신용카드 사용자 연체 예측 AI 경진대회

지구온나나나팀: 박태익 손성만 백한별

2021-06-04

MLR

1. MLR?
2. MLR 모수추정
3. MLR vfold 구하기
4. Q&A

1. MLR?

Multinomial Logistic Regression, 다항 로지스틱 회귀, 다중 분류

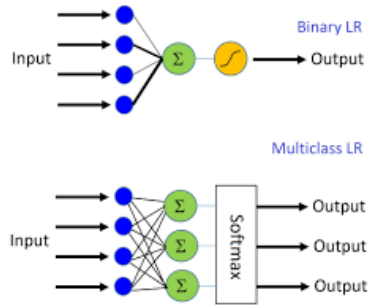


그림1

▷ 로지스틱의 일종으로 선형 회귀가 아닌 **분류** 형태(범주형 회귀)

▷ 로지스틱에서 시그모이드를 써서 선형 회귀 값을 0과 1로 분류**(2항 분류)**한 것처럼,

- MLR은 **선형 회귀 값을 후처리**해서 **3개 이상**으로 분류하는 **일반화**된 형태
- 예) 수능 등급(1~9등급), 학점(A+, A, ...), 등

1. MLR?

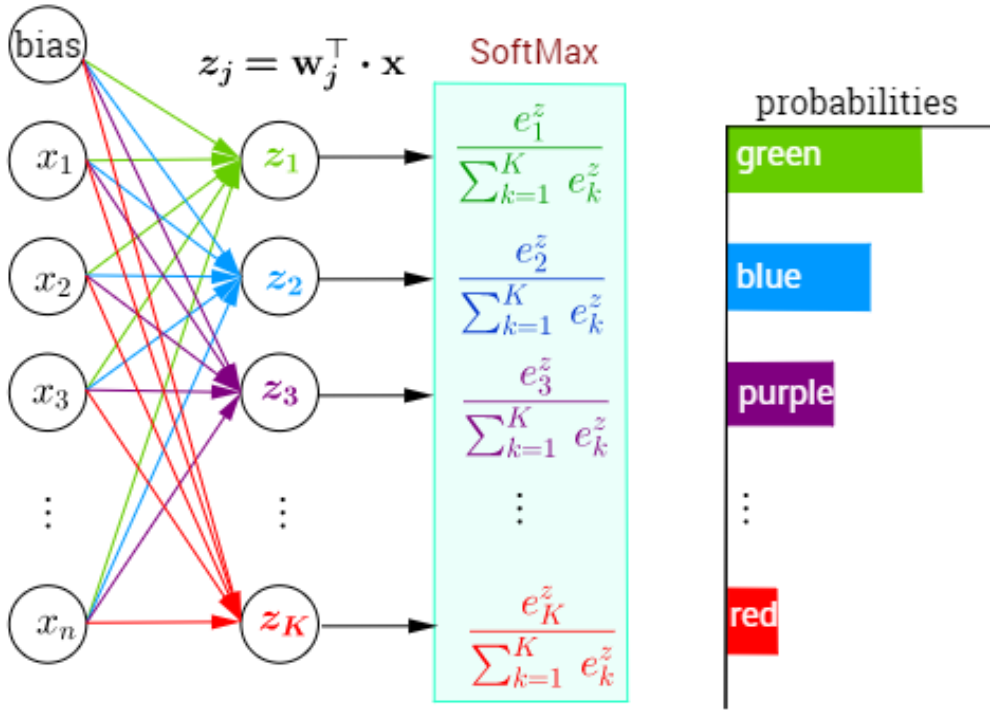


그림2

▷ 후처리 함수, 활성화함수(Activation function), Softmax

1. MLR?

▷ **(장점)** 다중 분류 용이, ML 등에 사용 **(단점)** 계수의 직관적 해석 어려움

▷ 데이콘 신용카드 대회 사례

- 성별, 차량소유, 등등의 변수에 따라 신용등급을 0, 1, 2로 분류

```
1) 0인가 아닌가
2) 1인가 아닌가
3) 2인가 아닌가
```

