Compulsory Exercise 2

TMA4268 Statistical Learning V2022

Daesoo Lee, Emma Skarstein, Stefanie Muff, Department of Mathematical Sciences, NTNU

Hand out date: March 21, 2022

The submission deadline is: Monday April 4, 23:59h using Blackboard

Introduction

Maximal score is 50 points. Your score will make up 10% points of your final grade.

Supervision

We will use the times where we would have lectures and exercises for supervision (4×2 hours).

Supervision hours:

- Monday, March 28, 08:15-10:00 and 14.15-16.00
- Wednesday, March 30, 14.15-16.00
- Thursday March 31, 08.15-10.00

Remember that there is also the Mattelab forum, and we strongly encourage you to use it for your questions outside the supervision hours – this ensures that all other students benefit from the answers (try to avoid emailing the course staff).

Practical issues (Please read carefully)

- You should work in the same groups as for compulsory exercise 1.
- Remember to write your names and group number on top of your submission file!
- The exercise should be handed in as **one R Markdown file and a pdf-compiled version** of the R Markdown file (if you are not able to produce a pdf-file directly please make an html-file, open it in your browser and save as pdf - no, not landscape - but portrait please). We will read the pdf-file and use the Rmd file in case we need to check details in your submission.
- In the R-chunks please use both `echo=TRUE` and `eval=TRUE` to make it simpler for us to read and grade.
- Please do not include all the text from this file (that you are reading now) - we want your R code, plots and written solutions - use the template from the course page (https://wiki.math.ntnu.no/tma4268/2022v/subpage6).
- Please **not more than 14 pages** in your pdf-file! (This is a request, not a requirement.)
- Please save us time and **do not submit word or zip**, and do not submit only the Rmd. This only results in extra work for us!

Multiple choice problems

There will be a few *multiple choice questions*. This is how these will be graded:

- **Multiple choice questions (2P)**: There are four choices, and each of them can be TRUE or FALSE. If you make one mistake (either wrongly mark an option as TRUE/FALSE) you get 1P, if you have two or more mistakes, you get 0P. Your answer should be given as a list of answers, like TRUE, TRUE, FALSE, FALSE, for example.

R packages

You need to install the following packages in R to run the code in this file. It is of course also possible to use more or different packages.

```
install.packages("ggplot2")
install.packages("tidyverse")
install.packages("palmerpenguins")
install.packages("GGally")
install.packages("MASS")
install.packages("caret")
install.packages("leaps")
install.packages("glmnet")
install.packages("pls")
install.packages("gam")
install.packages("e1071")
install.packages("tree")
install.packages("randomForest")
install.packages("ggfortify")
```

Problem 1 (10P)

In the following questions in this problem, we use the *Boston Housing Price* dataset. A detailed description for the dataset can be found here. Note that a response variable is `medv` (= housing price) and predictor variables are the rest of the covariates. The dataset is preprocessed and split into 80% and 20% for a training set and test set, respectively. In this problem, several feature selection methods are addressed.

```
library(MASS)
str(Boston)
```

```
## 'data.frame':    506 obs. of  14 variables:
##  $ crim   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
##  $ zn     : num  18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
##  $ indus  : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
##  $ chas   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ nox    : num  0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...
##  $ rm     : num  6.58 6.42 7.18 7 7.15 ...
##  $ age    : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
##  $ dis    : num  4.09 4.97 4.97 6.06 6.06 ...
##  $ rad    : int  1 2 2 3 3 3 5 5 5 5 ...
##  $ tax    : num  296 242 242 222 222 222 311 311 311 311 ...
##  $ ptratio: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
##  $ black  : num  397 397 393 395 397 ...
```

```
##  $ lstat  : num  4.98 9.14 4.03 2.94 5.33 ...
##  $ medv   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
```

```r
set.seed(1)

# pre-processing by scaling NB! Strictly speaking, pre-processing should be done
# on a training set only and it should be done on a test set with statistics of
# the pre-processing from the training set. But, we're preprocessing the entire
# dataset here for convenience.
boston <- scale(Boston, center = T, scale = T)

# split into training and test sets
train.ind = sample(1:nrow(boston), 0.8 * nrow(boston))
boston.train = data.frame(boston[train.ind, ])
boston.test = data.frame(boston[-train.ind, ])
```

a) (2P)

Perform *Forward Stepwise Selection* and *Backward Stepwise Selection* on `boston.train` method, and plot a graph of adjusted R^2 on the y-axis and a number of predictors on the x-axis.

