IAML – INFR10069 (LEVEL 10): Assignment #1

s1869672

Question 1 : (22 total points) Linear Regression

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

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

The data has 50 rows and 2 columns. The attribute `revision_time` ranges from 2.723 to 48.011 hours and the target `exam_score` ranges from 14.731 to 94.945. Both columns are of data type float64.

Question 1 continued on next page. . .

(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* $\boldsymbol{\phi}(x_i) = [1, x_i]$.

The estimated model parameters $\mathbf{w}$ is:

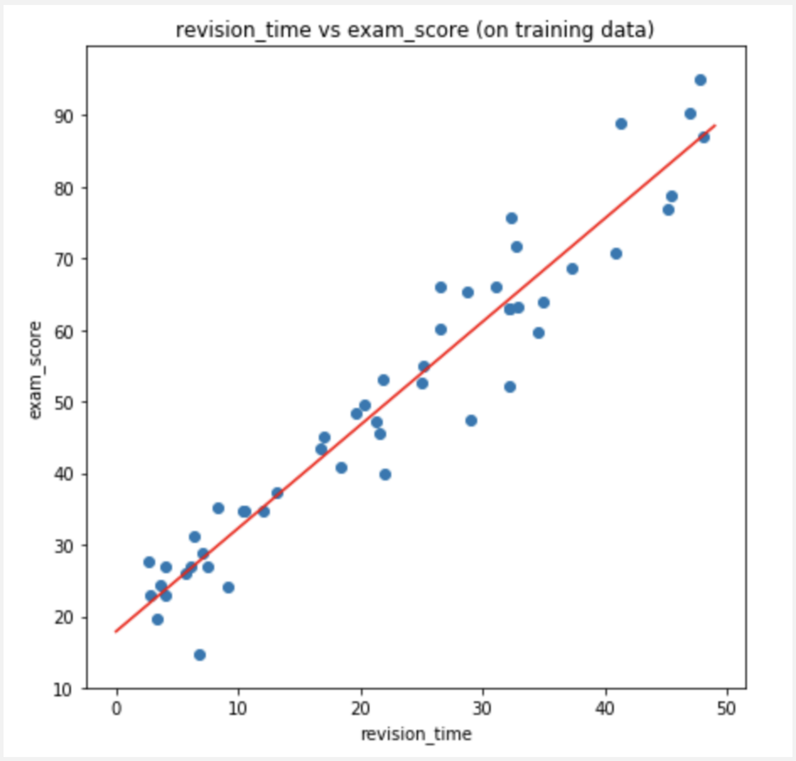
$$w = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} 17.9 \\ 2.44 \end{bmatrix}$$

The parameter w_0 represents the intercept.
The parameter w_1 represents the slope of the regression line.

Question 1 continued on next page. . .

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

The graph below shows the fitted linear model and the input data.



(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 **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.*

The following code is the closed-form solution for fitting a linear regression model using the pseudoinverse solution, where $X = (1, x_1, ..., x_{50})$ and $y = (y_1, y_2, ..., y_{50})$:

```
pseudoInverse = np.linalg.inv((X.T).dot(X)).dot(X.T)
w = pseudoInverse.dot(y)
y_preds_cf = X.dot(w)
```

Question 1 continued on next page. . .

(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.

The expression for MSE is given by:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

where n is the number of data points.
One limitation is that it is very sensitive to outliers.

Question 1 continued on next page…

(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.

