Team: Machine Partners

By Thang Le, Tri Pham, Apurva Rai, Warren Wang

Project Inspiration

- Our team really enjoyed learning about the Support Vector Machine (SVM) section of the class as it represents an efficient approach to classification and is one of the most powerful and widely used algorithm today.
- Unfortunately, we did not have a chance to implement SVM in our projects throughout the semester. Therefore, we decided to use it on the final project.
- We started by looking more into research and optimizations done relating to SVM.

Project Overview

- For this project we implemented a Machine Learning algorithm that was developed by John Platt, the former Managing Director of Microsoft Research, in his research paper Fast Training of Support Vector Machine through Sequential Minimal Optimization (SMO).
- We then tested our implementation of the algorithm by using it to predict outputs for a few classification problems.
- In order to test the improved speed and accuracy of our optimized algorithm. We also measured the speed and accuracy of a standard SVM imported through the scikit-learn library and compared them side by side.

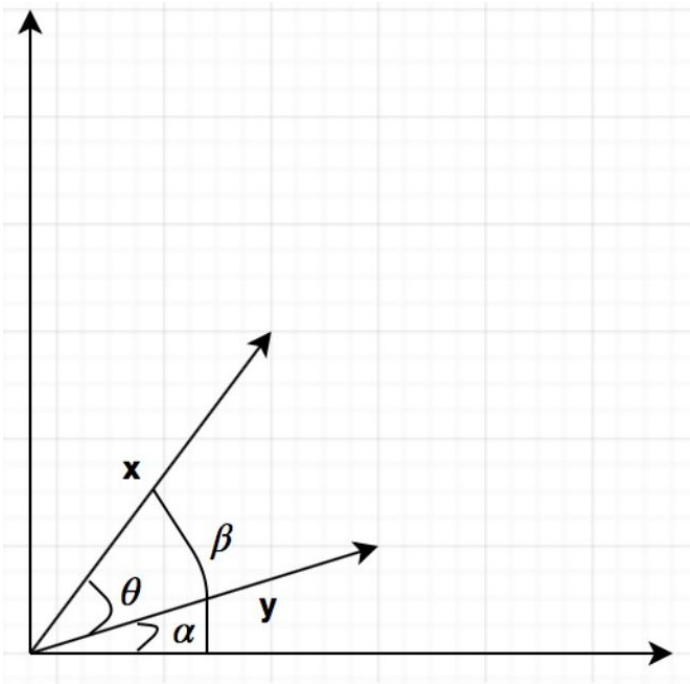
Some background

- SVM is a powerful out-of-the-box supervised machine learning algorithms.
- SVM need not be tweaked very much for good results unlike other ML algorithms like Neural Networks (but it will always serve to help performance if you do)
- We will delve into some of the math behind SVMs like kernels, optimal hyperplanes and handling of non-linearly separable data.

Linear Algebra bases

- State Vector Machines deal with - vectors.
- SVMs use dot products.

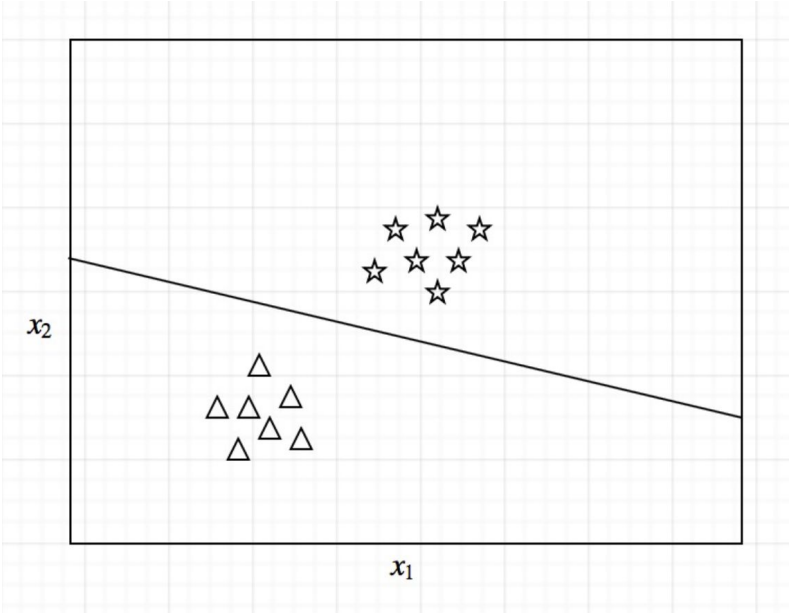
$$x \cdot y = ||x||\ ||y||\ cos(\theta)$$



$$x \cdot y = \sum_{i=1}^{n} x_i y_i$$

Linear separability

- Linear separability is an important part of SVM.
- In 3-dimensions the separation is called a plane and in higher dimensions it is generally called a hyperplane.



