Assignment 1 - Probability, Linear Algebra, Programming, and Git

Ruiqi Wang

Netid: *rw195*

Probability and Statistics Theory

1

Let $f(x) = \begin{cases} 0 & x < 0 \\ \alpha x^2 & 0 \leq x \leq 2 \\ 0 & 2 < x \end{cases}$

For what value of α is $f(x)$ a valid probability density function?

Note: for all assignments, write out all equations and math for all assignments using markdown and LaTeX (https://tobi.oetiker.ch/lshort/lshort.pdf) and show all work

ANSWER

Let $f(x)$ to be a valid probability density function, then $\int_{-\infty}^{+\infty} f(x) = 1$.

$\int_{-\infty}^{0} f(x) + \int_{0}^{2} f(x) + \int_{2}^{+\infty} f(x) = 1$

$\int_{0}^{2} \alpha x^2 \,\mathrm{d}x = 1$

$\alpha = 3/8$

2

What is the cumulative distribution function (CDF) that corresponds to the following probability distribution function? Please state the value of the CDF for all possible values of x.

$$f(x) = \begin{cases} \frac{1}{3} & 0 < x < 3 \\ 0 & \text{otherwise} \end{cases}$$

ANSWER

Let the CDF for all x be P(x), when 0<X<3:

$$P(X) = \int_{-\infty}^{0} 0\mathrm{d}x + \int_{0}^{X} 1/3\mathrm{d}x$$

$$P(X) = 1/3X$$

Therefore,

$$P(x) = \begin{cases} 0 & x \leq 0 \\ 1/3x & 0 < x \leq 3 \\ 1 & \text{otherwise} \end{cases}$$

3

For the probability distribution function for the random variable X,

$$f(x) = \begin{cases} \frac{1}{3} & 0 < x < 3 \\ 0 & \text{otherwise} \end{cases}$$

what is the (a) expected value and (b) variance of X. *Show all work*.

ANSWER

(a) Expected value:

$E(x) = \int_{-\infty}^{\infty} xf(x)\mathrm{d}x$
$= 0 + \int_0^3 \ 1/3x\mathrm{d}x + 0$
$= 3/2$

(b) Variance:

$Var(x) = \int_{-\infty}^{\infty}(x - \mu)^2 f(x)\mathrm{d}x$
$= \int_0^3 1/3x^2\,\mathrm{d}x - \mu^2$
$= 3 - \mu^2$

Derived from (a), $\mu = 2/3$, therefore

Var(x) = 3/4

4

Consider the following table of data that provides the values of a discrete data vector $\mathbf{x}$ of samples from the random variable X, where each entry in $\mathbf{x}$ is given as x_i.

Table 1. Dataset N=5 observations

	x_0	x_1	x_2	x_3	x_4
$\mathbf{x}$	2	3	10	-1	-1

What is the (a) mean, (b) variance, and the of the data?

Show all work. Your answer should include the definition of mean, median, and variance in the context of discrete data.

ANSWER

(a) Mean

$Mean = \frac{1}{N}\sum_0^N x_i$ = (2+3+10-1-1)/5 = 13/5

(b) Variance

$Variance = \frac{1}{N}\sum_0^N (x_i - Mean)^2$ = 1/5((2-13/5)^2 + (3-13/5)^2 + (10-13/5)^2 + (-1-13/5)^2 + (-1-13/5)^2) = 406/25

(c) Median

The median is the value separating the higher half from the lower half of a data sample.
Here the Median of X is 2.

5

Review of counting from probability theory.

(a) How many different 7-place license plates are possible if the first 3 places only contain letters and the last 4 only contain numbers?

(b) How many different batting orders are possible for a baseball team with 9 players?

(c) How many batting orders of 5 players are possible for a team with 9 players total?

(d) Let's assume this class has 26 students and we want to form project teams. How many unique teams of 3 are possible?

Hint: For each problem, determine if order matters, and if it should be calculated with or without replacement.

