Graphical Methods for Assessing Logistic Regression Models

(on Haberman's survival data)

Yuting Wang(yw3167)   Zhansen Shen(zs2390)

Introduction

Graphical displays are always helpful in predicting and detecting features in medical area. For linear regression, it is very easy to fit a model to data. For logistic regression, binary data and discrete points can be relatively hard to display features and make a prediction. We first tried three methods which are used in Graphical Methods for assessing Logistic Regression Models [1]. These methods are extensions to linear models that work on logistic regressions including local mean deviance plots, empirical probability plots and partial residual plots. Local mean deviance plots can be used in general to measure how fit a model is. Empirical probability plots can help to identify some outliers and regions with lack of fit. Partial residual plots can reach out to reasons that lead to lack of fit and provide some guidelines to a better fit. In addition to the three methods proposed in the paper, we tried Ridge and lasso to remove several predictors in order to improve our graphical results. We also tested the model from a statistical perspective, using likelihood ratio test and pseudo R squared test.

Dataset and paper

At university of Chicago's Billings Hospital, a study related to patients' survival conditions after breast cancer surgery between 1958 and 1970 has been conducted. After that, a dataset consisting of 306 observations has been formed. Each observation includes threes predictors and their relative labels. Giving a deep research in this dataset, it is good for predicting the survival rate of this surgery and providing references for both hospitals and patients.

Haberman's survival dataset has three predictors: x_{i1} is the age of patient i at time of operation, x_{i2} is the year of operation for patient i minus 1990; x_{i3} is the number of positive axillary nodes related to patient i. The label y_i equals to 1 if patients survived 5 years or longer and 0 otherwise.

The paper we are using is Graphical Models for Assessing Logistic Regression Models by Landwehr, J. M., Pregibon, D., and Shoemaker, A. C. (1984).

Algorithms presented in the paper

Local mean deviance

The first method we are going to introduce is the local mean deviance plot. To give an overview, the plot is first about deviance, a commonly used term that looks like log-likelihood but slightly different. The plot is about mean deviance, which takes into account some statistics and find out a mean of them. The plot is also about local deviance. We separate data into some groups and calculate the deviance from them.

To illustrate the algorithm, we first show what a global deviance looks like when evaluating logistic regression. The formula is given below:

$$\mathrm{D} = -2\sum_{i=1}^{N}(y_i \log(p_i) + (1-y_i)\log(1-p_i))$$

Notice that this is exactly negative 2 times the log likelihood function. It provides a global deviation between the observations and the fitted probabilities. The problem is that there is no exact distribution of global deviance. You can not get a p value from it and test if it rejects the null hypothesis or something similar. Another issue is that this number is somehow too general and global for the highly discrete logistic regression. It is just not thorough to conclude something about the whole model with one statistic that has no references. The paper suggests that this global deviance can be separated into two parts: pure-error component and lack-of-fit component. The first one happens with measurements and is only related to data itself, while the second one is determined by the chosen model. We then give a general algorithm on how to plot the values of local mean deviance:

1. Partition the data into some smaller groups by applying, for example, hierarchical clustering. This step is for the local measurements. We will discuss its trade-off later.
2. Fit a logistic regression model to each cluster of data using the parameters used for the whole model.
3. Find out the deviance contribution of each observation, which is the component being

summed in the global variance.

4. Find the local deviance for each cluster by summing up all deviance inside the cluster.

5. Order the groups based on their inhomogeneity. Clusters that has tighter distribution of data should be put up front. In case of hierarchical clustering, the tightness can be found by ordering the height of each cluster.

6. For t = 1:K, where K is total number of clusters, find out the sum of local deviance divided by degree of freedom for tightest t groups. This gives us a list of values in which the first one comes from the tightest cluster, and the last one comes from all clusters.

7. Plot the result in step 6 against the degrees of freedom.

We will present the clustering results and the final local mean deviance plot.