☞ 2항 분류인 로지스틱 모델 말고 **MLR 모델을 적용**

2. MLR 모수추정

▷ 오버피팅 방지를 위한 튜닝(정규화)

: 모수 `penalty`와 `mixture`의 최적값을 찾기

```
mlr_spec <- multinom_reg(penalty = tune(),
                         mixture = tune()) %>%
  set_engine("glmnet") %>%
  set_mode("classification")
```

`multinom_reg()`

> - `penalty`와 `mixture`를 `tune()`으로 세팅
> - `penalty`: 람다(λ), 정규화를 위한 배수
> - `mixture`: 알파(α), LASSO 비율 (1:LASSO ~ 0:Ridge)
> - `set_engine`: glmnet package안에 있는 multinom_reg을 사용
> - `set_mode`: classification 문제

2. MLR 모수추정

▷ 튜닝에 들어갈 `penalty`와 `mixture`의 샘플 만들기

```
set.seed(2021)
mlr_grid <- grid_latin_hypercube(penalty(), mixture(),  size = 100)
```

`grid_latin_hypercube()`

- seed를 고정 : 고정을 하지 않으면 그리드가 바뀌어서 다시 돌릴때 최적값이 바뀌는 경우 발생
- 100개의 `penalty`와 `mixture` 샘플을 임의로 생성

2. MLR 모수추정

▷ workflow 설정

```
mlr_wflow <-
  workflow() %>%
  add_model(mlr_spec) %>%
  add_formula(credit ~ .)
```

```
## == Workflow ====================================================================
## Preprocessor: Formula
## Model: multinom_reg()
##
## -- Preprocessor ----------------------------------------------------------------
## credit ~ .
##
## -- Model -----------------------------------------------------------------------
## Multinomial Regression Model Specification (classification)
##
```

2. MLR 모수추정

▷ 튜닝하기

```
> mlr_wflow <-
+   workflow() %>%
+   add_model(mlr_spec) %>%
+   add_formula(credit ~ .)
> tic()
> tune_result <- mlr_wflow %>%
+   tune_grid(validation_split,
+             grid = mlr_grid,
+             control = control_stack_resamples(),
+             metrics = metric_set(mn_log_loss))
> toc()
1153.3 sec elapsed
```

100개

```
> tic()
> tune_result <- mlr_wflow %>%
+   tune_grid(validation_split,
+             grid = mlr_grid,
+             control = control_stack_resamples(),
+             metrics = metric_set(mn_log_loss))
> toc()
12141.09 sec elapsed
```

