Beyond OLS : Ridge, Lasso and Logistic Regression

Exercise 1
In this dataset, we are going to predict the salary of Major League Baseball players in US depending on several parameters. The dataset can be downloaded and is described on the following website : `https://www.kaggle.com/floser/hitters`
We shall compare different version of the linear model to solve this regression task.

1. We first import and preprocess the data
   (a) Import the data. Remove missing values using the function `dropna()` of the library `pandas` to obtain the dataframe `df_clean`
   (b) Display the columns corresponding to the features `'League'`, `'Division'` and `'NewLeague'`. What do you see? Transform these categorical variables into indicators
   Hint : see `https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.get_dummies.html`
   (c) We now define features and target. Comment the following code
   ```
   X_ = df_clean.drop(['Salary', 'League', 'Division', 'NewLeague'], axis
   = 1).astype('float64')
   X = pd.concat([X_, dummies[['League_N', 'Division_W', 'NewLeague_N']]],
   axis = 1)
   y = df_clean.Salary
   ```
   Split data into train and test.
2. Fit a classical linear regression on the train set. Display the coefficient of the regression and calculate the r2 score on the test set. Comment
3. We now improve OLS with ridge regression
   (a) Fit a ridge regression on the training data with hyperparameter $\alpha = 4$. Display the coefficient of the regression and calculate the r2 score on the test set.
   (b) Same with $\alpha = 20$. Compare
   (c) We now allow α to vary from $5 \cdot 10^{-4}$ to $5 \cdot 10^{3}$. Display the coefficient of the regression and calculate the r2 score on the test set. Comment
4. We now improve OLS with lasso
   (a) Fit a lasso regression on the training data allowing the hyperparameter α to vary from $5 \cdot 10^{-4}$ to 5. Display the coefficient of the regression and calculate the r2 score on the test set. Comment
   (b) Use now cross-validation to thune α. Comment
   Hint : see `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LassoCV.html`

Exercise 2
In this exercise, we are going to predict diabetes using Logistic Regression Classifier. We want to evaluate the performance of Logistic Regression on this classification task.

1. Load the dataset on the following website and store it in the dataframe `pima`
   `https://www.kaggle.com/uciml/pima-indians-diabetes-database`

2. We now split dataset in features and target variable
   ```
   feature_cols = ['Pregnancies','Insulin','BMI', 'Age','Glucose','BloodPressure','DiabetesPedigreeFunction']
   X = pima[feature_cols]
   y = pima.Outcome
   ```
   Split into train and test.

3. Fit a logistic regression on the train set.

4. We now want to evaluate the model on the test set. We recall that :

   - The confusion matrix is the matrix

   $$M = \begin{pmatrix} TN & FP \\ FN & TP \end{pmatrix}$$

   where TP : True Positive, TN : True Negative, FP : False Positive, FN : False Negative. We recall the notion of precision and recall

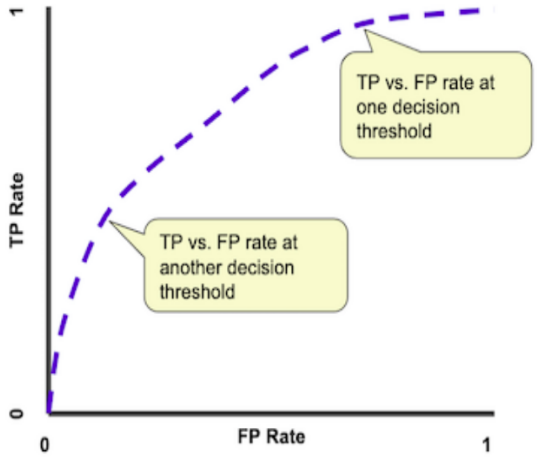
   $$\text{Precision} = \frac{TP}{TP + FP}, \text{ Recall} = \frac{TP}{TP + FN}$$

   - The accuracy is then defined as

   $$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

   Estimate the confusion matrix, accuracy precision and recall using the functions `confusion_matrix`, `accuracy_score`, `precision_score` and `recall_score` of the library `metrics`. Comment the relevance of these metrics.

5. Logistic regression outputs class membership probabilities instead of binary output. We can then deduce from this output the class of the observation we need to set a threshold and define the ROC (receiver operating characteristic curve) graph which shows the performance of a classification model at all classification thresholds.



Plot this ROC curve using the function `roc_curve` of the library `metrics`. Comment.