Logistic regression

Maruša Oražem[1]

[1] *mo3757@student.uni-lj.si, 63200439*

Part 1 - Models

We had to implement multinomial logistic regression and ordinal regression as described in lecture notes. We had to derive the log-likelihood fot both of the models and implement an algorithm using maximum likelihood estimation that can also make predictions.

Multinomial logistic regression

Multinomial logistic regression is a method that generalizes logistic regression to multi class problems. Let's say we were given a data set with N numerical independent variables x_i and dependent ordinal target y_i with k different target variables. We are assuming that there is an underlying real value u_{ij} that we can make it dependent on the term $u_{ij} = \beta_j^T x_i$, where β_j is the vector of coefficients determing the j-th class. From that we get the vector of probabilities $p_i = softmax(u_{1i}, u_{2i}, ..., u_{k-1,i}, 0)$. We can set the last value to 0 because this model is identifiable. Thats how we are effectively choosing the reference object. We got:

$$y_i|x_i, \beta \quad \sim \quad Categorical(softmax(p_i)), \tag{1}$$

where β is $(k-1) \times m$ vector of coefficient (k is number of classes and m is the number of attributes). Softmax function is defined as:

$$softmax(x_1, x_2, ..., x_m) = \left[\frac{e^{x_1}}{\sum_{i=1}^{m} e^{x_i}}, \frac{e^{x_2}}{\sum_{i=1}^{m} e^{x_i}}, ..., \frac{e^{x_m}}{\sum_{i=1}^{m} e^{x_i}} \right]^T. \tag{2}$$

From here we can derive the likelihood function.

$$L(\beta; y, x) = \prod_{i=1}^{N} \prod_{j=1}^{k} p_{ij}^{\mathbb{1}(y_i == j)}, \tag{3}$$

where p_{ij} is the probability for j-th class in the softmax vector of probabilities for i-th observation. From that we get the log-likelihood:

$$l(\beta; y, x) = \sum_{i=1}^{N} \sum_{j=1}^{k} \mathbb{1}(y_i == j) \ln(p_{ij}). \tag{4}$$

We now look for the maximum of log-likelihood function. For that we used *fmin_l_bfgs_b* function from *sklearn.optimize*. We have to put $-$ sign in front of log-likelihood because function is looking for minimum. From there we have fitted our model and we now have determined the values for the β matrix. For predicting on this model, for x_i we first calculate the $u_{ij} = \beta_j^T x_i$ values for every class j. We put the vector into softmax function and the prediction is the class j, where the probability is the highest.

Ordinal logistic regression

Ordinal data are categorical data where there is a natural ordering to the categories. We are assuming that there is an underlying real value u_i that we can make it dependent on the linear term $\beta^T x_i$. Our mission is to determine the values in β. We were looking for log-likelihood function that would fit the model using maximum likelihood estimation. We have again used *fmin_l_bfgs_b* function from *sklearn.optimize* to find the maximum values for maximum likelihood estimation.
Let's say we were given a data set with N numerical independent variables x_i and dependent ordinal target y_i with k different target variables. We denote it as:

$$y_i|x_i, \beta, t \quad \sim \quad Categorical(p_i), \tag{5}$$

where β is vector of coefficient and t is the vector of tresholds that determine each class. Probability vector p_i stores probabilities for each class. In vector p_i, the j-th index tells us the probability that x_i belongs to class j. We calculate it with next equation:

$$p_i(j) = F(t_j - \beta^T x_i) - F(t_{j-1} - \beta^T x_i), \tag{6}$$

where t_j is the j-th element in vector t and F is the link function in our case, the inverse logit function.
From that we derive the likelihood function:

$$L(\beta, t; y, x) = \prod_{i=1}^{N} \prod_{j=1}^{k} p_i(j)^{\mathbb{1}(y_i == j)}. \tag{7}$$

From that we get the log-likelihood:

$$l(\beta, t; y, x) = \sum_{i=1}^{N} \sum_{j=1}^{k} \mathbb{1}(y_i == j) \ln(p_i(j)). \tag{8}$$

We put the derived likelihood (we again change the sign, because we are looking for maximum), with initial guess for

β and t, in *fmin_l_bfgs_b* and with that we got the model we wanted. We now know the values of β and t. Now to predict on that model, we first calculate $u_i = \beta^T x_i$. After that, we calculate all the probabilities, so the $p_i(j) = F(t_j - \beta^T x_i) - F(t_{j-1} - \beta^T x_i)$ for every class j. The prediction is the index of where the maximum probability is of all the calculated ones.