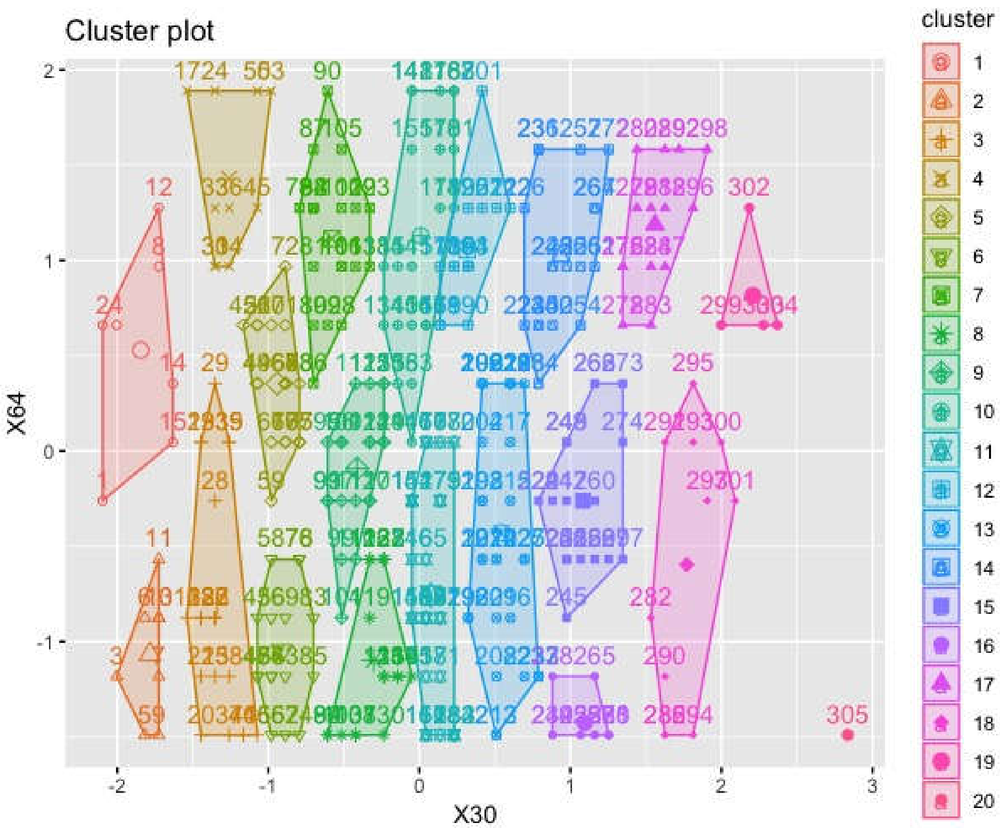


Figure 1. Hierarchical clustering results

This is a separation result based on hierarchical clustering. Local deviance is then calculated and plotted as follows:

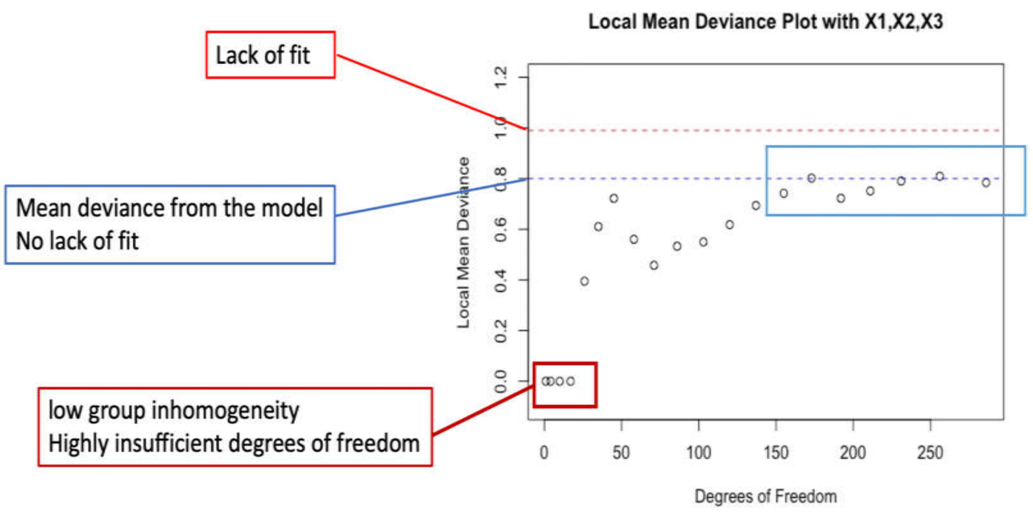


Figure 2. Local mean deviance plot with input predictors x1, x2 and x3.

For a correct model we would expect that at high degrees of freedom, the local mean deviance will be close to the global deviance. This can be interpreted as the global variability is the same as the local variability. If the line of global deviance passes through the points with high degrees of freedom, we conclude that the model is a good fit. Otherwise, we may conclude that the model is a bad fit. In the above figure, the red line is the mean deviance from the model fitting x1, x2 and x3, which does not match the points giving the local mean deviance estimates, indicating lack of fit. The blue line is the mean deviance which passes points with large degrees of freedom, so no lack of fit is indicated.

When interpreting step 1 in the algorithm, we need to focus on the trade-off between insufficient degrees of freedom and group inhomogeneity. Group inhomogeneity represents how dissimilar it is between two groups. If points in a cluster simply replicate or almost are the same, then their group inhomogeneity is very small. We would like to calculate local mean deviance of highly homogeneity, which results in a very small degree of freedom. Otherwise, when group number increases, the degree of freedom gains, while the similarity between groups become smaller. At the same time, it is unsure that our prediction can correctly show local variability.

Empirical probability

Local mean deviance plot is powerful when detecting goodness of fit of a model. However, the fitted model eventually depends on the properties of data. In linear regression, we have seen many cases in which the data points have high variance or are not closely enough to the fitted straight line. Data points in some regions are close to the fitted line while those in other regions are not. If there are few points that show lack of fit, they might be outliers. However, if a considerable number of points lie out of the linear area, the model shows a lack of fit.

The same thing happens to logistic regression as well, even though the regression output has only discrete values (classes). However, there is some difference in the analysis. One major issue is that the predicted probabilities for the data points are values between classes (in our case it is either 0 or 1). Points might lie above or below the fitted line if we actually plot the classes against the predicted probabilities. It is unlikely that a large percentage of the points will lie close to the graph.

We introduce another method called empirical probability to analyze such outliers and lack of fit. Rather than analyzing deviance, we focus on the residuals of logistic regression instead. The basic idea is to calculate the residuals using the fitted probability and the original labels of the data points. We then generate new data samples using binomial distribution given the fitted probabilities. After that, we fit another logistic regression model to the new samples and calculate the residuals again.

