Machine Learning Homework 4
Brian McKean
CSCI 5622 Fall 2017
Due Time Nov 10, 2017

1.1 Programming questions (20 pts)
Finish svm.py.

1. Given a weight vector, implement the find support function that returns the indices of the support vectors.
2. Given a weight vector, implement the find slack function that returns the indices of the vectors with nonzero slack.
3. Given the alpha dual vector, implement the weight vector function that returns the corresponding weight vector.

See code.

1.2 Analysis (30 pts)
Use svm fours nines.py to help answer the analysis questions. Please do NOT submit svm fours nines.py to Moodle. This file is to just help read in data and run the GridSearch.

1. Use the Sklearn implementation of support vector machines to train a classifier to distinguish 4's from 9's (using the MNIST data from the KNN homework).
2. Experiment with linear, polynomial, and RBF kernels. In each case, perform a GridSearch to help determine optimal hyperparameters for the given model (e.g. C for linear kernel, C and p for polynomial kernel, and C and γ for RBF). Comment on the experiments you ran and optimal hyperparameters you found. Hint: http://scikit-learn.org/stable/modules/grid search.html

Model	Best Accuracy	Parameters at best accuracy
Linear	0.964	C=1
polynomial	0.991	Degree=2, C=1000
RBF	0.991	C=10, 100, or 1000 and gamma = .01

I ran experiments on gridsearch and five fold cross validation:

tuned_parameters =
[{'kernel': ['rbf'], 'gamma': [0.1, 0.01, 1e-3, 1e-4],'C': [1, 10, 100, 1000]},
{'kernel': ['linear'], 'C': [1, 10, 100, 1000]},
{'kernel': ['poly'], 'degree': [2, 3, 4], 'C': [1, 10, 100, 1000, 5000, 1000]}],

3. Comment on classification performance for each model for optimal parameters by either testing on a hold-out set or performing cross-validation

I used 5 fold cross-validation across a variety of parameters for each of the kernels.
Best parameters set found on development set:
{'C': 1000, 'degree': 2, 'kernel': 'poly'}

