Goal: practice derivatives of Probability distributions for MLE and MAP

Given data, we want to estimate some parameters to describe the data. Once we have values for fitting parameters we can make simple predictions. The simplest parameter fitting exercise are for distributions of random variables.

One of the simplest distributions is a Bernoulli distribution, a RV with 2 states. We can use a fair coin, a categorical variable for Sex(M/F), a T/F feature, a buy/no buy decision, etc… as examples and sources of data for Bernoulli distributions.

There are 2 main strategies for fitting parameters, MLE and MAP.

Both of these strategies assume iid, independent and identically distributed data. This means all the data or samples of the data have the same probability mass function if it is discrete data or pdf if it is continuous.

**Maximum likelihood estimation:**
Find the parameters which make the data the most probable.

Find the theta which fits the following:

$argmax_\theta P(D|\theta).$

**MLE Bernoulli:**
We are going to standardize using the Tom Mitchell book to avoid confusion with other docs and to verify our work:
Mitchell uses p instead of theta.

The likelihood function is the product of the probabilities of the data.

$$L(\theta) = \prod_{i=1}^{n} \theta^{X_i}(1-\theta)^{1-X_i}$$

Before we take the derivative or a product of n terms using the chain rule, it is easier to transform the product to a sum by adding a log to the expression and computing argmin.

$$-LogLikelihood(\theta) = -argmin_\theta \sum_{i=1}^{n} log(\theta^{X_i}(1-\theta)^{1-X_i})$$

$$-argmin_\theta \sum_{i=1}^{n} X_i(log(\theta)) + (1-X_i)log(1-\theta)$$

$$-argmin_{\theta} \sum_{1}^{n} (Ylog(\theta) + (n - Y)log(1 - \theta))$$

Where $Y = \sum_{i}^{n} (X_i)$

To find the maximum take the derivative WRT $\theta$ and set to 0. We are going to drop the minus sign because we set the derivative to 0. Solve for $\sum X_i$ or $Y$.

$$\frac{\partial LogLikelihood(\theta)}{\partial \theta} = \frac{\partial (Ylog(\theta) + (n - Y)log(1 - \theta))}{\partial (\theta)}$$

$$\frac{\partial log(\theta)}{\partial (\theta)} = \frac{1}{\theta} d\theta$$

$$\frac{\partial (Ylog(\theta) + (n - Y)log(1 - \theta))}{\partial (\theta)} = \frac{Y}{\theta} d\theta - \frac{n - Y}{1 - \theta} d\theta = 0$$

$$\frac{Y}{\theta} = \frac{n - Y}{1 - \theta}$$

$$(Y)(1 - \theta) = (n - Y)(\theta)$$

$$Y - Y\theta = n\theta - Y\theta$$

$$Y = n\theta$$

$$\frac{Y}{n} = \theta$$

After replacing the summation for Y we see $\theta$ is the mean of $X_i$

**MLE Examples:**
Given the following datasets what is the MLE estimate?

H=1, T=0,

{H,H.H}, p = 3/3 = 1.0, predict H for next event
{H,T,H,T}, p= 2/4 = 0.5, predict H or T for next event
{T,T,T,H}, p=1/4=0.25, predict .25% of H. Predict 1H and 3T for next 4. Order is not specified.

Let's try the same example but starting with the PMF. This allows us to write a python program to automate the process.
What is the MLE for $\theta$ for a coin with {H,T,T,T,H}? We know this is 2/5 from our above examples. But we can also start from the PMF and take the derivative.

$$L(\theta) = \prod_{i=1}^{n} p^{X_i}(1-p)^{1-X_i}$$

$$L(\theta) = \theta^2(1-\theta)^3$$

taking derivative WRT theta and set to 0 to solve for theta gets us….

$$\frac{\partial L(\theta)}{\partial \theta} = 2\theta(1-theta)^3 + \theta^2(3)(1-\theta)^2(-1)$$

