NMAI061-22-EX5

Matej Nemec

Visualization of multi-dimensional data and Principal Component Analysis

1)Select appropriate data and try to select predictor variables (potentially with applied transformations) in order for predicted values to fit the observed data as good as possible.

We decided to go with the Animals2 dataset which only has 2 variables and we will be trying to predict brain wight based on animals body weight. Right away it makes intuitive sense not expect a completely linear relationship. But let us test this assumption a bit.

```
library(robustbase)
library(sigmoid)
```

```
## Warning: package 'sigmoid' was built under R version 4.1.3
```

```
library(ggpubr)
```

```
## Loading required package: ggplot2
```

```
library(lmtest)
```

```
## Warning: package 'lmtest' was built under R version 4.1.3
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
data=Animals2
cor.test(data$body,data$brain)
```

```
##
##  Pearson’s product-moment correlation
##
## data:  data$body and data$brain
## t = 0.41062, df = 63, p-value = 0.6827
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.1946886  0.2918848
## sample estimates:
##        cor
## 0.05166368
```

As we can see when not trasforming the variables at all there seems to be no correlation. Because the data is on mamals what we would intuitively expect is that there is both a point under which the brain size doesn’t really decrease (or not nearly as much) with decreasing body size and at the same time we would expect brain size increase to plateau with very large body sizes. When we think about which functions could model such behavior, log, tanh and sigmoid come to mind. It would also make sense to test simply taking square roots of the body size.

```
cor.test(log(data$body),data$brain)
```

```
##
##  Pearson’s product-moment correlation
##
## data:  log(data$body) and data$brain
## t = 3.8923, df = 63, p-value = 0.0002423
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2200194 0.6178424
## sample estimates:
##       cor
## 0.4402914
```

```
cor.test(tanh(data$body),data$brain)
```

```
##
##  Pearson’s product-moment correlation
##
## data:  tanh(data$body) and data$brain
## t = 1.6345, df = 63, p-value = 0.1072
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.04439228  0.42469789
## sample estimates:
##       cor
## 0.2016906
```

```
cor.test(sigmoid(data$body),data$brain)
```

```
##
##  Pearson’s product-moment correlation
##
```

```
## data:  sigmoid(data$body) and data$brain
## t = 2.041, df = 63, p-value = 0.04545
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.005472015 0.464711155
## sample estimates:
##       cor
## 0.2490388
```

```
best=0
best_r=0
for(i in 1:10000){
  r=cor(data$body^(1/i),data$brain)
  if(r>best_r){
    best_r=r
    best=i
  }
}
print(paste0(best, '-th root of bodyweight has the best correlation with brain weight.'))
```

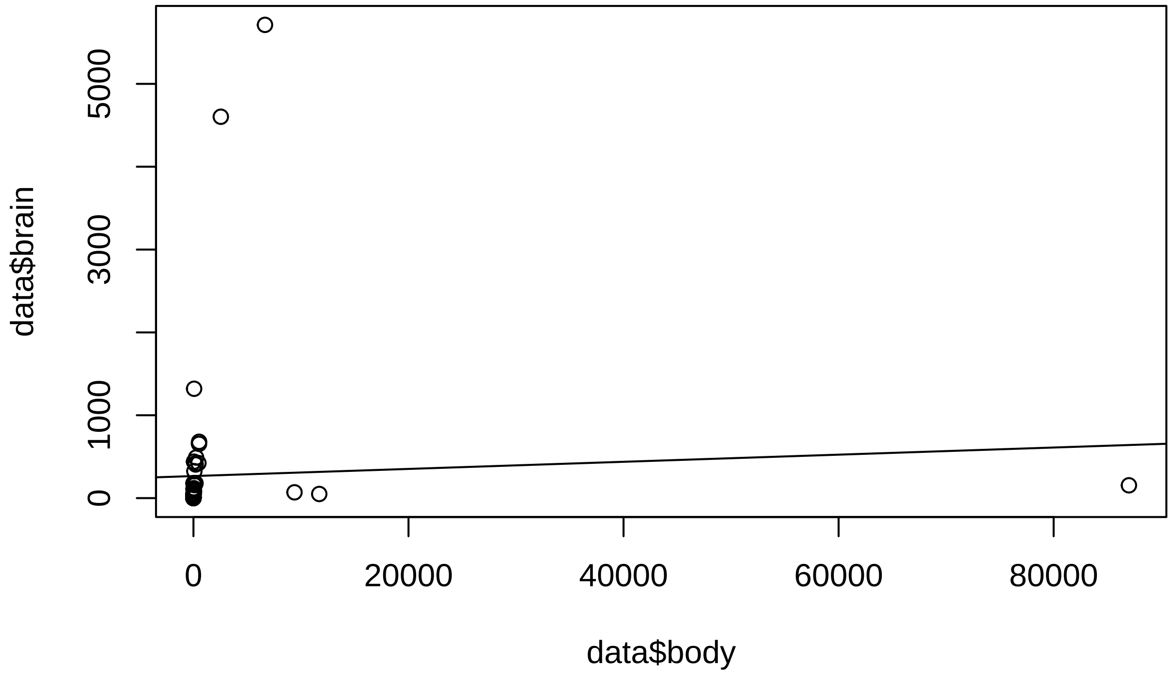
```
## [1] "10-th root of bodyweight has the best correlation with brain weight."
```

```
cor.test(data$body^(1/best),data$brain)
```

```
##
##  Pearson’s product-moment correlation
##
## data:  data$body^(1/best) and data$brain
## t = 4.1629, df = 63, p-value = 9.718e-05
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2487488 0.6362871
## sample estimates:
##       cor
## 0.4644688
```

