Homework 2021-22: Regression and PCA

Deadline: 2022-04-30

Name1 and Name2

2022-04-28

Contents

Deliverables

1. **2022-04-30:** a `nom1_nom2.Rmd` file that can be `knitted` under `rstudio`

- Your file should be knitted without errors and the result should be an html document that can be viewed in a modern browser.
- The file should contain the code used to generate plots and numerical summaries

- The file should contain the texts of your comments (either in English or in French)
- Comments should be written with care, precision and should be concise
- File `nom1_nom2.Rmd` will be uploaded on `Moodle` on due date

2. A short defense (15 minutes + 15 minutes questions) with slides (May)

You may work in pairs

Objectives

This notebook aims at

- Working with **tables** (`data.frames`, `tibbles`, `data.tables`, ...) using `dplyr` or any other query language (as provided for example by `data.table`)
- Exploring a multivariate dataset
- Using **PCA** to cope with collinearity of explanatory variables
- Performing simple and multiple linear regression
- Interpreting numerical and graphic diagnostics
- Perform variable selection using penalization
- Assess predictive performance of linear models
- Transform response and explanatory variables so as to improve interpretation and predictive performance

---

```
pacman::p_load(readr)
pacman::p_load(dplyr)
pacman::p_load(car)
pacman::p_load(lmtest)
pacman::p_load(ggplot2)
pacman::p_load(GGally)
pacman::p_load(gridExtra)
pacman::p_load(MASS)
pacman::p_load(leaps)
pacman::p_load(glmnet)
pacman::p_load(caret)
pacman::p_load(gbm)
pacman::p_load(knitr)
pacman::p_load(tidyverse)  # metapackage
pacman::p_load(tidymodels) # metapackage
pacman::p_load(vcd)
pacman::p_load(formula)
pacman::p_load(glue)
pacman::p_load(here)
```

Data loading

Load the data.

```
data <- read.csv("data4dm.csv")
```

Make columns with less than ten distinct values factors.

```
data %>%
  summarise_all(n_distinct)
```

```
##   X1  X2  X3 X4   X5   X6  X7  X8  Y
## 1  3 134 111 51 2429 1515 880 926 28
```

```
data <- data %>%
  mutate(X1 = as.factor(X1))
```

Sanity checks

Are there missing values?

All numerical entries should be positive. 0 is a surrogate for `NA`.

```
colSums(data == 0)
```

```
## X1 X2 X3 X4 X5 X6 X7 X8  Y
##  0  0  0  2  0  0  0  0  0
```

Numerical summary for categorical columns

The numerical summary is a contingency table

```
cat_cols <- unlist(lapply(data, is.factor))

summary(data[,cat_cols])
```

```
##    A    B    C
## 1307 1342 1527
```

Numerical summary for numerical columns

For each column, a numerical summary contains as much information as the output of function `summary()` (when applied to a numeric vector).

```
num_cols <- unlist(lapply(data, is.numeric))
summary(data[,num_cols])
```

```
##        X2              X3              X4              X5
##  Min.   : 15.0   Min.   : 11.00   Min.   :  0.00   Min.   :  0.4
##  1st Qu.: 90.0   1st Qu.: 70.00   1st Qu.: 23.00   1st Qu.: 88.3
##  Median :109.0   Median : 85.00   Median : 28.00   Median :159.9
##  Mean   :104.8   Mean   : 81.58   Mean   : 27.91   Mean   :165.8
##  3rd Qu.:123.0   3rd Qu.: 96.00   3rd Qu.: 33.00   3rd Qu.:230.7
##  Max.   :163.0   Max.   :130.00   Max.   :226.00   Max.   :565.1
##        X6              X7              X8              Y
##  Min.   :  0.20   Min.   :  0.10   Min.   :  0.30   Min.   : 1.000
##  1st Qu.: 37.20   1st Qu.: 18.68   1st Qu.: 26.00   1st Qu.: 8.000
##  Median : 67.20   Median : 34.20   Median : 46.80   Median : 9.000
##  Mean   : 71.88   Mean   : 36.12   Mean   : 47.77   Mean   : 9.932
##  3rd Qu.:100.40   3rd Qu.: 50.60   3rd Qu.: 65.80   3rd Qu.:11.000
##  Max.   :297.60   Max.   :152.00   Max.   :201.00   Max.   :29.000
```

Make any relevant comment

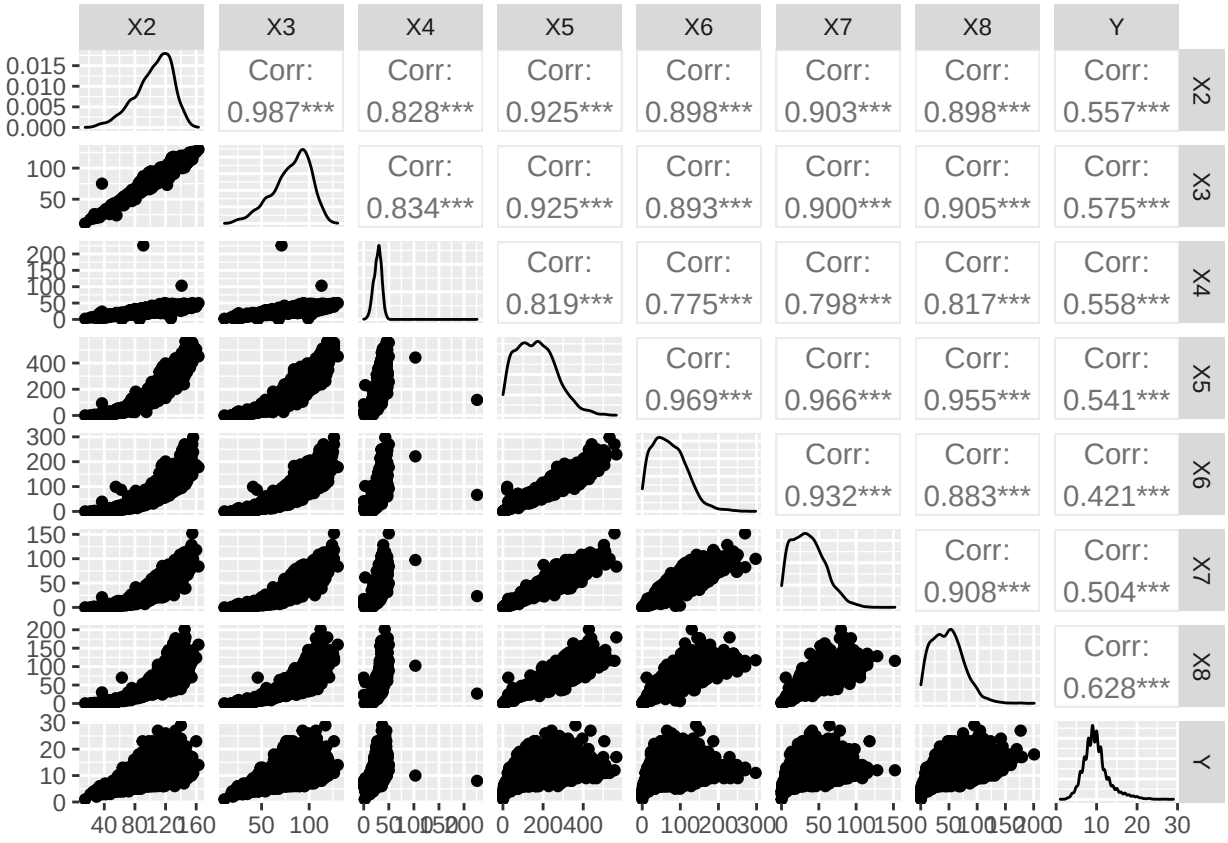
I have to make some comment here.