After some iterations we should have a set of vectors that contain residuals of each iteration. Intuitively speaking, if the first model (the fitted model of the original data) is exactly the one we are looking for (has the best fit), the following iterations should generate residuals that are centered at the first residuals. We can then plot the original residuals against a statistic of the generated residuals, say, median residuals for each data point in all iterations. We expect this plot to approximate a straight line with unit slope, passing the origin. The detailed algorithm is given below:

1. Calculate residuals based on original labels and fitted probabilities $r_i = y_i - p_i$ for i from 1 to N
2. Sort the residuals in ascending order. This helps make the plot look smooth.
3. Generate data from binomial distribution under the fitted probabilities in step 1.
4. Fit a logistic regression model to the data in step 3 and compute residuals.
5. Repeat steps 2-4 individually.

6. Calculate a typical statistic (median, mean etc.) of residuals for each point over the iterations.
7. Plot the statistic against the residuals calculated in step 1
8. Plot confidence intervals by taking upper and lower boundaries of the iterated residuals at each point and connect those quantiles for each data point.

This is the empirical probability plot. For the Haberman survival dataset, we apply two different models and use this plot to test the accuracy of the model. The first one is a simple model with linear combination of all three predictors. The second one is a complex model involving products and logs of multiple predictors. The original paper results show that the first one shows lack of fit while the second one does not. The results are given below.

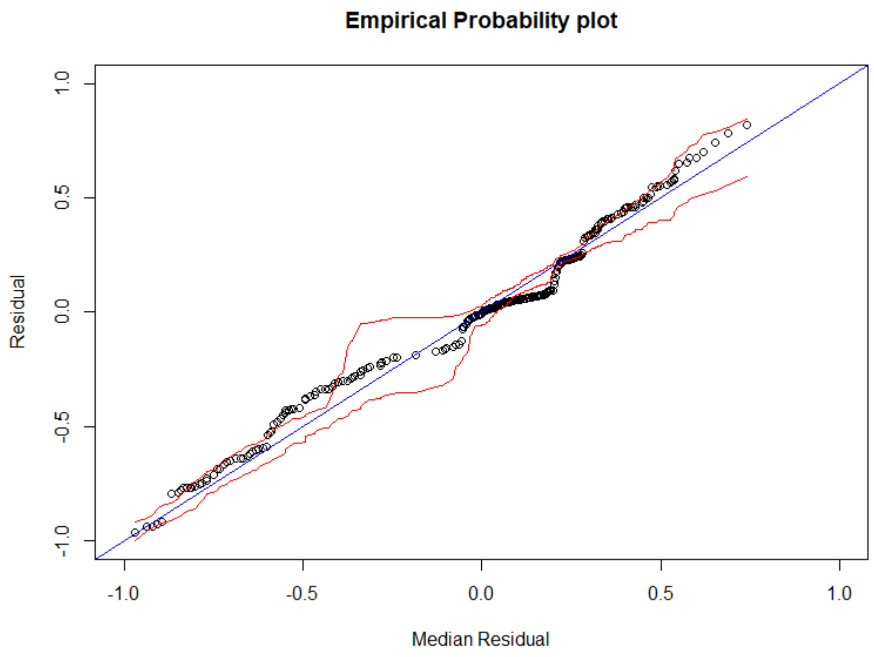


Figure 3: Empirical probability plot for model with linear predictors

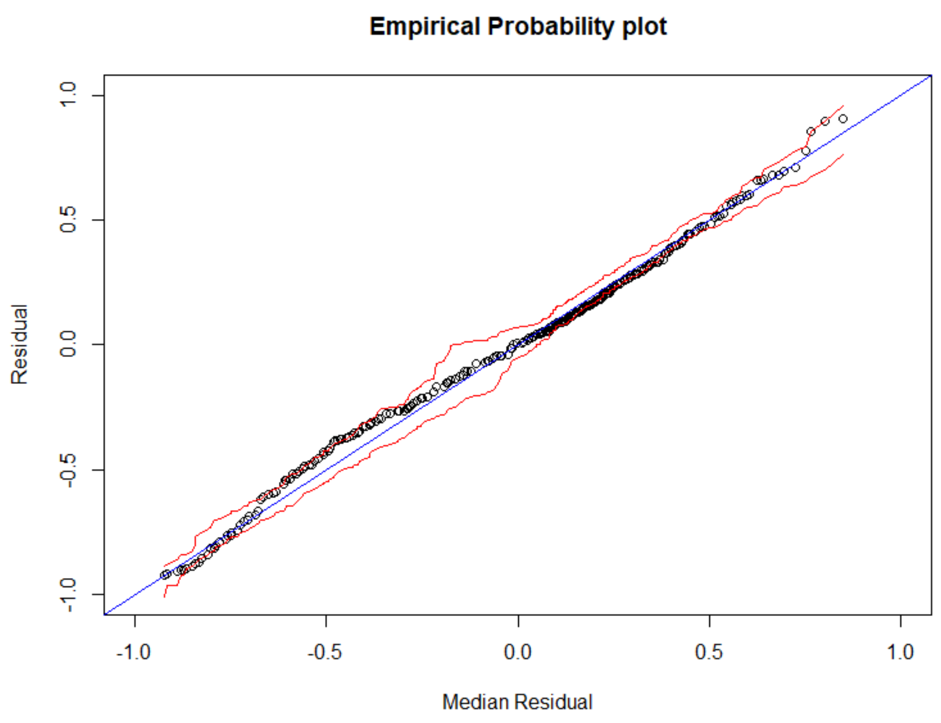


Figure 4: Empirical probability plot for model with complex predictors

We start from the first figure. A group of points lie outside the region at x=-0.5 and x=0.2, indicating lack of fit in some region of the model. If there is only one point then it will be classified as an outlier and has no effect in determining the model. However, as more points lie outside the confidence band, we should be aware that this model is not a good fit. For the second model, the points all lie in the confidence band, which shows no lack of fit.

