Mandatory Assignment 1

October 16, 2019

STK-IN4300, Simen Håpnes

This assignment with code might also be found at my **github**.

1 Exercise 1

Contrast the ability of a lasso and a ridge regression model to predict the Duke CAD index (CADi) from the gene expression.

1.1 Algorithm

The data file which is to be analysed contains 110 patients **with** Coronary Artery Disease.

The X-data contains 22283 various gene expressions, $X \in \mathbb{R}^{110 \times 22,283}$.

The y-data contains the true CADi value , $y \in \mathbb{R}^{110 \times 1}$.

I have written a python code, which uses the packages from Scikit-learn to perform Ridge and LASSO regression. The script to solve the exercise is presented. I will provide explainatory information between the code blocks as well as comments in the actual code.

To begin, I will do the necessary imports:

```
[ ]: %matplotlib inline
     import matplotlib.pyplot as plt # for plotting
     import pandas as pd # for reading csv file
     import numpy as np # array/linear algebra library
     import os # convenient when reading/writing files to other directories
     from sklearn.linear_model import LassoCV, RidgeCV # Lasso and ridge with cross␣
      ↪validation
     from sklearn.model_selection import train_test_split # simple split of train␣
      ↪and test set
     from sklearn.preprocessing import scale # scaling of the data with both mean␣
      ↪and std
```

1.1.1 Definition of functions

The two following functions are meant to easily read a file in the **./data/** directory and to plot/save figures to the **./figures/** directory.

```
[ ]: def read_file(filename):
         """Read a csv data file located in the ./data/ directory
         Return design matrix X and y-data"""
         DATA_DIR = "./data"
         if not os.path.exists(DATA_DIR):
             os.mkdir(DATA_DIR)
         DATA_FILE = os.path.join(DATA_DIR,filename)
         df = pd.read_csv(DATA_FILE).values
         # 0th column contains file names
         y = np.array(df[:,1]) # 1st column is y-data
         X = np.array(df[:,2:]) # remaining columns is the X-data
         return X, y


     def save_fig(y_test, y_pred, score, fig_id):
         """Save a matplotlib figure to the ./figure/ directory.
         If/else statements ask the user if they wish to overwrite an
         already existing figure."""
         # Plot
         l = y_test.shape[0]
         x = np.linspace(1,l,l)
         # plot of y_pred vs. y_test:
         plt.plot(y_test, y_pred, 'ro', label=fr"prediction, $R^2 = ${score:2.2f}")
         # plot of the ideal linear case
         min = np.min(y_test)
         max = np.max(y_test)
         plt.plot([min,max],[min,max])
         # plot options
         plt.xlabel(r"$y$")
         plt.ylabel(r"$\hat{y}$")
         plt.legend()
         plt.grid()
         # Saving of figure
         FIGURE_DIR = "./figures"
         if not os.path.exists(FIGURE_DIR):
             os.mkdir(FIGURE_DIR)
         FIGURE_FILE = os.path.join(FIGURE_DIR, fig_id)
         if os.path.exists(FIGURE_FILE):
             overwrite = str(input("The file \"" + FIGURE_FILE + \
                 "\" already exists, do you wish to overwrite it [y/n]?\n"))
             if overwrite == "y":
                 plt.savefig(FIGURE_FILE, format="png")
```

```
            plt.close()
            print("Figure was overwritten.")
        else:
            print("Figure was not saved.")
    else:
        plt.savefig(FIGURE_FILE, format="png")
        plt.close()
    return None
```

The purpose of the main function is to use scikit-learn to perform Ridge regression and LASSO regression.

Note that X and y are scaled so that both mean = 0 and std = 1.

[ ]:
```
def main():
    """
    Info:
    Perform Ridge and Lasso regression on gene data. Try to predict
    The output which is the Duke CAD index
    (Coronary Artery Disease index).

    Notes:
    Data file "data_E1.csv" contains the 110 pasients with case:
    * The first column contains file IDs (not necessary for analysis)
    * the second column contains the y data: true CADi
    * the remaining columns contains 22283 features
    """
    # Read data from file
    X, y = read_file("data_E1.csv")
    # Scale data with both mean and std: e.g. y = y/std(y) - mean(y)
    std = np.std(y)
    mean = np.mean(y)
    X = scale(X)
    y = scale(y)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)

    # Ridge (fit_intercept=False assumes that data is already scaled)
    clf_r = RidgeCV(fit_intercept=False).fit(X_train, y_train)
    y_pred_r = clf_r.predict(X_test)
    score_r = clf_r.score(X_test, y_test)
    y_test_r = y_test*std + mean
    y_pred_r = y_pred_r*std + mean
    # Plot and save figure
    save_fig(y_test_r, y_pred_r, score_r, "ridge.png")

    # Lasso (fit_intercept=False assumes that data is already scaled
    # Note: cv=5 is the default of a future version of scikit-learn
    clf_l = LassoCV(fit_intercept=False, cv=5).fit(X_train, y_train)
    y_pred_l = clf_l.predict(X_test)
    score_l = clf_l.score(X_test, y_test)
    y_test_l = y_test*std + mean
    y_pred_l = y_pred_l*std + mean
    # Plot and save figure
    save_fig(y_test_l, y_pred_l, score_l, "lasso.png")
```

