

MSLE, MLE, MAP model 의 수식 유도와 Auto MPG Dataset 을 통한 예측 정확도 확인 및 알고리즘의 비교

컴퓨터소프트웨어학부 2021025205 강태욱

1. Abstract

본 레포트에서는 Loss function 을 이용해 linear model 의 parameter 를 찾는 MSLE model, likelihood 를 이용해 parameter 를 찾는 MLE model, likelihood 와 prior 를 이용해 posterior 를 계산해 parameter 를 찾는 MAP model 을 유도하고 구현, 학습과 예측을 통해 알고리즘을 서로 비교한다. 세 알고리즘 모두 각각의 loss function 의 미분과 Gradient Descent 를 통해 parameter 를 추정한다. Feature 를 바탕으로 자동차의 연비를 추정하는 Auto MPG dataset 을 이용해 모델들을 학습시킨 후 MSE 를 이용한 model 과 세 model 을 비교한 결과 세 알고리즘 중 model 에 uncertainty 를 도입한 MLE model 과 parameter 에 uncertainty 를 도입한 MAP model 이 비슷한 성능을 보였고, MSLE model 은 그에 비해 0.4 배 낮은 성능을 보였다. 세 알고리즘 모두 closed-form solution 을 사용한 MSE model 에 비해 낮은 성능을 보였지만, uncertainty 를 고려하는 MLE 과 MAP model 은 적절한 variance 선정을 통한 성능 향상을 기대할 수 있다.

2. Introduction

Linear Regression 은 여러 feature 들을 통한 특정 값의 prediction 에 주로 사용된다. Prediction 과정에서는 데이터의 noise 등의 이유로 예측 결과가 바뀔 수 있으며, 이산적인 값을 도출하는 Classification Problem 에 비해 비이산적인 값을 도출하는 Prediction Problem 에서 그러한 noise 는 model 의 성능에 큰 영향을 미칠 수 있다. 따라서, Prediction Problem 을 주로 다루는 Linear Regression model 에서는 noise 를 고려하여 예측을 하는 것이 중요하다. 본 보고서를 통해 noise 를 고려하지 않은 model 과 noise 를 고려한 model 의 정확도 차이, 더 나아가 단순히 likelihood 의 noise 만 고려하는 model 과 prior 및 model 의 noise 를 고려하는 model 의 차이를 파악하고자 한다. 이러한 이유로 Linear Regression 을 위한 linear model 을 본 과제를 위한 모델로 설정하였다. 본 과제에서 사용한 linear model 은 다음과 같다.

$$f(\mathbf{x}; \mathbf{w}, b) = \mathbf{w}^T \mathbf{x} + b$$

해당 모델을 학습시킬 Dataset 은 다음과 같이 이루어져 있다.

$$\mathcal{D} = \{\mathbf{X}_i, y_i\}_{i=1}^N$$

지금까지 소개한 Data \mathbf{X}, \mathbf{y} 와 Parameter \mathbf{w}, b 의 Dimension 은 다음과 같다.

$$\mathbf{X} \in \mathbf{R}^{D \times N}, \mathbf{y} \in \mathbf{R}^{1 \times N}, \mathbf{w} \in \mathbf{R}^{D \times 1}, b \in \mathbf{R}$$

b 는 scalar 이므로 해당 값을 \mathbf{w} 의 마지막 row 에 넣고, \mathbf{X} 의 마지막 row 의 모든 원소를 1 로 설정한 후 \mathbf{X} 를 이용해 parameter 인 \mathbf{w} 를 구하여도 위에서 정의한 linear model 의 학습 및 예측 결과와 동일한 결과를 얻을 수 있다. 따라서, 이후 계산의 편의를 위해 \mathbf{X} 와 \mathbf{w} 를 다음과 같이 변경한다.

$$\mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \\ b \end{pmatrix} = \mathbf{R}^{(D+1) \times 1}$$

$$\mathbf{X} = \begin{pmatrix} \begin{matrix} | \\ \mathbf{X}_1 \\ | \end{matrix} & \begin{matrix} | \\ \mathbf{X}_2 \\ | \end{matrix} & \cdots & \begin{matrix} | \\ \mathbf{X}_N \\ | \end{matrix} \\ 1 & 1 & \cdots & 1 \end{pmatrix} = \mathbf{R}^{(D+1) \times N}$$

