MACHINE LEARNING HW2

HPE Machine Learning
Julian
6/24/2024

Assignment Instructions

Your assignment is to create a Jupyter notebook that demonstrates how to do the following (use methods discussed in the class materials shared so far):

1. Load the dataset in the file named `winequality_white.csv` and produce at least one table and one graph that summarize the dataset statistics. Set up a classification problem: predicting the quality value (a single variable with seven classes labeled 3, 4, 5, ..., 9) based on the values of all the other variables in the file (acidity, alcohol, pH, etc.) and split the dataset into separate training and test sets in a reproducible way; (2 points)

2. Train two models to solve this classification problem: one based on Decision Trees (e.g., `DecisionTreeClassifier`, `RandomForestClassifier`) and one based on SVMs (e.g., an `SVC` with your choice of kernel). Use the training set you created in part 1 to cross-validate the performance of each model. Report statistics for the scoring method of your choice (e.g., accuracy, weighted precision, macro recall, f1 score); (6 points)

3. Use `GridSearchCV()` and the training dataset from part 1 to tune the Decision Tree family model from part 2; compare its performance when changing at least two different hyperparameters (e.g., tree depth) across a range of values. (6 points)

4. Use the `make_pipeline()` method and the training dataset from part 1 to study and describe the impact of data transformations on the performance of the SVM family model from part 2. You can try dimension reduction (e.g., using different `n_component` values for `PCA`) and/or data scaling (e.g., `MinMaxScaler`). (6 points)

5. Train the `DummyClassifier()` on your training set. Use your test set to compare the performance of this `DummyClassifier()` and the best model versions from parts 3 and 4. Discuss your overall results. (2 points)

`DummyClassifier()`: `https://scikit-learn.org/stable/modules/generated/sklearn.dummy.DummyClassifier.html`

Submission Instructions

- Please name your homework submission as follows: `311_lastName_firstName_assignmentNumber.ipynb`
- How to submit: Please submit homework in Moodle.

1. Dataset Upload and Summary

Our task is to set up a classification problem to predict wine quality using the features in the dataset. As a preliminary practice, I produce a numerical and graphical summary using the pandas `describe()` command and box-whiskers plot functionality.

Index	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
count	4898	4898	4898	4898	4898	4898	4898	4898	4898	4898	4898	4898
mean	6.8548	0.27824	0.33419	6.3914	0.045772	35.308	138.36	0.99438	3.1883	0.48985	10.514	5.8779
std	0.84387	0.10079	0.12102	5.0721	0.021848	17.007	42.498	0.0029909	0.151	0.11413	1.2306	0.88564
min	3	0.08	0	0.6	0.009	2	9	0.98711	2.72	0.22	8	3
25%	6.3	0.21	0.27	1.7	0.036	23	108	0.99172	3.09	0.41	9.5	5
50%	6.8	0.26	0.32	5.2	0.043	34	134	0.99374	3.18	0.47	10.4	6
75%	7.3	0.32	0.39	9.9	0.065	46	167	0.99614	3.28	0.55	11.4	6
max	14.2	1.1	1.66	65.8	0.346	289	440	1.039	3.82	1.08	14.2	9

Table 1: Summary statistics of the wine quality dataset.

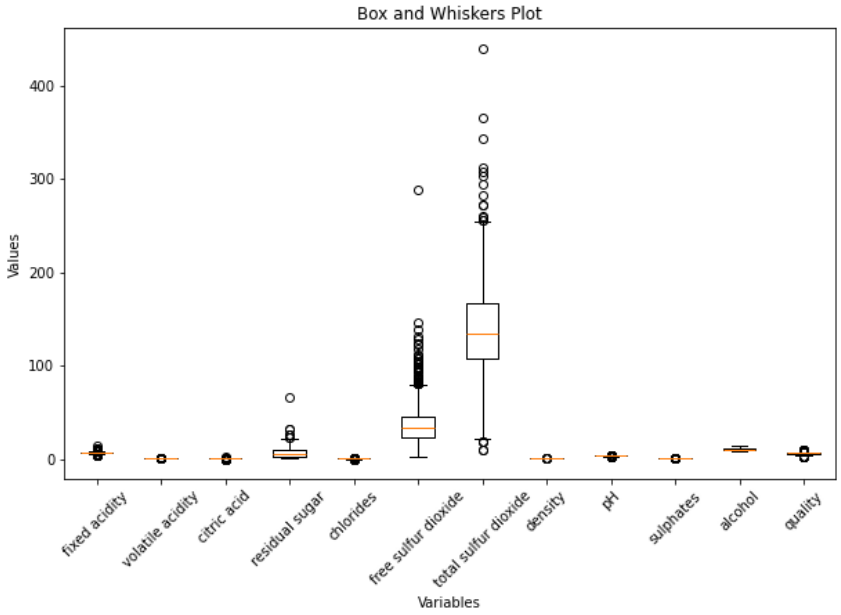

Figure 1: Box and Whiskers Plot

2. Training a Support Vector Machine and a Decision Tree Classifier Model

We first use `scipy` to train a support vector machine with a regularization constant equal to 1 and a linear kernel. A linear kernel means that the decision boundary between the classes is a straight line (or a hyperplane in higher dimensions). It is the simplest kernel and is useful when the data is linearly separable. C is a regularization parameter. Regularization helps prevent overfitting by controlling the trade-off between achieving a low training error and a low testing error. The parameter C controls the strength of the regularization. Similarly, we use `scipy` to train a decision tree model. We evaluate the model using cross-validation via 4-fold validation and the F1 Macro score. We report the average across four folds.
Mean F1 score for SVM: 0.43835089822707496
Mean F1 score for Decision Tree: 0.33079022958830573

3. Tuning the Decision Tree

Our next objective is to tune the decision tree. We initialize an instance of the decision tree with `scipy`, and then use grid search to tune the decision tree over the following hyperparameters: max depth: 5, 6, 7 and criterion: gini, entropy. After fitting the model and validating, we discover that the optimal settings are with a maximum depth of 7 and criterion gini. We compare this optimal decision tree to variants in the table below:

	Max Depth	**Criterion**	**F1 Score**
Model 1	7	Gini	0.2573
Model 2	6	Gini	0.24

Table 2: Comparison of models based on hyperparameters and F1 score

4. Analyzing the Impact of Data Transforms

Now we analyze the impact on F1 score when using different data transformations. We vary the number of principal components and even change the transformation to binarizer as well. As expected, using more principal components improves performance. The binarizer appears to not perform well at all, likely because our data labels are not binary.

	Transformation	**PCA**	**4-fold F1 Score (Average)**
Model 1	MinMaxScaler	1	0.377211997
Model 2	MinMaxScaler	2	0.378371155
Model 3	MinMaxScaler	3	0.41655738
Model 4	MinMaxScaler	Not used	0.435117719
Model 5	Binarizer	Not used	0.27351196

Table 3: Comparison of models based on transformations and F1 score

5. Final Evaluation of Models

We test the following models on the test set and evaluate their performance using accuracy. Our results suggest that the dummy classifier performs poorly in comparison to the decision tree and SVM. Decision trees and SVMs utilize the features of the data to make informed decisions, whereas a dummy classifier does not.

Model Names	**Accuracy**
Dummy Classifier	0.465306
Tuned Decision Tree (6 depth, gini)	0.55408
Minmax SVM	0.55102

Table 4: Comparison of models based on accuracy