Accuracy scores on development set:
0.976 (+/-0.005) for {'C': 1, 'gamma': 0.1, 'kernel': 'rbf'}
0.985 (+/-0.007) for {'C': 1, 'gamma': 0.01, 'kernel': 'rbf'}
0.962 (+/-0.008) for {'C': 1, 'gamma': 0.001, 'kernel': 'rbf'}

```
0.937 (+/-0.006) for {'C': 1, 'gamma': 0.0001, 'kernel': 'rbf'}
0.978 (+/-0.004) for {'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}
0.991 (+/-0.006) for {'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}
0.973 (+/-0.007) for {'C': 10, 'gamma': 0.001, 'kernel': 'rbf'}
0.960 (+/-0.007) for {'C': 10, 'gamma': 0.0001, 'kernel': 'rbf'}
0.978 (+/-0.004) for {'C': 100, 'gamma': 0.1, 'kernel': 'rbf'}
0.991 (+/-0.006) for {'C': 100, 'gamma': 0.01, 'kernel': 'rbf'}
0.982 (+/-0.007) for {'C': 100, 'gamma': 0.001, 'kernel': 'rbf'}
0.969 (+/-0.007) for {'C': 100, 'gamma': 0.0001, 'kernel': 'rbf'}
0.978 (+/-0.004) for {'C': 1000, 'gamma': 0.1, 'kernel': 'rbf'}
0.991 (+/-0.006) for {'C': 1000, 'gamma': 0.01, 'kernel': 'rbf'}
0.986 (+/-0.008) for {'C': 1000, 'gamma': 0.001, 'kernel': 'rbf'}
0.971 (+/-0.007) for {'C': 1000, 'gamma': 0.0001, 'kernel': 'rbf'}
0.964 (+/-0.009) for {'C': 1, 'kernel': 'linear'}
0.959 (+/-0.009) for {'C': 10, 'kernel': 'linear'}
0.953 (+/-0.007) for {'C': 100, 'kernel': 'linear'}
0.950 (+/-0.009) for {'C': 1000, 'kernel': 'linear'}
0.948 (+/-0.004) for {'C': 5000, 'kernel': 'linear'}
0.947 (+/-0.003) for {'C': 10000, 'kernel': 'linear'}
0.957 (+/-0.008) for {'C': 1, 'degree': 1, 'kernel': 'poly'}
0.948 (+/-0.005) for {'C': 1, 'degree': 2, 'kernel': 'poly'}
0.526 (+/-0.007) for {'C': 1, 'degree': 3, 'kernel': 'poly'}
0.507 (+/-0.000) for {'C': 1, 'degree': 4, 'kernel': 'poly'}
0.967 (+/-0.006) for {'C': 10, 'degree': 1, 'kernel': 'poly'}
0.974 (+/-0.008) for {'C': 10, 'degree': 2, 'kernel': 'poly'}
0.946 (+/-0.012) for {'C': 10, 'degree': 3, 'kernel': 'poly'}
0.548 (+/-0.012) for {'C': 10, 'degree': 4, 'kernel': 'poly'}
0.968 (+/-0.007) for {'C': 100, 'degree': 1, 'kernel': 'poly'}
0.985 (+/-0.006) for {'C': 100, 'degree': 2, 'kernel': 'poly'}
0.976 (+/-0.007) for {'C': 100, 'degree': 3, 'kernel': 'poly'}
0.928 (+/-0.026) for {'C': 100, 'degree': 4, 'kernel': 'poly'}
0.963 (+/-0.007) for {'C': 1000, 'degree': 1, 'kernel': 'poly'}
0.991 (+/-0.006) for {'C': 1000, 'degree': 2, 'kernel': 'poly'}
0.987 (+/-0.005) for {'C': 1000, 'degree': 3, 'kernel': 'poly'}
0.967 (+/-0.012) for {'C': 1000, 'degree': 4, 'kernel': 'poly'}
0.960 (+/-0.009) for {'C': 5000, 'degree': 1, 'kernel': 'poly'}
0.991 (+/-0.005) for {'C': 5000, 'degree': 2, 'kernel': 'poly'}
0.990 (+/-0.008) for {'C': 5000, 'degree': 3, 'kernel': 'poly'}
0.984 (+/-0.005) for {'C': 5000, 'degree': 4, 'kernel': 'poly'}
0.963 (+/-0.007) for {'C': 1000, 'degree': 1, 'kernel': 'poly'}
0.991 (+/-0.006) for {'C': 1000, 'degree': 2, 'kernel': 'poly'}
0.987 (+/-0.005) for {'C': 1000, 'degree': 3, 'kernel': 'poly'}
0.967 (+/-0.012) for {'C': 1000, 'degree': 4, 'kernel': 'poly'}
```

My best score was the degree '2' polynomial with C=1000. -> 0.991
Polynomial accuracy in general decreased with degree greater than 2.

Rbf kernel best was also 0.991 with C at 10, 100, 1000 and gamma = .01

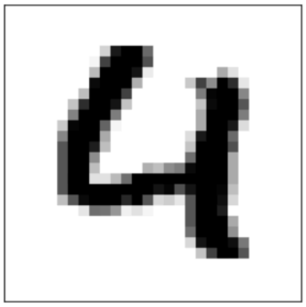
The best score for the linear kernel was 0.964 with C=1

The rbf kernel performed well across all paramaters. The linear model did not reach the percomand levels of rbf and polynomial. Polynomial of degree 2 was better than higher degree. Polyinomial wth C=1 was the worse performing. This is probably due to overfitting.

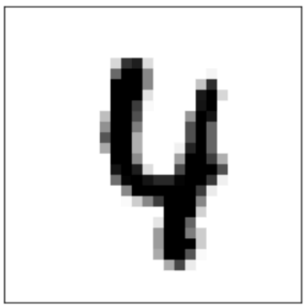
4. Give examples (in picture form) of support vectors from each class when using a polynomial kernel.

Here are example support vectors for '4' and '9'

