Part A (1)

Initial Data Analysis

>The diamond dataset contains 53940 instances of diamond records. From the 11 features, there are 3 categorical (cut, colour, clarity) and 8 numerical (Unnamed, carat, depth, table, x, y, z, price).
>The dataset will contain 53940 instances, this is large enough to ensure we do not require cross validation.
>The scale of the features are different, however we cannot normalise as it would effect the relationship of features x, y, z. Instead we will standardise.
>There are no missing values in the data-set. However theres an indication that there are incorrect values in x, y, z (measurement of height, width, length).
>Our target class price is more heavily distributed towards lower price diamonds with fewer instances of diamonds in higher price ranges. This may effect the accuracy of our prediction for higher priced diamonds.

Preprocess Data

>Remove incorrect instance

From our initial data analysis we identified that there were incorrect values in x, y, z (measurements of 0 which is impossible for a 3d object). To improve the quality of the dataset we remove instances with values of 0 in x or y or z.

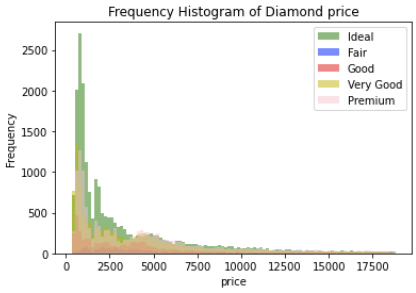
>Remove irrelevant features

From our initial data analysis we identified that there is an unnecessary feature 'Unnamed: 0'. To reduce unnecessary noise we remove this feature to allow for our models to perform more accurately

Exploratory data analysis

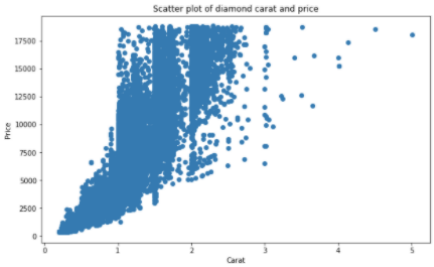
>Histogram (Frequency Histogram of Diamond price segregated by cut)

The results of the graph indicate that some of the lowest quality cut of diamonds can be found in some of the highest price range, this may be an indicator that cut is not a strong feature that can be used to identify the price of a diamond.



>Scatter-plot (Scatter plot of diamond carat and price)

The result of the graph indicates that there is a strong positive correlation between price and carat of a diamond (carat increase results in price increase). This could be a strong feature which we could used to identify a diamond's price.



>Categorical variables numerical

To better understand the relationship between features, to perform a correlation analysis and to perform standardisation we convert categorical features 'cut, colour, clarity' to numerical features based on the ranking associated with each category.

>Correlation

After performing a correlation analysis we identify that x, y, z had a strong positive correlation with each other (over 0.9) as-well as having a strong positive correlation with price (over 0.87) and carat (over 0.92).
Price had a strong positive relationship with carat (0.9).
The dependencies among these features as analysed above can negatively impact our learning model due to "multicollinearity" which can cause misleading and skewed results on a predictive model as it causes a model's coefficients and weights to be very sensitive (overfitting), which will reduce predictive performance when ran on a different dataset (our test dataset). To remedy this we could perform a dimensionality reduction.

>Dimensionality reduction

To reduce noise we remove features x, y, z. This will reduce the risk of multicollinearity increasing the accuracy of our models.

>Standardise (Observe data)

From our data observation (scatter plot) we identify that each feature has different weights, for example without standardise the attribute 'clarity' will play a dominant role as it has a higher range (30-100) that a feature like 'carat' (0-5). To ensure attributes have balance weights we standardise our data.

Regression (default)

MSE (mean_square_error)

Mean Square Error is refers to the square of expect value of the difference between the predicted value and the real value. MSE can evaluate the change extent of the data - the smaller MSE values, the better accuracy of the prediction model.

RMSE (root mean square error)

In the above MSE, we squared the expected value. In this way might lead to dimensional problems - RSME Square root MSE to shows the deviation between the predicted value and the value in a direct way.

