Linear Probability Model

R Tutorials for Applied Statistics

Note on required packages: The following code requires the packages `sandwich`, `lmtest` and `tidyverse`. The packages `sandwich` and `lmtest` include functions to estimate regression error variance that may change with the explanatory variables. The package `tidyverse` is a collection of packages convenient for manipulating and graphing data. If you have not already done so, download, install, and load the libraries with the following code:

```
# This only needs to be executed once for your machine
install.packages("lmtest")

# This only needs to be executed once for your machine
install.packages("sandwich")

# This only needs to be executed once for your machine
install.packages("tidyverse")

# This needs to be executed every time you load R
library("lmtest")

# This needs to be executed every time you load R
library("sandwich")

# This needs to be executed every time you load R
library("tidyverse")
```

---

1 Introduction

We established in a previous tutorial that binary variables can be used to estimate *proportions* or *probabilities* that an event will occur. If a binary variable is equal to 1 for when the event occurs, and 0 otherwise, estimates for the mean can be interpreted as the probability that the event occurs.

A **linear probability model (LPM)** is a regression model where the *outcome* variable is a binary variable, and one or more explanatory variables are used to predict the outcome. Explanatory variables can themselves be binary, or be continuous.

2 Data Set: Mortgage loan applications

The data set, `loanapp.RData`, includes actual data from 1,777 mortgage loan applications, including whether or not a loan was approved, and a number of possible explanatory variables including demographic information of the applicants and financial variables related to the applicant's ability to pay the loan such as the applicant's income and employment information, value of the mortgaged property, and credit history.

The code below loads the `R` data set, which creates a data frame called `df`, and a list of descriptions for the variables called `desc`.

```
load(url("http://murraylax.org/datasets/loanapp.RData"))
```

3 Estimating a Linear Probability Model

3.1 Model Setup

Let us estimate a linear probability model with loan approval status as the outcome variable (`approve`) and the following explanatory variables:

- `loanprc`:Loan amount relative to price of the property
- `loaninc`: Loan amount relative to total income
- `obrat`: Value of other debt obligations relative to total income
- `mortno`: Dummy variable equal to 1 if the applicant has no previous mortgage history, 0 otherwise
- `unem`: Unemployment rate in the industry where the applicant is employment

