Machine Learning and Computational Statistics
Homework 7: Bayesian Statistics

Due: Monday, May 15, 2017, at 10pm (Submit via Gradescope)

Instructions: Your answers to the questions below, including plots and mathematical work, should be submitted as a single PDF file. It's preferred that you write your answers using software that typesets mathematics (e.g. LaTeX, LyX, or MathJax via iPython), though if you need to you may scan handwritten work. You may find the minted package convenient for including source code in your LaTeX document. If you are using LyX, then the listings package tends to work better.

1 Introduction

In this homework we work through several basic concepts in Bayesian statistics via one of the simplest problems there is: estimating the probability of heads in a coin flip. Later we'll extend this to the probability of estimating click-through rates in mobile advertising.

2 Coin Flipping: Maximum Likelihood

1. Suppose we flip a coin and get the following sequence of heads and tails:

$$\mathcal{D} = (H, H, T)$$

Give an expression for the probability of observing $\mathcal{D}$ given that the probability of heads is θ. That is, give an expression for $p\left(\mathcal{D} \mid \theta\right)$. This is called the **likelihood of θ for the data $\mathcal{D}$**.

Answer:

$$p(\mathcal{D} \mid \theta) = \theta^2(1-\theta)$$

2. How many different sequences of 3 coin tosses have 2 heads and 1 tail? If we coss the coin 3 times, what is the probability of 2 heads and 1 tail? (Answer should be in terms of θ.)

   Answer:
   There exists 3 different sequences that contain 2 heads and 1 tail. (H, H, T),(H, T, H) and (T, H, T). The probability of observing 2 heads and 1 tail is: $3\theta^2(1-\theta)$

3. More generally, give an expression for the likelihood $p(\mathcal{D} \mid \theta)$ for a particular sequence of flips $\mathcal{D}$ that has n_h heads and n_t tails. Make sure you have expressions that make sense even for $\theta = 0$ and $n_h = 0$, and other boundary cases. You may use the convention that $0^0 = 1$, or you can break your expression into cases if needed.

Answer:
We take the order of a sequence into account, therefore:

$$p(\mathcal{D} \mid \theta) = \theta^{n_h}(1-\theta)^{n_t}$$

4. Prove that the maximum likelihood estimate of θ given we observed a sequence with n_h heads and n_t tails is

$$\hat{\theta}_{\mathrm{MLE}} = \frac{n_h}{n_h + n_t}.$$

You may assume that $n_h + n_t \geq 1$. (Hint: Maximizing the log-likelihood is equivalent and is often easier.)

Answer:
From part 3, we have $p(\mathcal{D} \mid \theta) = \binom{n_h+n_t}{n_h}\theta^{n_h}(1-\theta)^{n_t}$. Take the derivative of $\log p(\mathcal{D} \mid \theta)$ yields:

$$\frac{\partial \log p(\mathcal{D} \mid \hat{\theta}_{\mathrm{MLE}})}{\partial \theta} = 0 \implies \frac{n_h}{\hat{\theta}_{\mathrm{MLE}}} - \frac{n_t}{(1-\hat{\theta}_{\mathrm{MLE}})} = 0 \tag{1}$$

$$\hat{\theta}_{\mathrm{MLE}} = \frac{n_h}{n_h + n_t} \tag{2}$$

3 Coin Flipping: Bayesian Approach with Beta Prior

We'll now take a Bayesian approach to the coin flipping problem, in which we treat θ as a random variable sampled from some prior distribution $p(\theta)$. We'll represent the ith coin flip by a random variable $X_i \in \{0,1\}$, where $X_i = 1$ if the ith flip is heads. We assume that the X_i's are conditionally indendent given θ. This means that the joint distribution of the coin flips and θ factorizes as follows:

$$\begin{aligned} p(x_1, \ldots, x_n, \theta) &= p(\theta) p(x_1, \ldots, x_n \mid \theta) \text{ (always true)} \\ &= p(\theta) \prod_{i=1}^{n} p(x_i \mid \theta) \text{ (by conditional independence).} \end{aligned}$$

1. Suppose that our prior distribution on θ is Beta(h,t), for some $h, t > 0$. That is, $p(\theta) \propto \theta^{h-1} (1-\theta)^{t-1}$. Suppose that our sequence of flips $\mathcal{D}$ has n_h heads and n_t tails. Show that the posterior distribution for θ is Beta$(h + n_h, t + n_t)$. That is, show that

$$p(\theta \mid \mathcal{D}) \propto \theta^{h-1+n_h} (1-\theta)^{t-1+n_t}.$$

We say that the Beta distribution is **conjugate** to the Bernoulli distribution since the prior and the posterior are both in the same family of distributions (i.e. both Beta distributions).

Answer:

$$\begin{aligned} p(\theta \mid \mathcal{D}) \propto p(\theta) p(\mathcal{D} \mid \theta) &= \theta^{h-1} (1-\theta)^{t-1} \theta^{n_h} (1-\theta)^{n_t} \\ &= \theta^{h-1+n_h} (1-\theta)^{t-1+n_t} \end{aligned}$$

2. Give expressions for the MLE, the MAP, and the posterior mean estimates of θ. [Hint: You may use the fact that a Beta(h, t) distribution has mean $h/(h+t)$ and has mode $(h-1)/(h+t-2)$ for $h, t > 1$. For the Bayesian solutions, you should note that as $h+t$ gets very large, and assuming we keep the ratio $h/(h+t)$ fixed, the posterior mean and MAP approach the prior mean $h/(h+t)$, while for fixed h and t, the posterior mean approaches the MLE as the sample size $n = n_h + n_t \to \infty$.