There are some side comments that can help analyze this graph. First, the empirical probability plot should always lie within the region with boundaries (-1, 1) on each axis. This is because the fitted probability lies between (0, 1). Points with label 1 are plotted on the upper right side of the graph while those points with label 0 are plotted on the lower left side. The lines that depict confidence intervals should converge to (-1, -1) and (1, 1). Second, there are not many characteristics that the graph shows. The confidence bands need not be symmetric, either about the line y=x or around the origin. The width of the confidence band is also not a big consideration. In regions with fewer points, the sampling process will show some inhomogeneity, which results in a wider confidence band, like in region (-0.5, 0) in both graphs. The distance from each point to the y=x axis does not matter, as long as the point lies in the confidence band. We expect points that do not fit in the model to lie outside the confidence band. Therefore, it is not surprising when we find a couple of data points on the boundary. After all, the generative process has some randomness as well for small datasets. Third, this is just a graphical view that shows intuitively the goodness of fit for the model. There are no quantitative measures that depict in detail "how bad or how good" the model is. It just shows whether the model needs modification or improvements, instead of how to improve. The paper suggests that this measure should not be used as a formal test like z-score or t statistic in linear regression. It should be used to find out which data points are not correctly fitted into the model, and take measures to make sure every point (except extreme outliers) is fitted in the model correctly.

Partial residual plot

We have just examined two ways to check goodness of fit: local mean deviance and empirical probability. Both graphs tell us about the appropriateness of a model over data. However they only test models that we have set. We can only tell whether a model shows good of fit or not, instead of getting some advice on how to improve it. Certainly we can go through all the models available in our mind: powers, logs, or even absolute values etc. Yet trying everything is not only inefficient, but also introduces some dummy predictors that have no effect other than over-fitting.

There is a much faster way to reveal what other estimators we are missing: partial residual plots. Let's first try to understand this intuitively: Consider a true model with some function other than linear addition, as well as a predictor that can be any combination of predictors in the data. We need a way to find out the function and the predictor. A naive way to do so is by subtracting the labels and the linear contributors, and then plot the rest against the predictor of our choice. This method, however, does not take into account covariance between predictors. If there surely lies some correlation between them then this approach may not work.

We will now explain some mathematical principles about the partial residual plot. We start from the log-likelihood of logistic regression model with the simplest form:

$$l(\beta) = \sum_{i=1}^{N} (y_i \log p_i + (1 - y_i) \log(1 - p_i))$$

Or, alternatively:

$$l(\beta) = \sum_{i=1}^{N} (y_i x_i^T \beta - \log(1 + \exp(x_i^T \beta)))$$

To find a maximum likelihood estimate with a nonlinear case in beta, we use Newton's method to solve it, which yields

$$\beta(t+1) = \beta(t) + (X^T V(t) X)^{-1} X^T r(t)$$

In this formula, V is the covariance matrix for y, r is the residuals. This equation can be rewritten as follows:

$$\beta(t+1) = (X^T V(t) X)^{-1} X^T V(t) * y(t), \qquad y(t) = X\beta(t) + V^{-1}(t) r(t)$$

As from a weighted linear regression evaluation scheme, the term y can be considered as the observation; $X\beta$ is the fitted value; $V^{-1}r$ is the residual. Notice that when we have y as observation, we take into account that log transforms are invalid at some points. This is just an analogy that links partial residual plots to weighted linear regression. Applying those to logistic regression, we have the logistic observations:

$$y = X\beta + z\gamma + V^{-1}(y - p)$$

which arise from extended model

$$\log it(p) = X\beta + z\gamma$$

And the partial residual is given by

$$r_{par} = V^{-1} r + z\gamma = (y - p)/(p(1-p)) + z\gamma$$

The formula above actually contains a standardization of r that is more natural. We then plot this partial residual against z, the predictors we choose.

We give two examples below. The first one consists of the partial residual against predictor X30, which refers to age of patient in the dataset. The black dots depict the partial residuals. They lie in two separate clusters since a data point has a label of either 0 or 1. The blue dashed line is the original fitted line in the linear model. The pink line is the smoothed mean values of the partial residual. From the shape we can see that the residuals follow a cubic function. Therefore, we can add a cubic function to make the model a better fit.