$$0 = 2\theta(1-\theta)^3 - 3\theta^2(1-\theta)^2$$

$$0 = 2(1-\theta) - 3\theta$$

$$0 = 2 - 2\theta - 3\theta$$

$$0 = 2 - 5\theta$$

$$theta = \frac{2}{5}$$

```
import torch,numpy as np
from torch.autograd import Variable

X_i = np.array([0.,1.,1.,0.,0.])
x =
Variable(torch.from_numpy(X_i)).type(torch.FloatTensor)
p = Variable(torch.rand(1),requires_grad=True)

learning_rate = .001
for a in range(1000):
    like = -torch.sum(torch.log(x*p+(1-x)*(1-p)))

    like.backward()
    if a%10==0:
        print("L(theta):",like.data.numpy(),
"p=",p.data.numpy(), "dL/dp:",p.grad.data.numpy())
    p.data -= learning_rate * p.grad.data
    p.grad.data.zero_()
#we are converged to .40 or 2/5 at 40 iterations
```

```
L(theta): 5.51934 p= [0.07067007] dL/dp: [-25.072392]
L(theta): 3.9385862 p= [0.19222593] dL/dp: [-6.690514]
L(theta): 3.6525915 p= [0.2458423] dL/dp: [-4.157349]
L(theta): 3.5274668 p= [0.28134567] dL/dp: [-2.9342246]
L(theta): 3.461875 p= [0.30705303] dL/dp: [-2.184197]
L(theta): 3.4245417 p= [0.32644662] dL/dp: [-1.672586]
L(theta): 3.4022973 p= [0.34141573] dL/dp: [-1.3027349]
L(theta): 3.3886638 p= [0.353134] dL/dp: [-1.0258265]
L(theta): 3.3801515 p= [0.36239296] dL/dp: [-0.8137784]
L(theta): 3.3747687 p= [0.36975554] dL/dp: [-0.64892197]
L(theta): 3.371334 p= [0.37563658] dL/dp: [-0.5194011]
L(theta): 3.3691285 p= [0.38034973] dL/dp: [-0.41687727]
L(theta): 3.3677044 p= [0.38413614] dL/dp: [-0.33528137]
L(theta): 3.3667822 p= [0.38718358] dL/dp: [-0.27007866]
L(theta): 3.366183 p= [0.38963965] dL/dp: [-0.21781874]
L(theta): 3.365793 p= [0.39162138] dL/dp: [-0.17583418]
L(theta): 3.3655386 p= [0.39322165] dL/dp: [-0.1420455]
L(theta): 3.3653724 p= [0.3945147] dL/dp: [-0.11481619]
L(theta): 3.3652644 p= [0.39556015] dL/dp: [-0.09284782]
L(theta): 3.3651934 p= [0.39640564] dL/dp: [-0.07511139]
L(theta): 3.3651466 p= [0.39708978] dL/dp: [-0.06077909]
L(theta): 3.3651164 p= [0.39764342] dL/dp: [-0.04919338]
L(theta): 3.3650963 p= [0.39809155] dL/dp: [-0.03982353]
L(theta): 3.3650832 p= [0.39845434] dL/dp: [-0.03224325]
L(theta): 3.3650746 p= [0.3987481] dL/dp: [-0.02610874]
L(theta): 3.365069 p= [0.39898598] dL/dp: [-0.02114296]
L(theta): 3.3650656 p= [0.3991786] dL/dp: [-0.01712418]
L(theta): 3.365063 p= [0.3993346] dL/dp: [-0.01387024]
L(theta): 3.3650615 p= [0.399461] dL/dp: [-0.01123381]
```

**MAP Estimation**

MAP estimation is the first step in bayesian methods which finds a set of parameters $\theta$ which satisfies:

$$argmax_\theta P(\theta \mid D)$$

To find theta we rewrite the above MAP estimator using Bayes rule:

$$\theta^{MAP} = argmax_\theta P(\theta \mid D) = argmax_\theta \frac{P(D \mid \theta)P(\theta)}{P(D)}$$

We ignore P(D) since it has no theta parameter and will fall out when taking a derivative.
Take derivative and set to 0 to solve for parameters.

We assume a prior distribution as a beta distribution of 2 parameters, which are also known as alpha and beta in the python APIs. We will explore other priors such as a uniform distribution in another handout.

$$P(\theta) = Beta(\beta_0, \beta_1) = \frac{\theta^{\beta_1 - 1}(1-\theta)^{\beta_0 - 1}}{B(\beta_0, \beta_1)}$$

where $B(\beta_0, \beta_1)$ is a normalization constant to make sure the integral is 1.

From the MLE estimate section we have seen the PMF of the Bernoulli as:
$P(D \mid \theta) = \theta^{\alpha_1}(1-\theta)^{\alpha_0}$
Where $\alpha_1$ is the number of heads and $\alpha_0$ is the number of tails or vice-versa using the notation in the Mitchell pdf. You can substitute $\alpha_1$ with 1-$\alpha_0$ or $\alpha_1$ with 1-$\alpha_0$.

$$\theta^{MAP} = argmax_\theta P(\theta \mid D) = argmax_\theta \theta^{\alpha_1}(1-\theta)^{\alpha_0} \frac{\theta^{\beta_1 - 1}(1-\theta)^{\beta_0 - 1}}{B(\beta_0, \beta_1)}$$

$$= argmax_\theta \frac{\theta^{\alpha_1+\beta_1-1}(1-\theta)^{\alpha_0+\beta_0-1}}{B(\beta_0,\beta_1)}$$

we can remove the normalizing constant $B(\beta_0,\beta_1)$ because it does not have any theta terms in it and will fall out when taking the derivative and setting to 0.

