The exam will focus on conceptual and algorithmic questions, and will assess material covered from Lecture 1 to Lecture 17.

Allowed Material:

- You are allowed 1-page letter-sized front and back of **formulas** (handwritten or typed).
    - Other than labeling your equations, **DO NOT** write down definitions, pseudo-code or any algorithm description
- You are allowed a scientific calculator, but will probably not need it
- Bring clean paper to write your answers (go ahead and get extra, just in case)
    - Some questions will be typed directly in the quiz, others will be solved on paper
- Bring a second device (e.g., phone, tablet or other) with the CamScanner (https://www.camscanner.com/) or Scannable (https://evernote.com/products/scannable) app installed. This device is only to be used at the end of the exam to take pictures of your **handwritten solutions AND formula sheet**.
- **TOTAL TIME:** 2 hours + 15 minutes

Communications between students or anyone else is considered cheating. Turn off all Slack notifications and other communications channels!

Ten (10) minutes prior to the time you are scheduled to take the exam, you will be able to see the Exam quiz in Canvas.

Make sure your testing environment is comfortable, and start whenever you are ready.

The total allocated time 2 hours + 15 minutes, the "clock" starts counting when you begin to take the quiz.

Lecture 18 - Exam 1 Review Session

Thank you for posting your questions!

I will be answering the ones with the most likes first.

In [3]:

```
import numpy as np
import numpy.random as npr
import scipy.stats as stats

import matplotlib.pyplot as plt
%matplotlib inline
plt.style.use('seaborn-colorblind')

import pandas as pd

from IPython.display import Image
```

Post 1

I had a lot of trouble with some of the questions of Homework 2 part 1, so covering the questions on that would be helpful.

The Bayesian interpretation question and the k-means question are two that come to mind.

Post 8

Could you also go over the last problem in Hw2 part 1 please?

I think both these questions can be answered simultaneously. If anything is left out, let me know!

Let's take a look at the questions here: [https://github.com/Fundamentals-of-Machine-Learning-F20/Assignment-Solutions (https://github.com/Fundamentals-of-Machine-Learning-F20/Assignment-Solutions)](https://github.com/Fundamentals-of-Machine-Learning-F20/Assignment-Solutions), and discuss the solutions.

Post 2

I don't necessarily have any specific content questions but can we rehash basically how to prepare for the exam, i.e, what content is most useful to study whats least useful as well the format of the exam?

Post 3

I think that a short overview of all the algorithms that we should know would be extremely helpful. Having a one-stop lecture to assess how well we know everything would be really helpful for knowing areas of studying that we need to focus on. I also struggled with HW2 part 3. I think an overview of that question would be very beneficial as well. Thank you.

I think both these questions can be answered simultaneously. If anything is left out, let me know!

- Let's take a look at the list below and discuss the material.

All material covered in the lecture notes and homework assignments is useful. All the questions will be designed based on what we learned and discussed in class.

To better prepare for the midterm: read, review and redo all the derivations presented in the lecture notes and assignment solutions (at least Part 1).

Here is a highlight of the topics to be assessed in the Midterm exam:

Topic 1: What is Machine Learning? Lectures 1-2

- Define and differentiate ML from Deep Learning and Artificial Intelligence.
- Design at least one example for each type
- How do we define *learning* in Machine Learning?
- Design the flowchart for supervised learning.

Topic 2: Regression Lectures 3-5

- What type of learning does regression use?
- What is regression (and how is it different from e.g. classification and clustering)?
- Define the linear regression model with basis functions.
- What is the Least Squares solutions for regression with and without regularization term?
- Derive the solution for the polynomial regression, and basis function regression model.
- Name at least one application of regression.

Topic 3: Experimental Design Lectures 5-6

- Why experimental design important?
- What are the steps one takes to *design an experiment* in Machine Learning?
- How do you use k-fold Cross-validation?

Topic 4: Generalization and Regularization Lectures 6-7

- What is overfitting and underfitting?
- What is the relationship of number of samples, model complexity and overfitting?
- What is the bias-variance trade-off?
- What can you do to prevent overfitting?
- What is regularization and how is it used in the design of a predictive model?
- Describe the difference between L1- and L2-norm regularizers.

Topic 5: Parameter Estimation Lectures 8-10

- What is MLE?
- What is MAP?
- What is the difference between MLE and MAP?
- Discuss the effects of the prior belief in the estimation of MAP solutions.
- What is a conjugate prior and how can we use it in online update of the model's parameters.
- What is the Bayesian interpretation of the polynomial regression model with and without regularization term.
- Derive the MLE and MAP solutions for the Bernoulli-Beta example in Lecture 9.
- Review the final result for the MAP solution for the Gaussian-Gaussian example in Lecture 10.
- Review the effects of prior and likelihood parameters in the online update code example presented in Lecture 10.

Topic 6: Classification Lectures 11-12

- How can we describe the two different types of classification?
- Describe Naive Bayes Classifier.
- Name at least one application of generative classification.