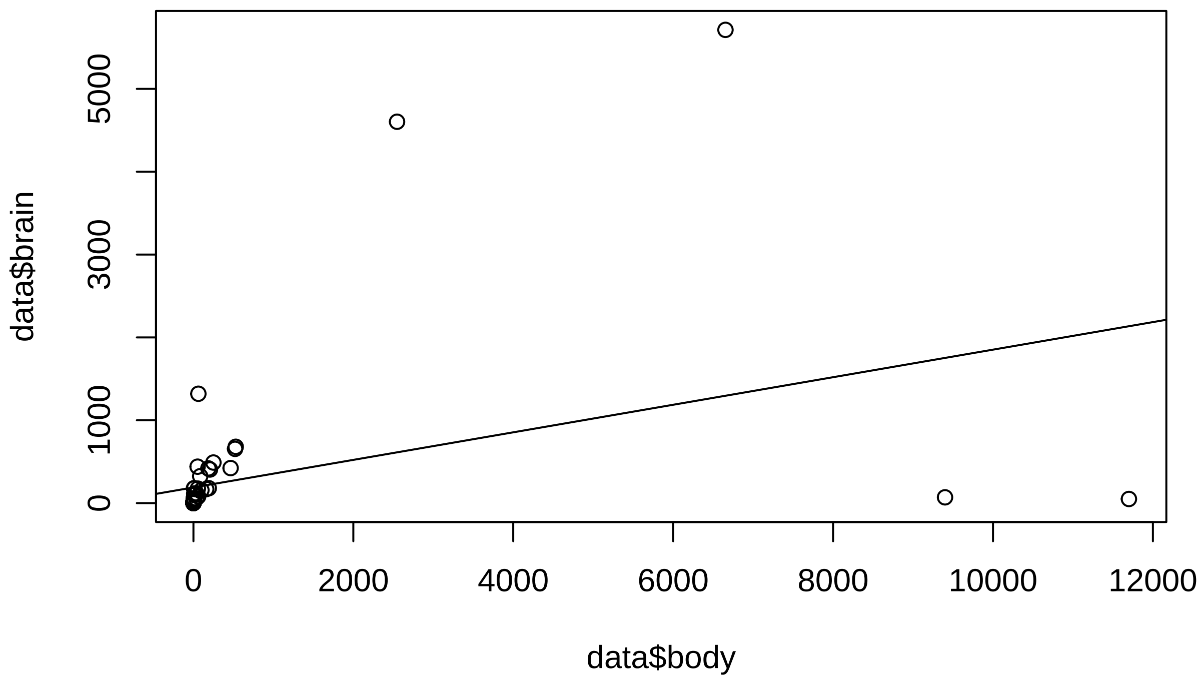
We can see that best correlation is achieved with 10-th root of bodyweight. It is still not very high though.
## 2)Visualize the data including fitted values from a selected linear model.

```
m1=lm(brain~body,data=data)
plot(data$body,data$brain)
abline(m1)
```



We can see that there is one extreme outlier in the data we should try and remove it to help our model.

```
data=data[data$body!=max(data$body),]
m2=lm(brain~body,data=data)
plot(data$body,data$brain)
abline(m2)
```



This seems better. Still not a great model. But sometimes best we can get is not that good. Especially with linear models.