R-hints:

- use `regsubsets(..)` from `library(leaps)` for the selection methods
- use `set.seed(1)`

b) (2P)

Choose the *four* "best" (selected) predictors from the forward stepwise selection. You can use the results obtained in a).

c) (4P)

(i) Run K-fold cross-validation (K=5) on `boston.train` with Lasso and show a plot where the x-axis shows the lambda (or log(lambda)) of Lasso and the y-axis shows MSE (mean squared error). (2P)

(ii) Report the best lambda that you find (i.e., λ with minimum MSE). (1P)

(iii) Report the fitted coefficients at the best λ. (1P)

R-hints:

- Use `set.seed(1)`
- you can use `coef()` to print the coefficients (see here)

d) Multiple choice (2P)

Say for *each* of them if it is true or false.

(i) When comparing computational speed between step-wise feature selection methods and Lasso for features selection, Lasso is much faster.

(ii) It is easier for ridge regression than Lasso to result in coefficients equal zero, namely due to the quadratic penalization term in ridge.

(iii) For the purpose of feature selection, both Ridge and Lasso are equally appropriate.

(iv) Elastic Net is a combination of Lasso and Ridge.

Problem 2 (6P)

In this problem, a synthetic dataset is used. This dataset is purposefully created to show the difference between PCR and PLSR (PLS regression). We recommend you to explore the dataset before moving on.

```
library(MASS)
set.seed(1)

# load a synthetic dataset
id <- "1CWZYfrLOrFdrIZ6Hv73e3xxt0SFgU4Ph"  # google file ID
synthetic <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download",
    id))

# split into training and test sets
train.ind = sample(1:nrow(synthetic), 0.8 * nrow(synthetic))
synthetic.train = data.frame(synthetic[train.ind, ])
synthetic.test = data.frame(synthetic[-train.ind, ])

# show head(..)  Y: response variable; X: predictor variable
head(synthetic)
```

```
##             Y          X1          X2         X3         X4         X5
## 1 -1.43753239 -0.75905055 -0.69720326 -0.3016852 -0.7434697  0.8807558
## 2 -1.70972989 -0.28635632  0.04809182  0.5791725 -0.7446170  0.9935311
## 3  1.33931240  0.09574117 -0.89605758 -0.9636347  0.5554647 -0.5341800
## 4  0.20354906 -0.28702695  1.72952687  1.4289705 -0.1596993 -0.7161976
## 5 -0.09261896  0.02345825  0.51201583  0.1544345  0.4318039 -0.8674060
## 6  1.69952325  1.19231791 -0.98179754 -0.9567773 -0.6933918  0.4656891
##           X6         X7         X8          X9         X10
## 1 -0.8705750 -0.7448252 -0.4639697  0.62502272 -0.8149674
## 2  0.3532248 -0.5860332 -0.7964403  0.84868110 -0.1065119
## 3  0.4707434 -0.6588069 -0.7327518 -0.29429307  0.6588927
## 4 -0.7774007  0.2502145  0.5987052 -0.04428773  0.6247479
## 5 -0.9066908  0.8946086 -0.9700185  0.09082626  0.6102134
## 6 -0.7381794  0.8650175  0.4108119  0.75677429 -0.2281439
```

a) (2P)

Fit PCR and PLSR on `synthetic.train` and show a graph of MSEP (mean squared error of prediction) with respect to the number of principal components for PCR and PLS, respectively.

Hint:

- use `validationplot(.., val.type="MSEP")` for the graph.

b) (4P)

Given the two plots from a), explain what causes the difference between the results from the two methods.

NB!

- The answer must be associated with characteristics of PLS and PCR and be associated with the training set the model is fitted on.

Problem 3 (5P)

a) (2P) - Multiple choice

Say for *each* of them if it is true or false.

