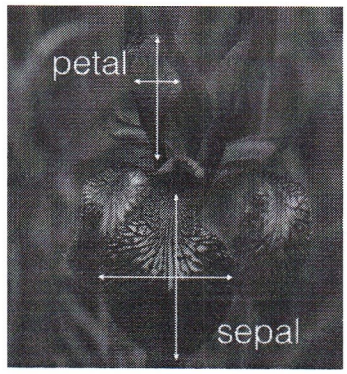
01 - Linear Regression

Let us consider the iris dataset.

In the dataset we have data regarding specific species of flowers :

- Sepal length;
- Sepal width;
- Petal length;
- Petal width;
- Species (*Iris setosa*, *Iris virginica* e *Iris versicolor*).

In the specific, we have N = 150 total samples (50 per class).



Loading

We need to import **matplotlib** and **pandas** to handle data and plots.

In [1]:
```
import pandas as pd
from pandas.plotting import scatter_matrix
import matplotlib.pyplot as plt
```

We can find the dataset we need to analyse online. We use pandas to load the csv to a **pandas.DataFrame**.

In [2]:
```
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
dataset = pd.read_csv(url, names=names)
```

We can start to have a look the data we have

In [3]:
```
dataset.head()
```

Out[3]:

	sepal-length	sepal-width	petal-length	petal-width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

we do not care about the flower species in this lesson, hence we remove that column:

In [4]:
```
dataset = dataset.drop('class', axis=1)
```

We will try to understand how the feature are distributed, by printing some statistics:

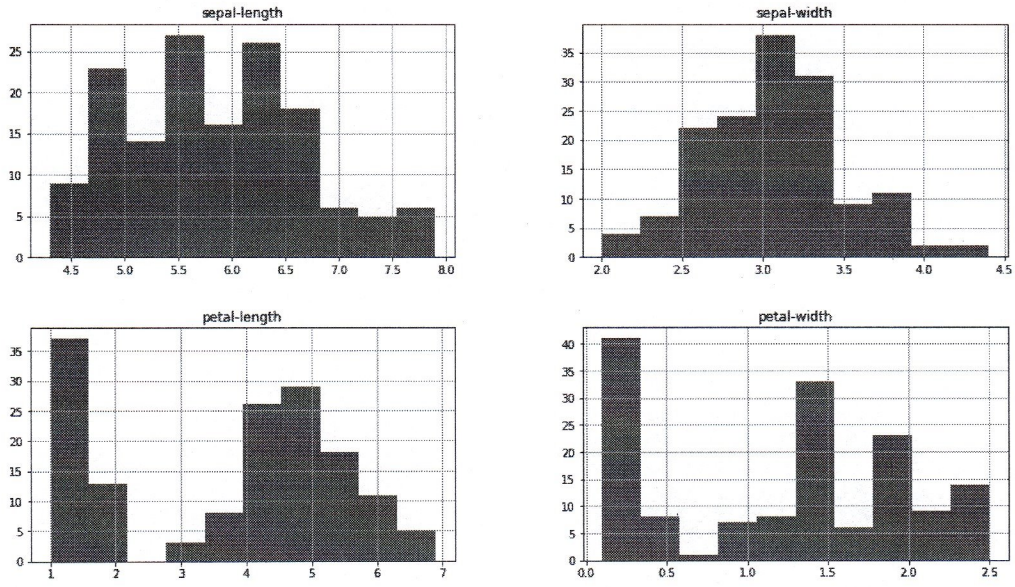
In [5]:
```
dataset.describe()
```

Out[5]:

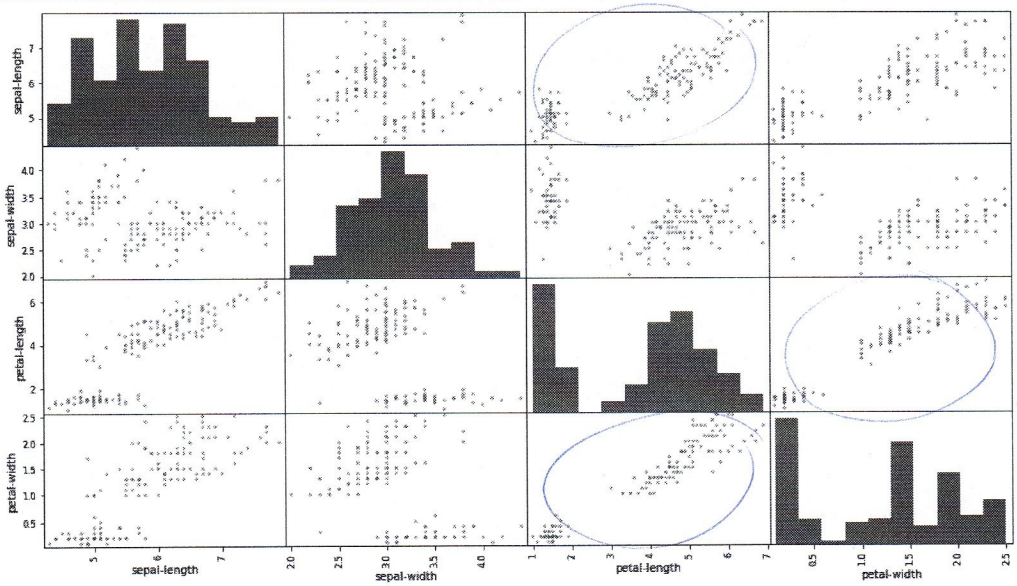
	sepal-length	sepal-width	petal-length	petal-width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

