1.(20 points for implementation + 15 points for MSE Screenshot) Please use Maximum Likelihood and Least Squares to train the model. Then, use your trained linear model to predict the chance of admit and compute the mean squared error for each data in Validation_set to get MSE < 0.01.

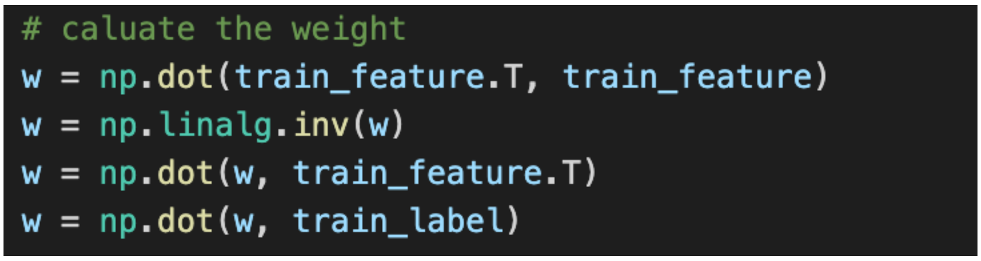
Base the formula on the test book we have the likelihood function:

$$p(\boldsymbol{t}|\boldsymbol{X},\boldsymbol{w},\beta)=\prod_{n=1}\mathcal{N}(t_n|\underbrace{\boldsymbol{w}^\top\phi(\boldsymbol{x}_n)}_{\text{mean}},\underbrace{\beta^{-1}}_{\text{var}})$$

We can use the machinery of MLE to estimate the parameters w and the precision β:

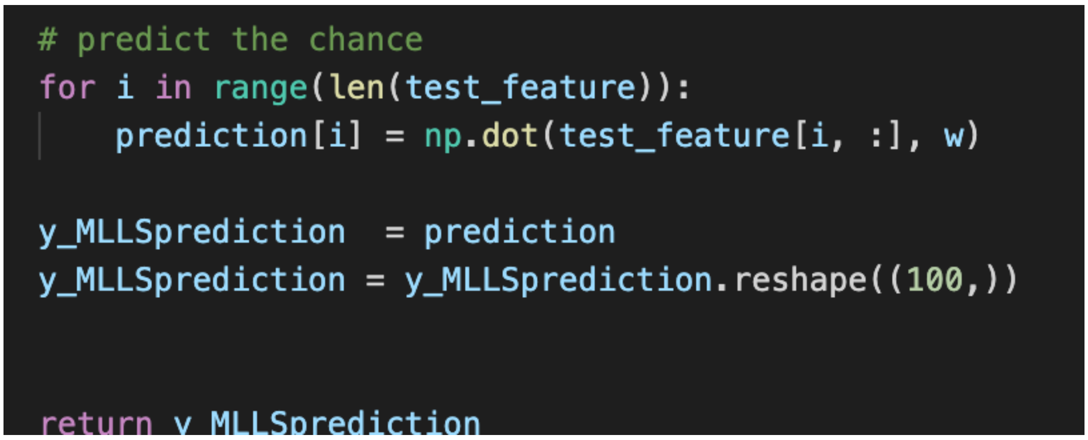
$$\boldsymbol{w}_{ML}=(\Phi^\top\Phi)^{-1}\Phi^\top\boldsymbol{t}\text{ with }\Phi_{M\times N}=[\phi_{mn}(\boldsymbol{x}_n)]$$

Here is our implementation



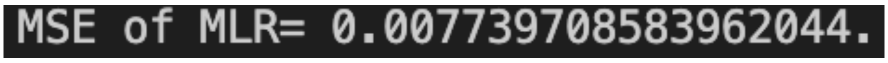
```
# caluate the weight
w = np.dot(train_feature.T, train_feature)
w = np.linalg.inv(w)
w = np.dot(w, train_feature.T)
w = np.dot(w, train_label)
```

Then we use the weight to predict the chance



```
# predict the chance
for i in range(len(test_feature)):
    prediction[i] = np.dot(test_feature[i, :], w)

y_MLLSprediction  = prediction
y_MLLSprediction = y_MLLSprediction.reshape((100,))


return y_MLLSprediction
```

And we use the least square error to observe the result



```
MSE of MLR= 0.007739708583962044.
```

2.(20 points for implementation + 15 points for MSE Screenshot) Please use Bayesian Linear Regression to estimate w. Then, use your estimated parameter to predict the chance of admit and compute the mean squared error for each data in Validation_set to get MSE < 0.01.

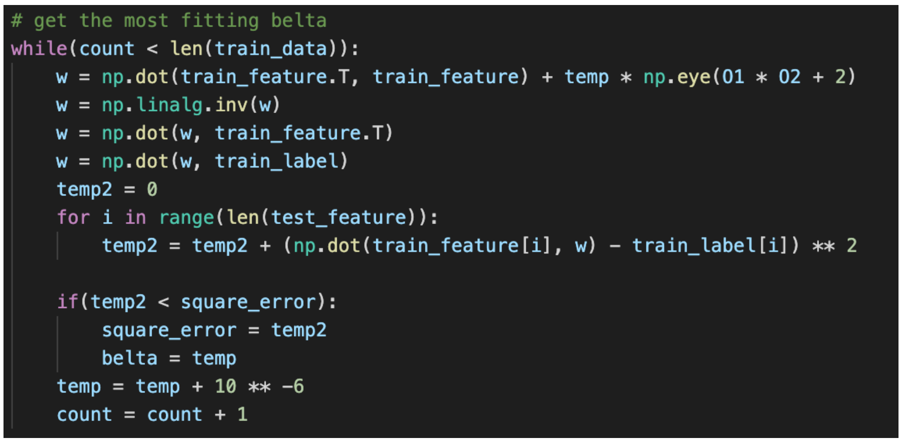
Base the formula on the test book we have the likelihood function:

