Scikit Learn Lab

Fall 2019

Due Date: Nov 8, 2019

Instructions

- This lab will involve coding in SciKit Learn. You are free to build on the examples presented in the class or develop your own code.
- All instructions for compiling and running your code must be placed in the README file. Please do not hard code any paths from your local computer. In addition, you should write a report detailing the results, and what you learned by doing these experiments.
- If you use any references or online sources, please list them.
- You should use a cover sheet, which can be downloaded from here
- This is an individual assignment.
- No free days can be used for this lab assignment.
- Please ask all questions on Piazza, not via email.

1 Parameter Tuning of Classification Models using GridSearch

We learned that parameter tuning is one of the most important parts in a machine learning project. Rather than having to manually try every combination of parameters, Scikit Learn provides tools that can help automate this process. In this lab assignment, you will use Grid search for automatic parameter tuning for some of the classification algorithms that we have studied in class.

1.1 Example Code

An example code for SVC classifier applied on digits dataset is provided. In that snippet of code, the *tuned_parameters* variable contains the parameters and their values that need to be checked, and the best combination is to be used further. The output metrics are shown on the screen.

1.2 Algorithms To Train

You have to train the following algorithms mentioned below on another classification dataset of your choice taken from UCI ML Repository - `https://archive.ics.uci.edu/ml/index.php`.
Note that you cannot use the datasets below

1. Digits Dataset that is part of Scikit Learn
2. Iris Dataset, as it its overly simplistic

The parameter names for each algorithm are mentioned below. It's up to you to decide the number of values to test for each parameter. The idea is that you should search for and discover the best set of parameters. If you don't understand the meaning of a parameter, please lookup the documentation page. Links are provided with the name of the algorithm in the list below:

List of Algorithms and Prameters

1. Decision Tree
   Parameters: At least four out of the following:
   max_depth,
   min_samples_split,
   min_samples_leaf,
   min_weight_fraction_leaf,
   max_features,
   max_leaf_nodes,
   min_impurity_decrease
2. Neural Net
   Parameters: At least four out of the following:
   hidden_layer_sizes,
   activation,
   alpha,
   learning_rate,
   max_iter,
   tol,
   momentum,
   early_stopping
3. Support Vector Machine
   Parameters: At least four out of the following:
   C (Error Penalty),

kernel and associated parameters i.e. if you choose polynomial kernel, you need to vary *degree* or if you choose rbf kernel, you need to vary *gamma*
max_iter,
random_state

4. Gaussian Naive Bayes
   Parameters Only one parameter available:
   priors

5. Logistic Regression
   Parameters: At least four out of the following:
   penalty,
   tol,
   C (inverse of regularization strength),
   fit_intercept,
   class_weight,
   max_iter,
   multi_class

6. k-Nearest Neighbors
   Parameters: At least four out of the following:
   n_neighbors,
   weights,
   algorithm,
   p (power parameter for the Minkowski metric)

7. Bagging
   Parameters: At least four out of the following:
   n_estimators,
   max_samples,
   max_features,
   random_state

8. Random Forest
   Parameters: At least four out of the following:
   n_estimators,
   criterion,
   max_depth,
   max_features,
   min_samples_split,
   min_samples_leaf

9. AdaBoost Classifier
   Parameters: At least four out of the following:
   n_estimators,
   learning_rate,
   algorithm,
   random_state

10. Gradient Boosting Classifier
    Parameters: At least four out of the following:
    loss,
    learning_rate,
    n_estimators,

max_depth,
min_samples_split,
min_samples_leaf,
max_features,
min_impurity_decrease

11. XGBoost
    If the above link doesn't provide you with the full details, consult this one - https://xgboost.readthedocs.io/en/latest/python/python_api.html#module-xgboost.sklearn
    Parameters: At least four out of the following:
    learning_rate,
    n_estimators,
    min_child_weight,
    max_delta_step,
    booster,
    seed

1.3 Evaluation Methodology and Metrics

For evaluating these algorithms, you will need to use a **GridSearchCV** object. Details are available at : http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

Below are some more requirements:

- The **cv** parameter should be at least 5
- The **scoring** parameter can be *accuracy*, or *precision_macro* or *recall_macro*
- The **param_grid** parameter needs to be specified, just like in the example code
- Please do not hard code any paths from your local computer

The output should be in a tabular form like below:

Algorithm	Best Parameters	Avg Precision	Avg Recall	Avg F1	Accuracy_Score
Decision Tree	max_depth=5, ...	0.99	0.97	0.98	0.98
Neural Net	hidden_layer_sizes = (10, 10), ...	0.95	0.96	0.96	0.97
...	...	...	...	...	...

1.4 What to Submit

Submit the following:
- Public link to your notebook in Google Colab
- A brief report explaining the results. Which method(s) performed best and why do you think so? What could be done to improve the results?