POLS6382 Quantitative Method III
Maximum Likelihood Estimation

Lab 3: Binary Choice Models (1)

Ling Zhu and Emily Lee
Department of Political Science
University of Houston

2025/09/24

1. Learning Objectives

- Learn how to estimate logit and probit models in `R`.
- Learn how to substantively interpret the results of a logit/probit model.
- Learn how to assess model fit and compare different model specifications.

2. Describing Data

We use the datafile `Court.dta` to show the estimation with `glm()`. The data file involves assessing the relative impact of legal and "attitudinal" (read: political) factors on the Supreme Court's decision making in cases involving habeas corpus. Legal academics' view of Supreme Court decision making in such cases focuses on matters relating to the law of the case. For example, it is widely believed that claims based on an assertion of ineffective counsel, because they cannot be brought at the trial level, stand a better chance of a pro-petitioner outcome than other types. In contrast, political scientists view the process as driven by political ideology: law-and-order conservatives oppose habeas claims, while liberals are more open to such arguments. Other political factors are also expected to be influential as well, such as whether the federal government is a party to the litigation (since the U.S. wins the vast majority of the cases in which it appears, opposing the federal government in a habeas case is generally believed to be much more difficult than bringing the action against a state or local government). And it is conventional wisdom that the Court frowns on second and successive petitions for habeas, making such multiple petitions more difficult to win than those of first instance.

The variable of interest is Supreme Court decision making in cases involving habeas corpus. *propetit* is the dependent variable, coded as 1 if the decision is pro-petitioner, and 0 otherwise. If we theorize that the Court's decision is influenced by its political preferences (measured as liberalism), we can empirically explore this theoretical argument by including `liberal` as the key explanatory variable. This variable is scaled between 0 and 1. We change it into a 0-to-100 scale in the subsequent analysis. Other explanatory variables are coded as the following:

- `ineffcou`– coded 1 if the habeas claim was based on an assertion of ineffective counsel, 0 otherwise.
- `tcterror`– coded 1 if the habeas claim was based on an assertion of trial court error, 0 otherwise.
- `multpet`– coded 1 if the case was a second or subsequent petition for habeas, 0 if it was a first-time petition.
- `us party`–coded 1 if the United States federal government was a party to the litigation, 0 otherwise.

- `liberal`– a continuous measure of the Court ideology scale, “liberalism", where 0 indicates the most conservative Court and 1 indicates the most liberal.

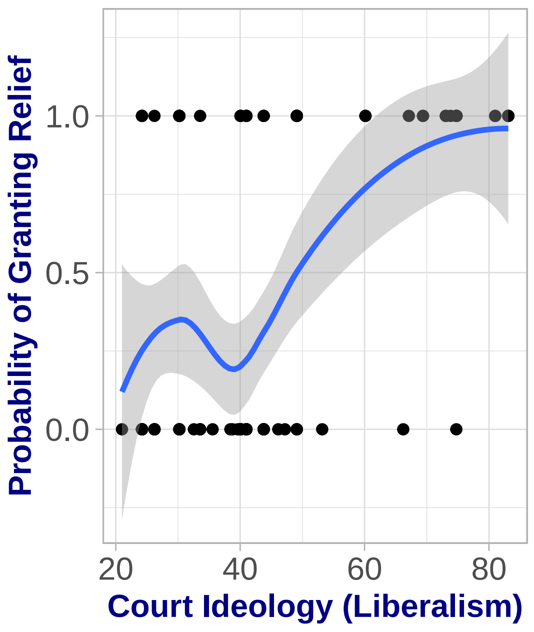
```
> rm(list=ls())
> setwd("/Users/lingzhu/Dropbox/UH Teaching/POLS6382_2025/2025 Labs/Lab 3")
> require(haven)
> mydata<-read_dta("Court.dta")
```

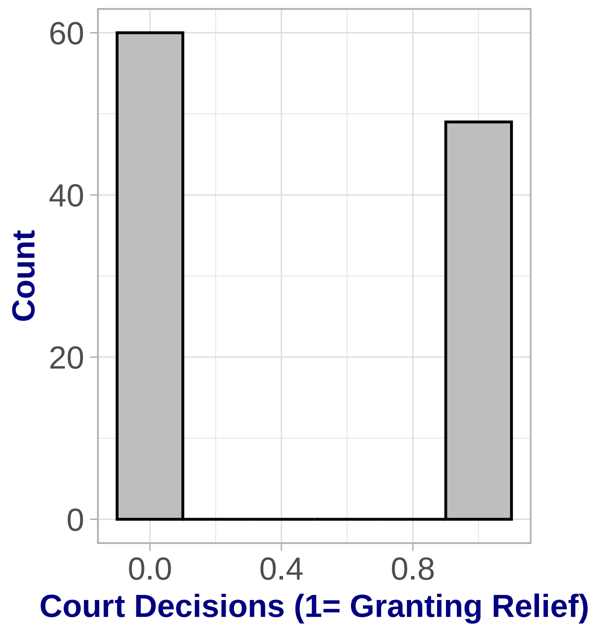
2.1 Describing Variables

Estimating a binary response model in `R` is quite simple. The `glm()` can be used to fit either a logit or a probit model. Once a binary response model has been estimated, it is often (almost always!) useful to illustrate the results using either graphs or tables. This document takes you through those steps in `R` and discusses some of the results.

We can first assess key variables by plotting decisions against the liberalism scale. The following figure also shows that fitting an OLS model is inappropriate, because the dependent variable is a dichotomous variable.

```
> require(ggplot2)
> require(ggpubr)
> mydata$liberal2<-round(mydata$liberal*100,2)
> attach(mydata)
> # Visual 1. Scatter plot of liberalism and decision.
> visual1<-ggplot(data=mydata)+
+   geom_point(aes(x=liberal2, y=propetit))+
+   geom_smooth(aes(x=liberal2, y=propetit))+
+   labs(x="Court Ideology (Liberalism)",
+        y="Probability of Granting Relief")+
+   theme_light()+
+   theme(axis.text = element_text(size = 12),
+       axis.title=element_text(size= 12,face="bold", colour="navy"),
+       plot.margin = unit(c(1,1,1,1), "cm"))
>
> # Visual 2. Histogram of the dependent variable
> visual2<-ggplot(data=mydata)+
+   geom_histogram(aes(x=propetit), binwidth = 0.2, color="black", fill="gray")+
+   labs(x="Court Decisions (1= Granting Relief)",
+        y="Count")+
+   theme_light()+
+   theme(axis.text = element_text(size = 12),
+       axis.title=element_text(size= 12,face="bold", colour="navy"),
+       plot.margin = unit(c(1,1,1,1), "cm"))
>
> # Setting figure layout as 2X1
> ggarrange(visual1, visual2, ncol=2, nrow=1)
```





2.2 Estimating a Logit or Probit Model

The proper model specification is a logit (or probit) model. We reach to the same substantive conclusion regarding how `liberalism` affects the probability of making pro-petitioner decisions.