Visualizing data can also be very helpful:

In [6]:
```
dataset.hist(figsize=(16,9))
plt.show()
```



In [7]:
```
scatter_matrix(dataset, figsize=(16, 9))
plt.show()
```

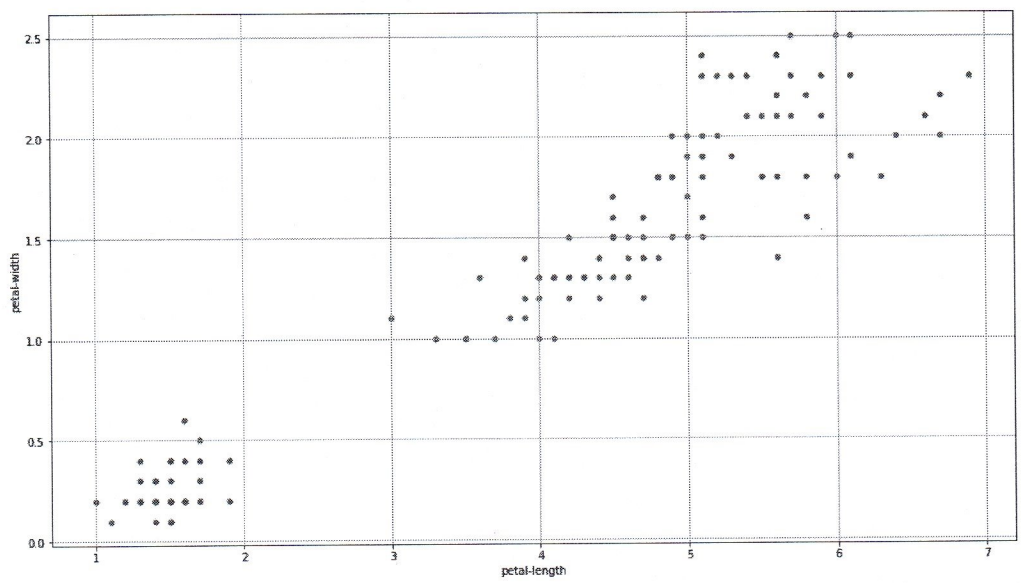


we can see that among some features there is a trend

petal-lenght and *petal-width* seem to have a strong relationship... we should investigate it more in detail!

In [8]:
```
dataset.plot.scatter('petal-length', 'petal-width', grid=True, figsize=(16,9))
```

Out[8]: `<matplotlib.axes._subplots.AxesSubplot at 0x7f7db7c18410>`



Preprocessing

Once we inspected the data, we should operate some preprocessing procedures. On a generic dataset one should perform:

- shuffling; (data might have an order that we're not interested about)
- remove inconsistent data; (data that make no sense)
- remove outliers;
- normalize or standardize data; (in the case of linear regression we do it to not have "crazy" weights. If the y is 0,00... and one feature is in millions, the weight may be very small or, viceversa, very large (if y and the feature are exchanged))
- fill missing data.

In this case we are going to use the entire dataset, with a non-iterative method, hence we do not need to **shuffle**.

There seems not to be **outliers** from previous inspection.

Is there any **missing data**?

In [9]:
```
import numpy as np
```

In [10]:
```
np.any(np.isnan(dataset.values))
```

Out[10]:
```
False
```

we are lucky, no missing data, no outliers...

However it is always better to work with data in the same scale, hence we should normalize the columns we are going to use.

$$s \leftarrow \frac{s - \bar{s}}{S}$$ (standard deviation)

$$s \leftarrow \frac{s - \bar{s}}{\max_n\{s_n\} - \min_n\{s_n\}}$$

The **zscore** function operates a standardization of its inputs.

In [11]:
```
from scipy.stats import zscore
```

In [12]:
```
x = zscore(dataset['petal-length'].values).reshape(-1, 1) # we reshape our feature column as a (n_sample, n_features) matrix
y = zscore(dataset['petal-width'].values)
```
(without the reshape we just have an array of values)

In [13]:
```
x.shape
```

Out[13]:
```
(150, 1)
```

Using Scikit-Learn Toolbox

A linear model seems to be a good choice to predict *petal-width* given petal-length, let's use **scikit-learn** tools to do a linear regression:

In [14]:
```
from sklearn import linear_model
```

In [15]:
```
lin_model = linear_model.LinearRegression()
lin_model.fit(x, y)
```

Out[15]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

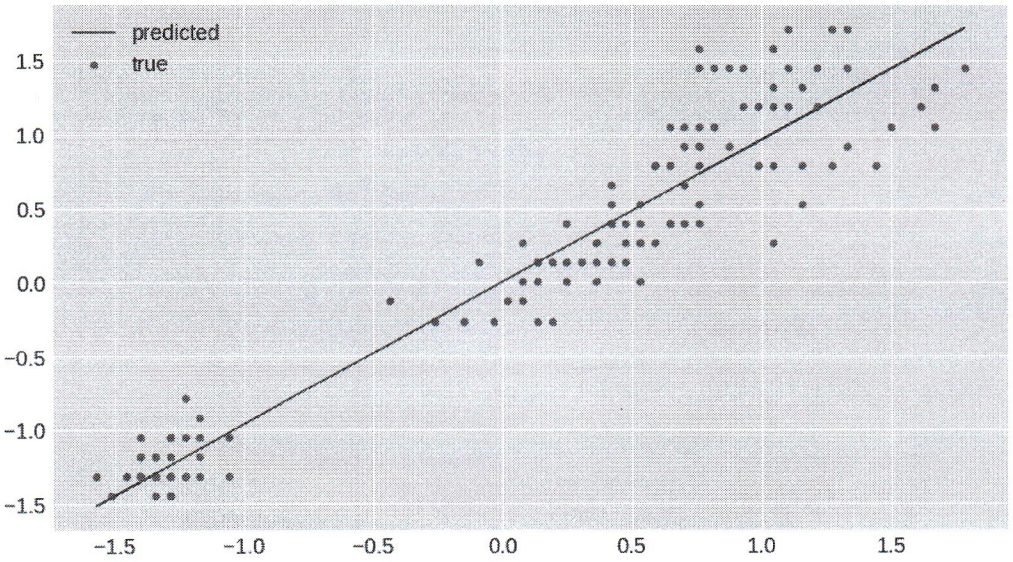
*

since we want to customize our plot, we will use matplotlib directly this time:

In [16]:
```
with plt.style.context('seaborn'): # use your favorite style, if you don't like the standard one
 plt.figure(figsize=(16,9))
 plt.scatter(x, y, label='true')

 w1 = lin_model.coef_ # weights of the model are stored here
 w0 = lin_model.intercept_ # and here it is the intercept

 # Compute the y component of the regression line

 y_pred = lin_model.predict(x)
 #y_pred = [w1 * sample + w0 for sample in x.flatten()]
 # (we used a list comprehension here, have a look to the python tutorial
 #  if you don't know what it is!)

 plt.plot(x, y_pred, label='predicted', color='red')

 # enlarging fonts
 plt.legend(prop={'size': 20})
 plt.xticks(fontsize=20)
 plt.yticks(fontsize=20)

 plt.show()
```

* we can write: "linear_model.__dict__" to obtain some informations:
{'residues': 10.9648,
 'coeff': array([0.96275717]),
 ... :

(__dict__: method that every python object has; it is needed to inspect the object)



To evaluate the quality of our regression we can analyse some metrics:

In [17]:
```
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.feature_selection import f_regression
```

Residual Sum of Squares

$RSS = \sum_n (\hat{t}_n - t_n)^2$, it tells us how much of the prediction differs from the true value. (total error of the prediction)

In [18]:
```
lin_model._residues
```

Out[18]: 10.964815811699454

Coefficient of determination

$R^2 = 1 - \frac{RSS}{\sum_n (\bar{t} - t_n)^2}$, it tells us the fraction of the variance of the data explained by the model (how much better we are doing w.r.t. just using the mean of the target $\bar{t} = \frac{\sum_n t_n}{N}$).

In spaces with a single feature this is equal to the correlation coefficient between the input and the output;

For a more detailed explanation: https://en.wikipedia.org/wiki/Coefficient_of_determination

In [18]:
```
```

In [19]:
```
r2_score(y, y_pred)
```

Out[19]: 0.9269012279220037

Mean Squared Error

$MSE = \frac{\sum_n (\hat{t}_n - t_n)^2}{N}$, it tells approximately how much error we get on a predicted data over the training set (i.e., a normalized version of the RSS).

In [20]:
```
mean_squared_error(y, y_pred)
```

Out[20]: 0.07309877207799637

Under the assumption that the observations t_n are i.i.d. and satisfies $t_n = w_0 + \sum_j w_j x_{nj} + \epsilon$, where ϵ is a Gaussian noise with zero mean and variance σ^2 (i.e., the data are generated by a linear model with noise), the computed coefficients $\hat{w}_j$ are distributed as follows:

$$\frac{\hat{w}_j - w_j}{\hat{\sigma}\sqrt{v_j}} \sim t_{N-M-1}$$

where w_j is the true parameter, $\hat{\sigma}$ is the unbiased estimated for the target variance, i.e., $\hat{\sigma}^2 = \frac{\sum_n (t_n - \bar{t}_n)^2}{N-M-1}$, v_j is the j-th diagonal element of the matrix $(X^T X)^{-1}$ and t_{N-M} is the t-student distribution with $N - M - 1$ degrees of freedom.

N = # samples
M = # features
(+1 because of the intercept)

This allow us to formulate some **statistical tests**:

1. ## Single coefficients statistical test:

$$H_0 : w_j = 0 \qquad \text{vs.} \qquad H_1 : w_j \neq 0$$

$$t_{stat} = \frac{\hat{w}_j - w_j}{\hat{\sigma}\sqrt{v_j}} \sim t_{N-M-1}$$

where t_{N-M-1} is the T-Student distribution with $N - M - 1$ degrees of freedom

2. Overall significance of the model: F-statistic

It considers the following hypothesis test:

$$H_0 : w_0 = w_1 = \cdots = w_M = 0 \text{ vs. } H_1 : \exists w_j \neq 0$$

The F-statistic can be computed and is distributed as follows:

$$F = \frac{dfe}{M-1} \frac{\sum_n (\hat{t}_n - t_n) - RSS}{RSS} \sim F_{M-1,N-M}$$

where $F_{M-1,N-M}$ is the Fisher-Snedecor distribution with parameters $M-1$ and $N-M$.