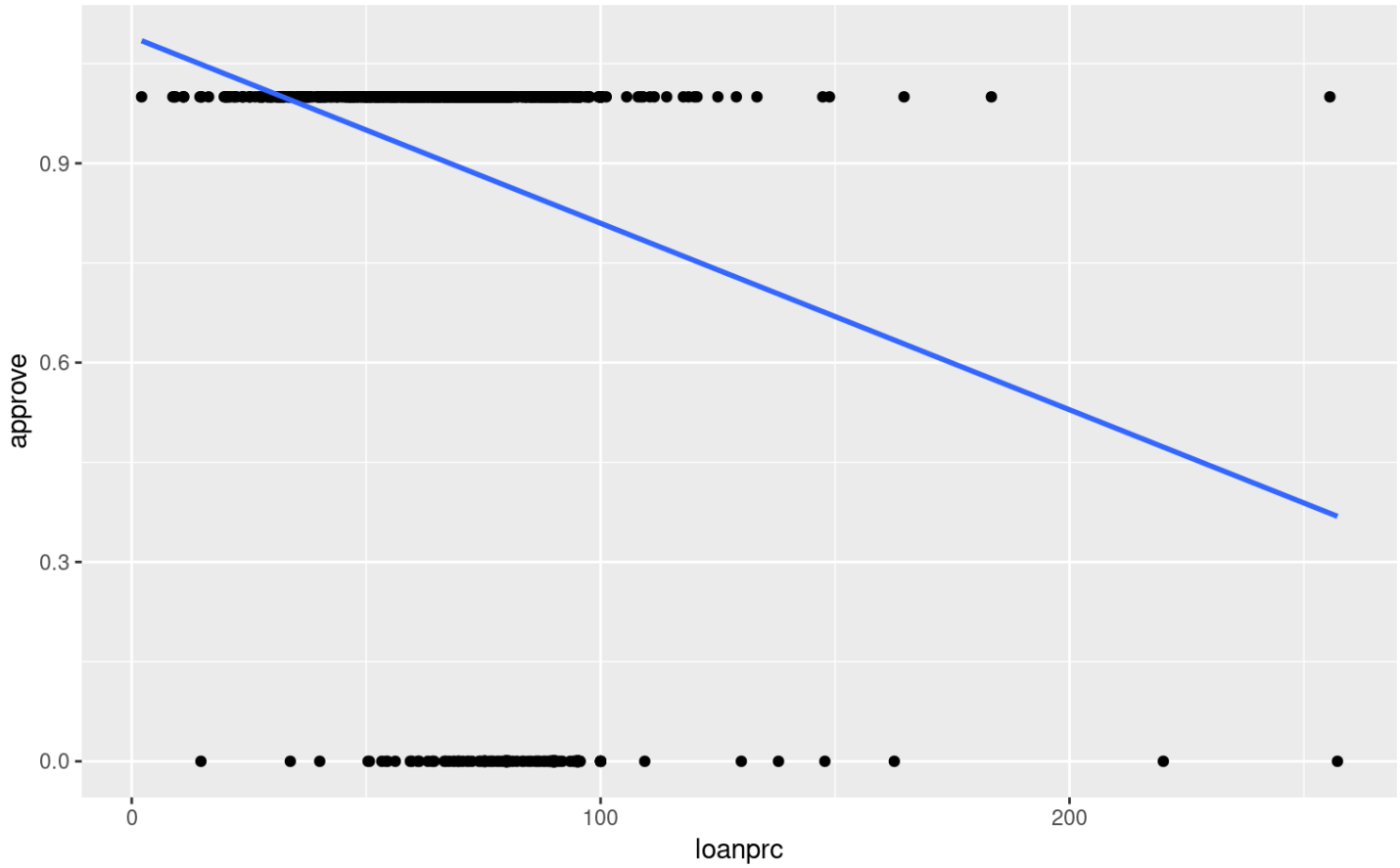
```
lmapp <- lm(approve ~ loanprc + loaninc + obrat + mortno + unem, data=df)
summary(lmapp)
```

```
##
## Call:
## lm(formula = approve ~ loanprc + loaninc + obrat + mortno + unem,
##     data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.05548  0.03789  0.11512  0.16194  0.52705
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.240e+00  4.374e-02  28.356  < 2e-16 ***
## loanprc     -1.927e-03  4.188e-04  -4.601 4.51e-06 ***
## loaninc     -4.676e-05  5.604e-05  -0.835  0.40409
## obrat       -5.906e-03  9.597e-04  -6.154 9.31e-10 ***
## mortno       5.358e-02  1.661e-02   3.225  0.00128 **
## unem        -8.628e-03  3.520e-03  -2.451  0.01433 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3228 on 1771 degrees of freedom
## Multiple R-squared:  0.05718,    Adjusted R-squared:  0.05452
## F-statistic: 21.48 on 5 and 1771 DF,  p-value: < 2.2e-16
```

3.2 Visualizing the Linear Probability Model

Let us visualize the actual and predicted outcomes with a plot. The code below calls the `ggplot()` function to visualize the how loan approval depends on the size of the loan as a percentage of the price of the property.

```
ggplot(data=df, mapping=aes(x=loanprc, y=approve)) +
  geom_point() + geom_smooth(method="lm", se=FALSE)
```



On the vertical axis we have the actual value of `approve` (equal to 0 or 1) or the predicted probability of a loan approval. The black points show the actual values and the blue line shows the predicted values.

The first parameter sets the data layer, pointing to the data frame, `df`.

The second parameter sets the aesthetics layer (also known as mapping layer). We call the function `aes()` to map the variable `loanprc` to the x-axis and `approve` to the y-axis.

Next we add the geometry layer with a call to geom_point(). This produces a scatter plot with points.

Finally, we create the best fit linear regression line using the function `geom_smooth(method="lm", se=FALSE)`. This function creates both a geometry and a statistics layer. The function estimates the best fit simple linear regression function (using `loanprc` as the only explanatory variable) using the function `lm()`. We set `se=FALSE` because we do not wish to view the confidence bounds around the line. As we discuss below, the standard errors computed by the `lm()` function that are used to create the confidence bounds are incorrect for a linear probability model.

It is a strange looking scatter plot because all the values for approve are either at the top (=1) or at the bottom (=0). The best fitting regression line does not visually appear to describe the behavior of the values, but it still is chosen to minimize the average squared vertical distance between all the observations and the predicted value on the line.

The strange look of the scatter plot is telling as to how well the model predicts the data. You can see that the model fails to predict very well the many number of unapproved loans (`approve`=0) with values of `loanprc` between 0 and 150. While all of these loans were not approved, the linear model predicts a probability for approval between 60% and 100%.

