COSC 425 Intro to Machine Learning
Project 5: Support Vector Classification

Ryan Pauly
University of Tennessee Knoxville

Date Submitted: December 7, 2019.

Project Description

The purpose of project 5 is to firstly become familiar with support vector classification, a machine learning method for making predictions based upon given parameters. Secondly, and arguably more importantly, is scikit-learn, a Python library which excels at providing optimized functions for machine learning problems. This project aims to give a more realistic approach to using machine learning, as the purpose of this lab is to find results with scikit-learn's built in functionality.

Pre-processing Steps

Pre-processing steps for the three different datasets in this project was very simple. In part 1, the binary classification problem asks to have "good" and "bad" predictions made. The last column in part 1's ionosphere.data contains all numeric values except for the last column in which the values are objects of "g" or "b." To handle these values, "g" values were replaced with a numeric value of 0 and "b" values with a numeric value of 1. Next, part 2's dataset, vowel-context.data is again very straightforward. In the vowel-context.data the project 5 outline informs that the first three columns of the dataset are irrelevant to this project and thus these columns are removed from the dataset.

For parts 1 and 2, the data is divided into training and testing, where training is 80% and testing is 20% of the complete dataset respectively. Part 3's dataset is already pre-partitioned into two different files. No validation set is necessary for this project as validation is handled in a k-fold cross validation method in which the training data is partitioned into training and validation within the k-fold parameter provided by the scikit-learn function *GridSearchCV*. Finally, the data is all standardized thanks to sklearn's ("sklearn" is a common abbreviation for the scikit-learn library) built in preprocessing functionality, in which the standard scalar function is used to standardize all the training and testing data.

Description

Support vector classification (henceforth referred to as SVC), is a method of discriminant classification. SVC essentially discriminates the classifiers with the use of a hyperplane on a 2-D space to "group" different items into their respective categories or classifications. These hyperplanes can take linear forms as well as non-linear forms to provide the generalization of a group of datapoints. In this project scikit-learn provides functions to implement SVC. An example in more mathematical terms can be expressed as the following:

Given a training vector $x_i \in \mathbb{R}^P, i = 1, \ldots, n$ in tow classes, and a vector $y \in \{1, -1\}^n$, SVC solves the problem with the simplified formula of:

$$\min_{\alpha} \frac{1}{2}\alpha^T Q \alpha - e^T \alpha$$
$$\text{subject to } y^T \alpha = 0$$
$$0 \leq \alpha_i \leq C, i = 1, \ldots, n$$

Where e is the vector of all ones, C > 0 is the upper bound, Q is an n by n positive semidefinite matrix, $Q_{ij} \equiv y_i y_j K(x_i, x_j)$, where $K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$ is the kernel in which training vectors are implicitly mapped into a higher dimensional space by the function ϕ. ("1.4. Support Vector Machines — Scikit-Learn 0.22 Documentation"). The decision function thus is as follows, courtesy of scikit-learn's SVM documentation:

$$\operatorname{sgn}\left(\sum_{i=1}^{n} y_i \alpha_i K(x_i, x) + \rho\right)$$

GridSearchCV is a built-in optimized exhaustive search function which helps to find the best possible parameters required for the SVC machine learning algorithm to perform and estimate to its highest potential. Additionally, GridSearchCV also has a built-in k-fold cross validation capability, and thus it also handles the cross validation by splitting the training set passed in to perform k-fold cross validation. The two primary concerns in project 5 is to find the best error penalty and kernel scale parameters. Error penalty is referred to as "C", and the kernel scale is referred to as "gamma" in the python files of part 1, 2, and 3. A coarse grid search is first performed to find the "range" in which the best parameters, then a fine grid search is performed over the range discovered by the coarse grid search. With these newfound parameters, the SVC can be fitted with the new parameters and be used to improve the accuracy of the estimator. Lastly, the kernel for the SVC is set for parts 1, 2, and 3 as "rbf" which is an exponential algorithm and the default when no other kernel is provided due to its ability to generalize fairly well on most classifier problems.

To standardize the dataset, we perform a calculation which scales the variables down significantly. The goal with standardization is to scale the data down to a mean which equals 0 and a standard deviation equal to 1. To achieve this, we use the z-normalization formula:

$$z = \frac{X - \mu}{\sigma} \tag{10}$$

X represents the value we wish to standardize, and *Z* is the result of the standardization. We subtract from *X* the mean of *X* or μ, and lastly, we divide it by the standard deviation of *X* or σ. Sklearn's preprocessing functionality handles this standardization process.

Analysis

Apart from part 3 due to its pre-partitioned files, the dataset is randomly shuffled and then partitioned into an 80% and 20% split for training and testing respectively. Thus, results in parts 1 and 2 are at the mercy of the randomization of the datasets as well as how well the built-in k-fold cross validation did. A default of 5 k-folds was entered across all three parts of project

5 for consistency. All results are in decimal, meaning an accuracy of 1.0 is 100% accurate. Accuracy is calculated by simply taking the correct estimations divided by the total estimations.