In [21]:
```
f_regression(x, y) # it outputs a tuple: (value of the F-statistics, its p-value)
```

Out[21]:
```
(array([1876.65781288]), array([5.77666099e-86]))
```

If one wants all the information about the output of a linear model in a single instruction, just use the library **statsmodels** and use the function **summary()** on the result of the Ordinary Least Square optimization procedure

In [22]:
```
from statsmodels import api as sm
lin_model2 = sm.OLS(x, y).fit()
print(lin_model2.summary())
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testi
ng is deprecated. Use the functions in the public API at pandas.testing instead.
  import pandas.util.testing as tm
                                 OLS Regression Results
==============================================================================================
Dep. Variable:                          y   R-squared (uncentered):                   0.927
Model:                                OLS   Adj. R-squared (uncentered):              0.926
Method:                     Least Squares   F-statistic:                              1889.
Date:                    Wed, 30 Jun 2021   Prob (F-statistic):                    1.56e-86
Time:                            20:54:14   Log-Likelihood:                         -16.645
No. Observations:                     150   AIC:                                      35.29
Df Residuals:                         149   BIC:                                      38.30
Df Model:                               1
Covariance Type:                nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
x1             0.9628      0.022     43.467      0.000       0.919       1.007
==============================================================================
Omnibus:                        2.326   Durbin-Watson:                   1.437
Prob(Omnibus):                  0.313   Jarque-Bera (JB):                1.852
Skew:                           0.210   Prob(JB):                        0.396
Kurtosis:                       3.347   Cond. No.                         1.00
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Custom Implementation

We can also implement Least-Squares from scratch, using its closed-form:

$$\hat{\mathrm{w}}_{OLS} = (\Phi^\top \Phi)^{-1} \Phi \, \mathrm{t}, \tag{1}$$

where $\Phi = (\phi(x_1), \ldots, \phi(x_N))^\top$ and $\mathrm{t} = (t_1, \ldots, t_N)^\top$.

features matrix — target vector

By using **numpy**:

In [23]:
```
from numpy.linalg import inv

n_samples = len(x)
Phi = np.ones((n_samples, 2))
Phi[:, 1] = x.flatten() # the second column is the feature
# the field 'T' represents the transposed matrix, @ is the matrix product, the method 'dot' is the matri
mpinv = inv(Phi.T @ (Phi)).dot(Phi.T);
w = mpinv.dot(y);
```

matrix product

In [24]:
```
w
```

Out[24]:
```
array([-3.19189120e-16,  9.62757097e-01])
```

Regularization

If we need to mitigate over-fitting effects in a model we might resort to some regularization techniques, like Ridge regression or Lasso regression.

Ridge Regression

Linear least squares with l2 regularization.

In [25]:
```
ridge_model = linear_model.Ridge(alpha=10)
ridge_model.fit(x, y)
```

Out[25]:
```
Ridge(alpha=10, copy_X=True, fit_intercept=True, max_iter=None, normalize=False,
      random_state=None, solver='auto', tol=0.001)
```

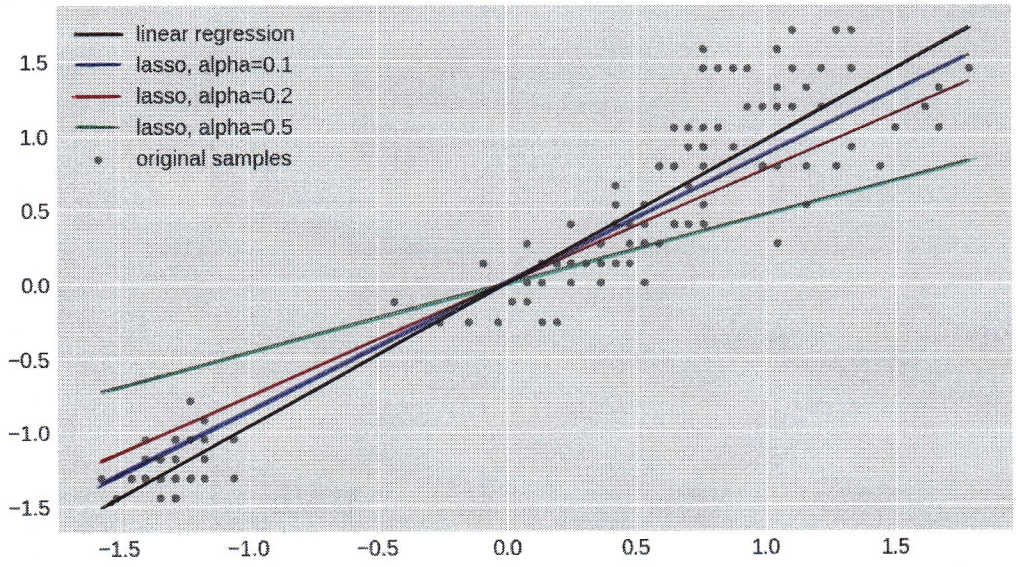
Lasso Regression

Linear Model trained with L1 prior as regularizer.

In [26]:
```
lasso_model = linear_model.Lasso(alpha=10)
lasso_model.fit(x, y)
```

Out[26]:
```
Lasso(alpha=10, copy_X=True, fit_intercept=True, max_iter=1000, normalize=False,
      positive=False, precompute=False, random_state=None, selection='cyclic',
      tol=0.0001, warm_start=False)
```

In [27]:
```
with plt.style.context('seaborn'):
 plt.figure(figsize=(16,9))
 plt.scatter(x, y, label='original samples')
 y_linear = [lin_model.coef_ * x_i + lin_model.intercept_ for x_i in x]
 plt.plot(x, y_linear, label='linear regression', color='red')
 for alpha in [0.1, 0.2, 0.5]:
   # lasso regression
   lasso_model = linear_model.Lasso(alpha=alpha)
   lasso_model.fit(x, y)
   y_lasso = [lasso_model.coef_ * x_i + lasso_model.intercept_ for x_i in x]
   plt.plot(x, y_lasso, label='lasso, alpha={}'.format(alpha))