(i) For the polynomial regression (where polynomial functions of features are used as predictors), variance increases when including predictor with a high order of the power.
(ii) If the polynomial functions from (i) are replaced with step functions, then the regression model is too simple to be overfitted on a dataset even with multiple cutpoints.
(iii) The smoothing spline ensures smoothness of its function, g, by having a penalty term $\int g'(t)^2 dt$ in its loss.
(iv) The K-nearest neighbors regression (local regression) has a high bias when its parameter, k, is high.

b) (3P)

Fit an additive model on `boston.train` using the function `gam()` from package `gam` with the following conditions, and plot the resulting curves.

- response: `medv`; predictors: `rm`, `ptratio`, `lstat` (use these three predictors only).
- `rm` is a linear function
- `ptratio` is a smoothing spline with `df=3`.
- `lstat` is a polynomial of degree 2.

Problem 4 (11P)

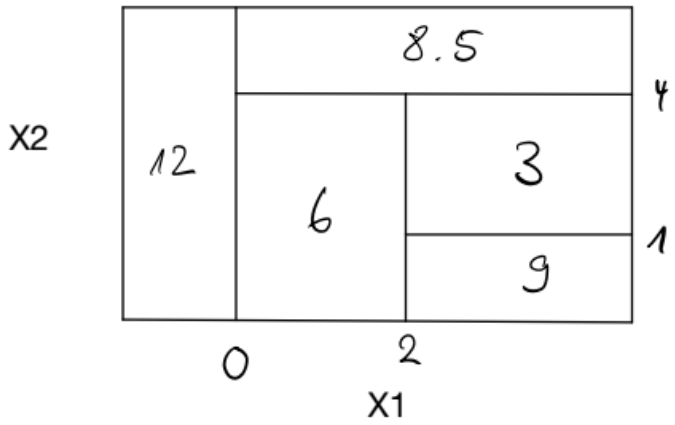
a) (2P) - Multiple choice

Which of the following statements are true, which false?

(i) A downside of simple regression trees is that they cannot handle interaction terms.
(ii) In boosting, the parameter d controls the number of splits allowed in each try. When $d = 2$, we allow for models with 2-way interactions.
(iii) The random forest approach improves bagging, because it reduces the variance of the predictor function by decorrelating the trees.
(iv) The number of trees B in boosting is a tuning parameter.

b) (2P)

Sketch the tree corresponding to the partition of the predictor space illustrated in the figure. The numbers inside the boxes are the mean of y within the regions (feel free to draw the tree by hand and upload a figure, or plot it with the computer).



c) (4P)

We are now again looking at the penguin dataset that we used in compulsory exercise 1 (the one that Basil, the cat, did not analyze very well). This time we are interested in a tree-based method to build a model that predicts the three different penguin species. In addition, we want to understand which factors are most relevant in discriminating the species.

We start again by loading and preparing the data set, similarly to exercise 3 in compulsory 1. We are also splitting the data into a training and a test set. You can run the following code without any changes:

```
library(tidyverse)
library(palmerpenguins)  # Contains the data set 'penguins'.
data(penguins)

names(penguins) <- c("species", "island", "billL", "billD", "flipperL", "mass", "sex",
    "year")

Penguins_reduced <- penguins %>% dplyr::mutate(mass = as.numeric(mass), flipperL = as.numeric(flipperL)
    year = as.numeric(year)) %>% drop_na()

# We do not want 'year' in the data (this will not help for future predictions)
Penguins_reduced <- Penguins_reduced[, -c(8)]

set.seed(4268)
# 70% of the sample size for training set
training_set_size <- floor(0.7 * nrow(Penguins_reduced))
train_ind <- sample(seq_len(nrow(Penguins_reduced)), size = training_set_size)
train <- Penguins_reduced[train_ind, ]
test <- Penguins_reduced[-train_ind, ]
```

Tasks:

(i) Start by generating a simple classification tree using the Gini index (using default `control` parameters) to find the splits and plot the resulting tree using the training data (1P).
(ii) Apply cost-complexity pruning using 10-fold CV, still using the training data (1P).
(iii) Find the optimal tree (1P) and report the misclassification error rate on the test set (1P).

R-hints:
(i) When plotting the tree, use the argument `type="uniform"` in the `plot()` function.

