FGV EMAp
ESCOLA DE MATEMÁTICA APLICADA

Técnicas e Algoritmos em Ciência de Dados
Assessed Coursework 1

This assignment must be submitted by April 16th, 2020 at 8:00 am.
Late submissions will NOT be accepted.

Learning outcomes assessed

This coursework will test some basic concepts of Python programming for data analysis (Pandas, Seaborn, Scikit-learn) including data loading and cleaning, building and evaluating simple classification models and basic plotting.

Instructions

Identifier

Please choose a random number of 6 digits and write it in the first cell of the notebook. Make sure that you keep a copy of that number as it will be used to provide the feedback (please avoid trivial numbers, such as 000000 or 123456 – thank you).

Submission

Submit your files through ECLASS. The files you submit cannot be overwritten by anyone else, and they cannot be read by any other student. You can, however, overwrite your submission as often as you like, by resubmitting, though only the last version submitted will be kept. Submission after the deadline will NOT be accepted.

If you have issues, at the very last minute, email your coursework as an attachment at alberto.paccanaro@fgv.br with the subject "URGENT – COURSEWORK 1 SUBMISSION". IN the body of the message, explain the reason for not submitting through ECLASS.

IMPORTANT

- Your submission will consist of a single Python notebook implementing your solutions.
- The name of the file will be the random number that identifies you (for example 568423.ipynb)
- This coursework consists of 3 parts. Please make sure that the 3 parts are clearly separated and identifiable in your Python Notebook.
- DO NOT SUBMIT ANY DATASET, only the code.
- Any utility function that you will use should be included in the notebook – do not submit utility scripts.

ADVICE ON BONUS EXERCISES

The value of each exercise in percentages is provided. However, note that the total sum of the marks for the exercises is 110/100. This is because 10 extra points are given for **BONUS exercises –** these are clearly marked in the text below**.** *Note that these exercises are difficult and time-consuming. <u>I advise you to have a final working solution for the entire coursework before attempting to answer these optional exercises</u>.*

All the work you submit should be solely your own work. Coursework submissions will be checked for this.

Marking Criteria

This coursework is assessed and mandatory and is worth 25% of your total final grade for this course. In order to obtain full marks for each question, you must answer it correctly but also completely. Marks will be given for writing well structure code.

COURSEWORK

Part 1 – Data Loading and pre-processing – value of this section: 25 %

Download the file “Part1.tsv” from the ECLASS platform.
Note that this file has a custom format: each datapoint has 5 features (the first 5 values) and an associated class label (the last value). In this part, you will load and visualize the dataset.

Here are the steps that you will need to implement:

a) Load the data into a pandas DataFrame
b) Clean the dataset by removing the datapoints with missing values
c) For each feature in the dataset, make a figure containing 2 subplots showing:
   - The histogram of the values of the feature (all classes combined).
   - The histogram of the values of the feature separately for each class. Use a different colour for each class.

   Note that all your histograms should have 50 bins and should be normalized (the area under the histogram should sum up to 1).
   For each feature, the figure should look something like Figure 1.