 # enlarging fonts
 plt.legend(prop={'size': 20})
 plt.xticks(fontsize=20)
 plt.yticks(fontsize=20)

 plt.show()
```



Homeworks

Here we propose some exercises in python for you. They are not mandatory, but they can be helpful to better understand the contents of the lecture, by giving you the opportunity to develop some code by yourself.

1) Predicting petal width

Consider again the Iris dataset, and complete the following code, by writing a script which is able to predict the petal width by using, this time, **all** the other features as input.

In [ ]:
```
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
names = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'class']
dataset = pd.read_csv(url, names=names)

# Get input and output
x = ### WRITE YOUR CODE HERE ###
y = zscore(dataset['petal-width'].values)

# Fit your model
### WRITE YOUR CODE HERE ###
```

Comment on the parameters we would like to introduce or exclude from the prediction process.

Does this model is better than the one trained with a single input?

How do you check if the two models are significantly different from each other?

(hint: look at the exercise session on Bias-Variance tradeoff)

2) Implementing closed-form ridge regression

Ridge regression can be obtained in closed form, as we have seen at lesson. Implement it by yourself, by completing the code below.

In [ ]:
```
alpha = 100
ridge_model = linear_model.Ridge(alpha=alpha)
```

```
ridge_model.fit(x, y)