3)Comment on coefficient statistical significance. Test the null hypothesis of all coeffiecients being equal to 0.

```
summary(m1)
```

```
##
## Call:
## lm(formula = brain ~ body, data = data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -486.3 -262.4 -249.3 -109.7 5417.0
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.663e+02  1.152e+02   2.313    0.024 *
## body        4.304e-03  1.048e-02   0.411    0.683
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 915.1 on 63 degrees of freedom
## Multiple R-squared:  0.002669,   Adjusted R-squared:  -0.01316
## F-statistic: 0.1686 on 1 and 63 DF,  p-value: 0.6827
```

```
summary(m2)
```

```
##
## Call:
## lm(formula = brain ~ body, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2085.1  -185.9  -173.4   -48.4  4416.1
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 189.30406  110.87790   1.707  0.09277 .
## body          0.16631    0.05329   3.121  0.00274 **
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 858.6 on 62 degrees of freedom
## Multiple R-squared:  0.1357, Adjusted R-squared:  0.1218
## F-statistic: 9.738 on 1 and 62 DF,  p-value: 0.00274
```

For the first model we have not reached statistical significance, however when we remove the outlier we do. While the coefficient is significant the R^2 is very poor Now we can try to fit our best transformation - $10-th$ root of bodyweight.

```
m3=lm(brain~I(body^(1/10)),data=data)
summary(m3)
```

```
##
## Call:
## lm(formula = brain ~ I(body^(1/10)), data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1660.8  -290.0   -90.6    93.4  4154.5
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -1083.0      290.5  -3.728 0.000421 ***
## I(body^(1/10))   1094.9      220.4   4.967 5.64e-06 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 781.2 on 62 degrees of freedom
## Multiple R-squared:  0.2846, Adjusted R-squared:  0.2731
## F-statistic: 24.67 on 1 and 62 DF,  p-value: 5.64e-06
```

It is better but still not very good. We perform anova.

```
anova(m3)
```

```
## Analysis of Variance Table
##
## Response: brain
##                 Df   Sum Sq  Mean Sq F value   Pr(>F)
## I(body^(1/10))  1 15054217 15054217  24.671 5.64e-06 ***
## Residuals      62 37833002   610210
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coef=summary(m3)$coefficients[2,1]
err=summary(m3)$coefficients[2,2]
coef + c(-1,1)*err*qt(0.975, 42)
```

```
## [1]  650.0367 1539.7532
```

We cannot reject the hypothesis that coefficient is 0. We even reaffirm this by looking at the coefficient 95% confidence interval.

3)Calculate residuals a based on appropriate visualizations comment on fulfilling the prerequisites of the selected model (heteroskedascity, normality, regression model shape, etc.)

