**Linear Regression**

Yan Deng, yd256

**1. Methodology and data processing**

1.1 Linear regression

Given a data set of n statistical units, a linear regression model assumes that the relationship between the dependent variable and the p-vector of regressors is linear. This relationship is modeled through a disturbance term or error variable — an unobserved random variable that adds noise to the linear relationship between the dependent variable and regressors. Linear regression model takes the form .

In linear regression, we assume that first, are mutually independent; second, are independent of ; are normally distributed; have a constant variance. To check whether these assumptions are true, we must look at the residuals. Figure 1 shows the residuals of the linear regression of the city’s revenue. From Figure 1, we noticed that the residuals do not meet the assumptions. Therefore, linear regression without any transformation is not a good model for the data. We take the Linear regression Models with Logarithmic Transformations.

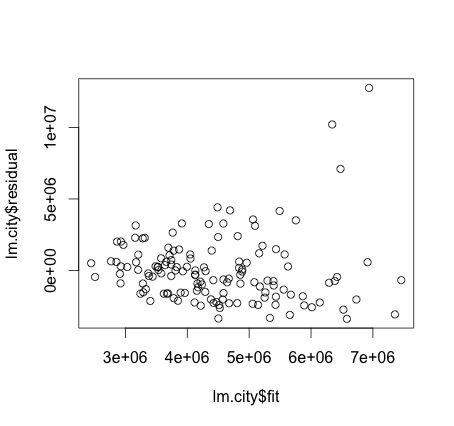


Figure 1.

1.2 Linear regression Models with Logarithmic Transformations

Logarithmically transforming variables in a regression model is a very common way to handle situations, where a non-linear relationship exists between the independent and dependent variables. Using the logarithm of one or more variables instead of the unlogged form makes the effective relationship nonlinear, while still preserving the linear model.

Log-linear model:

In this model, we use log-linear model to fit the data, as shown in Figure 2. In the log-linear model, the literal interpretation of the estimated coefficient β is that a one-unit increase in X will produce an expected increase in log Y of β units. In terms of Y itself, this means that the expected value of Y is multiplied by. From figure 2, we observed that the residuals meet the assumptions of linear regression. In linear regression, we assume that first, are mutually independent; second, are independent of ; are normally distributed; have a constant variance.

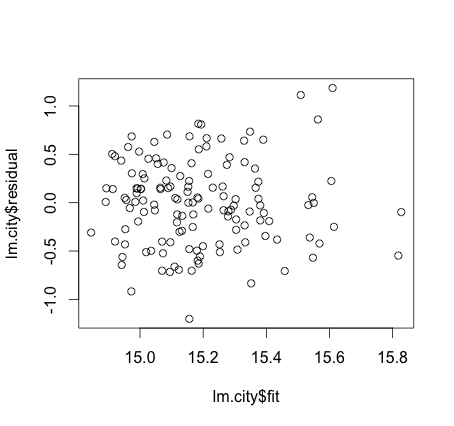


Figure 2. Log-linear model

**2. Implementation in R**

2.1 Ordinary least square

Ordinary least square is the simplest and thus most common estimator. It is conceptually simple and computationally straightforward. OLS estimates are commonly used to analyze both [experimental](http://en.wikipedia.org/wiki/Experiment) and [observational](http://en.wikipedia.org/wiki/Observational_study) data. The OLS method minimizes the sum of squared [residuals](http://en.wikipedia.org/wiki/Errors_and_residuals_in_statistics), and leads to a closed-form expression for the estimated value of the unknown parameter β:

#close form for linear regression

TP=data$Type

T1=rep(0,length(TP))

T2=rep(0,length(TP))

for (i in 1:length(data$Type)){

if (TP[i]=="IL") {

T1[i]=1

}

else if (TP[i]=="FC")

{T2[i]=1}

}

CG=data$City.Group

C=rep(0,length(CG))

for (i in 1:length(data$City.Group)){

if (CG[i]=="Big Cities") {

C[i]=0

}

else{C[i]=1}

}

lr=function(X,y){

beta=solve(t(X)%\*%X, t(X)%\*%y)

return(beta)

}

a=rep(1,length(data$revenue))

X=cbind(a,data$Open.Date,C,T1,T2, data$V1,data$V2, data$V3)

y=log(data$revenue)

lm=lr(X,y)

Table 2-1. Description of Selected Variables

|  |  |
| --- | --- |
| Variable | Description |
| Open Date | Opening date for a restaurant |
| City | City that the restaurant is in |
| City Group | Type of the city: big cities or other |
| Type | Type of the restaurant: FC (Food Court), IL (Inline), DT (Drive Thru) |

|  |  |
| --- | --- |
| Variable | Indicator |
| City Group | , if the city is a big city; , otherwise. |
| Type | , if the restaurant is Food Court; , otherwise.  , if the restaurant is Inline; , otherwise. |

2.2 Gradient descent algorithm linear regression

Since it’s not necessary that all matrixes are reversible, the method of close form cannot work for any case. Another method to implement the linear regression is Gradient descent algorithm linear regression. At a theoretical level, gradient descent is an algorithm that minimizes functions. Given a function defined by a set of parameters, gradient descent starts with an initial set of parameter values and iteratively moves toward a set of parameter values that minimize the function. This iterative minimization is achieved using calculus, taking steps in the negative direction of the function gradient.

norm\_vec <- function(x) sqrt(sum(x^2))

derivative=function(X,y,beta){

beta=rep(0,8)

step=0.01

a=rep(1000,dim(X)[2])

x\_old=beta+rep(10,8)

x\_new=beta

while (norm\_vec(a)>20){

a=as.numeric(-t(X)%\*%y+t(X)%\*%X%\*%x\_new)/length(y)

print(norm\_vec(y-X%\*%x\_new))

x\_old=x\_new

x\_new=x\_old-step\*a

}

beta=x\_new

return(beta)

}

constant=rep(1,length(data$revenue))

X=as.matrix(cbind(constant,data$Open.Date, C, T1, T2, data$V1,data$V2, data$V3))

y=log(data$revenue)

estimate=derivative(X,y,rep(0,8))

lm2=derivative(X,y,rep(0,dim(X)[2]))