w = ### WRITE YOUR CODE HERE ###

# Compare your solution it with the scikit-learn one!
assert np.isclose(w, ridge_model.coef_), 'Something wrong!, try again...'
```

In [ ]:
```
### Solution - TO BE REMOVED
# N = len(x)
# Phi = np.ones((N, 2))
# Phi[:, 1] = x.flatten() # the second column is the feature
# # the field 'T' represents the transposed matrix, the method 'dot' is the matrix product
# ridge_mpinv = inv(alpha * np.eye(2) + Phi.T.dot(Phi)).dot(Phi.T);
# w = ridge_mpinv.dot(y);
```

3) Implementing LS for multiple outputs

We have seen at lesson that LS is possible also when we have multiple outputs.

Implement it by extending the LS custom implementation that we have seen.

In [ ]:
```
### WRITE YOUR CODE HERE ###
```

4) Try it on another dataset

Try to repeat the procedure that we have seen for the Iris dataset on a new dataset of your choice:

- select a dataset (many are available online, e.g. https://www.kaggle.com/datasets)
- visualize data, in order to spot interesting relationships
- preprocess data
- apply linear regression

In [ ]:
```
### WRITE YOUR CODE HERE ###
```

2 Linear Regression

Exercise 2.1

Given the relationship:

$$S = f(TV, R, N),$$

where S is the amount of sales revenue, TV, R and N are the amount of money spent on advertisements on TV programs, radio and newspapers, respectively, explain what are the:

1. Response;
2. Independent variables;
3. Features;
4. Model.

Which kind of problem do you think it is trying to solve?

✗ Exercise 2.2

Which one/ones of the following is a good definition of Machine Learning? Motivate your answer.

✓ 1. A computer program is said to learn from experience with respect to some class of tasks and performance measure, improves with experience;

✗ 2. Machine Learning are all the techniques using relationship to provide prediction or to suggest the action to perform in practical situations;

✓ 3. Machine Learning is the sub-field of Artificial Intelligence where the knowledge comes from the use of experience to perform induction;

✗ 4. Machine Learning is the process of creating new information to provide meaningful suggestions to human beings;

✓ 5. Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.

X Exercise 2.3

Explain why the following problems can or cannot be addressed by Machine Learning (ML) techniques:

√ 1. Partition a set of employees of a large company;

X 2. Fortune-telling a person information about her/his personal life;

X 3. Determine the truthfulness of a first order logic formula;

X 4. Compute the stress on a structure given its physical model;

√ 5. Provide temperature predictions.

In the case the problem can be addressed by ML, provide a suggestion for the technique you would use to solve the problem. (hint: wait until the end of this course to answer these questions)

X Exercise 2.4

Categorize the following ML problems:

1. Predicting housing prices for real estate;
2. Identify inside trading among stock market exchange;
3. Detect interesting features from an image;
4. Determine which bird species is/are on a given audio recording;
5. Teach a robot to play air hockey;
6. Predicting tastes in shopping/streaming;
7. Recognise handwritten digits;
8. Pricing goods for an e-commerce website.

For each one of them suggest a set of features which might be useful to solve the problem and a method to solve it.

Exercise 2.5

Why is linear regression important to understand? Select all that apply and justify your choice:

1. The linear model is often correct;

2. Linear regression is extensible and can be used to capture nonlinear effects;
3. Simple methods can outperform more complex ones if the data are noisy;
4. Understanding simpler methods sheds light on more complex ones;
5. A fast way of solving them is available.

Exercise 2.6

Consider a generic regression model. Tell if one should consider the LS method as a viable option in each one of the following 4 different situations. Motivate your answer.

1. Small number of parameters;
2. The loss function is $L(\mathbf{w}|x_n, t_n) = |y(x_n, \mathbf{w}) - t_n|$;
3. Huge number of samples;
4. The loss function is $L(\mathbf{w}|x_n, t_n) = \begin{cases} (y(x_n, \mathbf{w}) - t_n)^2 & \text{if} |y(x_n, \mathbf{w}) - t_n| < \delta \\ |y(x_n, \mathbf{w}) - t_n| & \text{if } |y(x_n, \mathbf{w}) - t_n| > \delta \end{cases}$

Exercise 2.7

Consider a linear regression with input x, target t and optimal parameter θ^*.

1. What happens if we consider as input variables x and $2x$?
2. What we expect on the uncertainty about the parameters we get by considering as input variables x and $2x$?
3. Provide a technique to solve the problem.
4. What happens if we consider as input variables x and x^2?

Motivate your answers.

* Exercise 2.8

Consider a data set in which each data point $(\mathbf{x}_n, t_n)$ is associated with a weighting factor $r_n > 0$, so that the error function becomes:

$$J(\mathbf{w}) = \frac{1}{2N} \sum_{n=1}^{N} r_n (\mathbf{w}^\top \mathbf{x}_n - t_n)^2$$

Find an expression for the solution that minimizes this error function. Give two alternative interpretations of the weighted sum-of-squares error function in terms of (i) data

dependent noise variance and (ii) replicated data points.

Exercise 2.9

Consider an initial parameter $\mathbf{w}^{(0)} = [0\ 0\ 1]^\top$ and a loss function of the form:

$$J(\mathbf{w}) = \frac{1}{2N}\sum_{n=1}^{N}(\mathbf{w}^\top \mathbf{x}_n - t_n)^2.$$

Derive the update given from the gradient descent for the datum $\mathbf{x}_1 = [1\ 3\ 2]^\top$, $t_1 = 4$, and a learning rate $\alpha = 0.3$.

What changes if we want to perform a batch update with $K = 10$ data?

Exercise 2.10

After performing Ridge regression on a dataset with $\lambda = 10^{-5}$ we get one of the following one set of eigenvalues for the matrix $(\Phi^T\Phi + \lambda I)$:

1. $\Lambda = \{0.000000000178,\ 0.014,\ 12\}$;
2. $\Lambda = \{0.0000178,\ -0.014,\ 991\}$;
3. $\Lambda = \{0.0000178,\ 0.014,\ 991\}$;
4. $\Lambda = \{0.0000178,\ 0.0000178,\ 991\}$.

Explain whether these sets are plausible solutions or not.

Exercise 2.11

We run a linear regression and the slope estimate is $\hat{w}_k = 0.5$ with estimated standard error of $\hat{\sigma}\ v_k = 0.2$. What is the largest value of w for which we would NOT reject the null hypothesis that $\hat{w}_1 = w$? (hint: assume normal approximation to t distribution, and that we are using the $\alpha = 5\%$ significance level for a two-sided test).

Exercise 2.12

Which of the following statements are true? Provide motivations of your answers.

1. The estimate w_1 in a linear regression for many variables (i.e., a regression with many predictors in addition to x_1) is usually a more reliable measure of a causal relationship than w_1 from a univariate regression on X_1;