따라서, 변경된 parameter 를 이용하면 다음과 같은 식의 parameter 를 구함으로써 초기에 설정한 model 과 동일한 model 을 생성할 수 있다.

$$f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^T \mathbf{x}$$

3. Mean Square Log Error (MSLE)

Mean Square Log Error(MSLE)는 흔히 사용되는 Loss function 인 Mean Square Error(MSE)에 Log 를 취한 Loss Function 이다. 그 형태는 다음과 같다.

$$L = \frac{1}{N} \sum_{i=1}^N \{\ln(y_i+1) - \ln(f(\mathbf{X}_i; \mathbf{w})+1)\}^2 = \frac{1}{N} \sum_{i=1}^N (\ln(y_i + 1) - \ln(\mathbf{w}^T \mathbf{X}_i + 1))^2$$

여기서 \ln 안의 식에 1을 더한 이유는 만약 y 값이 0에 가까워질 경우 로그 내의 값이 음의 무한대로 발산하기 때문에 이를 막고자 사용한 해결책이다. 위의 식을 행렬을 이용해 표현하면 다음과 같다.

$$L = \frac{1}{N} \{\ln(\mathbf{y} + 1) - \ln(\mathbf{w}^T \mathbf{X} + 1)\} \{\ln(\mathbf{y} + 1) - \ln(\mathbf{w}^T \mathbf{X} + 1)\}^T$$

Loss function 을 미분하여 그 값을 토대로 Loss function 의 미분값이 0이 되도록 하는 \mathbf{w} 를 찾거나 (analytic method), 미분값을 바탕으로 Gradient Descent 를 통해 Loss function 이 최솟값을 가지도록 하는 \mathbf{w} 를 구할 수 있다 (numerical method). 다음의 식은 해당 과정을 수행하기 위해 Loss function 을 parameter \mathbf{w} 에 대해 미분하는 과정이다.

$$\frac{\partial L}{\partial \mathbf{w}} = -\frac{2}{N} \left(\frac{\mathbf{X}}{\mathbf{w}^T \mathbf{X} + 1} \right) \{\ln(\mathbf{y} + 1) - \ln(\mathbf{w}^T \mathbf{X} + 1)\}^T$$

식의 \ln 함수로 인해 위의 미분값이 0이 되도록 하는 \mathbf{w} 의 closed-form solution 을 구하기에는 어려움이 있다. 따라서, Gradient Descent 를 적용해 최솟값을 가지는 \mathbf{w} 를 구하고자 한다.

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \frac{\partial L}{\partial \mathbf{w}} = \mathbf{w}_t + \frac{2\alpha}{N} \left(\frac{\mathbf{X}}{\mathbf{w}^T \mathbf{X} + 1} \right) \{\ln(\mathbf{y} + 1) - \ln(\mathbf{w}^T \mathbf{X} + 1)\}^T$$

해당 과정을 통해 model 의 parameter \mathbf{w} 를 구할 수 있다.

4. Maximum Likelihood Estimation (MLE)

Introduction 에서 설정한 linear model 에 noise 를 추가함으로써 model 의 overfitting 을 방지하고, 불확실성을 표현하는 등의 장점을 얻을 수 있다. 아래의 식은 이를 위해 앞서 선정한 model 에 noise 를 추가한 Probabilistic model 이다.

$$f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^T \mathbf{x} + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2)$$

학습 과정에서 사용되는 parameter 에 대한 likelihood function 은 다음과 같다.

$$p(\mathcal{D}|\mathbf{w}) = p(\mathbf{y}|\mathbf{X}, \mathbf{w})$$

Probabilistic model 은 parameter \mathbf{w} 가 고정되어 있는 상태에서 model 의 noise 를 고려하므로 prior 인 $p(\mathbf{w})$ 를 고려하지 않는다. 따라서, 해당 likelihood function $p(\mathcal{D}|\mathbf{w})$ 을 maximize 하는 것은 Data 를 가장 잘 설명하는 \mathbf{w} 를 찾는 것과 동일하다. 아래는 likelihood function 의 최댓값을 구하기 위해 식을 변형하는 과정이다.

