Topic02a : Prelim Problem Set I

Case 1

Represent the following representations into its vectorized form using LaTeX.

> **Problem 1.a. System of Linear Equations**
>
> $$\begin{cases} -y+z=\frac{1}{32} \\ \frac{1}{2}x-2y=0 \\ -x+\frac{3}{7}z=\frac{4}{5} \end{cases}$$
>
> **Problem 1.b. Linear Combination**
>
> $$\cos(\theta)\hat{i}+\sin(\theta)\hat{j}-\csc(2\theta)\hat{k}$$
>
> **Problem 1.c. Scenario**
>
> A conference has 200 student attendees, 45 professionals, and has 15 members of the panel. There is a team of 40 people on the organizing committee. Represent the *percent* composition of each *attendee* type of the conference in matrix form.

Express your answers in LaTeX in the answer area.

```
import numpy as np
import math
import matplotlib.pyplot as plt
%matplotlib inline
```

Problem 1.a. ANSWER: System of Linear Equations

```
x = np.array([[-1, 1, 1/32], [1/2, -2, 0], [-1, 3/7, 4/5]])
x
```

```
array([[-1.        ,  1.        ,  0.03125   ],
       [ 0.5       , -2.        ,  0.        ],
       [-1.        ,  0.42857143,  0.8       ]])
```

$$x=\begin{bmatrix} -1 & 1 & \frac{1}{32} \\ \frac{1}{2} & -2 & 0 \\ -1 & \frac{3}{7} & \frac{4}{5} \end{bmatrix}$$

Problem 1.b. ANSWER: Linear Combination

```
csc = 1/math.sin(2**180)
cos = math.cos(180)
sin = math.sin(180)

t = np.array([cos, sin, csc]) ##Values from the upper part
t
```

```
array([-0.59846007, -0.80115264,  1.00032034])
```

$$t = \begin{bmatrix} cos(\theta)\hat{i} \\ sin(\theta)\hat{j} \\ -csc(2\theta)\hat{k} \end{bmatrix}$$

Problem 1.c.ANSWER: Scenario

```
attendees = np.array([200, 45, 15, 40])

np.true_divide(attendees,300) * 100
```

```
    array([66.66666667, 15.        ,  5.        , 13.33333333])
```

$$\begin{array}{c} Students \\ Professionals \\ Panels \\ Committee \end{array} \begin{bmatrix} 200 \\ 45 \\ 15 \\ 40 \end{bmatrix} = \begin{bmatrix} 66.67\% \\ 15.00\% \\ 5.00\% \\ 13.33\% \end{bmatrix}$$

Case 2

> **Problem 2.a: Vector Magnitude**

> The magnitude of a vector is usually computed as:
> $$||v|| = \sqrt{a_0^2 + a_1^2 + \ldots + a_n^2}$$
> Whereas v is any vector and a_k are its elements wherein k is the size of v. Re-formulate $||v||$ as a function of an inner product. Further discuss this concept and provide your user-defined function.

> **Problem 2.b: Angle Between Vectors**

> Inner products can also be related to the Law of Cosines. The property suggests that:
> $$u \cdot v = ||u|| \cdot ||v|| \cos(\theta)$$
> Whereas u and v are vectors that have the same sizes and θ is the angle between u and v.

> Explain the behavior of the dot product when the two vectors are perpendicular and when they are parallel.

Problem 2.a. ANSWER: Vector Magnitude

The magnitude of a vector is determined by its length. The length of a vector must be calculated first before computing its magnitude. Velocity, displacement, force, momentum, and other quantities with direction are examples of vector quantities.

To re-formulate $||v||$ as a function of an inner product it would be presented as:

$$||v|| = \sqrt{\langle v|v\rangle}$$

```
def vect_mag(a):
  return np.sqrt(sum(k**2 for k in a))

v = np.array([1,2,3,4,5])
vector = vect_mag(v)
print(v, vector, sep = '\n')
```

```
[1 2 3 4 5]
7.416198487095663
```

Problem 2.b. ANSWER: Angle Between Vectors

Given that vectors are quantities with direction, there would be something that would indicate that direction. In this case, it's an angle.

