**The Regression Part**

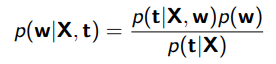
Firstly, let us look at linear regression. Since we have training data and the testing data, the linear regression are supervised machine learning problems. The idea of regression is to find a model that could fit the training data sets with the least lose.

**Question 1**

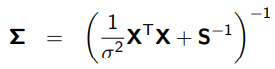
**What are the models behind the two regression algorithms you chose and implemented?**

**Bayesian Linear Regression**

Bayesian linear regression is a regression approach that takes statistical analysis under the context of Bayesian inference. Its errors follow the normal distribution. We assume a particular form of prior distribution and get the results from the posterior probability distributions.

According to the Bayes rule

Posterior density equals to the likelihood multiple prior density divide by marginal likelihood. But computing the posterior is hard since the marginal likelihood is hard to compute. Only in case when the prior and likelihood are conjugate, we know the form of posterior, therefore we know the form of the normalizing constant. So, we do not need the marginal likelihood. By calculating the covariance and mean of the training dataset, we can simply find its covariance and mean.



By minimizing loss with regularization, we adapt multivariate normal with post mean and post covariance to get result w. Finally, multiple the test data set with w to get the test result.

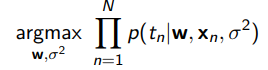
**Maximum Likelihood Regression**

Maximum likelihood Estimation of Linear Regression is a probabilistic framework for efficiently getting the probability distribution and finding the best model to describe a dataset. Specifically, the model choosing, and parameters define as a modeling hypothesis. We are trying to find the modeling hypothesis that maximize the likelihood function

Here are functions for maximum likelihood estimator for w and sigma square



Likelihood evaluates the quantity obtained when evaluating the density. The higher the value, the more likely training data is fitted. For each input-response pair, we have a Gaussian likelihood to compute the log-joint likelihood at the maximum likelihood estimates. Assume that the training data is independent, we combine them to get the joint likelihood.



Then, we calculate the covariance matrix of w



Finally, we get sample w by multivariate normal between the calculated w and cov{w}. Finally, multiple the test data set with sample w to get the test result.

**Question 2**

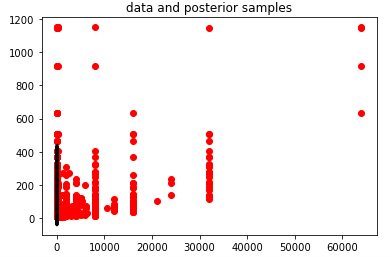
**What is your experiment setup for training these methods?**

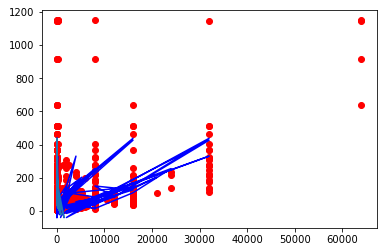
**Bayesian Linear Regression**

1. Split dataset into six parts by its features ---- Cache memory size, minimum and maximum number of I/O channels, machine cycle time, and minimum and maximum main memory. Each of them is a (168,1) list.

2. Define a Gaussian prior over 𝐰, with mean 0, a (2,2) covariance matrix and the fixed 𝜎2=2.

3. Using NumPy’s multivariate normal to generate samples from a multivariate Gaussian and sampling some 𝐰 vectors and plotting the models with all features.

4. Write functions to construct polynomial design matrix, and compute posterior mean and covariance where 𝐒 is the covariance matrix of the prior 𝑝(𝐰). Since there are 6 features, we add each polynomial dataset and compute the expected result sequentially.

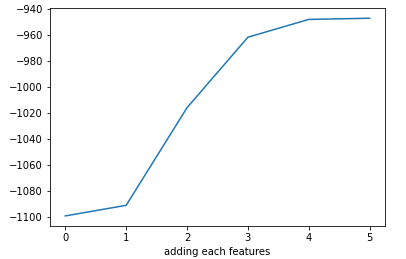
5. Look at functions for posterior prediction by plotting error bar for a general checking.

