Multinomial Regression

Jim Harner

1/12/2021

`sparklyr` requires a `dplyr` compatible back-end to Spark.

```
library(dplyr, warn.conflicts = FALSE)

library(sparklyr)
# master <- "spark://master:7077"
master <- "local"
sc <- spark_connect(master = master)
```

6.6 Multinomial Regression

We now look at the classification problem in which there are $k > 2$ groups.

6.6.1 Basics

Multinomial regression is a relatively simple extension of logistic regression. We now have $k-1$ logit transformations expressed linearly in terms of the X's. The last group is used as the denominator in these logits.

We can then compute $P(G = l \,|\, X = x)$, for $l = 1, 2, \ldots, k-1$. $P(G = k \,|\, X = x)$ is obtained by subtraction.

6.6.2 Multinomial Models

The diabetes data from the Reaven and Miller study has diabetic-related measurements on 145 patients:
* RelWeight - relative weight
* GluFast - blood sugar level prior to the glucose tolerance test
* GluTest - average blood sugar level during the test
* InsTest - average insulin level during the test
* SSPG - a measure of how glucose and insulin interact
* CClass - clinical diagnosis (3=Normal, 2=Chemical Diabetic, 1=Overt Diabetic)
The dataset is small, but it illustrates binomial models using two levels (by combining Overt and Chemical Diabetics) or multinomial models using three levels.

The `diabetes.csv` file is read into an R data frame and the `CClass` variable is converted from an `int` to a factor with `chr` values.

```
diabetes_df <- read.csv("diabetes.csv", header = TRUE) %>%
  mutate(GluDiff = GluTest - GluFast) %>%
  mutate(CClass = factor(CClass, labels = c("o", "c", "n")))
diabetes_sdf <- copy_to(sc, diabetes_df, "diabetes_sdf")
head(diabetes_sdf)
```

```
## # Source: spark<?> [?? x 7]
##   RelWeight GluFast GluTest InsTest  SSPG CClass GluDiff
##       <dbl>   <int>   <int>   <int> <int> <chr>    <int>
```

```
## 1      0.81      80     356     124    55 n          276
## 2      0.95      97     289     117    76 n          192
## 3      0.94     105     319     143   105 n          214
## 4      1.04      90     356     199   108 n          266
## 5      1         90     323     240   143 n          233
## 6      0.76      86     381     157   165 n          295
```

It would be possible to binarize `CClass` by combining `o` and `c` to `d` using:

```
mutate(CClass = recode(CClass, "o" = "d", "c" = "d", "n" = "n"))
```

but we will keep 3 groups.

The `ml_logistic_regression` function accommodates $k > 2$.

```
diabetes_logistic_fit <- diabetes_sdf %>%
  ml_logistic_regression(CClass ~ GluDiff)
diabetes_logistic_fit
```

```
## Formula: CClass ~ GluDiff
##
## Coefficients:
##   (Intercept)     GluDiff
## n    57.71914 -0.17196850
## c   -21.56761  0.07146432
## o   -36.15153  0.10050418
```

The output gives the coefficient estimates for each of the three groups. The normal group appears to be quite different than the chemical and overt disbetics.

```
diabetes_logistic_predict <- ml_predict(diabetes_logistic_fit)
diabetes_logistic_predict
```

```
## # Source: spark<?> [?? x 16]
##    RelWeight GluFast GluTest InsTest  SSPG CClass GluDiff features label
##        <dbl>   <int>   <int>   <int> <int> <chr>    <int> <list>   <dbl>
##  1      0.81      80     356     124    55 n          276 <dbl [1~     0
##  2      0.95      97     289     117    76 n          192 <dbl [1~     0
##  3      0.94     105     319     143   105 n          214 <dbl [1~     0
##  4      1.04      90     356     199   108 n          266 <dbl [1~     0
##  5      1         90     323     240   143 n          233 <dbl [1~     0
##  6      0.76      86     381     157   165 n          295 <dbl [1~     0
##  7      0.91     100     350     221   119 n          250 <dbl [1~     0
##  8      1.1       85     301     186   105 n          216 <dbl [1~     0
##  9      0.99      97     379     142    98 n          282 <dbl [1~     0
## 10      0.78      97     296     131    94 n          199 <dbl [1~     0
## # ... with more rows, and 7 more variables: rawPrediction <list>,
## #   probability <list>, prediction <dbl>, predicted_label <chr>,
## #   probability_n <dbl>, probability_c <dbl>, probability_o <dbl>
```

The f1 performance measure, which combines precision and recall is:

```
ml_multiclass_classification_evaluator(diabetes_logistic_predict, label_col = "CClass",
                           prediction_col = "prediction", metric_name = "f1")
```

whereas the accuracy is:

```
ml_multiclass_classification_evaluator(diabetes_logistic_predict, label_col = "CClass",
                       prediction_col = "prediction", metric_name = "accuracy")
```

```
spark_disconnect(sc)
```