First are the results of part 1. Part 1 only requires a linear hyperplane as there are only two classifiers (good or bad, or 0 and 1), a binary classification problem.

```
best_coarse_parameters =  {'C': 10, 'gamma': 0.1}
best_fine_parameters =  {'C': 1, 'gamma': 0.05}

Accuracy =  0.9428571428571428
```

Figure 1. Results for Part 1, accuracy after finding best parameters with *GridSearchCV*.

The kernel used in Figure 1's results is "rbf," or an exponential support vector classifier, thus it maybe failing to generalize well with the linear classification problem it has been given. However, in Figure 2 below, the results are much better, likely due to it being a multi-class classifier problem.

```
best_coarse_parameters =  {'C': 10, 'gamma': 0.1}
best_fine_parameters =  {'C': 6, 'gamma': 0.14}

Accuracy =  0.9849246231155779
```

Figure 2. Results for Part 2, accuracy after finding best parameters with *GridSearchCV*.

The results in Figure 2 are very good at a 98% accuracy on its estimations for the multiclass classifier problem. This result is expected as the chosen kernel for SVC is exponential and thus should perform better on multiclass classifier problems.

```
best_coarse_parameters =  {'C': 10, 'gamma': 0.1}
best_fine_parameters =  {'C': 5, 'gamma': 0.15000000000000002}

Accuracy =  0.811
```

Figure 3. Results for Part 3, accuracy after finding best parameters with *GridSearchCV*.

Results for Part 3 are the lowest of the three parts as shown in Figure 3. This is possibly due to the pre-partitioned training and testing dataset files provided. The extensive features (36 features in part 3) may have made it more difficult to converge on a good generalizer for the data and thus a poor estimation accuracy is found. While part 3 has less classifiers in comparison to part 2, the number of features it has may be working to the estimator's detriment.

Discussion

This project was a very good learning experience with sklearn (scikit-learn), and it showed a more realistic usage of machine learning by using libraries that industry and hobbyists might use to estimate outcomes for various datasets. Results in part 1 were a little less than I was hoping for at about 94% accurate, but this is likely due to the chosen kernel as an exponential algorithm rather than a linear option. Results in part 3 are the lowest of the three parts at 81% estimation accuracy. My first assumption was that part 3's results may be as low as they are due to the pre-partitioned training and testing datasets, but it is more likely the number of features given that is causing this (36 total features). A way to potentially improve performance of the estimator in part 3 might be to perform a dimensionality reduction, or simply remove some of the features which may be unnecessary to affectively generalize the data.

Sources

Alpaydin, Ethem. *Introduction to Machine Learning*. third ed., MIT Press, 2014.

"Confusion Matrix." *Wikipedia*, Wikimedia Foundation, 22 Oct. 2019, en.wikipedia.org/wiki/Confusion_matrix. Accessed 4 Nov. 2019. https://en.wikipedia.org/wiki/Confusion_matrix.

Patel, Savan. "Chapter 2 : SVM (Support Vector Machine) — Theory." *Medium*, Machine Learning 101, 3 May 2017, medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72. Accessed 7 Dec. 2019. https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72.

"RBF SVM Parameters — Scikit-Learn 0.22 Documentation." *Scikit-Learn.Org*, 2019, scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html#sphx-glr-auto-examples-svm-plot-rbf-parameters-py. Accessed 7 Dec. 2019. https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html#sphx-glr-auto-examples-svm-plot-rbf-parameters-py.

"Sklearn.Model_selection.GridSearchCV — Scikit-Learn 0.22 Documentation." *Scikit-Learn.Org*, 2019, scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html#sklearn.model_selection.GridSearchCV. Accessed 7 Dec. 2019. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html#sklearn.model_selection.GridSearchCV.

"Sklearn.Preprocessing.StandardScaler — Scikit-Learn 0.21.2 Documentation." *Scikit-Learn.Org*, 2019, scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html. Accessed 7 Dec. 2019. https://scikitlearn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html.

"Sklearn.Svm.SVC — Scikit-Learn 0.22 Documentation." *Scikit-Learn.Org*, 2019, scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC. Accessed 7 Dec. 2019. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC.

"Support-Vector Machine." *Wikipedia*, Wikimedia Foundation, 14 May 2019.en.wikipedia.org/wiki/Support-vector_machine. Accessed 15 May 2019. https://en.wikipedia.org/wiki/Support-vector_machine.

"1.4. Support Vector Machines — Scikit-Learn 0.22 Documentation." *Scikit-Learn.Org*, 2019, scikit-learn.org/stable/modules/svm.html#svm-classification. Accessed 7 Dec. 2019. https://scikit-learn.org/stable/modules/svm.html#svm-classification.

"3.2. Tuning the Hyper-Parameters of an Estimator — Scikit-Learn 0.22 Documentation." *Scikit-Learn.Org*, 2012, scikit-learn.org/stable/modules/grid_search.html. Accessed 7 Dec. 2019. https://scikit-learn.org/stable/modules/grid_search.html.