PHP2511 Homework 3

Kuan-Min Lee

2023-03-16

Test-Train Split and Initial Predictor Selection:

In this section, the dataset will be conducted with some preprocssing procedures to better fit the linear regression model.

First, the variable “black” is filtered out from the original dataset.

```
data_test <- read.csv("kidney_small.csv",header=TRUE,sep=",") # reading datasheet
data_test <- subset(data_test,select=-c(black)) # filter out black variable
```

After filtering out the variable “black”, the following code is inserted to create a train and test set from the original data.

```
set.seed(1)
data_test$id <- 1:nrow(data_test)
kidney_train <- data_test %>% dplyr::sample_frac(0.75)
kidney_test <- dplyr::anti_join(data_test, kidney_train, by = 'id')
```

Initial Transformations

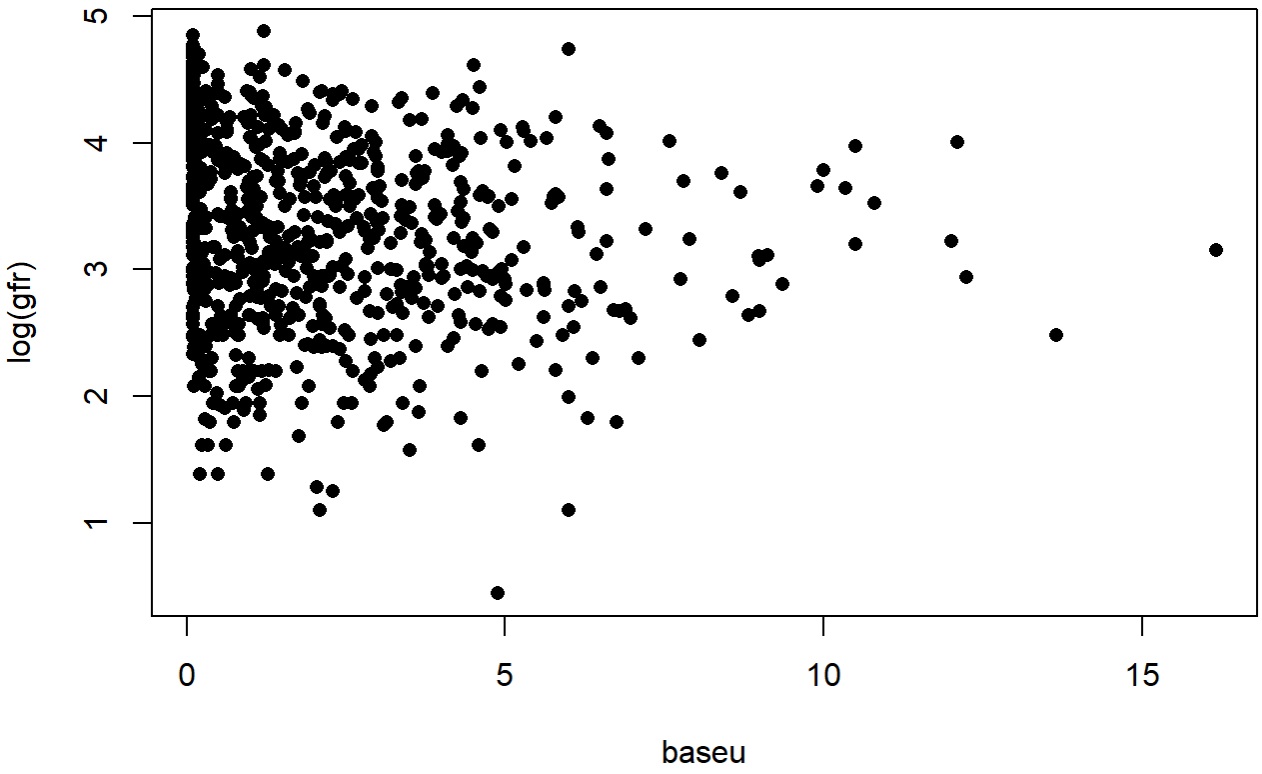
The main goal of this section is to conduct the transformation to the variables of the linear model that is going to be fitted.

The transformation is conducted on the continuous variables of the models, and in this models, the continuous variables are the variable “baseu”, “bascre”, and “gfr”. Therefore, several transformations will be conducted and different combination of transformed variables will be utilized into the model to view the fitting results.

Plotting for log(gfr) vs baseu

```
kidney_train$log_gfr<-log(kidney_train$gfr) # conduct log transform on training dataset
kidney_test$log_gfr<-log(kidney_test$gfr) # conduct log transform on training dataset

plot(kidney_train$log_gfr~kidney_train$baseu,ylab=c("log(gfr)"),xlab=c("baseu"),pch=16) # create a plot of log(gfr) vs baseu
```