Answer:
The maximum likelihood estimation θ_{MLE} should be the mode of likelihood function: $p(\mathcal{D} \mid \theta) = \binom{n_h+n_t}{n_h}\theta^{n_h}(1-\theta)^{n_t}$, and should have zero derivative at θ_{MLE}:

$$\frac{\partial p(\mathcal{D} \mid \theta)}{\theta} = 0 \implies \theta_{\text{MLE}} = \frac{n_h}{n_h + n_t}$$

From part 1, we have $p(\theta \mid \mathcal{D}) \propto \theta^{h-1+n_h}\left(1-\theta\right)^{t-1+n_t} = Beta(n_h + h, n_t + t)$. The MAP estimation$\theta_{\text{MAP}}$ should be the mode of posterior distribution:

$$\hat{\theta}_{\text{MAP}} = \frac{n_h + h - 1}{n_h + h + n_t + t - 2}$$

The posterior mean of θ is:

$$\hat{\theta}_{\text{POSTERIOR MEAN}} = \frac{n_h + h}{n_h + h + n_t + t}$$

3. What happens to $\hat{\theta}_{\text{MLE}}$, $\hat{\theta}_{\text{MAP}}$, and $\hat{\theta}_{\text{POSTERIOR MEAN}}$ as the number of coin flips $n = n_h + n_t$ approaches infinity?

Answer:
When n approaches infinity, the effect of prior on posterior is negligible. Therefore we expect $\hat{\theta}_{\text{MAP}}$, and $\hat{\theta}_{\text{POSTERIOR MEAN}}$ to converges to θ. By the definition of $\hat{\theta}_{\text{MLE}}$, the variance of $\hat{\theta}_{\text{MLE}}$ would be decrease with the increase of n and finally converges to θ.

4. The MAP and posterior mean estimators of θ were derived from a Bayesian perspective. Let's now evaluate them from a frequentist perspective. Suppose θ is fixed and unknown. Which of the MLE, MAP, and posterior mean estimators give **unbiased** estimates of θ, if any? [Hint: The answer may depend on the parameters h and t of the prior. Also, let's consider the total number of flips $n = n_h + n_t$ to be given (not random), while n_h and n_t are random, with $n_h = n - n_t$.]

Answer:
$\hat{\theta}_{\text{MLE}}$ is unbiased because:

$$\mathbb{E}[\hat{\theta}_{\text{MLE}}] = \mathbb{E}[\frac{n_h}{n}] = \frac{1}{n}\mathbb{E}[n_h] = \frac{n\theta}{n} = \theta$$

$\hat{\theta}_{\text{MAP}}$ could be biased, since:

$$\mathbb{E}[\hat{\theta}_{\text{MAP}}] = \mathbb{E}[\frac{n_h + h - 1}{n + h + t - 2}] = \frac{n\theta + h - 1}{n + h + t - 2}]$$

$\hat{\theta}_{\text{MAP}}$ is unbiased only when h=t=1.
$\hat{\theta}_{\text{POSTERIOR MEAN}}$ is biased since:

$$\mathbb{E}[\hat{\theta}_{\text{POSTERIOR MEAN}}] = \mathbb{E}[\frac{n_h + h}{n + h + t}] = \mathbb{E}[\frac{n\theta + h}{n + h + t}]$$

$\hat{\theta}_{\text{POSTERIOR MEAN}}$ is unbiased only when h=t=0. But h and t cannot be zero.

5. Suppose somebody gives you a coin and asks you to give an estimate of the probability of heads, but you can only toss the coin 3 times. You have no particular reason to believe this is an unfair coin. Would you prefer the MLE or the posterior mean as a point estimate of θ? If the posterior mean, what would you use for your prior?

Answer:

I would prefer posterior mean. Because for 3 coins tosses, $\hat{\theta}_{\text{MLE}}$ is very instable. For example if you get three heads, $\hat{\theta}_{\text{MLE}} = 1$. But the results of three heads is totally observable even with a fair coin.

If I chose posterior mean, I would choose prior that have mode in $\theta = \frac{1}{2}$ since I don't have a particular reason to believe this is an unfair coin. But I am not that sure it is a fair coin either. Therefore I choose Beta(2,2) as my prior.

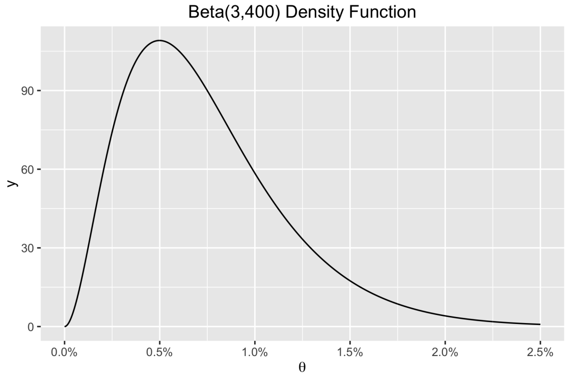
4 Hierarchical Bayes for Click-Through Rate Estimation

In mobile advertising, ads are often displayed inside apps on a phone or tablet device. When an ad is displayed, this is called an "impression." If the user clicks on the ad, that is called a "click." The probability that an impression leads to a click is called the "click-through rate" (CTR).

Suppose we have $d = 1000$ apps. For various reasons[1], each app tends to have a different overall CTR. For the purposes of designing an ad campaign, we want estimates of all the app-level CTRs, which we'll denote by $\theta_1, \ldots, \theta_{1000}$. Of course, the particular user seeing the impression and the particular ad that is shown have an effect on the CTR, but we'll ignore these issues for now. [Because so many clicks on mobile ads are accidental, it turns out that the overall app-level CTR often dominates the effect of the particular user and the specific ad.]

If we have enough impressions for a particular app, then the empirical fraction of clicks will give a good estimate for the actual CTR. However, if we have relatively few impressions, we'll have some problems using the empirical fraction. Typical CTRs are less than 1%, so it takes a fairly large number of observations to get a good estimate of CTR. For example, even with 100 impressions, the only possible CTR estimates given by the MLE would be $0\%, 1\%, 2\%, \ldots, 100\%$. The 0% estimate is almost certainly much too low, and anything 2% or higher is almost certainly much too high. Our goal is to come up with reasonable point estimates for $\theta_1, \ldots, \theta_{1000}$, even when we have very few observations for some apps.