ANSWER

(a) Order matters, with replacement.
26^3 * 10^4 = 175760000

(b) Order matters, without replacement.
9! = 362880

(c) Order matters, without replacement.
$\binom{9}{5}$ * 5! = 15120

(d) Order doesn't matter, withou replacement.
$\binom{26}{3}$ = 2600

Linear Algebra

6

Matrix manipulations and multiplication. Machine learning involves working with many matrices, so this exercise will provide you with the opportunity to practice those skills.

Let $\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} -1 \\ 3 \\ 8 \end{bmatrix}$, $\mathbf{c} = \begin{bmatrix} 4 \\ -3 \\ 6 \end{bmatrix}$, and $\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Compute the following or indicate that it cannot be computed:

1. $\mathbf{AA}$
2. $\mathbf{AA}^T$
3. $\mathbf{Ab}$
4. $\mathbf{Ab}^T$
5. $\mathbf{bA}$
6. $\mathbf{b}^T\mathbf{A}$
7. $\mathbf{bb}$
8. $\mathbf{b}^T\mathbf{b}$
9. $\mathbf{bb}^T$
10. $\mathbf{b} + \mathbf{c}^T$
11. $\mathbf{b}^T\mathbf{b}^T$
12. $\mathbf{A}^{-1}\mathbf{b}$
13. $\mathbf{A} \circ \mathbf{A}$
14. $\mathbf{b} \circ \mathbf{c}$

Note: The element-wise (or Hadamard) product is the product of each element in one matrix with the corresponding element in another matrix, and is represented by the symbol "$\circ$".

In [3]:
```
import numpy as np
from numpy import linalg as la

A = np.array([
    [1,2,3],
    [2,4,5],
    [3,5,6]
])
b = np.array([-1,3,8]).reshape(3,1)
c = np.array([4,-3,6]).reshape(3,1)
I = np.array([
    [1,0,0],
    [0,1,0],
    [0,0,1]
])
```

In [7]:
```
print('1.\nAA=\n', A@A)
print('\n2.\nAAT=\n', A@(A.T))
print('\n3.\nAb=\n', A@b)
print('\n4.\nAbT:', 'cannot be computed')
print('\n5.\nbA:', 'cannot be computed' )
print('\n6.\nbTA=\n', (b.T)@A)
print('\n7.\nbb:', 'cannot be computed')
print('\n8.\nbTb=\n', (b.T)@b)
print('\n9.\nbbT=\n', b@(b.T))
print('\n10.\nb+cT:', 'cannot be computed')
print('\n11.\nbTbT:', 'cannot be computed')
print('\n12.\nA^(-1)b=\n', la.inv(A)@b)
print('\n13.\nA∘A=\n', A*A)
print('\n14.\nb∘c=\n', b*c)
```

```
1.
AA=
 [[14 25 31]
 [25 45 56]
 [31 56 70]]

2.
AAT=
 [[14 25 31]
 [25 45 56]
 [31 56 70]]

3.
Ab=
 [[29]
 [50]
 [60]]
```

```
4.
AbT: cannot be computed

5.
bA: cannot be computed

6.
bTA=
 [[29 50 60]]

7.
bb: cannot be computed

8.
bTb=
 [[74]]

9.
bbT=
 [[ 1 -3 -8]
 [-3  9 24]
 [-8 24 64]]

10.
b+cT: cannot be computed

11.
bTbT: cannot be computed

12.
A^(-1)b=
 [[ 6.]
 [ 4.]
 [-5.]]

13.
A∘A=
 [[ 1  4  9]
 [ 4 16 25]
 [ 9 25 36]]

14.
b∘c=
 [[-4]
 [-9]
 [48]]
```

6

Eigenvectors and eigenvalues. Eigenvectors and eigenvalues are useful for some machine learning algorithms, but the concepts take time to solidly grasp. For an intuitive review of these concepts, explore this interactive website at Setosa.io (http://setosa.io/ev/eigenvectors-and-eigenvalues/). Also, the series of linear algebra videos by Grant Sanderson of 3Brown1Blue are excellent and can be viewed on youtube here (https://www.youtube.com/playlist?list=PLZHQObOWTQDPD3MizzM2xVFitgF8hE_ab).

