saxtonjosh
300375062

A4 Performance Metrics and Optimisation

Part 1: Performance Metrics in Regression

Based on an initial look at the dataset and domain knowledge about diamonds, the Cut, Clarity and Color features are ordinal in nature, but when looking at the histograms in initial EDA it is clear these are not in the correct order by default. Hence, I decided to use an ordinal encoder with the orderings as shown below.

Cut Scale	0	1	2	3	4	5	6	7	8	9	10
	AGS Ideal	AGS Excellent	AGS Very Good	AGS Good		AGS Fair			AGS Poor		

Color Scale	0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0	5.5	6.0	6.5	7.0	7.5	8.0	8.5	9.0	9.5	10.0	To	Fancy Yellow
	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	Fancy Yellow
	COLORLESS			NEAR COLORLESS				FAINT			VERY LIGHT					LIGHT								

Clarity Scale	0	1	2	3	4	5	6	7	8	9	10
	FLAWLESS / IF	VVS1	VVS2	VS1	VS2	SI1	SI2		I1	I2	I3

```
numeric_data.skew(axis = 0)
```

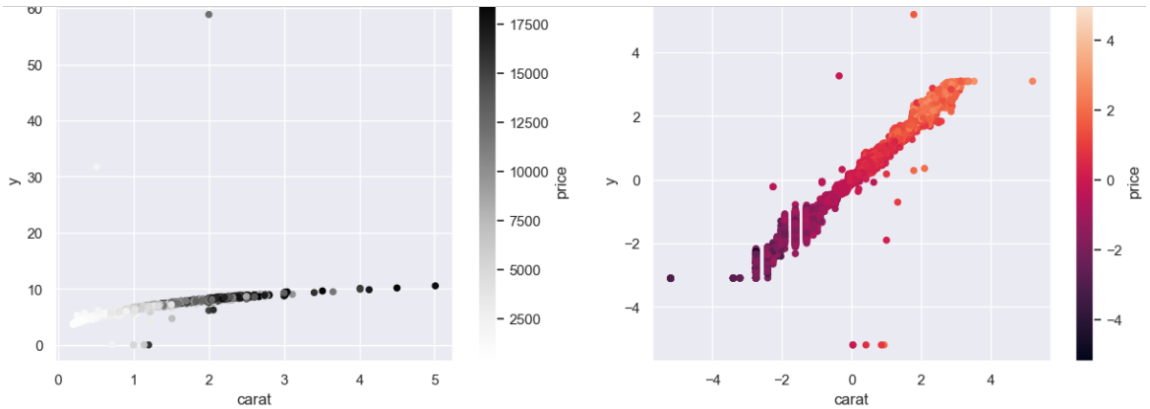
```
carat    1.115709
depth   -0.174914
table    0.827762
x        0.372581
y        3.229834
z        2.002532
price    1.613360
dtype: float64
```

```
numeric_data.kurtosis(axis = 0)
```

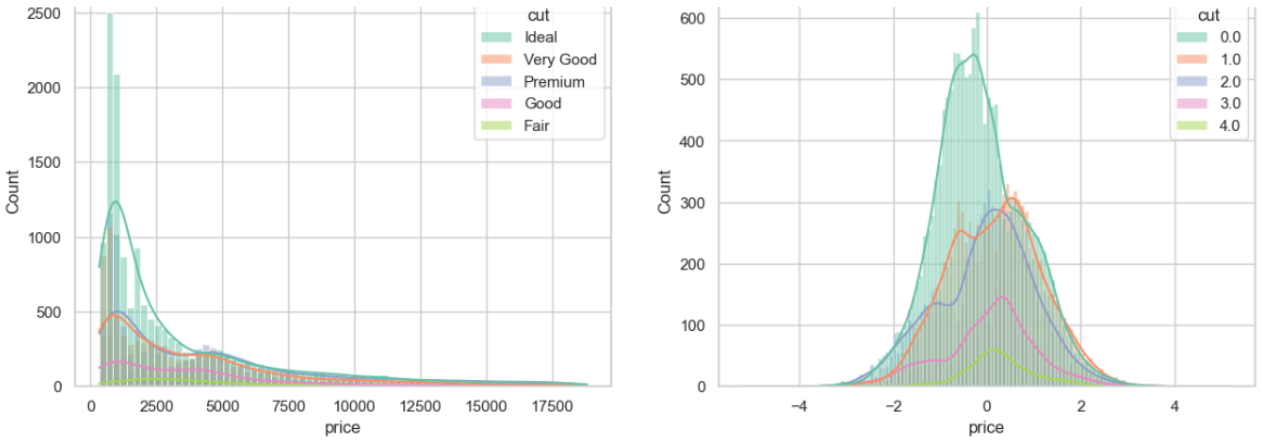
```
carat      1.306816
depth      5.938881
table      3.493664
x         -0.614729
y        125.733270
z         66.397789
price      2.172815
dtype: float64
```

To deal with the right skewness of the target variable price used a TransformedTargetRegressor using a QuantileTransformer to map the target variable price to a normal distribution when training but transforming it back when calculating errors as this will give average error of the price of the diamond instead of some number between 0 and 1 that is hard to interpret its meaning.