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|0, \alpha^{-1}\mathbf{I})$$

$$\text{then} \quad \left| \begin{array}{rcl} \mathbf{m}_N & = & \beta\mathbf{S}_N\mathbf{\Phi}^{\mathrm{T}}\mathbf{t} \\ \mathbf{S}_N^{-1} & = & \alpha\mathbf{I} + \beta\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi} \end{array} \right.$$

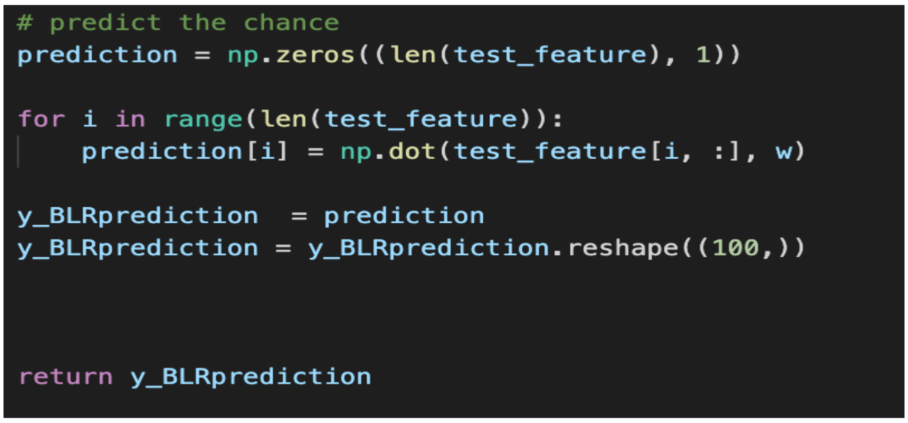
▶ Note $\alpha \rightarrow 0$ implies $\mathbf{m}_N \rightarrow \mathbf{w}_{\mathrm{ML}} = (\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi})^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{t}$

Then we use it to train on out model through the data we get the most fitting value



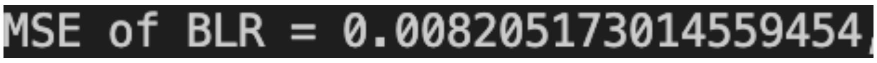
```
# get the most fitting belta
while(count < len(train_data)):
    w = np.dot(train_feature.T, train_feature) + temp * np.eye(O1 * O2 + 2)
    w = np.linalg.inv(w)
    w = np.dot(w, train_feature.T)
    w = np.dot(w, train_label)
    temp2 = 0
    for i in range(len(test_feature)):
        temp2 = temp2 + (np.dot(train_feature[i], w) - train_label[i]) ** 2

    if(temp2 < square_error):
        square_error = temp2
        belta = temp
    temp = temp + 10 ** -6
    count = count + 1
```

Then we can predict the value:



```
# predict the chance
prediction = np.zeros((len(test_feature), 1))

for i in range(len(test_feature)):
    prediction[i] = np.dot(test_feature[i, :], w)

y_BLRprediction  = prediction
y_BLRprediction = y_BLRprediction.reshape((100,))



return y_BLRprediction
```

And here is the mean square error



```
MSE of BLR = 0.00820517301455945
```

r

3.(10 points) Please discuss the difference between Maximum Likelihood and Bayesian Linear Regression, and the impact of different choices of O1 and O2 and results in your report.

Maximum Likelihood Estimate:

With MLE,we seek a point value for θθ which maximizes the

likelihood, $p(D|\theta)$p(D|θ), shown in the equation(s) above. We can denote this value

as $\hat{\theta}$ θ^. In MLE, $\hat{\theta}$ θ^ is a point estimate, not a random variable.

In other words, in the equation above, MLE treats the term $p(\theta)p(D)$p(θ)p(D) as a

constant and does NOT allow us to inject our prior beliefs, $p(\theta)$p(θ), about the likely

values for θθ in the estimation calculations.

Bayesian Estimate

Bayesian estimation, by contrast, fully calculates (or at times approximates) the

posterior distribution $p(\theta|D)$p(θ|D). Bayesian inference treats θθ as a random

variable. In Bayesian estimation, we put in probability density functions and get out probability density functions, rather than a single point as in MLE.

In conclusion, the choose of O1 and O2 is basically fit the curve below

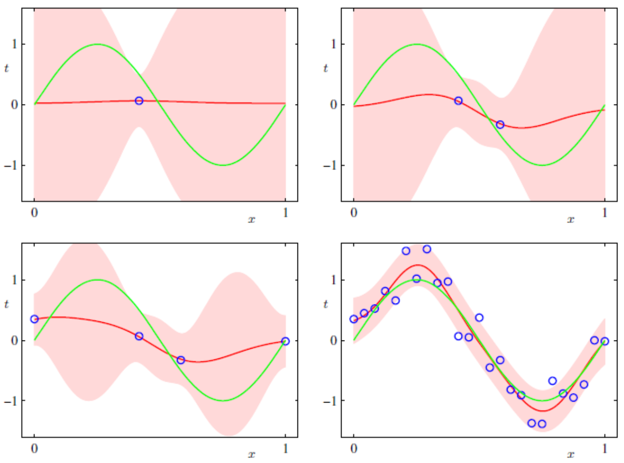


Figure 3.8 Examples of the predictive distribution (3.58) for a model consisting of 9 Gaussian basis function of the form (3.4) using the synthetic sinusoidal data set of Section 1.1. See the text for a detailed discussion.

There the setting of O1 and O2 will fit the distribution. Therefore, we should observe the curve and choose wisely.