MATH 10, SAMPLE MIDTERM

Thursday, 4 November 2021[1]

Name: Peter Anteater

Student ID: 12345678

Instructions: You are allowed to have handwritten notes on the notecard that was distributed in class (both sides). Your name must be at the top of both sides of the notecard. No other resources are allowed. Your work will be graded on clarity as well as correctness; if your code works correctly but is significantly more complicated than necessary, that will not receive full points. Points will not be deducted for small syntax errors if your meaning is clear. Cross out incorrect work. Do work in the space provided. Good luck.

Question	Score	Maximum
Total		50

[1] Version A

1. Short answer.

   a. Briefly explain an advantage of a NumPy array over a list, and an advantage of a list over a set.

   Numpy array over a list:
   Many vectorized operations; optimized for speed
   list over set:
   list can have duplicates & is ordered

   b. Rewrite the following code following the DRY principle (Don't Repeat Yourself). You should assume that x can't be any value other than 1, 2, 3, or 4.

```
if x == 1:
    print("You are the first respondent")
elif x == 2:
    print("You are the second respondent")
elif x == 3:
    print("You are the third respondent")
elif x == 4:
    print("You are the fourth respondent")
```

```
d= {1: "first", 2:"second", 3:"third", 4:"fourth" }
print(f"you are the {d[x]} respondent"}
```

   c. Assume A is a 2-dimensional array in NumPy. Rewrite the following code in a shorter way (do not use a for loop or a while loop).

```
_,n = A.shape
i = 0
while True:
    if i >= n:
        break # leave the while loop
    A[:,i] = 0
    i = i+2
```

```
A[:, 0:n:2] = 0
```

d. Assume you have a pandas DataFrame df already defined. Write code in Streamlit which gets `st.text_input` from a user, checks if their input is a column name in the DataFrame, and asks the user to try again if their input is not the name of a column in df.

```
input = st.text_input("Please enter a column name")
if input *not "" and input not in df.colums:
    st.write("Please pick a different column name")
```

* it's okay if you choose not to have this part

e. What is the "residual sum of squares" cost function in linear regression? Be sure to define any notation you use. What is this cost function used for?

Let m be the number of data points
Let $\vec{x}^{(i)} \in \mathbb{R}^n$ be the input of the i^{th} data point
Let $y^{(i)} \in \mathbb{R}$ be the output of the i^{th} datat point

Then, the loss function is

$$J = \frac{1}{m}\sum_{i=1}^{m}\Big(\overbrace{y^{(i)} - (\theta_0+\theta_1 x_1^{(i)}+\dots+\theta_n x_n^{(i)})}^{\text{vertical distance between actual \& expected value}}\Big)^2$$

minimization gives "best" linear approxi. of data w/ $\theta_0+\theta_1 x+\dots+\theta_n x_n$

coeff. in line approx data

f. Explain the following error. Explain both what causes the error, and what is a way to correct the error.

```
Input cell:
from sklearn.linear_model import LinearRegression
reg = LinearRegression()
reg.fit([5,1,2,0,0],[6,1,8,3,10])
```

→ need to reshape!

```
Output cell:
ValueError
----> 1 reg.fit([5,1,2,0,0],[6,1,8,3,10])
...
ValueError: Expected 2D array, got 1D array instead:
array=[5 1 2 0 0].
```

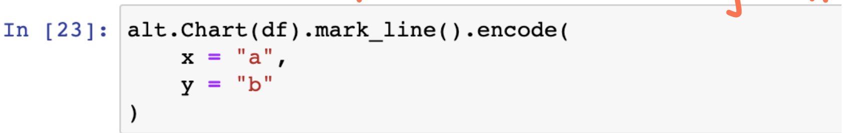
correction: [5,1,2,0,0] → np.reshape([5,1,2,0,0],(-1,1)]

ambiguous
df.iloc[:,3] max()
okay too!

g. How can you find the largest element in the third column of a pandas DataFrame?

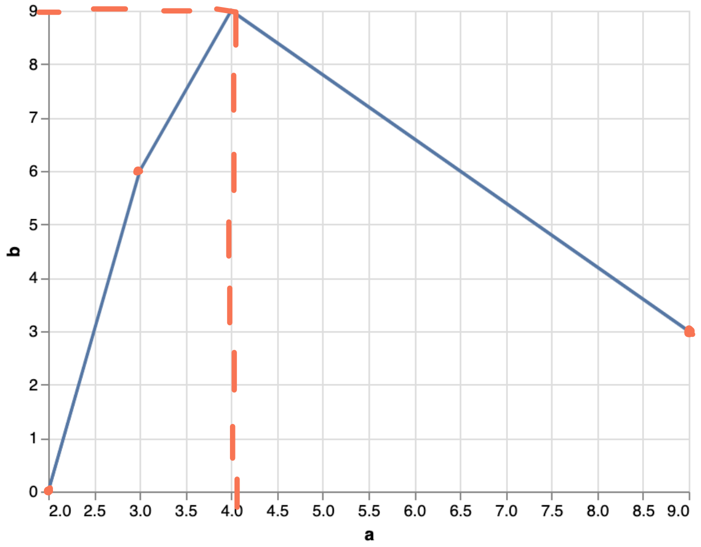
df.iloc[:,2].max()

h. Given the following example, what is a possible value of df? (Can you explain why there are other possible values?) df could have many other columns— we're just plotting "a" and "b" here



```
In [23]: alt.Chart(df).mark_line().encode(
             x = "a",
             y = "b"
         )
```

Out[23]:



(4.0, 9.0)
↑ a ↑ b

Possible df

	"a"	"b"
0	2.0	0
1	3.0	6
2	4.0	9
3	9.0	3

maybe not in this order either

2. Assume we have run the following code:

```
import numpy as np
import pandas as pd
rng = np.random.default_rng()
```

Describe in about one sentence each, what each of the following commands does.

```
A = 20*rng.random(size=(50,10)) - 8
```

creates a 50x10 array populated with random numbers living in [-8, 12)

```
df = pd.DataFrame(A)
```

creates a Pandas data frame from the array in part (a)

```
df.loc[df.loc[4] < 0]
```

→ I think this will throw an error

do we mean df.loc[df[4]<0]

↑ in this case, return subdataframe where entries in column 4 are smaller than 0.

Does that last command change the DataFrame df? Why or why not?

no, we did not write df=df.loc...

3. Assume A is some two-dimensional NumPy array. If we choose a random element from A, what is the probability that the element is strictly larger than 3 and less than or equal to 10? Write code to compute this exact probability. (If the array were too big to compute the probability exactly, how would you estimate the probability by using 10^6 "experiments"?)

```
B= A.reshape(-1) #Flatten A
n=len(B)
prob= sum((B>3) & (B≤10))/n
```

with Experiments

```
n=10**6
count=0
for x in range(n).
        y=rng.choice(B)
        if (y>3) and (y≤10):
                count=count +1
Prob= count/n
```

with Experiments, w/o for loops

```
n=10**6
B= A.reshape(-1)
C= rng.choice(B, size=n)
sum((C>3) & (C≤10))/n
```