1. Calculate the eigenvalues and corresponding eigenvectors of matrix $\mathbf{A}$ above, from the last question.
2. Choose one of the eigenvector/eigenvalue pairs, $\mathbf{v}$ and λ, and show that $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$. Also show that this relationship extends to higher orders: $\mathbf{A}\mathbf{A}\mathbf{v} = \lambda^2\mathbf{v}$
3. Show that the eigenvectors are orthogonal to one another (e.g. their inner product is zero). This is true for real, symmetric matrices.

In [145]:
```
val,vec = la.eigh(A)
print('1.\neigenvalue1=', val[0], '\neigenvector1=', vec[:,0])
print('\neigenvalue2=', val[1], '\neigenvector2=', vec[:,1])
print('\neigenvalue3=', val[2], '\neigenvector3=', vec[:,2])
```

```
1.
eigenvalue1= -0.5157294715892574
eigenvector1= [-0.73697623 -0.32798528  0.59100905]

eigenvalue2= 0.17091518882717727
eigenvector2= [ 0.59100905 -0.73697623  0.32798528]

eigenvalue3= 11.344814282762073
eigenvector3= [-0.32798528 -0.59100905 -0.73697623]
```

In [135]:
```
v = vec[:,0]; λ = val[0]
print('2.\nAv = ',A@v, '\nλv = ', λ*v, ' = Av\n')
print('AAv = ', A@A@v, '\nλ^2v = ', λ*λ*v, ' = AAv')
```

```
2.
Av =  [ 0.38008036  0.16915167 -0.30480078]
λv =  [ 0.38008036  0.16915167 -0.30480078]  = Av

AAv =  [-0.19601864 -0.0872365   0.15719475]
λ^2v =  [-0.19601864 -0.0872365   0.15719475]  = AAv
```

In [150]:
```
v0 = vec[:,0]; v1 = vec[:,1]; v2 = vec[:,2]
print('3.\nv0v1=',v0@v1, '\nv0v2=',v0@v2, '\nv1v2=', v1@v2)
```

```
3.
v0v1= -2.7755575615628914e-17
v0v2= -5.551115123125783e-17
v1v2= -2.220446049250313e-16
```

Numerical Programming

7

Speed comparison between vectorized and non-vectorized code. Begin by creating an array of 10 million random numbers using the numpy random.randn module. Compute the sum of the squares first in a for loop, then using Numpy's `dot` module. Time how long it takes to compute each and report the results and report the output. How many times faster is the vectorized code than the for loop approach?

*Note: all code should be well commented, properly formatted, and your answers should be output using the `print()` function as follows (where the # represents your answers, to a reasonable precision):

```
Time [sec] (non-vectorized): ######

Time [sec] (vectorized):     ######

The vectorized code is ##### times faster than the vectorized code
```

ANSWER

In [38]:

```
import numpy as np
import time

# Generate the random samples
l = np.random.randn(10000000)

# Compute the sum of squares the non-vectorized way (using a for loop)
t1=time.time()

sum1 = 0
for i in l:
    sum1 += i*i

t2=time.time()
t21=t2-t1

# Compute the sum of squares the vectorized way (using numpy)
t3=time.time()
sum2 = l@l
t4=time.time()
t43=t4-t3
tint=t21/t43

# Print the results
print('7.\nThe result of the non-vectorized way is ',sum1)
print('The result of the vectorized way is ',sum2)
print('\nTime [sec] (non-vectorized): {0:.4f}\nTime [sec] (vectorized)
:     {1:.4f}'.format(t21,t43))
print('\nThe vectorized code is {0:.1f} times faster than the non-vect
orized code'.format(tint))
```

```
7.
The result of the non-vectorized way is  9999267.874018246
The result of the vectorized way is  9999267.874017272

Time [sec] (non-vectorized): 2.7721
Time [sec] (vectorized):     0.0046

The vectorized code is 608.6 times faster than the non-vectorized co
de
```