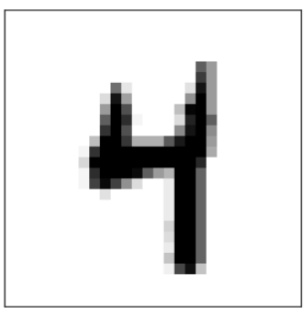
Example of -1 SV, idx = 0



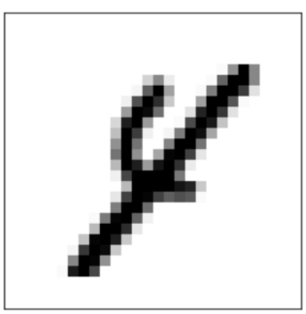
Example of -1 SV, idx = 1



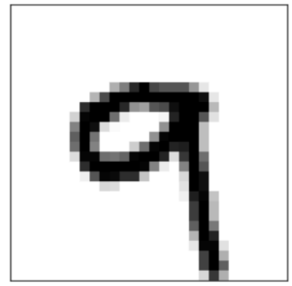
Example of -1 SV, idx = 2



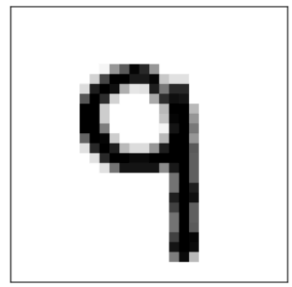
Example of -1 SV, idx = 3



Example of +1 SV, idx = 3250

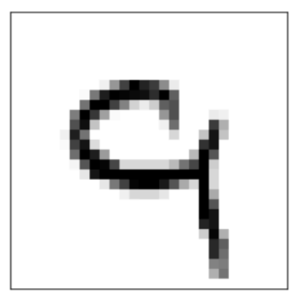


Example of +1 SV, idx = 3251



Example of +1 SV, idx = 3252

Example of +1 SV, idx = 3253



3. Learnability (25 pts)

Consider the class C of concepts defined by triangles with distinct vertices of the form (i, j) where i and j are integers in the interval [0, 99]. A concept c labels points on the interior and boundary of a triangle as positive and points on the exterior of the triangle as negative.

Give a bound on the number of randomly drawn training examples sufficient to assure that for any target class c in C, any consistent learner will, with probability 95%, output a hypothesis with error at most 0.15.

Note: To make life easier, we'll allow degenerate triangles in C. That is, triangles where the vertices are collinear. The following image depicts an example of a degenerate and non-degenerate triangle.

ANSWER:

From Lecture 14 ...

h(x) = 1 if point is inside the triangle or on an edge
h(x) = -1 if point is outside the triangle

For a finite conistent Hypothesis Class

$$m >= \frac{1}{\epsilon}(ln|H| + ln(\frac{1}{\delta}))$$

We want 95 percent probablility of being correct so

$$\delta = 0.05$$

The error allowed is 0.15.

We need |H|

There points are from 0 to 99 on the x and the y axis.
There are 100x100 = 10000 Points To pick a triangle you need thre points so there are 10,000 x 9,999 x 9998 = possible triangles ln (10,000 x 9,999 x 9998) = 66.2

$$m >= \frac{1}{0.15}(66.2 + ln(1/0.05)) = 204.1$$

So m is at least 205.

4. VC Dimension (25 pts)

This questions concerns feature vectors in two-dimensional space. Consider the class of hypotheses defined by circles centered at the origin. A hypothesis h in this class can either classify points as positive if they lie on the boundary or interior of the circle, or can classify points as positive if they lie on the boundary or exterior of the circle. State and prove (rigorously) the VC dimension of this

ANSWER:

The VC dimension is at least 2. Given two points at different distances from the origin r1, r2 a circle at the origin of radius r can shatter the if r1 < r < r2

If three points are selected then r1 <= r2 <= r3. There is no r that can be selected that allows r1 and r3 to be classified as positive and r2 to be classified as negative. The converse is also true, there is no r that allows r1 and r3 to be classified as negative and r2 to be classified as positive.

Therefore, the VC dimension is 2.

EXTRA CREDIT (10 pts): Consider the class of hypotheses defined by circles anywhere in 2D space. A hypothesis h in this class can either classify points as positive if they lie on the boundary or interior of the circle, or can classify points as positive if they lie on the boundary or exterior of the circle. State and prove (rigorously) the VC dimension of this family of classifiers.

ANSWER:
With two points p1, p2 a circle can be draw either around both points, either point separately or neither point. So VCd >= 2.

With three points p1,p2,p3, unless the points are collinear it will always be possible to draw a circle containing 0, 1, 2 or all three of the points. So Vcd >=3. If the points are collinear and the middle point is negative, the classifier will not work if the outer points are both positive.

With four points if you label the points farthest away from each other at positive and the other two points as negative then no circle can be drawn to correctly classifiy the points. Therfroe VCd < 4

Therefore VCd = 3