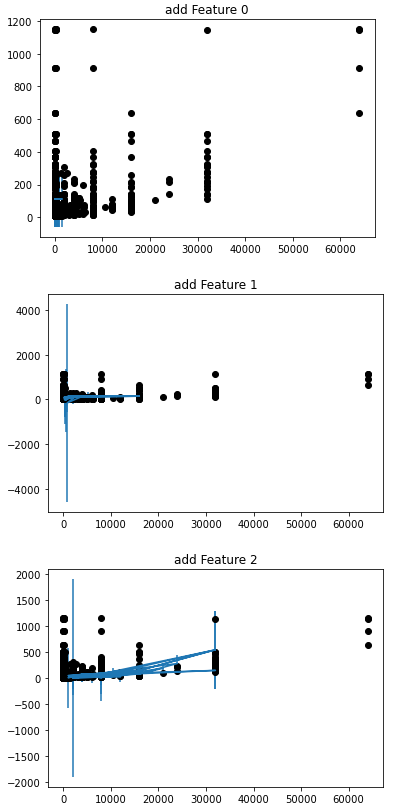
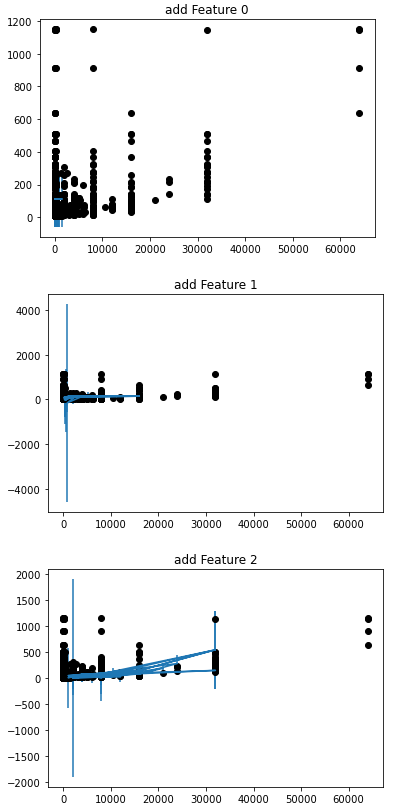
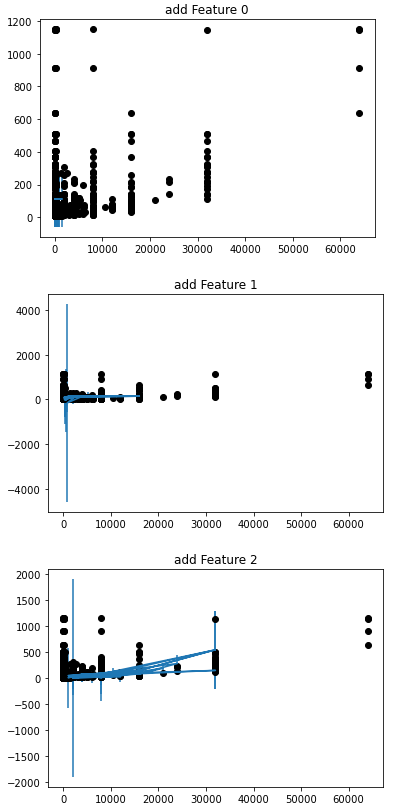
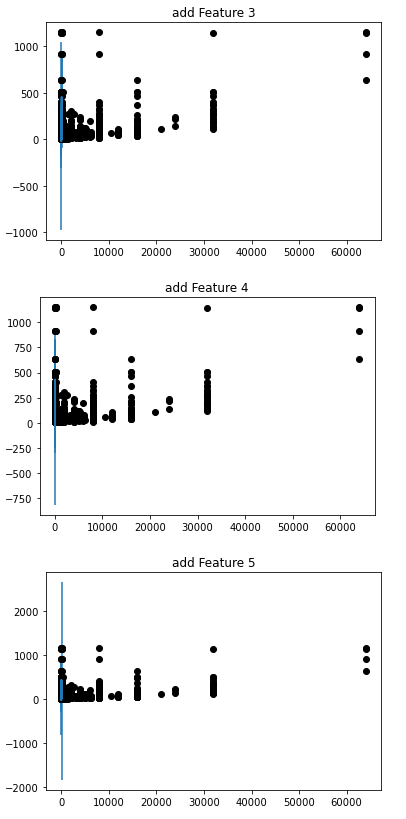