Topic 7: Data Likelihood Model and Clustering Lectures 12-16

- What are Mixture Models?
- What are Gaussian Mixture Models (GMMs) used for?
- How can we optimize a GMM likelihood model?
- Describe the general steps of the Expectation-Maximization algorithm.
- What type of optimization does EM use, and, does it converge to *global* solution?
- Describe the pseudo-code for GMM optimized using the EM algorithm.
- What is Clustering?
- Describe the pseudo code for K-Means.
- Why is GMM considered soft clustering?
- Compare and discuss advantages/disadvantages of GMM vs k-Means.
- Be able to analyze clustering results and identify which technique generated which result (Lecture 16 + Short Assignment 3).

Topic 8: Cluster Validity Lecture 17

- Discuss challenges of validating an unsupervised model.
- What type of index criteria are there when defining cluster validity metrics?
- Describe the silhouette index.
- Describe the rand index.
- Be able to analyze a cluster validity result and determine how many clusters one should select.

Post 4

Could you please go over online updating using conjugate prior relationship through an example?

We saw an example in Lecture 10, for the case Gaussian-Gaussian.

Let's consider the experiment where we flip *a* coin. We are interested in estimating the probability of flipping heads as new samples from this experiment arrive.

Let heads=1 and tails=0, so our sample space is $S = \{1, 0\}$. We can model the data likelihood as a Bernoulli distribution:

$$P(x|\mu) = \mu^x(1-\mu)^{1-x} = \begin{cases} \mu & \text{if } x = 1 \\ 1-\mu & \text{if } x = 0 \end{cases}$$

Further assume that we have *prior knowledge* that the unknown parameter we are trying to estimate ($\mu \equiv$ probability of flipping heads) is a random variable and we will assume that it follows a Beta distribution:

$$\text{Beta}(\mu|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}\mu^{\alpha-1}(1-\mu)^{\beta-1}$$

where $\Gamma(x) = (x-1)!$ and $\alpha, \beta > 0$.

We can now apply solve for the parameter μ using the MAP approach:

$$\arg_\mu \max P(\mu|X) = \arg_\mu \max \ln P(\mu|X)$$

Let $\mathcal{L} = \ln P(\mu|X)$.

We write the posterior probability as:

$$\begin{aligned} P(\mu|X) &= \frac{P(X|\mu)P(\mu)}{P(X)} \\ &\propto P(X|\mu)P(\mu) \\ &\propto \left(\prod_{n=1}^{N} \mu^{x_n}(1-\mu)^{1-x_n}\right)\mu^{\alpha-1}(1-\mu)^{\beta-1} \\ &= \mu^m(1-\mu)^l\mu^{\alpha-1}(1-\mu)^{\beta-1} \\ &= \mu^{m+\alpha-1}(1-\mu)^{l+\beta-1} \end{aligned}$$

where m the number of heads, l the number of tails, and $N = m + l$ the total number of coin flips.

- This defines a Conjugate Prior relationship as the prior and the posterior follow the same shape.
- In an online environment (i.e. as new samples are coming), we can update the prior with the posterior distribution.By inspecting the posterior result, we see that:
  - $\alpha \leftarrow \alpha + m$ and $\beta = \beta + l$.
  - We "replace the prior with the posterior" by updating the parameters of the prior distributions with those of the new posterior.

Then

$$\mathcal{L} = (m + \alpha - 1)\ln(\mu) + (l + \beta - 1)\ln(1 - \mu)$$

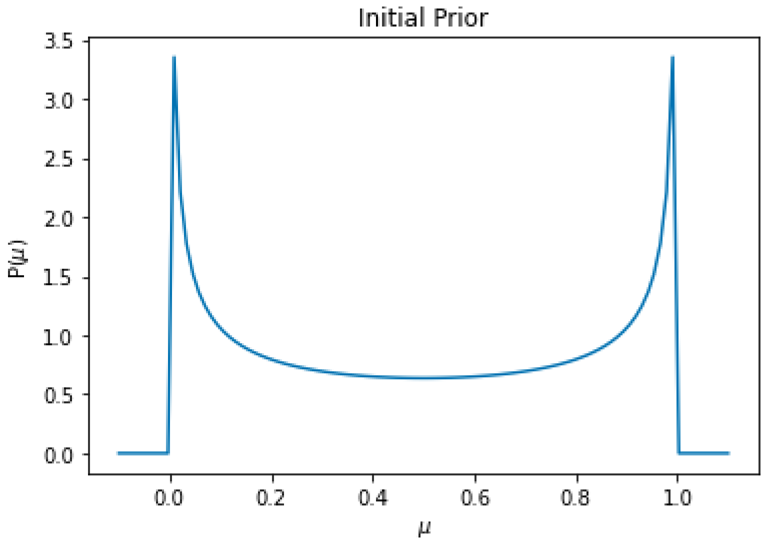
We can now *optimize* our posterior probability:

$$\begin{aligned} \frac{\partial \ln(P(\mu|E))}{\partial \mu} &= 0 \\ \frac{m+\alpha-1}{\mu} + \frac{l+\beta-1}{1-\mu} &= 0 \\ \mu &= \frac{m+\alpha-1}{m+l+\alpha+\beta-2} \end{aligned}$$

