

# Bayes Estimators and Bayes Classifier

Sean Tey, Shyam Sudhakaran, Rock Gu

2018-05-08

# Contents

<b>1</b>	<b>Abstract</b>	<b>1</b>
<b>2</b>	<b>Bayes Estimators</b>	<b>1</b>
2.1	Point Estimation . . . . .	1
2.2	Example of Point Estimation . . . . .	3
2.3	Interval Estimation . . . . .	3
2.4	Example of Interval Estimation . . . . .	3
<b>3</b>	<b>Gaussian Naive Bayes Classifier</b>	<b>4</b>
3.1	How it works . . . . .	4
3.1.1	Main idea . . . . .	4
3.1.2	Naive Assumption . . . . .	4
3.1.3	Empirical estimate for discrete values . . . . .	5
3.1.4	Empirical estimate for continuous values . . . . .	5
3.2	Application . . . . .	6
3.2.1	Bayesian Estimation of Normal Distribution Parameters . . . . .	6
3.2.2	Description . . . . .	6
3.2.3	R Code . . . . .	7
3.2.4	Findings . . . . .	10

# 1 Abstract

Many popular methods of parameter estimation, such as Maximum Likelihood Estimation estimate parameters by solely using information from given data. However, in many real life scenarios, you generally have prior knowledge about the parameters. Classical statistical methods disregard this knowledge in their estimators, providing a weaker estimator than if they included it. Bayesian parameter estimation attempts to incorporate prior knowledge (distributional behavior) of the parameter in question.

In this paper, we will go through some simple examples of using bayesian parameter estimation and interval estimation. Afterwards we will demonstrate an application using bayesian inference to create a classifier. We will also compare the performance of the classifier when we use MLE vs Bayes Estimators.

## 2 Bayes Estimators

### 2.1 Point Estimation

#### Bayesian Estimation of Binomial Distribution's Parameter

Bayes Rule says:

$$\begin{aligned} f_{\theta|y}(\theta|y) &= \frac{f_{\theta|y}(\theta|y)f_{\theta}(\theta)}{f_y(y)} \\ &= \frac{f_{y|\theta}(y|\theta)f_{\theta}(\theta)}{\int f_{\theta|y}(\theta|y)f_{\theta}(\theta)d\theta} \quad (\text{By law of total probability}) \end{aligned}$$

$Y \sim \text{BIN}(n, \theta)$ , we want to find a bayes estimator for  $\theta$ , which we can get from  $\hat{\theta}_{\text{Bayes}} = T = E[f_{\theta|y}(\theta|y)]$  which is the expected value of the posterior. To use the bayes rule, we have to assume a prior distribution for  $\theta$ . For simplicity in the calculation, we will assume  $\theta \sim \text{Beta}(\alpha, \beta)$  where  $\alpha$  and  $\beta$  are known.

So now we will calculate:  $f_{\theta|y}(\theta|y) = \frac{f_{\theta|y}(\theta|y)f_{\theta}(\theta)}{\int f_{\theta|y}(\theta|y)f_{\theta}(\theta)d\theta}$  by first calculating the denominator  $f_y(y) = \int f_{\theta|y}(\theta|y)f_{\theta}(\theta)d\theta$

$$\begin{aligned}
\int f_{\theta|y}(\theta|y)f_{\theta}(\theta)d\theta &= \int \binom{n}{y} \theta^y (1-\theta)^{n-y} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} d\theta \\
&= \binom{n}{y} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int \theta^y (1-\theta)^{n-y} \theta^{\alpha-1} (1-\theta)^{\beta-1} d\theta \\
&\text{Define } \binom{n}{y} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \text{ to be } c_1. \\
&= c_1 \int \theta^{\alpha+y-1} (1-\theta)^{\beta+n-y-1} d\theta
\end{aligned}$$