RSE (relative squared error)

The relative squared error (RSE) is relative to what it would have been if a simple predictor had been used - best used to determine performance between different models. The larger the RSE value the better performance of the models

MAE (mean absolute error)

Mean absolute error is the average of the absolute error. Which can better reflect the actual situation of the prediction error - the smaller MAE values, the better accuracy of the prediction model.

	Linear	KNN	Ridge	Decision -tree	Random-forest	Gradient - boosting	SGD	SVR	linear SVR	MLP
MSE	1722624.9	719912.91	1722639.6	526902.50	309640.28	430038.49	1726471.0	10151657.	2663680.0	1047053.6
Rank	**6**	**4**	**6**	**3**	**1**	**2**	**7**	**9**	**8**	**5**
RMSE	1312.49	848.48	1312.49	725.88	556.45	655.77	1313.95	3186.17	1632.08	1023.26
RSE	0.89	0.96	0.89	0.97	0.98	0.97	0.89	0.38	0.84	0.94
Rank	**5**	**3**	**5**	**2**	**1**	**2**	**5**	**7**	**6**	**4**
MAE	893.64	476.29	893.63	360.50	285.38	359.63	887.46	1725.24	872.54	607.52
Rank	**9**	**4**	**8**	**3**	**1**	**2**	**7**	**10**	**6**	**5**
Total	***20***	***11***	***19***	***8***	***3***	***6***	***19***	***26***	***20***	***14***
TIME (s)	0.018	0.717	0.012	0.279	15.986	2.984	0.089	194.226	0.048	32.845
Rank	**2**	**6**	**1**	**5**	**8**	**7**	**4**	**10**	**3**	**9**
Total + time	***22***	***17***	***20***	***13***	***11***	***13***	***23***	***36***	***23***	***23***

Top 3 performers are Random-forest, Gradient-boosting and Decision tree. They provide fewer errors, higher RSE and have a reasonable execution time. These three regression predictors are best suited to perform predictions of diamond prices.
SVR was the worst performer providing a high number of error, low RSE and the longest execution time.

Regression (Optimisation)

After tuning the models we found that only SVR, linear SVR and multi-layer perceptron regression benefitted from modification of their default parameters.

```
# Optimisation
# (8-op) support vector regression (SVR)
PrintResult(SVR(C=500), "SVR Regression Optimised")

# (9-op) linear SVR
PrintResult(LinearSVR(C=5, loss='squared_epsilon_insensitive', dual=True), "Linear SVR Regression Optimised")

# (10-op) multi-layer perceptron regression
PrintResult(MLPRegressor(activation = 'relu', solver='lbfgs', learning_rate= 'adaptive'), "MLP Regression Optimised")
```

	Linear	KNN	Ridge	Decision -tree	Random-forest	Gradient - boosting	SGD	SVR	linear SVR	MLP
MSE	1722624.9	719912.91	1722639.6	526902.50	309640.28	430038.49	1726471.0	827959.31	1721745.5	654364.79
Rank	8	5	8	3	1	2	9	6	7	4
RMSE	1312.49	848.48	1312.49	725.88	556.45	655.77	1313.95	909.92	1312.15	808.93
RSE	0.89	0.96	0.89	0.97	0.98	0.97	0.89	0.95	0.89	0.96
Rank	5	3	5	2	1	2	5	4	5	3
MAE	893.64	476.29	893.63	360.50	285.38	359.63	887.46	474.64	894.08	469.22
Rank	9	6	8	3	1	2	7	4	6	5
Total	***22***	***14***	***21***	***8***	***3***	***6***	***21***	***14***	***18***	***12***
TIME (s)	0.018	0.717	0.012	0.279	15.986	2.984	0.089	203.688	9.665	25.119
Rank	2	5	1	4	8	6	3	10	7	9
Total + time	***24***	***19***	***22***	***12***	***11***	***12***	***24***	***24***	***25***	***21***

While the optimisation significantly improved the performance of SVR, linear SVR and multi-layer perceptron regression, this is not significant enough to effect the ranking of Random-forest, Gradient-boosting and Decision tree.