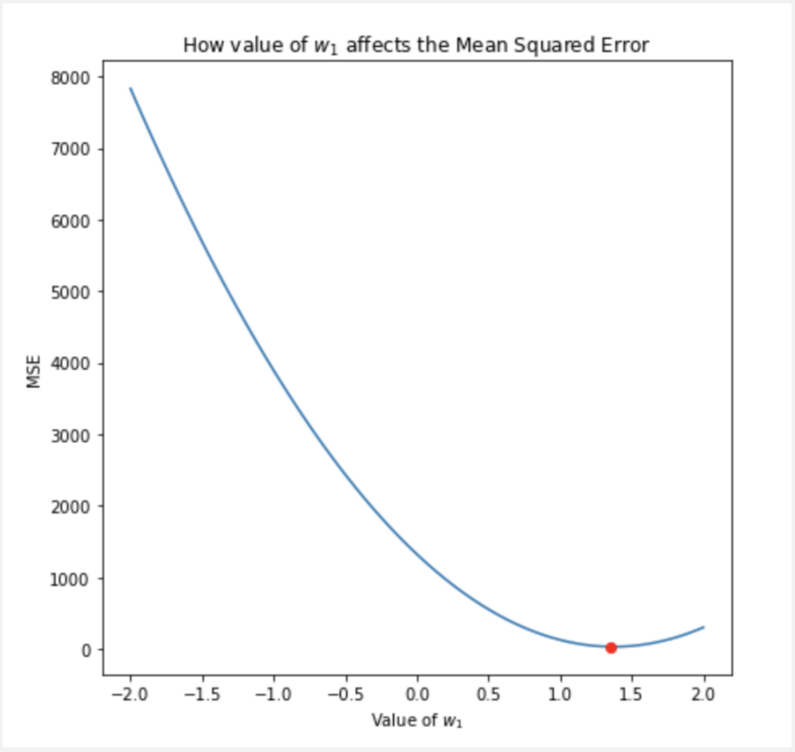
Results are presented in the table below.

	MSE
sklearn	30.985472614541294
closed-form solution	30.985472614541290

The difference is very small and probably comes from the calculations. Therefore, both methods practically give the same 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)`.

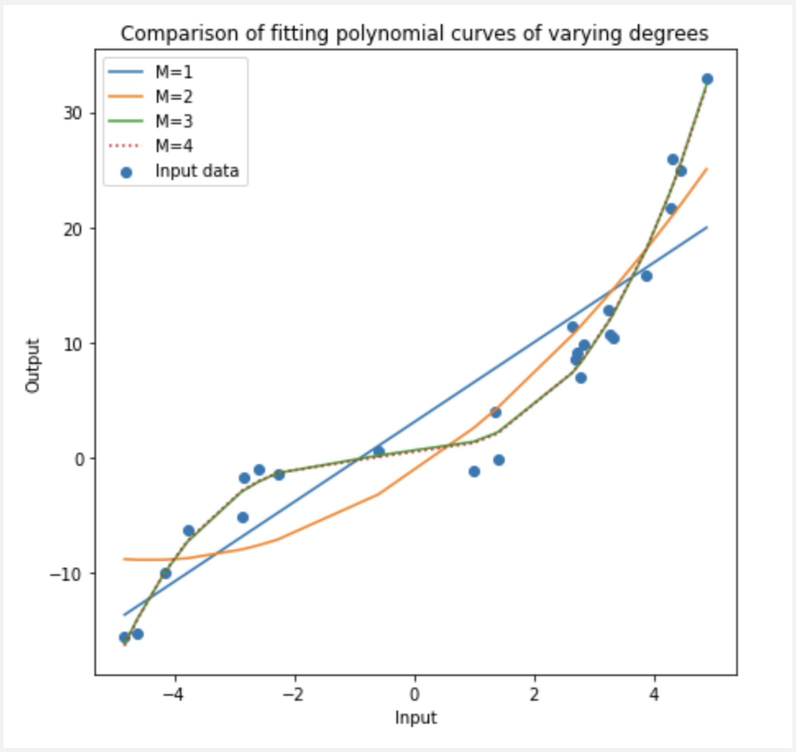


The MSE is minimum when the value of w_1 is equal to 1.35 (represented by the red dot on the graph above). There is a positive correlation between *revision_time* and *exam_score*, so we expect the value of w_1 to be positive. It is also relatively close to the value of w_1 that we computed in part (b). Therefore, this value is a reasonable minimum value.