Pair plots

Use `ggpairs` from `GGally` to get a *big picture* of the dataset

Display linear correlations between the numerical columns (packages `corrr` or `corrplot` may be useful)

```
ggpairs(data[,num_cols])
```



Compare the conditional distributions of the numerical columns given the categories from categorical columns

Perform qqplots (not normal qqplots), compute two-sample Kolmogorov-Smirnov statistics

Is it worth collapsing some qualitative categories (use `fct_collapse()` from `forcats`)?

Track functional dependencies

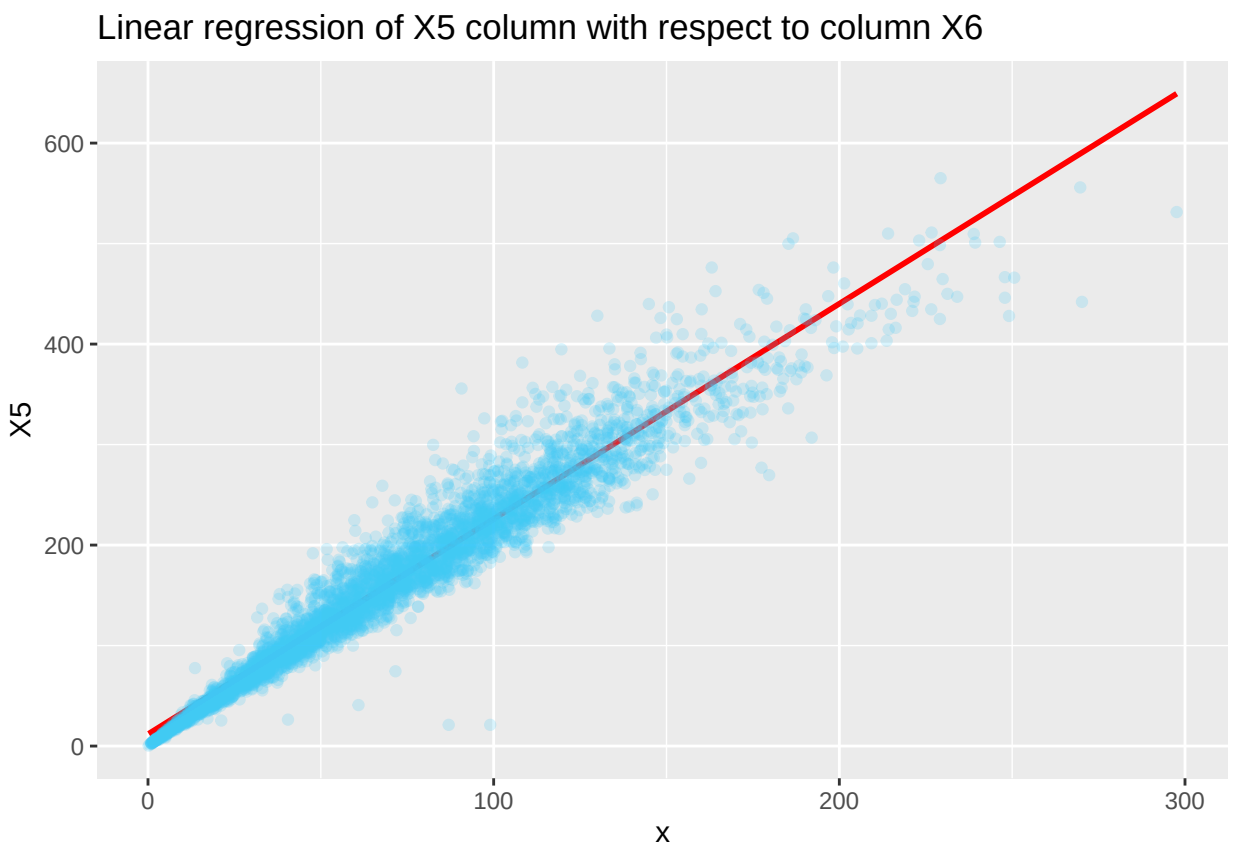
Perform linear regression of column `X5` with respect to columns `X6`, `X7`, `X8`.

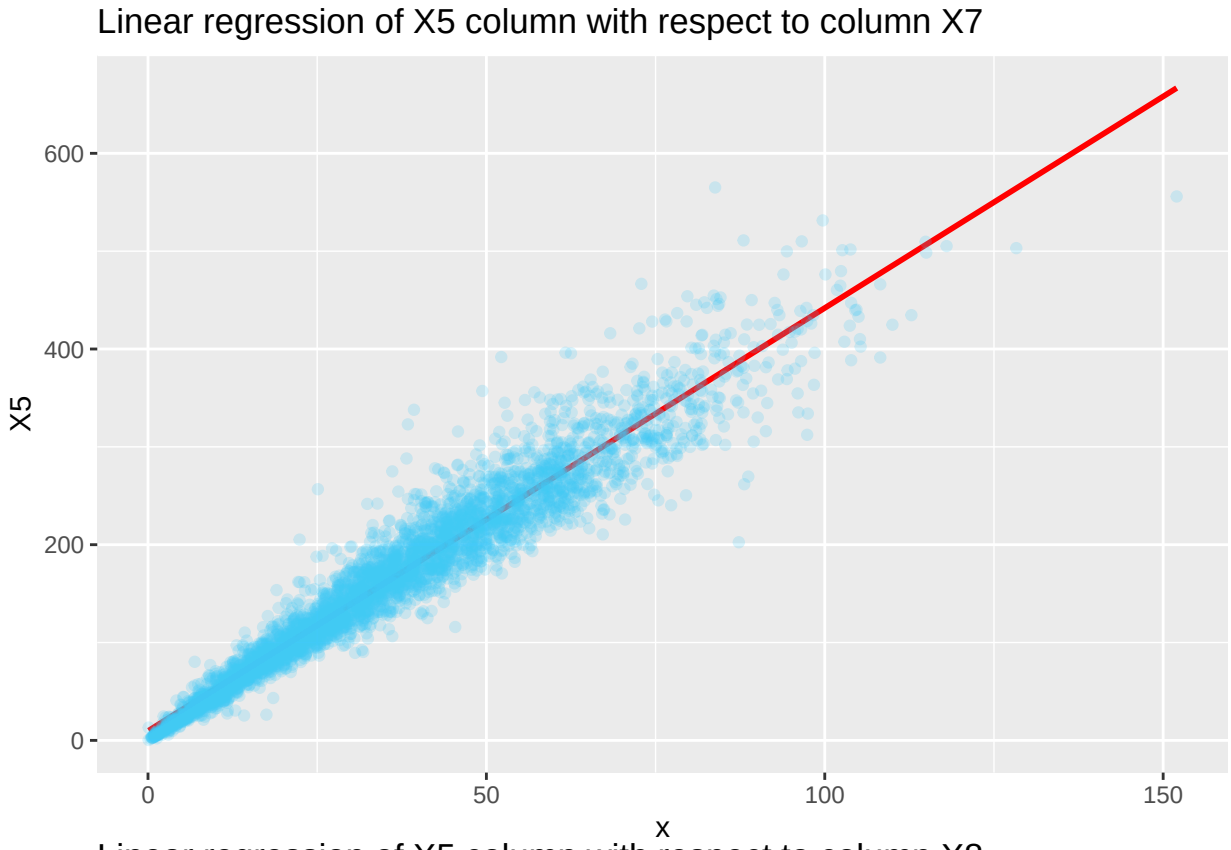
```
lm0 <- lm(X5 ~ X6, data=data)
lm0
```