Recall that the CDF of the Beta Distribution is  $= \frac{\int x^{\alpha-1} (1-x)^{\beta-1} dx}{B(\alpha, \beta)}$

$$\begin{aligned}
&\text{Define } B(\alpha_0, \beta_0) = \frac{\Gamma(\alpha_0)\Gamma(\beta_0)}{\Gamma(\alpha_0 + \beta_0)}, \text{ where for our equation } \alpha_0 = \alpha + y - 1 \text{ and } \beta_0 = \beta + n - y \\
&= c_1 B(\alpha_0, \beta_0) \int \frac{\theta^{\alpha+y-1} (1-\theta)^{\beta+n-y-1}}{B(\alpha_0, \beta_0)} d\theta \quad (\text{Multiply equation with } \frac{B(\alpha, \beta)}{B(\alpha, \beta)} = 1) \\
&= c_1 B(\alpha_0, \beta_0) \int \frac{\theta^{\alpha+y-1} (1-\theta)^{\beta+n-y-1}}{B(\alpha_0, \beta_0)} d\theta \quad \xrightarrow{1} \\
&= c_1 B(\alpha_0, \beta_0) \\
&= \binom{n}{y} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \times \frac{\Gamma(\alpha+y-1)\Gamma(\beta+n-y)}{\Gamma(\alpha+\beta+n)} \\
&= f_y(y)
\end{aligned}$$

Once we've calculated  $f_y(y)$  we can now calculate  $f_{\theta|y}(\theta|y)$ .

$$\begin{aligned}
f_{\theta|y}(\theta|y) &= \frac{f_{\theta|y}(\theta|y)f_{\theta}(\theta)}{\int f_{\theta|y}(\theta|y)f_{\theta}(\theta)d\theta} \\
&= \frac{\cancel{\binom{n}{y}} \theta^y (1-\theta)^{n-y} \cancel{\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}} \theta^{\alpha-1} (1-\theta)^{\beta-1}}{\cancel{\binom{n}{y}} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \times \frac{\Gamma(\alpha+y-1)\Gamma(\beta+n-y)}{\Gamma(\alpha+\beta+n)}} \\
&= \frac{\Gamma(\alpha+\beta+n)}{\Gamma(\alpha+y-1)\Gamma(\beta+n-y)} \theta^{\alpha+y-1} (1-\theta)^{\beta+n-y-1} \\
&= \text{Beta}(\alpha+y, \beta+n-y)
\end{aligned}$$

So  $P(\theta|Y=y) \sim \text{Beta}(\alpha+y, \beta+n-y)$  and Bayes estimator for  $\theta = E[f_{\theta|y}(\theta|y)] = \frac{\alpha+y}{\alpha+\beta+n}$

We can also set  $\alpha = \beta = 1$  to make  $\theta \sim \text{UNIF}(0, 1)$  because  $\text{Beta}(1, 1) \sim \text{UNIF}(0, 1)$ . This essentially says we have no prior info on  $\theta$  it also simplifies a lot of the computation.

## 2.2 Example of Point Estimation

Let's say we flip a coin 300 times and we observe heads 90 times. We can model this as a random variable  $X \sim \text{BIN}(300, p)$  where  $p$  is the probability of success of getting heads. We want to find the bayes estimate for  $p$ , assuming the prior distribution for  $p$  is  $f_p(p) \sim \text{Beta}(1, 1)$

From the previous proof we know that the Bayes estimator for  $p$  is  $\hat{p} = E[f_{p|y}(p|x)] = \frac{\alpha+x}{\alpha+\beta+n}$ , therefore  $\hat{p} = \frac{1+90}{1+1+300} = \frac{91}{302}$ . In comparison, the MLE of  $\hat{p} = \frac{\text{number of success}}{\text{number of trials}} = \frac{x}{n} = \frac{90}{300}$

## 2.3 Interval Estimation

Consider some parameter  $\theta$  of some distribution  $X \sim f(\theta)$ .

The Bayesian confidence interval at confidence  $(1 - \alpha)\%$  is equal to  $[F_{\frac{\alpha}{2}}(x), F_{1-\frac{\alpha}{2}}(x)]$ , where  $F_v(x)$  is the quantile of order  $v$  for the posterior distribution  $f_{\theta|X}(\theta|X)$

## 2.4 Example of Interval Estimation

Let's say flipping a coin has probability  $p$  to get heads. We toss the coin 200 times and we observe that the coin lands on heads 80 times. We assume the prior probability of parameter  $p$  to be  $P(p) \sim \text{Beta}(1, 1) = \text{UNIF}(0, 1)$ .

$X \sim \text{BIN}(200, p)$ ,  $y = 80$ ,  $n - y = 120$  and from the previously calculated bayesian point estimator for binomial we know that  $f_p(p|x) \sim \text{Beta}(1 + y, 1 + (200 - y)) = \text{Beta}(81, 121)$ .