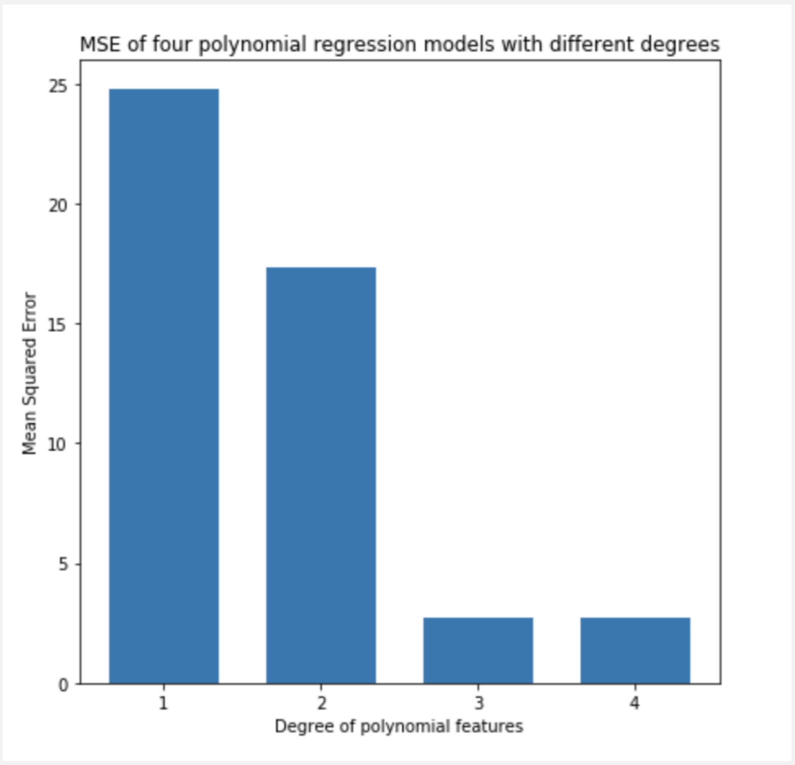
Question 2 : (18 total points) Nonlinear Regression

In this question we will tackle regression using basis functions.

(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 $\phi(x_i) = [1, x_i, x_i^2, x_i^3]$. 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.



Question 2 continued on next page...

(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?

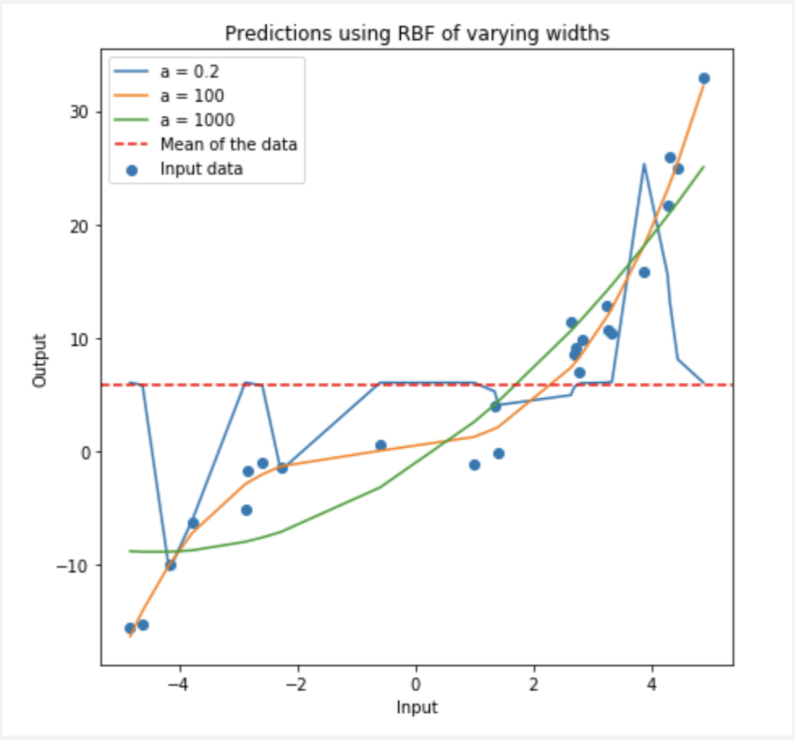
Both models produce a similar fit and MSE values. The fit looks like it captures the shape of the dataset well, indicating that it is a good model of the relationship between the input and the output.

M	MSE
3	2.745
4	2.739

Although MSE value when $M = 4$ is lower than that when $M = 3$, it can be concluded that they result in very similar performance considering the very slight difference, hence I would choose the $M = 3$ model to reduce computations.