If we wanted to apply the Bayesian approach worked out in the previous problem, we could come up with a prior that seemed reasonable. For example, we could use the following $\text{Beta}(3, 400)$ as a prior distribution on each θ_i:



In this basic Bayesian approach, the parameters 3 and 400 would be chosen by the data scientist based on prior experience, or "best guess", but without looking at the new data. Another approach would be to use the data to help you choose the parameters a and b in $\text{Beta}(a, b)$. This would **not** be a Bayesian approach, though it is frequently used in practice. One method in this direction is called **empirical Bayes**. Empirical Bayes can be considered a frequentist approach, in which we estimate a and b from the data $\mathcal{D}$ using some estimation technique, such as maximum likelihood. The proper Bayesian approach to this type of thing is called **hierarchical Bayes**, in which we put another prior distribution on a and b. We'll investigate each of these approaches below.

[1] The primary reason is that different apps place the ads differently, making it more or less difficult to avoid clicking the ad.

Mathematical Description

We'll now give a mathematical description of our model, assuming the prior parameters a and b are directly chosen by the data scientist. Let $n_1, \ldots, n_d$ be the number of impressions we observe for each of the d apps. In this problem, we will not consider these to be random numbers. For the ith app, let $c_i^1, \ldots, c_i^{n_i} \in \{0, 1\}$ be indicator variables determining whether or not each impression was clicked. That is, $c_i^j = 1(j\text{th impression on } i\text{th app was clicked})$. We can summarize the data on the ith app by $\mathcal{D}_i = (x_i, n_i)$, where $x_i = \sum_{j=1}^{n_i} c_i^j$ is the total number of impressions that were clicked for app i. Let $\theta = (\theta_1, \ldots, \theta_d)$, where θ_i is the CTR for app i.

In our Bayesian approach, we act as though the data were generated as follows:

1. Sample $\theta_1, \ldots, \theta_d$ i.i.d. from Beta(a, b).
2. For each app i, sample $c_i^1, \ldots, c_i^{n_i}$ i.i.d. from Bernoulli(θ_i).

4.1 Empirical Bayes for a single app

We start by working out some details for Bayesian inference for a single app. That is, suppose we only have the data $\mathcal{D}_i$ from app i, and nothing else. Mathematically, this is exactly the same setting as the coin tossing setting above, but here we push it further.

1. Give an expression for $p(\mathcal{D}_i \mid \theta_i)$, the likelihood of $\mathcal{D}_i$ given the probability of click θ_i, in terms of θ_i, x_i and n_i.

Answer:

$$p(\mathcal{D}_i \mid \theta_i) = \theta^{x_i}(1-\theta)^{n_i - x_i}$$

2. We will take our prior distribution on θ_i to be Beta(a, b). The correpoding probability density function is given by

$$p(\theta_i) = \text{Beta}(\theta_i; a, b) = \frac{1}{B(a, b)} \theta_i^{a-1} \left(1 - \theta_i\right)^{b-1},$$

where $B(a, b)$ is called the Beta function. Explain (without calculation) why we must have

$$\int \theta_i^{a-1} \left(1 - \theta_i\right)^{b-1} \, d\theta_i = B(a, b).$$

Answer:
Because $p(\theta_i)$ is defined to be a probability density function and therefore we should have $\int p(\theta_i) d\theta_i = 1$

3. Give an expression for the posterior distribution $p(\theta_i \mid \mathcal{D}_i)$. In this case, include the constant of proportionality. In other words, do not use the "is proportional to" sign $\propto$ in your final expression. You **may** reference the Beta function defined above. [Hint: This problem is essentially a repetition of an earlier problem.]

Answer:

$$\begin{aligned} p(\theta_i \mid \mathcal{D}_i) &\propto p(\mathcal{D} \mid \theta_i)p(\theta_i) && (3) \\ &= \frac{\theta_i^{x_i+a-1}(1-\theta_i)^{n-x_i+b-1}}{B(a,b)} && (4) \\ &\propto \theta_i^{x_i+a-1}(1-\theta_i)^{n-x_i+b-1} && (5) \\ & && (6) \end{aligned}$$

We know that $p(\theta_i \mid \mathcal{D})$is a probability density function, therefore we must have $\int p(\theta_i \mid \mathcal{D}_i) = 1$. Combined with conclusion of part 2, we have:

$$p(\theta_i \mid \mathcal{D}_i) = \frac{\theta_i^{x_i+a-1}(1-\theta_i)^{n-x_i+b-1}}{B(x_i+a, n-xi+b)} \qquad (7)$$

4. Give a closed form expression for $p(\mathcal{D}_i)$, the marginal likelihood of $\mathcal{D}_i$, in terms of the a, b, x_i, and n_i. You may use the normalization function $B(\cdot, \cdot)$ for convenience, but you should not have any integrals in your solution. (Hint: $p(\mathcal{D}_i) = \int p\left(\mathcal{D}_i \mid \theta_i\right) p(\theta_i)\, d\theta_i$, and the answer will be a ratio of two beta function evaluations.)

Answer:

$$\begin{aligned} p(\mathcal{D}_i) &= \int_0^1 p(\mathcal{D}_i \mid \theta_i) p(\theta_i) d\theta_i && (8) \\ &= \frac{1}{B(a,b)} \int_0^1 \theta^{x_i+a-1}(1-\theta)^{n_i-x_i+b-1} && (9) \\ &= \frac{B(x_i+a, n_i-x_i+b)}{B(a,b)} && (10) \end{aligned}$$

5. The maximum likelihood estimate for θ_i is x_i/n_i. Let $p_{\mathrm{MLE}}(\mathcal{D}_i)$ be the marginal likelihood of $\mathcal{D}_i$ when we use a prior on θ_i that puts all of its probability mass at x_i/n_i. Note that

$$\begin{aligned} p_{\mathrm{MLE}}(\mathcal{D}_i) &= p\left(\mathcal{D}_i \mid \theta_i = \frac{x_i}{n_i}\right) p\left(\theta_i = \frac{x_i}{n_i}\right) \\ &= p\left(\mathcal{D}_i \mid \theta_i = \frac{x_i}{n_i}\right). \end{aligned}$$

Explain why, or prove, that $p_{\mathrm{MLE}}(\mathcal{D}_i)$ is larger than $p(\mathcal{D}_i)$ for any other prior we might put on θ_i. If it's too hard to reason about all possible priors, it's fine to just consider all Beta priors. [Hint: This does not require much or any calculation. It may help to think about the integral $p(\mathcal{D}_i) = \int p\left(\mathcal{D}_i \mid \theta_i\right) p(\theta_i)\, d\theta_i$ as a weighted average of $p(\mathcal{D}_i \mid \theta_i)$ for different values of θ_i, where the weights are $p(\theta_i)$.]