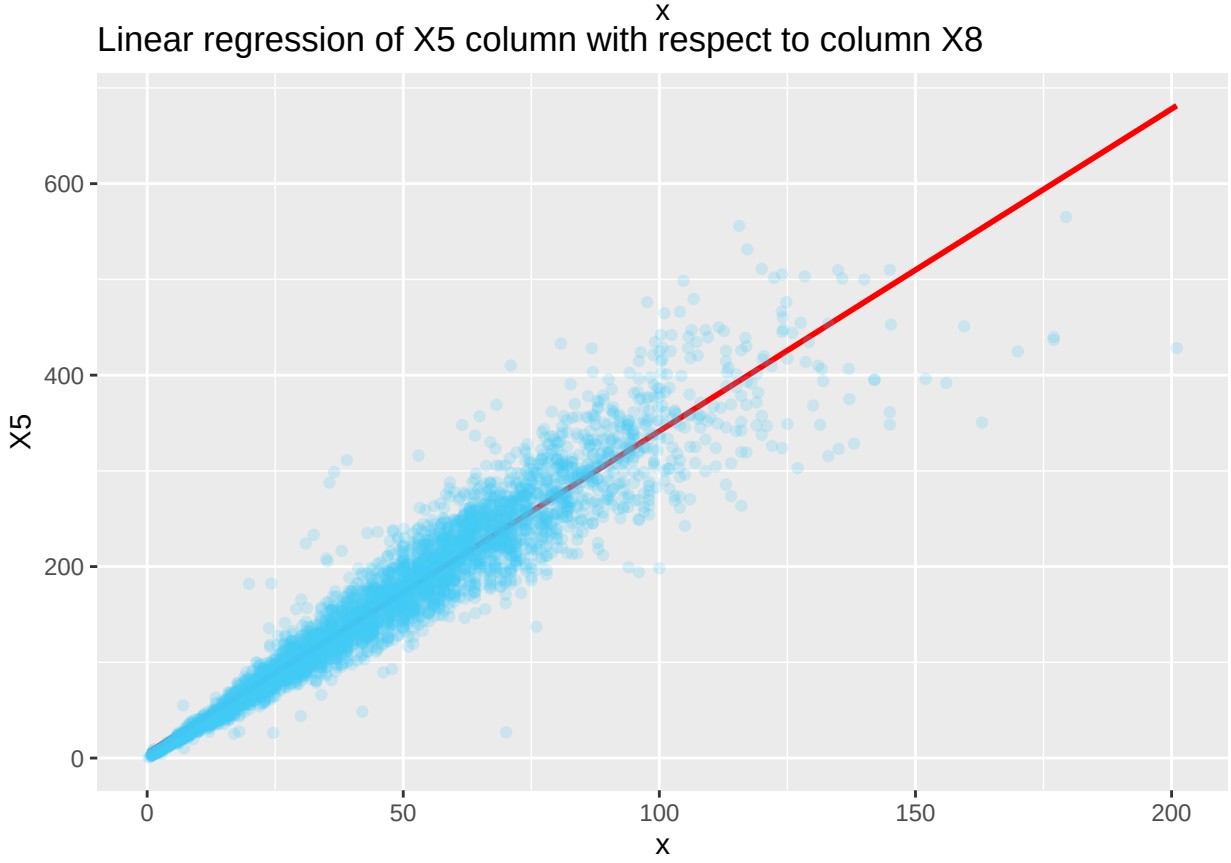
```
##
## Call:
## lm(formula = X5 ~ X6, data = data)
##
## Coefficients:
## (Intercept)           X6
```

```
##      11.817        2.142
```

```
i<-6

for(x in data %>% dplyr::select(X6,X7,X8)){
  data %>%
  ggplot() +
  aes(x=x) +
  aes(y=X5) +
  geom_smooth(
    color='red',
    method = "lm",
    formula = y ~ x,
    se=FALSE
  )+
  geom_point(
    color= '#41CAF7',
    alpha = .2
  ) +
  ggtitle(sprintf("Linear regression of X5 column with respect to column %s ",colnames(data)[i]))->p
  p %>% print()
  i<-i+1
  #Sys.sleep(2)
}
```







Is it worth considering regression of `Y` with respect to `X5`?

The corelation factor between the two columns is very low (0.541)