In conclusion, there is a strong liner regression in this dataset. Random Forest regression has the best performance overall when taking all factors into consideration (MSE, RMSE, RSE, MAE and time). Random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

Part A (2)

Initial Data Analysis

>The adult dataset contains 32561 instances of adult records in train and 16281 instances of adult records in test. From the 15 features, there are 9 categorical (Workclass, Education, Marital Status, Occupation, Relationship, Race, Sex, Country, Above/Below 50K) and 6 numerical (Age, Final_Weight,Education Num, Capital Gain, Capital Loss,Hours/Week).
>The train dataset will contain 32561 instances, this is large enough to ensure we do not require cross validation.
>There are inconsistencies of how Above/Below 50k is recorded in the test and train dataset
>There are no null values in the data-set. However theres entries which are not completed marked by '?'
>Our target class income is more heavily distributed towards income over 50k with fewer instances of adults in with income below 50k. This may effect the accuracy of our prediction for adults who earn below 50k.

Preprocess Data

>Remove incorrect instance

From our initial data analysis we identified that there were incorrectly entered values ('?'). To improve the quality of the dataset we remove instances with values of '?'.

We also found that there were inconsistencies in how Above 50k is recorded in the test and train dataset - we correct this by removing the full-stop in this feature's instances in the train and test dataset. We then convert these values to binary (0,1) for convenience.

>Remove irrelevant features

Education and Education Num represents the same thing. Education is a categorical item which will be converted to numerical. As we have Education Num we do not require the feature Education thus it can be removed.

Final weight represents estimated populations who have similar background or census information this does not have a strong relationship with determining a person's income. We do not require the feature Final weight thus it can be removed.

>Convert categorical data to numerical

Due to the nature of the categorical features in our dataset such as Countries where a scale would not be appropriate to represent the features, we instead create new features from these categories and give them a value of 0,1 based on if the feature is false or true for an instance.

>Removing unnecessary column in train

After converting the categorical data to numerical by increasing the number of features we found that there was a greater amount of features in the train dataset. We identified which feature this was and removed it from the dataset as it is unnecessary if not found in the test dataset.

Exploratory data analysis

>Bar graph

From observing the bar-graph we were able to better understand the distribution of the numerical feature. There seems to be some values which are distant from the central bars - this may indicate that there are outliers within the data however after removing outliers we found that it was more beneficial to maintain these instances.

Classification (default)

	KNN	GaussianNB	SVM	Decision-Tree	Random-forest	AdaBoost	Gradient Boosting	LinearDiscriminantAnalysis	MLP	Logistic Regression
Accuracy	0.80	0.58	0.78	0.77	0.81	0.83	0.84	0.82	0.82	0.82
Rank	***5***	***8***	***6***	***7***	***4***	***2***	***1***	***3***	***3***	***3***
Precision	0.86	0.95	0.81	0.86	0.86	0.87	0.87	0.87	0.88	0.87
Rank	***4***	***1***	***5***	***4***	***4***	***3***	***3***	***3***	***2***	***3***
Recall	0.88	0.48	0.93	0.83	0.89	0.92	0.92	0.89	0.88	0.90
Rank	***5***	***7***	***1***	***6***	***4***	***2***	***2***	***4***	***5***	***3***
F1	0.87	0.63	0.86	0.85	0.87	0.89	0.89	0.88	0.88	0.88
Rank	***3***	***7***	***5***	***6***	***4***	***1***	***1***	***2***	***2***	***2***
Auc	0.72	0.7	0.63	0.71	0.72	0.74	0.75	0.75	0.75	0.74
Rank	***3***	***5***	***6***	***4***	***3***	***2***	***1***	***1***	***1***	***2***
Total (excluding accuracy)	***15***	***20***	***17***	***20***	***15***	***8***	***7***	***10***	***10***	***10***
TIME (s)	19.144	0.259	134.542	0.555	5.971	6.170	6.412	1.285	61.623	13.067
Rank	8	1	10	2	4	5	6	3	9	7
Total + time	***23***	***21***	***27***	***22***	***19***	***13***	***13***	***13***	***19***	***17***

