor and we cannot safely predict the will of a professor

3.

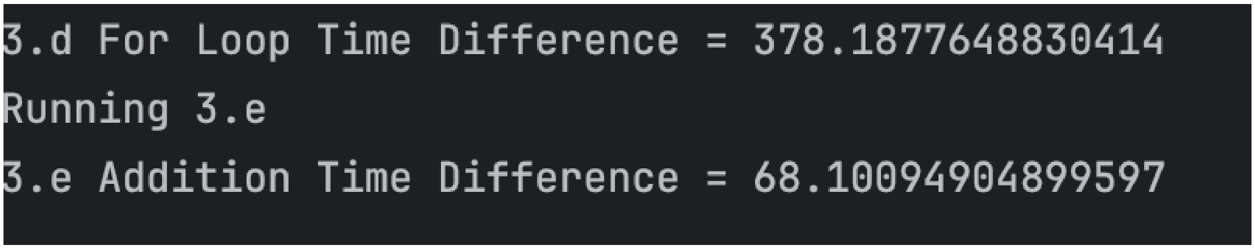


Ps1-3-c-1.png



ps1-3-c-2.png

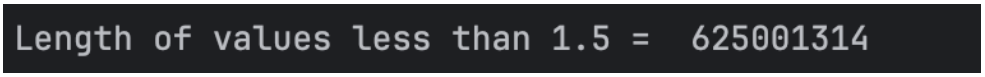
c. I believe the histogram of x gives us a rough gaussian distribution, it follows a curve that peaks in the middle of the predefined range. The histogram of z also has a uniform distribution where its points are spread evenly across all bins



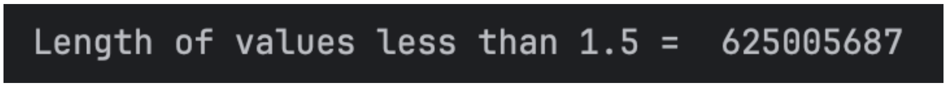
```
3.d For Loop Time Difference = 378.1877648830414
Running 3.e
3.e Addition Time Difference = 68.10094904899597
```

e. It is much for efficient to do addition with the predefined numpy function

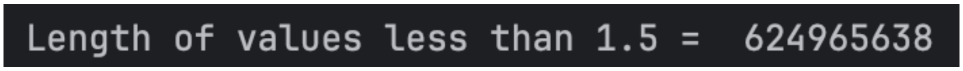
f.



```
Length of values less than 1.5 =  625001314
```



```
Length of values less than 1.5 =  625005687
```

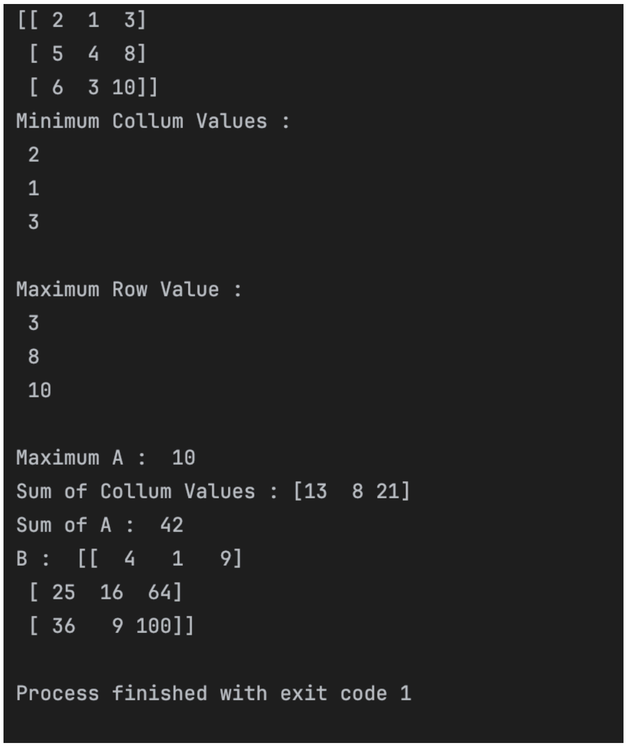


```
Length of values less than 1.5 =  624965638
```

Because this is a random distribution, the numbers that fall into certain buckets will never be the same, however because this is a uniform distribution of points, we know that it will be similar number of points in each bin because uniform distribution implies all bins have mostly equal number of values

4.

a.



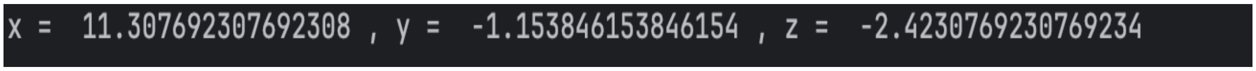
```
[[ 2  1  3]
 [ 5  4  8]
 [ 6  3 10]]
Minimum Collum Values :
 2
 1
 3

Maximum Row Value :
 3
 8
 10

Maximum A :  10
Sum of Collum Values : [13  8 21]
Sum of A :  42
B :  [[  4   1   9]
 [ 25  16  64]
 [ 36   9 100]]

Process finished with exit code 1
```

b.



```
x =  11.307692307692308 , y =  -1.153846153846154 , z =  -2.4230769230769234
```

c.

$L_1(x) = \sum | x_n |$

$\Downarrow$

$\text{L}_1(x1) = | -1.5 | + | 0.5 | + | 0 | = 2$

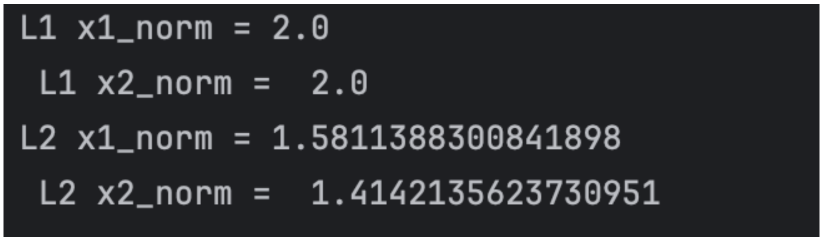
$L_1(x2) = | -1 | + | -1 | + | 0 | = 2$

$L_2(x) = \sqrt{\sum x_n^2}$

$\Downarrow$

$\text{L}_2(x_1) = \sqrt{-1.5^2 + 0.5^2 + 0^2} = \sqrt{2.5} = 1.5811$

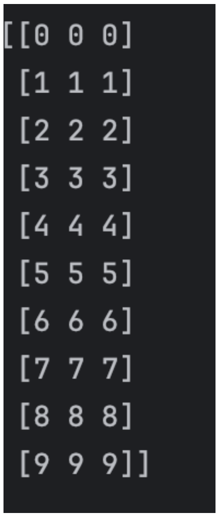
$L_2(x_2) = \sqrt{-1^2 + -1^2 + 0^2} = \sqrt{2} = 1.414$



```
L1 x1_norm = 2.0
 L1 x2_norm =  2.0
L2 x1_norm = 1.5811388300841898
 L2 x2_norm =  1.4142135623730951
```

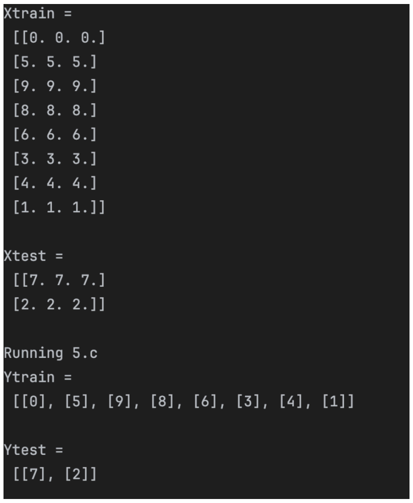
5.

a. Image : ps1-5-a.png



```
[[0 0 0]
 [1 1 1]
 [2 2 2]
 [3 3 3]
 [4 4 4]
 [5 5 5]
 [6 6 6]
 [7 7 7]
 [8 8 8]
 [9 9 9]]
```

b. Images : ps1-5-d-(1-3).png

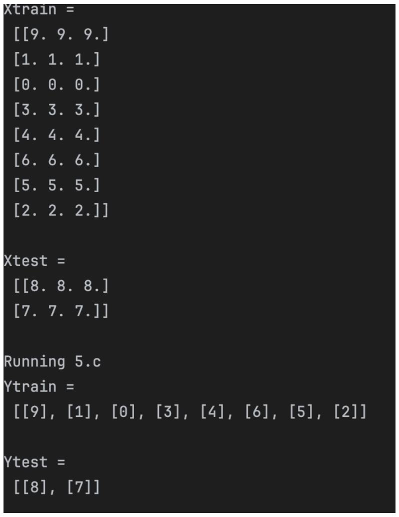


```
Xtrain =
 [[0. 0. 0.]
 [5. 5. 5.]
 [9. 9. 9.]
 [8. 8. 8.]
 [6. 6. 6.]
 [3. 3. 3.]
 [4. 4. 4.]
 [1. 1. 1.]]

Xtest =
 [[7. 7. 7.]
 [2. 2. 2.]]

Running 5.c
Ytrain =
 [[0], [5], [9], [8], [6], [3], [4], [1]]

Ytest =
 [[7], [2]]
```

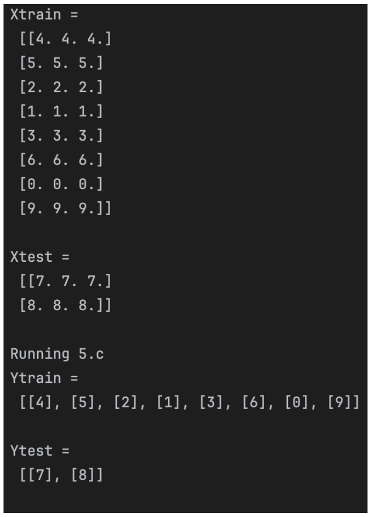


```
Xtrain =
 [[9. 9. 9.]
 [1. 1. 1.]
 [0. 0. 0.]
 [3. 3. 3.]
 [4. 4. 4.]
 [6. 6. 6.]
 [5. 5. 5.]
 [2. 2. 2.]]

Xtest =
 [[8. 8. 8.]
 [7. 7. 7.]]

Running 5.c
Ytrain =
 [[9], [1], [0], [3], [4], [6], [5], [2]]

Ytest =
 [[8], [7]]
```



```
Xtrain =
 [[4. 4. 4.]
 [5. 5. 5.]
 [2. 2. 2.]
 [1. 1. 1.]
 [3. 3. 3.]
 [6. 6. 6.]
 [0. 0. 0.]
 [9. 9. 9.]]

Xtest =
 [[7. 7. 7.]
 [8. 8. 8.]]

Running 5.c
Ytrain =
 [[4], [5], [2], [1], [3], [6], [0], [9]]

Ytest =
 [[7], [8]]
```

No, because the “shuffle”
numpy function is considered random, we will never get the same shuffle 2x in a row