Project 1: Linear Regression

(First discussion: Sept 25; Last questions: Oct 9; Deadline: Oct 16; In charge: Jean-Loup Dupret)

This project explores linear regression and associated regularization techniques. You will work with the *Housing* dataset, containing 1460 observations and 79 explanatory variables describing various aspects of residential homes in Ames, Iowa, USA. Among these features, 37 are numerical and 42 are categorical. The aim of this project is to explain and predict the target variable *SalePrice*, giving the house value in \$. You can refer to the file *description.txt* for a complete description of all variables in the dataset.

1. This first question will help you import the dataset into Python in the correct format.

   a) Import the dataset *Housing.csv* in Python as a pandas DataFrame.

   b) Determine graphically whether the target variable *SalePrice* has a Gaussian distribution. If not, suggest a suitable transformation to bring *SalePrice* close to a Gaussian distribution and apply this transformation to the dataset.

   c) Use one-hot encoding for the categorical features and replace missing numeric values (NaN's) with the mean of their respective columns. **Hint**: *To implement one-hot encoding in Python, you can use the function 'get_dummies' from the pandas package.*

   d) Based on the original DataFrame *Housing*, create a second pandas DataFrame *Housing2* excluding all categorical variables.

2. The second question helps you to build a linear regression model to predict the (transformed) variable *SalePrice* using the DataFrame *Housing2* based only on the 37 numerical variables.

   a) Since the linear regression model has to be evaluated on a different dataset than the one used for training, split the *Housing2* data into two subsets: $(X, Y)_{train}$ and $(X, Y)_{test}$. Randomly assign 70% of the observations to the training set and the remaining 30% to the test set.

   b) Fit a linear regression model on the training dataset using the *sklearn* Python package and present a table with the regression coefficients for each feature. Compare the in-sample and out-of-sample Mean Squared Error (*MSE*) and R^2.

   c) The *sklearn* package does not provide standard errors for the estimated regression coefficients, which are essential tools to assess the statistical precision of an estimate[1]. Therefore, you will now use matrix algebra in Python with the *numpy* package to compute the standard errors of the estimated coefficients $\hat{\beta}$.

      (i) Compute the estimated coefficients $\hat{\beta}$ using the *numpy* package with

      $$\hat{\beta} = (A^\top A)^{-1} A^\top y\,,$$

      where A is the design matrix (which needs to include a column of ones for the intercept term) and y the observed values of the target variable. Note that $\hat{\beta}_0$ denotes the estimate of the intercept.

[1] Under the assumption that the error terms are normally distributed with constant variance across all levels of the explanatory variables (homoscedasticity), the standard errors are normally distributed, which allows us to derive confidence intervals for the $\hat{\beta}$-parameters and determine whether they are significantly different from zero.

(ii) Compute the standard error of each coefficient $\hat{\beta}_j$, $j = 0, \ldots, d$, using

$$SE(\hat{\beta}_j) = \sqrt{\frac{\sum_{i=1}^{m}(y_i - \hat{y}_i)}{m - (d+1)} a_j}\,,$$

where a_j denotes the j^{th} element of $\text{diag}((A^\top A)^{-1})$, m the number of observations in the training set, d the number of explanatory variables (without the intercept) and $\hat{y}$ the predicted values from the linear regression.

(iii) Compute the (in-sample) MSE and R^2 using matrix algebra in Python.

(iv) Do the results change using pseudoinversion instead of standard matrix inversion in (i) and (ii)?

(v) Confirm your results using the OLS function from the *statsmodels* package.

d) Suggest instead a second-order polynomial regression model (including all quadratic and mixed terms) to predict *SalePrice*. Does this improve the linear regression model 2.b)? What happens with higher-order polynomial regression models?

3. In this question, you will implement regularization techniques and compare their performance to linear regression. Consider now the full *Housing* DataFrame, including the categorical (dummy) variables. Ensure that the training and test sets are again split as in Question 2 based on the same observation indices.

a) Perform a linear regression for the (transformed) variable *SalePrice* using the *Housing* DataFrame. How do the in-sample and out-of-sample MSE and R^2 metrics compare to the results of Question 2.b) on the *Housing2* DataFrame?

b) Implement the truncated pseudoinverse, Ridge and Lasso regularization techniques. Use 8-fold cross-validation to tune the hyperparameters of each regularization technique based on the MSE metric. Compare their performance in terms of in-sample and out-of-sample MSE with the linear regressions of Questions 2.b) and 3.a).

c) For the Lasso regularization technique, how many coefficients are non-zero ($\hat{\beta}_j \neq 0$)? Compare this number with the number of coefficients retained by the Ridge and truncated pseudoinverse techniques and provide an explanation.

d) Based on your findings from Questions 2 and 3, which model would you recommend? Justify your choice.