$$\operatorname{argmax}_{\mathbf{w}} p(\mathcal{D}|\mathbf{w}) = \operatorname{argmax}_{\mathbf{w}} p(\mathbf{y}|\mathbf{X}, \mathbf{w})$$

$p(y_i|\mathbf{X}_i, \mathbf{w})$ 는 Gaussian distribution 을 따른다고 가정하자.

$$p(y_i|\mathbf{X}_i, \mathbf{w}) \sim \mathcal{N}(\mathbf{w}^T \mathbf{X}_i, \sigma^2)$$

$$p(y_i|\mathbf{X}_i, \mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(y_i - \mathbf{w}^T \mathbf{X}_i)^2}{2\sigma^2}\right\}$$

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \prod_{i=1}^N C \exp\left(-\frac{(y_i - \mathbf{w}^T \mathbf{X}_i)^2}{2\sigma^2}\right), C = \text{constant}$$

Likelihood 의 최댓값을 구하기 위해선 이를 미분하여야 한다. 그러나 product 가 있기에 식을 단순 미분하기 어렵다. 이 문제를 해결하기 위해 양 변에 \ln 함수를 적용하여 product 를 sum 으로 바꾸어 미분이 쉬운 형태로 식을 변형할 수 있다.. 로그 함수는

단조 증가 함수이기에 likelihood 에 로그 함수를 취해도 likelihood 가 최댓값이 되는 \mathbf{w} 는 변하지 않는다. 따라서, 위의 식을 아래처럼 변경할 수 있다.

$$\ln p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \ln \sum_{i=1}^N \exp(-\frac{(y_i - \mathbf{w}^T \mathbf{X}_i)^2}{2\sigma^2}) + C = -\sum_{i=1}^N \frac{(y_i - \mathbf{w}^T \mathbf{X}_i)^2}{2\sigma^2} + C$$

$$\therefore \ln p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = -\frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{w}^T \mathbf{X})(\mathbf{y} - \mathbf{w}^T \mathbf{X})^T + C$$

즉, 다음의 식은 \mathbf{w} 를 구하기 위해 최종적으로 계산해야 하는 식이다.

$$\operatorname{argmax}_{\mathbf{w}} \ln p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \operatorname{argmax}_{\mathbf{w}} -\frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{w}^T \mathbf{X})(\mathbf{y} - \mathbf{w}^T \mathbf{X})^T + C$$

위의 식에서 최대화해야 하는 식은 하나의 Loss function 으로 볼 수 있다. 따라서, 새로운 Loss function L 은 다음과 같이 정의된다.

$$L = -\frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{w}^T \mathbf{X})(\mathbf{y} - \mathbf{w}^T \mathbf{X})^T + C$$

L 을 미분하면 다음의 식과 같다.

$$\frac{dL}{d\mathbf{w}} = \frac{\mathbf{X}(\mathbf{y} - \mathbf{w}^T \mathbf{X})}{\sigma^2}$$

해당 식에 대해 Gradient Descent 를 적용하여 parameter \mathbf{w} 를 구할 수 있다. 최댓값을 구하는 문제이므로 MSLE 의 경우와 달리 L 의 미분값을 더해주며 \mathbf{w} 를 update 한다.

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \alpha \frac{dL}{d\mathbf{w}} = \mathbf{w}_t + \frac{\alpha \mathbf{X}(\mathbf{y} - \mathbf{w}^T \mathbf{X})}{\sigma^2}$$

5. Maximum A Posteriori (MAP)

MLE 에서는 parameter \mathbf{w} 에 대해 Bayesian extension 을 적용하지 않았기 때문에 likelihood 를 통해 \mathbf{w} 를 구했다. 그러나, 사전 정보를 활용하기 위해 \mathbf{w} 또한 Bayesian

distribution 을 따름을 가정한 후 likelihood 와 prior 를 통해 posterior 를 계산하여 더 정확한 model 을 만들 수 있다.

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$$

$p(\mathcal{D}|\mathbf{w})$ 는 likelihood(model), $p(\mathbf{w})$ 는 prior, $p(\mathbf{w}|\mathcal{D})$ 은 posterior 이다.

Likelihood 와 prior 모두 gaussian 이라고 가정하면, posterior 또한 gaussian 이다.