The dot product when two vectors are perpendicular is null as $cos(90) = 0$.

$$u \cdot v = ||u|| \cdot ||v|| cos(90)$$
$$u \cdot v = ||u|| \cdot ||v|| 0$$
$$u \cdot v = 0$$

The dot product when two vectors are parallel is just the dot product of two vectors as $cos(0) = 1$.

$$u \cdot v = ||u|| \cdot ||v|| cos(0)$$
$$u \cdot v = ||u|| \cdot ||v|| 1$$
$$u \cdot v = u \cdot v$$

```
def angle_vectors(a,b):
  inner = np.inner(a, b)
  norms = np.linalg.norm(a) * np.linalg.norm(b)

  cos = inner/norms
  rad = np.arccos(np.clip(cos, -1.0, 1.0))
  deg = np.rad2deg(rad)
  print("Radiant: ", rad)
  print("Degree: ", deg)
```

```
a = np.array([10,11])
b = np.array([2,5])
print(a,b)
```

```
[10 11] [2 5]
```

```
angle_vectors(a,b)
```

```
Radiant:  0.35730868300810004
Degree:  20.472279519741925
```

Case 3

For the final cases analysis we will be looking at series of equations building up a single feed-forward computation of a logistic regression. The case will not require you to learn fully what is logistic regression.

$$X = \begin{bmatrix} —(x^{(1)})^T— \\ —(x^{(2)})^T— \\ \vdots \\ —(x^{(m)})^T— \end{bmatrix}, Y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}, \text{and } \theta = \begin{bmatrix} \theta^{(1)} \\ \theta^{(2)} \\ \vdots \\ \theta^{(m)} \end{bmatrix}$$

The dataset X has m entries with n features while Y is the vector containing the groud truths of a the entries of X, and θ are the parameters or weights of the vectors. We first compute the vector product of the dataset and the parameters as:

$$z = x^{(i)}\theta^{(i)} = X \cdot \theta$$

$$\text{Eq. 3.1}$$

We then solve for the hypothesis of the logistic regression alogrithm as:

$$h_\theta(x) = g(z)$$

$$\text{Eq. 3.2}$$

Where g is an acitvation function that maps the values of the hypothesis vector between a range of 0 and 1. We computed the activation as a sigmoid function:

$$g(z) = \frac{1}{1+e^{-z}}$$

$$\text{Eq. 3.3}$$

Finally we compute the loss of the logistic regression algorithm using J. Wheras $J(\theta)$ is a function that computes the logistic loss of the hypothesis with respect to the ground truths y. it is then computed as:

$$J(\theta) = \frac{1}{m}\sum_{i=0}^{m} = [-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))]$$

$$\text{Eq. 3.4}$$

> **Problem 3.a: Matrix Equivalences**
>
> In Eq. 1, z can also be solved as $X\cdot\theta$ which is the vectorized form of $x^{(i)}\theta^{(i)}$. However, it can also be expressed as $\theta^T\cdot X$. Prove the equality of $X\cdot\theta$ with $\theta^T\cdot X$ in this case.
>
> **Problem 3.b: Matrix Shapes**
>
> Determine the shape of h_θ if X has a shape of $(300, 5)$.
>
> **Problem 3.c: Vectorization**
>
> Express $J(\theta)$ into its vectorized form.
>
> **Problem 4.c: Computational Programming (Also Laboratory 2)**
>
> Encode Equations 3.1 to 3.4 as the class `LRegression` wherein:
>
> - `LRegression` should be instantiated with a dataset X, a ground truth vector y, and a parameter vector θ. Each parameter should have a data type of `numpy.array`.
> - It should further have `methods` reflecting to at least the four (4) aforementioned equations. Each should have a return value.

Problem 3.a. ANSWER: Matrix Equivalences

Unlike scalar multiplication, multiplication involving matrices is not commutative. However, commutative property is allowed when getting the dot product between two vectors.

It means that $A\cdot B = B\cdot A$ can be applied even if one of the vectors is transposed.