8

One popular Agile development framework is Scrum (a paradigm recommended for data science projects). It emphasizes the continual evolution of code for projects, becoming progressively better, but starting with a quickly developed minimum viable product. This often means that code written early on is not optimized, and that's a good thing - it's best to get it to work first before optimizing. Imagine that you wrote the following code during a sprint towards getting an end-to-end system working. Vectorize the following code and show the difference in speed between the current implementation and a vectorized version.

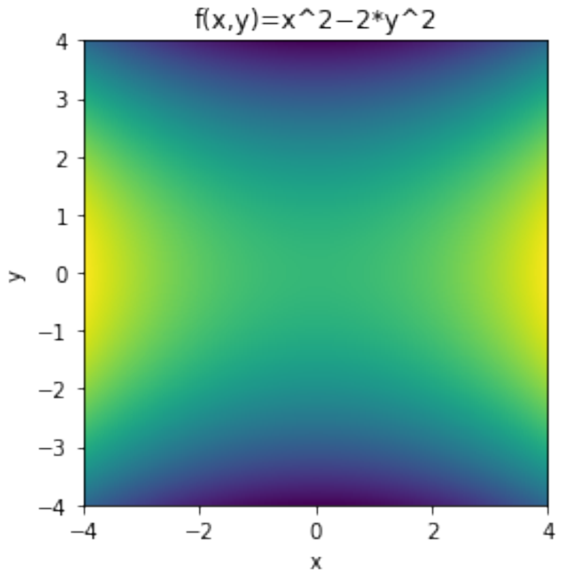
The function below computes the function $f(x, y) = x^2 - 2y^2$ and determines whether this quantity is above or below a given threshold, `thresh=0`. This is done for $x, y \in \{-4, 4\}$, over a 2,000-by-2,000 grid covering that domain.

(a) Vectorize this code and demonstrate (as in the last exercise) the speed increase through vectorization and (b) plot the resulting data - both the function $f(x, y)$ and the thresholded output - using `imshow` (https://matplotlib.org/api/_as_gen/matplotlib.pyplot.imshow.html?highlight=matplotlib%20pyplot%20imshow#matplotlib.pyplot.imshow) from `matplotlib`.

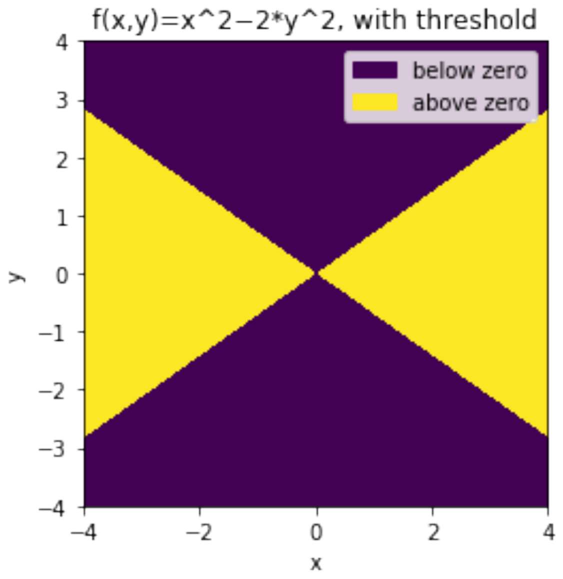
Hint: look at the `numpy` `meshgrid` (https://docs.scipy.org/doc/numpy-1.13.0/reference/generated/numpy.meshgrid.html) documentation