Gaussian distribution 을 따르는 posterior 를 구하기 위해 likelihood 와 prior 를 다음과 같이 정의할 수 있다.

$$p(\mathbf{w}) \sim \mathcal{N}(\mathbf{0}, \sigma_0^2 \mathbf{I}), \quad p(\mathcal{D}|\mathbf{w}) \sim \mathcal{N}(\mathbf{w}^T \mathbf{X}, \sigma^2)$$

$$p(\mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{|\mathbf{w}|^2}{2\sigma_0^2}\right)$$

$$p(\mathbf{y}|\mathbf{X}, \mathbf{w}) = \prod_{i=1}^N p(y_i|\mathbf{X}_i, \mathbf{w}) = \prod_{i=1}^N \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(\mathbf{y}_i - \mathbf{w}^T \mathbf{X}_i)^2}{2\sigma^2}\right\}$$

정확도가 높은 model 을 생성하기 위해서는 Dataset 에 대한 parameter 의 예측 확률이 높은, 즉 높은 $p(\mathbf{w}|\mathcal{D})$ 를 만드는 \mathbf{w} 를 찾아야 한다. 그러나, $p(\mathbf{w}|\mathcal{D})$ 에는 product 를 포함하고 있으므로 이를 바로 미분하는 것은 어렵다. 이 문제를 해결하기 위해 MLE 와 같은 이유로 posterior 의 양 변에 로그를 취한다.

$$\ln p(\mathbf{w}|\mathcal{D}) = \ln p(\mathcal{D}|\mathbf{w}) + \ln p(\mathbf{w}) - \ln p(\mathcal{D})$$

여기서 $\ln p(\mathcal{D})$ 는 Data 에 대한 확률이므로, \mathbf{w} 의 분포에 영향을 주지 않는다. 따라서, 이를 상수 취급하고 식을 미분하여도 무방하다. 즉, 아래의 식을 통해 최적의 \mathbf{w} 를 찾을 수 있다.

$$\operatorname{argmax}_{\mathbf{w}} \ln p(\mathbf{w}|\mathcal{D}) = \operatorname{argmax}_{\mathbf{w}} \{\ln p(\mathcal{D}|\mathbf{w}) + \ln p(\mathbf{w})\}$$

$$\ln p(\mathcal{D}|\mathbf{w}) = \ln \sum_{i=1}^N \exp\left\{-\frac{(\mathbf{y}_i - \mathbf{w}^T \mathbf{X}_i)^2}{2\sigma^2}\right\} + C$$

$$\ln p(\mathbf{w}) = -\frac{\mathbf{w}^T \mathbf{w}}{2\sigma_o^2} + C$$

$$\operatorname{argmax}_{\mathbf{w}} \ln p(\mathbf{w}|\mathcal{D}) = \operatorname{argmax}_{\mathbf{w}} -\frac{(\mathbf{y}-\mathbf{w}^T \mathbf{X})(\mathbf{y}-\mathbf{w}^T \mathbf{X})^T}{2\sigma^2} -\frac{\mathbf{w}^T \mathbf{w}}{2\sigma_o^2}$$

$$\therefore \operatorname{argmax}_{\mathbf{w}} p(\mathbf{w}|\mathcal{D}) = \operatorname{argmin}_{\mathbf{w}} \frac{(\mathbf{y}-\mathbf{w}^T \mathbf{X})(\mathbf{y}-\mathbf{w}^T \mathbf{X})^T}{2\sigma^2} + \frac{\mathbf{w}^T \mathbf{w}}{2\sigma_o^2}$$

여기서 최소화해야 하는 값은 하나의 Loss function 이라고 볼 수 있다. 따라서, 새로운 Loss function 은 다음과 같이 정의된다.

$$L = \frac{(\mathbf{y}-\mathbf{w}^T \mathbf{X})(\mathbf{y}-\mathbf{w}^T \mathbf{X})^T}{2\sigma^2} + \frac{\mathbf{w}^T \mathbf{w}}{2\sigma_o^2}$$

위의 식을 미분하면 아래와 같다.

$$\frac{dL}{d\mathbf{w}} = -\frac{\mathbf{X}(\mathbf{y}-\mathbf{w}^T \mathbf{X})^T}{\sigma^2} + \frac{\mathbf{w}}{\sigma_o^2}$$

해당 식은 \mathbf{w} 에 대한 closed-form solution 을 구하기 어려운 형태이다. 따라서, Gradient Descent 를 통해 Loss function 을 최소화하는 parameter \mathbf{w} 를 구한다.