The 95% Bayesian confidence interval for  $p$  is  $[F_{0.025}(x), F_{0.975}(x)]$ , where  $F_v$  is the quantile of order  $v$  for the posterior distribution  $f_p(p|x) \sim \text{Beta}(81, 121)$ .

95% Confidence Interval = [0.3346, 0.4693]

### 3 Gaussian Naive Bayes Classifier

#### 3.1 How it works

##### 3.1.1 Main idea

A Bayes classifier is an application of Bayes' Theorem:  $P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$  used to predict the likelihood of a class Y given the observed data X.

For example, let's say we are trying to predict the type(class) of fruit given the features  $X = x_1, x_2, x_3$  representing color, shape, and taste. The classes of fruit in our example are grape, orange, and other. We observed that  $x_1 = \text{purple}, x_2 = \text{round}, x_3 = \text{sweet}$ . What is the probability that this fruit is of class "grape"?

We can estimate this by calculating:

$$\begin{aligned} P(Y = \text{grape}|X = x_1, x_2, x_3) &= \frac{P(X|Y = \text{grape})P(Y = \text{grape})}{P(X)} \\ &= \frac{P(x_1, x_2, x_3|\text{grape})P(\text{grape})}{P(X)} \\ P(Y = \text{grape}|X = \text{purple, round, sweet}) &= \frac{P(\text{purple, round, sweet}|\text{grape})P(\text{grape})}{P(\text{purple, round, sweet})} \end{aligned}$$

##### 3.1.2 Naive Assumption

There is a problem with the previous equation, because calculating  $P(x_1, x_2, x_3|\text{grape})$  can be difficult due to the joint conditional probability, especially if we have even more features. This is where the "naive" assumption comes in, we will **assume conditional independence** between the features  $x_1, x_2, x_3$ . Hence the name, Naive Bayes Classifier. By using this assumption and the chain rule for probability, this will give us:

$$\begin{aligned} P(Y = \text{grape}|X = \text{purple, round, sweet}) &= \frac{P(\text{purple, round, sweet}|\text{grape})P(\text{grape})}{P(X)} \\ &= \frac{P(\text{purple}|\text{grape})P(\text{round}|\text{grape})P(\text{sweet}|\text{grape})P(\text{grape})}{P(\text{purple, round, sweet})} \end{aligned}$$

We can also manipulate  $P(\text{purple, round, sweet})$  such that it is easier to calculate using the **law of total probability**. We define the set of all classes to be k, in this fruit example, k = grape, orange, other.

$$P(X) = \sum_k P(X|Y = k)P(Y = k)$$

Giving us a new equation:

$$\begin{aligned} P(Y = \text{grape}|X = \text{purple, round, sweet}) &= \frac{P(\text{purple, round, sweet}|\text{grape})P(\text{grape})}{P(X)} \\ &= \frac{P(\text{purple}|\text{grape})P(\text{round}|\text{grape})P(\text{sweet}|\text{grape})P(\text{grape})}{\sum_k P(X|Y = k)P(Y = k)} \\ &= \frac{P(\text{purple}|\text{grape})P(\text{round}|\text{grape})P(\text{sweet}|\text{grape})P(\text{grape})}{P(X|\text{grape})P(\text{grape}) + P(X|\text{orange})P(\text{orange}) + P(X|\text{other})P(\text{other})} \end{aligned}$$

### 3.1.3 Empirical estimate for discrete values

The calculation will be much easier now that we no longer have the joint probability distributions. In fact we can get an empirical estimate of  $P(x_1 = \text{purple} | Y = \text{grape})$  by simply counting the proportion of occurrences of the color purple within the subset data entries pre-labeled as “grape”.

For example if the subset of observations  $X_i$  with label = “grape” is size 100 and we see the color feature to be purple for 80 of them, then the empirical estimate would be 80/100. Storing this information allows us to “train” the classifier to make future predictions on new observations.

### 3.1.4 Empirical estimate for continuous values

There is one last component to this puzzle which is how do we make empirical estimates for the conditional probability  $P(x_i | Y = \text{grape})$  if  $x_i$  is a continuous value. So far the features we have: color, shape, taste are discrete features. What if we incorporate continuous features to our model such as diameter?

