Imperial College London

Department of Mathematics
MATH96007 - MATH97019 - MATH97097
Methods for Data Science
Years 3/4/5

Coursework 1 – Supervised learning

Submission deadline: Wednesday, 24 February 2021, 5 pm

General instructions

The goal of this coursework is to analyse two datasets using several of the tools and algorithms introduced in the lectures, which you have also studied in detail through the computational tasks in the course.

You will solve the tasks in this coursework using Python. You are allowed to use Python code that you will have developed in your coding tasks. You are also allowed to use any other basic mathematical functions contained in numpy. However, importantly, you are **not** allowed to use any model-level Python packages like sklearn, statsmodels, or similar, for your solutions unless we explicitly state it.

Submission

Your coursework will be presented as a **Jupyter notebook (file format: ipynb)** with all your tasks clearly labelled. The notebook should contain the cells with your code and their output, and some brief text explaining your calculations, choices, mathematical reasoning, and explanation of results. The notebook must be run and outputs of cells printed. You may produce your notebook with Google Colab, but we recommend to develop your local Jupyter notebook through the Anaconda environment (or any local Python environment) installed on your computer.

Once you have executed all cells in your notebook and their outputs are printed, also save the notebook as an **html file**, which you will also submit.

Submission instructions

The submission will be done **online via Turnitin on Blackboard**.

The deadline is **Wednesday, 24 February 2021 at 5 pm**.

*You will upload **two documents** to Blackboard, wrapped into a **single zip file**:*

1) *Your Jupyter notebook as an **ipynb file**.*
2) *Your notebook exported as an **html file**.*

You are also required to comply with these specific requirements:

- *Name your **zip file** as 'SurnameCID.zip', e.g. Smith1234567.zip. Do not submit multiple files.*
- *Your ipynb file must produce all plots that appear in your html file, i.e., make sure you have run all cells in the notebook before exporting the html.*
- *Use clear headings in your notebook to indicate the answers to each question, e.g. 'Task 1.1'.*

***Note about online submissions:** There are known issues with particular browsers (or settings with cookies or popup blockers) when submitting to Turnitin. If the submission 'hangs', please try another browser.*

You should also check that your files are not empty or corrupted after submission.

To avoid last minute problems with your online submission, we recommend that you upload versions of your coursework early, before the deadline. You will be able to update your coursework until the deadline, but having these early versions provides you with some safety back up.

***Needless to say, projects must be your own work:** You may discuss the analysis with your colleagues but the code, writing, figures and analysis must be your own. The Department may use code profiling and tools such as Turnitin to check for plagiarism, as plagiarism cannot be tolerated.*

Marks

The coursework is worth **40% of your total mark for the course.**
This coursework contains a ***mastery component*** for MSc and 4th year MSci students.

Guidance about solutions and marking scheme:

Coursework tasks are different from exams: sometimes they can be more open-ended and may require going beyond what we have covered explicitly in lectures. In some parts of the tasks, initiative and creativity will be important, as is the ability to pull together the mathematical content of the course, drawing links between subjects and methods, and backing up your analysis with relevant computations that you will need to justify.

To gain the marks for each of the Tasks you are required to:

(1) complete the task as described;
(2) comment any code so that we can understand each step;
(3) provide a brief written introduction to the task explaining what you did and why you did it;
(4) provide appropriate, relevant, clearly labelled figures documenting and summarising your findings;
(5) provide an explanation of your findings in mathematical terms based on your own computations and analysis and linking the outcomes to concepts presented in class or in the literature;
(6) consider summarising your results of different methods and options with a judicious use of summary tables.

The quality of presentation and communication is very important, so use good combinations of tables and figures to present your results, as needed.

Explanation and understanding of the mathematical concepts are crucial.

Competent Python code is expected. As stated above, you are allowed to use your own code and the code developed in the coding tasks in the course, but you are not allowed to use Python packages like sklearn, statsmodels, etc unless explicitly stated.

Hence marks will be reserved and allocated for: presentation; quality of code; clarity of arguments; explanation of choices made and alternatives considered; mathematical interpretation of the results obtained; as well as additional relevant work that shows initiative and understanding beyond the task stated in the coursework.

Note that the mere addition of extra calculations (or ready-made 'pipelines') that are unrelated to the task without a clear explanation and justification of your rationale will not be beneficial in itself and, in fact, can also be detrimental if it reveals lack of understanding.

Overview of the coursework

In this first coursework, you will work with two different datasets of high-dimensional samples:

- **a housing market dataset**
- **a collection of credit applications**

You will perform a **regression task** with the former, and a **binary classification task** with the latter.

Task 1: Regression (50 marks)

Dataset: Your first task deals with a *modified* dataset that we have prepared based on a collection of household information over various locations across Boston in the US. Each sample in the dataset corresponds to a household characterised by 18 features, from per capita crime rate to proportion of non-retail business acres in the town of the household. We will take one of these features (namely, *the median value of owner-occupied homes in US$ 1000s*, found in the **last column** of the `csv` files) as the target variable, and we use the other 17 features as predictors.

This modified Boston housing dataset has been split into a *training set* and a *test set*, and is made available to you on Blackboard as `regression_train.csv` and `regression_test.csv`. The *test set* should **not** be used in any learning or tuning of the models. It should only be used *a posteriori* to support your conclusions and to evaluate the out-of-sample performance of your models.