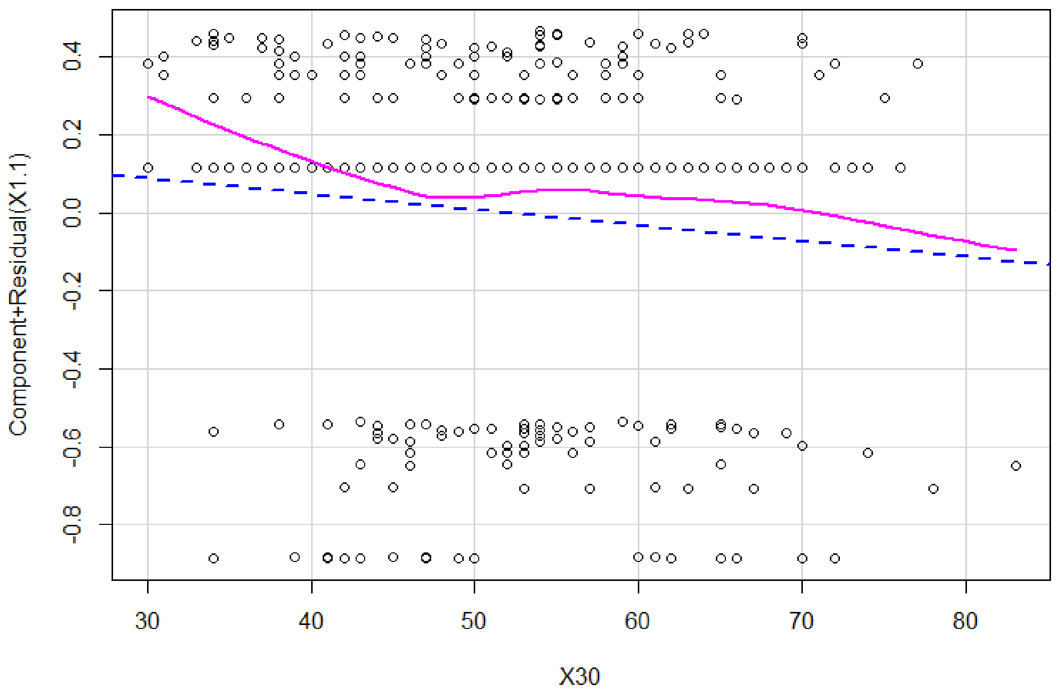


Figure 5. Partial residual plot for the age predictor

We then take a look at another predictor, the product of age and year of operation. This is a suggested model in the paper, and we fetch this result to show what a good fit should look like. The blue line is the fitted line for X30*X64 as a linear predictor. The pink line now matches the blue line, indicating that no more transforms are required for this predictor and it is already a good fit.

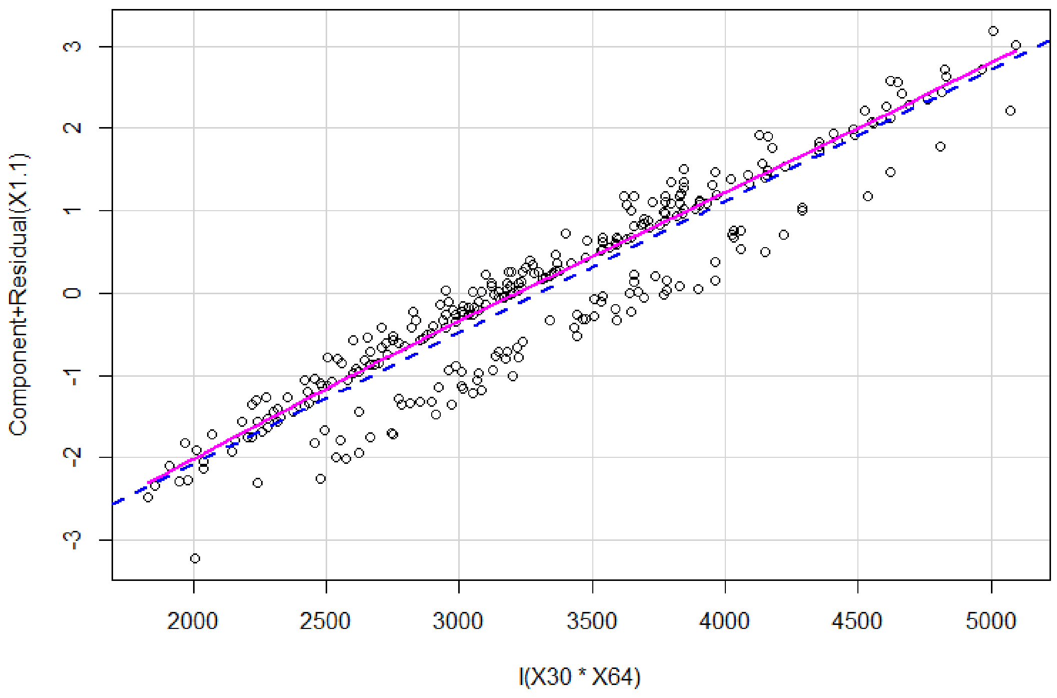


Figure 6. Partial residual plot for age times year of operation

The partial residual plot is a very powerful tool in determining which transformation on a predictor is necessary to obtain a good fit. The shape of the pink curve approximates a transform, by which we can implement a linear coefficient to make the model look better. However, when implementing this algorithm, do check every predictor using the initial model. The same plot under different model fit can be significantly different. We suggest using the original linear model to check everything, and then apply necessary transforms to the model and use other techniques to determine goodness of fit, like the two methods shown earlier.

We should stop here for a moment and see what we have already done. We have implemented three models that has different usage and power: Local mean deviance partitions the deviance into two elements and analyze locally the deviance contributions of each tight cluster. Empirical probability shows regions that the model might not be appropriate enough. Partial residual suggests possible transforms that can be added to the linear basic logistic regression model to improve performance. We summarized the algorithms in the table below:

Methods	Usage	Drawback
Local mean deviance	Check goodness of fit	Trade-off between DOF and group inhomogeneity
Empirical probability	Detect outliers/ Find lack-of-fit regions	Informal and intuitive/region with few points can result in errors
Partial residuals	Provide clues to new estimators	Other estimators may interfere the result

Experiments on other evaluation methods

Ridge and Lasso

We have just examined the functions of the methods proposed in the paper. However, me might go a little bit too far: Linear regression requires a balance between accuracy and number of predictors, and it is also the case for logistic regression. We may use the methods above to achieve an almost perfect fit, but may introduce over-fitting when we test it with incoming data. Although using cross validation may resolve this problem somehow, we need to shrink the model by eliminating some factors that actually does not play a significant role in the model.