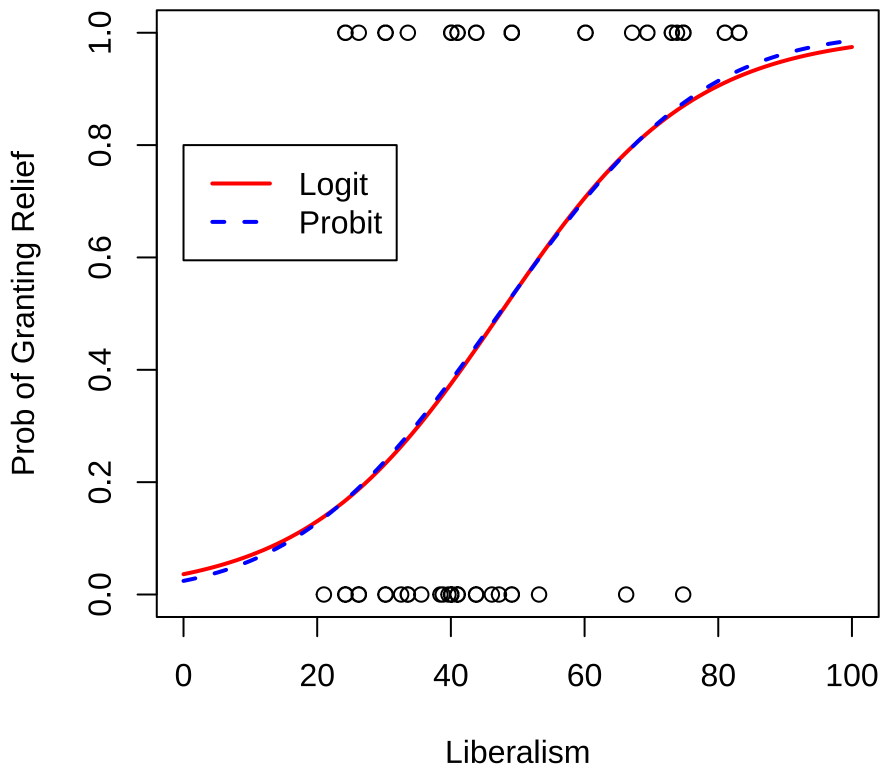
```
> # Logit Model
> logitmod <- glm(propetit ~ liberal2, family=binomial(link=logit), mydata)
> # Probit Model
> probitmod<-glm(propetit ~ liberal2, family=binomial(link=probit), mydata)
> require(stargazer)
> stargazer(logitmod, probitmod, type="text")
```

```
==============================================
                      Dependent variable:
                  ----------------------------
                           propetit
                     logistic       probit
                        (1)           (2)
----------------------------------------------
liberal2             0.069***      0.042***
                     (0.016)        (0.009)

Constant            -3.282***     -1.973***
                     (0.727)        (0.407)

----------------------------------------------
Observations           109            109
Log Likelihood       -61.561        -61.544
Akaike Inf. Crit.    127.121        127.088
==============================================
Note:              *p<0.1; **p<0.05; ***p<0.01
```

We can obtain predicted probabilities based on the above estimated logit model and probit model. In-sample prediction uses observed values of the liberalism scale.

```
> # In-sample Point prediction
> summary(mydata$liberal2)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  21.00   30.22   41.00   44.89   49.11   83.11
```

```
> # Prediction when liberal is at its min, median, and max
> library(faraway)
> ilogit(-3.2828+0.06929*21)

[1] 0.1385113

> pnorm (-1.9731+0.04176*21)

[1] 0.1365088

> ilogit(-3.2828+0.06929*41)

[1] 0.3912859

> pnorm (-1.9731+0.04176*41)

[1] 0.3970694

> ilogit(-3.2828+0.06929*83.11)

[1] 0.9224344

> pnorm (-1.9731+0.04176*83.11)

[1] 0.932878

> plot(propetit~liberal2,mydata, xlim=c(0,100), ylim = c(0,1),
+ xlab="Liberalism", ylab="Prob of Granting Relief")
> x <- seq(0,100,1)
> lines(x,ilogit(-3.2828+0.06929*x),lty=1,col="red",lwd=2)
> lines(x,pnorm(-1.9731+0.04176*x),lty=2,col="blue",lwd=2)
> legend(0,0.8, lty=c(1,2), lwd=c(2,2),col=c("red","blue"),
+ c("Logit","Probit"),cex=1)
```

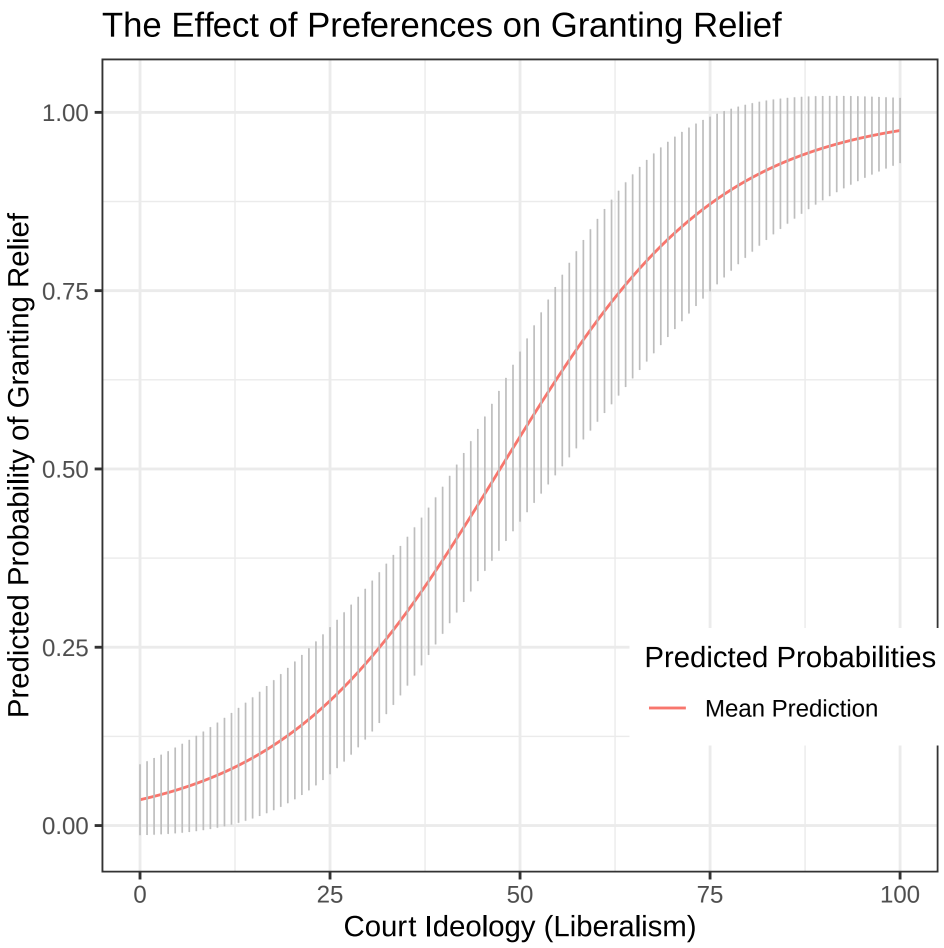


2.3 Predicted Probabilities with Confidence Intervals

Once we estimated a logit/probit model, we can use out-sample prediction to calculate predicted probabilities corresponding to specific values of `liberalism`, and their corresponding confidence intervals. It takes a few steps as the following:

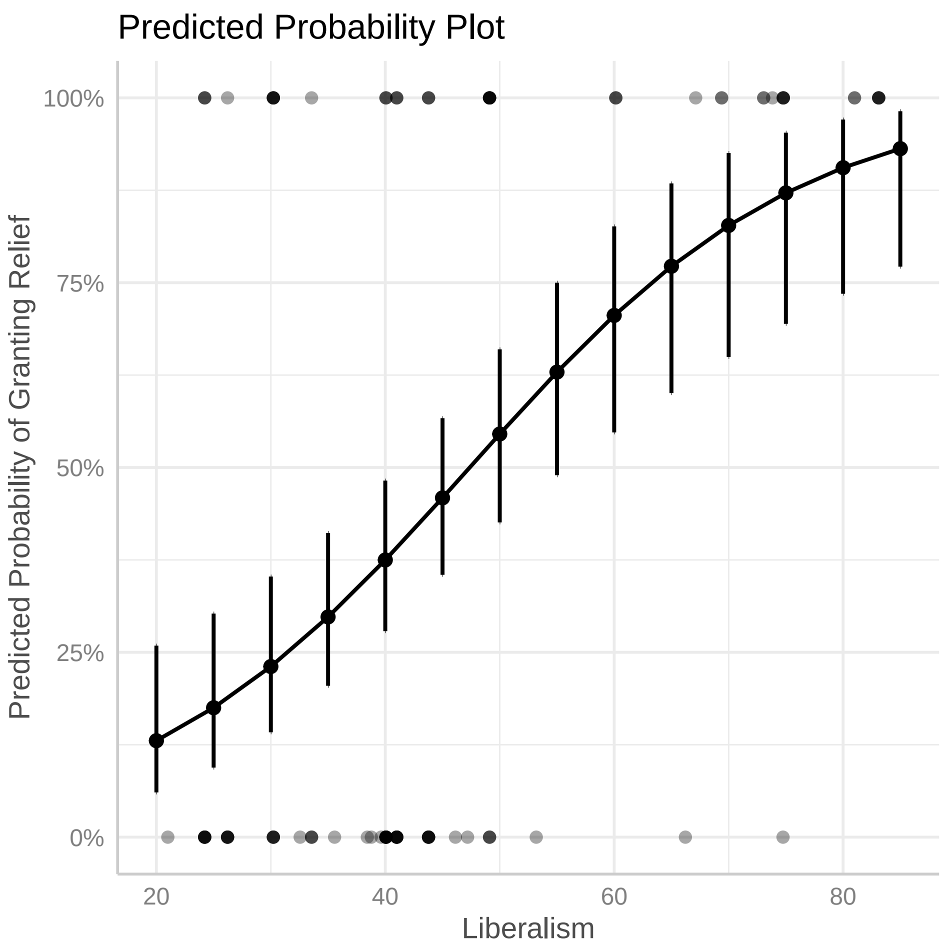
- Set up a hypothetical scale of liberalism, ranging from 0 to 100. You can also choose to set the range to be the observed range, (21,83).
- Using the function, `predict()` to calculate predicted probabilities based on the logit model. Make sure to set "se=TRUE", so that R will return the standard errors of the predicted values.
- Obtain predicted values, "fit", then calculate the confidence intervals.
- Plot the predicted probabilities and CIs.

```
> newcourt <- data.frame(liberal2 = seq(from=0,to=100,length.out=109))
> #newcourt
> newcourt<- cbind(newcourt, predict(logitmod, newdata = newcourt,se = TRUE, type="response"))
>
> # Add CIs
> newcourt <- within(newcourt, {
+   LL <- fit - (1.96 * se.fit)
+   UL <- fit + (1.96 * se.fit)
+   PredictedProb <- fit
+ })
>
> # Plot predicted probabilities with CIs
> ggplot(newcourt, aes(x=liberal2, y=PredictedProb,colour="Mean Prediction")) +
+   geom_line()+
+   geom_errorbar(aes(ymin = LL, ymax = UL,colour="gray"),
+                 colour="gray",
+                size=.3,    # Thinner lines
+                width=0.01) + #width of the whiskers
+   labs(x="Court Ideology (Liberalism)",
+        y="Predicted Probability of Granting Relief",
+        colour="Predicted Probabilities") +
+   ggtitle("The Effect of Preferences on Granting Relief") +
+   theme_bw()+
+   theme(legend.justification=c(.7,1),
+         legend.position=c(.9,.3))
```



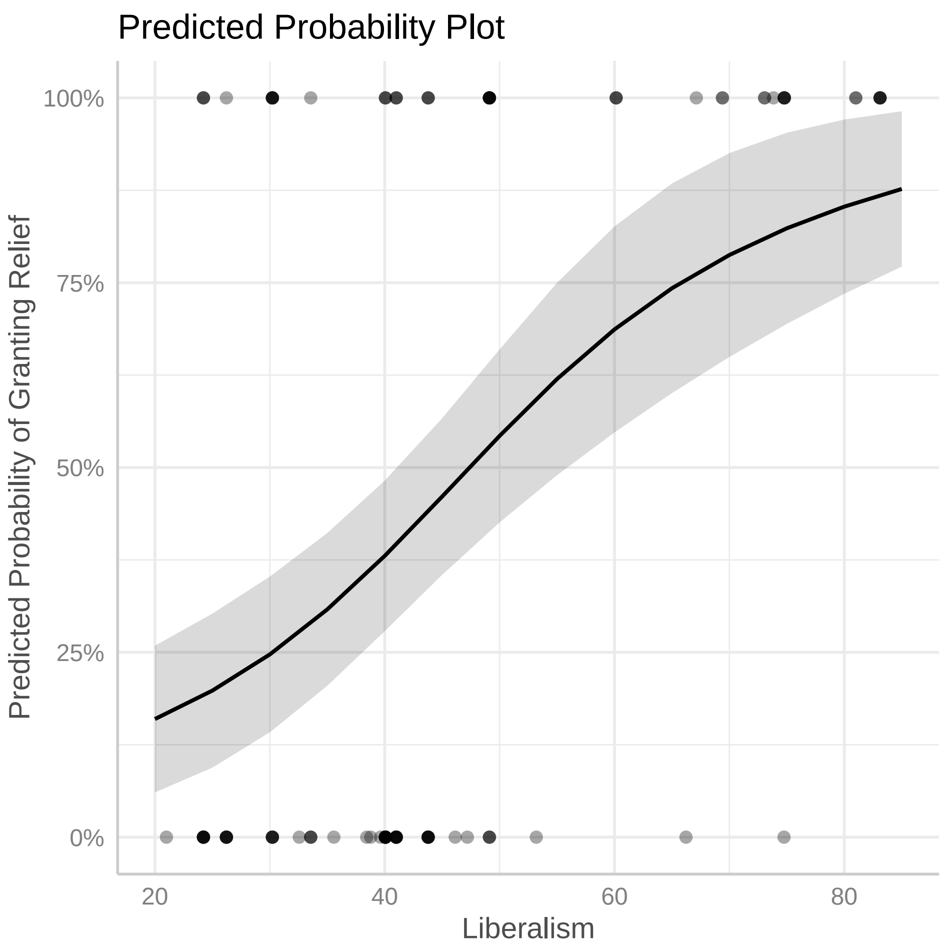
Package `ggeffects` can also be used to produce the visual of predicted probabilities. But be aware that ggeffect() or ggpredict() will produce asymmetric CIs.

```
> # Using package ggeffects
> require(tidyverse)
> require(ggeffects)
> ggeffect(logitmod, terms="liberal2", type="link") |> plot(ci_style="errorbar", show_data=TRUE)+
+   theme_ggeffects()+
+   labs(x="Liberalism",
+        y="Predicted Probability of Granting Relief",
+        title="Predicted Probability Plot ")
```