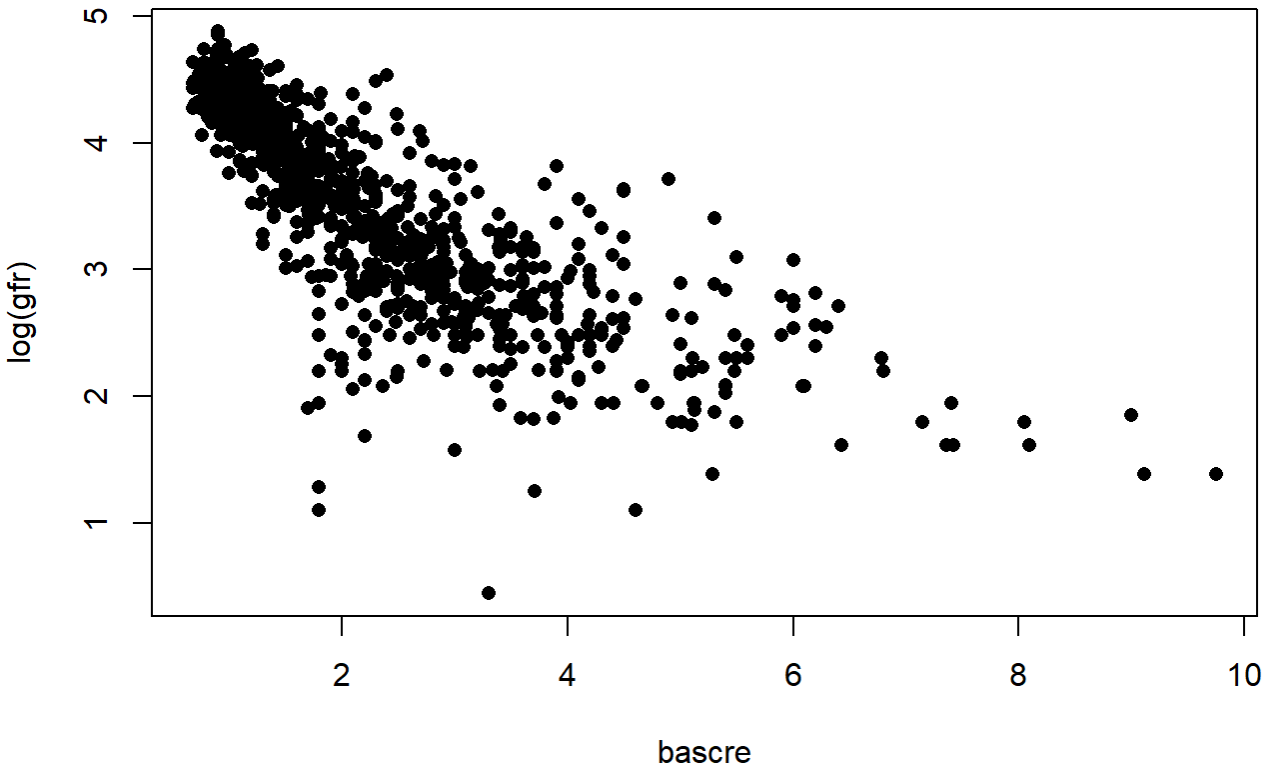


From the above graph, tit's really hard to observe the relationship between the variable of log(gfr) and baseu since there's no obvious linear or nonlinear trend of the line between these two variables.

log(gfr) vs bascre:

In this portion, a scatter plot of gfr vs sbase variables is created to view the potential relationship between these two variables.

```
plot(kidney_train$log_gfr~kidney_train$bascre,ylab=c("log(gfr)"),xlab=c("bascre"),pch=16) # crea
te a plot of log(gfr) vs bascre
```

