Foundations of Data Science (CS F320)
Assignment 3

Submission Date: TBD on 30th Oct class
Points: 30

The Problem

Bob and Lisa would like to find out the probability of getting head, μ, of a biased coin. They are excited to get a probability distribution of μ but not just a point estimate.

You will have to randomly generate a dataset for this problem wherein the size of the dataset should be around 160 and $\mu_{ML} \notin (0.4,0.6)$. μ_{ML} is maximum likelihood estimator of the data that is being generated by you.

As we know, Posterior distribution $\propto$ Likelihood distribution x Prior distribution

$$P(\mu \mid D, a, b) \propto P(D \mid \mu) \times P(\mu \mid a, b)$$

Our goal is to find the distribution followed by the mean of the coin tosses after observing the data points.

As we know, coin tossing follows a Bernoulli distribution. It's probability density function is given by

$$\mathrm{Bern}(x \mid \mu) = \mu^x (1-\mu)^{1-x}$$

where μ is the mean of the Bernoulli distribution.

Thus, for a dataset D of N points, we get the likelihood function as

$$p(\mathcal{D}|\mu) = \prod_{n=1}^{N} p(x_n|\mu) = \prod_{n=1}^{N} \mu^{x_n}(1-\mu)^{1-x_n}.$$

We will take the prior to be a beta distribution. The PDF for a beta distribution is given by

$$\mathrm{Beta}(\mu|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\mu^{a-1}(1-\mu)^{b-1}$$

where a and b are the parameters and Γ is the gamma function. Choose appropriate a and b such that the mean of the prior is 0.4.

As seen in class, we know how to find out the posterior distribution given prior distribution and likelihood function. There are two approaches to find out the posterior distribution – one is to use all 160 data points at one go and another is use each data point sequentially.

Part A: Sequential Learning

Bob would like to solve this problem using this approach. One example will be taken at a time and hence the likelihood function will have only one term in it's product. As with any sequential learning problem, this posterior after viewing the first example becomes the prior for the next example. Thus, you start off with a prior and will observe how examples coming in change the appearance of the prior. You will have to make plots of the prior at each stage (similar to Fig 2.2 in Bishop). Combine all the plots together to make a GIF.

Part B

This is Lisa's view of the problem. In this case, the likelihood will be over the entire dataset. Compute the posterior after viewing the entire dataset at once and make plot of the posterior for the same.

Part C

Comment on the differences/similarities between the two models obtained.
The size of the dataset has been restricted to 160 data points. What happens if more points are added? What would the posterior distribution look like if $\mu_{ML} = 0.5$?

Reference: You may go through 71 – 74 of Bishop's book on Pattern Recognition and Machine Learning

Implementation Details

You will have to implement this in python using numpy, scipy and matplotlib.
No other libraries are allowed.

Programming tips in Python:

1.numpy.random has a large number of functions to generate random numbers. There is a function "numpy.random.randint" to generate integers within a range and a function "numpy.random.random" to generate real numbers within [0, 1]. (This will be used for plotting)

2. A small code snippet is given below on how to generate an array of 5 numbers within the range [0, 1] and to exponentiate them to the fourth power. This will be used to compute the Beta function PDF.

```
>>> import numpy as np
>>> np.random.random ( [ 5 , 1 ] )
>>> n=np.random.random ( [ 5 , 1 ] )
>>> n
```

```
array ( [ [ 0 .94652734] ,
[0.02478432] ,
[0.58019283] ,
[0.39018742] ,
[0.00348265]])
>>> n∗∗4
```

3. For computing the value of the gamma function, you can use scipy.special.gamma function from the scipy module.

```
>>> import scipy.special
>>> scipy.special.gamma ( 5 )
24.0
>>> scipy.special.gamma (0.1)
9.513507698668732
>>> scipy.special.gamma (0.01)
99.43258511915059
```

4. Given below is code in matplotlib to plot the probability density function. x are points in the range [0, 1] and y is the computed pdf value.

```
import matplotlib.pyplot as plt
import matplotlib.ticker as mt

def plot ( x , y , i ) :
plt.figure (1)
ax = plt.gca ( )
mx = 10
ax.setylim( 0 , mx)
ax.xaxis.setmajorlocator (
mt.FixedLocator ( [ i ∗ 0.1 for i in range( 1,1 ) ] ) )
ax.plot( x , y , linewidth =0, marker= '.' , markersize =4)
#To display the plot on the screen
#plt.show( )
#To save the plot to a file
#plt.savefig("Fig / { } . png " . format( i ))
plt.close (1)
```

Submission

- ✓ Gather all your code into a single .py file.

- ✓ Make a document with your answer for Part C.
- ✓ Also, include the plots for Part A and Part B in this document.
- ✓ You can put plots after every 20 examples for Part A, and the final plot for Part B.
- ✓ Zip the code and this document and name it with your ID numbers.

For Queries

Itiyala Sonika (f20160099@hyderabad.bits-pilani.ac.in)