1000개

- `metrics` : 최적값 평가 지표는 mean log loss

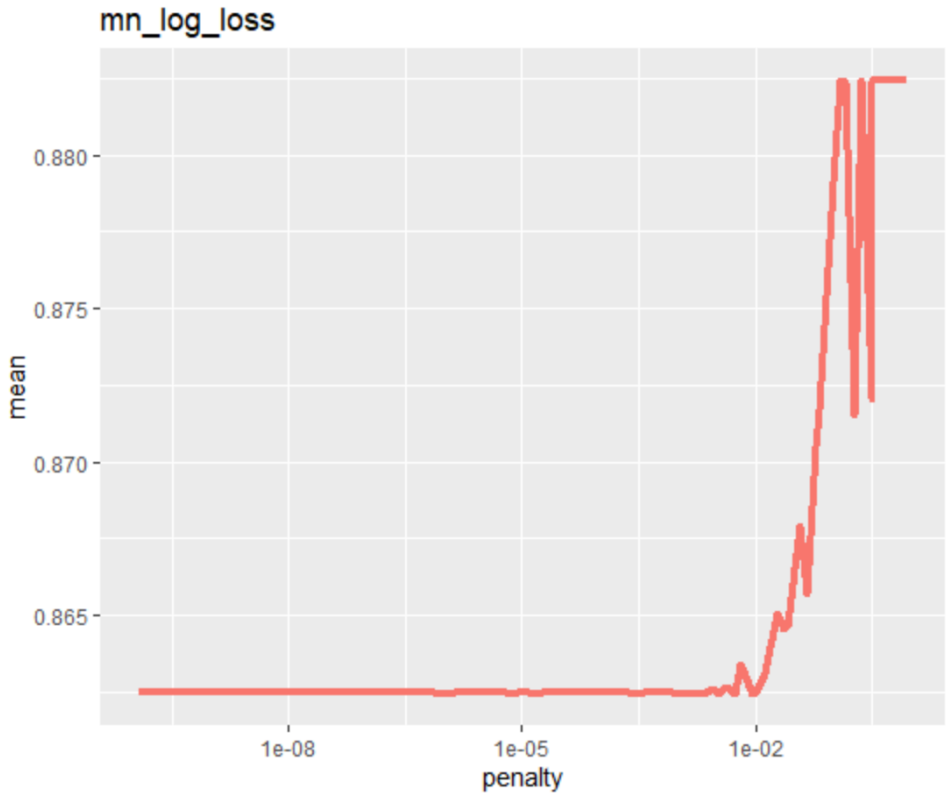
2. MLR 모수추정

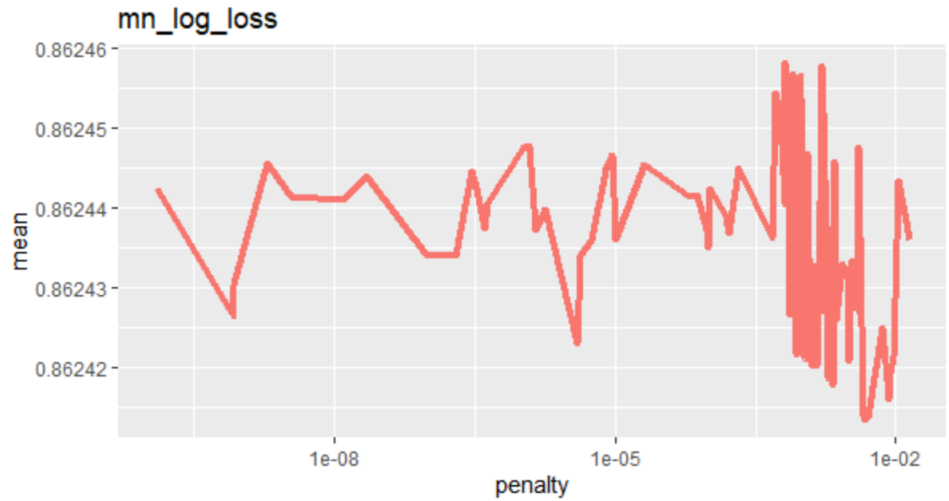
▷ 튜닝결과

```
> tune_result %>%
+   collect_metrics()
# A tibble: 100 x 8
         penalty mixture .metric     .estimator  mean     n std_err .config
           <dbl>   <dbl> <chr>       <chr>      <dbl> <int>   <dbl> <fct>
 1 0.000000813   0.00562 mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model001
 2 0.00000104    0.0109  mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model002
 3 0.000314      0.0203  mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model003
 4 0.0000130     0.0302  mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model004
 5 0.00000759    0.0486  mn_log_loss multiclass 0.862     5 0.00131 Preprocessor1_Model005
 6 0.00928       0.0559  mn_log_loss multiclass 0.862     5 0.00128 Preprocessor1_Model006
 7 0.0000504     0.0610  mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model007
 8 0.00000324    0.0700  mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model008
 9 0.000000000123 0.0885 mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model009
10 0.00549       0.0994  mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model010
# ... with 90 more rows
> tune_result %>%
+   show_best()
# A tibble: 5 x 8
      penalty mixture .metric     .estimator  mean     n std_err .config
        <dbl>   <dbl> <chr>       <chr>      <dbl> <int>   <dbl> <fct>
1 0.00549     0.0994  mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model010
2 0.00928     0.0559  mn_log_loss multiclass 0.862     5 0.00128 Preprocessor1_Model006
3 0.00000104  0.0109  mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model002
4 0.000000813 0.00562 mn_log_loss multiclass 0.862     5 0.00130 Preprocessor1_Model001
5 0.00000759  0.0486  mn_log_loss multiclass 0.862     5 0.00131 Preprocessor1_Model005
```