A solution to this is to assume that  $P(x_i | Y = \text{grape})$  follows some distribution. In our case, we will assume that  $P(x_i | Y = \text{grape})$  follows a **gaussian distribution**. To calculate  $P(x_i | Y = \text{grape})$  we will need to estimate the mean and variance from a given feature with a given label. We can simply use sample mean and sample variance or try a bayes estimator. Note that if we had  $p$  features and  $k$  classes, we would need  $p \times k$  number of means and another  $p \times k$  variances.

Once we have the estimated means and variances, we will just calculate the probability density for each continuous entry in data  $X$ . Notice that we are calculating the density function, this makes the notation  $P(x_i | Y = \text{grape})$  inaccurate, but we will stick to it for simplicity. This brings up another question, will the value of  $P(Y | X) = \frac{P(X | Y)P(Y)}{P(X)}$  still result in a probability ranging from 0 to 1? The answer is yes, because looking at the previously expanded equation carefully:

$$\begin{aligned} P(Y = \text{grape} | X = \text{purple, round, sweet}) &= \frac{P(X | Y = \text{grape})P(Y = \text{grape})}{\sum_k P(X | Y = k)P(Y = k)} \\ &= \frac{P(X | \text{grape})P(\text{grape})}{P(X | \text{grape})P(\text{grape}) + P(X | \text{orange})P(\text{orange}) + P(X | \text{other})P(\text{other})} \end{aligned}$$

Notice that the numerator is part of the denominator. This essentially normalizes the equation to  $[0,1]$  because at most the highest possible value for  $P(Y | X)$  is 1, which is the case where  $P(X | \text{grape})P(\text{grape}) \neq 0$  and  $P(X | \text{orange})P(\text{orange}) = P(X | \text{other})P(\text{other}) = 0$ .

## 3.2 Application

### 3.2.1 Bayesian Estimation of Normal Distribution Parameters

Assume we have a random variable  $X \sim N(\mu, \sigma^2)$ , where  $\sigma^2$  is known.

prior distribution:  $f_\mu(\mu) \sim N(\mu_0, \sigma_0)$

posterior distribution:  $f_{\mu|x}(\mu|x) \sim N(((\frac{n}{\sigma^2} + \frac{1}{\mu_0})^{-1}) \times (\frac{n}{\sigma^2} \times \bar{x}] + \frac{1}{\sigma_0^2} \mu_0), (\frac{n}{\sigma^2} + \frac{1}{\mu_0})^{-1})$

### 3.2.2 Description

Here we will attempt to use the gaussian naive bayes classifier described previously to predict the movement of the Standard & Poor's 500 stock market index. The possible prediction for classes are  $Y = k$  where  $k = 0, 1$ .

$Y = 1$ , means the S&P500 has positively increased since the previous day.

$Y = 0$ , means the S&P500 has decreased or has not changed since the previous day.

We will use stock market data/metrics such as price and volume as features or predictors. We define the total number of features to be  $p$ . The total number of observation is  $n$  where each observation is one trading day.

The data matrix  $X$  will be a  $n \times p$  matrix and response(label) vector  $Y$  will be a  $n \times 1$  vector which looks like:

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1p} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{np} \end{bmatrix}, Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

where  $i = 1, \dots, n$  trading days and  $j = 1 \dots p$  stock metrics (features),  $y_n \in 0, 1$

Stock market Model:

$x_i$  =  $i$ th column of  $X$

$f_{Y|X}(Y|X) \sim \text{BIN}(1, p)$ , where  $p$  is probability that the S&P500 increases.

$f_{x_i|Y}(x_i|Y) \sim N(\mu_i, s_i^2)$ , we are assuming the variance is known and is the sample variance  $s_i^2$

$f_Y(Y)$  follows the empirical distribution

$$f_{Y|X}(Y|X) = \frac{f_{X|Y}(X|Y)}{\sum_{y \in Y} f_{X|Y}(X|Y) f_Y(Y)} \sim \text{BIN}(1, p)$$

$$f_{\mu_i}(\mu_i) \sim N(0, 1)$$

$$f_{\mu_i|x_i}(\mu_i|x_i) \sim N((\frac{n}{s_i^2} + 1)^{-1} \times [\frac{n}{s_i^2} \bar{x}_i], (\frac{n}{s_i^2} + 1)^{-1})$$

$$\text{Bayes estimator: } \hat{\mu}_{i \text{ Bayes}} = E[f_{\mu_i|x_i}(\mu_i|x_i)] = (\frac{n}{n+s_i^2} \times \bar{x}_i)$$

This shows that if we assume a prior of the standard normal distribution, the sample variance is small, the bayes estimator for the mean is very close to the mle estimator, which is just the sample average.