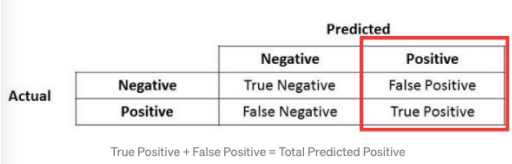
Accuracy

Accuracy represents the ratio of correctly predicted instances. It is a good aspect to judge the performance of the algorithm. In many cases, it is a useful indicator. However, accuracy is very dependent on the quality of the data.

The training set for adults is an imbalanced data set which means the model we predict maybe not very accurate. This is because instances of adults earning less than 50k make up less than a quarter of our train dataset. We can observe this in the performance of all our models being less than 85% accurate. Thus Accuracy is not a suitable form of measurement of performance for this dataset.

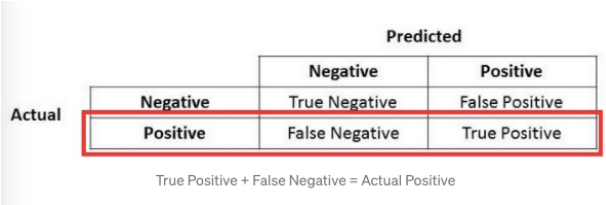
Precision

Precision is the ratio of correctly predicted positive observations (true positive) to the total predicted positive observations. A model that produces no false positives has a precision of 1.0 - Precision is a good measure to determine, when the costs of False Positive is high (making incorrect guess of a person who earns under 50k as over 50k is unacceptable).



Recall

Recall the ratio of correctly predicted positive observations (true positive) to the total predicted of true positive and false negative observation. We use to select our best model when there is a high cost associated with False Negative (making incorrect guess of a person who earns over 50k as under 50k is unacceptable).



F1

F1 Score is the weighted average of Precision and Recall. F1 Score might be a better measure to use if we need to seek a balance between Precision and Recall and there is an uneven class distribution. The nature of the adult data is uneven and there is no significant weights associated with false negative nor false positive.

AUC

AUC provides an aggregate measure of performance across all possible classification thresholds.AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0.0; one whose predictions are 100% correct has an AUC of 1.0. AUC tells us how much the model is capable of distinguishing between classes (wether a person earns over or under 50K).

Two best algorithms

Accuracy: Gradient Boosting, AdaBoost
Precision: GuassianNB, MLP
Recall: SVM, Gradient Boosting, AdaBoost
F1: Gradient Boosting, AdaBoost
AUC: Gradient Boosting, LinearDiscriminantAnalysis, MLP

Overall: Gradient Boosting, AdaBoost

Observed by the rank in the table Gradient Boosting and AdaBoost performs the best in the overall performance measures. However as our dataset is imbalanced and there is no significant weight associated with false negative nor false positive (we do not know what the classifier outcome will be used for hence this could change) it is most appropriate to prioritise a classifier's performance based on F1 and AUC followed by precision and recall then accuracy.

After ranking we are able to distinguish our best classification performer Gradient boosting, however it is unclear which is our second best performer as LinearDiscriminantAnalysis, MLP, AdaBoost performed equally well in F1 and AUC. However after taking precision and recall into account we can identify AdaBoost as our second best performer.

Gradient boosting generates learners during the learning process. It build first learner to predict the values/labels of samples, and calculate the loss (the difference between the outcome of the first learner and the real value). It will build a second learner to predict the loss after the first step. The step continues to learn the third, forth… until certain threshold.

Adaptive boosting requires users specify a set of weak learners (alternatively, it will randomly generate a set of weak learner before the real learning process). It will learn the weights to modify weak learners into a strong learner. The weight of each learner is learned by whether it predicts a sample correctly or not. If a learner is mis-predict a sample, the weight of the learner is reduced. This process repeats until convergence.