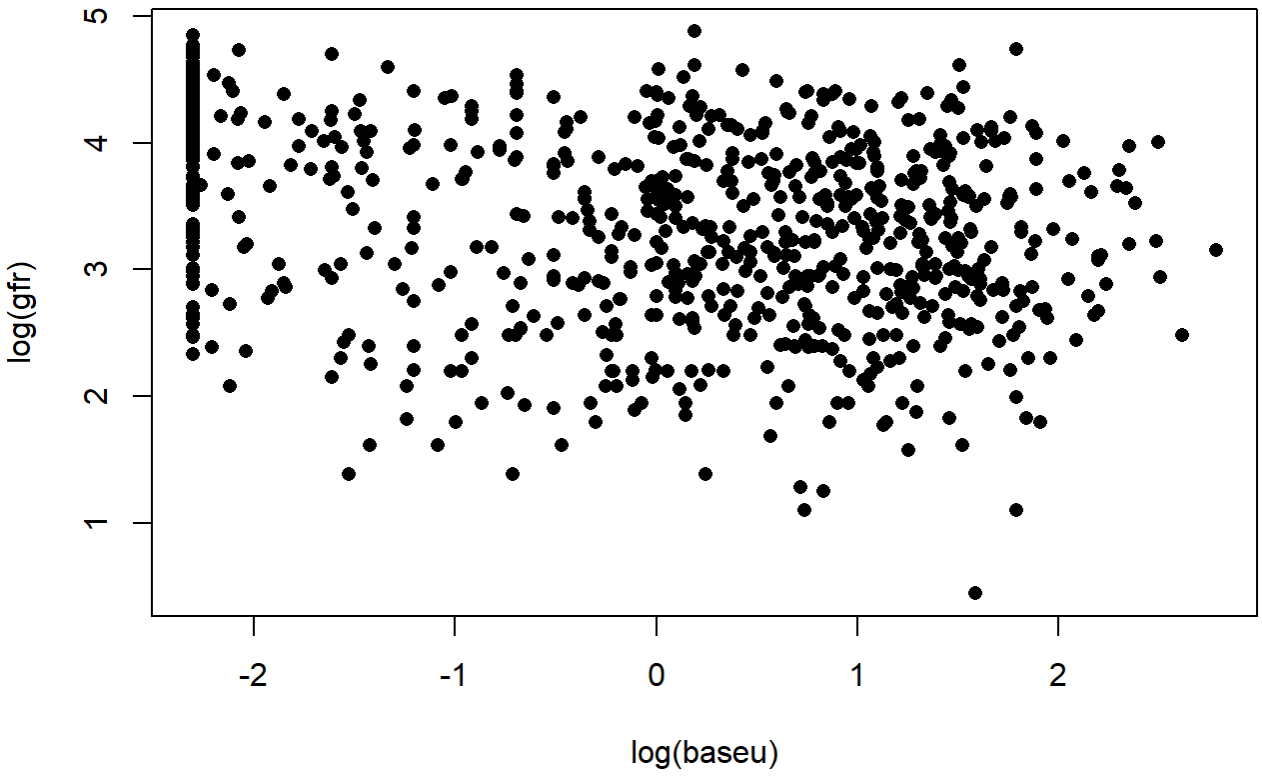


From the above plot, there seems to be a decline linear relationship between the variable log(gfr) and bascre based on the dot distributions on the graph.

log(gfr) vs log(baseu):

```
# conduct log transform on variable baseu
kidney_train$log_baseu<-log(kidney_train$baseu)
kidney_test$log_baseu<-log(kidney_test$baseu)

plot(kidney_train$log_gfr~kidney_train$log_baseu,ylab=c("log(gfr)"),xlab=c("log(baseu)"),pch=16)
# create a plot of log(gfr) vs log(baseu)
```



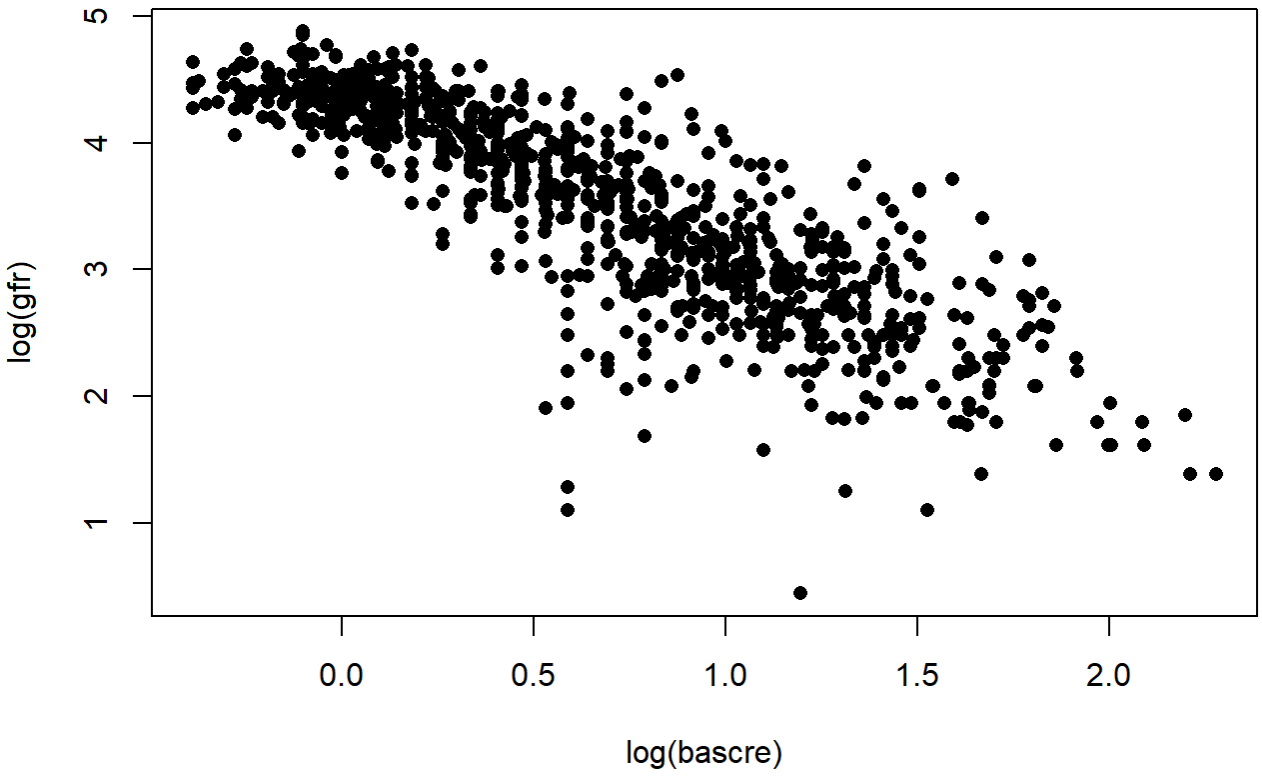
From the above plot, there seems to be a linear relationship that represents a horizontal line between the variable log(gfr) and log(baseu). From this observation, there's still not obvious linear or nonlinear relationship between the two variables.

log(gfr) vs log(bascre):

In this portion, a scatter plot of gfr vs baseu variables is created to view the potential relationship between these two variables.

```
# conduct log transformation on variable bascre
kidney_train$log_bascre<-log(kidney_train$bascre)
kidney_test$log_bascre<-log(kidney_test$bascre)

plot(kidney_train$log_gfr~kidney_train$log_bascre,ylab=c("log(gfr)"),xlab=c("log(bascre)"),pch=1
6) # create a plot of log(gfr) vs log(bascre)
```



From the above plot, the relationship between the variable log(gfr) and log(bascre) becomes more linear compared to the one with bascre only. Therefore, from this prospect, the log transformation works on improving the linearity of the relationship between the two variables.