Part 2 - Intermezzo

In this part, we are going to show an example where an ordinal regression performs better than multinomial logistic regression. We know that in most cases, result would be the other way around, because multinomial model has more parameters and can therefore perform better. But this is also exactly the problem of the model. If the training data is too small, parameters can not be set efficiently and model can't learn from that small dataset. We created such dataset and is stored in csv file *multinomial_bad_ordinal_good_train.csv*. Data consist of 6 different classes and it is created from 3 different functions. When creating the data, we also added some noise to it. Data set has only 6 examples. In csv file called *multinomial_bad_ordinal_good_test.csv*, we can find our test data set, that consist of 1000 instances. In table 1 we can see how models performes. As we discussed above, the ordinal model now performs better.

	Accuracy	Log loss
Multinomial	43.7%	15.68
Ordinal	85.3%	0.56

Table 1. Table shows how multinomial and logistic models perform on our created dataset with small train set.

Part 3 - Application

We have a dataset that contains information about 250 students responses to the question *Overall, how would you rate this course?* for some Master's course (answers are 5-level ordinal ranging from very poor to very good). We are interested in the relationship between this variable and other available information, which includes age, sex, year of study (1st or 2nd) and grades (% scored on the exam) for the course in question. We have fitted both dataset to multinomial and ordinal logistics models. The results we got are in the table 2. We splited the data into first 80% rows for train test and the last 20% of rows for test set. Some attributes in the dataset didn't have numerical values and because our models are made for numerical values, we needed to prepare the data. The values for the *sex* column were changes to 0 for male and 1 for female. The dependant values are 'very poor', 'poor', 'average', 'good', 'very good', which were changed to 0-4 accordingly. We can see results in table 2.

Cross validation for estimating the log loss

We have estimated the log-loss for every model using k-fold cross validation. As a baseline for comparison we used the naive model that always predict the same values. Log -loss function is defined as

	Accuracy on test test
Multinomial logistic regression	0.58
Ordinal logistic regression	0.62

Table 2. Table shows us the accuracies on the test set.

	95% CI
Multinomial logistic regression	(1.138, 1.456)
Ordinal logistic regression	(1.066, 1.295)
Naive regressor	(1.255, 1.429)

Table 3. Table shows us the estimation of log-loss with 10-fold cross validation for different models.

$$L_{log}(Y,P) = -\ln Pr(Y|P) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{k} \mathbb{1}(y_i == j)\ln(p_{ij}), \tag{9}$$

which is just the minus sum of all the log probabilities of the instance being classified into the correct class. After performing the l0-fold cross validation, we got the results in table 3.

Regression coefficients

Regression coefficients, like any estimate, contain uncertainty. We have reinterpret the coefficients with this extra information. In practice, we should never interpret uncertain quantities without extra information about the uncertainty. We have performed bootstrap to determine confidence intervals. We can see results on figure 1. We can see that the grade obtained in that course has a big value on students rating, while other courses, as expected, do not. First column tells us the average response from students. We can see also see the positive coefficient for age and negative for year. We could interpret that the students that took the class twice or more (they first failed the class), will first vote negative and later positive.

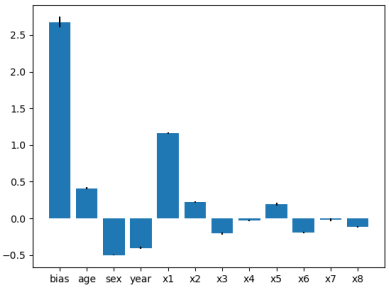

Figure 1. Figure shows us the importance of each coefficient.