AIML421_A4

kakauchad_300212228

2022-10-13

Performance metrics in Regression

Part I: Learn a regression model to discover the relationship between the price of round cut diamonds and the 10 input variables provided in as the feature vector.

Expect to use the following steps:

1. Load data
2. Initial data analysis
3. Preprocess data
4. Exploratory data analysis
5. Build regression models using the training data
6. Evaluate models by using cross validation
7. Assess model on the test data

Use random_seed = 309, 70% train: 30% test split and test these 10 algorithms:

- linear regression
- k-neighbours regression
- ridge regression
- decision tree regression
- random forest regression
- gradient boosting regression
- stochastic gradient descent (SGD) regression
- support vector regression (SVR)
- linear SVR
- Multi-layer perceptron (MLP)

Initial Data analysis

The diamonds dataset has:
- 53940 rows of data with 11 features: - the target feature is 'price', a numeric feature that represents the dollar price of a given diamond - each feature describes some quality of a diamond
- numeric features:
- carat - mass of the diamond, 1 carat = 200mg; positive float - depth - depth excluding the diamond's face; positive integer - table - width of the face of the diamond - x - width of the diamond at it's widest point; should be a positive float - y - length of the diamond at its longest point; should be a positive float - z - depth of the diamond; should be a positive float - 'cut', 'color', and 'clarity' are categorical features representing graded qualities of a diamond, and are ordinal - 'cut' has 5 levels: ['Ideal' 'Premium' 'Good' 'Very Good' 'Fair'] - 'color' has 7 levels: ['E' 'I' 'J' 'H' 'F' 'G' 'D'] 'clarity' has 8 levels: ['SI2' 'SI1' 'VS1' 'VS2' 'VVS2' 'VVS1' 'I1' 'IF']

The dataset has no missing values and no null values.

Preprocess the data includes ordinal encoding spliting for training

- The dataset has three categoric features ('color', 'clarity', and 'cut'), these are ordinal categoric features. For a regression model, we want numeric data so we should encode these features, ensuring ordinality is retained
- The dataset has no missing features so we don't need to handle missing features or null values (i.e. no need for imputing, removal etc.)
- We should separate the input feature set from the target feature ('price'), and we can use the train, test split function to separate target features and train/test sets

Exploratory data analysis shows well-behaved numeric features

The numeric features appear fairly well behaved:
- 'depth', 'table', 'x' all appear to have generally normal distribution
- 'carat' appears to have an exponential distribution - 'y' and 'z' have very heavy right tails, and tend to have most instances at the lower end of their respective scales

Because the numeric features are fairly well-behaved (i.e. conform to known distributions) we will do minimal preprocessing and turn to building the regression models

Build regression models using the training data

We will use sklearn to build a pipeline to efficiently manage our 10 models and apply our target performance metrics, using Mean Square Error (MSE) as our primary metric. Our preprocessing pipeline will include: - ordinal encoding of categorical features - scaling our data (using StandardScaler()), and - fitting a regression model (initially Linear Regression)

We will test the following regression methods: 1. linear regression
2. k-neighbours regression
3. ridge regression
4. decision tree regression
5. random forest regression
6. gradient boosting regression
7. stochastic gradient descent (SGD) regression
8. support vector regression (SVR)
9. linear SVR
10. Multi-layer perceptron (MLP)

Regression results

The results of the regression models are presented below:

- almost all models perform consistently across each metric (i.e. the models rank in the same order across all metrics)...
- ... apart from MLPRegressor, which performs worse on the RMSE than the MSE metric
- for the primary metric (MSE) LinearSVR is an order of magnitude ahead of the next two models MLPRegressor and SVR, which are also an order of magnitude ahead of SGDRegressor
-
- The best performing model against all metrics was:

Performance metrics in classification