SVM Optimization Problem

- SVM finds optimal hyperplane which could best separate the data.
- Different measures of comparing hyperplanes.
- Using softer margins.

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i \cdot x_j$$

$$subject\ to\ 0 \leq \alpha_i \leq C, i = 1...m, \sum_{i=1}^{m} a_i y_i = 0$$

Kernel tricks

- Soft margin SVM can handle non-linearly separable data caused by noisy data.
- If data is characteristically non-linearly separable we can use the kernel trick.
- Some popular kernels are linear kernel, polynomial kernel and RBF kernel.

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i \cdot x_j)$$

$$subject\ to\ \alpha_i \geq 0, i = 1...m, \sum_{i=1}^{m} a_i y_i = 0$$

$$K(x_i, x_j) = exp(-\gamma ||x_i - x_j||^2)$$

$$K(x_i, x_j) = (x_i \cdot x_j + c)^d$$

What's Sequential Minimal Optimization (SMO)?

- SMO algorithms are used for solving the quadratic programming subsection problems that arise during the training of SVM.

- Previously available methods for SVM training were much more complex and expensive. The worst case performance for SMO is merely $O(n^3)$. (https://en.wikipedia.org/wiki/Sequential_minimal_optimization)

Implementing SVM with SMO

- SMO works by breaking down the SVM optimization problem into many smaller optimization problems that are more easily solvable. The algorithm works like this:
- Two multiplier values (αi and αj) are selected out and their values are optimized while holding all other α values constant.
- Once these two are optimized, another two are chosen and optimized over.
- Choosing and optimizing repeats until the convergence, which is determined based on the problem constraints. Heuristics are used to select the two α values to optimize over, helping to speed up convergence. The heuristics are based on the model's parameters that are stored while training the model.
- More information about SMO can be found here: http://cs229.stanford.edu/materials/smo.pdf

Dataset used & Classification Goal #1

- For the first dataset, we used the heart dataset containing 13 features of patient data including age, sex, blood pressure, cholestoral level, etc, and a goal field indicating whether if the patient has a presence of heart disease.
- Dataset contains 270 samples.
- The classification goal is to correctly predict whether a patient has a heart disease.

Dataset used & Classification Goal #2

- For the second dataset, we used the Diagnostic Brain Cancer dataset containing 30 different features computed per cell nucleus including informations such as radius, texture, perimeter and etc.
- Dataset contains 569 samples
- The classification goal is to correctly predict whether a patient has breast cancer

Running and Training SVM

SVM params for Heart Dataset

```
Regularization: 10
Max iterations: 100
Learning rate: 0.01
Margin of tolerance : 1e-05
kernel_type: linear
```

SVM params for Breast Cancer Dataset

```
Regularization: 5
Max iterations: 100
Learning rate: 0.001
Margin of tolerance : 0.0001
kernel_type: linear
```

We trained the SVM with 80% of the sample size and tested it with the remaining 20% for both datasets.

Comparison and Results

Results for Heart Dataset

Results from our implementation

```
Train time is 0.9366650581359863
Train accuracy is 0.8703703703703703
Test accuracy is 0.7777777777777778
```

Results from scikit-learn

```
Train time from sklearn is 1.2942283153533936
Train accuracy with sklearn's svm is 0.8796296296296297
Test accuracy with sklearn's svm is 0.7777777777777778
```

Results for Breast Cancer Dataset

Results from our implementation

```
Train time is 1.212848424911499
Train accuracy is 0.9384615384615385
Test accuracy is 0.9736842105263158
```

Results from scikit-learn

```
Train time from sklearn is 5.4254724979400635
Train accuracy with sklearn's svm is 0.9714285714285714
Test accuracy with sklearn's svm is 0.9649122807017544
```

Conclusion & Going Forward

- The implementation of SMO to train SVM works relatively well comparing to sklearn SVC.
- Going forward, we will be looking into expanding classifying multi classes as well as adding regression.
- We will also be looking at testing the model with different kernel schemes (polynomial, radial basis, sigmoid, etc.). They can be found here: https://ai6034.mit.edu/wiki/images/SVM_and_Boosting.pdf

Thank You!

Any Questions?