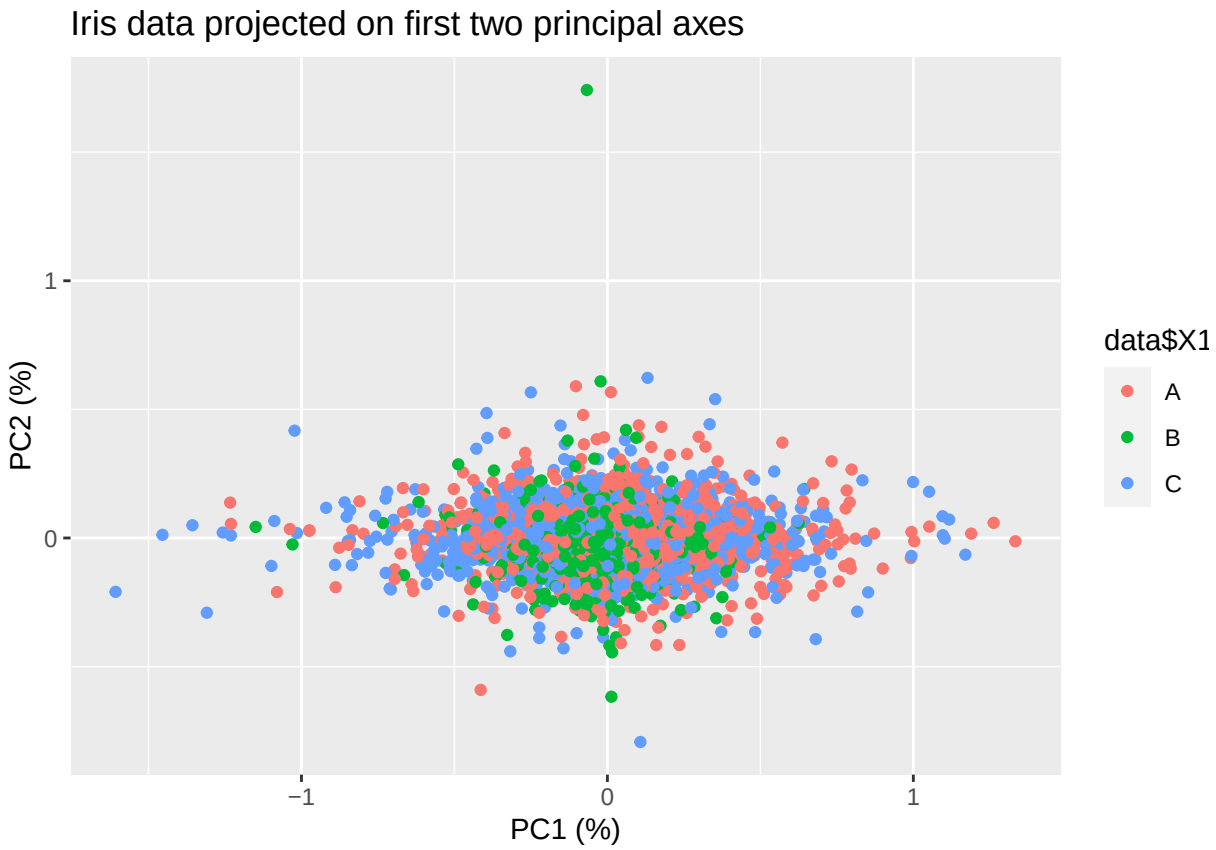
PCA

Perform PCA on the explanatory variables (`X1, ..., X8`).

```
data_pca <-data %>% dplyr::select(!c(X1,Y))
pca<- data_pca %>% prcomp(scale = T, center = T)
data_pca %>%
  broom::tidy(matrix="pcs") %>%
  knitr::kable(format="markdown", digits=2)
```

column	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
X2	4176	104.80	24.02	109.00	106.50	16.00	15.0	163.0	148.0	-0.64	3.06	0.37
X3	4176	81.58	19.85	85.00	82.94	13.00	11.0	130.0	119.0	-0.61	2.95	0.31
X4	4176	27.91	8.37	28.00	28.05	5.00	0.0	226.0	226.0	3.13	78.95	0.13
X5	4176	165.76	98.08	159.95	159.93	71.25	0.4	565.1	564.7	0.53	2.97	1.52
X6	4176	71.88	44.40	67.20	68.79	31.70	0.2	297.6	297.4	0.72	3.59	0.69
X7	4176	36.12	21.92	34.20	34.67	15.90	0.1	152.0	151.9	0.59	3.08	0.34
X8	4176	47.77	27.84	46.80	46.11	19.90	0.3	201.0	200.7	0.62	3.53	0.43

```
share_variance <- broom::tidy(data_pca, "pcs")[["percent"]]
pca<- broom::augment(pca, data_pca)
pca %>%
  ggplot() +
  aes(x=.fittedPC5, y=.fittedPC6) +
  aes(colour=data$X1) +
  geom_point() +
  ggtitle("Iris data projected on first two principal axes") +
  xlab(paste("PC1 (", share_variance[1], "%)", sep="")) +
  ylab(paste("PC2 (", share_variance[2], "%)", sep="")) ->p


  #calculate total variance explained by each principal component
#results$sdev^2 / sum(results$sdev^2)
p
```




```
#pca
```

Is it possible to connect qualitative variables with PCA (which is performed on quantitative columns)?

Regression framework

The data were not collected from experimental results. They were rather collected as a collection of i.i.d. multivariate observations (this corresponds to the *random design* setting whereas we have studied the *fixed design* framework during the course). We aim at predicting the value of Y from $X^1, \dots, X^8$, that is to find a function f such that

$$\mathbb{E}\left[\left(Y - f(X^1, \dots, X^8)\right)^2\right]$$

is minimized.

If there is no prescription about f (beyond measurability and square integrability), the minimizer is (a version of) the conditional expectation of Y with respect to $\sigma(X^1, \dots, X^8)$.

We will look for functions that satisfy criteria.

1. First we look for simple linear functions: $aX^i + b$ for $i \in \{1, \dots, 8\}$ and X^i a numerical variable.
2. Second, we look at the best mutivariate linear function $\sum_{i=1} \beta_i X^i$ where qualitative columns have been properly encoded.
3. We look at the mutivariate linear functions of the *principal components*: $\sum_{i=1}^q \beta_i \widetilde{X}^i$ where the $\widetilde{X}^i$ are the principal component columns (use `broom::augment()` to retrieve them for `prcomp` objects).
4. We attempt to perform *variable selection*
5. If diagnostics (either numerical or graphical) suggest that, conditionally on the design matrix, the homoschedastic Gaussian noise assumptions are violated, we may transform the data, look for a

prediction of $(Y^\lambda - 1)/\lambda$ for some $\lambda \in \mathbb{R}$ (notice that Y is positive), with the convention that for $\lambda = 0$, $(Y^\lambda - 1)/\lambda = \log(Y)$.

6. In a further step, we look for coefficients that minimize:

$$\sum_{i=1}^{n} \left((Y_i^\lambda - 1)/\lambda - \left(\beta_0 + \sum_{k=1}^{d} \sum_{j=1}^{8} \beta_{j,k} (X^i)^k \right) \right)^2$$

for $d = 3$.

Split the data

As we dive into predictive modelling, it is useful to split the data into two parts: a training set (two thirds of observations) and a testing set (one third of observations). This splitting is best performed at random. Try to balance evenly the different levels of the categorical variables in the training and testing set.

Package `rsample` could be useful.

```
set.seed(130)
#MonterCarlo sample
data %>% mc_cv(prop= 2/3 ,times=1)-> sample

sample$splits[[1]] %>% assessment()->testing_set
sample$splits[[1]] %>% analysis()->training_set

quant_col <- data %>% dplyr::select(!c(X1,Y)) %>% colnames()
```

Simple linear regressions

Perform simple linear regressions of Y with respect to each quantitative numerical column.

```
lr_col <-c()
for(c in training_set[quant_col]){
  lr<-lm(training_set$Y~c)
  lr %>% summary() %>% print()
  #lr %>% plot()
  lr_col<-c(lr_col,lr)
}
```

```
##
## Call:
## lm(formula = training_set$Y ~ c)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8681 -1.6545 -0.7221  0.8117 16.7387
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.227199   0.224100   9.938   <2e-16 ***
## c           0.073037   0.002087  35.001   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.645 on 2782 degrees of freedom
## Multiple R-squared:  0.3057, Adjusted R-squared:  0.3055
## F-statistic:  1225 on 1 and 2782 DF,  p-value: < 2.2e-16
##
##
## Call:
## lm(formula = training_set$Y ~ c)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.1132 -1.6543 -0.6935  0.8166 16.0704
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.394213   0.208189   11.50   <2e-16 ***
## c           0.091779   0.002483   36.97   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.6 on 2782 degrees of freedom
## Multiple R-squared:  0.3294, Adjusted R-squared:  0.3292
## F-statistic:  1367 on 1 and 2782 DF,  p-value: < 2.2e-16
##
##
## Call:
## lm(formula = training_set$Y ~ c)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -41.633  -1.694  -0.697   0.895  15.494
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.282553   0.172347   24.85   <2e-16 ***
## c           0.200665   0.005915   33.93   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.67 on 2782 degrees of freedom
## Multiple R-squared:  0.2927, Adjusted R-squared:  0.2924
## F-statistic:  1151 on 1 and 2782 DF,  p-value: < 2.2e-16
##
##
## Call:
## lm(formula = training_set$Y ~ c)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.0637 -1.7240 -0.6693  0.9463 15.7962
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
```