Run the main function:

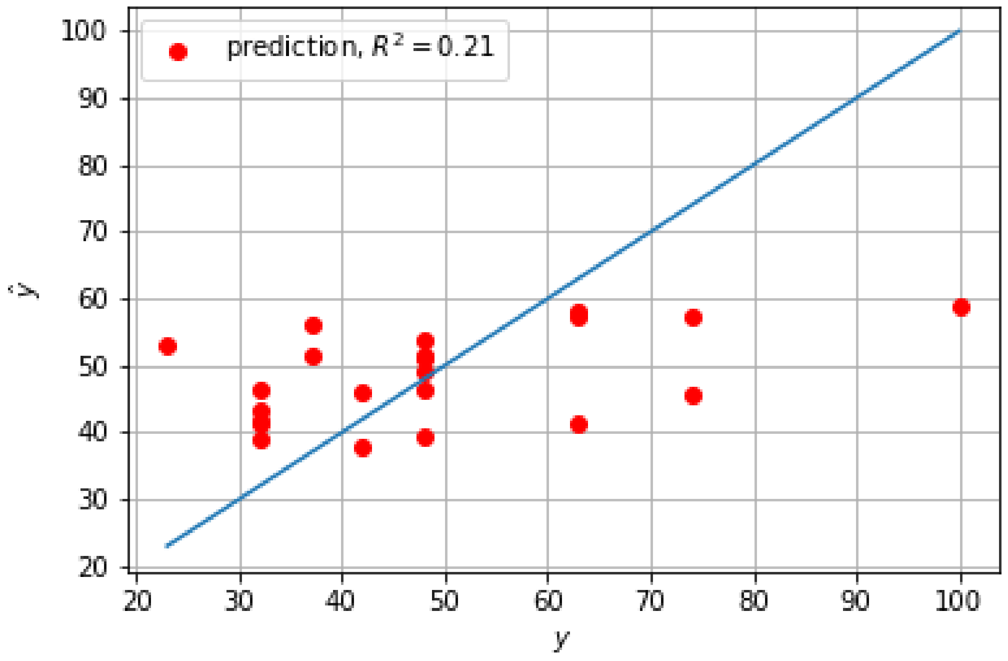
```
[ ]: if __name__ == "__main__":
         np.random.seed(1) # affect the random splitting of train/test set: same␣
      ↪seed obtain the same results each run
         main() # call the main function where the regressions are performed

     # Test run, information about versions:
     # python 3.7.3
     # scikit-learn 0.21.3
     # numpy 1.17.2
     # pandas 0.25.1
     # matplotlib 3.1.1
     """
     $ python main.py
     The file "./figures/ridge.png" already exists, do you wish to overwrite it [y/
      ↪n]?
     y
     Figure was overwritten.
     The file "./figures/lasso.png" already exists, do you wish to overwrite it [y/
      ↪n]?
     y
     Figure was overwritten.
     """
```

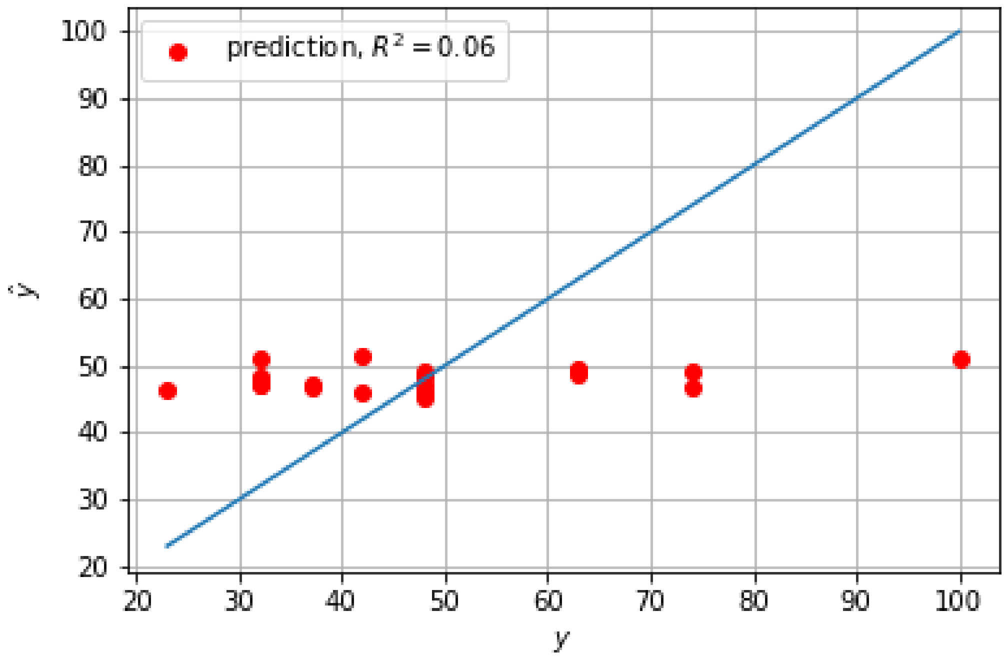
1.2 Results

The blue lines in the figures shows the desired output if the models were perfect $\hat{y} = y$. The red dots shows the model predictions: $\hat{y}$ vs. y.