We immediately recall the penalty measures when we evaluate the model error: ridge regression and lasso constraints. Ridge penalties tend to make coefficients smaller, while lasso helps to set some of the coefficients to zero, thus shrinking the model with fewer predictors.

Although ridge and lasso are mostly considered in linear regression, they can be deployed in logistic regression as well, with some simple tweaks. Notice that in linear regression we measure the error using least squares, plus the penalty term. In logistic regression we use log likelihood instead. We want to maximize the log likelihood. Our error formula is the negative log likelihood plus the ridge or lasso penalty term. We want to minimize this error.

The error measurement are given here for reference. The negative log likelihood is

$$L = -\log\left(\prod_{i:Y_i=+} p(X_i) \prod_{j:Y_j=-} (1 - p(X_j))\right)$$

And the penalty measurements are

$$L + \lambda \sum \beta_1^2. \quad \text{for ridge regression}$$

$$L + \lambda \sum |\beta_1|. \quad \text{for lasso}$$

Where lambda is chosen arbitrarily. A higher lambda value increase the weight of the penalty, thus resulting in lower model coefficients. A lower lambda value decreases that instead, leading the model to the original one with less errors. So this is a trade-off between accuracy and model complexity. We will give the graphs for ridge and lasso below:

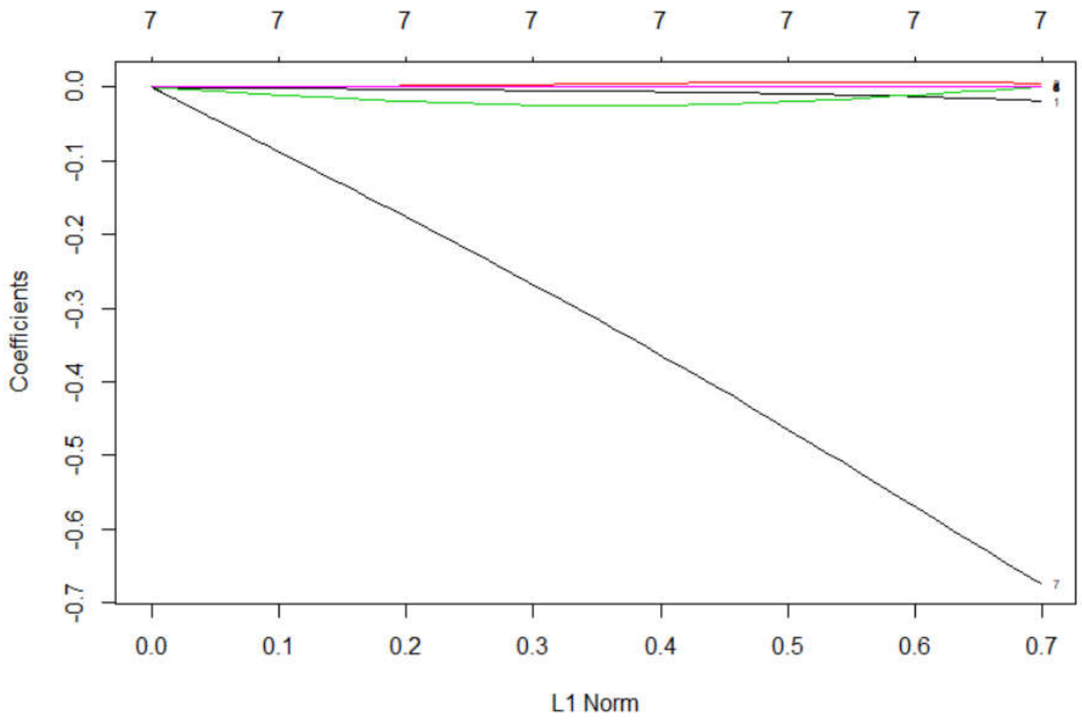


Figure 7. Ridge regression on the full model.