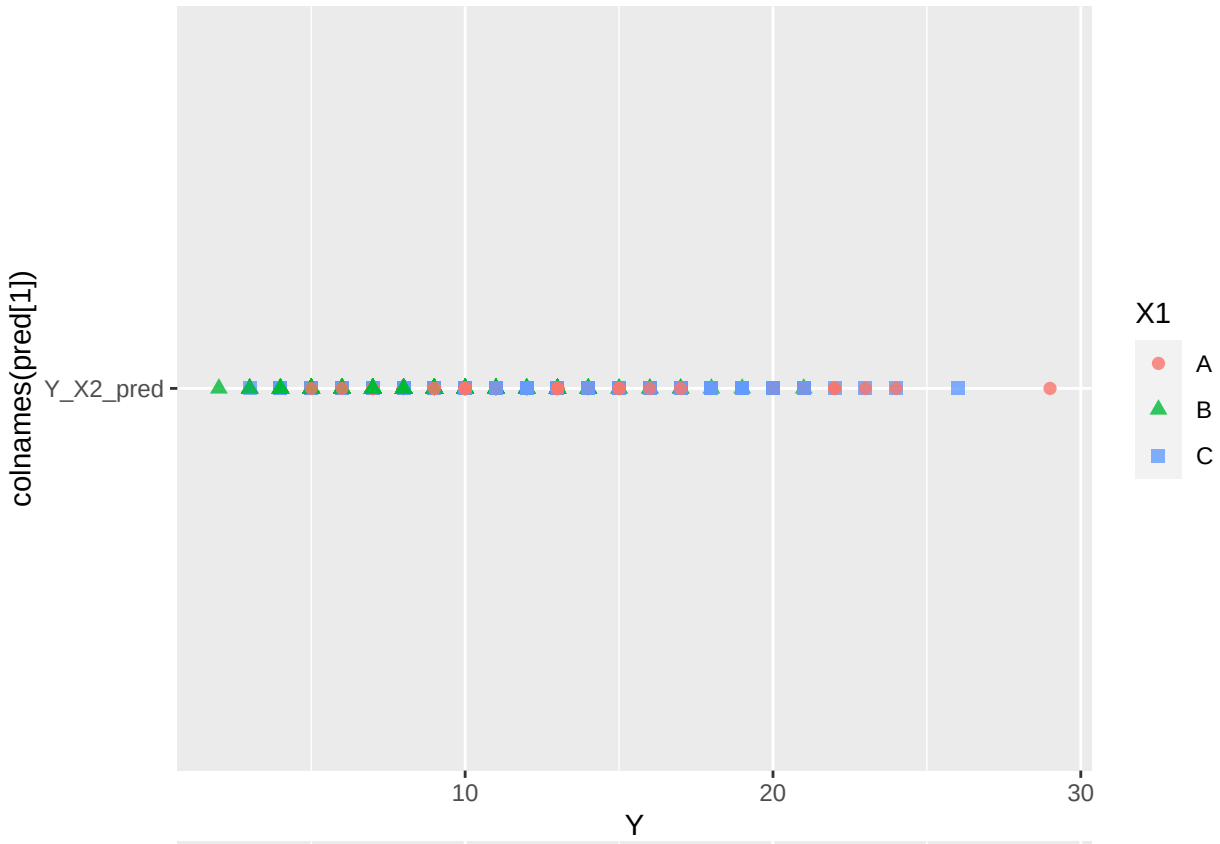
```
## (Intercept) 7.0175297  0.0990532   70.85   <2e-16 ***
## c           0.0172628  0.0005144   33.56   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.679 on 2782 degrees of freedom
## Multiple R-squared:  0.2882, Adjusted R-squared:  0.2879
## F-statistic:  1126 on 1 and 2782 DF,  p-value: < 2.2e-16
##
##
## Call:
## lm(formula = training_set$Y ~ c)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.7255 -1.8565 -0.7679  1.0195 17.3365
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7.719521   0.102903   75.02   <2e-16 ***
## c           0.029953   0.001215   24.66   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.876 on 2782 degrees of freedom
## Multiple R-squared:  0.1794, Adjusted R-squared:  0.1791
## F-statistic: 608.2 on 1 and 2782 DF,  p-value: < 2.2e-16
##
##
## Call:
## lm(formula = training_set$Y ~ c)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.2763 -1.7276 -0.7251  0.9825 16.7712
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7.256244   0.099414   72.99   <2e-16 ***
## c           0.072500   0.002349   30.86   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.74 on 2782 degrees of freedom
## Multiple R-squared:  0.255,  Adjusted R-squared:  0.2547
## F-statistic: 952.3 on 1 and 2782 DF,  p-value: < 2.2e-16
##
##
## Call:
## lm(formula = training_set$Y ~ c)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8096 -1.5774 -0.5633  0.9343 13.0908
```

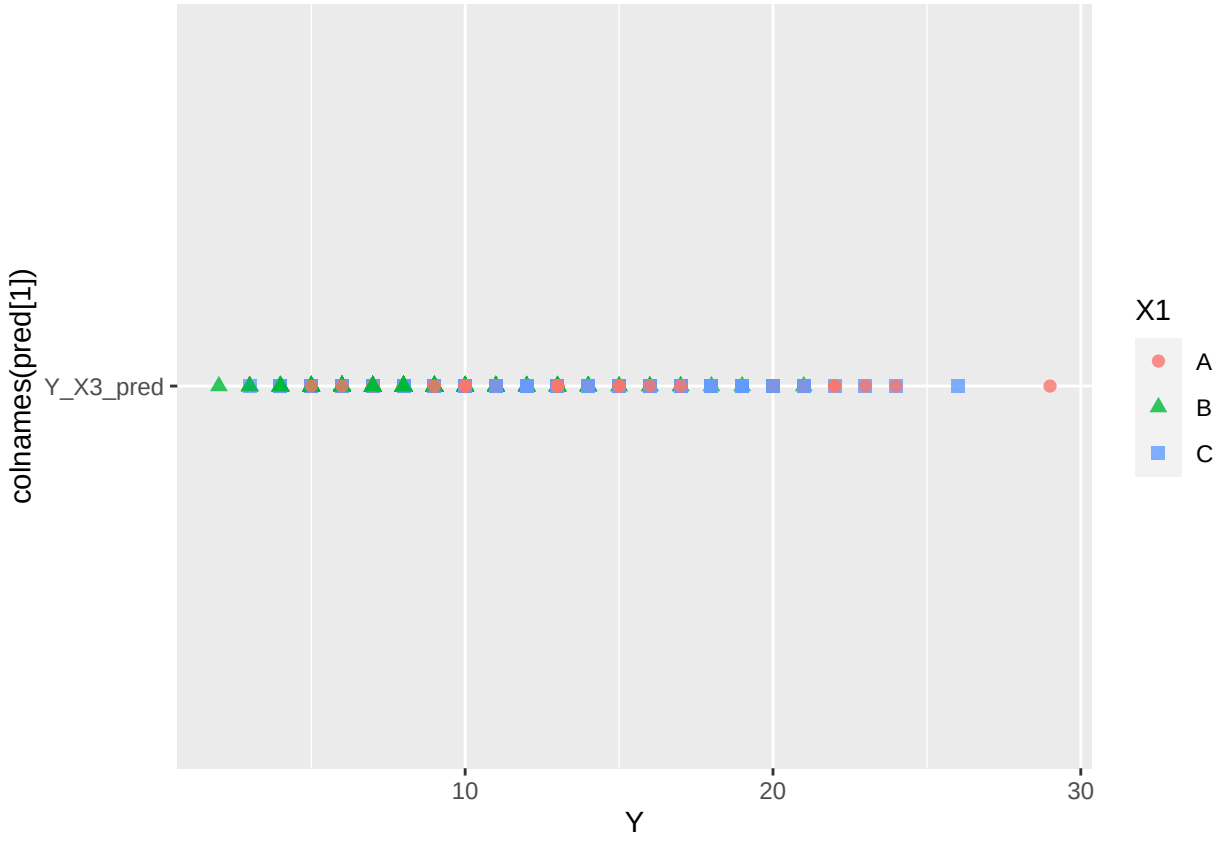
```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.496965   0.093263   69.66   <2e-16 ***
## c           0.071089   0.001695   41.93   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.485 on 2782 degrees of freedom
## Multiple R-squared:  0.3872, Adjusted R-squared:  0.387
## F-statistic:  1758 on 1 and 2782 DF,  p-value: < 2.2e-16
```

