IAML – INFR11182 (LEVEL 11): Assignment #1

Due on Tues, October 20, 2020 @ 16:00

NO LATE SUBMISSIONS

IMPORTANT INFORMATION

N.B. *This document is best viewed on a screen as it contains a number of (highlighted) clickable hyperlinks.*
It is very important that you read and follow the instructions below to the letter. You will be deducted marks for not adhering to the advice below.

Good Scholarly Practice: Please remember the University requirement regarding all assessed work for credit. Details about this can be found at:

http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct

Specifically, this assignment should be your own individual work. Moreover, please note that Piazza is **NOT** a forum for discussing the solutions of the assignment. You may, in exceptional circumstances, ask **private** questions to the instructors if you deem that something may be incorrect, and if we feel that the issue is justified, we will send out an announcement.

General Instructions

- There are two versions of this assignment. One for INFR10069 (level 10) and the other for INFR11182 (level 11). The level 11 version has some additional parts. **MAKE SURE you are doing the assignment that corresponds to the course you are registered on; you can check this on EUCLID.**

- You should use Python for implementing your solutions as this will standardise the output and also provide a consistent experience with the labs. Set up your environment as specified in the Labs. **It is VERY IMPORTANT that you use**

the exact same package versions as those specified in the requirements file from the labs! Using the correct environment (i.e. `py3iaml`) is necessary to ensure that your outputs are consistent with the expected solutions. The correct package versions are specified here.

- If running `import sklearn; print(sklearn.__version__)` in your Jupyter Notebook does not print the package version `0.19.1`, then you are *not* using the correct environment.
- This assignment consists of multiple questions. **MAKE SURE to use the correct dataset for each question.**
- This assignment accounts for 20% of your final grade for this course and is graded based on a written report (compiled from a latex template which we provide).
- The criteria on which you will be judged include the quality of the textual answers and/or any plots asked for. While code will be needed to generate the results, this is not a programming assignment, and you are not expected to provide code unless explicitly requested.
- Read the instructions carefully, answering what is required and only that. Keep your answers brief and concise. Specifically, **for textual answers**, the size of the text-box in the latex template will give you an idea of the **maximum** length of your answer. You do **not need** to fill in the whole text-box but you will be **penalised** if you go over. This does not apply to figure-based answers.
- For answers involving figures, make sure to clearly label your plots and provide legends where necessary. You will be penalised if the visualisations are not clear.
- For answers involving numerical values, use correct units where appropriate and format floating point values to an appropriate number of decimal places.

Submission Mechanics

Important: ***You must submit this assignment by Tuesday 20/10/2020 at 16:00. We do not accept Late Submissions for this coursework, except in the case of mitigating circumstances. Please refer to the ITO Website for further details.***

- We will use the Gradescope submission system for uploading PDF assignments. Information describing how to upload your completed assignment will be made available on the IAML Learn page.
- You should clone or download the Assignment Repository from https://github.com/uoe-iaml/INFR11182-2020-CW1.

  This contains:

  1. The data you will need for the assignment under the `data` directory.

2. Two `tex` files, `Assignment_1.tex` and `style.tex`. These provide the template for you to fill out the assignment questions. In particular, the template forces your answers to appear on separate pages and also controls the length of textual answers.

- You should **only** modify the `Assignment_1.tex` template by:

1. Uncommenting and specifying your student number at the top of the document (compilation will automatically fail if you forget to do this). Remove the '%' and enter your student number e.g.

   ```
   \newcommand{\assignmentAuthorName}{s1234567}
   ```

2. Filling in the answers in the provided `answerbox` environment.

DO NOT modify anything else in the template and certainly **DO NOT** edit the style file. **We reserve the right to not mark assignments which do not adhere to the template**.

