Homework 0

Alex Shah

9/7/16

1 Python Proof 2.1

1.1 Python Installation Proof and Output

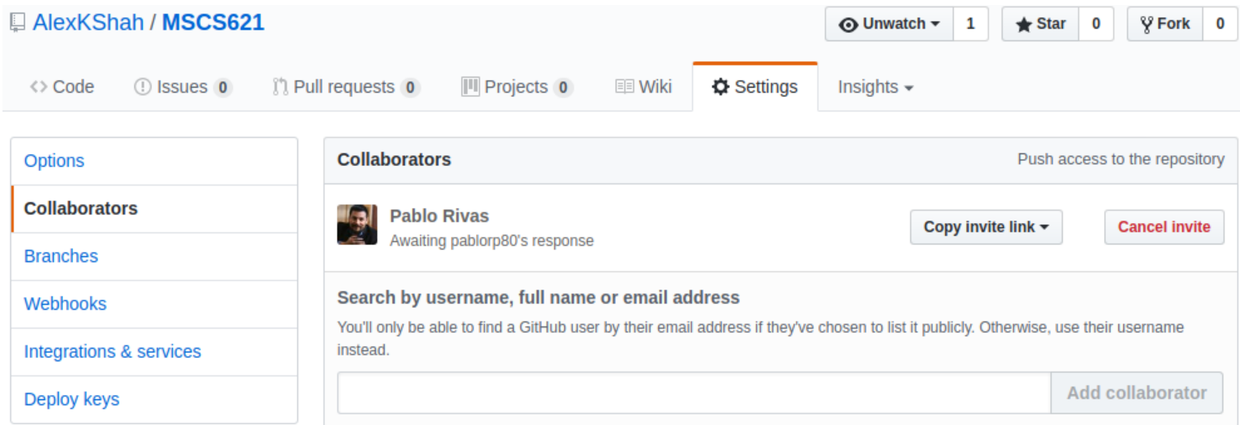
```
import sys
import numpy
import scipy
import sklearn
import matplotlib
import pandas
import tensorflow

print(sys.version)
print(numpy.__version__)
print(scipy.__version__)
print(sklearn.__version__)
print(matplotlib.__version__)
print(pandas.__version__)
print(tensorflow.__version__)
```

Output:

```
3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609]
1.11.0
0.19.1
0.19.0
1.5.1
0.20.3
1.3.0
```