In [9]:
```
# import numpy as np
import time
import matplotlib.pyplot as plt

# Initialize variables for this exerise
x = np.arange(-4,4,8/2000)
y = np.arange(-4,4,8/2000)
X, Y = np.meshgrid(x, y)
thresh=0

def fun1(x,y):
    return x**2 - 2*y**2

# Nonvectorized implementation
z1_start = time.time()

z1=np.zeros_like(X)
for i, xi in enumerate(x):
    for j,yj in enumerate(y):
        z1[i,j] = fun1(xi,yj)

z1_end = time.time()
z1_int = z1_end - z1_start

# Vectorized implementation
z2_start = time.time()

z2 = fun1(X,Y)

z2_end = time.time()
z2_int = z2_end - z2_start

# Print the time for each and the speed increase
print('\nTime [sec] (non-vectorized): {0:.4f}\nTime [sec] (vectorized)
:     {1:.4f}'.format(z1_int,z2_int))
print('\nThe vectorized code is {0:.1f} times faster than the non-vect
orized code'.format(z1_int/z2_int))

# Plot the result
plt.imshow(z2, extent=(-4,4,-4,4))
plt.xlabel('x')
plt.ylabel('y')
plt.title('f(x,y)=x^2-2*y^2')
plt.show()
```

```
Time [sec] (non-vectorized): 6.2038
Time [sec] (vectorized):     0.0438

The vectorized code is 141.7 times faster than the non-vectorized co
de
```



In [59]:
```
import matplotlib.patches
z2[z2>0] = 1
z2[z2<0] = -1
im = plt.imshow(z2, extent=(-4,4,-4,4))
vals = np.unique(z2.ravel())
cols = [im.cmap(im.norm(value)) for value in vals]
patches = [matplotlib.patches.Patch(color=cols[0], label='below zero')
, matplotlib.patches.Patch(color=cols[1], label='above zero')]
plt.legend(handles = patches)
plt.xlabel('x')
plt.ylabel('y')
plt.title('f(x,y)=x^2-2*y^2, with threshold')
plt.show()
```



9

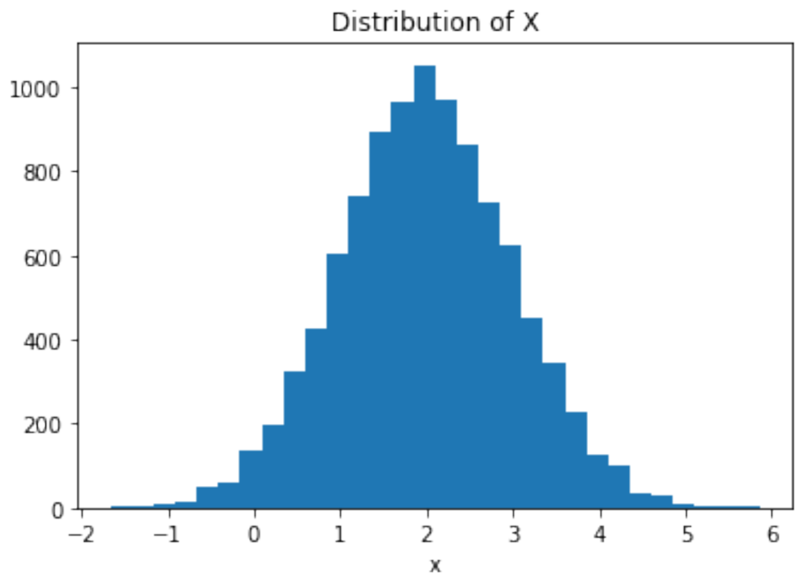
This exercise will walk through some basic numerical programming exercises.