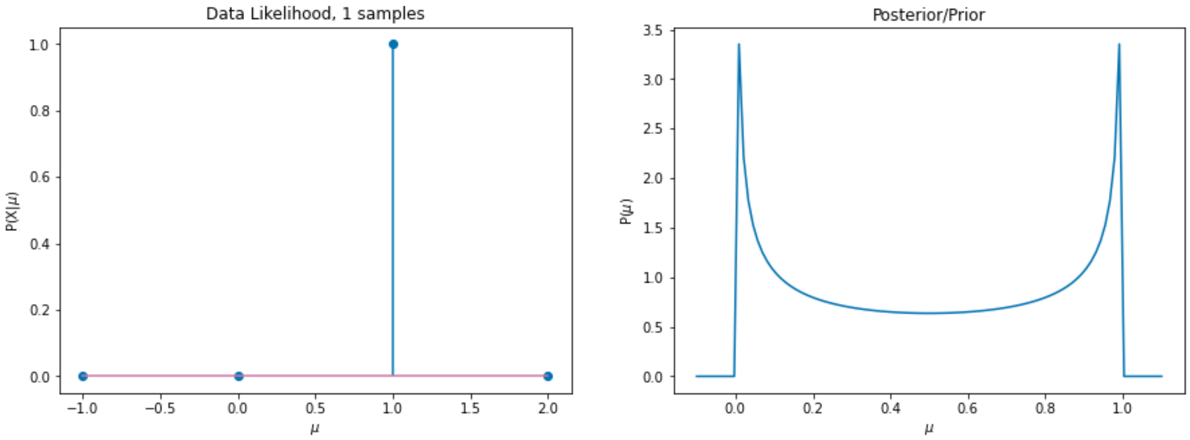
This is our estimation of the probability of heads using MAP!

In [4]:

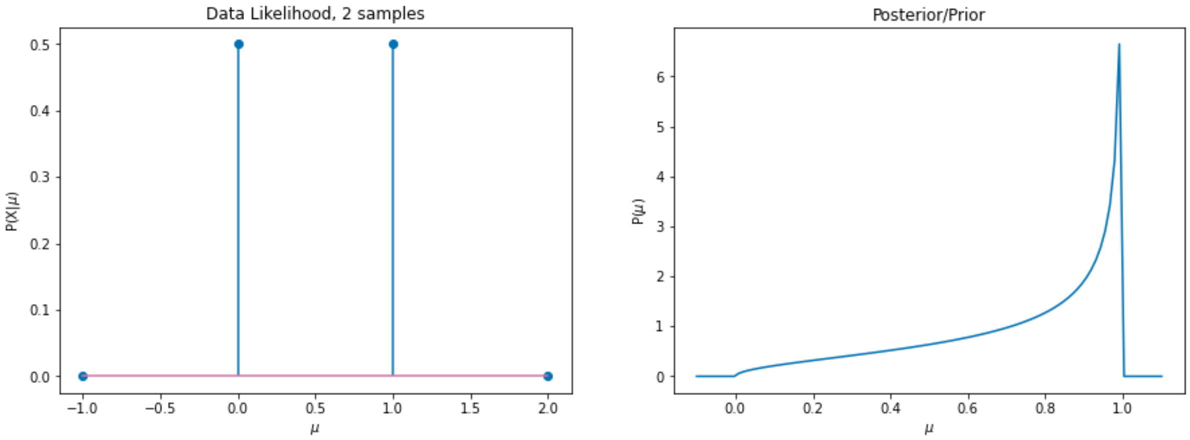
```
trueMU = 0.5 # 0.5 for a fair coin
Nflips = 15
a = 0.5
b = 0.5

xr = range(-1,3)
x = np.linspace(-0.1,1.1,100)
plt.plot(x, stats.beta(a,b).pdf(x))
plt.xlabel('$\mu$'); plt.ylabel('P($\mu$)'); plt.title('Initial Prior')
plt.show()
Outcomes = []
for i in range(Nflips):
    Outcomes += [stats.bernoulli(trueMU).rvs(1)[0]]
    estimate_mu = (np.sum(Outcomes)+a-1)/(len(Outcomes)+a+b-2)

    # Visualization:
    fig=plt.figure(figsize=(15,5))
    fig.add_subplot(1,2,1)
    plt.stem(xr,np.array([0,len(Outcomes)-np.sum(Outcomes),np.sum(Outcomes),0
])/(i+1),
            use_line_collection=True)
    plt.xlabel('$\mu$'); plt.ylabel('P(X|$\mu$)');
    plt.title('Data Likelihood, '+str(i+1)+' samples')
    fig.add_subplot(1,2,2)
    plt.plot(x, stats.beta(a,b).pdf(x))
    plt.xlabel('$\mu$'); plt.ylabel('P($\mu$)'); plt.title('Posterior/Prior')
    plt.show()

    # Print estimate for mu
    print('Data: ',Outcomes)
    print('MAP estimate mu = ', estimate_mu)

    # Update Prior distribution
    a += np.sum(Outcomes)
    b += len(Outcomes)-np.sum(Outcomes)
```