(ii) Please use `set.seed(123)` before your run cross-validation, so that it is easier to reproduce your results.

d) (3P)

Now construct a classification tree based on a more advanced method. Train the model using the training data and report the misclassification error for the test data (1P). Explain your choice of the (tuning) parameters (1P). Which two variables are the most influential ones in the prediction of the penguin species (1P)?

Problem 5 (6P)

a) (2P) - Multiple choice

Imagine you have gene expression data for leukemia patients, with $p = 4387$ gene expressions measured on blood samples for a total of $n = 92$ patients, of wich 42 have leukemia and 50 patients are healthy. Which statements are true?

(i) Logistic regression is the preferred method for this data set, because it gives nice interpretable parameter estimates.
(ii) In this dataset we are guaranteed to find a separating hyperplane, unless there are exact feature ties (two patients with the exact same gene data, but different outcome).
(iii) When fitting a support vector classifier, we usually have to standardize the variables first.
(iv) By choosing a smaller budget parameter C we are making the model less biased, but introduce more variance.

b) (4P)

We are a last time looking at the penguin dataset, using again the same training and test sets as in Problem 4.

(i) (2P) Fit a support vector classifier (linear boundary) and a support vector machine (radial boundary) to find good functions that predict the three dolphin species. Use cross-validation to find a good cost parameter (for the linear boundary) and a good combination of cost *and* γ parameters (for the radial boundary), and report the error rates in the training set for both cases.
(ii) (1P) Report the confusion tables and misclassification error rates for the test set in both cases, using the best parameters you found in (i).
(iii) (1P) Which classifier do you prefer and why?

R-hints:
To run cross-validation over a grid of two tuning parameters, you can use the `tune()` function where `ranges` defines the grid points as follows:

```
tune(svm, formula, kernel = ..., ranges = list(cost = c(...), gamma = c(...)))
```

Problem 6 (12P)

In the following code, we're importing a word-happiness-report-2021 dataset. This dataset has a response of happiness score and several predictors such as GDP (gross domestic product), social support, life expectancy, freedom, generosity, and corruption level of 149 countries. One of the typical uses of this dataset is analysis of important variables that largely contribute to the happiness level by using a method such as PCA.

```r
# load a synthetic dataset
id <- "1NJ1SuUBebl5P8rMSIwm_n3S8a7K43yP4" # google file ID
happiness <- read.csv(sprintf("https://docs.google.com/uc?id=%s&export=download", id))

colnames(happiness)
```