Why are the CIs asymmetric? In a logistic regression, the model is fit on the log-odds (logit) scale, where the link function is linear. Confidence intervals are symmetric around the linear predictor (logit(p)), because that is where standard errors are computed.The transformation from log-odds to probabilities is nonlinear, so symmetric intervals in logit-space become asymmetric intervals in probability space. If we want to use ggeffect() or ggpredict(), we will need to manually adjust the mean predictions to obtain symmetric CIs.

```
> logit_pred<-ggpredict(logitmod, terms="liberal2")
> logit_pred$predicted=(logit_pred$conf.low+logit_pred$conf.high)/2
> plot(logit_pred,ci_style="ribbon", show_data=TRUE)+
+   theme_ggeffects()+
+   labs(x="Liberalism",
+        y="Predicted Probability of Granting Relief",
+        title="Predicted Probability Plot ")
```



# 3. Hypothesis Testing, Inference and Substantive Interpretation

3.1 Hypothesis Testing

For both the logit and probit model, `glm()` returns the slope coefficients, their corresponding standard errors and significance levels. We can also get confidence intervals for the estimated coefficients. `confint()` calculates the CIs by profiling likelihood. `confint.default()` does so using an asymptotic normal distribution. The two functions will produce comparable CIs. As we can see, the logit model produces a slope coefficient of 0.069, but its 95% confidence intervals are [0.040 0.104] (using `confint()`), and [0.038 0.101] (using `confint.default()`).

```
> require(MASS)
> # Profiling likelihood
> confint(logitmod)
```

```
                  2.5 %     97.5 %
(Intercept) -4.83276706 -1.9507744
liberal2     0.04036164  0.1042497
```

```
> confint.default(logitmod)
```

```
                  2.5 %     97.5 %
(Intercept) -4.70817715 -1.8568006
liberal2     0.03774766  0.1008244
```

Summarizing estimation results, we can assess the z-scores for hypothesis testing. A typical Null hypothesis is that the estimated coefficient equals to 0: $H_0 = 0$. But we can use `wald.test()` from the `aod` package to evaluate different hypotheses.

```
> require(aod)
> wald.test(Sigma=vcov(logitmod),b=coef(logitmod),Terms=2:2,H0=1,verbose=TRUE)
```

```
Wald test:
----------

Coefficients:
(Intercept)    liberal2
     -3.282       0.069

Var-cov matrix of the coefficients:
            (Intercept) liberal2
(Intercept)  0.52912    -0.01116
liberal2    -0.01116     0.00026

Test-design matrix:
   (Intercept) liberal2
L1           0        1

Positions of tested coefficients in the vector of coefficients: 2

H0:  liberal2 = 1

Chi-squared test:
X2 = 3345.4, df = 1, P(> X2) = 0.0
```

The first argument in `wald.test()` is that `Sigma=vcov(logitmod)`. This statement calls out the variance estimates from the variance-covariance matrix. Note that variance associated with each parameter are listed along the diagonal in the Var-Cov matrix. Hence, for the slope, we want to get the scalar number in [2:2] from the Var-Cov matrix. The second statement, `b=coef(logitmod)` calls out the vector of coefficients. `H0=1` defines the null hypothesis. `verbose=TRUE` is a logic statement defining that `R` will produce detailed outputs. If defined as “FALSE”, `R` only returns a minimum set of information. The test returns Chi-square statistics. A near-zero p value indicates that the null hypothesis should be rejected, therefore, the slope coefficient is different from 1.

3.2 Odds Ratios (only for Logit Models)

After estimating a logit model, we can also convert coefficients into odds ratios, which are easy to interpret. Based on the mean odds ratio, a one-point increase in the Court’s liberalism (on a 1-to-100 scale), will increase the probability of making pro-petitioner decision by 7%.

```
> exp(logitmod$coefficients)
```

```
(Intercept)    liberal2
 0.03753472  1.07174270
```

4. Consider Alternative Specification

4.1 Unconstrained Model

The logit/probit model reported in Table 1 may be under-specified, because it only includes `liberal2" as an independent variable. We may consider an alternative model, which includes three more independent variables:`usparty”, `ineffcou", and`multpet”. The function `update()` is used to define our second logit regression model as to updating first logit model by adding these three variables. We see that both `ineffcou" and`multpet" are significant predictors of the dependent variable.

```
> logitmod2<-update(logitmod, .~.+usparty+ineffcou+multpet)
> stargazer(logitmod, logitmod2, type="text")
```

```
==============================================
                      Dependent variable:
                  ----------------------------
                            propetit
                       (1)            (2)
----------------------------------------------
liberal2            0.069***      0.080***
                    (0.016)        (0.018)

usparty                            -0.777
                                   (0.784)

ineffcou                           1.459*
                                   (0.810)

multpet                            -1.411*
                                   (0.832)

Constant           -3.282***      -3.655***
                    (0.727)        (0.799)

----------------------------------------------
Observations          109            109
Log Likelihood      -61.561        -58.131
Akaike Inf. Crit.   127.121        126.263
==============================================
Note:              *p<0.1; **p<0.05; ***p<0.01
```

4.2 Adding an Interaction Term

Variable`usparty" is not a significant predictor of the dependent variable in our second logit model. But if we theorize that this is a conditional variable to`liberalism" (i.e., the effect of `liberalism" on`propetit" is conditional on the variable `usparty"), we can respecify our model by including an interaction term between the two variables. In this model, we only consider`liberal2", `usparty" and the multiplicative term`liberal2" × `usparty". We find that only`liberal2" has a significant coefficient. The coefficients for `usparty" and the interaction term are both insignificant. This indicates that the effect of`liberals" on the probability of granting relief (i.e., making pro-petitioner decision) may not be conditioned by "usparty".

```
> logitmod3<-update(logitmod, .~.+usparty+liberal2*usparty)
> stargazer(logitmod, logitmod2, logitmod3, type="text")
```