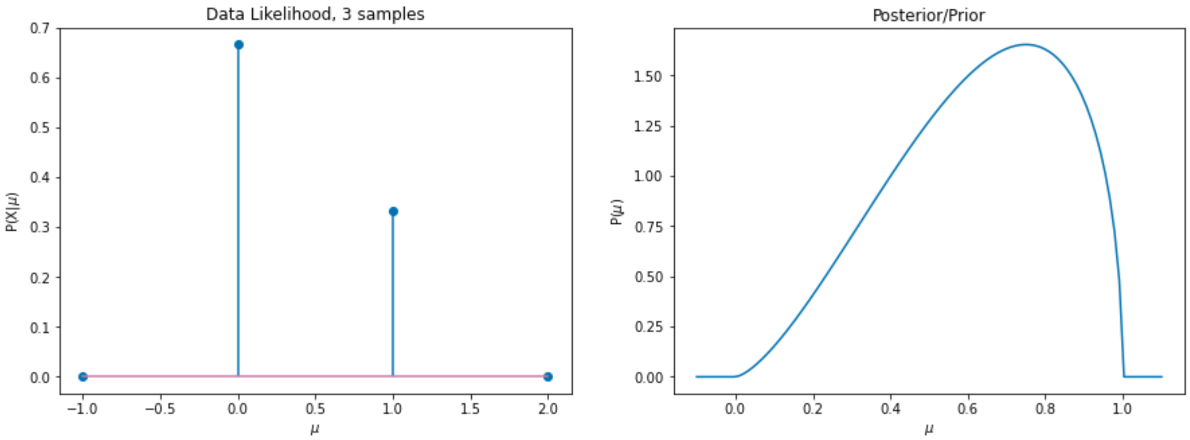
```
C:\Users\catia\anaconda3\lib\site-packages\ipykernel_launcher.py:14: RuntimeW
arning: divide by zero encountered in double_scalars
```



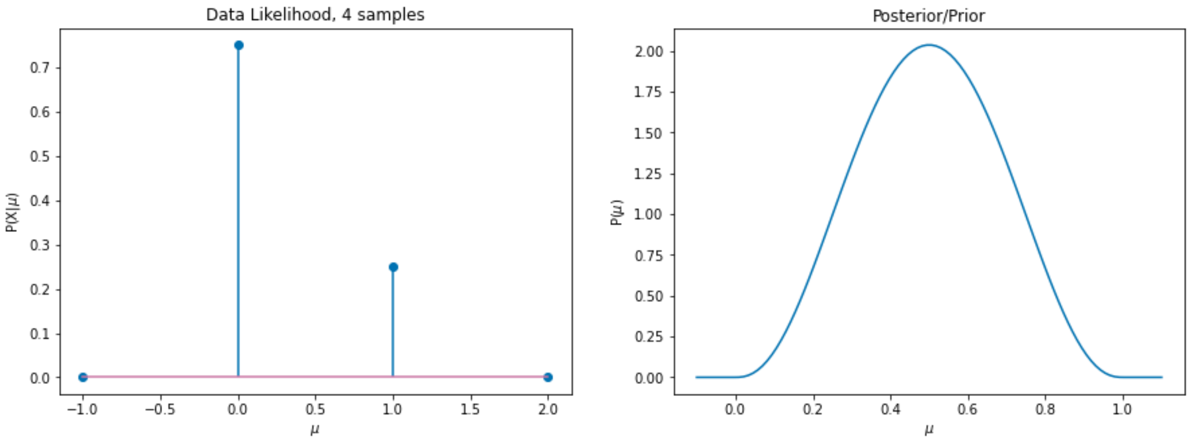
```
Data:  [1]
MAP estimate mu =  inf
```



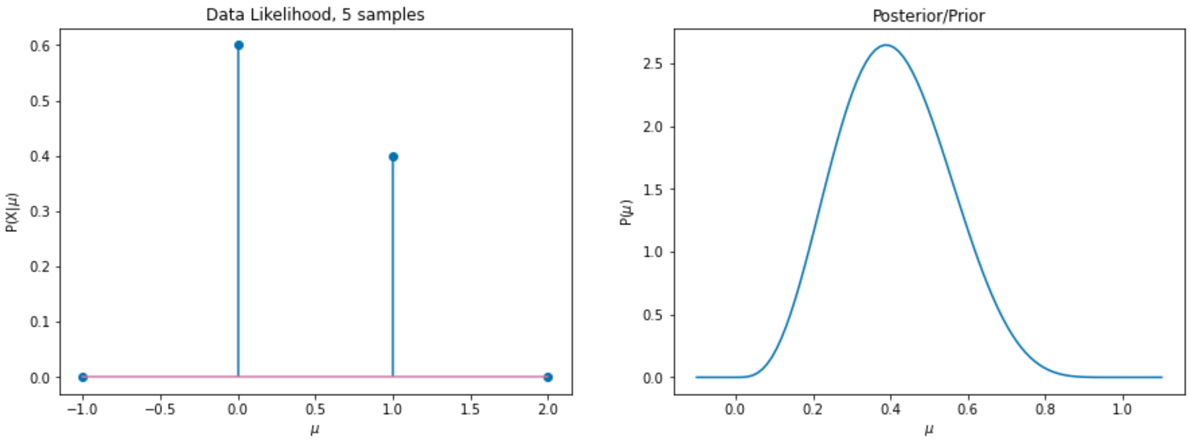
```
Data:  [1, 0]
MAP estimate mu =  0.75
```