Questions:

1.1 Linear regression (10 marks)

1.1.1 - For the modified Boston housing data set, obtain a linear regression model to predict the *median value of owner-occupied homes in USD 1000's* as your target variable using all the other features as predictors. Report the parameters of the model and the in-sample mean squared error (MSE) from the training set.

1.1.2 - Use the model on the test data to predict the target variable, and compute the out-of-sample MSE on the test set. Compare the out-of-sample and the in-sample MSE, and explain your answer.

1.2 Ridge regression (20 marks)

1.2.1 - Repeat task 1.1.1 employing ridge regression using a 5-fold cross validation algorithm to tune the ridge model on the set `regression_train.csv`. Use one of the folds to demonstrate with plots how you scan the penalty parameter of the ridge model to find the optimal value of the penalty by examining the MSE on the corresponding validation subset. Report the values of the penalty parameter obtained for the five folds.

1.2.2 - Obtain the in-sample MSE by retraining the model using the optimal penalty parameter on the entire training set, and compute its out-of-sample MSE (on the test set `regression_test.csv`). Compare these values to the results found in 1.1.2 using linear regression. Explain the differences observed between the two methods justifying your answer in terms of particular predictors of interest.

1.3 Regression with *k* nearest neighbours (*kNN*) (20 marks)

1.3.1 - Repeat task 1.2.1 employing the *kNN* algorithm as a regression model. Tune your *kNN* model using a 5-fold cross validation strategy on the same splits as in 1.2.1. Using one of the folds: (i) demonstrate the process by which you scan over a range of *k* to find your optimal value within this range; (ii) examine both the MSE and the distribution of the errors obtained on the corresponding validation subset. Explain your results.

1.3.2 - Obtain the in-sample MSE by retraining the model using the optimal penalty parameter on the entire training set, and compute its out-of-sample MSE (on the test set `regression_test.csv`). Compare these values to the results obtained in 1.1 and 1.2. Compare the observed performance of the

kNN algorithm to the models in 1.1 and 1.2, considering the homogeneity of the data and the possible nonlinearity of the descriptors.

Task 2: Classification (50 marks)

Dataset: Your second dataset is a modified dataset based on a collection of credit applications to an unknown bank. You have information on 11 different descriptors (or features) of each applicant and the decision of the banker whether the applicant is granted a credit or not (found in the **last column** of the `csv` files). We will take the 11 features as numerical predictors in our classification tasks.

The data set is accessible on Blackboard and has been split into a *training set* and a *test set:* `classification_train.csv` and `classification_test.csv`. Again, the *test set* should only be used *a posteriori* to support your conclusions and to evaluate the out-of-sample performance of your models.

Questions:

2.1 Logistic regression (10 marks)

2.1.1 - Train a logistic regression classifier on your training data with gradient descent for 5000 iterations. Demonstrate that you have used a grid-search with 5-fold cross validation to find your optimal set of hyperparameters (learning rate, decision threshold).

2.1.2 - Compare the performance of your optimal model on the training data and on the test data by their *mean accuracies*.

2.2 Random forest (20 marks)

2.2.1 - Train a random forest classifier on the training data. You should use the same 5-fold cross validation subsets to explore and optimise over suitable ranges of the following hyperparameters: (i) number of decision trees; (ii) depth of trees, (iii) maximum number of descriptors (features) randomly chosen at each split. Use *cross-entropy* as your information criterion for the splits.

2.2.2 - Compare the performance of your optimal model on the training data and on the tes data using different measures computed from the *confusion matrix*.

2.3 Support vector machines (SVMs) (20 marks)

2.3.1 - This task will deal with two *hard margin SVM classifiers*: (i) Implement the standard linear SVM with hard margin on the training data; (ii) implement a hard margin kernel SVM with radial basis function (RBF) kernel, and demonstrate that you have used a grid-search with 5-fold cross validation (same folds as above) to find the RBF kernel with the optimal hyperparameter with respect to the F1-score. Compare the results of the linear SVM and the optimal kernel SVM.

2.3.2 - Evaluate the performance of the RBF SVM on the *test data* over the range of the hyperparameter of the kernel and represent the results using a *receiver operating characteristic (ROC) curve*. Use this ROC curve to evaluate the quality of the optimal kernel SVM obtained in 2.3.1 through cross-validation on the training set.

Task 3 *(mastery component)*: Sigmoid Kernelised SVM: hard versus soft margin (25 marks)

3.1 Hard margin sigmoid kernel (10 marks)

Train a hard margin kernelised SVM classifier with a sigmoid kernel on the training data. Demonstrate that you have used a grid-search with 5-fold cross validation to find the optimal hyperparameters for this kernel with respect to the F1-score.

3.2 Soft margin sigmoid kernel (15 marks)

Repeat task 3.1 but now using a soft margin objective (instead of hard margin) on the training data. *You will need to implement this soft margin SVM yourself*. Demonstrate that you have used a grid-search with 5-fold cross validation to find the optimal hyperparameters for this kernel with respect to the F1-score.

Compare the performance of the hard margin vs soft margin versions on the test data, and explain the differences considering your kernel and the data distribution.