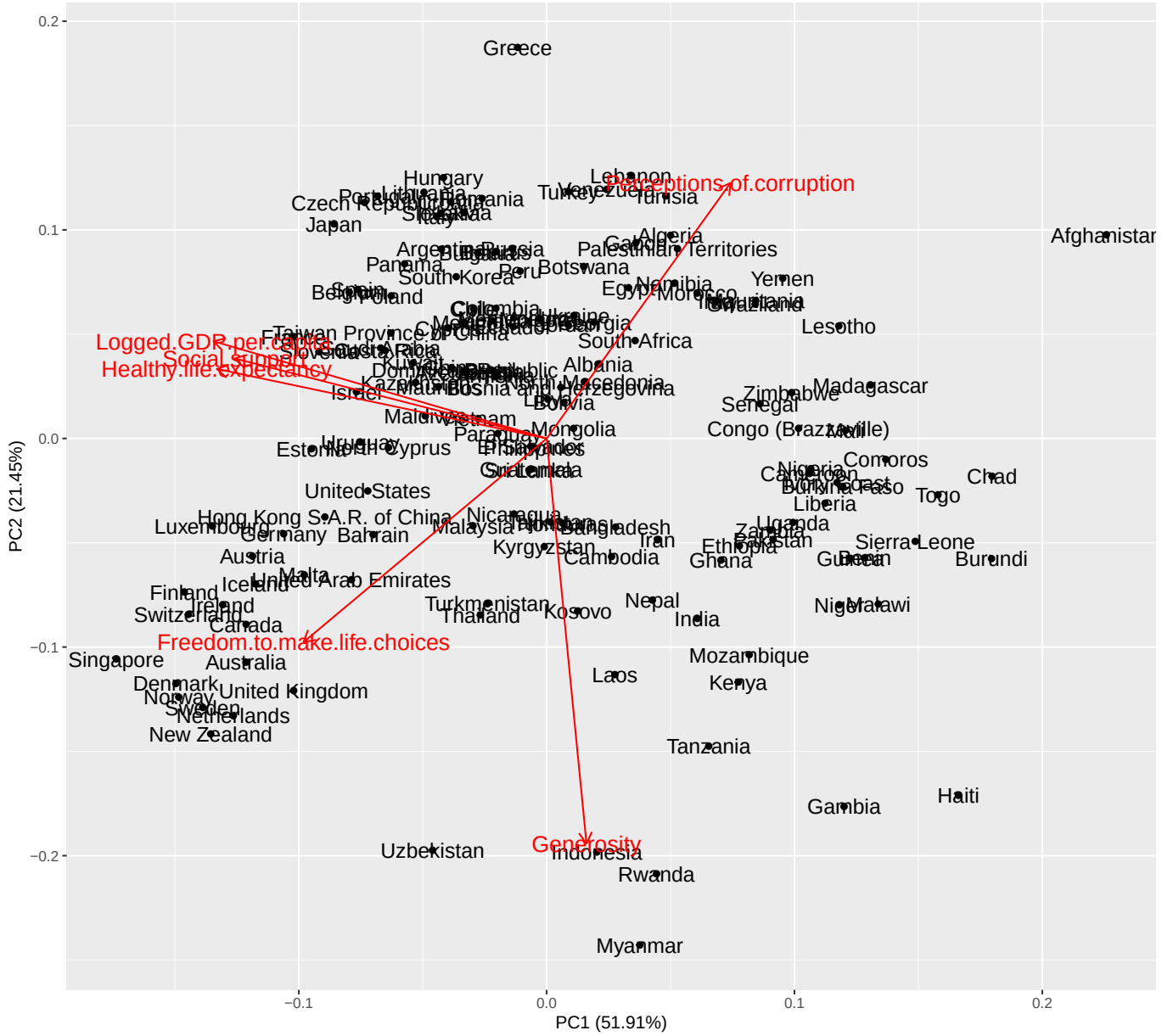
```
##  [1] "Country.name"
##  [2] "Regional.indicator"
##  [3] "Ladder.score"
##  [4] "Standard.error.of.ladder.score"
##  [5] "upperwhisker"
##  [6] "lowerwhisker"
##  [7] "Logged.GDP.per.capita"
##  [8] "Social.support"
##  [9] "Healthy.life.expectancy"
## [10] "Freedom.to.make.life.choices"
## [11] "Generosity"
## [12] "Perceptions.of.corruption"
## [13] "Ladder.score.in.Dystopia"
## [14] "Explained.by..Log.GDP.per.capita"
## [15] "Explained.by..Social.support"
## [16] "Explained.by..Healthy.life.expectancy"
## [17] "Explained.by..Freedom.to.make.life.choices"
## [18] "Explained.by..Generosity"
## [19] "Explained.by..Perceptions.of.corruption"
## [20] "Dystopia...residual"
```

```r
cols = c('Country.name',
         'Ladder.score',  # happiness score
         'Logged.GDP.per.capita',
         'Social.support',
         'Healthy.life.expectancy',
         'Freedom.to.make.life.choices',
         'Generosity',  # how generous people are
         'Perceptions.of.corruption')

# We continue with a subset of 8 columns:
happiness = subset(happiness, select = cols)
rownames(happiness) <- happiness[, c(1)]

# And we creat an X and a Y matrix
happiness.X = happiness[, -c(1, 2)]
happiness.Y = happiness[, c(1, 2)]
happiness.XY = happiness[, -c(1)]

# scale
happiness.X = data.frame(scale(happiness.X))


str(happiness)
```

```
## 'data.frame':    149 obs. of  8 variables:
##  $ Country.name                : Factor w/ 149 levels "Afghanistan",..: 41 34 129 55 97 104 128 79 98
##  $ Ladder.score                : num  7.84 7.62 7.57 7.55 7.46 ...
```

```
##  $ Logged.GDP.per.capita       : num  10.8 10.9 11.1 10.9 10.9 ...
##  $ Social.support              : num  0.954 0.954 0.942 0.983 0.942 0.954 0.934 0.908 0.948 0.934 ..
##  $ Healthy.life.expectancy     : num  72 72.7 74.4 73 72.4 73.3 72.7 72.6 73.4 73.3 ...
##  $ Freedom.to.make.life.choices: num  0.949 0.946 0.919 0.955 0.913 0.96 0.945 0.907 0.929 0.908 ...
##  $ Generosity                  : num  -0.098 0.03 0.025 0.16 0.175 0.093 0.086 -0.034 0.134 0.042 ..
##  $ Perceptions.of.corruption   : num  0.186 0.179 0.292 0.673 0.338 0.27 0.237 0.386 0.242 0.481 ...
```

```r
library(ggfortify)
pca_mat = prcomp(happiness.X, center = T, scale = T)

# Score and loadings plot:
autoplot(pca_mat, data = happiness.X, colour = "Black", loadings = TRUE, loadings.colour = "red",
    loadings.label = TRUE, loadings.label.size = 5, label = T, label.size = 4.5)
```