Question 2 continued on next page. . .

(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.



For $a = 2$, it can be seen that the input data is not captured well. The plot gives a decent prediction when the input is equal to the centers chosen, but aside from those points, the line tries to stay close to the mean of the data (shown in red dotted line above). For $a = 100$, it can be seen that the shape of the data is captured the best, hence giving the best model to the data. For $a = 1000$, the general trend of positive correlation between input and output data is captured but it does not give a model as good as when a is set to 100.

Question 3 : (26 total points) Decision Trees

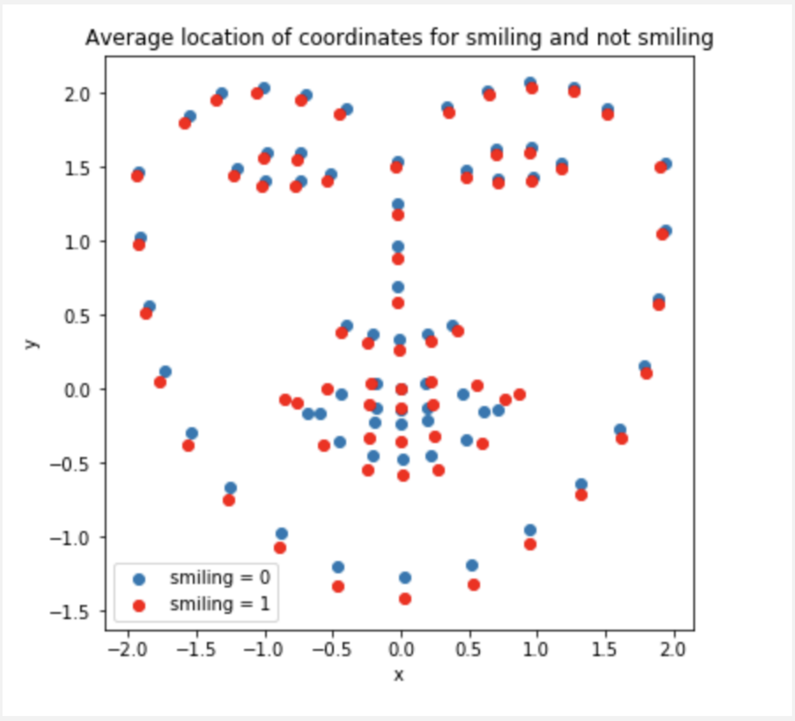
In this question we will train a classifier to predict if a person is smiling or not.

(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.

The training dataset has 4800 entries and the testing dataset has 1200 entries. Each dataset has 136 attributes with values of data type float64. The mean of the target attribute, *smiling*, for both training dataset and testing dataset is approximately equal to 0.5. Since this is a two-class case where *smiling* $\in \{0, 1\}$, this indicates that each set of data has balanced classes.

Question 3 continued on next page...

(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.



From the plot above, the main differences between the case of smiling faces (smiling = 1) and the case of not smiling faces (smiling = 0) can be seen around the lips, including the corner of the lips, and the chin. For the case of smiling, the corner of the lips is more extended out than for the case of not smiling and the chin is extended downwards. Also, it can be seen that the gap between the upper lip and the lower lip is greater in the case of smiling.

Question 3 continued on next page...

(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?

In sklearn, $DecisionTreeClassifier$ uses the Gini index to measure purity by default. An advantage of using this instead of entropy is that you do not need to compute logarithms, which could make implementation slightly faster.

Question 3 continued on next page...

(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.

Larger values for the maximum depth of the tree will cause more nodes to be expanded, which may result in overfitting of the training data. This then results in a decrease in accuracy when predicting the test data as the model has become too complex.
On the other hand, smaller values will cause less nodes to be expanded, which may result in underfitting of the training data.

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.*

Results are presented in the table below.