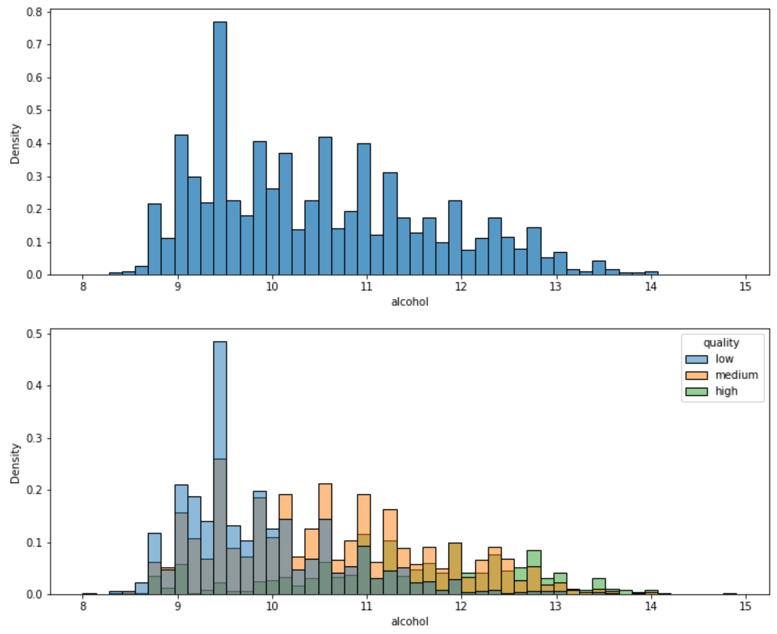

Figure1

d) Make a figure containing a matrix of subplots. These will present projections of the dataset onto each pair of features, histogram of their distributions and values of Pearson correlation coefficients.
   The matrix of subplots will be organized like figure 2.
   - The upper triangular section of the matrix will contain in the (i, j) entry the Pearson correlation between feature i and feature j.

- The main diagonal will contain in position (i, i) the histograms of the values of feature i. These histograms are the same as you already plotted in point c): you should use a different colour for each class and histograms should have 50 bins and should be normalized (the area under the histogram should sum up to 1).
- The lower triangular section of the matrix will contain in the (i, j) entry the projections of the points onto the i and j features. Use a different colour for each class and the colours should math those used in the histograms.