For the numerical data, I decided to use the QuantileTransformer() method from sci-kit learn before sending the data to the regression models as a lot of the numeric features seemed to have outliers that are causing the distributions to have high kurtosis and the distribution of some of the features namely (price, carat, y, z) are heavily right-skewed. Even table is moderately right-skewed. This handles the outliers by mapping them to the extreme ends of the transformed data like in the scatter plot example below showing how in the case of y the large outliers now have less of an impact on the training.



We see in a lot of these projections and in the heatmaps that we have features that are highly correlated with each other using their spearman correlation values as spearman is good for ordinal values and more robust to the outliers that are present in the data.



The plot above shows how the distribution of price is handled in the training stage.

Tried using SelectKBest from sci-kit learn using r_regression Pearson correlation between each feature and the target variable as the scoring for feature selection. However, this decreased the performance across all regressors with some minor decreases in execution time after taking away 3 or more features. While taking away 1 or 2 features using this method didn't affect performance by too much having slightly worse performance (differences of 1-8 in MAE scores vs differences in the hundreds with 6 or fewer features.).

Results

	fit_time	score_time	test_mse	test_rmse	test_rse	test_mae
random_forest	12.19	0.41	323376.80	568.66	0.02	267.57
gradient_boosting	3.20	0.05	404892.08	636.31	0.02	324.19
multi_layer_perceptron	6.38	0.04	408876.50	639.43	0.03	342.51
support_vector	35.10	21.73	474266.97	688.67	0.03	374.01
decision_tree	0.33	0.03	585564.39	765.22	0.04	368.44
k_neighbors	0.24	0.68	644830.20	803.01	0.04	423.56
stochastic_gd	0.13	0.03	959889.79	979.74	0.06	565.74
ridge	0.11	0.04	990344.86	995.16	0.06	576.68
linear	0.14	0.05	990855.81	995.42	0.06	576.81
linear_svr	2.27	0.03	1035373.39	1017.53	0.06	555.82

.
The table above is ordered by MSE and shows the order of classifiers from best to worst. Most notably the MAE of the linear SVR and support vector regressors appear to show a difference in order to the other metrics, if ordered by MAE linear SVR would move into 7th above SGD and the support vector regressor would move down one place to 5th.

This has some interesting implications being that RMSE gives a higher weight to large errors so is rather important when the target is money the difference between being $1000 off and $500 off on average is important when the prices range between $300 and $20000. In fact, across all regressors, the RMSE is almost double that of the MAE. this shows that there are either some large variances in the errors or perhaps a small number of large outliers that are affecting this score.

In terms of the RSE error, this is perhaps not the best measurement of fitness for this task as without looking at the residual patterns to determine if they are non-random we can't tell if the model is consistently over and underestimating the price showing a bias towards a high r2_score leading to a low RSE. However, in conjunction with the other error metrics gives a decent estimate of model performance.