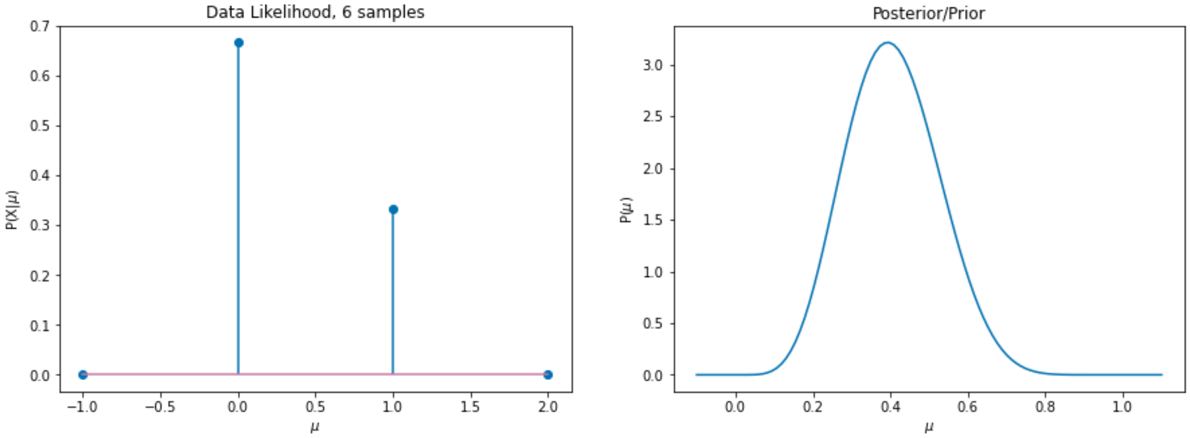
```
Data:  [1, 0, 0]
MAP estimate mu =  0.5
```



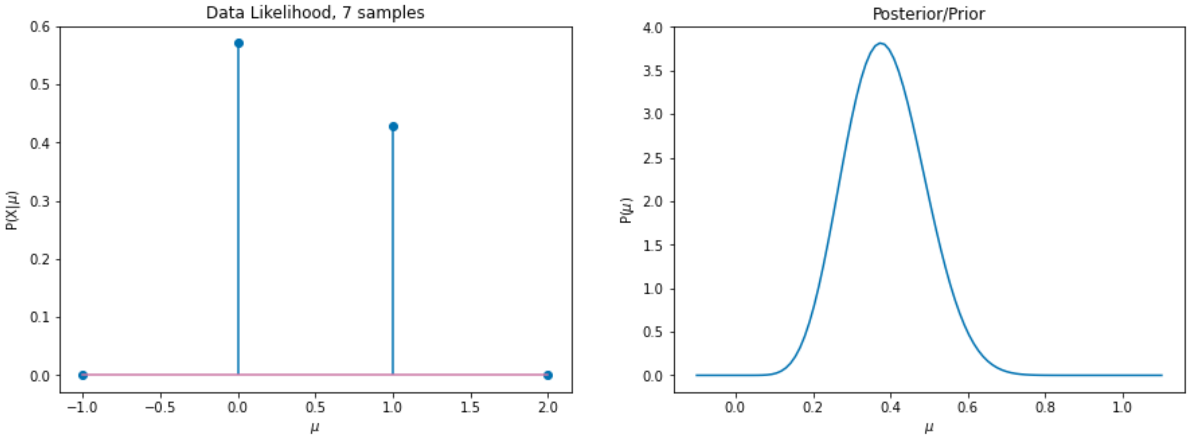
```
Data:  [1, 0, 0, 0]
MAP estimate mu =  0.3888888888888889
```



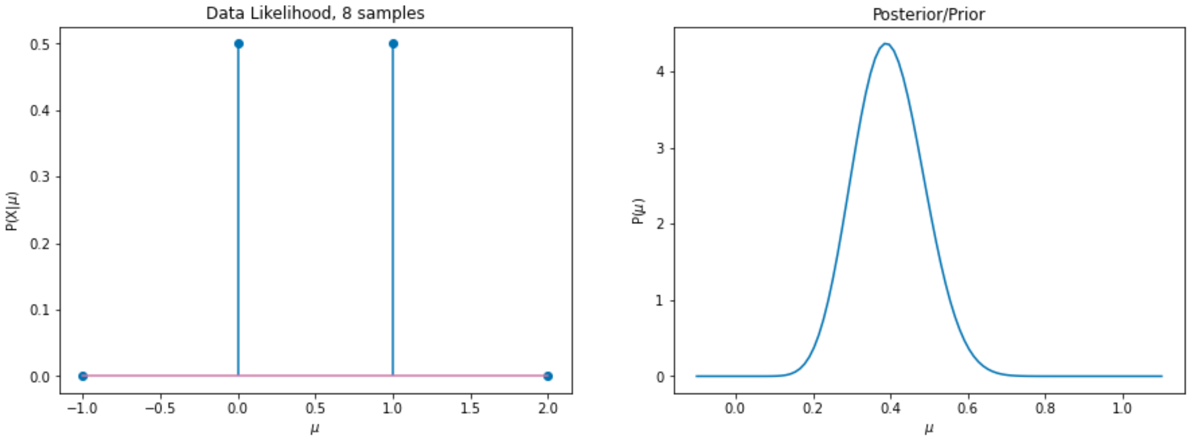
```
Data:  [1, 0, 0, 0, 1]
MAP estimate mu =  0.39285714285714285
```