The negative slope of the line is indicative that an increase in the size of the loan relative to the property price leads to a decrease in the probability that the loan is accepted. The magnitude of the slope indicates how much the approval probability decreases for each 1 percentage point increase in the size of the loan relative to the property price.

3.3 Predicting marginal effects

Since the average of the binary outcome variable is equal to a probability, the predicted value from the regression is a prediction for the *probability that someone is approved for a loan*.

Since the regression line is sloping downwards for `loanprc`, we see that as an applicant's loan amount relative to the property price increases, the probability that he/she is approved for a loan *decreases.*

The coefficient on `loanprc` is the estimated **marginal effect** of `loanprc` on the *probability* that the outcome variable is equal to 1. With a coefficient equal to -0.0019, our model predicts that for every 1 percentage point increase in housing expenses relative to income, the probability that the applicant is approved for a mortgage loan decreases by 0.19%.

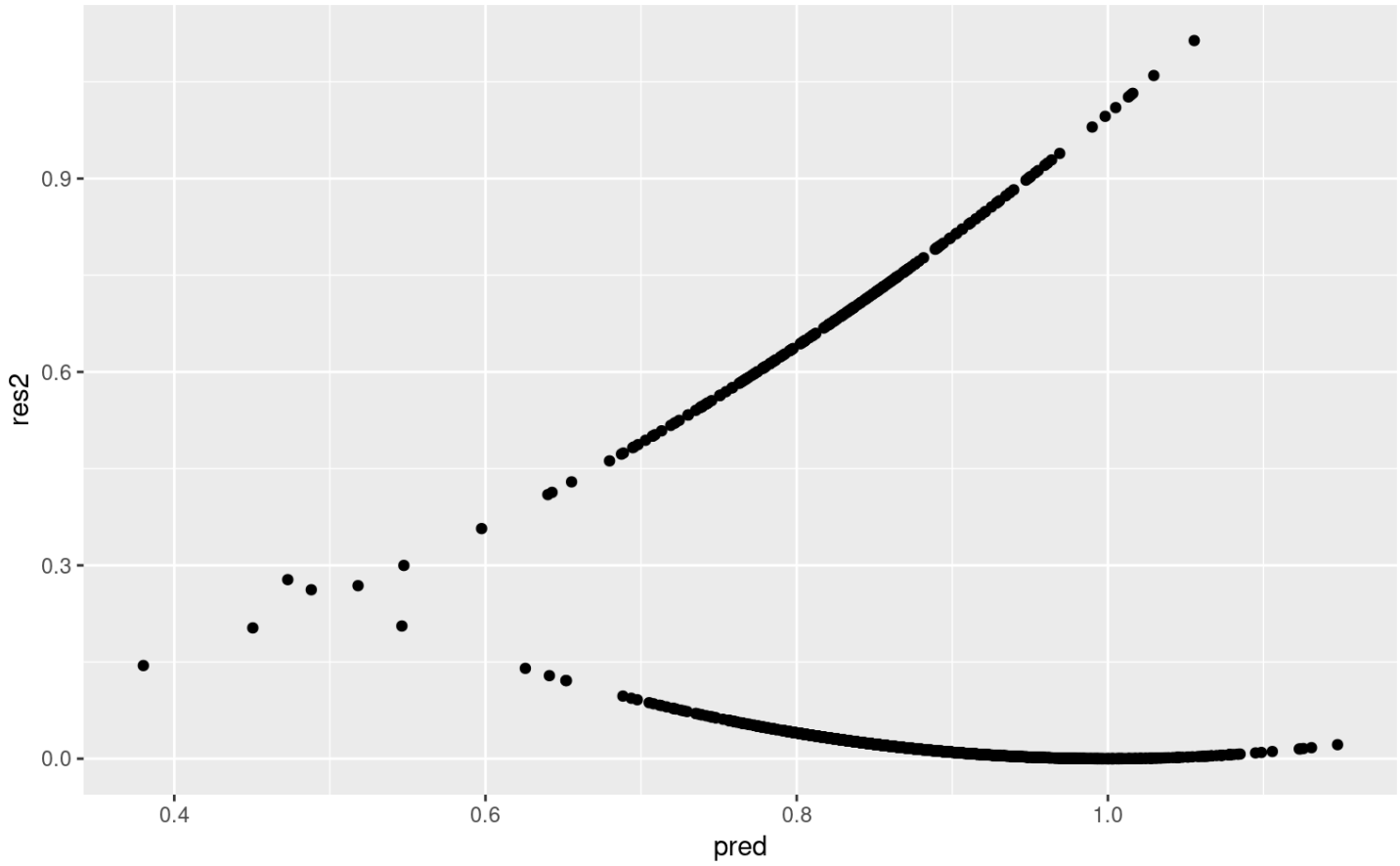
4 Heteroskedasticity

All linear probability models have heteroskedasticity. Because all of the actual values for y_i yi are either equal to 0 or 1, but the predicted values are probabilities anywhere between 0 and 1 (and sometimes even greater or smaller), the size of the residuals grow or shrink as the predicted values grow or shrink.