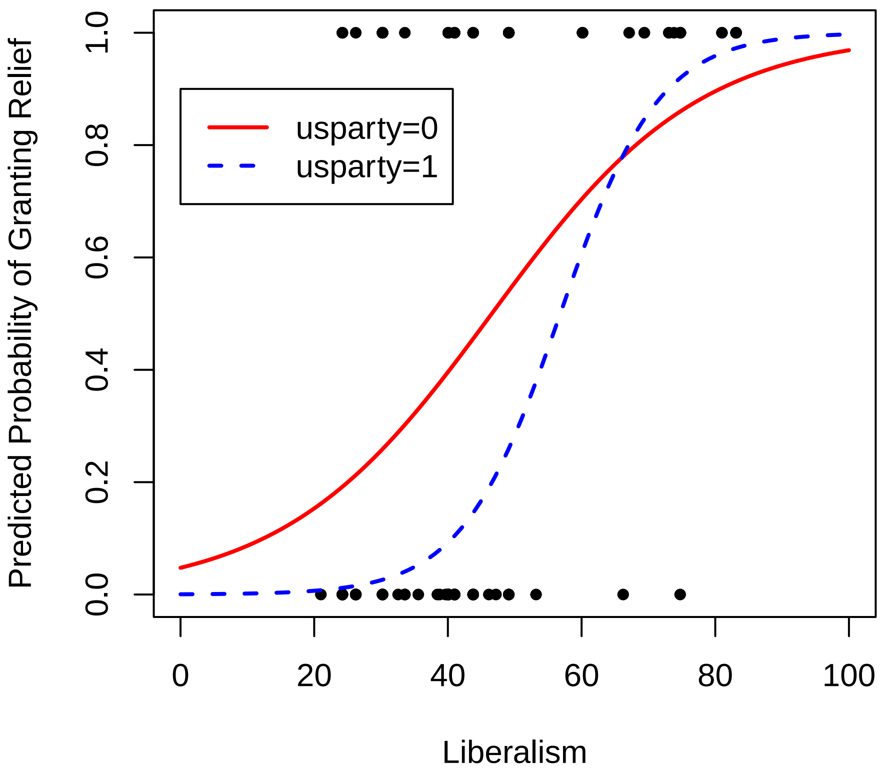
```
===============================================
                      Dependent variable:
                  -----------------------------
                            propetit
                     (1)       (2)       (3)
-----------------------------------------------
```

```
liberal2          0.069***  0.080***  0.064***
                   (0.016)   (0.018)   (0.017)

usparty                     -0.777    -4.721
                            (0.784)   (3.798)

ineffcou                     1.459*
                            (0.810)

multpet                     -1.411*
                            (0.832)

liberal2:usparty                       0.071
                                      (0.065)

Constant          -3.282*** -3.655*** -2.997***
                   (0.727)   (0.799)   (0.769)

-----------------------------------------------
Observations         109       109       109
Log Likelihood     -61.561   -58.131   -60.134
Akaike Inf. Crit.  127.121   126.263   128.267
===============================================
Note:               *p<0.1; **p<0.05; ***p<0.01
```

4.3 Interaction Figure

When estimating an interaction model, only looking at the reported coefficients is not a sufficient way to gauge if there is any significant interactive relationship between the two explanatory variables. We need to plot an interaction figure to draw the conclusion. A commonly used method to show an interactive relationship is to plot predicted probabilities across the full range of variable A , holding variable B (the conditional variable) at its low and high value. Because `usparty" is a dummy variable, we can just plot the predicted probabilities across the full range of`liberal2" for two scenarios: (1) `usparty"=1, and (2)`usparty"=0. The first statement in `plot()` simply plot the observed cases along the liberalism scale. Next, we obtain the predicted probabilities using the the function `predict()`. We plot these predicted probabilities across a hypothetical liberalism scale (from 0 to 100, length-out 100 values.) Note that the equations used to calculate predicted probabilities when `usparty" equals to 0 and 1 are different. When`usparty" equals to 0, the interaction term and variable `usparty" are excluded from the calculation. When`usparty" equals to 1, we have to include the coefficients of `usparty" and the interaction term, and just define the value for`usparty" as 1.

```
> plot(propetit~liberal2,mydata, xlim=c(0,100), ylim = c(0,1), pch=20,
+      xlab="Liberalism", ylab="Predicted Probability of Granting Relief")
> lines(x,ilogit(-2.99693+0.06435*x),lty=1,col="red",lwd=2)
> lines(x,ilogit(-2.99693+0.06435*x-4.72104*1+0.07148*x),lty=2,col="blue",lwd=2)
> legend(0,0.9, lty=c(1,2), lwd=c(2,2),col=c("red","blue"),c("usparty=0","usparty=1"),cex=1)
```



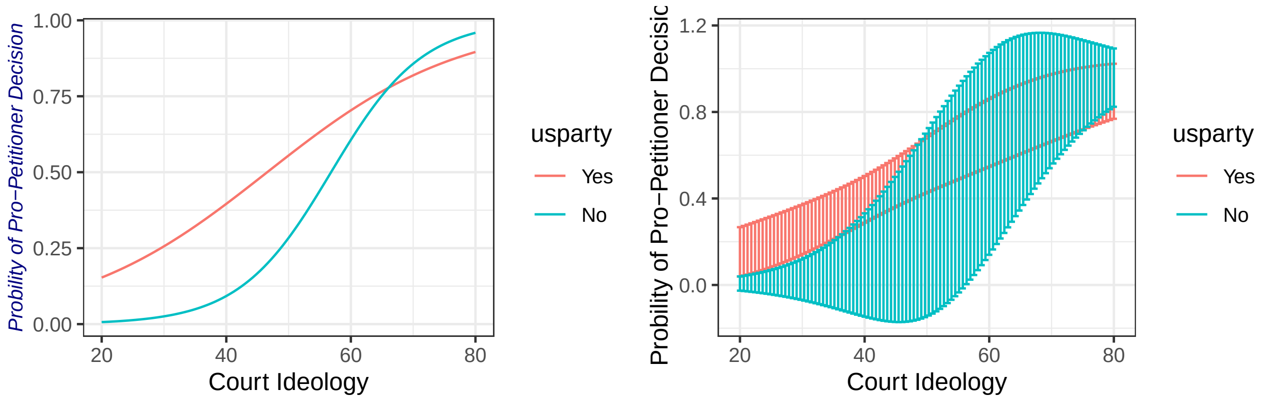
From the above figure, we observe that the predicted probabilities associated with `liberal2" are different when`usparty” takes different values. The only cross-point is when liberalism is around 65. The figure also suggests that when the Court is conservative, the decisions seem to be very different when U.S. government is an involved party and when U.S. government is not an involved party. But can we really draw the conclusion that the two sets of predicted probabilities are statistically different from each other? Not really. We need to estimate their corresponding confidence intervals to evaluate the observed difference.

We first create a new data frame for the purpose of obtaining predictions. Now, we produce out-of-sample predictions, by setting `liberal2" at its full range, setting`usparty” at two different values. If we included another predictor in the model, we can add code as “variable=mean(variable)” to hold the other variable at its mean. We then create a third data frame, which combines this simulated data frame and the two columns of predicted values. Using `predict()`, we get fitted values by specifying `type="response"`. By setting the statement `se=TRUE`, we also get the standard errors. Next, we convert predicted probabilities. Calculating the confidence intervals is just simple–we use `fit - (1.96 * se.fit)` to get the lower bound, and `fit + (1.96 * se.fit)` to get the upper bound. Scalar 1.96 is chosen to produce the 95% CIs.

Plotting the mean predicted probabilities, we get a similar figure to the previous one. We then plot another figure with CIs, using the error bar option from the `ggplot2` package. Now we see that the CIs for the two sets of predictions overlap. This means that the predicted probabilities are not statistically different when “usparty” equals to 1 or 0.