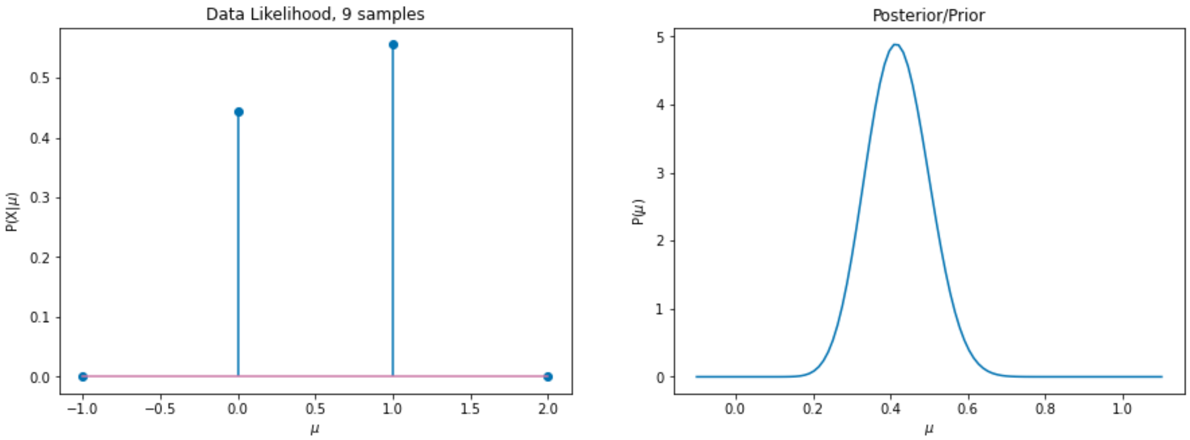
```
Data:  [1, 0, 0, 0, 1, 0]
MAP estimate mu =  0.375
```



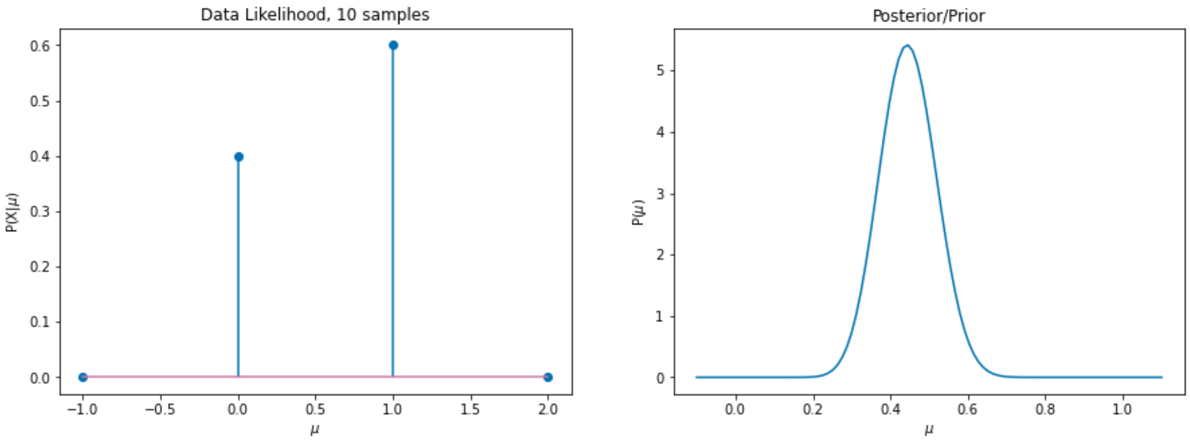
```
Data:  [1, 0, 0, 0, 1, 0, 1]
MAP estimate mu =  0.3888888888888889
```