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \frac{dL}{d\mathbf{w}} = \mathbf{w}_t + \frac{\alpha \mathbf{X}(\mathbf{y}-\mathbf{w}^T \mathbf{X})^T}{\sigma^2} - \frac{\alpha \mathbf{w}}{\sigma_o^2}$$

6. Experiments

위 문단에서 유도한 MSLE, MLE, MAP 의 성능을 확인하기 위해 모델을 코드로 작성해 데이터셋을 이용해 모델을 학습시킨 후, 그 정확도를 측정하는 실험을 고안하였다. 대표적인 prediction dataset 인 Auto MPG dataset 을 이용해 실험을 설계했다.

a. Auto MPG dataset

Auto MPG 는 자동차의 연비와 그와 관련된 데이터를 담고 있는 데이터셋이다. 각 샘플은 mpg, cylinders, displacement, horsepower, weight,

acceleration, model year, origin, car name 의 feature 를 가지고 있다. 샘플 의 feature 를 통해 mpg 를 predict 하는 linear model 을 만든 후, model 의 정확도 및 uncertainty 를 서로 비교하는 것이 이번 실험의 목적이다.

기본 Auto MPG Dataset 에는 feature 와 결과값인 y 가 같이 포함되어 있다. 따라서, 학습 전 feature 와 y 를 분할하는 작업이 필요하다. 또한, 수식 유도 편의를 위해 bias 를 parameter \mathbf{w} 내에 삽입하였기 때문에 이에 대한 데이터셋의 처리 또한 필요하다. 마지막으로, 기본 데이터셋은 $N \times D$ 차원의 행렬이므로 앞선 수식에 맞도록 데이터셋의 차원을 $D \times N$ 으로 변경하는 작업이 필요하다.

예측의 정확도를 위해 기존 feature 의 값을 그대로 사용하고자 했으나, 계산 과정에서 overflow 가 발생하였다. 기존 feature 에서는 샘플 중 일부가 큰 값의 feature 를 가져 이러한 오류가 발생하는 것으로 추정하고 대부분의 해당 특성을 지니는 feature (displacement, horsepower, acceleration, weight)를 0 과 1 사이의 값으로 정규화 시키는 Preprocessing 을 진행했다.

Model 을 학습시키고, 그 성능을 테스트하기 위해서는 데이터셋을 training data 와 test data 로 분할하여야 한다. Training data 와 test data 를 8:2 비율로 분할하여 학습과 성능 테스트를 수행하였다.

b. MSLE, MLE, MAP

각 Model 들을 생성하기 위해 Gradient Descent 을 사용하였다. 실험 과정에서 모델들에 적용되는 learning rate 을 0.000001, iteration 수를 100000 로 모두 동일하게 함으로서 모델 비교 과정에서 변인을 통제하였다. 앞서 유도한 각 알고리즘의 gradient 를 다음과 같이 코드로 작성하여 학습 과정에서 model 의 핵심 기능을 구현하였다.

```
# Loss function의 미분값 계산
p1 = np.log(y+1) - np.log(y_hat+1)
dpl = X/(y_hat+1)
dL = -1 * (2/N) * np.dot(dpl, p1.T)
```

Figure 1 MSLE 의 Gradient 구현

```
# Loss function의 미분값 계산
p1 = y - np.dot(self.w.T, X)
dL = np.dot(X, p1.T) / var
```

Figure 2 MLE 의 Gradient 구현


```
# Loss function의 미분값 계산
pl = y - np.dot(self.w.T, X)
dM = (np.dot(X, pl.T) / self.var_model) - (self.w / self.var_w)
```

Figure 3 MAP 의 Gradient 구현

위의 방법을 통해 구한 Gradient 를 이용해 아래와 같은 방법으로 Gradient Descent 를 최종 구현한다. 아래의 코드는 MSLE 에서의 Gradient Descent 구현의 예시이다.

```
for i in range(self.iter):
    y_hat = np.dot(self.w.T, X)

    # Loss function의 미분값 계산
    pl = y - np.dot(self.w.T, X)
    dL = np.dot(X, pl.T) / var

    # parameter update
    self.w += (self.learning_rate * dL)
```

Figure 4 MSLE 에서의 Gradient Descent 구현

또한, MLE 에서 likelihood 의 variance, MAP 에서 prior 와 likelihood 의 variance 는 모두 1로 설정하였다.

c. Model 평가 방법

Model 을 평가하기 위해 흔히 사용되는 Loss function 인 Mean Square Error (MSE)를 활용했다. MSE 의 수식은 다음과 같다.