2. One advantage of using linear models is that the true regression function is often linear;

3. If the F-statistic is significant, all of the predictors have statistically significant effects;

4. In a linear regression with several variables, a variable has a positive regression coefficient if and only if its correlation with the response is positive.

Exercise 2.13

Let us assume that the solution with the LS method of a regression problem on a specific dataset has as a result:

$$\hat{t} = 5 + 4x.$$

We would like to repeat the same regression with a Gaussian Bayesian prior over the parameter space $[w_0, w_1]$ with mean $\mu = [3, 2]^T$ and covariance matrix $\sigma^2 = I_2$ (i.e., an identity matrix of order 2).

Which one/ones of the following paramters $\mathbf{w}$ is/are consistent solution/solutions to the previous regression problem with the previously specified Bayesian prior?

1. $\mathbf{w} = [5, 4]$;
2. $\mathbf{w} = [4, 3]$;
3. $\mathbf{w} = [6, 5]$;
4. $\mathbf{w} = [3, 2]$.

* Exercise 2.14

Derive the analytical solution for the *Ridge Regression*. We remember that it considers as loss function the following one:

$$J(\mathbf{w}) = \sum_{n=1}^{N} (\mathbf{w}^T x_n - t_n)^2 + \lambda ||\mathbf{w}||_2^2$$

Derive the gradient descent scheme for the Ridge Regression.

Answers

Answer of exercise 2.1

In the proposed relationship we have:

1. the response (or target or output) is the amount of sales S;
2. the independent variables (or input) are TV, R and N;
3. the features (or input) are TV, R and N ;
4. the model is identified by the function $f(\cdot)$.

Since the amount of sales S is a continuous and ordered variable, we are trying to solve a regression problem (supervised learning).

Answer of exercise 2.2

1. OK: this is one of the most formal definition for ML, provided by Mitchell, highlighting the central role of experience to solve a problem;
2. KO: many application fields uses models and relationships to solve problems. For instance, some techniques uses physical model to provide predictions;
3. OK: here we underline that the experience is able to generate some models which are used to perform induction and use the learned models to generalize;
4. KO: ML does not generate new information. The ML techniques are only able to gather the most interesting information from raw data;
5. OK: this is one of the more informal definition for ML, provided by Samuel, highlighting that with ML we are able to make inference from data without an explicit implementation of the procedure.

ML techniques are not the only ones that uses relationships, e.g. also physics uses relationships to retrieve infos/predictions

ML doesn't create new informations, it tries to explain the already existing ones

we don't start (= we don't program) with all the ratings (the weights are determined ...)

Answer of exercise 2.3

1. The problem of dividing into categories the employee of a company can either be a machine learning problem (e.g., if we do not know the criterion used to partition) or not (e.g., if we are given the criteria). In the first case, we would use some unsupervised ML techniques (e.g., clustering) by considering for instance the personal data of each employee.
2. In principle, fortune-telling is not science. If you consider it as a process where a fortune-teller is able to infer information about a person by looking at her/him,

we could use a ML algorithm to determine the important features to infer information from a person picture (feature selection) and another algorithm able to couple the appropriate phrase to provide to each person (classification). For instance a "magic" methodology has been used in the past to predict what a person is thinking `http://www.google.com/patents/US20060230008`;

3. In the case we have a logical formula and we simply want to derive if it is true or false, it would be unnecessarily complicate to use a ML algorithm, since it exists a deterministic procedure to provide the answer in a fast way.

4. The computation of the stress consists in the application of the mathematical or physical model to a specific case. Here, we might resort to analytical solutions, in the case they exists, or techniques in the field of scientific computing, where an approximation is given by using a discretization scheme.

5. In this case, since we want to predict a continuous value (e.g., temperature), we are considering a regression problem. One might consider the use of linear regression models, regression trees or neural networks. If we consider these methodology and take into account the temperature records of each day as independent from the previous one, then the considered methodologies are consistent with the problem. If we want to take into account the fact that the temperature is a time series (there exists correlation among days) we should resort to different techniques, e.g., time series prediction techniques like AutoRegressive (AR) models.

Answer of exercise 2.4

1. **Regression problem** One might consider as features the amount of rooms in a house, its distance to the city center, the presence of nearby facilities (primary school, church, underground), age of the building.

2. **Classification problem** In the specific, this problem can be categorized as an anomaly detection one. We can consider as interesting variables the amount of stock actions of a firm exchanged by traders, their affiliation and information about their money transaction.

3. **Feature extraction/selection** The interesting features in a image recognition problem are often application dependent. One of the most popular approach considered nowadays is the one offered by deep networks which automatically identifies the interesting features from an image for a given task.

4. **Classification problem** In this case, we would need a dataset composed by the couples audio recording/species from all the species we would like to discriminate.

5. **Reinforcement learning problem** One of the techniques we might consider is

q-learning and as features we might consider records from air hockey matches played by humans, for instance where you have information about the position of the hockey mallet and the forces used to kick the puck.

6. **Multiple interpretation** In this kind of problem one might consider the problem as a missing data reconstruction problem, a classification problem or a clustering one. In all the previous interpretation the features we would like to have are records (evaluations) coming from previously seen users.