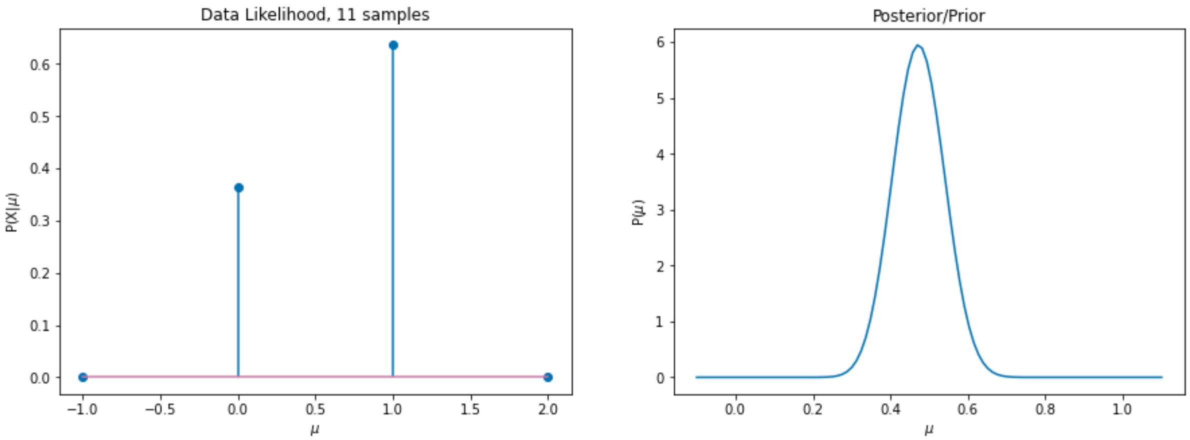
```
Data:  [1, 0, 0, 0, 1, 0, 1, 1]
MAP estimate mu =  0.4142857142857143
```



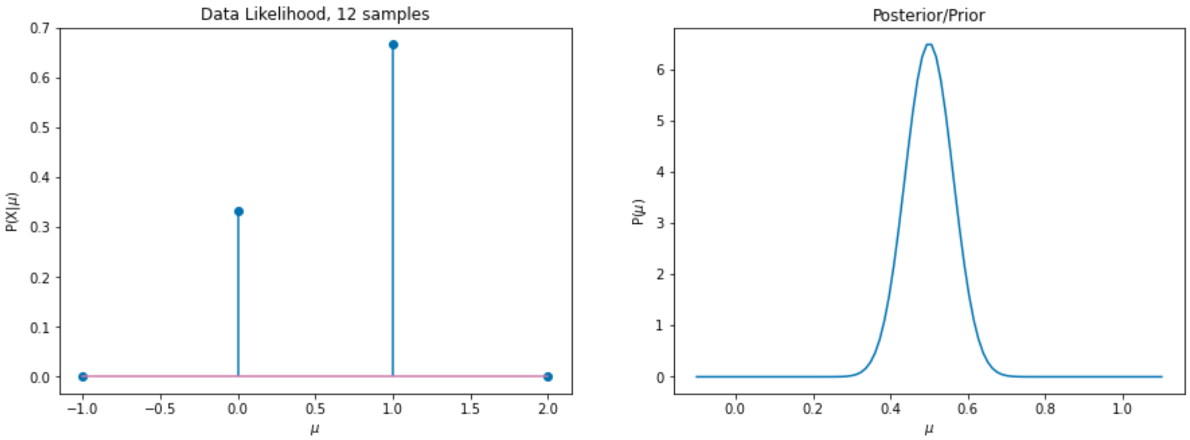
```
Data:  [1, 0, 0, 0, 1, 0, 1, 1, 1]
MAP estimate mu =  0.4431818181818182
```



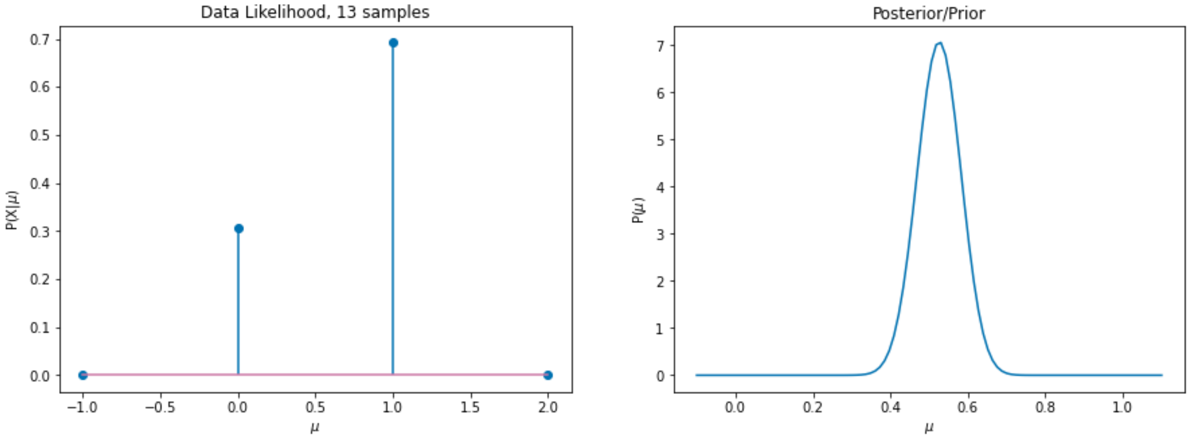
```
Data:  [1, 0, 0, 0, 1, 0, 1, 1, 1, 1]
MAP estimate mu =  0.4722222222222222
```