Latex Tips

- To fill in text answers, you can modify the corresponding `answerbox`:

```
\begin{answerbox}{5em}
Your answer here
\end{answerbox}
```

with your answer (replacing 'Your answer here'):

```
\begin{answerbox}{5em}
Steam locomotives were first developed in the United Kingdom during the early 19th century and used for railway transport until the middle of the 20th century.
\end{answerbox}
```

which, when compiled gives:

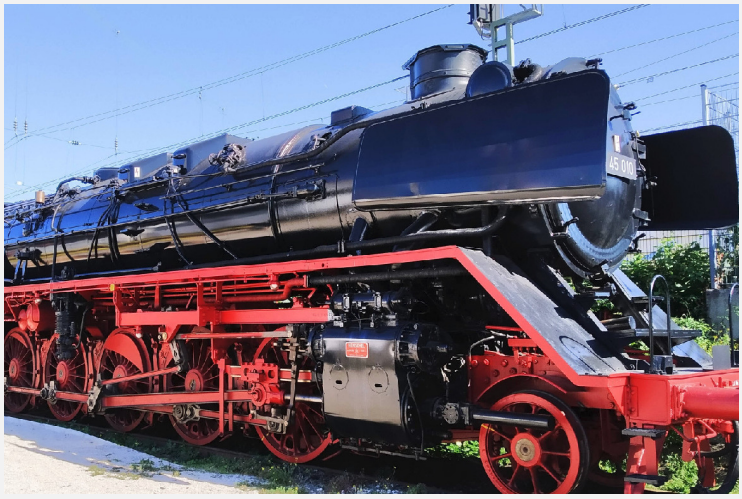
> Steam locomotives were first developed in the United Kingdom during the early 19th century and used for railway transport until the middle of the 20th century.

- To add an image, you can use:

```
\begin{answerbox}{18em}
This image shows a train.
\begin{center}
\includegraphics[width=0.7\textwidth]{stock_image.jpg}
\end{center}
\end{answerbox}
```

which will be compiled to:

This image shows a train.



Make sure that you specify the correct path to your image. For example. if your image was stored in a directory called `results`, you would change the relevant line to read:

```
\includegraphics[width=0.7\textwidth]{results/stock_image.jpg}
```

You can find more information about inserting images into latex documents here.

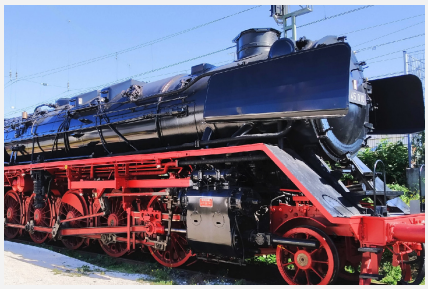
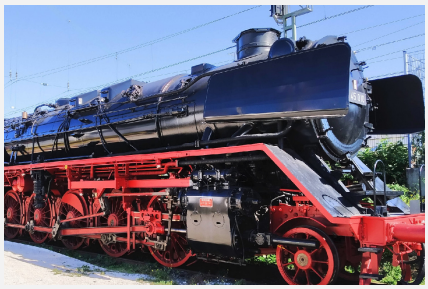
- You can also add two images side-by-side:

```
\begin{answerbox}{18em}
Below we see two trains.

\begin{center}
\begin{tabular}{ll}
\includegraphics[width=0.4\textwidth]{stock_image.jpg}
&
\includegraphics[width=0.4\textwidth]{stock_image.jpg}
\end{tabular}
\end{center}
\end{answerbox}
```

which will be compiled to:

Below we see two trains.

- To add an inline equation, you can use the '$' symbol to write:

```
\begin{answerbox}{3em}
I am using the following model, $y = \mathbf{x}^T\mathbf{w}$.
\end{answerbox}
```

which compiles to:

I am using the following model, $y = \mathbf{x}^T\mathbf{w}$.

- To add a table for numerical results you can use:

```
\begin{answerbox}{7em}
Results are presented in the table below.

\begin{center}
\begin{tabular}{ |c|c|c| }
 \hline
 Parameter Value & Train Accuracy & Test Accuracy \\ \hline
 1 & 10.1\% & 9.1\% \\
 2 & 12.5\% & 10.1\% \\
 \hline
\end{tabular}
\end{center}
\end{answerbox}
```

which compiles to:

Results are presented in the table below.

