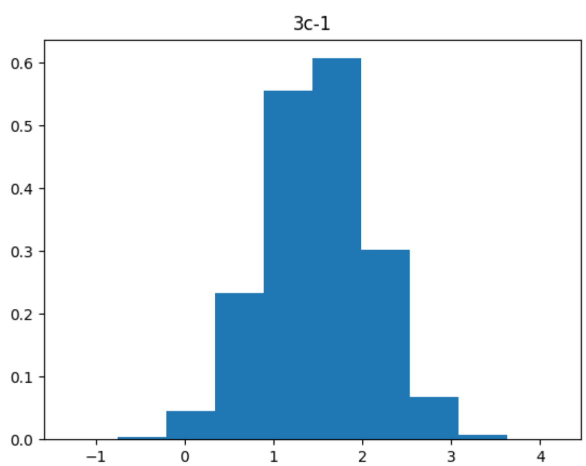
Alex Ivensky - 4485057
ECE 1395
Homework 1

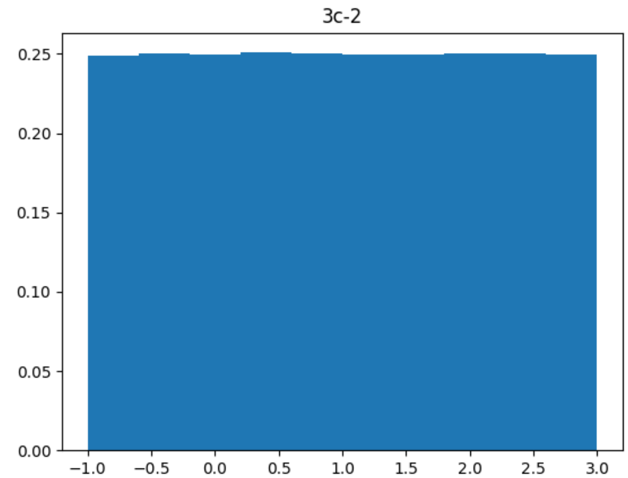
1. A good problem to solve using a regression supervised learning technique would be item pricing in retail based on the current time of day/month/year (for example, Christmas items are more popular around November so their prices can be increased to generate more revenue).
    a. As features (x), I would use the time of day/month/year.
    b. As labels (y), I would use the price of items.
    c. I would collect data by conducting a trial over some period given existing price trends, and I would also consider artificially fluctuating prices to see how the market responds.
    d. This problem can be challenging because there are multiple factors to consider that can affect a buyer's shopping habits, and it will be unclear if the current time alone is enough to affect habits.
2. A good problem to solve using a classification supervised learning technique would be speech recognition to classify certain sounds as words.
    a. As features (x), I would use the sound input.
    b. As labels (y), I would use the words that the sounds represent.
    c. I would collect data by providing various samples of recordings along with the words that they represent.
    d. This problem can be difficult because there are infinite ways that a word can be represented in sound, due to accents, tones of voice, and other factors.

3c.

The histogram for x does look like a Gaussian distribution.



The histogram for z does look like a uniform distribution.



3d.

```
Loop execution time: 1.1336390972137451
```

3e.

```
Numpy function time: 0.00015497207641601562
```

The NumPy function is far more efficient.

3f.

Trial 1:

```
Number of elements retrieved: 374828
```

Trial 2:

```
Number of elements retrieved: 374627
```

Trial 3:

```
Number of elements retrieved: 374426
```

There is a slight difference, but the numbers always float around 375000. This is because we used a randomization function to choose the numbers, but the given range will on average account for about 37.5% of the data because we have used a uniform distribution.

4a.

```
4a results:
Minimum value in each column:
[[2 1 3]]
Max value in each row:
[[ 3]
 [ 8]
 [18]]
Highest value in A:
18
Sum of columns:
[[10 15 29]]
Sum of A:
54
B:
[[  4   1   9]
 [  4  36  64]
 [ 36  64 324]]
```

4b.

```
Solution for 4b:
[[0.3]
 [0.4]
 [0. ]]
```

4c.

4c.)

$x_1 = [-0.5 \quad 0 \quad 1.5]$

$|x_1|_1 = 0.5 + 0 + 1.5 = 2$

$|x_1| = \sqrt{(0.5)^2 + (0)^2 + (1.5)^2} = 1.58$

$x_2 = [-1 \quad -1 \quad 0]$

$|x_2|_1 = 1 + 1 + 0 = 2$

$|x_2| = \sqrt{(1)^2 + (1)^2 + (0)^2} = 1.41$

```
4c NumPy calculations:
x1_l1: 2.0
x1_l2: 1.5811388300841898
x2_l1: 2.0
x2_l2: 1.4142135623730951
```

5a.

```
5a X:
 [[ 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]
 [10 10 10]]
```

<table>
<tr><th>Trial 1:</th><th>Trial 2:</th><th>Trial 3:</th></tr>
<tr>
<td>X_train:
[[7 7 7]
[8 8 8]
[6 6 6]
[1 1 1]
[4 4 4]
[10 10 10]
[2 2 2]
[5 5 5]]
X_test:
[[9 9 9]
[3 3 3]]
y_train:
[[7]
[8]
[6]
[1]
[4]
[10]
[2]
[5]]
y_test:
[[9]
[3]]</td>
<td>X_train:
[[1 1 1]
[10 10 10]
[6 6 6]
[5 5 5]
[8 8 8]
[2 2 2]
[4 4 4]
[3 3 3]]
X_test:
[[9 9 9]
[7 7 7]]
y_train:
[[1]
[10]
[6]
[5]
[8]
[2]
[4]
[3]]
y_test:
[[9]
[7]]</td>
<td>X_train:
[[2 2 2]
[4 4 4]
[8 8 8]
[5 5 5]
[9 9 9]
[3 3 3]
[10 10 10]
[1 1 1]]
X_test:
[[6 6 6]
[7 7 7]]
y_train:
[[2]
[4]
[8]
[5]
[9]
[3]
[10]
[1]]
y_test:
[[6]
[7]]</td>
</tr>
</table>

We do not get the same submatrices each time. This is because, again, we used a randomization function to shuffle the matrix rows to choose the submatrices. Thus, we should expect different results each time.