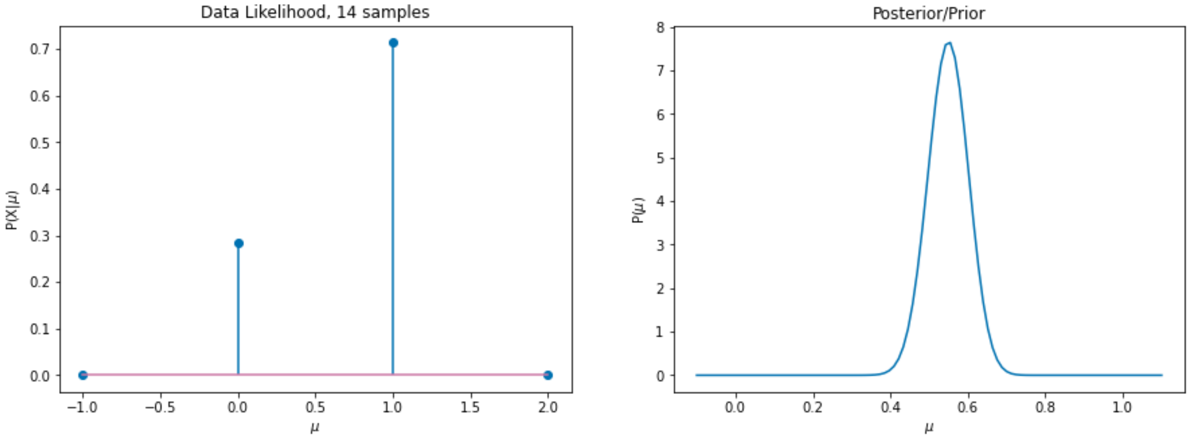
```
Data:  [1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1]
MAP estimate mu =  0.5
```



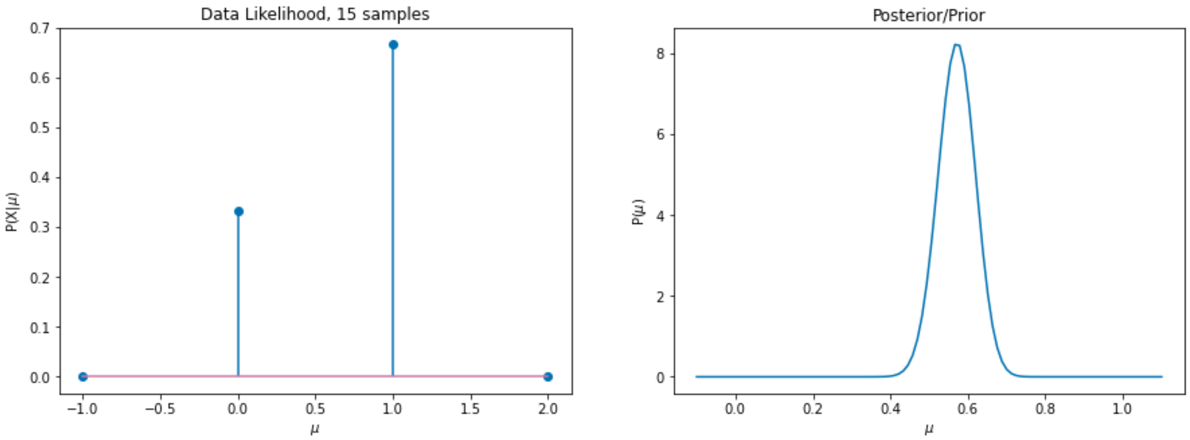
```
Data:  [1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1]
MAP estimate mu =  0.525974025974026
```



```
Data:  [1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1]
MAP estimate mu =  0.55
```



```
Data:  [1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1]
MAP estimate mu =  0.5721153846153846
```



```
Data:  [1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0]
MAP estimate mu =  0.5840336134453782
```

In [ ]:

Post 5

If possible can we cover short assignment 3?

Any specific questions?

Short Assignment 3: [https://ufl.instructure.com/courses/404363/assignments (https://ufl.instructure.com/courses/404363/assignments)](https://ufl.instructure.com/courses/404363/assignments)

Post 6

In the supervised learning flowchart, there is a block called learning algorithm. Could you give an example of what we did in class that is a learning algorithm? Is finding the best parameters in experimental design considered a learning algorithm?

For example, when we introduced classification using the Naive Bayes classifier, we define the model as a parametric representation of the data likelihood for each class. The objective is to maximize the data likelihood fitting. The learning algorithm is one that actually changes the parameters of the model. In this case, we can use either MLE or MAP as the learning algorithm to find the parameters of the data likelihood that maximize our objective.

Post 7

Will the exam be more application or theory based? As in, will we be expected to perform large calculations/derivations or know the overall concept of ML thus far? I ask because a lot of these large functions can get very difficult to understand through derivations.

The exam will cover both conceptual questions, and derivation-like questions.

To best prepare for the derivation-like questions, define the steps that are needed to take. For example, what are the steps to derive the best update equation for the μ_k, the mean of a Gaussian component in a GMM model?

We will solve this using the EM algorithm:

1. Write down the observed data likelihood
2. Define the hidden latent variables
3. Write down the complete data likelihood
4. Take the log of the complete data likelihood
5. Take the derivative of the log-likelihood with respect to μ_k

In [ ]: