Recitation 10
Ensemble Methods, SVM, and Kernels

10-701: Introduction to Machine Learning

4/19/2024

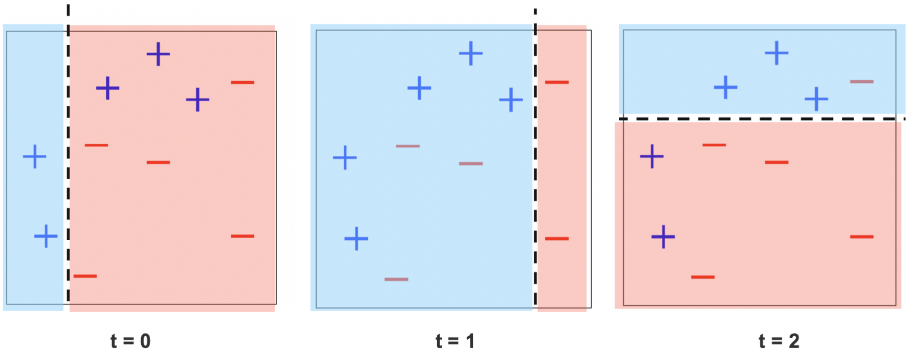
1 Ensemble Methods

The idea of ensemble methods is to build a model for prediction by combining the strengths of a group of simpler models. We'll cover two examples of ensemble methods: random forests and AdaBoost.

1.1 Adaboost

1.1.1 Practical Example

(Adapted from Eric Xing's 10701 slides) The graphs below show three iterations running Adaboost with a depth 1 decision tree. Each dashed line represents the decision boundary of h_t, and the shaded regions represent the predictons, positive (blue) or negative (red). For each iteration find the weighted training error ϵ_t and importance α_t of h_t. For $t = 0$ and $t = 1$ also find the weight normalization Z_t and record the updated weight for each point.



1. **t = 0**

$\epsilon_0 = .1 \times 3 = .3$, since we have 3 incorrectly classified points and initialize weights to $\frac{1}{n} = .1$

$$\alpha_0 = \frac{1}{2}\ln(\frac{1-.3}{.3}) = .42$$

New non-normalized weights for correct points:

$$w_{correct} = .1 \times e^{-.42} = .0657$$

New non-normalized weights for incorrect points:

$$w_{incorrect} = .1 \times e^{.42} = .1522$$

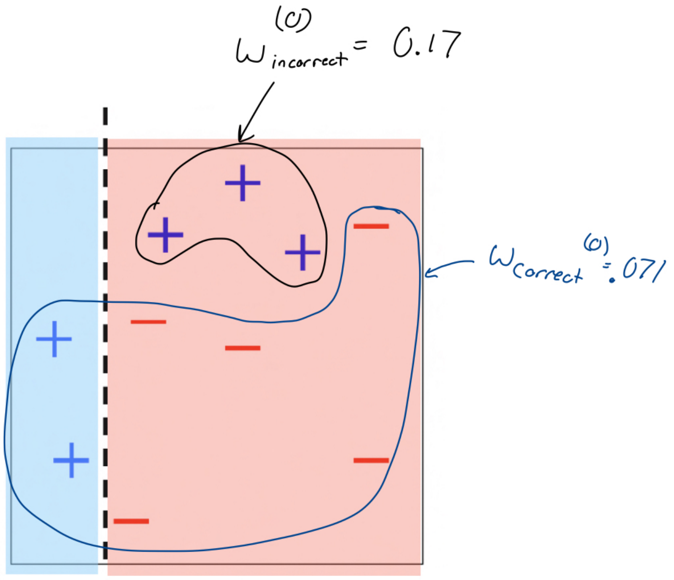
$$Z_0 = 3 * .1522 + 7 * .0657 = 0.9165$$

New normalized weights for correct points:

$$w_{correct} = \frac{.0657}{0.9165} = 0.071$$

New normalized weights for incorrect points:

$$w_{incorrect} = \frac{.1522}{0.9165} = 0.17$$



t = 0

2. **t = 1**

$\epsilon_1 = 0.071 \times 3 = 0.21$, since we have 3 incorrect points and they were all previously correct

$$\alpha_1 = \frac{1}{2}\ln(\frac{1-.21}{.21}) = .66$$

New non-normalized weights for correct points:

$$w_{correct|correct} = .071 \times e^{-.66} = 0.0367$$

$$w_{correct|incorrect} = .17 \times e^{-.66} = 0.0879$$

New non-normalized weights for incorrect points:

$$w_{incorrect|correct} = .071 \times e^{.66} = 0.1374$$

$$Z_1 = 3 * 0.1374 + 3 * 0.0879 + 4 * 0.0367 = 0.8227$$

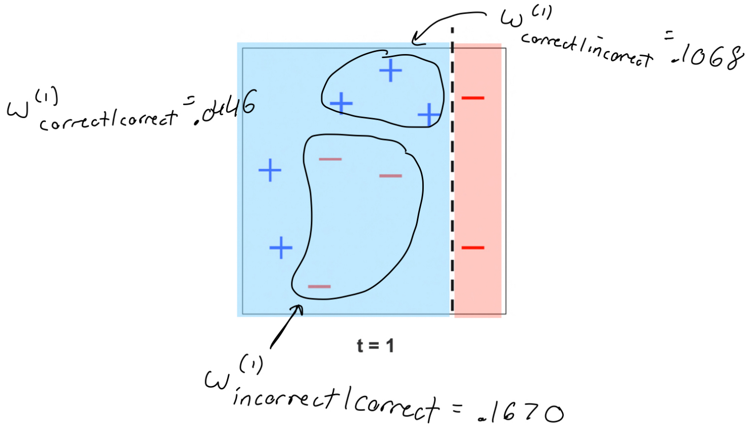
New non-normalized weights for correct points:

$$w_{correct|correct} = 0.0367/0.8227 = 0.0446$$

$$w_{correct|incorrect} = 0.0879/0.8227 = 0.1068$$

New non-normalized weights for incorrect points:

$$w_{incorrect|correct} = 0.1374/0.8227 = 0.1670$$


$\omega^{(1)}_{correct|incorrect} = .1068$
$\omega^{(1)}_{correct|correct} = .0446$
t = 1
$\omega^{(1)}_{incorrect|correct} = .1670$

3. **t = 2**

$$\epsilon_2 = 0.045 * 3 = 0.14$$

Since we have 3 incorrect points and they were all previously correct both times

$$\alpha_2 = \frac{1}{2}\ln(\frac{1-.14}{.14}) = .0.9076$$

1.2 Random Forests

1. What are some downsides of decision trees, and how can we explain this in the context of the bias-variance tradeoff?

   learned greedily, can overfit if depth isn't controlled, low bias but high variance

$$\text{Random Forests} = \text{Sample Bagging} + \text{Split-Feature Randomization}$$

2. What is **sample bagging**?

   Bagging stands for bootstrap aggregating. A bootstrapped dataset has the same number of rows as the original dataset, but its rows are drawn from the original dataset with replacement. Models are trained on each individual dataset, but since the training datasets are not as similar to each other, the learned models tend to be different and more variable as well. Aggregating refers to the model predictions being combined to form the final prediction.

3. What is **split-feature randomization**?

   In the splits for each decision tree, instead of choosing the split feature from the set of all features, we limit the feature set to a randomly chosen subset of all the features. This further reduces correlation between the learned decision trees as the "best" feature may not always be available to split on.

4. How do these techniques affect the bias and variance of an individual tree? Both

   increase bias and decrease variance by limiting the information available to train on, compared to a decision tree trained on the full original dataset.

5. How do these techniques affect the bias and variance of an ensemble of trees?

   The increase in bias carries over from the individual trees. The ensemble has lower variance because we are aggregating predictions over a set of trees that are not fully correlated. Both techniques are designed to make the trees less correlated with one another, as reducing covariance between trees reduces variance of the average over trees.

   Consider random variables X_1, X_2, each with variance σ^2. The variance of their average is given by

$$\begin{aligned}\text{Var}(\frac{1}{2}X_1 + \frac{1}{2}X_2) &= \text{Var}(\frac{1}{2}X_1) + \text{Var}(\frac{1}{2}X_2) + 2\text{Cov}(\frac{1}{2}X_1, \frac{1}{2}X_2)\\ &= \frac{1}{4}\text{Var}(X_1) + \frac{1}{4}\text{Var}(X_2) + \frac{1}{2}\text{Cov}(X_1, X_2)\\ &= \frac{1}{4}\sigma^2 + \frac{1}{4}\sigma^2 + \frac{1}{2}\sigma^2\frac{\text{Cov}(X_1, X_2)}{\sigma \cdot \sigma}\\ &= \frac{1}{4}\sigma^2 + \frac{1}{4}\sigma^2 + \frac{1}{2}\rho\sigma^2\\ &= \sigma^2\frac{1}{2}(1 + \rho)\end{aligned}$$

where ρ is the correlation between X_1, X_2.