$$= argmax_\theta \theta^{\alpha_1+\beta_1-1}(1-\theta)^{\alpha_0+\beta_0-1}$$

$$\frac{\partial \theta^{MAP}}{\partial \theta} = (\alpha_1+\beta_1-1)\theta^{\alpha_1+\beta_1-2}(1-\theta)^{\alpha_0+\beta_0-1} + \theta^{\alpha_1+\beta_1-1}(\alpha_0+\beta_0-1)(1-\theta)^{\alpha_0+\beta_0-2}(-1)$$

$$= (\alpha_1+\beta_1-1)(1-\theta) - (\alpha_0+\beta_0-1)(\theta)$$

$$(\alpha_1+\beta_1-1)(1-\theta) = (\alpha_0+\beta_0-1)(\theta)$$

$$(\alpha_1+\beta_1-1) - (\alpha_1+\beta_1-1)\theta = (\alpha_0+\beta_0-1)(\theta)$$

$$(\alpha_1+\beta_1-1) = (\alpha_1+\beta_1-1)\theta + (\alpha_0+\beta_0-1)\theta$$

$$(\alpha_1+\beta_1-1) = \theta(\alpha_1+\beta_1-1+\alpha_0+\beta_0-1))$$

$$\theta = \frac{\alpha_1+\beta_1-1}{\alpha_1+\beta_1-1+\alpha_0+\beta_0-1}$$

**Beta Distribution**

There are 2 parameters to the beta distribution which control the shape of the distribution. The beta distribution includes the uniform distribution which we will need later. The values of alpha and beta are graphed to help pick the right value for priors.

```
from scipy.stats import beta
import matplotlib.pyplot as plt
import numpy as np

#beta is used for uniform distribution also

fig, ax = plt.subplots(1, 1)

mean, var, skew, kurt = beta.stats(a, b, moments='mvsk')
#x = np.linspace(beta.ppf(0.01, a, b),
#                beta.ppf(0.99, a, b), 100)
x = np.linspace(0,1,1000)
#[xmin, xmax, ymin, ymax]
plt.axis([0,1.0 , 0, 2.5])
mean1, var1, skew1, kurt1 = beta.stats(2., .5, moments='mvsk')
ax.plot(x, beta.pdf(x, 2., .5),
       'blue', lw=5, alpha=0.6, label='beta1')

ax.plot(x, beta.pdf(x, 1, 3),
       'red', lw=5, alpha=0.6, label='beta2')
mean2, var2, skew2, kurt2 = beta.stats(1., 3., moments='mvsk')
ax.plot(x, beta.pdf(x, .5, .5),
       'green', lw=5, alpha=0.6, label='beta3')
mean3, var3, skew3, kurt3 = beta.stats(.5, .5, moments='mvsk')
ax.plot(x, beta.pdf(x, 2, 2),
       'orange', lw=5, alpha=0.6, label='beta4')
mean4, var4, skew4, kurt4 = beta.stats(2., 2., moments='mvsk')

ax.plot(x, beta.pdf(x, 1, 1),
       'purple', lw=5, alpha=0.6, label='uniform')
mean5, var5, skew5, kurt5 = beta.stats(2., 2., moments='mvsk')

print(f"blue mean1:{mean1:.2f},var1:{var1:.2f},skew1:{skew1:.2f},kurt1:
{kurt1:.2f}")
print("red mean2:%.2f,var2:%.2f,skew2:%.2f,kurt2:%.2f" %
(mean2,var2,skew2,kurt2))
print("green mean3:%.2f,var3:%.2f,skew3:%.2f,kurt3:%.2f:" %
(mean3,var3,skew3,kurt3))
print("orange mean4:%.2f,var4:%.2f,skew4:%.2f,kurt4:%.2f:" %
(mean4,var4,skew4,kurt4))
print("purple uniform mean5:%.2f,var5:%.2f,skew5:%.2f,kurt5:%.2f:" %
(mean5,var5,skew5,kurt5))
```

```
blue mean1:0.80,var1:0.05,skew1:-1.25,kurt1:0.82
red mean2:0.25,var2:0.04,skew2:0.86,kurt2:0.10
green mean3:0.50,var3:0.12,skew3:0.00,kurt3:-1.50:
orange mean4:0.50,var4:0.05,skew4:0.00,kurt4:-0.86:
purple uniform mean5:0.50,var5:0.05,skew5:0.00,kurt5:-0.86:
```