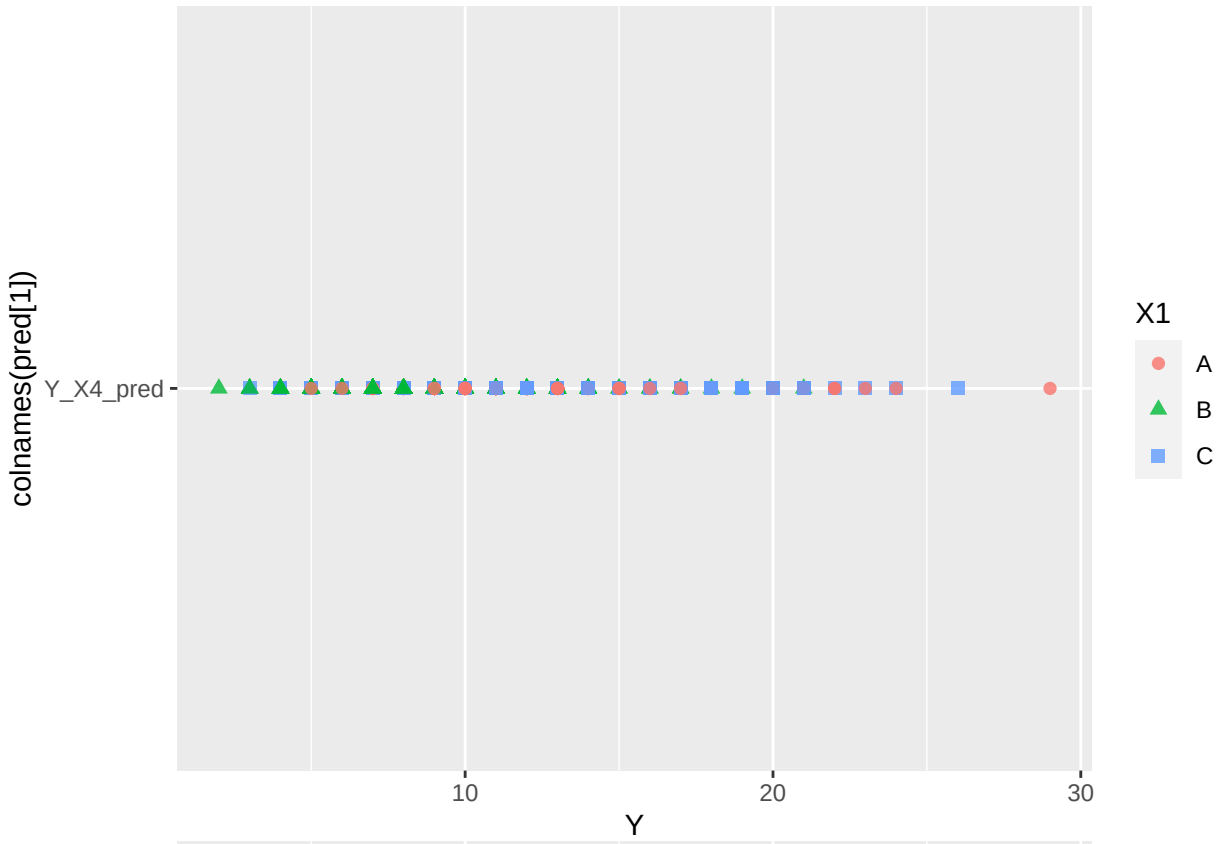
Comment on each numerical summary and comments the diagnostic plots. > the regression isn't neat, balues to predict aren't linear with respect quantitatvie columns.

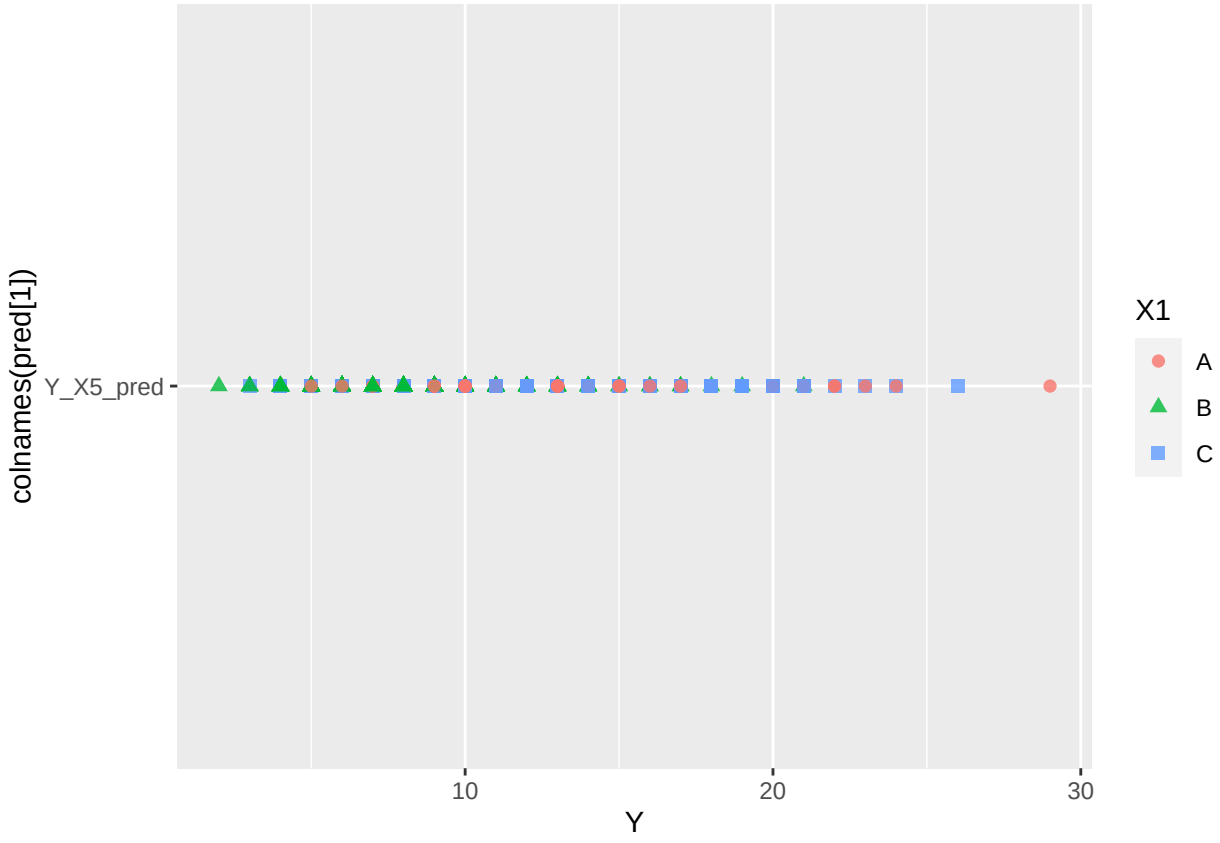
Use the coefficients estimated on the training set to predict Y on the testing set. Compare the training error and the testing error.