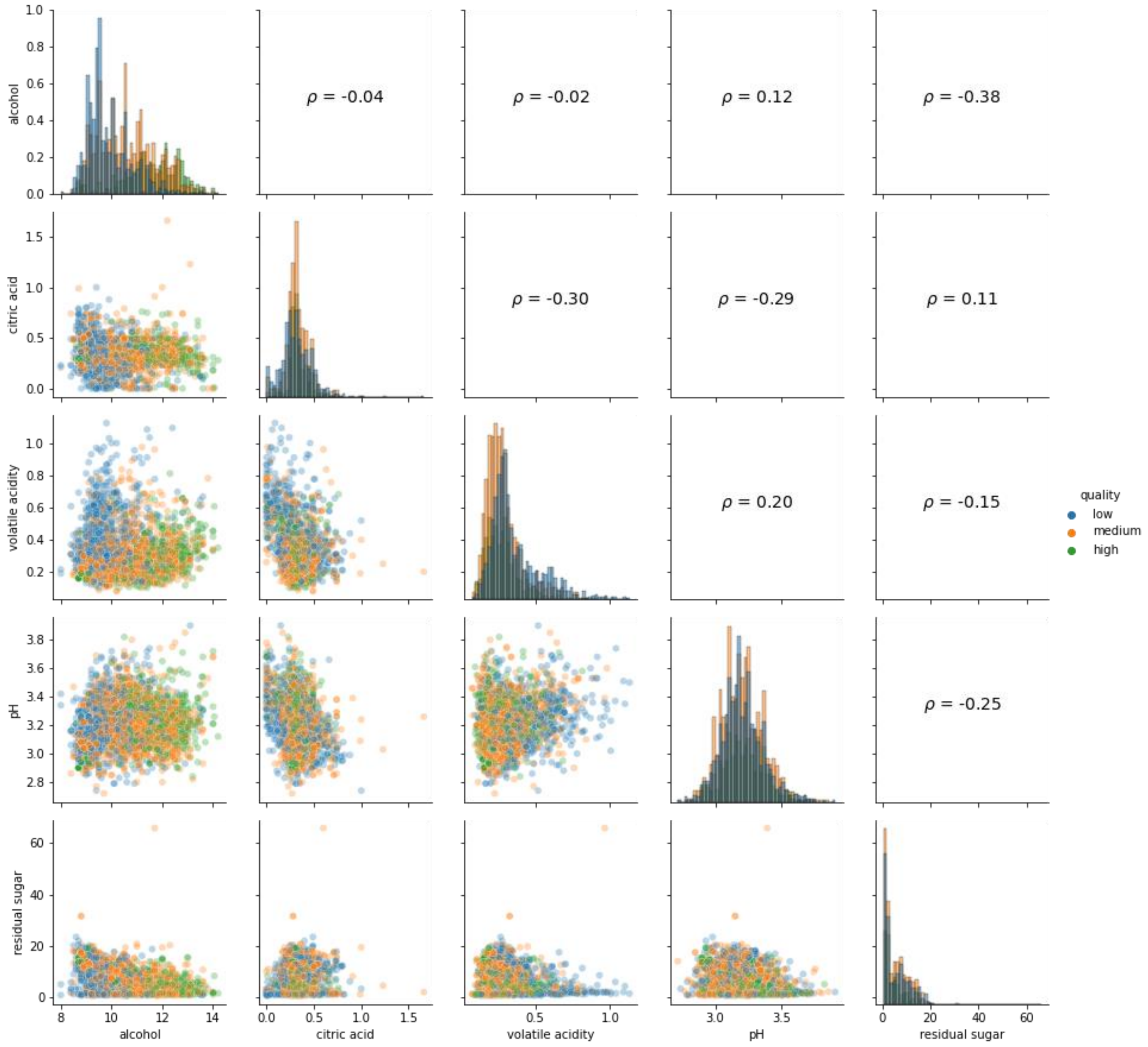

Figure 2

Part 2 – Mixture of Gaussians as a classifier – value of this section: 45 %

Download the file “Part2.tsv” from the ECLASS platform. In this file, each datapoint has 2 features (the first 2 values) and an associated class label (the last value). In this part, you will build a Bayesian classifier based on the Mixture of Gaussian model.

Here are the steps that you will need to implement:

a) Load the data into a pandas DataFrame
b) Make a scatterplot of the data, using different colours to denote points from different classes.
c) Use scikit-learn’s implementation of the Mixture of Gaussians to implement a Bayesian classifier for the data (use Bayes’ theorem).

Tip 1: be careful that scikit-learn’s implementation of the Mixture of Gaussians is oriented to clustering (which is unsupervised learning) not density estimation

Tip 2: be careful that scikit-learn’s implementation of the Mixture of Gaussians will provide with the log likelihood (not the likelihood!) of the data given the model. You will need to take this into account before you apply Bayes’ theorem.

BONUS (6% extra marks): Make your implementation compatible with the scikit-learn estimator API: https://scikit-learn.org/stable/developers/develop.html#rolling-your-own-estimator

Part 3 – Model Selection – value of this section: 30 %

This part uses the same data you used already in Part 2 (that is, file “Part2.tsv”). In this part, you will compare the generalization performance of different classifiers with respect to several performance measures. The generalization performances will be measured on a test set constituted by 25% of the data.

The classifiers are:

- scikit-learn’s implementation of Naive Bayes
- scikit-learn’s implementation of K-nearest-neighbour.
- a very simple model that classifies datapoints using only the class priors *(Tip: this is similar to what you did in the lab of week 5)*
- a Bayesian classifier that fits a single gaussian for each class. *(Tip: this is similar to what you did in the lab of week 6)*
- the Bayesian classifier based on the Mixture of Gaussians model that you implemented in Part 2 *(Note: if you have not managed to implement the Mixture of Gaussians model in part 2, then simply omit this model from your analysis).*

The performance metrics are (all implemented in scikit-learn):

- Accuracy
- Precision
- Recall
- AUC-ROC
- AUC-PR

Your code will:

a) Print a table with values of the performance metric (columns) for each model (rows) on the test set.
b) Make a single figure containing 2 subplots. In the first subplot, you will plot the ROC curves for each model. In the second subplot, you will plot the PR curves for each model.
c) For each model, make a figure containing the decision boundaries of the model.

BONUS (4% extra marks): Turn your implementations of the prior, and multivariate gaussian models into scikit-learn estimators.