Visualizing an SVM Classifier

Alexandre Bykov and Stanley Wang

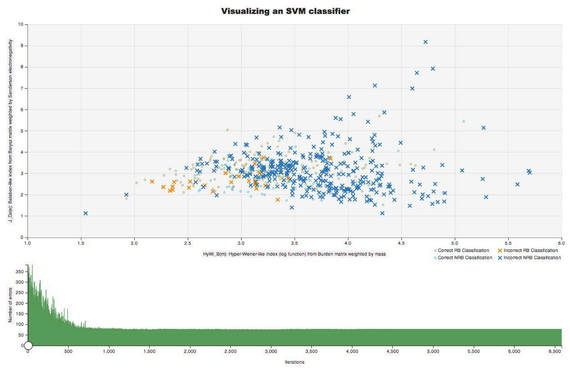
Problem and Motivation:

A key issue in any classification task is understanding the details of the machine learning algorithm's execution. A typical setup involves choosing an algorithm for the given problem, running that algorithm and then analyzing its performance through final measures, such as classification error. The issue with this is that there is very little intuition on why the classifier actually performs the way that it does. Most machine learning toolkits do not give the researcher useful information about how the classifier evolved over the runtime of the algorithm. In this project we aim to bridge that gap by visualizing the performance of a classifier over its execution.

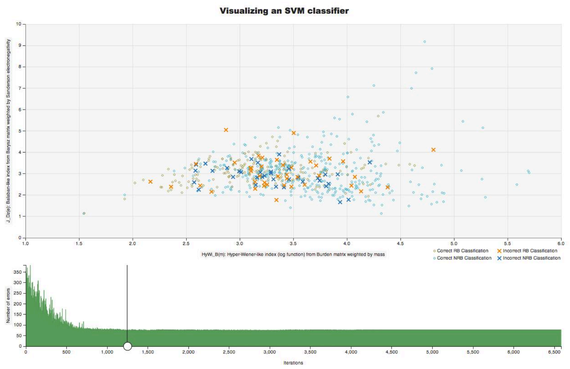
Our Approach:

To solve this issue we constructed a web application, using D3, that visualizes the behavior of an SVM (Support Vector Machine) classification algorithm over its running time. At each iteration we plot the entire dataset and mark which points were classified correctly or incorrectly with the classifier at its current state. The user is able to select which dimensions are being plotted to look at the data from different perspectives. The user is also able to interact with the data points to better understand what is causing data points to be misclassified. Below the plot we implemented a convenient slider which allows the user to quickly scroll through iterations of the algorithm. Overlaid on top of the slider are summary statistics per iteration to give the user an overview of the performance of the algorithm at each iteration. These summary statistics allow the user to quickly see which iterations of the algorithm were most important to the final outcome. The user can then scroll to these iterations for a closer view as to exactly what happened to the classifier.

UI Overview



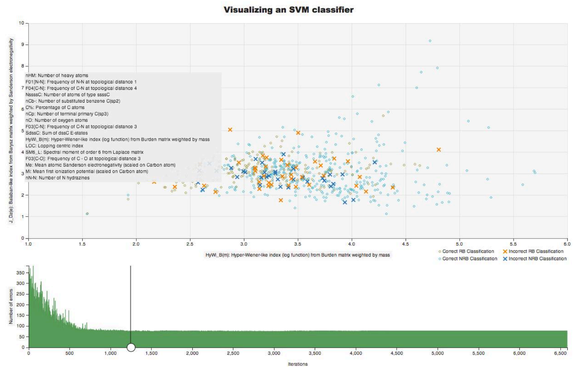
The above screenshot illustrates the user interface of the application. The majority of the interface is a scatterplot of every point projected onto two dimensions of the dataset. A shape encoding is used to demonstrate classifier performance – small circles are used to signify that the SVM model made a correct prediction and crosses stand for incorrect predictions. The color encodes the correct class for each data point. A compound color scheme was used to help the viewer quickly distinguish between correct and incorrect predictions within a class. A legend at the bottom right summarizes these encodings. Below the plot is the slider that allows the user to move between different iterations of the SVM algorithm. Overlaid on top of the slider is a graph of the number of errors per iteration. This gives the user a quick summary of the algorithm's performance over time.



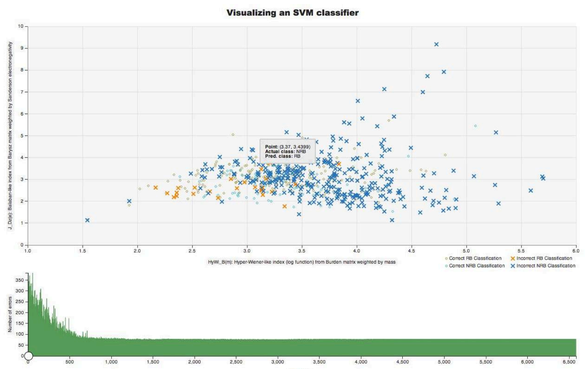
As the user drags the slider, the plot updates the predictions. The above slider position corresponds to much fewer classification errors, and the plot reflects that by showing much more circles and much fewer crosses. The class distribution of errors is easy to see.

Interaction

A key feature of our design was to make sure that the user had many ways to interact with the data so that they can clearly understand where classification mistakes are made. One way we achieved that was to allow the user to select which dimensions are being plotted.



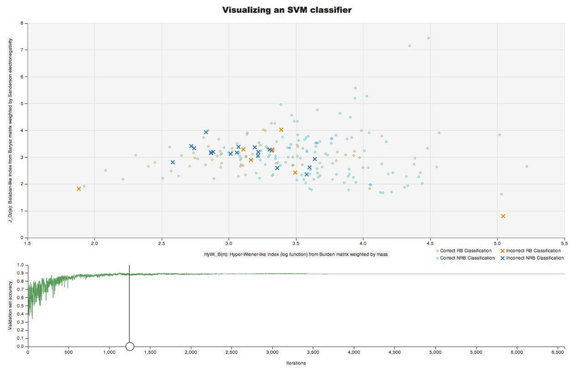
As can be seen above, clicking on a dimension label opens a menu that lists all of the possible dimensions. This menu is fully scrollable to allow for easy scalability to any number of dimensions. While not shown on the above screenshot, clicking on a new dimension will cause the scatterplot to be redrawn and rescaled to show all of the data optimally in the new view.



Another interaction techniques we utilized is illustrated above. When a user moves their mouse over a misclassified point, the exact values of the point are shown in a tooltip. We decided to only show the values of the point in the current dimensions to avoid confusion. Also, the correct and classified classes are displayed to further highlight the error.

Per-Iteration Measures

Aside from viewing the actual data points we felt that it would be important to provide summary statistics of the classifier's performance at each iteration. The graph plotted on top of the slider is what we used to illustrate these measures. The previous screenshots all showed a graph of the number of errors per iteration.



Illustrated above is a graph of the validation dataset accuracy per iteration. Classification accuracy is a common measure used to evaluate how well an algorithm works. An interesting point is that this accuracy is calculated on the validation set and thus the validation set is plotted on the scatterplot. We allow the user to also view the training (default) and test sets to better fit with the traditional setup of a machine learning experiment.

Future Work:

To make the application even more useful, we would like to integrate it with an existing SVM toolbox. That way we could allow the classifier to be visualized live, while it is training. This would be extremely helpful for models that have very long training times, as the user can track the performance of their models and iterate much more quickly. Additionally, we would like to implement support for multi-class classification and non-linear decision functions, as well as other classification algorithms.