7. **Classification problem** Here some of the solutions performing really well make use of deep neural networks, but also Support Vector Machine (SVM) might be an option. If we want to categorize a handwritten digit into the classes $0, \ldots, 9$ we need images of handwritten digits as features, coupled with the correct class. One might also consider it as a twofold problem of identifying the correct features in an image and then classification into 10 classes.

8. **Online learning problem** Here we might consider algorithm coming from the Multi-Armed Bandit (MAB) framework. The problem of pricing a good, in the case we do not have information about the user, is a problem of selecting the best action in the shortest time. Here, we simply need the outcome of newly seen buyers. In the case we have information about users we might resort to techniques coming from the recommendation field.

otherwise, regression

Answer of exercise 2.5

1. FALSE: It rarely happens that the problem we are modeling has linear characteristics.

2. TRUE: It is true that this model is easy to interpret and can be extended to also consider nonlinear relationships among variables, e.g., using basis functions.

3. TRUE: Since we are able only to minimize the discrepancy between the considered function and the real one and we can not reduce the variance introduced by noise, the use of linear model might be a better choice w.r.t. more complex ones since they usually are prone to overfitting, i.e., they try to model also the noise of the considered process.

4. TRUE: They are easy to interpret and might give suggestions on more sophisticated techniques which can be used to tackle specific problems.

5. TRUE/FALSE: For some loss functions we have a closed form solution for linear model (LS method), thus we can guarantee that we are able to find the parameters minimizing the loss function in an effective way. That is not always true and depends also on the loss function we want to minimize.

Answer of exercise 2.6

1. YES: since the most computationally complex part of the method implies the inversion of a design matrix, if we have only few parameter we only need invert a small matrix.

2. NO: it does not exists a closed form solution for the so called Laplace Loss function. Differently from the squared loss, this loss is not derivable in the origin, thus the definition of its derivative is not unique there.

3. YES/NO: in principle the inversion of the LS matrix is linear in the number of samples considered, therefore it does not depend on the number of samples. Nonetheless, to compute the design matrix $X^\top X$ it is required to perform a matrix multiplication, which is linear in the number of samples. Therefore for extremely large dataset this operations might be unfeasible.

4. NO: again the LS method only minimizes the RSS. The so called Huber Loss can be minimized by an iterative method. (FYI It is robust to noise, but differently from the squared one does not have a unique solution).

Answer of exercise 2.7

1. We are getting a badly conditioned design matrix.

2. The parameter we get have a high variance, since we have an infinite number of couples of parameters minimizing the loss of the samples in the considered problem. Indeed, if the parameters of the two inputs are w_1 and w_2 we would have that the true relationship would be:

   $$t = w_1 x + w_2 2x = (w_1 + 2w_2)x$$

   which can be satisfied by an infinite number of solutions.

3. In this case, the use of Ridge regression is able to partially cope with the influence of using highly linearly correlated features. Another viable option is to remove the variables which are linearly dependent, for instance by checking if they have correlation equal to 1 or -1

4. In this case we do not have a badly conditioned matrix since x and x^2 are not linearly dependent and the corresponding design matrix would not be ill-conditioned.

Answer of exercise 2.9

The update for a given parameter for the gradient descent for a single datum is:

$$w_j^{(1)} = w_j^{(0)} - \alpha(\mathbf{x}_1^T \mathbf{w}^{(0)} - t_1)x_{1j},$$

thus, we have:

$$w_0^{(1)} = 0 - 0.3(2-4)\cdot 1 = 0.6,$$
$$w_1^{(1)} = 0 - 0.3(2-4)\cdot 3 = 1.8,$$
$$w_2^{(1)} = 1 - 0.3(2-4)\cdot 2 = 2.2,$$

and the final coefficient is $\mathbf{w} = [0.6,\ 1.8,\ 2.2]$.

In the case we want to consider a set of $K = 10$ samples we could use the batch update formulation:

$$\mathbf{w}^{(1)} = \mathbf{w}^{(0)} - \frac{\alpha}{K}\sum_{n=1}^{K}(\mathbf{x}_n^T\mathbf{w}^{(0)} - t_n)\mathbf{x}_n$$

Answer of exercise 2.10

Since the matrix $\Phi^T\Phi + \lambda I$ is semi-definite positive and its eigenvalues should all be greater than $\lambda = 10^{-5}$, we have:

1. NOT PLAUSIBLE: one eigenvalue is smaller than 10^{-5};
2. NOT PLAUSIBLE: one eigenvalue is negative;
3. PLAUSIBLE: all positive and greater than 10^{-5};
4. PLAUSIBLE: all positive and greater than 10^{-5}.

Answer of exercise 2.11

We know that the variable $S = \frac{\hat{w}_k - w_k}{\hat{\sigma}\ v_k}$ is distributed as a t-student distribution with $N - M - 1$ degree of freedom. If we use the Gaussian approximation we might say that $S \sim \mathcal{N}(0,1)$. We are performing a two sided test, thus, we do not reject the null hypothesis if we have:

$$\left|\frac{\hat{w}_k - w_k}{\hat{\sigma} v_k}\right| \leq z_{1-\alpha/2}$$
$$-z_{1-\alpha/2} \leq \frac{\hat{w}_k - w_k}{\hat{\sigma}\ v_k} \leq z_{1-\alpha/2}$$
$$\hat{w}_k - z_{1-\alpha/2}\ \hat{\sigma}\ v_k \leq w_k \leq \hat{w}_k + z_{1-\alpha/2}\ \hat{\sigma}\ v_k,$$

where we are using the symmetry properties of the Gaussian distribution. The maximum value for which we would not reject the null hypothesis is:

$$w_k = \hat{w}_k + z_{1-\alpha/2}\ \hat{\sigma}\ v_k = 0.5 + 1.96\cdot 0.2 \approx 0.9\ \ (0.892).$$

Answer of exercise 2.12

1. FALSE: Adding extra predictors to the model can obfuscate the interpretation of w_1 in the model since there might be some interdependencies among features in the real process generating data.
2. FALSE: Even if the process considered is often non-linear, a linear model might give interesting insight on it.
3. FALSE: The significance of the F-statistic means that at least one of the considered variables are meaningful to model the considered relationship.
4. FALSE: Positive correlation and a positive coefficients are equivalent only in a univariate regression.

Answer of exercise 2.13

1. NOT CONSISTENT: Since the prior does not coincide with the MLE solution it is not possible that the solution does not change.
2. CONSISTENT: The effect of the prior moves the solution towards the areas in the parameter space where the prior has higher probability.
3. NOT CONSISTENT: The effect of the prior does not allow the solution to move further to the region of prior maximal probability, i.e., the point $[3,\ 2]$
4. NOT CONSISTENT: Since the prior does not coincide with the MLE solution it is not possible that the solution coincides with the prior.