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - f(\mathbf{x}_i; \mathbf{w}))^2$$

MSE 는 실제 결과값(y_{test})과 예측 결과값(prediction) 간 차의 제곱의 평균이다. 따라서, MSE 값이 작을수록 정확도가 높은 model 이라고 판단할 수 있다.

또한, MSE 자체를 Loss function 이라고 볼 수 있다. MSE 를 Loss function 으로 사용할 경우 parameter \mathbf{w} 를 analytic 하게 구할 수 있다. 해당 \mathbf{w} 는 다음과 같은 closed-form solution 을 가진다.

$$\mathbf{w}^* = (\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{X}\mathbf{y}$$

다음의 parameter 를 사용한 model 의 MSE 값과 MSLE, MLE, MAP 의 MSE 값을 비교해 모델의 성능을 비교할 수 있다.

또한, x 축으로 y_test, y 축으로 prediction 을 나타내는 점 그래프를 그려 실제 데이터와 model 을 통한 예측 값의 일치 정도와 그 경향성을 파악하고자 했다. 점의 분포가 $y = x$ 에 가까울수록 예측의 정확도가 높다는 것을 바탕으로 model 의 성능을 파악 및 비교하도록 하였다.

7. Experimental results

a. Prediction solution

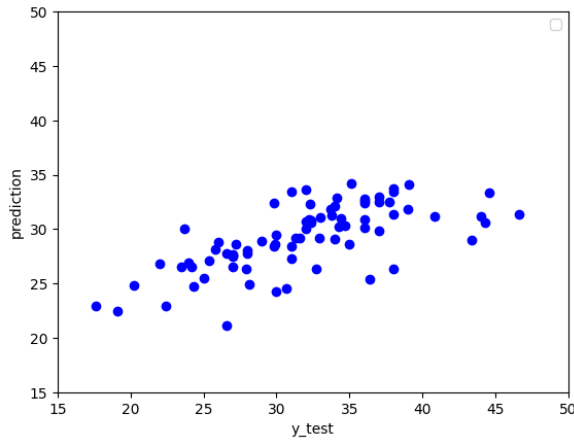


Figure 5 MSE Result

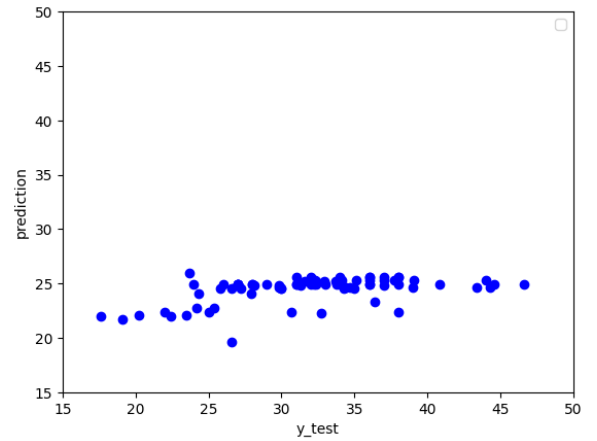


Figure 6 MSLE Result

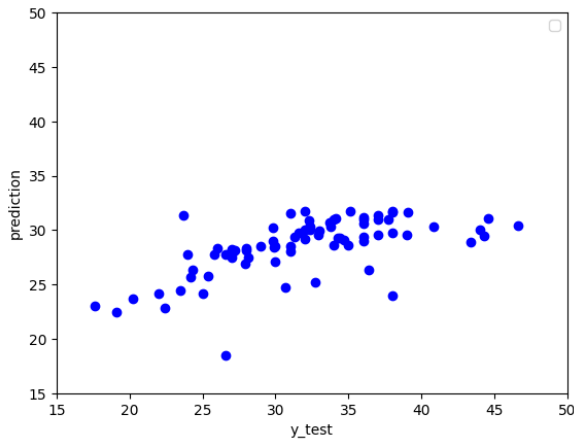


Figure 7 MLE Result

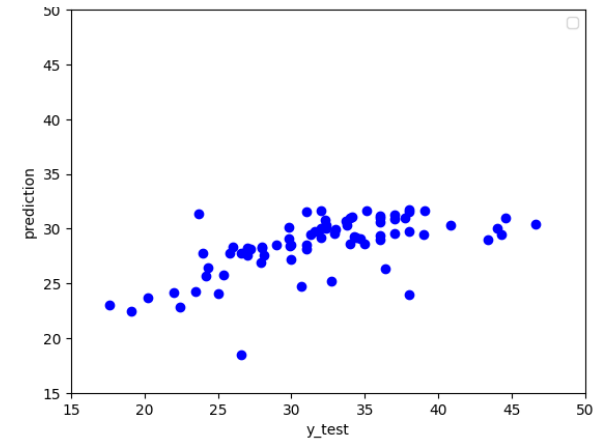


Figure 8 MAP Result

그래프 상으로 보았을 때 Closed-form solution 으로 parameter 를 직접 구한 MSE 를 사용한 model(비교군)의 점의 분포가 이상적인 분포에 가장 가깝게

나타났다. 그 뒤로 MLE 와 MAP 이 비슷한 분포를 보였으며, MSLE 는 $y = 25$ 에 가까운 점의 분포를 보였다.

MSE 는 Loss function 이 최솟값이 되는 w^* 를 직접 구했기 때문에 Gradient Descent 를 통해 최솟값의 근사치를 구한 다른 model 에 비해 더 높은 정확도를 보인 것으로 추측된다. 그러나, Data preprocessing 의 영향으로 인해 실제 데이터와는 값의 범위에 있어 어느정도 차이가 존재하는 예측 값의 분포를 보였다.