If $\rho = 1$ (fully correlated trees), then we achieve no variance reduction: the average has variance σ^2. In general, reducing ρ makes the trees less correlated and reduces variance.

Note that our techniques are generally not extreme enough to generate anticorrelated trees, where tree predictions would oppose each other. The more anticorrelated our predictions are, the closer our model gets to always predicting 0, which does reduce variance, but at the cost of performance on our task.

6. For each data point $\mathbf{x}^{(i)}$, define $t^{(-i)}$ to be the set of decision trees that $\mathbf{x}^{(i)}$ was not used to train. Use each tree in $t^{(-i)}$ to make a prediction for $\mathbf{x}^{(i)}$, and use these predictions to make an aggregated prediction $\overline{t^{(-i)}}(\mathbf{x}^{(i)})$ (i.e. for classification take the majority vote). Then, we can define the *out-of-bag* error as follows:

$$E_{OOB} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left(\overline{t^{(-i)}}(\mathbf{x}^{(i)}) \neq y^{(i)}\right)$$

Why can we use E_{OOB} for hyperparameter optimization even though it was calculated using training points we used to learn the decision trees with?

While every point was used to train a certain set of decision trees, the calculation of E_{OOB} takes advantage of the fact that the nature of bootstrapped datasets means that there will generally be a reasonably large proportion of them that do not contain any particular point.

Therefore, for the decision trees that were trained on these datasets, the training point is equivalent to a test point as the tree has never seen it before. Since each tree is trained independently, there will never be a scenario in which a tree is both trained on a data point and also evaluated on it.

7. **Random Forest Example:** Suppose we train a random forest with two decision trees on the following dataset, using the provided bootstrap samples. Assume that for ties, we predict $Y = 1$.

All	X_0	X_1	X_2	X_3	Y
1	1	0	0	0	1
2	0	0	1	0	1
3	0	0	0	1	1
4	0	0	0	0	0
5	0	1	0	1	1

Sample 1	X_0	X_1	X_2	X_3	Y
1	1	0	0	0	1
4	0	0	0	0	0
5	0	1	0	1	1

Sample 2	X_0	X_1	X_2	X_3	Y
3	0	0	0	1	1
4	0	0	0	0	0
5	0	1	0	1	1

(a) Suppose we train our first tree on Sample 1 and the split feature randomization chooses $\{X_1, X_2\}$ for the feature candidates at the root. What feature will we split on at the root? X_1

(b) Suppose we then recurse on the left child (with feature value 0) of the root and split feature randomization chooses $\{X_0, X_2\}$ for the feature indices. What feature will we split on? X_0

(c) Suppose we train our second tree on Sample 2 and the split feature randomization chooses $\{X_2, X_3\}$ for the feature candidates at the root. What feature will we split on at the root? X_3

(d) What is the training error of the ensemble? 1/5, as only point 2 is incorrect.

Point 1: tree 1 predicts 1, tree 2 predicts 0, so prediction is 1

Point 2: tree 1 predicts 0, tree 2 predicts 0, so prediction is 0

Point 3: tree 1 predicts 0, tree 2 predicts 1, so prediction is 1

Point 4: tree 1 predicts 0, tree 2 predicts 0, so prediction is 0

Point 5: tree 1 predicts 1, tree 2 predicts 1, so prediction is 1

(e) What is the out of bag error of the ensemble? 4/5, as only point 5 is correct

Point 1: only tree 2 is involved, prediction is 0

Point 2: both trees are involved, prediction is 0

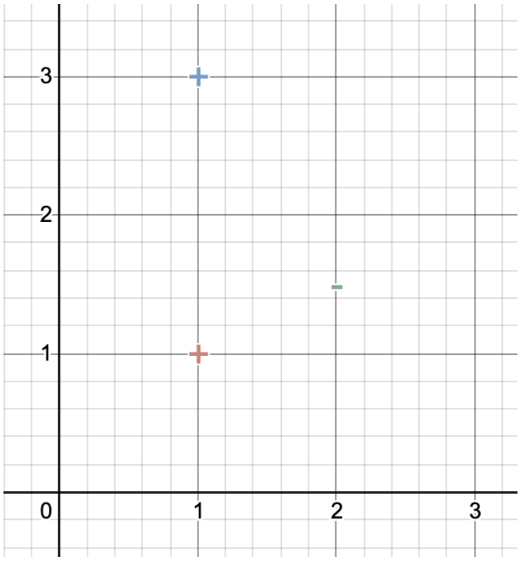
Point 3: only tree 1 is involved, prediction is 0

