The purpose of these notes is to discuss model checking for logistic regression. At the end of
the notes, I provide a sample SAS program for implementing the tools.

In classical linear regression, model checking is carried out by examining the residuals \( e_i = Y_i - \hat{Y}_i \). We do plots of \( e_i \) vs. \( \hat{Y}_i \) and plots of \( e_i \) versus \( X_{ij} \) for each specific covariate in the model (indexed by \( j \)). The latter type of plot can also be done for \( X \)'s that are not yet in the model but are under consideration.

This approach does not carry over directly to case of logistic regression. In logistic regression, the \( Y \)'s are all either 0 or 1, so if we do a plot of \( e_i = Y_i - \hat{p}_i \), with \( \hat{p}_i = \Pr(Y_i = 1|X_i) \), we will get a plot that has a jumpy appearance and therefore not very useful. In order to carry out a meaningful residual analysis for logistic regression, it is necessary to do some averaging first.

Suppose we want to do a plot in the spirit of the linear regression plot of \( e_i \) vs. \( \hat{Y}_i \). We can proceed as follows. First we compute

\[
\hat{p}_i = \Pr(Y_i = 1|X_i) = \frac{e^{\hat{\beta}^T X_i}}{1 + e^{\hat{\beta}^T X_i}}.
\]

The \( \hat{p}_i \) values, being probabilities, lie in the range \([0,1]\). We divide up the range \([0,1]\) into \( K \) intervals, which we will denote by \( \mathcal{I}_k \), \( k = 1, \ldots, K \). We denote by \( n_k \) the number of observations that fall in interval \( k \). There are two main ways to do the split. One way is to split into equal-sized intervals, so that \( \mathcal{I}_k = ((k-1)/K, k/K] \). The other way is to do the split in such a way that the \( n_k \)'s are roughly equal across the intervals. The second approach is often better, especially when the sample size is small to moderate.

Now, let \( C_k \) denote the set of observations \( i \) for which \( \hat{p}_i \in \mathcal{I}_k \), and define the averages

\[
\bar{Y}_k = \frac{1}{n_k} \sum_{i \in C_k} Y_i,
\]

\[
\bar{p}_k = \frac{1}{n_k} \sum_{i \in C_k} \hat{p}_i.
\]
and then

\[ r_k = \frac{\bar{Y}_k - \bar{p}_k}{\sqrt{\bar{p}_k(1 - \bar{p}_k)/n_k}}, \]
\[ \ell_k = \text{logit}(\bar{Y}_k) - \text{logit}(\bar{p}_k), \]

where \( \text{logit}(u) = \log(u/(1 - u)) \). A plot of \( r_k \) vs. \( \bar{p}_k \) can be used for outlier detection; if the fitted model is correct, we expect \( r_k \) to be distributed approximately as \( N(0, 1) \) (if the intervals are narrow enough). A plot of \( \ell_k \) vs. \( \bar{p}_k \) can be used to identify trends pointing to the need to add nonlinear terms to the model. It is useful to apply a nonparametric curve-fitting method such as LOESS to the points \((\bar{p}_k, \ell_k)\) to get an idea of the trend.

A global goodness of fitness test can be carried out by examining

\[ \chi^2 = \sum_{k=1}^{K} r_k^2, \]

which, under the null hypothesis that the fitted model is correct, has an approximate \( \chi^2 \) distribution. This test is discussed on page 72 of the Cox and Snell (1989) book, and also by Hosmer and Lemeshow in their 2000 book *Applied Logistic Regression* and in some previous papers they published. The test is commonly known as the “Hosmer-Lemeshow” test. There are various proposals for what degrees of freedom parameter to use for the chi-square distribution used in the test. Cox and Snell recommend \( K - (p + 1) \) degrees of freedom, with \( p \) being the number of variables in the model. This proposal is workable only if the data set is large enough to allow \( K \) to be taken to be reasonably large. The test can be carried out in SAS PROC LOGISTIC using the LACKFIT option, but PROC LOGISTIC forces \( K = 10 \). In the PROC LOGISTIC implementation, the default degrees of freedom is \( K - 2 \), but the user can specify a different choice. My SAS code below allows arbitrary \( K \).

In terms of the choice of \( K \), there is no firm rule. PROC LOGISTIC, as I said, forces \( K = 10 \). It is reasonable to do the plots for a few choices of \( K \) to see what happens.

The same type of scheme can be used to a plot in the spirit of an \( e_i \) vs. \( X_{ij} \) plot for a given covariate. Here, we break up the range of the variable \( X_j \) into \( K \) intervals \( \mathcal{I}_k \), and then we compute \( r_k \) and \( \ell_k \) as above. In addition, we compute \( (\bar{X}_j)_k = n^{-1}_k \sum_{i \in C_k} X_{ij} \). We then do plots of \( r_k \) vs. \( (\bar{X}_j)_k \) and \( \ell_k \) vs. \( (\bar{X}_j)_k \). The latter plot is particularly useful for identifying trends.
Sample SAS Code

options nocenter nodate ls=80 pageno=1;

** READ DATA **;
data indat;
infile 'c:\users\david\desktop\test.txt';
input Y X1-X5;

** RUN LOGISTIC MODEL AND PUT OUT PREDICTED VALUES **;
proc logistic descending;
   model Y = X1-X5;
   output out=odat p=predval;
run;

** CREATE GROUPS BASED ON VALUE OF X1 **;
proc rank;
   var X1;
   ranks rnkprd;
run;
data groups;
   set;
   *k = [here input the desired number of groups];
   *n = [here input the number of observations in the dataset];
   k = 50;
   n = 5362;
   rnk = k * rnkprd / n;
   grp = int(rnk-.001) + 1;

** COMPUTE \bar{p}_k, r_k, and \ell_k **;
proc sort;
   by grp;
run;
proc means noprint;
   by grp;
   var X1 predval Y X1;
   output out=gmeans mean = xbar pbar ybar n(X1)=ng;
run;
data smeans;
   set gmeans;
   resid = ybar - pbar;
   pvar = pbar * (1-pbar) / ng;
   r_k = resid/sqrt(pvar);
   lym = log(ybar/(1-ybar));
   lpm = log(pbar/(1-pbar));
   ell_k = lym - lpm;
   keep xbar r_k ell_k;

** GRAPH OF r_k VERSUS X1 - THIS IS TO CHECK FOR OUTLIER RESIDUALS **;
symbol1 color=black value=dot;
proc gplot;
  plot r_k*xbar;
run;

** GRAPH OF $\ell_k$ VERSUS X1 - THIS IS TO CHECK FOR NONLINEAR TREND **;
symbol1 color=black value=dot;
proc gplot;
  plot ell_k*xbar;
run;

** CREATE NONPARAMETRIC REGRESSION CURVE BASED ON ABOVE PLOT **;
proc loess;
  model ell_k = xbar / degree=2 direct scale=sd
    smooth=0.2 0.4 0.6 0.8 1.0;
  ods output outputstatistics=ldat1;
run;

** PLOT THE ORIGINAL POINTS AND THE SMOOTH CURVE ON THE SAME GRAPH **;
proc sort data=ldat1;
  by smoothingparameter xbar;
run;
symbol1 color=black value=dot;
symbol2 color=black interpol=spline value=none;
proc gplot data=ldat1;
  by smoothingparameter;
  plot (depvar pred) * xbar / overlay;
run;