Answer:
By the definition of MLE, we must have $p(\mathcal{D}_i \mid \theta) \leq p(\mathcal{D}_i \mid \theta_{\mathrm{MLE}})$. Therefore:

$$\begin{aligned} p(\mathcal{D}_i) &= \int p(\mathcal{D}_i \mid \theta_i) p(\theta_i) d\theta && (11) \\ &\leq \int p(\mathcal{D}_i \mid \theta_{\mathrm{MLE}}) p(\theta_i) d\theta && (12) \\ &= p(\mathcal{D}_i \mid \theta_{\mathrm{MLE}}) \int p(\theta_i) d\theta && (13) \\ &= p(\mathcal{D}_i \mid \theta_{\mathrm{MLE}}) && (14) \end{aligned}$$

Since $p(\theta)$ is a probability density distribution.

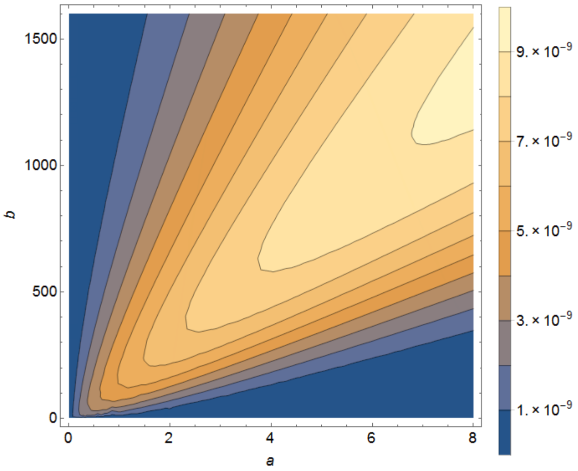


Figure 1: A plot of $p(\mathcal{D}_i \mid a, b)$ as a function of a and b.

6. One approach to getting an **empirical Bayes** estimate of the parameters a and b is to use maximum likelihood. Such an empirical Bayes estimate is often called an **ML-2** estimate, since it's maximum likelihood, but at a higher level in the Bayesian hierarchy. To emphasize the dependence of the likelihood of $\mathcal{D}_i$ on the parameters a and b, we'll now write it as $p(\mathcal{D}_i \mid a, b)$[2]. The empirical Bayes estimates for a and b are given by

$$(\hat{a}, \hat{b}) = \underset{(a,b)\in\mathbf{R}^{>0}\times\mathbf{R}^{>0}}{\arg\max}\; p(\mathcal{D}_i \mid a, b).$$

To make things concrete, suppose we observed $x_i = 3$ clicks out of $n_i = 500$ impressions. A plot of $p(\mathcal{D}_i \mid a, b)$ as a function of a and b is given in Figure 1. It appears from this plot that the likelihood will keep increasing as a and b increase, at least if a and b maintain a particular ratio. Indeed, this likelihood function never attains its maximum, so we cannot use ML-2 here. Explain what's happening to the prior as we continue to increase the likelihood. [Hint: It is a property of the Beta distribution (not difficult to see), that for any $\theta \in (0, 1)$, there is a Beta distribution with expected value θ and variance less than ε, for any $\varepsilon > 0$. What's going in here is similar to what happens when you attempt to fit a gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ to a single data point using maximum likelihood.]

Answer:
If we keep increase the likelihood, the effect of prior on posterior will increase and finally dominate the posterior distribution. (The posterior is a beta distribution that still dominated by the a,b term)

[2] Note that this is a slight (though common) abuse of notation, because a and b are not random variables in this setting. It might be more appropriate to write this as $p(\mathcal{D}_i; a, b)$ or $p_{a,b}(\mathcal{D}_i)$. But this isn't very common.

4.2 Empirical Bayes Using All App Data

In the previous section, we considered working with data from a single app. With a fixed prior, such as Beta(3,400), our Bayesian estimates for θ_i seem more reasonable to me[3] than the MLE when our sample size n_i is small. The fact that these estimates seem reasonable is an immediate consequence of the fact that I chose the prior to give high probability to estimates that seem reasonable to me, before ever seeing the data. Our earlier attempt to use empirical Bayes (ML-2) to choose the prior in a data-driven way was not successful. With only a single app, we were essentially overfitting the prior to the data we have. In this section, we'll consider using the data from all the apps, in which case empirical Bayes makes more sense.

1. Let $\mathcal{D} = (\mathcal{D}_1, \ldots, \mathcal{D}_d)$ be the data from all the apps. Give an expression for $p(\mathcal{D} \mid a, b)$, the **marginal likelihood** of $\mathcal{D}$. Expression should be in terms of a, b, x_i, n_i for $i = 1, \ldots, d$. Assume data from different apps are independent. (Hint: This problem should be easy, based on a problem from the previous section.)

   Answer:

$$
\begin{aligned}
p(\mathcal{D} \mid a, b) &= \prod_{i=1}^{d} p(\mathcal{D}_i \mid a, b) && (15) \\
&= \prod_{i=1}^{d} \frac{B(x_i + a, n_i - x_i + b)}{B(a, b)} && (16)
\end{aligned}
$$

[3] I say "to me", since I am the one who chose the prior. You may have an entirely different prior, and think that my estimates are terrible.

2. Explain why $p(\theta_i \mid \mathcal{D}) = p(\theta_i \mid \mathcal{D}_i)$, according to our model. In other words, once we choose values for parameters a and b, information about one app does not give any information about other apps.