Model Fitting Testing

In this portion, several linear model will be built to test the fitting results.

Model 1: log(gfr) ~ baseu + bascre + sbase + dbase + age + male

```
model1 <- lm(log_gfr~baseu+bascre+sbase+dbase+age+male,data=kidney_train)
summary(model1)
```

```
##
## Call:
## lm(formula = log_gfr ~ baseu + bascre + sbase + dbase + age +
##     male, data = kidney_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.54112 -0.22070  0.05663  0.28253  1.48363
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.335e+00  1.436e-01  30.192  < 2e-16 ***
## baseu       -4.797e-02  7.473e-03  -6.419 2.18e-10 ***
## bascre      -4.439e-01  1.176e-02 -37.759  < 2e-16 ***
## sbase       -1.566e-04  1.050e-03  -0.149    0.882
## dbase        1.676e-03  1.956e-03   0.857    0.392
## age          4.311e-05  1.342e-03   0.032    0.974
## male         1.984e-01  3.251e-02   6.101 1.54e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4664 on 930 degrees of freedom
## Multiple R-squared:  0.655,  Adjusted R-squared:  0.6528
## F-statistic: 294.3 on 6 and 930 DF,  p-value: < 2.2e-16
```

Model 2: log(gfr) ~ log(baseu) + bascre + sbase + dbase + age + male

```
model2 <- lm(log_gfr~log_baseu+bascre+sbase+dbase+age+male,data=kidney_train)
summary(model2)
```

```
##
## Call:
## lm(formula = log_gfr ~ log_baseu + bascre + sbase + dbase + age +
##     male, data = kidney_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.56754 -0.20505  0.04581  0.26411  1.34071
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.284e+00  1.381e-01  31.024  < 2e-16 ***
## log_baseu   -1.164e-01  1.143e-02 -10.186  < 2e-16 ***
## bascre      -4.177e-01  1.187e-02 -35.174  < 2e-16 ***
## sbase       -7.294e-07  1.017e-03  -0.001   0.9994
## dbase        1.339e-03  1.896e-03   0.706   0.4801
## age         -2.660e-03  1.347e-03  -1.975   0.0485 *
## male         2.221e-01  3.159e-02   7.030 3.99e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4521 on 930 degrees of freedom
## Multiple R-squared:  0.6759, Adjusted R-squared:  0.6738
## F-statistic: 323.3 on 6 and 930 DF,  p-value: < 2.2e-16
```

Model 3: log(gfr) ~ baseu + log(bascre) + sbase + dbase + age + male

```
model3 <- lm(log_gfr~baseu+log_bascre+sbase+dbase+age+male,data=kidney_train)
summary(model3)
```

```
##
## Call:
## lm(formula = log_gfr ~ baseu + log_bascre + sbase + dbase + age +
##     male, data = kidney_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4849 -0.1822  0.0286  0.2094  1.3739
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.3696499  0.1234790  35.388  < 2e-16 ***
## baseu       -0.0244318  0.0065525  -3.729 0.000204 ***
## log_bascre  -1.2386460  0.0262961 -47.104  < 2e-16 ***
## sbase       -0.0007354  0.0009081  -0.810 0.418266
## dbase        0.0008195  0.0016923   0.484 0.628324
## age         -0.0023200  0.0011668  -1.988 0.047074 *
## male         0.2514035  0.0281664   8.926  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4034 on 930 degrees of freedom
## Multiple R-squared:  0.7419, Adjusted R-squared:  0.7403
## F-statistic: 445.6 on 6 and 930 DF,  p-value: < 2.2e-16
```

Model 4: log(gfr) ~ log(baseu) + log(bascre) + sbase + dbase + age + male

```
model4 <- lm(log_gfr~log_baseu+log_bascre+sbase+dbase+age+male,data=kidney_train)
summary(model4)
```

```
##
## Call:
## lm(formula = log_gfr ~ log_baseu + log_bascre + sbase + dbase +
##     age + male, data = kidney_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.50432 -0.19390  0.02467  0.20948  1.42781
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.3280185  0.1218376  35.523  < 2e-16 ***
## log_baseu   -0.0545783  0.0105899  -5.154 3.12e-07 ***
## log_bascre  -1.2002524  0.0280727 -42.755  < 2e-16 ***
## sbase       -0.0006809  0.0009016  -0.755  0.45032
## dbase        0.0007646  0.0016806   0.455  0.64925
## age         -0.0033517  0.0011945  -2.806  0.00512 **
## male         0.2585792  0.0280118   9.231  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4007 on 930 degrees of freedom
## Multiple R-squared:  0.7453, Adjusted R-squared:  0.7437
## F-statistic: 453.6 on 6 and 930 DF,  p-value: < 2.2e-16
```

```
# compute AIC for different models
models <- list(model1,model2,model3,model4)
mod.names <- c('no_log','log_baseu','log_bascre','log_both')
aictab(cand.set=models,modnames=mod.names)
```