Points 4 and 5: majority vote over 0 trees predicts 1

2 SVM and Kernels

2.1 Questions on SVM with hard-margin

1. What is the decision boundary and the margin if we run a Hard-Margin SVM on the following set of points?



Decision Boundary: x = 1.5, Margin = 1

2. A few additional data points are added to the data set in Figures 3 (a) and 3 (b). Draw the new decision boundaries and give the margins corresponding to this boundaries. In which case does the decision boundary undergo a change and why?

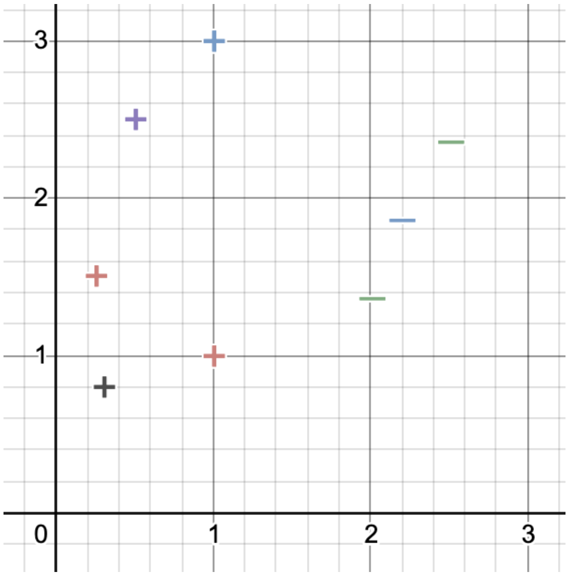


Figure 3(a)

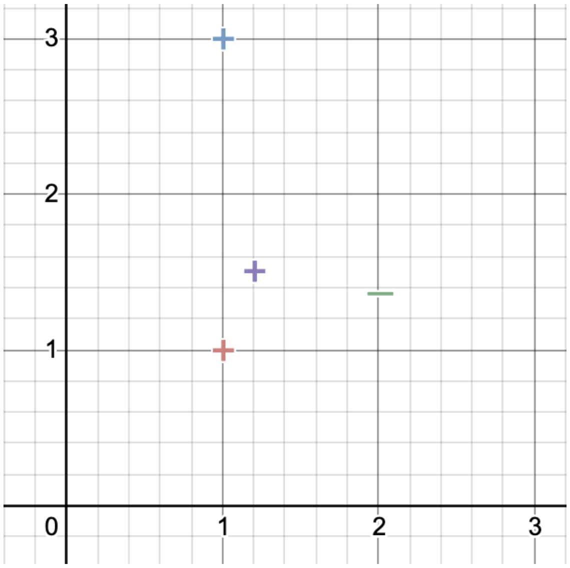


Figure 3(b)

For figure 3(a) the decision boundary remains unchanged because the support vector is the same. 3(b) there is a non-vertical decision boundary.

2.2 Soft-margin linear SVM

Given the following dataset in 1-d space (Figure 5), which consists of 4 positive data points and 3 negative data points. Suppose that we want to learn a soft-margin linear SVM for this data set. Remember that the soft-margin linear SVM can be formalized as the following constrained quadratic optimization problem. In this formulation, C is the regularization parameter, which balances the size of margin (i.e., smaller $w^T w$) vs. the violation of the margin (i.e. smaller $\sum_{i=1}^{m} \epsilon_i$).

$$\begin{aligned} &\textbf{argmin}_{\{w,b\}}\ \frac{1}{2} w^t w + C \sum_{i=1}^{m} \epsilon_i \\ &\textbf{Subject to :} y_i(w^t x_i + b) \geq 1 - \epsilon_i \\ &\qquad\qquad\qquad \epsilon_i \geq 0\ \forall i \end{aligned}$$

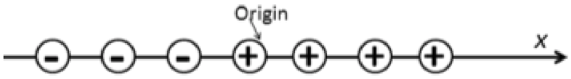


Figure 5

1. If $C = \infty$, which means that we only care the violation of the margin, how many support vectors do we have?

   2

2. If $C = 0$, which means that we only care about the size of the margin, how many support vectors do we have?

   7

2.3 Margin trivia

SVMs try to find the linear separator that maximises the *margin* between datapoints.