Answer:
By bayes rule

$$p(\theta_i \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \theta_i)p(\theta_i)}{p(\mathcal{D})} = \frac{p(\mathcal{D} \mid \theta_i)p(\theta_i)}{\prod_{k=1}^{p} p(\mathcal{D}_k)} \tag{17}$$

$p(\theta_i)$ is the prior on θ_i that depends only on a and b, which is same for all i. For $p(\mathcal{D} \mid \theta_i)$, we have:

$$p(\mathcal{D} \mid \theta_i) = \prod_{k=1}^{d} p(\mathcal{D}_k \mid \theta_i) \tag{18}$$

Since $\mathcal{D}_k$ only depends on θ_k for all k. θ_i will only affect $p(\mathcal{D}_i \mid \theta_i)$ and $p(\mathcal{D}_k \mid \theta_i) = p(\mathcal{D}_k)$ for all $k \neq i$. Plug this result to (15) and (14) we get:

$$\begin{align} p(\theta_i \mid \mathcal{D}) &= \frac{p(\mathcal{D}_i \mid \theta_i)p(\theta_i) \prod_{k \neq i} p(\mathcal{D}_k)}{\prod_{k \neq i} p(\mathcal{D}_k) p(\mathcal{D}_i)} \tag{19} \\ &= \frac{p(\mathcal{D}_i \mid \theta_i)p(\theta_i)}{p(\mathcal{D}_i)} \tag{20} \\ &= p(\theta_i \mid \mathcal{D}_i) \tag{21} \end{align}$$

3. Suppose we have data from 6 apps. 3 of the apps have a fair number of impressions, and 3 have relatively few. Suppose we observe the following:

	Num Clicks	Num Impressions
App 1	50	10000
App 2	160	20000
App 3	180	60000
App 4	0	100
App 5	0	5
App 6	1	2

Compute the empirical Bayes estimates for a and b. (Recall, this amounts to computing $(\hat{a}, \hat{b}) = \arg\max_{(a,b)\in\mathbf{R}^{>0}\times\mathbf{R}^{>0}} p(\mathcal{D} \mid a, b)$.) This will require solving an optimization problem, for which you are free to use any optimization software you like (perhaps scipy.optimize would be useful). The empirical Bayes prior is then $\text{Beta}(\hat{a}, \hat{b})$, where $\hat{a}$ and $\hat{b}$ are our ML-2 estimates. Give the corresponding prior mean and standard deviation for this prior.

Answer:
To compute the empirical Bayes estimates for a and b, we are looking for:

$$\arg\max_{a,b} \prod_{i=1}^{d} \frac{B(x_i + a, n_i - x_i + b)}{B(a, b)} = \arg\max_{a,b} [\sum_{i=1}^{d} \log(B(x_i + a, n_i - x_i + b)) - \log(B(a, b))]$$

The following codes are used for optimization which yields result of $a = 6.469$ and $b = 1181.437$. The prior mean is approximately 0.54% with standard deviation 0.21%:

```python
import numpy as np
import scipy
from scipy.special import beta,betaln
from scipy.misc import comb
from scipy.optimize import minimize

#data
x = [50,160,180,0,0,1]
n = [10000,20000,60000,100,5,2]

#Likelihood function
def pD(ab):
    assert len(ab)==2
    cumSum_list = []
    a = ab[0]
    b = ab[1]
    for i in range(len(x)):
        xi = x[i]
        ni = n[i]
        #print(beta(xi+a,ni-xi+b))
        #cumProd_list.append(comb(k=xi,N=ni,exact=True)*beta(xi+a,ni-xi+b)/beta(a,b
        cumSum_list.append(betaln(xi+a,ni-xi+b)-betaln(a,b))
    return -np.cumsum(cumSum_list)[-1]
#optimization
x0 = [8,1000]
minimize(fun=pD,x0=x0,method='Nelder-Mead',)

#optimization result
final_simplex: (array([[    6.46896461,  1181.43722485],
   [    6.46896415,  1181.43714184],
   [    6.46896422,  1181.43715013]]), array([ 2483.72514115,  2483.72514115,  2483
       fun: 2483.7251411499501
   message: 'Optimization terminated successfully.'
      nfev: 104
       nit: 48
    status: 0
   success: True
         x: array([    6.46896461,  1181.43722485])
```

4. Complete the following table:

	NumClicks	NumImpressions	MLE	MAP	PosteriorMean	PosteriorSD
App 1	50	10000	0.5%	**0.5%**	**0.5%**	**0.07%**
App 2	160	20000	0.8%	**0.78%**	**0.79%**	**0.06%**
App 3	180	60000	0.3%	**0.3%**	**0.3%**	**0.02%**
App 4	0	100	0%	**0.43%**	**0.5%**	**0.2%**
App 5	0	5	0%	**0.46%**	**0.54%**	**0.21%**
App 6	1	2	50%	**0.54%**	**0.23%**	**0.43%**

Make sure to take a look at the PosteriorSD values and note which are big and which are small.

Answer:

Calculation process is omitted here and the results are already been filled in the table above with bold characters.

We can observe that the PosteriorSD are much larger for those apps with less impressions. This makes sense because we have less data for them.

4.3 [Optional] Hierarchical Bayes

In Section 4.2 we managed to get empirical Bayes ML-II estimates for a and b by assuming we had data from multiple apps. However, we didn't really address the issue that ML-II, as a maximum likelihood method, is prone to overfitting if we don't have enough data (in this case, enough apps). Moreover, a true Bayesian would reject this approach, since we're using our data to determine our prior. If we don't have enough confidence to choose parameters for a and b without looking at the data, then the only proper Bayesian approach is to put another prior on the parameters a and b. If you are very uncertain about values for a and b, you could put priors on them that have high variance.

1. [Optional] Suppose P is the $\text{Beta}(a, b)$ distribution. Conceptually, rather than putting priors on a and b, it's easier to reason about priors on the mean m and the variance v of P. If we parameterize P by its mean m and the variance v, give an expression for the density function $\text{Beta}(\theta; m, v)$. You are free to use the internet to get this expression – just be confident it's correct. [Hint: To derive this, you may find it convenient to write some expression in terms of $\eta = a + b$.]

2. [Optional] Suggest a prior distribution to put on m and v. [Hint: You might want to use one of the distribution families given in this lecture.