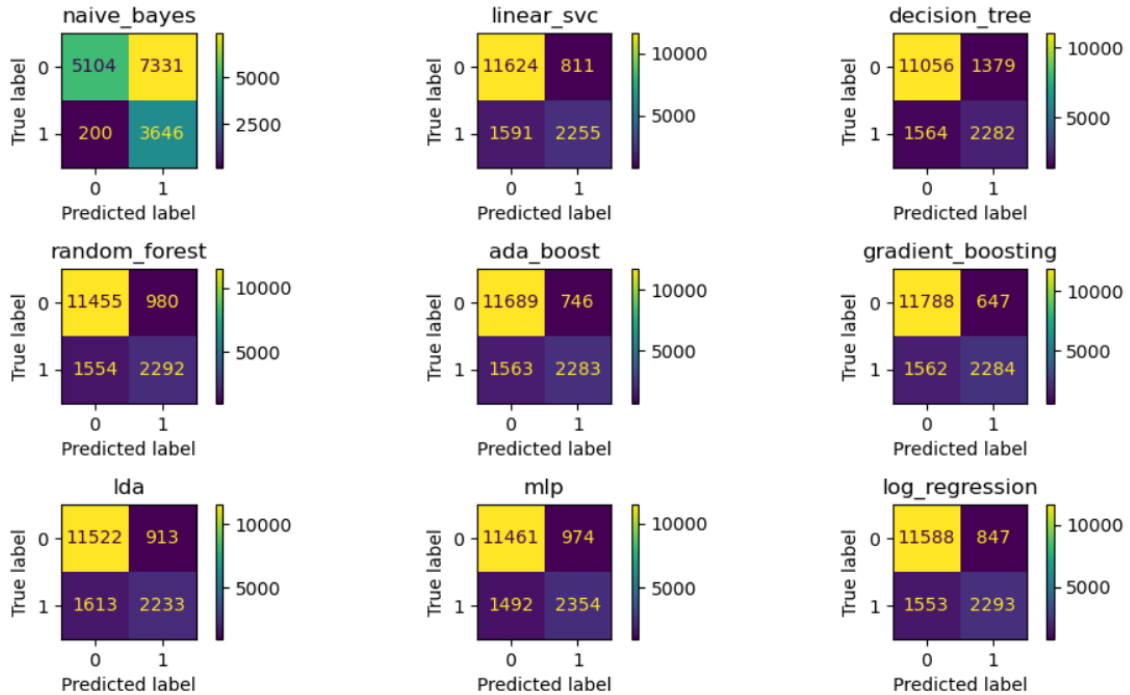
random_forest, gradient boosting, and MLP regressors are consistently top 3 across all metrics apart from execution time. Although in cross-validation the MLP regressor performance was worse by $90 in MAE dropping it to 5th below the decision tree regressor.

While SGD, ridge and linear regression models are among the bottom 4 they are the 3 fastest in terms of execution time. The decision tree regressor did the best out of the simple fast models and a model with a good compromise between accuracy and execution time was the gradient boosting regressor.

Part 2: Performance Metrics in Classification

Looking at the data initially it appears there is a space before every categorical variable string when loaded in so have to make sure this is accounted for when scaling and transforming variables by name. In the numerical variables, capital gain and capital loss are only a factor in 2700 and 1600 instances respectively however these two variables appear to split these particular instances quite well when used with education-num and hours per week. The “fnlwgt” feature doesn’t seem usable considering it is only able to compare people in the same state in America, which is information that is not in the data given so I dropped that feature. I also dropped the “education” feature as the “education-num” feature is just the ordinally encoded version of this feature which I used instead.

The missing data is in the “occupation”, “workclass” and “native-country” features with all of the instances missing “workclass” also missing “occupation” at 1836 and 1842 respectively just over 5% of our training data so should be accounted for while native-country is missing in 583 instances just under 2% of the training data. Decided to impute using the strategy of most common among the training data this being the classes “Private”, “Prof-specialty” and “United-States”. Decided to OneHot Encode the remaining categorical variables as they are not ordinal in nature only “education” is. Scaled the numerical features with StandardScaler from the sci-kit learn library as age and hours worked range up to 99 and the capital features are much larger capping at 99999. There was also the fact that in the test data the class labels had full stops at the end so had to reformat the test data so that each set was using the same class labels. United States is overwhelmingly the majority “native-country”.



	fit_time	score_time	test_acc	test_prec	test_rec	test_f1	test_roc_auc
gradient_boosting	4.37	0.05	0.86	0.78	0.59	0.59	0.77
ada_boost	1.60	0.24	0.86	0.75	0.59	0.59	0.77
log_regression	0.47	0.02	0.85	0.73	0.60	0.60	0.76
linear_svc	1.76	0.03	0.85	0.74	0.59	0.59	0.76
mlp	48.92	0.05	0.85	0.68	0.65	0.65	0.78
lda	0.39	0.02	0.84	0.71	0.58	0.58	0.75
random_forest	3.31	0.35	0.84	0.70	0.60	0.60	0.76
kneighbors	0.04	1.37	0.83	0.67	0.60	0.60	0.75
decision_tree	0.25	0.03	0.82	0.62	0.60	0.60	0.74
naive_bayes	0.06	0.05	0.54	0.33	0.95	0.95	0.68

The two best algorithms for accuracy and precision were gradient_boosting and ada_boost whereas for AUC ROC the mlp classifier did the best, followed by those two. However, for the recall and f1 scores, naive_bayes and the mlp classifier are the top 2. With the lower precision and accuracy of the naive bayes classifier combined with high recall, this means that the classifier is sampling a lot more of the >50K class but not getting the class label correct as shown in the confusion matrices above. The lower scores for the gradient boosting and ada_boost in these two metrics are likely due to the class imbalance in the training data as seen below. In order to improve these metrics, I would need to try some random over/under-sampling of the original training data to offset the class imbalance of 70% <=50K 30% >50K. Considering the uses for this data like setting thresholds for tax brackets I imagine having a few extra false positives for >50K wouldn't be the worst thing for those who were making the model.