	Modnames	K	AICc	Delta_AICc	ModelLik	AICcWt	LL	Cum.Wt
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
4	log_both	8	954.4624	0.00000	1.000000e+00	9.980571e-01	-469.1536	0.9980571
3	log_bascre	8	966.9456	12.48327	1.946672e-03	1.942890e-03	-475.3952	1.0000000
2	log_baseu	8	1180.3427	225.88036	8.926928e-50	8.909584e-50	-582.0938	1.0000000
1	no_log	8	1238.8174	284.35502	1.790987e-62	1.787508e-62	-611.3311	1.0000000

4 rows

From the above fitting results, from the observation of the summary table, model 4 fits the data the best since it has the best overall residual standard error with the lowest value, and with the highest adjusted R value and the lowest AIC scores.

And also,for respective log transformation on the two variables, the models that have the log-transformed variable perform better compared to the ones with the original variables. Therefore, from there, the conclusion that the log transformation on these two variables does work can be made.

Variables Selection:

In this section, first, the effect of the drop of sbase and dbase variables are examined.

To test it, the best model from the previous section, model 4, is picked as the sample and the other moels of droping sbase, dbase, and droping both are all built and tested.

```
model4_1 <- lm(log_gfr~log_baseu+log_bascre+dbase+age+male,data=kidney_train) # drop sbase
model4_2 <- lm(log_gfr~log_baseu+log_bascre+sbase+age+male,data=kidney_train) # drop dbase
model4_3 <- lm(log_gfr~log_baseu+log_bascre+age+male,data=kidney_train) # drop both
# compute AIC for different models
models <- list(model4,model4_1,model4_2,model4_3)
mod.names <- c('no_drop','drop_sbase','drop_dbase','drop_both')
aictab(cand.set=models,modnames=mod.names)
```

Modnames <chr>	K <dbl>	AICc <dbl>	Delta_AICc <dbl>	ModelLik <dbl>	AICcWt <dbl>	LL <dbl>	Cum.Wt <dbl>
4 drop_both	6	950.9882	0.000000	1.0000000	0.50505374	-469.4489	0.5050537
3 drop_dbase	7	952.6363	1.648102	0.4386511	0.22154237	-469.2579	0.7265961
2 drop_sbase	7	953.0022	2.014039	0.3653061	0.18449922	-469.4408	0.9110953
1 no_drop	8	954.4624	3.474200	0.1760301	0.08890466	-469.1536	1.0000000

4 rows

From the above AIC values, the conclustion that droping both achieve the lowest AIC. Therefore, the decision of dropping both variables can be validated.

Besides the AIC testing, and hypothesis testing on this decision can also be conducted on the effect of this decision.

The null hypothesis of such a testing model can be that the error between the two models are not statistically significant different from each other, and the alternative hypothesis is the opposite.

To test it, we begin with the model dropping sbase.

```
res_mod4<-model4$residuals # the residuals of model 4
res_mod4_1<-model4_1$residuals # the residuals of model 4_1
t.test(res_mod4,res_mod4_1,paired=TRUE)
```

```
##
##  Paired t-test
##
## data:  res_mod4 and res_mod4_1
## t = 1.6183e-14, df = 936, p-value = 1
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -0.0006341742  0.0006341742
## sample estimates:
## mean difference
##    5.229483e-18
```

From the previous observation, we first know that the residuals of model 4_1 is smaller than the residual of model 4. By further conducting the t test, it shows that the residual between the two are statistically significant from each other. Therefore, dropping sbase is necessary.

For testing the effect of dbase, the residual of the previous part and the one with both variables dropped are compared. Therefore, in this round, the t test will be conducted on these two models.

```
res_mod4_3<-model4_3$residuals # the residuals of model 4_3
t.test(res_mod4_3,res_mod4_1,paired=TRUE)
```

```
##
##  Paired t-test
##
## data:  res_mod4_3 and res_mod4_1
## t = -4.9263e-16, df = 936, p-value = 1
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -0.0001065075  0.0001065075
## sample estimates:
## mean difference
##   -2.673589e-20
```