4.1 Visualizing Heteroskedasticity

Let us plot the predicted values against the squared residuals to see this:

```
df.lmapp.results <- data.frame(pred=lmapp$fitted.values, res2=lmapp$residuals^2)
ggplot(data=df.lmapp.results, mapping=aes(x=pred, y=res2)) + geom_point()
```



You can see that as the predicted probability that a loan is approved (the x-axis) increases, the estimate of the variance increases for some observations and decreases for some others.

4.2 Correcting for Heteroskedasticity

In order to conduct hypothesis tests and confidence intervals for the marginal effects an explanatory variable has on the outcome variable, we must first correct for heteroskedasticity. We can use the White estimator for correcting heteroskedasticity.

We compute the White heteroskedastic variance/covariance matrix for the coefficients with the call to `vcovHC` (which stands for Variance / Covariance Heteroskedastic Consistent):

```
vv <- vcovHC(lmapp, type="HC1")
```

The first parameter in the call above is our original output from our call to `lm()` above, and the second parameter `type="HC1"` tells the function to use the White correction.

Then we call `coeftest()` to use this estimate for the variance / covariance to properly compute our standard errors, t-statistics, and p-values for the coefficients.

```
coeftest(lmapp, vcov = vv)
```

```
##
## t test of coefficients:
##
##                Estimate  Std. Error t value  Pr(>|t|)
## (Intercept)  1.2402e+00  4.5220e-02 27.4251 < 2.2e-16 ***
## loanprc     -1.9267e-03  4.0644e-04 -4.7404 2.303e-06 ***
## loaninc     -4.6765e-05  7.5451e-05 -0.6198 0.5354670
## obrat       -5.9063e-03  1.2134e-03 -4.8677 1.230e-06 ***
## mortno       5.3579e-02  1.4884e-02  3.5999 0.0003271 ***
## unem        -8.6279e-03  4.0353e-03 -2.1381 0.0326438 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Suppose we wish to test the hypothesis that a higher loan value relative to the property price leads to a decrease in the probability that a loan application is accepted. The null and alternative hypotheses are given by,

$$H_0 : \beta_{loanprc} = 0$$
H0:βloanprc=0

$$H_0 : \beta_{loanprc} < 0$$
H0:βloanprc<0

The coefficient is negative (-0.0019) and the p-value in the output is equal to 0.000. This is the p-value for a two-tailed test. The p-value for a one-tailed test is half that amount, or 0.000. Since 0.000 < 0.05, we reject the null hypothesis and conclude that we have statistical evidence that, given the estimated effects of all the other explanatory variables in the model, an increase in the value of the loan relative to the property price leads to a decrease in the probability a loan is approved.

5 Problems using the Linear Probability Model

There are some problems using a binary dependent variable in a regression.

There is heteroskedasticity. But that’s OK, we know how to correct for it.

A *linear* model for a *probability* will eventually be wrong for *probabilities* which are by definition bounded between 0 and 1. Linear equations (i.e. straight lines) have no bounds. They continue eventually upward to positive infinity in one direction, and negative infinity in the other direction. *It is possible* for the linear probability model to predict probabilities greater than 1 and less than 0.

Use caution when the predicted values are near 0 and 1. It is useful to examine the predicted values from your regression to see if any are near these boundaries. In the example above, all the predicted values are between 0.7 and 0.95, so fortunately our regression equation is not making any mathematically impossible predictions.

Also, be cautious when using the regression equation to make predictions outside of the sample. The predicted values in your regression may have all fallen between 0 and 1, but maybe a predicted value will move outside the range.

The error term is not normal. When it is, then with small or large sample sizes, the sampling distribution of your coefficient estimates and predicted values are also normal.

While the residuals and the error term are never normal, *with a large enough sample size*, the central limit theorem does deliver normal distributions for the coefficient estimates and the predicted values. This problem that the error term is not normal, is really only a problem with small samples.