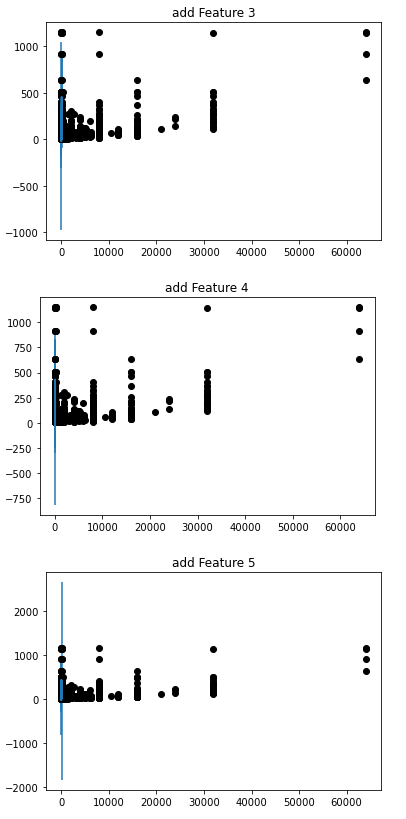
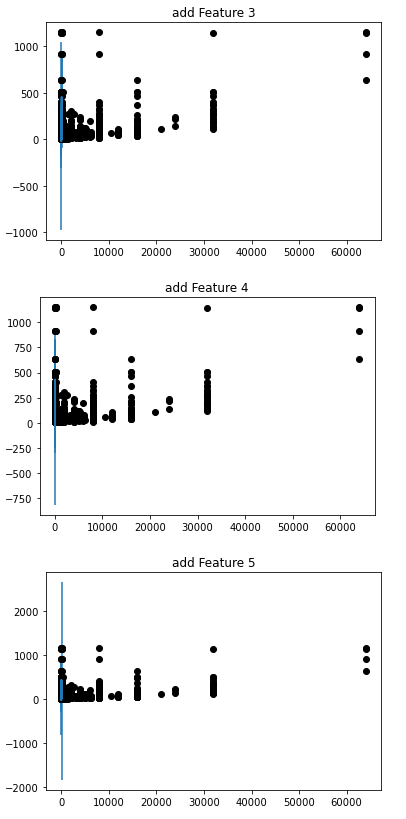
6. when the error bar looks fine, use w to generate test result from test dataset.