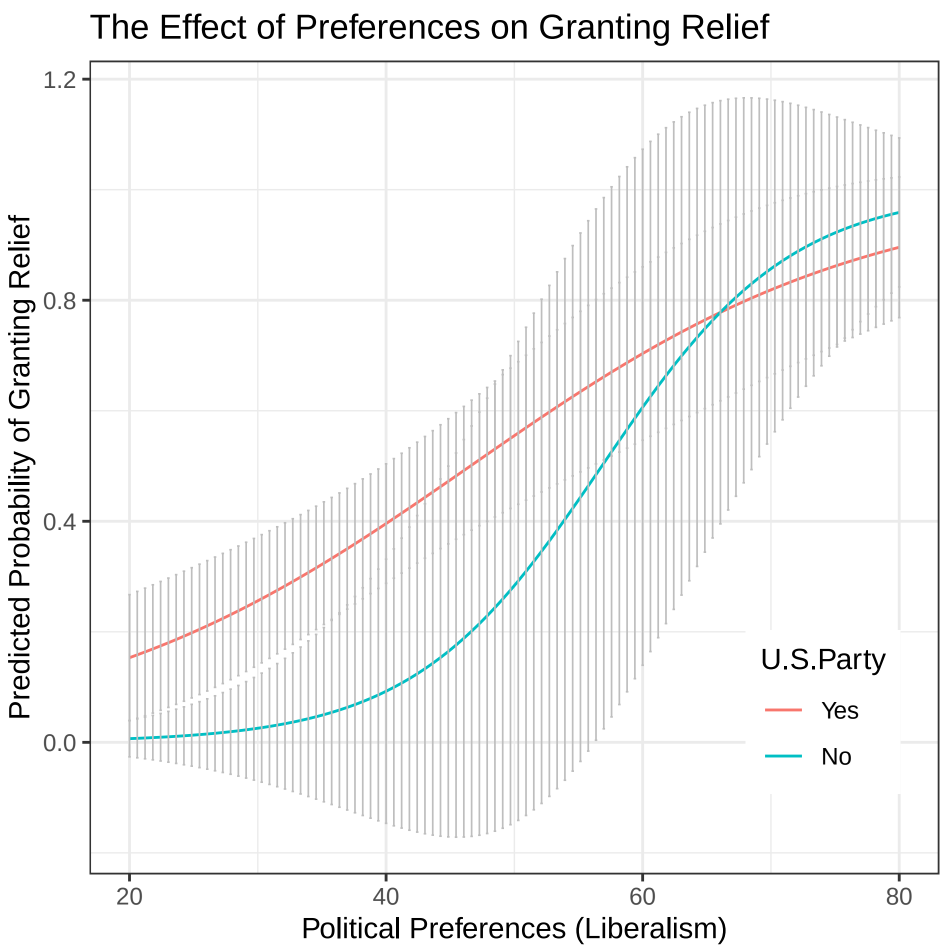
```
> newdata2 <- with(mydata, data.frame(liberal2 = rep(seq(from = 20, to = 80, length.out = 100),
+                  2), usparty = rep(0:1, each = 100)))
> newdata3 <- cbind(newdata2, predict(logitmod3, newdata = newdata2, type = "response",
+                  se = TRUE))
> # Add CIs
> newdata3 <- within(newdata3, {
+   LL <- fit - (1.96 * se.fit)
+   UL <- fit + (1.96 * se.fit)
+   PredictedProb <-fit
+ })
```

```
>
> # Recode usparty as a factor
> newdata3$usparty<-factor(newdata3$usparty, labels=c("Yes", "No"))
> attach(newdata3)
> # Plot out-sample predictions with CIs
> interaction1<-ggplot(newdata3, aes(x = liberal2, y = PredictedProb,
+     colour=usparty)) + xlab("Court Ideology")+
+     ylab("Probility of Pro-Petitioner Decision")+
+     geom_line()+theme_bw()+
+     theme(axis.title.y=element_text(size=9, color="navy", face="italic"))
>
> interaction2<-ggplot(newdata3, aes(x = liberal2, y = PredictedProb,
+       colour=usparty))+xlab("Court Ideology")+
+       ylab("Probility of Pro-Petitioner Decision")+
+       geom_errorbar(aes(ymin = LL, ymax = UL),width=1)+theme_bw()
>
> ggarrange(interaction1, interaction2, ncol=2, nrow=2)
```



We can also plot both the mean predictions and their CIs in one figure. A finished graph might look like this.

```
> ggplot(newdata3, aes(x=liberal2, y=PredictedProb, colour=usparty)) +
+   geom_line()+
+   geom_errorbar(aes(ymin = LL, ymax = UL),
+                 colour="gray",
+                 size=.3,     # Thinner lines
+                 width=.2,
+                 position=position_dodge(.9)) +
+   labs(x="Political Preferences (Liberalism)",
+        y="Predicted Probability of Granting Relief",
+        colour="U.S.Party") +
+   scale_fill_hue(breaks=c("Yes", "No"),
+                  labels=c("Yes",  "No")) +
+   ggtitle("The Effect of Preferences on Granting Relief") +
+   theme_bw()+
+   theme(legend.justification=c(.7,1),
+     legend.position=c(.9,.3))
```



5. Assessing Model Fit

5.1 Classification-Based Approach

We have ruled out the model specification that includes an interaction term between `liberal2" and`usparty". Our further consideration should be to compare which model produces a better fit, first logit model (only including one predictor) or the second logit model (including four predictors). Three are three different approaches to assess model fit.

Both ePCP (expected proportion of correct prediction) and ROC (receiver operating characteristic curve) are commonly used classification-based approach. In essence, we assess model fit by examining how good a model predicts responses, taking the observed outcomes as the baseline.

```
> y<-mydata$propetit
> pred1<-predict(logitmod,type="response")
> pred2<-predict(logitmod2,type="response")
> require(OOmisc)
> epcp1<-ePCP(pred1, y, alpha = 0.05)
> epcp1
```