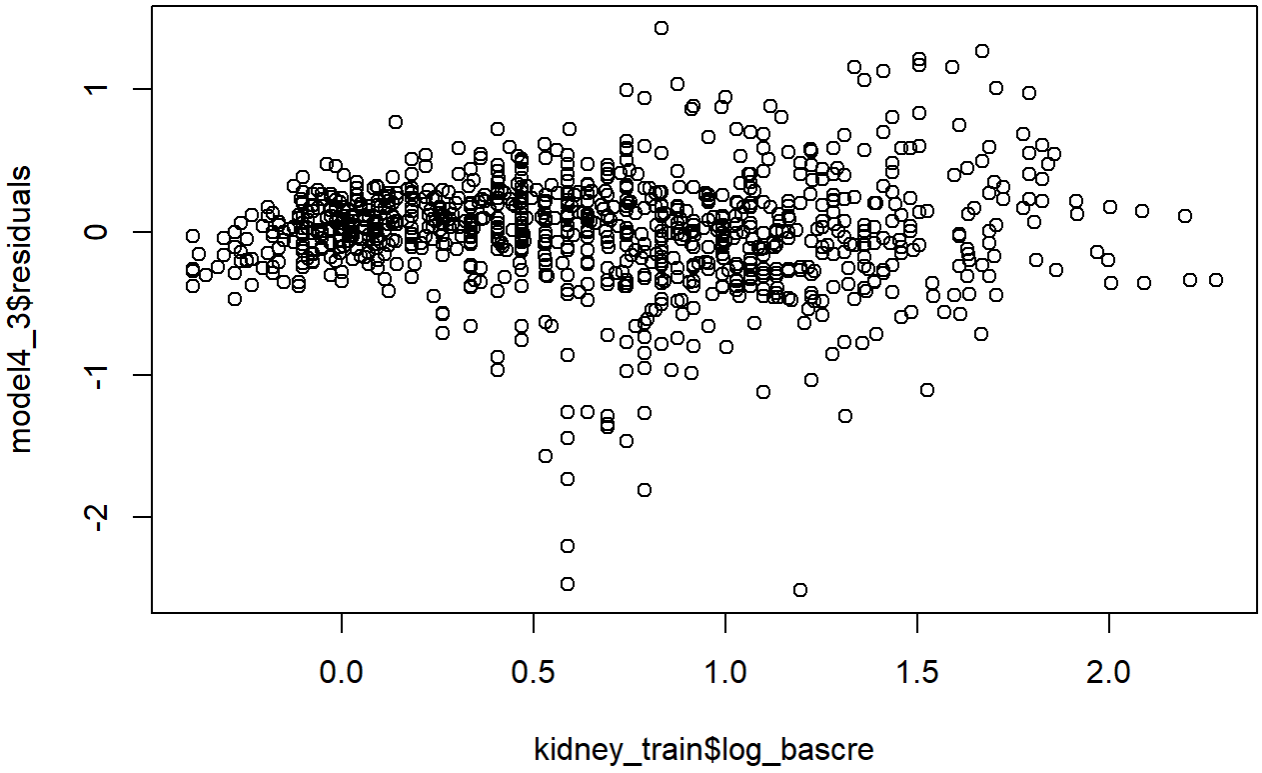
From the previous section, it can also be seen taht the residual of model 4_3 is smaller than model 4_1. By conducting t test, it can be seen that the residuals between these two models are statistically significant from each other. Therefore, it can be seen that dropping dbase is also necessary.

Adding Polynomial Transformation:

In this section of the work, the consideration of adding polynomial transformation on variable regarding serum creatinine is considered.

To start our observation, the residuals of the model vs the log(bascre) is created.

```
plot(kidney_train$log_bascre,model4_3$residuals)
```



From

the above plot, it seems that the plot follows like a up curvature parabola. Therefore, a polynoimal transformation for degree of 2 is decided on log(bascre) variable.

```
model4_3_2<-lm(log_gfr~log_baseu+age+male+poly(log_bascre,2),data=kidney_train)
summary(model4_3_2)
```

```
##
## Call:
## lm(formula = log_gfr ~ log_baseu + age + male + poly(log_bascre,
##     2), data = kidney_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5144 -0.1946  0.0272  0.2139  1.4507
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)             3.507781   0.060303  58.170  < 2e-16 ***
## log_baseu              -0.049618   0.010932  -4.539  6.4e-06 ***
## age                    -0.003828   0.001094  -3.498  0.00049 ***
## male                    0.269687   0.027952   9.648  < 2e-16 ***
## poly(log_bascre, 2)1 -20.193688   0.471475 -42.831  < 2e-16 ***
## poly(log_bascre, 2)2   0.812393   0.426544   1.905  0.05714 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3999 on 931 degrees of freedom
## Multiple R-squared:  0.7462, Adjusted R-squared:  0.7448
## F-statistic: 547.3 on 5 and 931 DF,  p-value: < 2.2e-16
```

Evaluating Final Model:

This is the final portion of this work.

The first work will be to compute the MAE and RMSE for the final model that has been obtained from the previous section.

```
predictions_trains<-model4_3_2 %>% predict(kidney_train) # compute the predictions of training
predictions_tests<-model4_3_2 %>% predict(kidney_test) # compute the predictions of testing
RMSE_trains<-rmse(predictions_trains,kidney_train$log_gfr) # compute the RMSE for training
RMSE_test<-rmse(predictions_tests,kidney_test$log_gfr) # compute the RMSR for testing
RMSE_trains
```

```
## [1] 0.3985708
```

```
RMSE_test
```

```
## [1] 0.4475714
```

```
MAE_trains<-mae(predictions_trains,kidney_train$log_gfr) # compute the RMSE for training
MAE_test<-mae(predictions_tests,kidney_test$log_gfr) # compute the RMSR for testing
MAE_trains
```

```
## [1] 0.276842
```