Maximum Depth	Training Accuracy	Testing Accuracy
2	79.5%	78.2%
8	93.4%	84.1%
20	100%	81.6%

The model with a maximum depth of 8 is the best because it gives the best testing accuracy. With the maximum depth set to 2, both the training and testing accuracy are lower compared to when the maximum depth is set to 8. With the maximum depth set to 20, the training data reaches a perfect accuracy of 100%. However, the testing accuracy is lower compared to the one for the model with a maximum depth of 8 due to overfitting of the training data.

Question 3 continued on next page...

(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`*.*

The top three most important attributes, in order of importance (starting with the most important): x_{50}, y_{48}, y_{29}.
The attribute x_{50} has the highest importance, meaning it gives a split that reduces uncertainty the most. Its values has a mean of -0.20, hence it makes sense in the context of this classification task as it can indicate the lip and chin area, which are the main differences in the case of smiling and not smiling.

Question 3 continued on next page...

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

One limitation of using 2D point locations as input attributes is that they fail to capture a face perfectly because a face is 3-dimensional.

Question 4 : (14 total points) Evaluating Binary Classifiers

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

(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?

Accuracy for each of the four different models are presented in the table below.

Algorithms	Classification Accuracy
alg_1	61.6%
alg_2	55.0%
alg_3	32.1%
alg_4	32.9%

The model that uses algorithm *alg_1* is the best since it gives the best accuracy. The problem with this metric is that it can be misleading if we have imbalanced classes. To improve this without changing the metric, we can multiply FP (False Positives) and FN (False Negatives) with different weights.

Question 4 continued on next page...

(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.

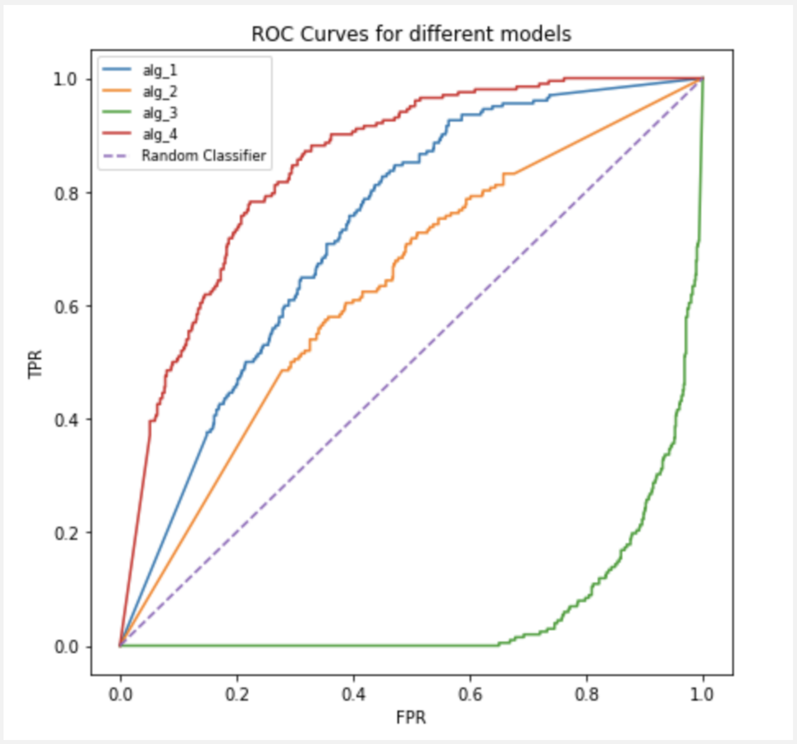
AUC for each of the four different models are presented in the table below.

Algorithms	AUC
alg_1	0.732
alg_2	0.632
alg_3	0.064
alg_4	0.847

The model with the best AUC does not have the best accuracy. This is because for calculating accuracy, we set a threshold of $>=$ 0.5 to convert the continuous classifier outputs into binary predictions. On the other hand, when calculating AUC, we try various threshold settings.

Question 4 continued on next page...

(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.



The ROC curve for alg_3 shows that the performance of the system is worse than that of a random classifier. The FPR is always greater than or equal to the TPR, which means that the proportion of labels that are incorrectly classified as 0 is greater than the proportion of labels that are correctly classified as 0 (and vice versa). Therefore, one way to improve the performance without retraining is to flip the class labels from 1 to 0 and 0 to 1.