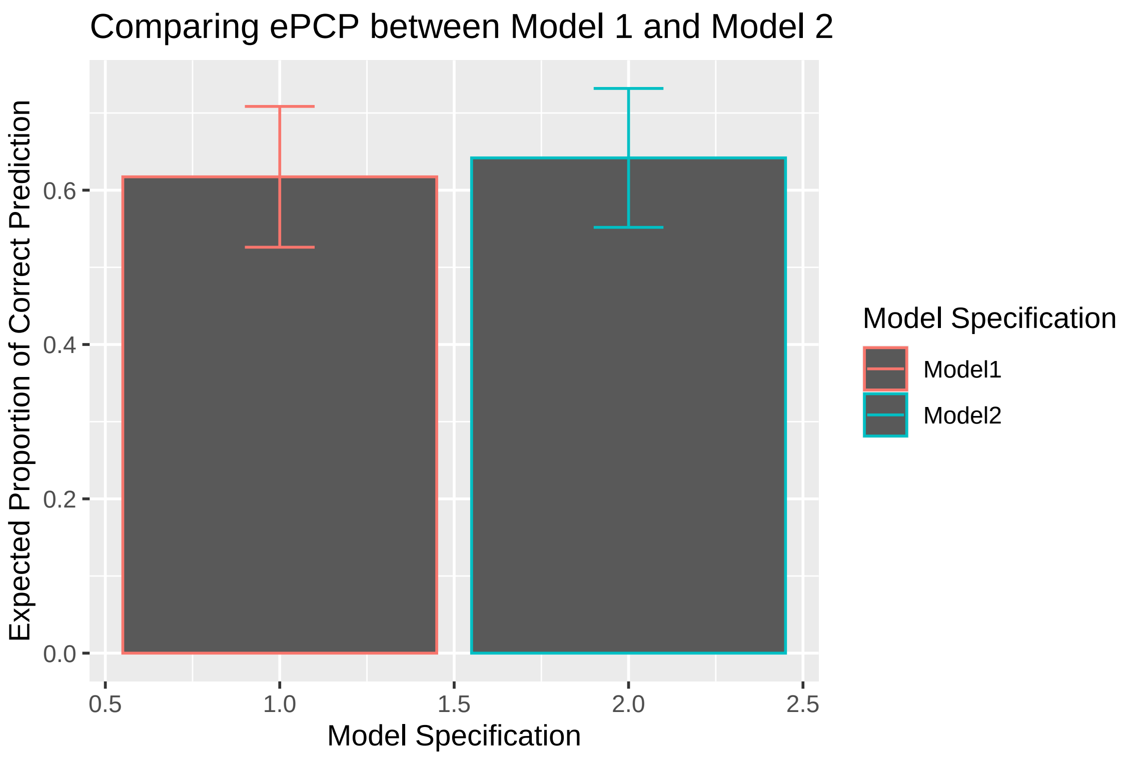
```
     ePCP     lower     upper
0.6173113 0.5260661 0.7085564
```

```
> epcp2<-ePCP(pred2, y, alpha = 0.05)
> epcp2
```

```
     ePCP     lower    upper
 0.6418239 0.5518139 0.731834
```

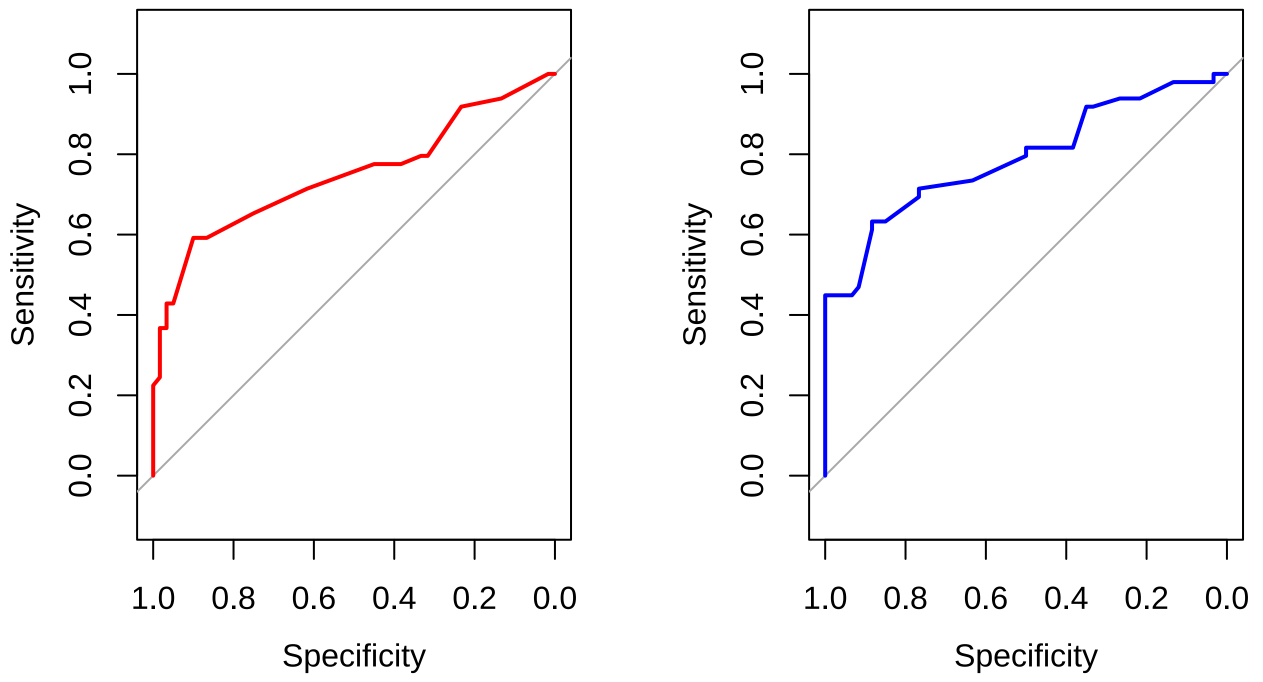
Using function `ePCP()`, we get the ePCP statistics associated with model 1 and 2. Model 2 has a greater mean expected proportion of correct prediction than model 1, therefore, based on mean ePCP, model 2 (with four predictors) produces a better fit than model 1. Because `ePCP()` also produce CIs, we can further compare the two sets considering CIs. Shown in the following figure, ePCP really is not that helpful. The two models have comparable levels of goodness of fit.

```
> epcpdata<-data.frame(rbind(epcp1,epcp2))
> epcpdata$model<-c(1,2)
> epcpdata$count<-factor(c(1,2),label=c("Model1","Model2"))
> ggplot(epcpdata, aes(x=model,y=ePCP,colour=count))+
+   geom_bar(position=position_dodge(), stat="identity")+
+   geom_errorbar(aes(ymin=lower, ymax=upper),
+                 width=.2,
+                 position=position_dodge(.9))+
+   labs(x="Model Specification",
+        y="Expected Proportion of Correct Prediction",
+        colour="Model Specification") +
+   ggtitle("Comparing ePCP between Model 1 and Model 2")
```



We can also use ROC to further compare the two models. A ROC plots correct predictions (sensitivity, true positive rate) against false predictions (specificity, false positive rate). A model with excellent fit will produce a curve that is very close to the left and top bound. In other words, the area under the curve is large (near 1). A model with poor fit will produce a curve close to the 45-degree diagonal line (whereby it is equally likely for us to see correct and false predictions). In other words, a model has a very poor fit, the area under the curve is very small (near 0). Based on ROC, model 2 produces a better fit than model 1.

```
> require(pROC)
> par(mfrow = c(1, 2))
> plot.roc(y,pred1,col="red")
> plot.roc(y,pred2,col="blue")
```



5.2 Likelihood-Based Approach

We can also compare the AIC statistics to assess model fit. Formally, AIC is defined as:

$$AIC = 2k - 2\ln(L) \tag{1}$$

A better model fit is associated with a smaller AIC. Model 2 produce slightly smaller AIC than Model 1. Therefore, model 2 produces a better fit.

```
> extractAIC(logitmod)
```

```
[1]   2.000 127.121
```

```
> extractAIC(logitmod2)
```

```
[1]   5.0000 126.2627
```