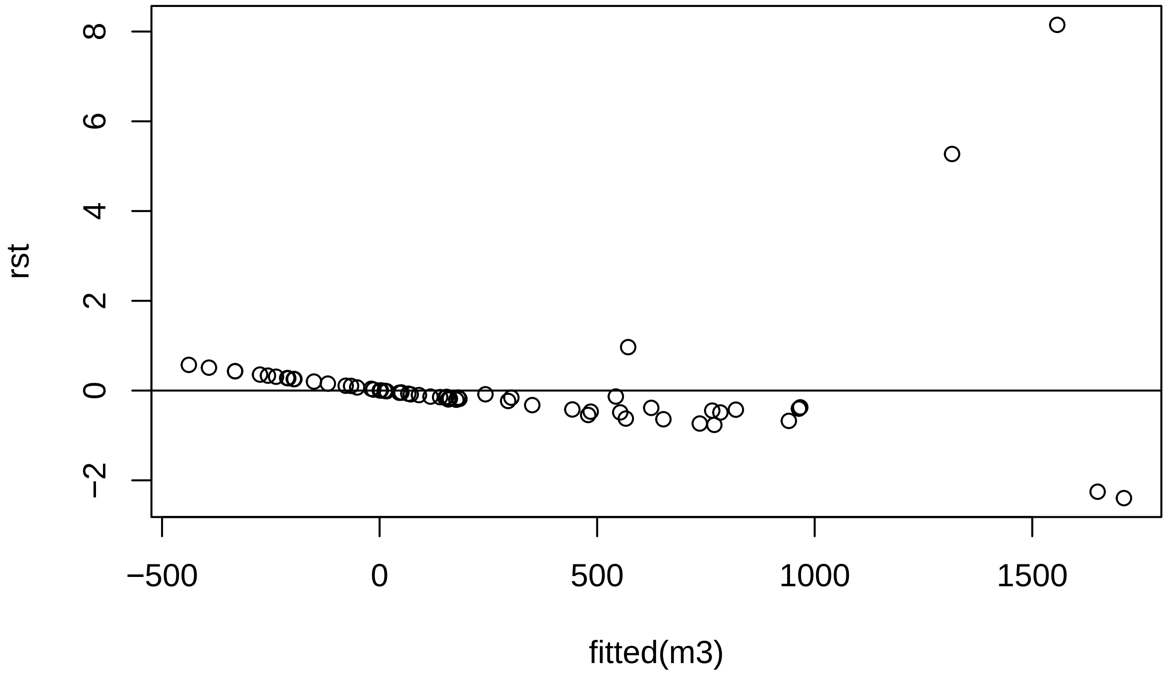
We use the appropriate function to get residuals and test heteroskedacity using studentized Breusch-Pagan test.

```
rst=rstudent(m3)
bptest(m3)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  m3
## BP = 17.916, df = 1, p-value = 2.309e-05
```

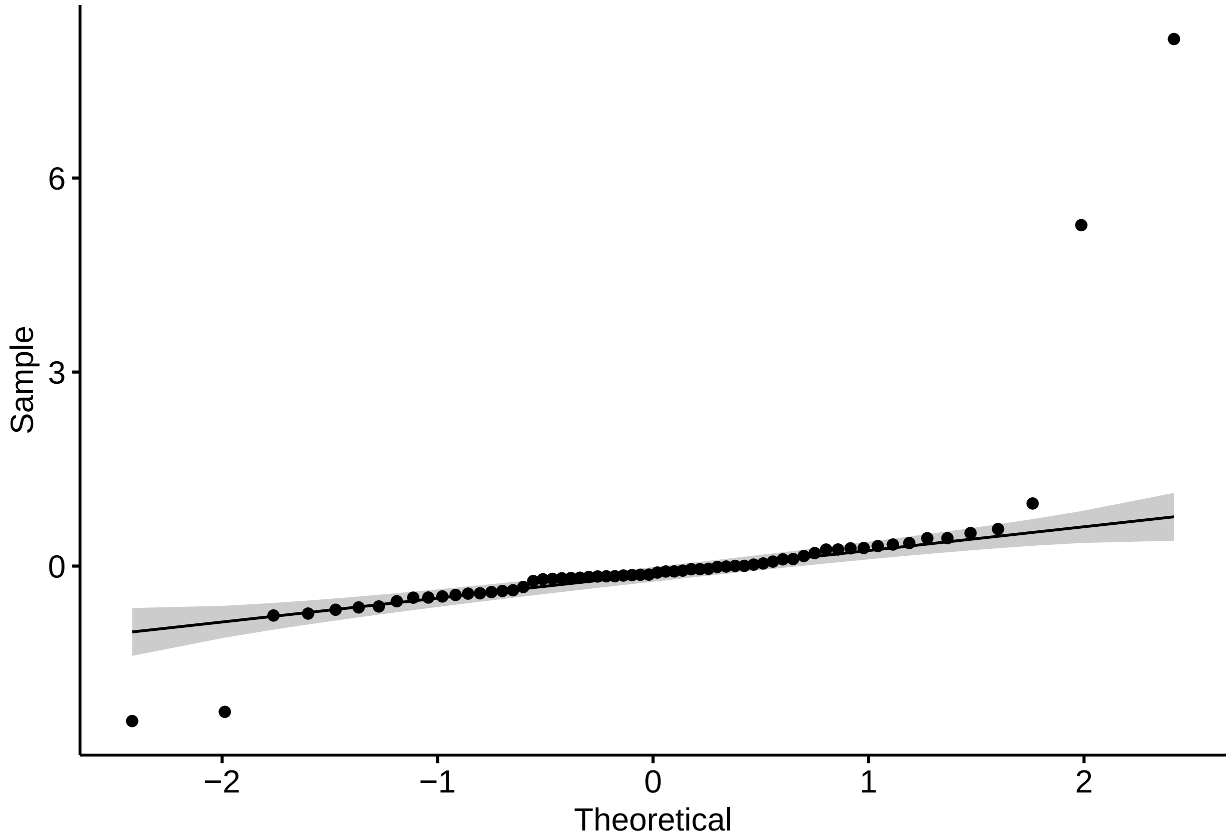
We can reject the null therefore hetoroskedacity is present.

```
plot(rst~fitted(m3))
abline(0,0)
```