MSLE 는 분포상으로 정확도가 네 알고리즘 가운데 가장 떨어지는 모습을 보였다. Loss function 내부에서 로그를 사용하기 때문에 실제 데이터와 예측 데이터 사이의 차이가 상대적으로 더 작게 나타났고, 이로 인해 parameter 의 gradient 또한 MSE 에 비해 작은 값으로 나타나 parameter 의 변화가 좁은 범위 내에서 이루어진 것으로 추측된다.

MLE 와 MAP 는 서로 비슷한 점의 분포를 보였다. 또한, 비교적 MSE 의 분포와 비슷한 분포를 보였다. 이를 통해 MSLE 와 비교할 때, Loss function 이 서로 달라 객관적인 비교는 어렵겠지만 model 의 uncertainty 를 고려하지 않는 것 보다 고려하는 것이 더 높은 정확도를 보이는 것을 알 수 있다. 또한 MLE 와 MAP 의 예측값 분포 범위가 MSE 의 분포 범위보다 좁다는 것을 알 수 있다. Varicance 의 영향, data preprocessing 의 영향 등 다양한 요인이 이러한 결과에 영향을 줄 수 있기에 추후 실험 및 검증을 통해 이 부분에 대한 파악이 필요해 보인다.

b. Uncertainty

Table 1 Model 별 Uncertainty(MSE)

	MSE	MSLE	MLE	MAP
Uncertainty (MSE)	27.463	84.743	34.330	34.409

위 표에서 볼 수 있듯, MSE, MLE, MAP, MSLE 순으로 낮은 Uncertainty 를 보였다. Uncertainty 가 낮을수록 model 의 정확도가 높다는 뜻이므로 MSE 의 정확도가 제일 높고 MSLE 의 정확도가 가장 낮다는 것으로 해석할 수 있다. Uncertainty 를 통해 likelihood 를 이용한 MLE 의 정확도가 likelihood 와 prior 를 모두 사용해 posterior 를 계산한 MAP 의 정확도보다 높다는 것을

관찰할 수 있다. 이론 상 prior 를 고려하는 MAP 가 MLE 보다 낮은 Uncertainty 를 보여야 하지만, 반대의 실험 결과가 도출되었다. 이에 대해 model 에 맞는 적당한 variance 를 설정하지 않았다는 점, parameter 의 uncertainty 를 고려하기에 model 의 uncertainty 가 더 높게 나타날 수 있다는 점 등을 해당 결과의 원인으로 추측했다. 그러나 MLE 와 MAP 간의 uncertainty 가 크게 차이가 나지 않으며, 데이터셋 자체의 원인 등이 존재할 수 있기에 해당 결과의 유의미한 해석에는 한계가 존재한다.