3. [Optional] Once we have our prior on m and v, we can go "full Bayesian" and compute posterior distributions on $\theta_1, \ldots, \theta_d$. However, these no longer have closed forms. We would have to use approximation techniques, typically either a Monte Carlo sampling approach or a variational method, which are beyond the scope of this course[4]. After observing the data $\mathcal{D}$, m and v will have some posterior distribution $p(m, v \mid \mathcal{D})$. We can approximate that distribution by a point mass at the mode of that distribution $(m_{\text{MAP}}, v_{\text{MAP}}) = \arg\max_{m,v} p(m, v \mid \mathcal{D})$. **Give expressions** for the posterior distribution $p(\theta_1, \ldots, \theta_d \mid \mathcal{D})$, with and without this approximation. You do not need to give any explicit expressions here. It's fine to have expressions like $p(\theta_1, \ldots, \theta_d \mid m, v)$ in your solution. Without the approximation, you will probably need some integrals. It's these integrals that we need sampling or variational approaches to approximate. While one can see this approach as a way to approximate the proper Bayesian approach, one could also be skeptical and say this is just another way to determine your prior from the data. The estimators $(m_{\text{MAP}}, v_{\text{MAP}})$ are often called **MAP-II estimators**, since they are MAP estimators at a higher level of the Bayesian hierarchy.

[4] If you're very ambitious, you could try out a package like PyStan to see what happens.

5 Bayesian Linear Regression

In this problem, we will implement Bayesian Gaussian linear regression, essentially reproducing the example from lecture, which in turn is based on the example in Figure 3.7 of Bishop's *Pattern Recognition and Machine Learning* (page 155). We've provided plotting functionality in "support_code.py". Your task is to complete "problem.py". The implementation uses np.matrix objects, and you are welcome to use[5] the np.matrix.getI method.

1. Implement likelihoodFunc.

Answer:
A major draw back of the following codes is the implementation of for loop. I did try to leverage scipy.stats.multivariate_normal.pdf but it behaves strangely Sorry for not having enough time to investigate.

```
def likelihoodFunc(W, x, y_train, likelihood_var):
    '''
    Implement likelihoodFunc. This function returns the data likelihood
    given f(y_train | x; W) ~ Normal(w^Tx, likelihood_var).

    Args:
        W: Weights
        x: Training design matrix with first col all ones (np.matrix)
        y_train: Training response vector (np.matrix)
        likelihood_var: likelihood variance

    Returns:
        likelihood: Data likelihood (float)
    '''
    likelihood = 1
    for i in range(len(x)):
        mu = np.dot(W.T, x[i, :].T)
        likelihood *= 1 / np.sqrt(2 * math.pi * likelihood_var) * np.exp((-(y_train
    return likelihood
```

[5] However, in practice we are usually interested in computing the product of a matrix inverse and a vector, i.e. $X^{-1}b$. In this case, it's usually faster and more accurate to use a library's algorithms for solving a system of linear equations. Note that $y = X^{-1}b$ is just the solution to the linear system $Xy = b$. See for example John Cook's blog post for discussion.

2. Implement getPosteriorParams.

Answer:
This code should work with the problem.py set up which has prior mean of zero and uncorrelated features.

```
def getPosteriorParams(x, y_train, prior, likelihood_var = 0.2**2):
    '''
    Implement getPosterioParams. This function returns the posterior
    mean vector \mu_p and posterior covariance matrix \Sigma_p for
    Bayesian regression (normal likelihood and prior).

    Note support_code.make_plots takes this completed function as an argument.

    Args:
        x: Training design matrix with first col all ones (np.matrix)
        y_train: Training response vector (np.matrix)
        prior: Prior parameters; dict with 'mean' (prior mean np.matrix)
               and 'var' (prior covariance np.matrix)
        likelihood_var: likelihood variance- default (0.2**2) per the lecture slide

    Returns:
        postMean: Posterior mean (np.matrix)
        postVar: Posterior mean (np.matrix)
    '''
    # This infact assume prior mean are all zero
    wdim = x.shape[1]
    postMean = (np.dot(x.T,x)+likelihood_var**2*prior['var'].getI()).getI().dot(x.T
    postVar = (likelihood_var**(-2)*np.dot(x.T,x)+prior['var'].getI()).getI()
    #postMean.resize(wdim)
    return postMean, postVar
```

3. Implement getPredictiveParams.

Answer:

```
def getPredictiveParams(x_new, postMean, postVar, likelihood_var = 0.2**2):
    '''
    Implement getPredictiveParams. This function returns the predictive
    distribution parameters (mean and variance) given the posterior mean
    and covariance matrix (returned from getPosteriorParams) and the
    likelihood variance (default value from lecture).

    Args:
        x: New observation (np.matrix object)
        postMean, postVar: Returned from getPosteriorParams
        likelihood_var: likelihood variance (0.2**2) per the lecture slides

    Returns:
        - predMean: Mean of predictive distribution
        - predVar: Variance of predictive distribution
    '''

    # TO DO
    x_new = x_new.reshape(-1,1)
    predMean = np.dot(postMean.T,x_new)
    predVar = x_new.T.dot(postVar).dot(x_new)+likelihood_var**2
    return predMean, predVar
```

4. Run ’python problem.py’ from inside the Bayesian Regression directory to generate the plots. Note we suggest you change the default behavior in support_code.make_plots from plt.show, to saving the plots for inclusion in your homework submission.

Answer:

5. Comment on your results. In particular, discuss how each of the following change with sample size and with the strength of the prior: (i) the likelihood function, (ii) the posterior distribution, and (iii) the posterior predictive distribution.

Answer:

- Likelihood: The strength of prior does not affect likelihood distribution.
- Posterior: Stronger prior would result in the posterior being closer to prior distribution and if the prior is far from the the true distribution, it takes the posterior longer to converge to real distribution.
- Predidctive distribution: Stronge prior would make the predictive distribution to have less variance in prediction in the beginning few steps and if prior is far from the true distribution the predictive distribution converges slower if the prior is strong.

