INFO370 Problem Set: Logistic Regression, Prediction

May 14, 2022

Introduction

This PS has the following goals:

- learn to use and interpret logistic regression results
- learn to handle categorical variables
- learn to predict the outcomes, and compare with the actual data.
- learn to use *sklearn* library
- understand confusion matrix and related concepts

1 Heart attack: inferential modeling (50pt)

In this question, we will construct a simple logistic regression model to predict the probability of a person having a heart attack. The dataset comes from Kaggle `www.kaggle.com/rashikrahmanpritom/heart-attack-analysis-prediction-dataset`, which contains health information of each person (predictors) and whether or not the person had a heart attack before (outcome variable). You can download the data *heart.csv* from Canvas. The variables are (as described on the webpage):

age age of the patient

sex sex of the patient (1 = male; 0 = female)

cp chest Pain type chest pain type (1: typical angina, 2: atypical angina, 3: non-anginal pain, 4: asymptomatic

trtbps resting blood pressure (in mm Hg)

chol cholestoral in mg/dl fetched via BMI sensor

fbs (fasting blood sugar > 120 mg/dl) ($1 =$ true; $0 =$ false)

restecg resting electrocardiographic results. 0: normal; 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV); 2: showing probable or definite left ventricular hypertrophy by Estes' criteria.

thalachh maximum heart rate achieved

exng exercise induced angina ($1 =$ yes; $0 =$ no)

caa number of major vessels (0-3)

output 0: did not have a heart attack, 1: had a heart attack

slp

oldpeak

thall

Note that the last three variables are not documented. Neither do we know how the data was collected.

Next, the detailed tasks. First, let's do some descriptive statistics.

1. Load data. The data should contain 303 rows, and 11 columns.
2. Do some basic checks. Do we have any missing values? What are the data types? What are ranges of numeric variables, and possible values of categorical variables? What is the percentage of heart attack among these patients?

   Compare the values with documentation and comment what do you see.
3. You probably noticed that all the above variables are coded as numbers. However, not all of these are in fact of numeric (interval, ratio) measure type. Which variables above are inherently non numeric (nominal or ordinal)?

   Hint: 1.1.1 Measures: Possible Mathematical Operations explains the measure types.

Now let's use `smf` to do some modeling and interpret the results.

4. Construct a simple logistic regression with `statsmodel.formula.api` where you diagnose heart attack using all other variables in data. Print the estimated coefficients. You can either use the summary table, or `.params` attribute to get just the values. You'll need these below.

   Remember to ensure that nominal variables are not entered as numeric!

   Hint: consult Python notes 10.2.1 for general usage of *statsmodels*.

5. Compute and display marginal effects.

   Hint: ME for *cp* value "1" should be 0.1297.

6. Interpret the following effects:

   (a) *age*. Is it statistically significant (at significance level of 0.05)?
   (b) *sex*. Is it statistically significant?
   (c) *cp*. Which types of chest pain are statistically significantly associated with heart attack? Which one is the reference category?
   (d) What are the variables that are associated with lower chance of having a heart attack, and are also statistically significant? Do they intuitively make sense? Why?
   Hint: one of these is *thall*.

You may have heard that older people suffer more often from heart attack. But if you did the above correctly then you noticed a small negative (though not significant) age effect. Why do we get such a result? Is it because if someone has symptoms like chest pain and *caa* (whatever it is...) then heart attack is more likely in case of younger people? Let's analyze it separately!

7. Create a variable for 3-4 age groups (pick meaningful groups, but something like 0-40, 40-55, 55-66, 65-100 makes sense). Now re-estimate the model of heart attack, but now including *only* age groups in the model, not other symptoms. Display the marginal effects. Which age group is the reference category?

   Hint: consult Python notes 6.2.1 discusses how to create such categories.

8. What do you find: is heart attack probability falling or growing in age? Why do you think you find such counter-intuitive result?

   Hint: you may consult Lecture Notes, Ch 1.6.1 *Statistical Fallacies*, in particular the the section "Incomplete/missing data" and Exercise 1.15.

9. Finally, although the focus here has been on inference, let's also do some predictions here. Use your full model (the one you did in 1.4).

   (a) Predict the heart attack probability for everyone in data. Print out the predicted heart attack probability for the first 10 observations.

Hint: python notes 12.2.3 discusses how to predict logistic regression results with *statsmodels*.

(b) Predict the predicted outcome (attack/no attack) for everyone in data. Print out the first 10 cases.

(c) Predict whether a 60 years old man has heart attack if he complains about chest pain type 2; has *restecg* 2; *caa* 2; `trtbps` 125; and *chol* 250. You can take the other values equal to the sample mean.

(d) Display confusion matrix and compute accuracy.

---

2 Heart attack: predictive modeling (50pt)

Now let's construct the same model using sklearn package. Remember that sklearn package requires the matrix of predictor values (X) to be separated from the vector of outcome variable ($\boldsymbol{y}$). The predictor values must be a matrix, not a vector. See Python Notes 11.2.2 for how to use *sklearn* for logistic regression.

1. Now replicate the same model as above in question 1.4 using sklearn. As sklearn does logistic regression slightly differently than statsmodels, we set some parameters to do it in a more similar fashion. Define the model as

   ```
   LogisticRegression(max_iter=5000, penalty="none")
   ```

   not just as

   ```
   LogisticRegression()
   ```

   Print out the coefficients and the intercept. You should get fairly similar results as above, but because the somewhat different approach behind the scenes, but because of the minor differences in the algorithms, the coefficients will not be exactly the same.

   Note: you should get similar *coefficients*, but coefficients and marginal effects differ. They may also be in a different order than in the *statsmodels*' output.

Now, after using both packages, you should have a clearer sense that smf package is good for inferences becuase it provides information about p-values,

z-values, CI etc., which can be used to determine statistical significance. Sklean package does not provide such information. However, the advantage of sklearn is that it is easier to construct models and make prediction with sklearn, especially when you are comparing multiple models.

Now it is time to use sklearn for predictive modeling. Use the model from 2.1 above (the same X and $\boldsymbol{y}$ you did above).

2. Predict the *probability* of having a heart attack $P(output = 1|\boldsymbol{x})$ for everyone in data. Print out the first 10 probabilities. The results should be very similar to what you got in 1.9 above.

   Note: print *only* probability for heart attack, not probability of non-heart attack!

   Hint: Python Notes 12.2.4 discusses predicting logistic regression results with *sklearn*.

3. Predict the *label* (*outcome*)—that is, whether someone has heart attack or not, instead of predicting the probability. Print out the first 10 labels.

4. You can predict labels in two ways: you can use `.predict` method, or alternatively, you can compute probabilities and find where they are larger than 0.5.

   Show that both methods give you the same results.

   Hint: you can use your probability/label predictions from above, check also out functions `np.all`, `np.any`.

5. Display the confusion matrix. Do you get the same matrix as in 1.9?

6. Compute and display precision and recall. What do you think, which measure–accuracy, precision, recall, or F-score should we try to improve in order to make this model more applicable in medicine? Explain!

7. Now imagine we create another very simple model ("naive model") that predicts everyone the same result (attack or no attack), whichever category is more common in data (the majority category).

   How would the confusion matrix of this model look like? What are the corresponding accuracy, precision and recall? Explain!

   Note: you should not fit any model here, you are able to compute these all these values manually with just a calculator!

Finally...

... how much time did you spend on this PS?