a) (3P)

(i) Look at the loading directions in the plot above. Describe two characteristics that you observe in the relations between the variables. (2P) (NB: Describe the variable relations that are obvious. If you don't see much relation between certain variables, you don't have to describe that relation.)

(ii) The scores in the above plot can also be used for doing outlier/anomaly detection. Which country can be considered to an outlier among the followings: {Norway, Vietnam, South Korea, Afghanistan, India}? (1P)

b) (4P)

Here, we're going to find out which variables are important by principal component analysis (PCA) and partial least squares regression (PLSR).

Note that we can naturally assume the followings:

- PCA will find out important variables w.r.t explainability of the dataset of the predictors.
- PLSR can find out important variables w.r.t the response in the model, that is, happiness (= `Ladder.score`).

(i) Make a graphical description of the absolute values of the first principal component (= `PC1`) by PCA. You can use a bar plot, or any other graphical description of your choice (see R-hints below). (1P)

(ii) Fit PLSR on `happiness.XY` with a response of `Ladder.score` (= happiness score) and all the remaining variables in that dataset as predictors. (1P)

(iii) Plot a bar graph of the absolute values of the first principal component for `X` (= predictors of `happiness.XY`) by PLSR. Use the same type of plot as in (i) in order to compare. (1P)

(iv) What are the three most important predictor to predict the happiness score based on the PLSR bar graph from (iii)? (1P)

R-hints:

(i)

- Use `data.frame(pca_mat$rotation)$PC1`.
- The x-axis should show the variable names.
- The y-axis should show the `abs(PC1)`. (Note that `abs(PC)` denotes the feature/variable importance.)

(ii)

- Use `plsr_model <- plsr(..., scale=T)`

(iii)

- Use `plsr_model$loadings[,c('Comp 1')]`

c) (2P) - Multiple choice

Say for *each* of them if it is true or false.

(i) K-means is optimizing clusters such that the within-cluster variance becomes large.
(ii) No matter how many times you run K-means clustering, its cluster centroids will always end up in the same locations.
(iii) Strong correlation between predictors allows PCR to be more effective for predicting a response when prediction is made based on the first two principal components.
(iv) We can do outlier/anomaly detection with PCA.

d) (3P)

(i) We are now doing a K-means clusterization. Run the k-means clustering on `happiness.X` given the following condition and visualize the clusters using the code below. (This question is given to let you explore how countries are clustered together based on `happiness.X`. There are multiple answers for `K`. You can use whatever `K` value that satisfies the condition.) (1P)

Condition:

- Norway, Denmark, Sweden, Finland should be in the same cluster, while United States is in a different cluster.

```
K = -1  # your choice
km.out = kmeans(happiness.X, K)

autoplot(pca_mat, data = happiness.X, colour = km.out$cluster, label = T, label.size = 5,
    loadings = F, loadings.colour = "blue", loadings.label = F, loadings.label.size = 3)
```

(ii) Give your interpretation of the clusters w.r.t the happiness score (= `Ladder.score`). One point is given per correct interpretation aspect, -1P for a wrong interpretation. (2P)