### 3.2.3 R Code

All of the code can be found at [https://github.com/seantey/gaussian\\_naive\\_bayes\\_for\\_sp500](https://github.com/seantey/gaussian_naive_bayes_for_sp500).

```
1 # bayes_estimator flag toggles whether to use sample mean vs bayes estimator
2 gaussian_nb <- function(X,y,bayes_estimator=FALSE){
3
4   gnb_check_input(X,y) # Predefined function which checks input validity
5
6   # sum(y) gives total days which we observe increase in S&P500
7   # length(y) gives total observations (i.e. days)
8   prior_Y_one <- sum(y) / length(y)
9   prior_Y_zero <- 1 - prior_Y_one
10
11   # Create subset data frames where Y = 0 or Y = 1
12   X_given_one_subset <- X[as.logical(y),]
13   X_given_zero_subset <- X[as.logical(1-y),]
14
15   if (bayes_estimator == FALSE){
16     mew_one = sapply(X_given_one_subset,mean)
17     sigma_one = sapply(X_given_one_subset,sd)
18
19     mew_zero = sapply(X_given_zero_subset,mean)
20     sigma_zero = sapply(X_given_zero_subset,sd)
21
22   } else {
23     # Some bayes estimator function output:
24     mew_one = sapply(X_given_one_subset,mean) # replace mean function with something
25     # else!
26     sigma_one = sapply(X_given_one_subset,sd) # replace sd function with something else
27     # !
28
29     mew_zero = sapply(X_given_zero_subset,mean) # replace mean function with something
30     # else!
31     sigma_zero = sapply(X_given_zero_subset,sd) # replace sd function with something
32     # else!
33
34   }
35
36   # Initialize a data frame of size n x p to hold P(Xij|Y=k) for each entry of matrix X
37   X_given_Y_one <- data.frame(matrix(nrow = nrow(X),ncol = ncol(X))) # Y = 1
38   X_given_Y_zero <- data.frame(matrix(nrow = nrow(X),ncol = ncol(X))) # Y = 0
39
40   # For every individual entry Xij, calculate the prob(Xij|Y = 1),
41   # recall that we are fitting a gaussian distribution for this
42
43   # Use appropriate mew & sigma corresponding to column p and class Y = k
44   for (i in 1:nrow(X)){
45     row_data = X[i,]
46
47     for (p in 1:ncol(X)){
48       prob_XY <- dnorm(row_data[[p]],mew_one[p],sigma_one[p])
49       X_given_Y_one[i,p] <- prob_XY
50     }
51   }
52
53   # For every individual entry Xij, calculate the prob(Xij|Y = 0)
54   for (i in 1:nrow(X)){
55     row_data <- X[i,]
56
57     for (p in 1:ncol(X)){
58       prob_XY <- dnorm(row_data[[p]],mew_zero[p],sigma_zero[p])
59       X_given_Y_zero[i,p] <- prob_XY
60     }
61   }
62 }
```

```

59
60 # Take product of entire row from rows 1...n
61
62 # Numerator =  $P(X_{i1}|Y=1) \dots P(X_{ip}|Y=1)P(Y=1)$  for  $i = 1 \dots n$ 
63 row_product_ones <- apply(X_given_Y_one,1,prod)
64 numerator_ones <- row_product_ones*prior_Y_one
65
66 row_product_zeros <- apply(X_given_Y_zero,1,prod)
67 numerator_zeros <- row_product_zeros*prior_Y_zero
68
69 # Add numerator for Y=1 and Y=0 to get denominator.
70 # Denominator =  $P(X_{i1}|Y=1) \dots P(X_{ip}|Y=1)P(Y=1) + P(X_{i1}|Y=0) \dots P(X_{ip}|Y=0)P(Y=0)$ 
71
72 # Calculate posterior probability  $P(Y|X)$ 
73 posteriors_ones <- numerator_ones / (numerator_ones + numerator_zeros)
74 posteriors_zeros <- numerator_zeros / (numerator_ones + numerator_zeros)
75
76 # Predictions
77 # If  $P(Y=1|X_{ij}) > P(Y=0|X_{ij})$ , i.e. likelihood of class 1 is greater,
78 # then we will label the observation row  $i$  as class  $Y = 1$ 
79 predictions <- ifelse(posteriors_ones>posteriors_zeros,1,0)
80
81 # Generate list of outputs: (1) Prior Probabilities of Y, (2) Class conditional
82 # probabilities  $P(X_{ij}|Y=k)$ , (3) Posterior probabilities, (4) Predicted Labels for
83 # each data row, (5) accuracy
84
85 output_priors <- c(prior_Y_zero,prior_Y_one)
86
87 class_conditional_DF_list <- list(X_given_Y_zero,X_given_Y_one)
88
89 posteriors_DF <- data.frame(posteriors_zeros,posteriors_ones)
90
91 accuracy <- sum(ifelse(predictions==y,1,0))/nrow(X)
92
93 # Generate a named list for easy access to elements.
94 # E.g. output_list$priors will access prior probabilities etc
95 output_list <- list(priors = output_priors, class_conditional_DF_list = class_
96 # conditional_DF_list,
97 # posteriors_DF = posteriors_DF, predictions = predictions,
98 # accuracy = accuracy)
99
100 return(output_list)
101
102 }