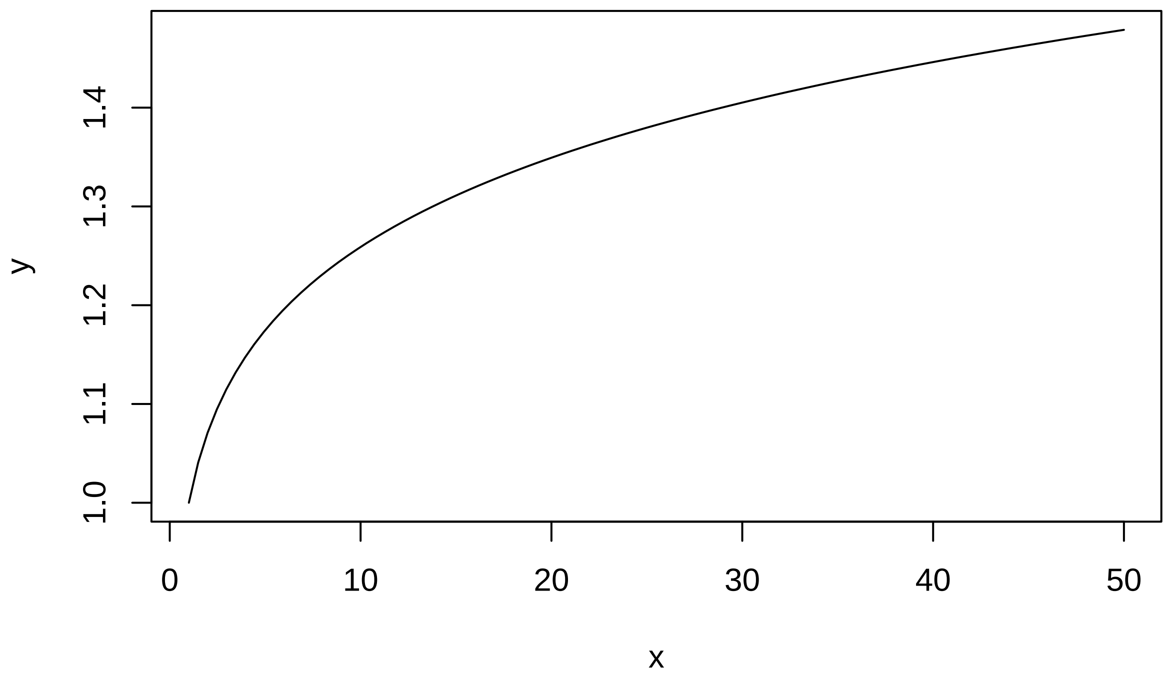
Maybe if we removed more outliers it wouldn't be, but that is probably not valid. (Removing the two dinosaurs would make some sense, but we already removed one and still even if we removed the other that would not fix everything.)

```
ggqqplot(rst)
```



There seem to be 2 outliers other than that our residuals look normal. Formula for our model is $Y = X^{1/10}$ which gives it this shape:

```
curve(x^(1/10), from=1, to=50, , xlab="x", ylab="y")
```


y
1.0
1.1
1.2
1.3
1.4
0
10
20
30
40
50
x