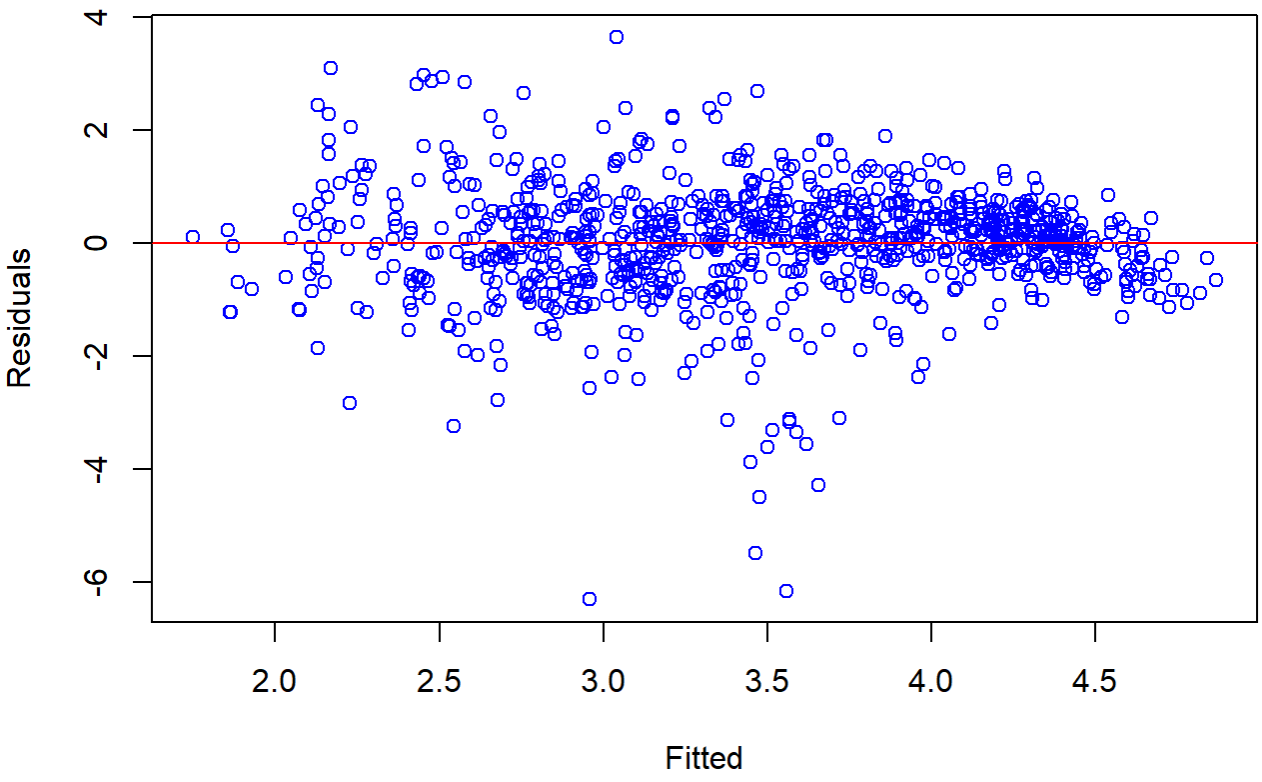
```
MAE_test
```

```
## [1] 0.3001768
```

In the later part, a series of diagnostic plots for model fitting are shown.

First is the residual vs fitted values plot

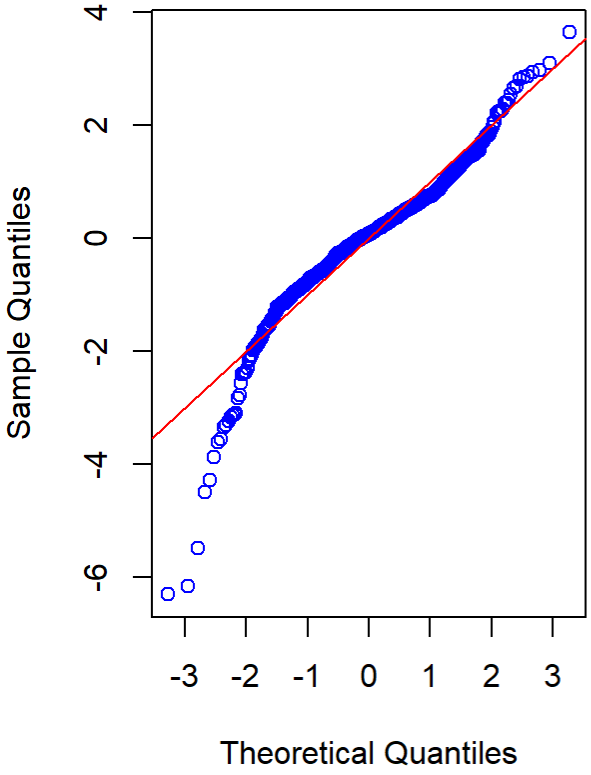
```
plot(fitted(model4_3_2), rstandard(model4_3_2),
xlab = "Fitted",
ylab = "Residuals", col = "blue")
abline(h=0, col = "red")
```