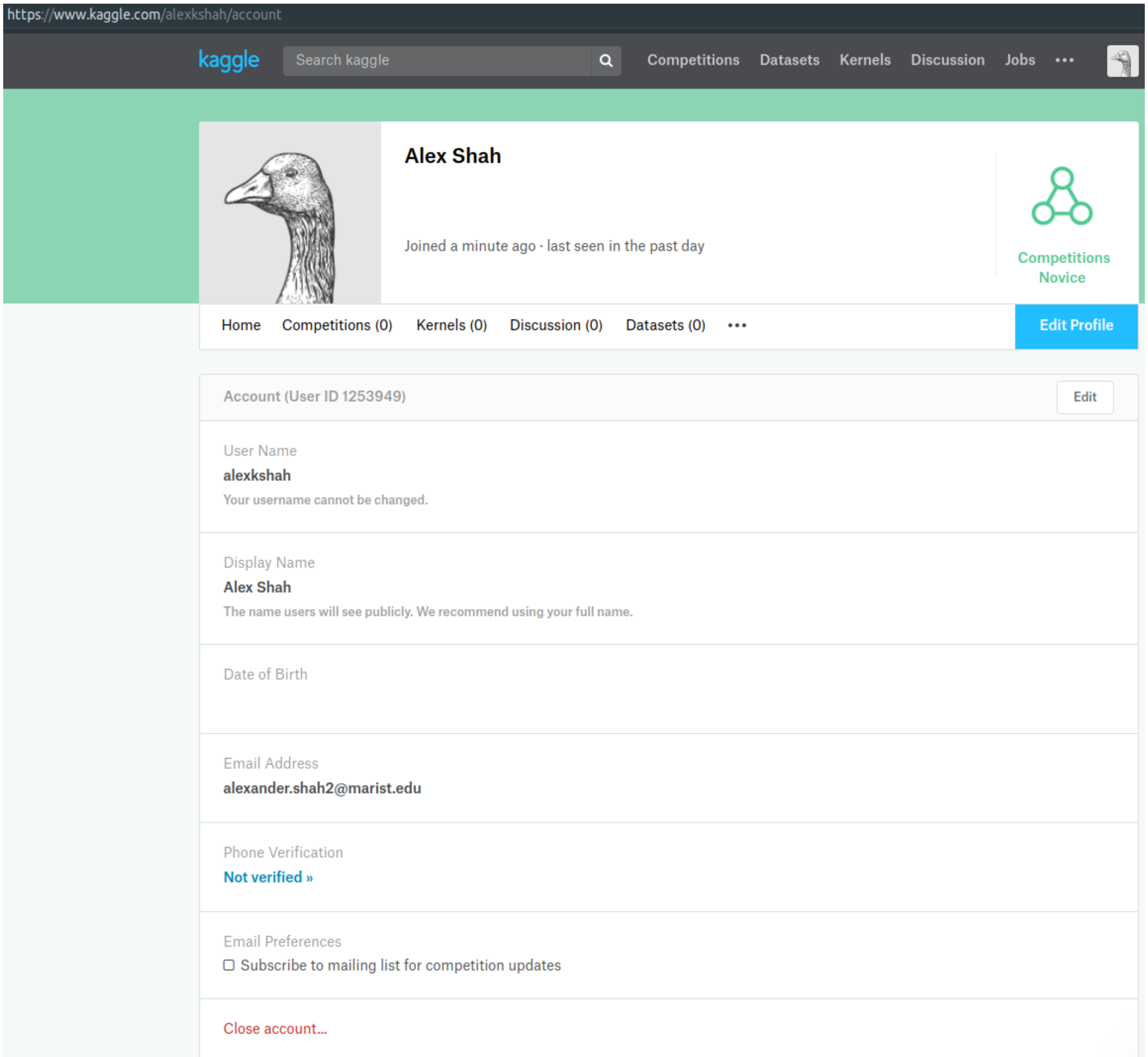
2 Github Proof 2.2

2.1 Github Collaborators



3 Kaggle Proof 2.3

3.1 Kaggle Account Page



4 Problems

4.1

This problem was solved using Scipy's fmin function, inverted to produce a maximum. The answer is 4.

```
# 4-1.py
# Alex Shah
# Homework 0

from scipy.optimize import fmin

def g(x): return -3.0*x**2+24*x-30

maximum = print(fmin(lambda x: -g(x), 0.0))

# Output:
# Optimization terminated successfully.
#          Current function value: -18.000000
#          Iterations: 29
#          Function evaluations: 58
# [ 4.] <-- Answer
```

4.2

Using x and y for readability, the partial derivative of f(x) with respect to x:

$$f(x) = 3x^3 - 3xy^2 + 4y - 8 \tag{1}$$

$$3 * 3 * x^2 - 3 * y^2 + 4 * 0 - 0 \tag{2}$$

$$9x^2 - 3y^2 \tag{3}$$

The partial derivative of f(x) with respect to y:

$$f(x) = 3x^3 - 3xy^2 + 4y - 8 \tag{4}$$

$$3 * 0 - 3x * 2y + 4 * 1 - 0 \tag{5}$$

$$6xy + 4 \tag{6}$$

4.3

```
# 4-3.py
# Alex Shah
# Homework 0

import tensorflow as tf
import numpy as np

# Matrices
a = np.array([[1.0,4,3], [2,-1,3]])
b = np.array([[-2,0,5], [0,-1,4]])
c = np.array([[1,0], [0,2]])

# tf const
a = tf.constant(a.astype(float))
b = tf.constant(b.astype(float))
c = tf.constant(c.astype(float))


# matrix cannot be multiplied as is,
# inner dimensions are different,
# transpose in order to multiply
At = tf.transpose(a)
Bt = tf.transpose(b)
Ci = tf.matrix_inverse(c)

with tf.Session() as sess:

    try: print(tf.matmul(a,b).eval())
    except Exception as err:
        print("\n A x B fails as dimensions are different:")
        print(err)

    print("\n At x B")
    res = tf.matmul(At,b).eval()
    print(res)

    print("\n Rank")
    print(np.linalg.matrix_rank(res))

    print("\n A x Bt")
    res2 = tf.matmul(a,Bt).eval()
    print(res2)

    print("\n A x Bt + C inverse")
    print(tf.add(res2, Ci).eval())
```

Output:

```
 A x B fails as dimensions are different:
Dimensions must be equal, but are 3 and 2 for 'MatMul' (op: 'MatMul') with
    input shapes: [2,3], [2,3].

 At x B
[[ -2.  -2.  13.]
 [ -8.   1.  16.]
 [ -6.  -3.  27.]]

 Rank
2

 A x Bt
[[ 13.    8.]
 [ 11.   13.]]

 A x Bt + C inverse
[[ 14.     8. ]
 [ 11.    13.5]]
```

4.4

$X \sim N(2,3)$

X is a random variable.

N(2,3) shows that the normal distribution has a mean, μ, of 2.

And a variance denoted as σ^2 which gives us $\sqrt{3}$ for σ.

In a normal distribution, the expected value is the mean (in this case 2) with a variance of $\sqrt{3}$.

Using these values, we can expect X within the range :

$$\mu - \sigma \leq X \leq \mu + \sigma \tag{7}$$

or

$$2 - \sqrt{3} \leq X \leq 2 + \sqrt{3} \tag{8}$$

for 1 standard deviation from the mean, or expected value.

Or within 2 standard deviations

$$2 - 2 * \sqrt{3} \leq X \leq 2 + 2 * \sqrt{3} \tag{9}$$