8. Discussion

MAP 에 대한 식을 전개하며 prior 와 likelihood 의 mean 과 variance 를 학습 과정에서 update 해줘야 하는지에 대한 궁금증이 생겼다. 이를 해결하고자 기계학습 알고리즘 수업 시간에 유도한 posterior 의 mean 과 variance 의 수식을 바탕으로 mean 과 variance 를 update 하는 MAP model 을 구현하여 이를 학습시키고자 했으나, 계산 과정에서 너무 작은 수의 계산으로 인해 overflow 가 발생하여 유의미한 결과를 도출하지 못했다. 계산의 overflow 와 관련된 문제를 해결해 mean 과 variance 를 계속 update 하는 것이 더 좋은 성능의 model 을 만드는지, 혹은 비효율적인 계산과 유의미하지 않은 성능 차이로 인해 해당 model 이 실사용에 문제를 가지고 있는지를 파악해보고 싶다.

또한, model 을 학습시키는 과정에서 식 전개 과정 중 전혀 고려하지 않았던 계산의 overflow 문제가 종종 발생하였다. 이를 방지하기 위해 preprocessing 을 진행했지만, 이로 인해 예측의 정확도가 떨어지는 문제가 발생했다. Overflow 를 최대한 방지할 수 있는 알고리즘을 설계하는 적절한 방법이 있는지, 또는 기존 데이터의 경향성을 최대한 보존하면서 model 의 학습 과정에서 overflow 가 발생하지 않도록 해 높은 정확도를 보이는 model 을 설계할 수 있는지에 대해 추가적으로 연구해보고자 한다.

9. Conclusion

본 보고서를 통해 단순한 Loss function 을 최소화하는 MSLE model, likelihood 를 최대화하는 parameter 를 찾는 MLE model, likelihood 와 prior 를 이용해 posterior 를 최대화하는 MAP model 의 알고리즘을 유도하고 구현하여 그 성능을 평가했다.

Noise 를 고려하지 않는 MSLE model 의 정확도가 제일 낮게 나왔으며, Noise 를 고려하는 model 인 MLE model 과 MAP model 의 성능은 서로 유사하게 나타났다. 그러나, 세 model 모두 analytic 한 solution 을 가진 MSE model 에 비해선 성능이 낮게 나왔다. Model 의 적절한 variance 를 파악하지 않고 모두 1로 통일해 학습을 진행했다는 점, 수식 전개 과정에서의 오류의 존재 가능성, 데이터셋 자체에 noise 가 크게 존재하지 않을 가능성 등 여러 요인이 해당 결과의 원인으로 추측된다. 그러나, MLE model 과 MAP model 모두 적절한 variance 설정으로 성능 향상의 가능성이 있는 점은 긍정적이다. 특히 MAP model 의 경우 prior 와 likelihood 의 variance 를 모두 고려하기에 적절한 variance 가 설정된 경우, 데이터셋에 noise 가 많이 포함된 경우 다른 Linear Regression model 보다 더 좋은 정확도를 보일 것으로 기대된다.

MAP model 과 같은 Bayesian approach 를 사용한 model 의 성능을 향상시키기 위해 적절한 variance 를 선정하는 방법에 대한 학습 및 연구가 더 필요할 것 같다. 또한, 본 과제에서는 단순히 Loss function 을 사용하는 model, likelihood 를 고려하는 model, posterior 를 고려하는 model 사이의 loss function 이 서로 달라 객관적인 성능의 비교에 어려움이 있었다. 만약 이에 관한 후속 연구를 진행한다면 세 model 모두 기본적으로 같은 loss function 을 이용하여 probabilistic model, Bayesian model 을 생성한 뒤 그들의 성능을 객관적으로 비교해보고자 한다.

본 과제를 수행하며 사용된 코드는 아래의 github 링크를 통해 확인할 수 있다.

https://github.com/twkang13/CSE3037_HYU_Machine_Learning_Theories