In problem 3.a, it is required to prove the equality of $X\cdot\theta$ with $\theta^T\cdot X$ given that z can also be solved as $X\cdot\theta$ in equation 3.1

$$X = \begin{bmatrix} -(x^{(1)})^T- \\ -(x^{(2)})^T- \\ \vdots \\ -(x^{(m)})^T- \end{bmatrix}, \theta = \begin{bmatrix} \theta^{(1)} \\ \theta^{(2)} \\ \vdots \\ \theta^{(m)} \end{bmatrix}$$

By computing the vector product of the dataset and the parameters

$$z = X \cdot \theta$$

we get:

$$z = \begin{bmatrix} —(x^{(1)})^T\theta— \\ —(x^{(2)})^T\theta- \\ \vdots \\ —(x^{(m)})^T\theta— \end{bmatrix}$$

By applying commutative property to $z = X \cdot \theta$ which is the vectorized form of $x^{(i)^T}\theta^{(i)}$, we can say that it is equal to

$$z = \theta^T \cdot X$$

$$z = \begin{bmatrix} —\theta^T(x^{(1)})— \\ —\theta^T(x^{(2)})- \\ \vdots \\ —\theta^T(x^{(m)})— \end{bmatrix}$$

Hence,

$$z = \begin{bmatrix} —(x^{(1)})^T\theta— \\ —(x^{(2)})^T\theta- \\ \vdots \\ —(x^{(m)})^T\theta— \end{bmatrix} = \begin{bmatrix} —\theta^T(x^{(1)})— \\ —\theta^T(x^{(2)})- \\ \vdots \\ —\theta^T(x^{(m)})— \end{bmatrix}$$

Problem 3.b. ANSWER: Matrix Shapes

The output derived from logic regression hypothesis h_θ is used in determining the estimated probability. In this problem, the shape of h_θ must be determined given that X has a shape of $(300, 5)$

Logic Regression Hypothesis:

$$h_\theta(x) = g(z)$$

wherein

$$g(z) = \frac{1}{1+e^{-z}}$$

and

$$z = X \cdot \theta$$

if X has a shape of $(300, 5)$ then

$$X = \begin{bmatrix} x^1 & x^{301} & x^{601} & x^{901} & x^{1201} \\ x^2 & x^{302} & x^{603} & x^{904} & x^{1205} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x^{300} & x^{600} & x^{900} & x^{1200} & x^{1500} \end{bmatrix}, \theta = \begin{bmatrix} \theta^1 \\ \theta^2 \\ \vdots \\ \theta^k \end{bmatrix}$$

It can also be written as

$$h_\theta = g(X.\theta)$$

Getting h_θ by multiplying X that has a shape of (n, m) or (300,5) to its parameter vector θ that has a dimension of (m, k) or (300,1) would result to getting the scalar value of the inner product $(300, 1)$

$$h_\theta = \begin{bmatrix} \theta^1(x^1) \\ \theta^2(x^2) \\ \vdots \\ \theta^{300}(x^{300}) \end{bmatrix}$$

Problem 3.c. ANSWER: Vectorization

$$\begin{bmatrix} \frac{\partial J}{\partial\theta_0} \\ \frac{\partial J}{\partial\theta_1} \\ \frac{\partial J}{\partial\theta_2} \\ \vdots \\ \frac{\partial J}{\partial\theta_n} \end{bmatrix} = \frac{1}{m}\begin{bmatrix} \sum_{i=1}^m[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] \\ \sum_{i=1}^m[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] \\ \sum_{i=1}^m[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] \\ \vdots \\ \sum_{i=1}^m[-y^{(i)}\log(h_\theta(x^{(i)})) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] \end{bmatrix}$$

$$\frac{\partial J(\theta)}{\partial\theta_0} = \frac{1}{m}\sum_{i=1}^{m}(h_0(x^{(i)}) - y^{(i)})x_j^{(i)}$$

$$\frac{\partial J(\theta)}{\partial\theta_0} = \left(\frac{1}{m}\sum_{i=1}^{m}(h_0(x^{(i)}) - y^{(i)})x_j^{(i)}\right) + \frac{\lambda}{m}\theta_j$$