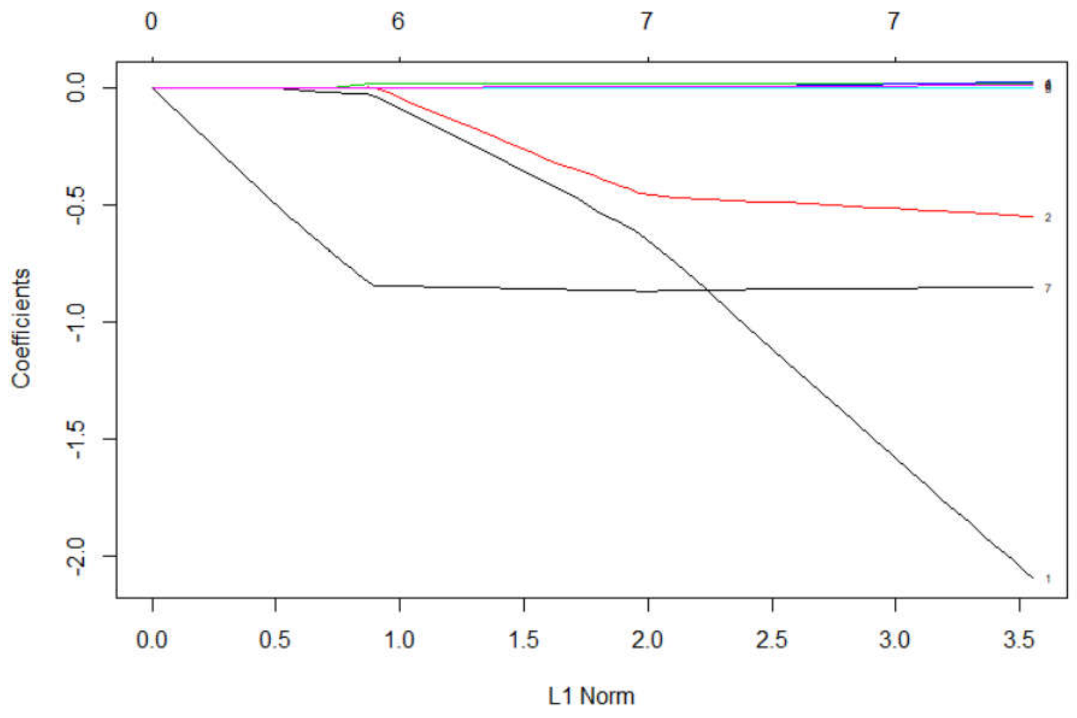


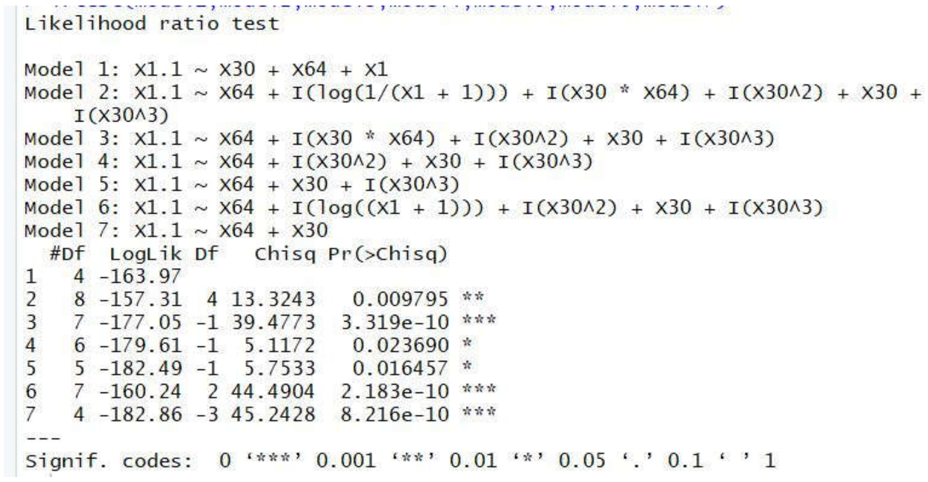
Figure 8. Lasso on the full model

The first graph is for ridge while the second one is for lasso. While ridge does not help us too much, lasso instead points out that some of the predictors are indeed more useful than others, since they have a higher coefficient and are the last ones to disappear as lambda goes up. We can then adjust this lambda based on our needs. We can alternatively find the lambda with the least cross validation error. One drawback of this is that lasso and ridge can fail when extracting features from a high dimensional data.

Likelihood Ratio Test

We have tried four techniques that uses graphs to its fullest. From the position of lines and dots we can tell whether a model is a good fit to the data or not. We will now present something that is purely statistical, to see if they can provide some information about the model that those plots fail to imply. We try two methods here. The first one is likelihood ratio test.

It is called a likelihood ratio test because it compares models. The input of this function is a series of models that we want to examine. The algorithm is to compare the likelihood of the model to the null model. In this way it reveals “how many times” a model is a better fit than other ones. The null model is the same as that in linear regression: it assumes that no predictors are associated with the output. We test some models and show their statistics below:



```
Likelihood ratio test

Model 1: X1.1 ~ X30 + X64 + X1
Model 2: X1.1 ~ X64 + I(log(1/(X1 + 1))) + I(X30 * X64) + I(X30^2) + X30 +
    I(X30^3)
Model 3: X1.1 ~ X64 + I(X30 * X64) + I(X30^2) + X30 + I(X30^3)
Model 4: X1.1 ~ X64 + I(X30^2) + X30 + I(X30^3)
Model 5: X1.1 ~ X64 + X30 + I(X30^3)
Model 6: X1.1 ~ X64 + I(log((X1 + 1))) + I(X30^2) + X30 + I(X30^3)
Model 7: X1.1 ~ X64 + X30
  #Df  LogLik Df   Chisq Pr(>Chisq)
1   4 -163.97
2   8 -157.31  4 13.3243   0.009795 **
3   7 -177.05 -1 39.4773  3.319e-10 ***
4   6 -179.61 -1  5.1172   0.023690 *
5   5 -182.49 -1  5.7533   0.016457 *
6   7 -160.24  2 44.4904  2.183e-10 ***
7   4 -182.86 -3 45.2428  8.216e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figure 9. R screenshot for likelihood ratio test

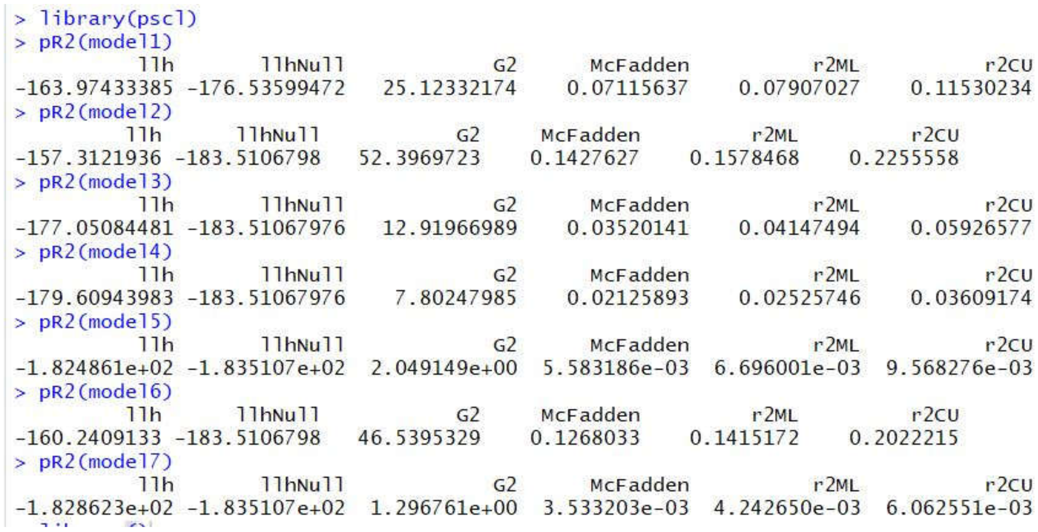
As shown in the figure, the likelihood ratio statistic follows a chi-squared distribution from which we can find out a p-value. This process is the same when we perform F test or T test on a linear regression model. A p value that is less than a certain value, for example models 3,6,7 given above, show a good fit of the model. We can then find what common parameters they have over the other ones. It seems that by adding cubic and log transforms to some predictors can actually improve our results.

Pseudo R squared

Another statistic that can be useful is the McFadden pseudo R squared value. This is a modified R squared from linear regression to fit the logistic regression model. It entertains the ratio between the log likelihood values of the given model and the null model. The formula is given below:

$$R^2_{\text{McFadden}} = 1 - \frac{log(L_c)}{log(L_{\text{null}})}$$

Where the numerator stands for the maximized likelihood value of the given model, and the denominator is the same statistic for null model with intercept only. To make sense, consider we are analyzing a model that has binary outcome. Although the likelihood will usually be greater than that of null model if the entertained model has some predictors, if it actually shows no significant improvement, it is not a good fit. R squared value tends to find out, like likelihood ratio test, how much the model is better than the null model. If the two model actually shows no better improvement in log likelihood, the ratio will be close to 1 and the R squared statistic is expected to be close to 0 and vice versa. We give the test results of the same set of models used in likelihood ratio test:



```
> library(pscl)
> pR2(model1)
          llh        llhNull             G2       McFadden            r2ML            r2CU
-163.97433385 -176.53599472   25.12332174     0.07115637      0.07907027      0.11530234
> pR2(model2)
        llh      llhNull           G2     McFadden          r2ML          r2CU
-157.3121936 -183.5106798   52.3969723    0.1427627     0.1578468     0.2255558
> pR2(model3)
          llh        llhNull             G2       McFadden            r2ML            r2CU
-177.05084481 -183.51067976   12.91966989     0.03520141      0.04147494      0.05926577
> pR2(model4)
          llh        llhNull             G2       McFadden            r2ML            r2CU
-179.60943983 -183.51067976    7.80247985     0.02125893      0.02525746      0.03609174
> pR2(model5)
          llh        llhNull             G2       McFadden            r2ML            r2CU
-1.824861e+02 -1.835107e+02  2.049149e+00   5.583186e-03    6.696001e-03    9.568276e-03
> pR2(model6)
        llh      llhNull           G2     McFadden          r2ML          r2CU
-160.2409133 -183.5106798   46.5395329    0.1268033     0.1415172     0.2022215
> pR2(model7)
          llh        llhNull             G2       McFadden            r2ML            r2CU
-1.828623e+02 -1.835107e+02  1.296761e+00   3.533203e-03    4.242650e-03    6.062551e-03
```

Figure 10. R screenshot for pseudo R squared test

We focus on the McFadden column. The best results we can get are not even close to 1, but we can quickly eliminate some values that are almost zero. The results are not close to 1 for a reason. The values in logistic regression are discrete, and no matter now smooth the fit the residuals will always be enormous. For most research attempts we can not find a strong predictor to reveal the characteristics of the data, so values of 0.1-0.2 are somehow reasonable for logistic regression. There is no exact reference value to that, but we can say that comparatively some models with higher pseudo R squared values are better than those with lower values.

Conclusion

We have examined some methods that can help us find out a better logistic regression model. Apart from the algorithms and implementations given above, we also show that they might be ineffective under certain circumstances. Our suggestion is that no models are necessarily right or wrong. It depends highly on what the data looks like. Some methods for linear regression work well on logistic regression with little tweaks, and some do not. Further work includes using cross validation on models obtained by each specific technique and see which performs best on the data. We could also combine them. For example we could use partial residuals to find out more predictors, and then use ridge and lasso to shrink the models and eliminate predictors, and then use the rest to examine the models' goodness of fit and see what they give us. One thing to be noticed is that when using statistics, we do not have a reference statistic for logistic regression, like what pseudo R squared value is acceptable. From this perspective, we can say for logistic regression, plots may be a more straightforward analysis of what models show good fit. There are other plots as well for us to examine further in other researches.

References

Haberman, S. J. (1976). Generalized Residuals for Log-Linear Models, Proceedings of the 9th International Biometrics Conference, Boston, pp. 104-122.

Landwehr, J. M., Pregibon, D., and Shoemaker, A. C. (1984), Graphical Models for Assessing Logistic Regression Models (with discussion), Journal of the American Statistical Association 79: 61-83.

Le Cessie, Saskia, and Johannes C. Van Houwelingen. "Ridge estimators in logistic regression." *Applied statistics* (1992): 191-201.