1. Synthesize $n = 10^4$ normally distributed data points with mean $\mu = 2$ and a standard deviation of $\sigma = 1$. Call these observations from a random variable X, and call the vector of observations that you generate, $\mathbf{x}$.
2. Calculate the mean and standard deviation of $\mathbf{x}$ to validate (1) and provide the result to a precision of four significant figures.
3. Plot a histogram of the data in $\mathbf{x}$ with 30 bins
4. What is the 90th percentile of $\mathbf{x}$? The 90th percentile is the value below which 90% of observations can be found.
5. What is the 99th percentile of $\mathbf{x}$?
6. Now synthesize $n = 10^4$ normally distributed data points with mean $\mu = 0$ and a standard deviation of $\sigma = 3$. Call these observations from a random variable Y, and call the vector of observations that you generate, $\mathbf{y}$.
7. Plot the histogram of the data in $\mathbf{y}$ on a (new) plot with the histogram of $\mathbf{x}$, so that both histograms can be seen and compared.
8. Using the observations from $\mathbf{x}$ and $\mathbf{y}$, estimate $E[XY]$

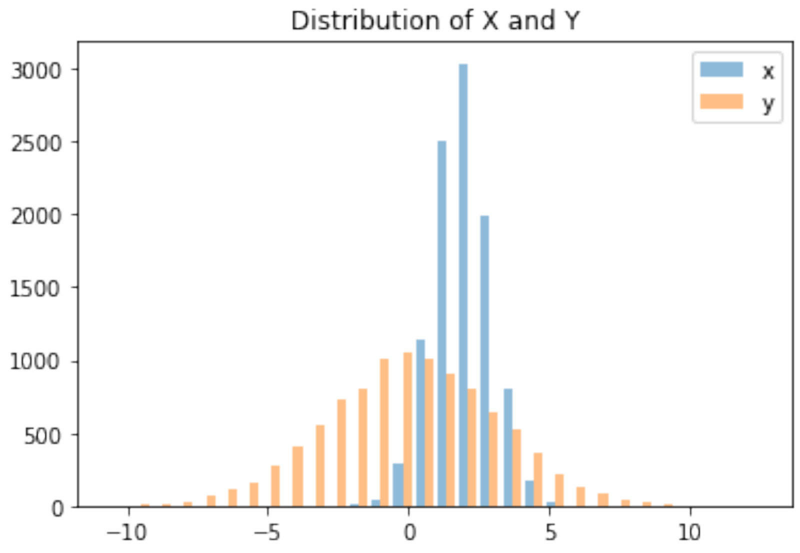
ANSWER

In [12]:
```
import numpy as np
import matplotlib.pyplot as plt

# 1
x = np.random.normal(2,1,10**4)

# 2
meanx = np.mean(x); print('The mean of x is {0:.4g}'.format(meanx))
stdx = np.std(x); print('The standard deviation of x is {0:.4g}'.forma
t(stdx))

# 3
plt.hist(x, bins=30)
plt.xlabel('x')
plt.title('Distribution of X')
plt.show()

# 4
x90 = np.percentile(x,90)
print('The 90th percentile of x is ', x90)

# 5
x99 = np.percentile(x,99)
print('The 99th percentile of x is ', x99)

# 6
y = np.random.normal(0,3,10**4)

# 7
plt.hist([x,y], bins=30, alpha=0.5,label=['x','y'])
plt.legend()
plt.title('Distribution of X and Y')
plt.show()
#???

# 8
exy = x@y/10**4
print('The estimate of E[XY] is ',exy)
```

```
The mean of x is 1.992
The standard deviation of x is 0.9947
```



```
The 90th percentile of x is  3.28179700229769
The 99th percentile of x is  4.3035926583464805
```



```
The estimate of E[XY] is  0.000406824505963138
```

10

Estimate the integral of the function $f(x)$ on the interval $0 \leq x < 2.5$ assuming we only know the following points from f:

Table 1. Dataset containing n=5 observations

x_i	**0.0**	**0.5**	**1.0**	**1.5**	**2.0**
y_i	6	7	8	4	1

In [44]:
```
import numpy as np

x10 = np.array([0.5]*5)
y10 = np.array([6,7,8,4,1])

inty = x10@y10
print('The estimation of the integral of f(x) on the interval is ', in
ty)
```

```
The estimation of the integral of f(x) on the interval is  13.0
```