```

Function to check input validity. Ensures that input X is a data frame and y is a binary vector of 1's and 0's. The function also ensures the rows(length) of X and Y are equal. E.g. If X is  $n \times p$ , then y must be a  $n \times 1$  vector.

```
1 gnb_check_input <- function(X,y){
2
3   if(!is.data.frame(X)) {
4     stop("Parameter X must be a data frame, please provide the data frame whose columns
5         are numeric variables and rows are observations")
6   }
7   # Check if length of vector y matches number of rows in matrix X
8   if(!(nrow(X)==length(y))) stop("Length of response vector y must be equal to number
9       of rows in X")
10
11  # Helper function to verify response vector input.
12  binary_vector_check <- function(input_vector){
13    if(!is.integer(input_vector)) stop("Please provide an integer vector")
14    is_binary <- all(input_vector <= 1) && all(input_vector >= 0)
15    if(!is_binary) stop("Please provide an integer vector of 1's and 0's")
16    return(TRUE)
17  }
18
19  # Check if y is made of 1's and 0's
20  # Where 1 = increase in price since yesterday, 0 = decrease or no increase
21  binary_vector_check(y)
22 }
```

### 3.2.4 Findings

For simplicity our data consists of the Adjusted Closing Price & Volume from the stocks of the following 7 companies: Apple, Amazon, Nvidia, Facebook, Google, Alibaba, Intel. (Total of 14 features or columns)

We collected data from the period 2017-04-01 to 2018-05-07, which means the number of observations is 275 trading days.

Estimated Values when Y=1			
Estimator	APPL	FB	GOOG
MLE Price Mean	161.34	167.99	1002.15
Bayes Price Mean	101.62	100.13	34.19

Sample Variance when Y=1			
Stocks	APPL	FB	GOOG
Sample Variance	161.57	186.35	7785.57

We can see that the mle estimate for Google's stock Price is much different than the bayesian estimate. This is because when we assume the prior of standard normal and that our posterior variance is the sample variance, our bayesian estimate is very sensitive to large sample variances. Google's Price has a very high sample variance so the bayesian estimate is very low and very different from the mle estimate. Apple's Price's sample variance is not that large so we can see that it's bayesian estimate is closer to its mle than Google's Price's.

### Conclusion

Overall in terms of accuracy, the model with MLE performed much better than the Bayesian. This is partly due to the fact that we are assuming the prior distribution follows a standard normal and the posterior variance is just the sample variance.

Final Thoughts: Bayesian Inference is a powerful methodology that includes prior information and assumptions of parameters to improve estimations. Depending on the prior distribution, Bayesian point and interval estimation can be very similar to MLE. However, if the prior distribution is poorly assumed, the estimation can be very useless. From our experience in using Bayesian estimation we would recommend to use it only if there is actual prior knowledge on the behavior of the parameters in question.