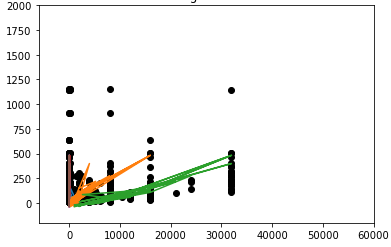
**Maximum Likelihood Regression**

1. Split dataset into six parts by its features ---- Cache memory size, minimum and maximum number of I/O channels, machine cycle time, and minimum and maximum main memory. Each of them is a (168,1) list.

2. Write function to compute log of gaussian pdf, maximum likelihood estimate of w, maximum likelihood estimate of sigma square and construct polynomial design matrix

3. Test the joint likelihood against the adding of features to check the influencing of each feature. Loop over all features, use maximum likelihood estimate of w and maximum likelihood estimate of sigma square to compute the log-joint likelihood at the maximum likelihood estimates. Plot lost between adding features.

4. Predictive variance example, step up

5. Look at functions for posterior prediction by plotting error bar for a general checking.

6. when the error bar looks fine, use w to generate test result from test dataset.

How are you going to measure the performance of your regression algorithms?



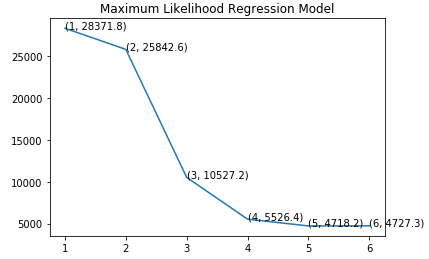
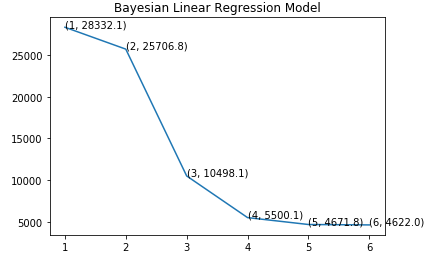
Plotting the predicted values against the real value is the best way to measure the goodness of a regression model. I use average squared loss, which measures the average error performed by the model in predicting the outcome for an observation. Mathematically, it is the average squared difference between the observed actual outcome values and the values predicted by the model. The lower the average squared loss reaches, the better the model accuracy will get.

Alternatively, we can test our test dataset on the Kaggle prediction. Somehow, it performs as a cross validation way to check the general performance. However, it is not as explicit as the average squared loss.

**Question 3**

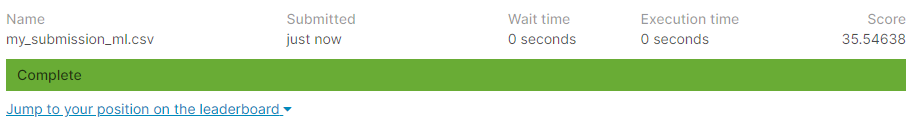
**Compare the performance of these two algorithms.**

In this section, I well adapt the average squared loss measurement onto the Bayesian Linear Regression Model and the Maximum Likelihood Regression Model.

In order to compare those two models in details, I choose to construct polynomials on training data. For each constructed feature, I generate their model and use the original data in the generated model to calculate predict data. Finally, computing the loss between the actual train data and test data by average squared loss. The least result indicates the better performance this model may have.

As we can see in the plot, in general, adding features may decrease the model lost. For some reasons, in Maximum Likelihood Regression Model, the performance of 5 features is better than adding one more feature. The performance of Bayesian Linear Regression Model is slightly better than Maximum Likelihood Regression Model.



If we predict for other datasets such as our Kaggle competition, Maximum Likelihood Regression Model may have more accuracy. As we can see the picture above, my\_submission\_ml.csv indicates a testing for Maximum Likelihood Regression Model which gets 35.5 score and my\_submission\_b.csv indicates the same testing for Bayesian Linear Regression Model which gets 38.7 score (the lower score the better performance).

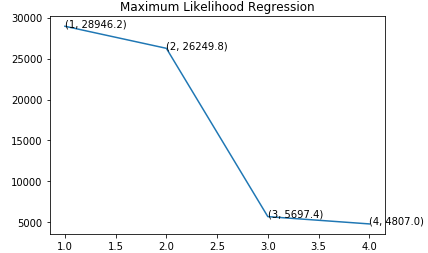
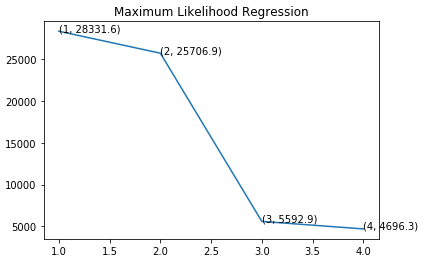
Bayesian Linear Regression Model fits the training data well and Maximum Likelihood Regression Model could perform a better prediction on wide scope of dataset. Thus, Maximum Likelihood Regression Model have a likely equally performance on general practical datasets comparing with Bayesian Linear Regression Model.

**Question 4**

**Can you obtain better performance by using only a subset of the features?**

For both models, I tried to combine some “sharpen” features. I combined minimum and maximum main memory as main memory feature and minimum and maximum number of I/O channels as I/O channels feature. So, I subtracted feature number from 6 to 4.

After that, I adapted model based on my model set up. And I used polynomial type of the original combined dataset to predict test dataset. Finally, I calculated the average squared loss and compared the performance between the flatten feature prediction.







According to the average squared loss score, the combining feature penalize the Maximum Likelihood Regression Model but slightly improve the Maximum Likelihood Regression Model. In the Kaggle competition testing data score, there is not a better performance shown.

Thus, in our hypothesis to combine the minimum and maximum dataset features, we cannot improve model performance. However, there may exist other combination ways to eliminate the dataset noise.