Version Control via Git

11

Complete the Atlassian Git tutorial (https://www.atlassian.com/git/tutorials/what-is-version-control), specifically the following sections. Try each concept that's presented. For this tutorial, instead of using BitBucket, use Github. Create a github account here if you don't already have one: https://github.com/ (https://github.com/)

1. What is version control (https://www.atlassian.com/git/tutorials/what-is-version-control)
2. What is Git (https://www.atlassian.com/git/tutorials/what-is-git)
3. Install Git (https://www.atlassian.com/git/tutorials/install-git)
4. Setting up a repository (https://www.atlassian.com/git/tutorials/install-git)
5. Saving changes (https://www.atlassian.com/git/tutorials/saving-changes)
6. Inspecting a repository (https://www.atlassian.com/git/tutorials/inspecting-a-repository)
7. Undoing changes (https://www.atlassian.com/git/tutorials/undoing-changes)
8. Rewriting history (https://www.atlassian.com/git/tutorials/rewriting-history)
9. Syncing (https://www.atlassian.com/git/tutorials/syncing)
10. Making a pull request (https://www.atlassian.com/git/tutorials/making-a-pull-request)
11. Using branches (https://www.atlassian.com/git/tutorials/using-branches)
12. Comparing workflows (https://www.atlassian.com/git/tutorials/comparing-workflows)

For your answer, affirm that you either completed the tutorial or have previous experience with all of the concepts above. Do this by typing your name below and selecting the situation that applies from the two options in brackets.

ANSWER

*I, **Ruiqi Wang**, affirm that I have **completed the above tutorial.***

12

Using Github to create a static HTML website:

1. Create a branch in your `machine-learning-course` repo called "gh-pages" and checkout that branch (this will provide an example of how to create a simple static website using Github Pages (https://pages.github.com/))
2. Create a file called "index.html" with the contents "Hello World" and add, commit, and push it to that branch.
3. Submit the following: (a) a link to your github repository and (b) a link to your new "Hello World" website. The latter should be at the address https://[USERNAME].github.io/ECE590-assignment0 (https://[USERNAME].github.io/ECE590-assignment0) (where [USERNAME] is your github username).

ANSWER
(a) The link to my github repository https://github.com/Ruiqi22Wang/machine-learning-course (https://github.com/Ruiqi22Wang/machine-learning-course)
(b) The link to my new "Hello World" website https://Ruiqi22Wang.github.io/machine-learning-course (https://Ruiqi22Wang.github.io/machine-learning-course)

Exploratory Data Analysis

13

Here you'll bring together some of the individual skills that you demonstrated above and create a Jupyter notebook based blog post on data analysis.

1. Find a dataset that interests you and relates to a question or problem that you find intriguing
2. Using a Jupyter notebook, describe the dataset, the source of the data, and the reason the dataset was of interest.
3. Check the data and see if they need to be cleaned: are there missing values? Are there clearly erroneous values? Do two tables need to be merged together? Clean the data so it can be visualized.
4. Plot the data, demonstrating interesting features that you discover. Are there any relationships between variables that were surprising or patterns that emerged? Please exercise creativity and curiosity in your plots.
5. What insights are you able to take away from exploring the data? Is there a reason why analyzing the dataset you chose is particularly interesting or important? Summarize this as if your target audience was the readership of a major news organization - boil down your findings in a way that is accessible, but still accurate.
6. Create a public repository on your github account titled "machine-learning-course". In it, create a readme file that contains the heading "ECE590: Introductory Machine Learning for Data Science". Add, commit, and push that Jupyter notebook to the master branch. Provide the link to the that post here.

ANSWER
The link to the Jupyter notebook in the master branch of the machine-learning-course repository is:
https://github.com/Ruiqi22Wang/machine-learning-course/blob/master/Assignment1_Q13.ipynb
(https://github.com/Ruiqi22Wang/machine-learning-course/blob/master/Assignment1_Q13.ipynb)