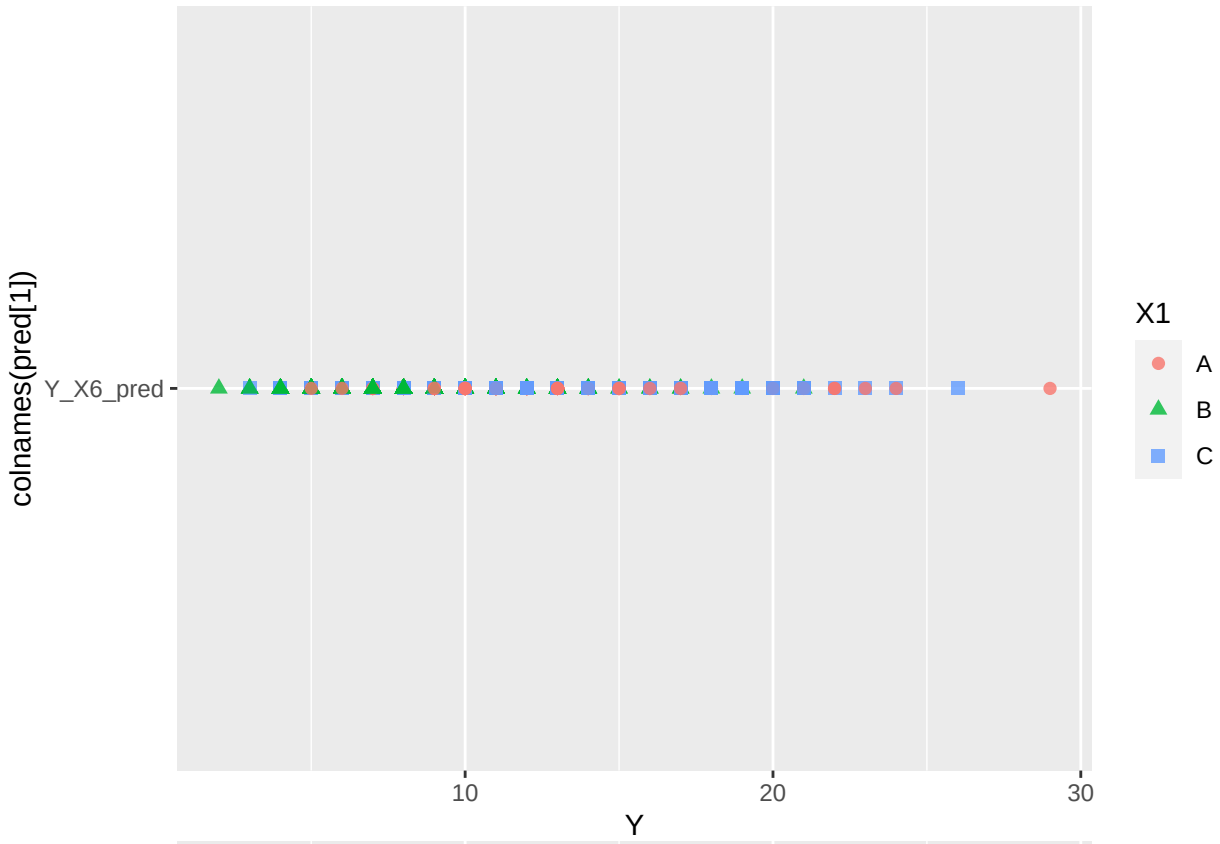
```
i<-1
for(c in training_set[quant_col]){
lr<-lm(training_set$Y~c)
lr$coefficients[[1]]-> a
lr$coefficients[[2]]-> b
pred<- testing_set[quant_col[i]]*a +b
#renaming
colnames(pred)[1] <- paste("Y_",colnames(pred)[1], "_pred",sep="")
testing_set[colnames(pred)[1]]<-pred
testing_set$Ypred<- pred


testing_set[colnames(pred[1])]
testing_set %>%
   ggplot()+
   aes(x=Y, y=colnames(pred[1]))+

  geom_point(
  alpha = .8,
  size = 2,
  aes(color = X1,
      shape = X1),
  )->g
g %>% print()
#plot(x=testing_set$Y, y=testing_set$Ypred)
#testing_set %>% dplyr::select(c(quant_col)) %>%
# predict(lr,.,se.fit = TRUE )->pred
#resid<- mean((testing_set$Y-pred$fit)^2)
#plot(pred$residual.scale)
i<-i+1




}
```