2. MLR 모수추정

▷ 튜닝결과 plot





2. MLR 모수추정

▷ 튜닝결과 최적값

- 최적값 : `penalty` = 0.00548831, `mixture` = 0.0993676
- `mn_log_tune_best`에 저장

```
> mn_log_tune_best <- tune_result %>% select_best(metric = "mn_log_loss")
> mn_log_tune_best$penalty
[1] 0.00548831
> mn_log_tune_best$mixture
[1] 0.0993676
```

3. MLR vfold 구하기

▷ 최적값을 가지고 학습하기

```
- -
> mlr_spec <- multinom_reg(penalty = 0.00548831,
+                          mixture = 0.0993676) %>%
+   set_engine("glmnet") %>%
+   set_mode("classification")
> mlr_wflow <-
+   workflow() %>%
+   add_model(mlr_spec) %>%
+   add_formula(credit ~ .)
> tic()
> mlr_fit_vfold <-
+   mlr_wflow %>%
+   fit_resamples(credit ~ .,
+                 data = train2,
+                 resamples = validation_split,
+                 metrics = metric_set(mn_log_loss),
+                 control = control_stack_resamples())
경고메시지(들):
The `...` are not used in this function but one or more objects were passed:
 '', 'data'
> toc()
10.28 sec elapsed
```

4. Q&A

▷ 어떤 변수가 신용등급 산정에 주요한 영향을 미치는가?

- 학습한 mlr_fit_vfold 모델의 coefficient값을 확인하는 방법은?

```
mlr_model <- multinom_reg(penalty = 0.00548831,
                          mixture = 0.0993676) %>%
  set_engine("glmnet")

mlr_wflow <-
  workflow() %>%
  add_model(mlr_model) %>%
  add_formula(credit ~ .)

tic()
mlr_fit <-
  mlr_wflow %>%
  fit(data = train2)
toc()

options(max.print = 18)
mlr_fit %>%
  tidy() %>%
  filter(estimate > 0.0001)
```

```
> options(max.print = 18)
> mlr_fit %>%
+   tidy() %>%
+   filter(estimate > 0.0001)
# A tibble: 13 x 4
   class term         estimate penalty
   <chr> <chr>           <dbl>   <dbl>
 1 1     reality       0.105   0.00549
 2 1     child_num     0.00604 0.00549
 3 1     income_type   0.0334  0.00549
 4 1     family_type   0.0579  0.00549
 5 1     house_type    0.0177  0.00549
 6 1     occyp_type    0.00114 0.00549
 7 2     (Intercept)   0.901   0.00549
 8 2     gender        0.00172 0.00549
 9 2     car           0.00248 0.00549
10 2     edu_type      0.0196  0.00549
11 2     family_size   0.0525  0.00549
12 2     begin_month   0.0159  0.00549
13 2     yrs_birth     0.00458 0.00549
```

4. Q&A

▷ grid 실행때마다 값이 리뉴얼되는 문제, 항상 setseed()와 같이 돌려야 하는지?

- size = 100에서는 tune_best가 변하는 경우 발생 (최적 패널티는 변하는데 mean은 0.862로 변화없음) ☞ setseed(2021)과 grid를 같이 실행
- size를 1000개로 늘리면 seed고정 없이도 튜닝이 안정적일 수는 있지만, 튠 시간이 대폭 증가

참고 및 출처

▶ 참고사이트

- https://blog.naver.com/hobbang143/221735605346
- https://blog.naver.com/pmw9440/222001218822
- https://blog.naver.com/jjy0501/221640825506
- https://lmlcr.gagolewski.com/shallow-and-deep-neural-networks.html

▶ 그림 출처

- 그림1 : https://www.cntk.ai/pythondocs/CNTK_103B_MNIST_LogisticRegression.html
- 그림2 : https://lmlcr.gagolewski.com/shallow-and-deep-neural-networks.html