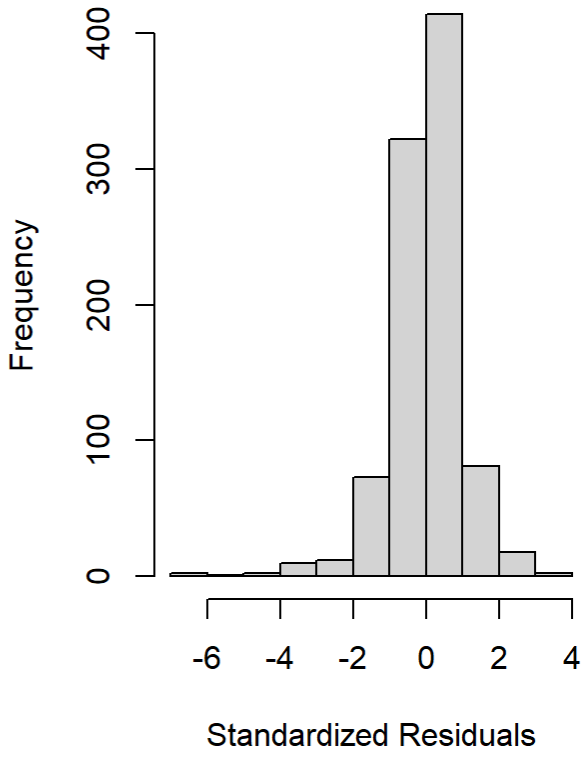


From the plot shown above, it can be seen that the residual values doesn't vary much with the increase of the fitted value. Therefore, no matter how large the fitted value is, the residual stay relatively stable.

Next, it's the qq plot and the histogram for the fitting model.

```
par (mfrow = c (1,2))
qqnorm(rstandard(model4_3_2), main = "", col = "blue")
abline(0,1, col = "red")
hist (rstandard(model4_3_2), main = "", xlab = "Standardized Residuals")
```



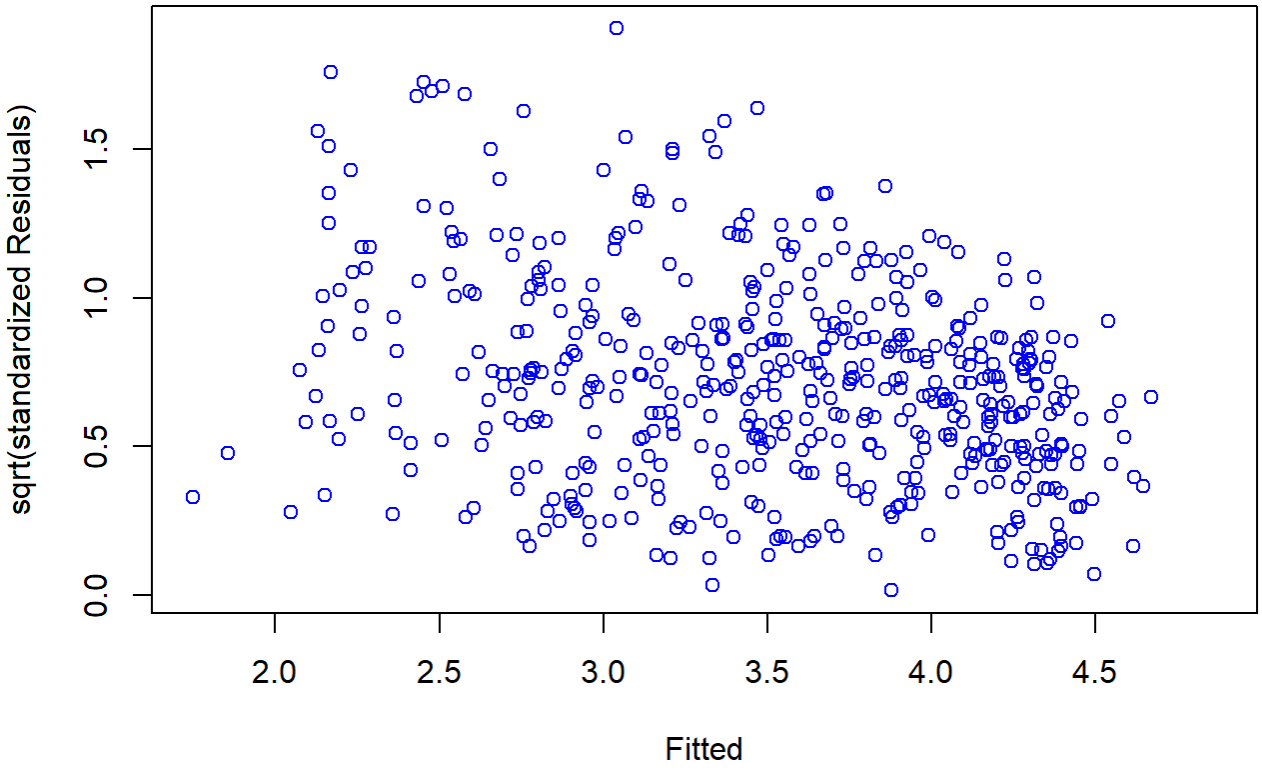


From the above plots, it can be seen that the theoretical quantiles and sample quantiles follow approximately a linear relationship, which shows that the fitting works fine in predicting the response variables with the given predictor variables. As for the distribution of the residuals, it follows approximately like a normal distribution with mean equal to 0.

Last, it's the scale location plot

```
plot(fitted(model4_3_2), sqrt(rstandard(model4_3_2)),
xlab = "Fitted",
ylab = "sqrt(standardized Residuals)", col = "blue")
```

```
## Warning in sqrt(rstandard(model4_3_2)): NaNs produced
```



From the above plot, it can be seen that the residuals between each fitted values are approximately eqaul for each fitted values.

To sum up the diagnostic conclusion for the model, based on the homoscedasticity testing, which the model has approximately equal variance for each fitted values and normality testing, which the model shows a linear relationship between the sample and theoretical quantiles, the model can predice a reasonable gfr value from a practical data from the outer world.

Therefore, the above model should be robust enough to conduct the prediction.