Parameter Value	Train Accuracy	Test Accuracy
1	10.1%	9.1%
2	12.5%	10.1%

You can find more information about tables in latex here.

- For a small number of questions we may ask you to report your code. You can include code as an image, but if you prefer you can use the following command:

```
\begin{answerbox}{5em}
\begin{verbatim}
import numpy as np
mean_time = 10.0
print('mean time', mean_time)
\end{verbatim}
\end{answerbox}
```

  which, when compiled gives:

```
import numpy as np
mean_time = 10.0
print('mean time', mean_time)
```

- Once you have filled in all the answers, compile the latex document to generate the PDF that you will submit. You can use Overleaf, your favourite latex editor, or just run `pdflatex Assignment_1.tex` twice on a DICE machine to compile the PDF.

Question 1 : (22 total points) Linear Regression

In this question we will fit linear regression models to data.

Here we will investigate the relationship between the amount of time (in hours) each student in a class studied for an exam and their end of semester exam performance. We will model this relationship for each student using $y_i = \boldsymbol{\phi}(x_i)\mathbf{w}$, where $\boldsymbol{\phi}(x_i) = [1, x_i]$ is a row vector and $\mathbf{w}$ are the model parameters we will learn. Here, y_i is the exam score for student i and x_i is the amount of time they spent revising.
The dataset is contained in `regression_part1.csv`. You should load it into a Pandas DataFrame using `pandas.read_csv()`.

(a) (3 points) Describe the main properties of the data, focusing on the size, data ranges, and data types.

(b) (3 points) Fit a linear model to the data so that we can predict `exam_score` from `revision_time`. Report the estimated model parameters $\mathbf{w}$. Describe what the parameters represent for this 1D data. For this part, you should use the sklearn implementation of Linear Regression.
Hint: By default in sklearn `fit_intercept = True`. *Instead, set* `fit_intercept = False` *and pre-pend* 1 *to each value of* x_i *yourself to create* $\boldsymbol{\phi}(x_i) = [1, x_i]$.

(c) (3 points) Display the fitted linear model and the input data on the same plot.

(d) (3 points) Instead of using sklearn, implement the closed-form solution for fitting a linear regression model yourself using numpy array operations. Report your code in the answer box. It should only take a few lines (i.e. <5).
Hint: Only report the relevant lines for estimating $\mathbf{w}$ *e.g. we do not need to see the data loading code. You can write the code in the answer box directly or paste in an image of it.*

(e) (3 points) Mean Squared Error (MSE) is a common metric used for evaluating the performance of regression models. Write out the expression for MSE and list one of its limitations.
Hint: For notation, you can use y *for the ground truth quantity and* $\hat{y}$ (`$\hat{y}$` *in latex) in place of the model prediction.*

(f) (3 points) Our next step will be to evaluate the performance of the fitted models using Mean Squared Error (MSE). Report the MSE of the data in `regression_part1.csv` for your prediction of `exam_score`. You should report the MSE for the linear model fitted using sklearn and the model resulting from your closed-form solution. Comment on any differences in their performance.

Question 1 continued on next page. . .

(g) (4 points) Assume that the optimal value of w_0 is 20, it is not but let's assume so for now. Create a plot where you vary w_1 from -2 to $+2$ on the horizontal axis, and report the Mean Squared Error on the vertical axis for each setting of $\mathbf{w} = [w_0, w_1]$ across the dataset. Describe the resulting plot. Where is its minimum? Is this value to be expected? *Hint: You can try 100 values of* w_1 *i.e.* `w1 = np.linspace(-2,2, 100)`.

Question 2 : (18 total points) Nonlinear Regression

In this question we will tackle regression using basis functions.

Here we will look at a regression dataset where the attribute we want to predict (`output`) does *not* have a linear relationship with the input attribute (`input`) we can measure. To overcome this problem we will first use polynomial basis functions, where again $y_i = \boldsymbol{\phi}(x_i)\mathbf{w}$. However, now the row vector $\boldsymbol{\phi}(x_i) = [1, x_i, x_i^2, ..., x_i^M]$, where M is an integer. The dataset is contained in `regression_part2.csv`. You should load it into a Pandas DataFrame using `pandas.read_csv()`.

(a) (5 points) Fit four different polynomial regression models to the data by varying the degree of polynomial features used i.e. $M = 1$ to 4. For example, $M = 3$ means that $\boldsymbol{\phi}(x_i) = [1, x_i, x_i^2, x_i^3]$. Plot the resulting models on the same plot and also include the input data.
Hint: You can again use the sklearn implementation of Linear Regression and you can also use PolynomialFeatures to generate the polynomial features. Again, set `fit_intercept = False`.

(b) (3 points) Create a bar plot where you display the Mean Squared Error of each of the four different polynomial regression models from the previous question.

(c) (4 points) Comment on the fit and Mean Squared Error values of the $M = 3$ and $M = 4$ polynomial regression models. Do they result in the same or different performance? Based on these results, which model would you choose?

(d) (6 points) Instead of using polynomial basis functions, in this final part we will use another type of basis function - radial basis functions (RBF). Specifically, we will define $\boldsymbol{\phi}(x_i) = [1, rbf(x_i; c_1, \alpha), rbf(x_i; c_2, \alpha), rbf(x_i; c_3, \alpha), rbf(x_i; c_4, \alpha)]$, where $rbf(x; c, \alpha) = \exp(-0.5(x - c)^2/\alpha^2)$ is an RBF kernel with center c and width α. Note that in this example, we are using the same width α for each RBF, but different centers for each.
Let $c_1 = -4.0$, $c_2 = -2.0$, $c_3 = 2.0$, and $c_4 = 4.0$ and plot the resulting nonlinear predictions using the `regression_part2.csv` dataset for $\alpha \in \{0.2, 100, 1000\}$. You can plot all three results on the same figure. Comment on the impact of larger or smaller values of α.

Question 3 : (26 total points) Decision Trees

In this question we will train a classifier to predict if a person is smiling or not.

Instead of images of faces, our dataset consists of a set of 2D coordinates that encode the location of points on the faces (e.g. the corners of the eyes, the nose, the chin, etc) of a set of different people. Each row in the dataset is a different person and the columns (i.e. the input attributes) are pairs of 2D coordinates e.g. the first coordinate is (x_0, y_0), the second is (x_1, y_1), etc. Note, in the notation used in the lectures we often use y_i to specify an output attribute, but here each (x_i, y_i) represents a 2D coordinate on the face and we will concatenate all D of them to form our attribute vector $[x_0, y_0, x_1, y_1, ..., x_D, y_D]$.
We assume that the location of these 2D points for a person that is smiling will be different than those for a person who is not smiling. Included with the 2D coordinates is an attribute called `smiling`, which is the binary class label that we want to predict. For a given row in the data, `smiling = 1` indicates that that person is smiling.
The training data is contained in `faces_train.csv`, and the test data can be found in `faces_test.csv`. You should load the data into two different Pandas DataFrames using `pandas.read_csv()`.

(a) (4 points) Load the data, taking care to separate the target binary class label we want to predict, `smiling`, from the input attributes. Summarise the main properties of both the training and test splits.

(b) (4 points) Even though the input attributes are high dimensional, they actually consist of a set of 2D coordinates representing points on the faces of each person in the dataset. Create a scatter plot of the average location for each 2D coordinate. One for (i) smiling and (ii) one not smiling faces. For instance, in the case of smiling faces, you would average each of the rows where `smiling = 1`. You can plot both on the same figure, but use different colors for each of the two cases. Comment on any difference you notice between the two sets of points.
Hint: Your plot should contain two faces.

(c) (2 points) There are different measures that can be used in decision trees when evaluating the quality of a split. What measure of purity at a node does the DecisionTreeClassifier in sklearn use for classification by default? What is the advantage, if any, of using this measure compared to entropy?

(d) (3 points) One of the hyper-parameters of a decision tree classifier is the maximum depth of the tree. What impact does smaller or larger values of this parameter have? Give one potential problem for small values and two for large values.

Question 3 continued on next page...

(e) (6 points) Train three different decision tree classifiers with a maximum depth of 2, 8, and 20 respectively. Report the maximum depth, the training accuracy (in %), and the test accuracy (in %) for each of the three trees. Comment on which model is best and why it is best.
Hint: Set `random_state = 2001` *and use the* `predict()` *method of the DecisionTreeClassifier so that you do not need to set a threshold on the output predictions. You can set the maximum depth of the decision tree using the* `max_depth` *hyper-parameter.*

(f) (5 points) Report the names of the top three most important attributes, in order of importance, according to the Gini importance from DecisionTreeClassifier. Does the one with the highest importance make sense in the context of this classification task?
Hint: Use the trained model with `max_depth = 8` *and again set* `random_state = 2001`.

(g) (2 points) Are there any limitations of the current choice of input attributes used i.e. 2D point locations? If so, name one.

Question 4 : (14 total points) Evaluating Binary Classifiers

In this question we will perform performance evaluation of binary classifiers.

You have been tasked with evaluating the performance of four different binary classification algorithms, `alg_1`, `alg_2`, `alg_3`, and `alg_4`. Unfortunately, you do not have access to the models themselves, only their predictions on a held-out test set. Your goal is to evaluate how well the different models perform at predicting the ground truth class labels `gt` for this test set.
The dataset is contained in `classification_eval_1.csv`. You should load it into a Pandas DataFrame using `pandas.read_csv()`.

(a) (4 points) Report the classification accuracy (in %) for each of the four different models using the `gt` attribute as the ground truth class labels. Use a threshold of $>= 0.5$ to convert the continuous classifier outputs into binary predictions. Which model is the best according to this metric? What, if any, are the limitations of the above method for computing accuracy and how would you improve it without changing the metric used?

(b) (4 points) Instead of using classification accuracy, report the Area Under the ROC Curve (AUC) for each model. Does the model with the best AUC also have the best accuracy? If not, why not?
Hint: You can use the roc_auc_score function from sklearn.

(c) (6 points) Plot ROC curves for each of the four models on the same plot. Comment on the ROC curve for `alg_3`? Is there anything that can be done to improve the performance of `alg_3` without having to retrain the model?
Hint: You can use the roc_curve function from sklearn.

Question 5 : (15 total points) Decision Tree Node Splitting

In this question we will explore how node splitting is performed in decision trees.

Decision tree classifiers operate by splitting the input attribute space into multiple subregions, allowing them to represent non-linear decision boundaries. Here we will explore how potential splits are evaluated.
For this question you will use the dataset contained in `faces_train.csv`. As a reminder, the binary class labels of interest are represented by the `smiling` attribute.

(a) (5 points) Compute the entropy of the training binary class labels in bits. Is this value to be expected? You should implement your own entropy function and report your code along with your answer.
Hint: Be sure that this works even when all the training labels belong to the same class. You do not need to include all your code (e.g. we do not need to see the data loading code), just the entropy computation function.

(b) (5 points) Plot a 1D histogram of the attribute labelled 'x48' from the training set for the instances where `smiling = 0`. Create a similar plot for the instances where `smiling = 1`. Comment on any differences or similarities between the two histograms for the attribute.
Hint: Set the number of bins in the histogram to 100.

(c) (5 points) Compute the information gain resulting from splitting the attribute labelled 'x48' from the training data using splitting thresholds of -0.9 and -0.7. Here, all the entries that are greater to a threshold t (i.e. $> t$) are sent to the right child node and all the entries that are less than or equal to t (i.e. $<= t$) are sent to the left child node.
Comment on which threshold results in a better split, and why it is better. You can make reference to your histograms from the previous answer.
Hint: You need to implement your own information gain function. You should report two separate numbers, one for each threshold. You do not need to include your code.