We can also use the likelihood ratio test to compare model fit. We define the test statistics as:

$$D = 2lnL(M_U) - 2lnL(M_C) \tag{2}$$

Where, $L(M_U)$ is the likelihood evaluated at the ML estimates for the unconstrained model (in our case, model 2). $L(M_C)$ is the likelihood evaluated at the ML estimates for the constrained model (in our case, model 1). Large Chi-square statistics and the small p value indicate that model 2 is better than model 1.

```
> require(lmtest)
> lrtest(logitmod,logitmod2)
```

```
Likelihood ratio test

Model 1: propetit ~ liberal2
Model 2: propetit ~ liberal2 + usparty + ineffcou + multpet
  #Df  LogLik Df  Chisq Pr(>Chisq)
1   2 -61.561
```

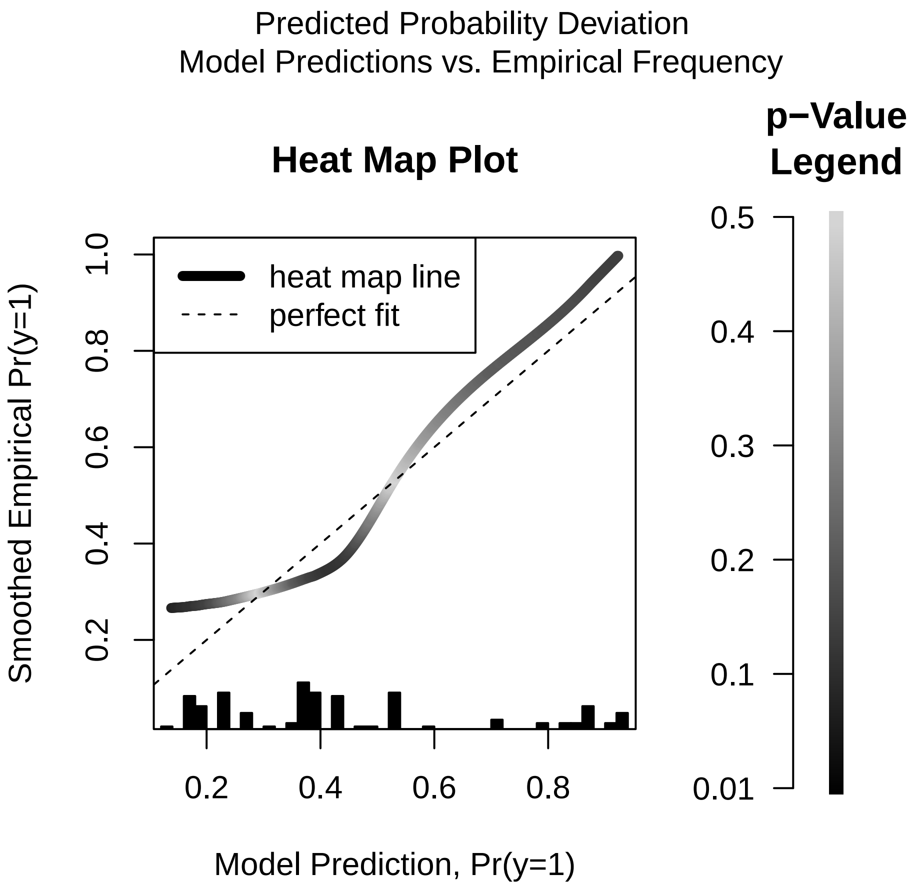
```
2   5 -58.131  3 6.8583    0.07655 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5.3 Heat Map Fit Plot

Last, but not least, we can plot a heat map fit plot to compare goodness of fit. See Esarey and Pierce's 2012 *Political Analysis* paper for more theoretical discussions on their new approach. When reading a heat map plot, we also compare model predictions with smoothed empirical predictions. The 45-degree line references a perfect fit. Any deviance from that line suggests a loss in goodness of fit. P-value legend shows if any deviance is statistically significant (dark color means statistical significance). The following two heat map plots show that model 2 outperforms model 1.

```
> require(heatmapFit)
> heatmap.fit(y,pred1,reps=1000,legend=TRUE)
```

```
Calculating optimal loess bandwith...
aicc Chosen Span =  0.8216853

Generating Bootstrap Predictions...
  |                                                                                   |
```

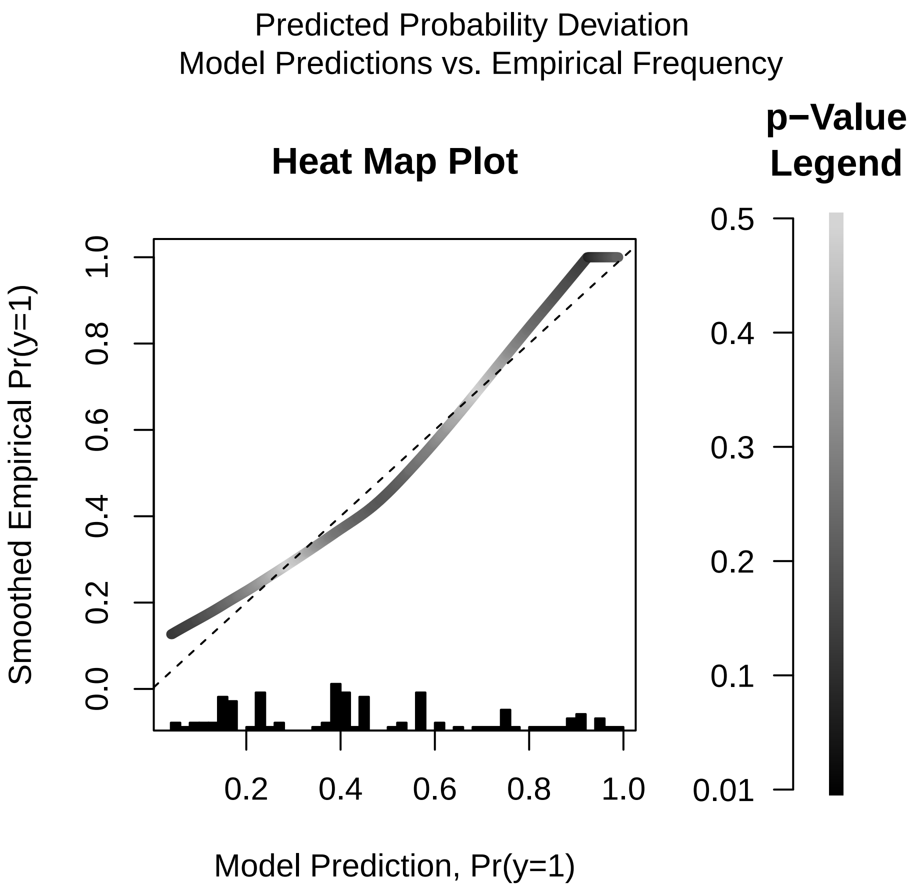


```
********************************************
0.9174312% of Observations have one-tailed p-value <= 0.1
```

```
Expected Maximum = 20%
********************************************
```

```
> heatmap.fit(y,pred2,reps=1000,legend=FALSE)
```

```
Calculating optimal loess bandwith...
aicc Chosen Span =  0.9751618

Generating Bootstrap Predictions...
  |                                                                                |
```



```
********************************************
0% of Observations have one-tailed p-value <= 0.1
Expected Maximum = 20%
********************************************
```