6. [Optional] Our work above was very much “full Bayes”, in that rather than coming up with a single prediction function, we have a whole distribution over posterior prediction functions. However, sometimes we want a single prediction function, and a common approach is to use the MAP estimate. As we discussed in class, for this setting, we can get the MAP estimate using ridge regression. Use ridge regression to get the MAP prediction function corresponding to the first prior covariance. What value did you use for the regularization coefficient? Why?

Figure 2: Prior variance: $\frac{1}{2}$

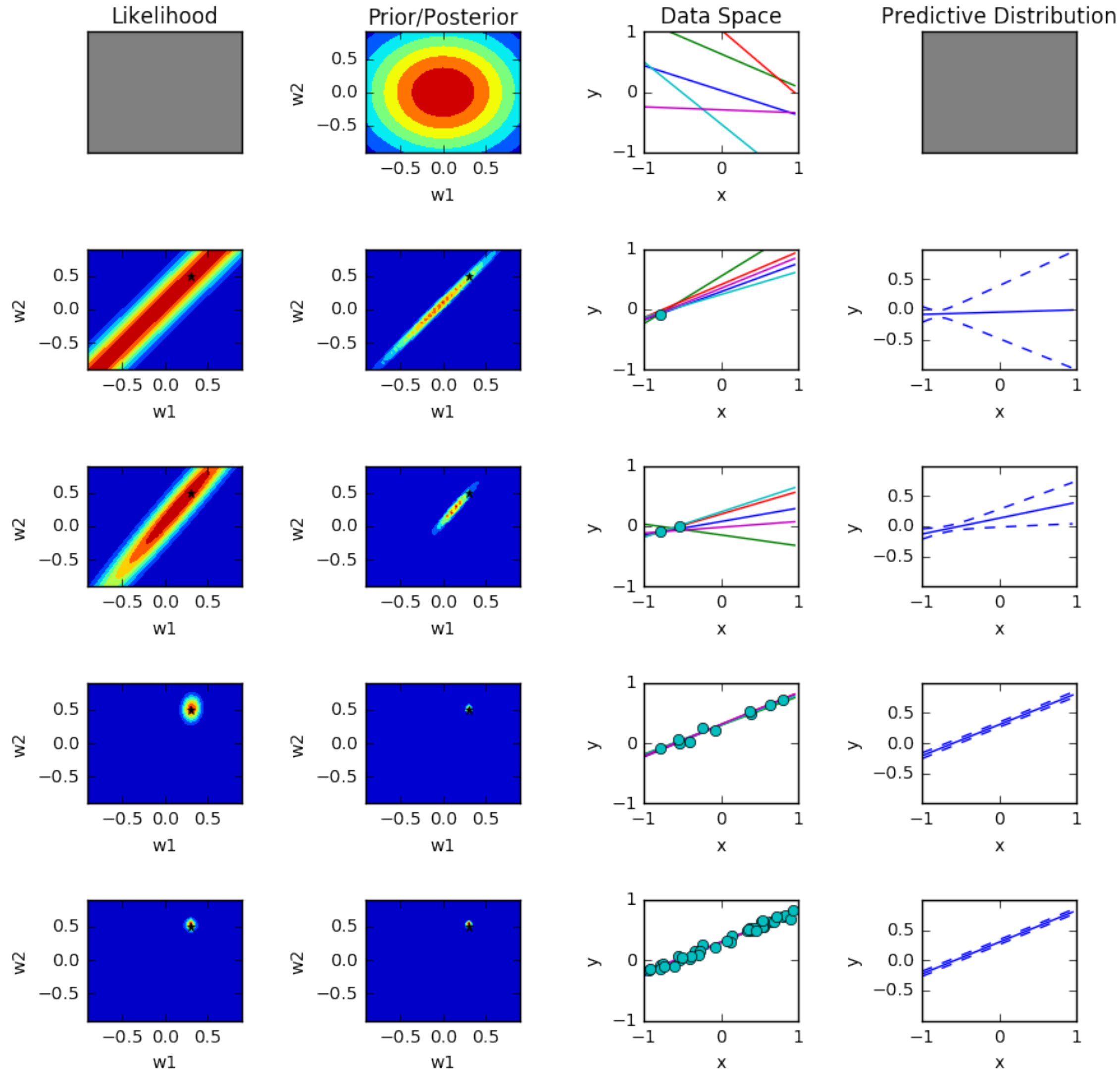


Figure 3: Prior variance: $\frac{1}{2^5}$

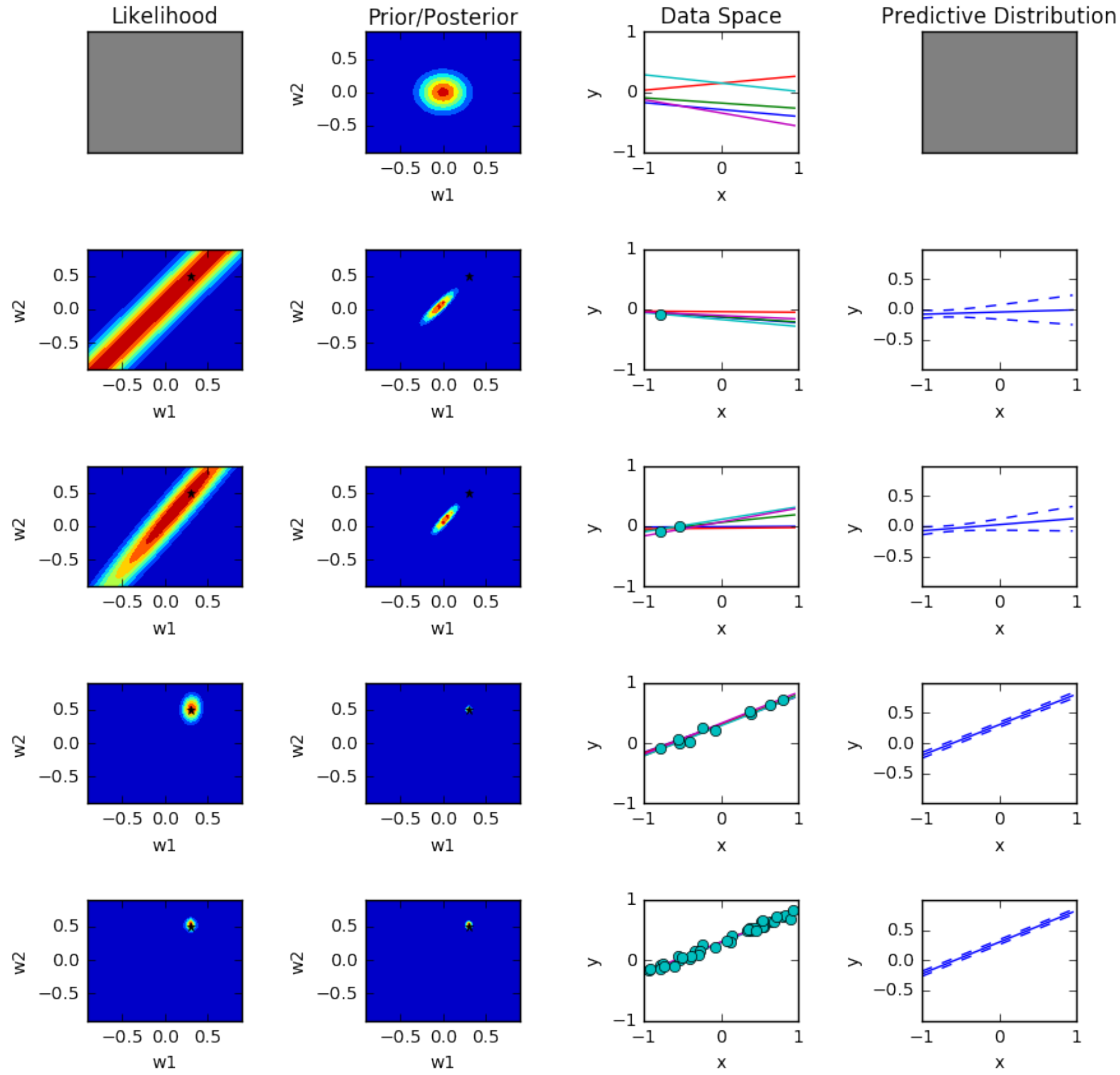
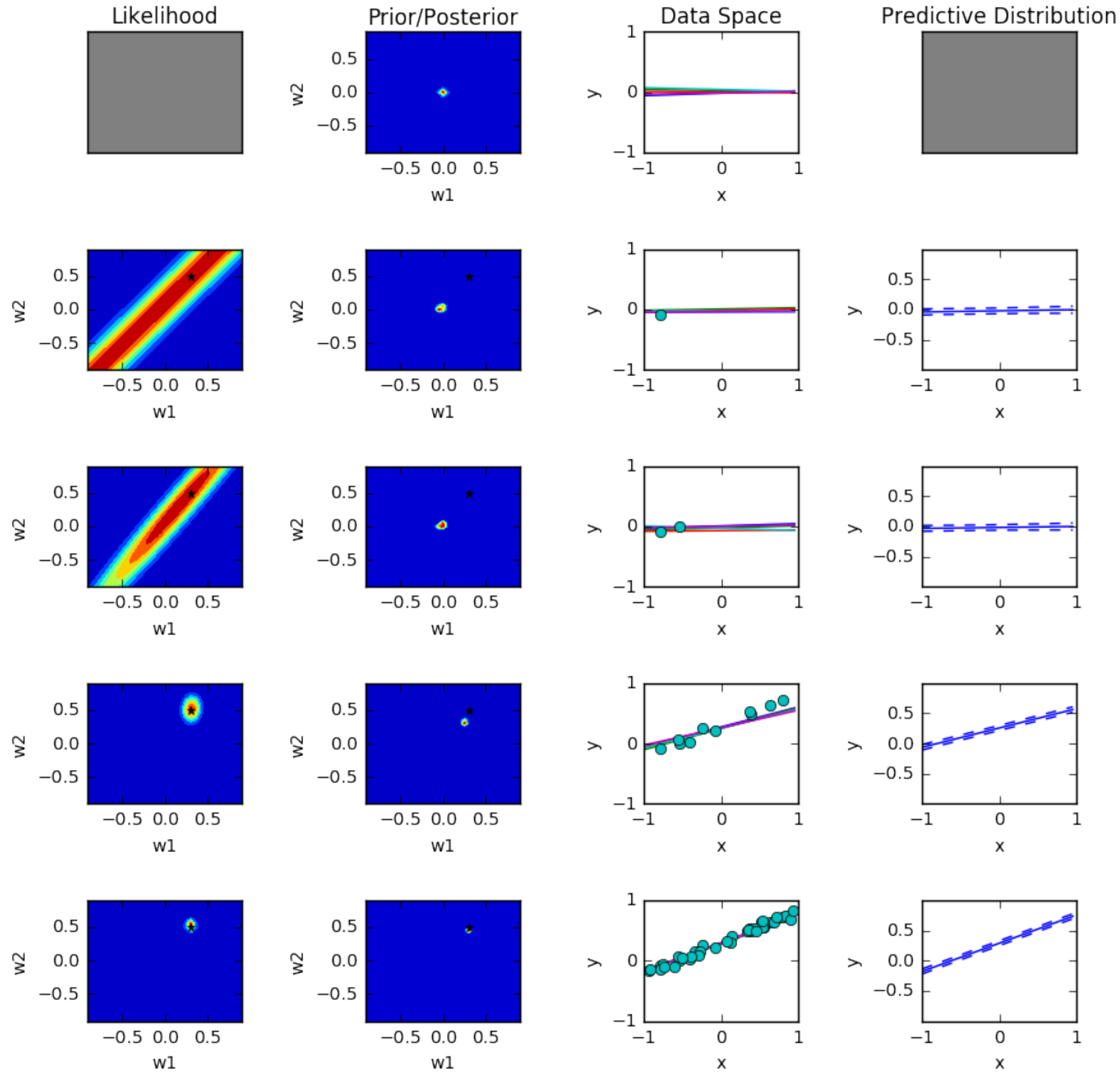


Figure 4: Prior variance: $\frac{1}{2^10}$


Likelihood
Prior/Posterior
Data Space
Predictive Distribution
w1
w2
x
y