1.2.1 Ridge Regression



1.2.2 LASSO Regression



1.3 Discussion

The first notable feature which is shown is that Ridge regression provides a better R^2-score than the Lasso regression.

Both models fail to predict values in the same range as the true CADi values. The true values are in the range $[20, 100]$, whereas Ridge regression only provides predictions in the range of roughly $[38, 59]$ and LASSO only provides predictions in the range of roughly $[44, 52]$.

Ridge regression has a higher variance of $\hat{y}$ than LASSO regression and this might also be the problem with this model and this data set: Ridge regression seems to suffer from a too high variance. If this is true, the model complexity is too high, and the *training set* is overfitted, hence do these parameters not fit the *test set* very well.

LASSO regression however has a much lower variance, and it seem to have a very small output range. In the figure with $\hat{y}$ vs. y, the correlation is near linear, but not in the right direction. If the model was perfect, all the predictions would be linear and on top on the blue line (which is $\hat{y} = y$). In this case the predictions appear almost at a flat line $\hat{y} = \mathit{const.} + \sigma$, and this is an indication that this model suffers from a high bias. If this is true, LASSO fails to find a correlation between the gene data and CADi by using this data set. It is also notable that the number of data points (110) might be too low compared to the number of features (22,283).

2 Exercise 2

Consider projection pursuit. Derive the expressions for w which minimizes the linearized expression for the object function

$$S = \sum_{i=1}^{N} g'(w_{old}^T x_i)^2 \left(\frac{y_i - g(w_{old}^T x_i)}{g'(w_{old}^T x_i)} + w_{old}^T x_i - w^T x_i \right)^2$$

In other words, we wish to minimize S wrt. w. First, let's try to rewrite this expression to linear equation by getting rid of the summation sign. It is convenient to identify: * which of the parameters are scalars: y_i, g and g. * which of the parameters are vectors: x_i, w_{old} and w must be vectors of length p.

Now, lets rename some of the expressions in the sum, in order to simplify the problem:

$$a_i = g'(w_{old}^T x_i)^2$$

$$b_i = \frac{y_i - g(w_{old}^T x_i)}{g'(w_{old}^T x_i)} + w_{old}^T x_i$$

Hence the equation is now

$$S = \sum_{i=1}^{N} a_i \left(b_i - w^T x_i\right)^2,$$

The output must be a scalar, therefore

$$S = \sum_{i=1}^{N} a_i \left(b_i - w^T x_i\right)^2 = (b^T - w^T X^T) A (b^T - w^T X^T)^T$$

where A is the diagonal matrix with the a along its diagonal. The matrix X now contains N rows of x_i. Further:

$$(b^T - w^T X^T) A (b - Xw) = b^T A b - b^T A X w - w^T X^T A b + w^T X^T A X w$$

Now lets confirm that all these terms are indeed scalars:

$$\begin{gathered} b^T A b \rightarrow (1 \times N)(N \times N)(N \times 1) \\ b^T A X w \rightarrow (1 \times N)(N \times N)(N \times p)(p \times 1) \\ w^T X^T A b \rightarrow (1 \times p)(p \times N)(N \times N)(N \times 1) \\ w^T X^T A X w \rightarrow (1 \times p)(p \times N)(N \times N)(N \times p)(p \times 1) \end{gathered}$$

Since all terms start and end with 1, the products are (1×1) which are scalars.

Notice that the the second term, transposed is $(b^T A X w)^T = w^T X^T A b$ which is the third term. Since the transpose of a scalar yields the same scalar these two terms are identical. The expression is now

$$S = b^T A b - 2 b^T A X w + (Xw)^T A X w$$

Now, to obtain the minimum, we wish to differentiate wrt. w and set this equal to 0:

$$\begin{gathered} \frac{\partial}{\partial w} \left(b^T A b - 2 b^T A X w + (Xw)^T A X w \right) = 0 \\ -2 b^T A X + 2 X^T A X w = 0 \end{gathered}$$

Now, the first term is $(1 \times p)$ and the second term is $(p \times n)$, I therefore transpose the first term to get the expressions on the same form:

$$\begin{gathered} b^T A X = X^T A X w \\ X^T A b = X^T A X w \\ w = (X^T A X)^{-1} X^T A b \end{gathered}$$

Hence the solution is

$$\underset{w}{\operatorname{argmin}}(S) = \left(X^T A X\right)^{-1} X^T A b$$