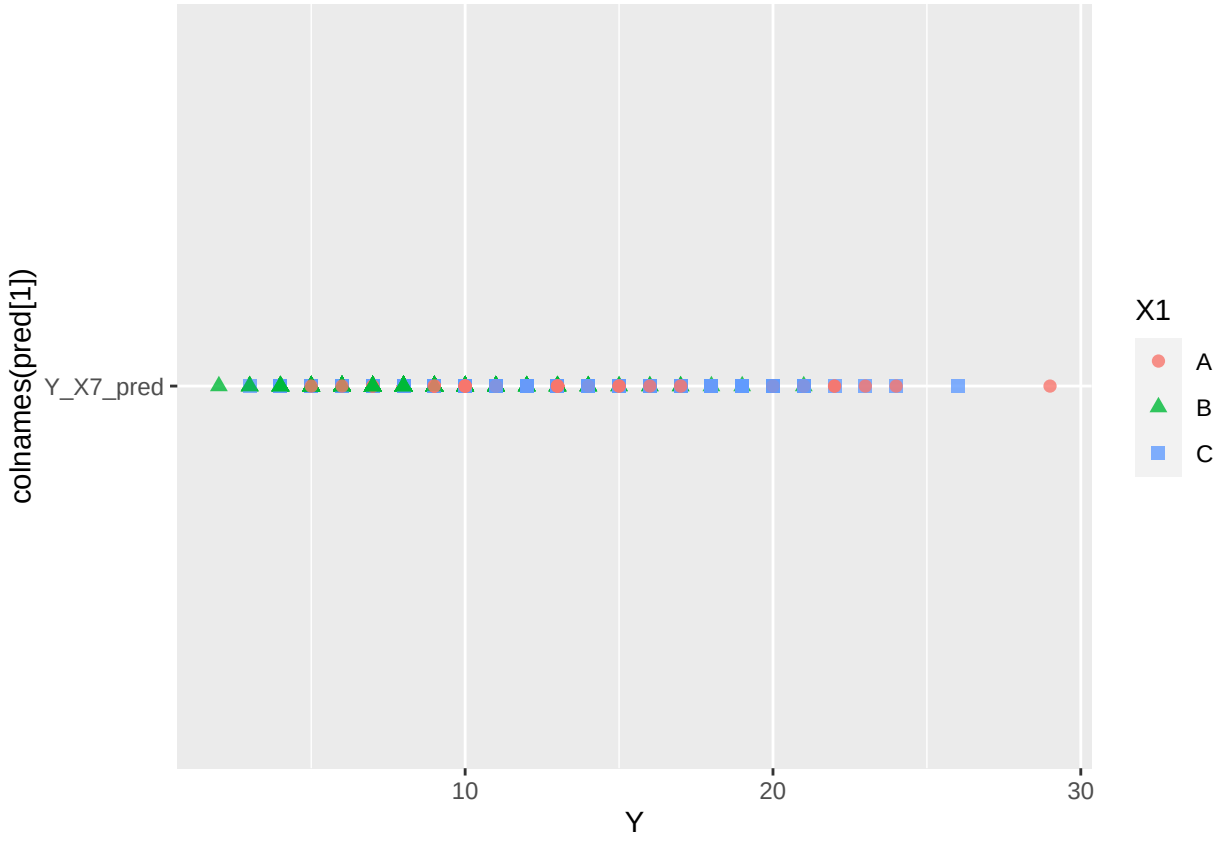

colnames(pred[1])
Y_X2_pred
10
20
30
Y
X1
A
B
C

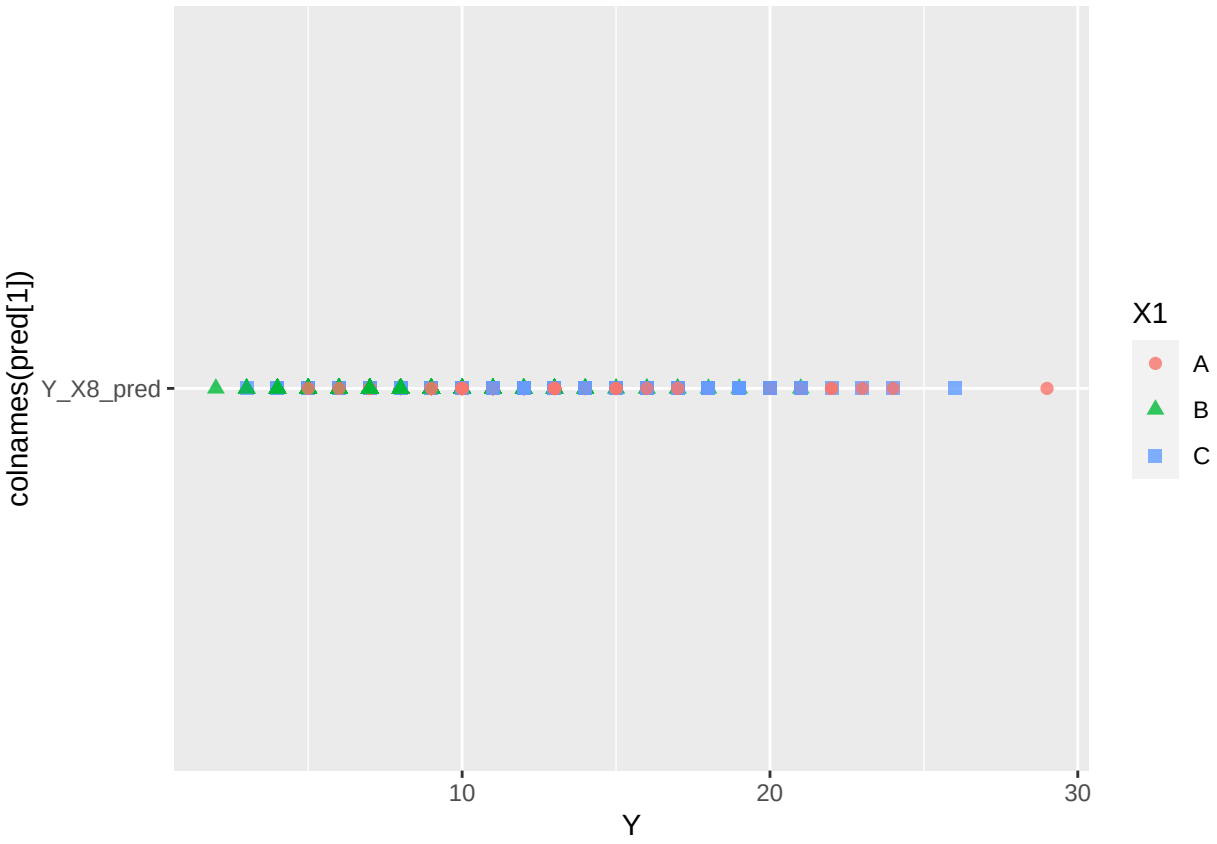

colnames(pred[1])
Y_X3_pred
10
20
30
Y
X1
A
B
C


colnames(pred[1])
Y_X4_pred
10
20
30
Y
X1
A
B
C


colnames(pred[1])
Y_X5_pred
10
20
30
Y
X1
A
B
C


colnames(pred[1])
Y_X6_pred
10
20
30
Y
X1
A
B
C


colnames(pred[1])
Y_X7_pred
10
20
30
Y
X1
A
B
C



What is your best predictor?

Do you think that the Gaussian Linear Modelling assumptions are satisfied?

Multiple linear regression

Perform linear regression of Y with respect to all predictors

Which coefficients are deemed non-zero?

Perform ANOVA testing to compare the full model (all predictors) with respect to your best simple model (one predictor). Comment.

Use again the coefficients estimated on the training set to predict Y on the testing set. Compare the training error and the testing error.

Regression with respect to principal components

Perform linear regression of Y with respect to one, two, ... principal components computed on the training set.

How many principal components would you use to perform prediction? (`stepAIC` from `MASS` can be useful)

Use again the coefficients estimated on the training set to predict Y on the testing set. Compare the training error and the testing error.

Package `broom` offers function to manipulate the outputs of `prcomp`.

Variable selection

Go back to Multiple linear regression with respect to the original variables. Use `stepAIC` from `MASS` to pick a subset of predictors.

In order to optimize the predictive capabilities of your model (perforance on the testing set), you may try to use *cross-validation* to identify a good set of predictors.

Response transformation

You may use `MASS::boxcox()` to estimate an optimal transformation of the response variable.

In any case, use the logarithmic transformation (`lambda=0`) and retrain your linear models

Do the linear models satisfy the Gaussian Linear Modelling assumptions (according to the diagnsotic plots)?

Again, try to pick the model with the best predictive performance

---

Combine response transformation and higer order modelling

Use `poly()` to perform linear modelling with respect to powers of the original predictors (up to degree `3`).

Do the linear models satisfy the Gaussian Linear Modelling assumptions (according to the diagnsotic plots)?

Again, try to pick the model with the best predictive performance