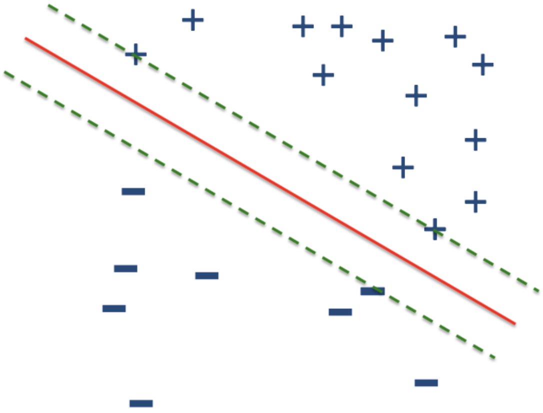
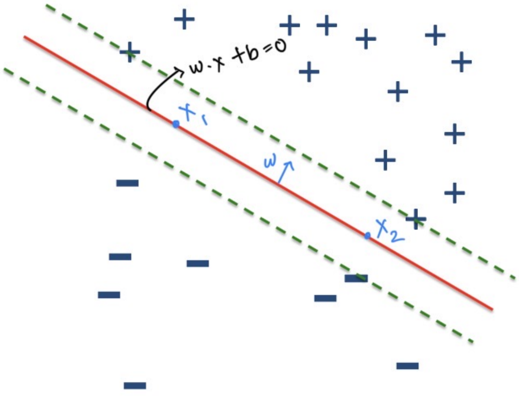


Figure 1: SVM decision boundary for some sample data

1. How can we show that w is perpendicular to the decision boundary of SVMs?



Let us take two points x_1 and x_2 on the decision boundary of the SVM. So we know that both these points will satisfy the equation $w^T x_1 + b = 0$ and $w^T x_2 + b = 0$.

Subtracting the two equations we get:

$$w^T x_1 + b - (w^T x_2 + b) = 0 \quad (1)$$

$$w^T (x_1 - x_2) = 0 \quad (2)$$

Now we know that $x_1 - x_2$ is a vector along the decision boundary and as its dot product with w is 0, we can conclude they are perpendicular.

2.4 Composite kernels

We can construct kernels by combining existing kernels in valid ways. We will verify three kernels of these properties by computing the implied feature transformations.

For any valid kernels K_1 and K_2 with implied feature transformations Φ_1 and Phi_2 and for nonnegative coefficients c_1, c_2, show that the following expressions are valid kernels:

1. $K(x, x') = c_1 K_1(x, x') + c_2 K_2(x, x')$

   We can show that K is a valid kernel by finding a transformation Φ that recovers the linear combination of kernels K_1 and K_2 on the right hand side.

   Recall that by definition, a kernel is some function K_Φ such that $K_\Phi(x, x') = \Phi(x)^\top \Phi(x')$ for all $x, x' \in \mathcal{X}$.

   We can decompose the linear combination into the inner products and then show that we can rewrite that expression as a new kernel-defining function Φ:

$$\begin{aligned} c_1 K_1(x, x') + c_2 K_2(x, x') &= c_1 \Phi_1(x)\Phi_1(x') + c_2 \Phi_2(x)\Phi_2(x') \\ &= [\sqrt{c_1}\Phi_1(x), \sqrt{c_2}\Phi_2(x)] \cdot [\sqrt{c_1}\Phi_1(x'), \sqrt{c_2}\Phi_2(x')]^\top \\ &= \Phi(x)\Phi(x') \end{aligned}$$

   where we have $\Phi(x) = [\sqrt{c_1}\Phi_1(x), \sqrt{c_2}\Phi_2(x)]$. By finding a valid Φ that can replicate the kernel transformation, we have shown that K is a valid kernel.

2. $K(x, x') = c_1 K_1(x, x') K_2(x, x')$

   As before, we will write out the inner products of the right hand side:

$$\begin{aligned} c_1 K_1(x, x') K_2(x, x') &= c_1 \Phi_1(x)\Phi_1(x')\Phi_2(x)\Phi_2(x') \\ &= (\sqrt{c_1}\Phi_1(x)\Phi_2(x))(\sqrt{c_2}\Phi_1(x')\Phi_2(x')) \\ &= \Phi(x)\Phi(x') \end{aligned}$$

   where we have $\Phi(x) = \sqrt{c_1}\Phi_1(x)\Phi_2(x)$

3. $K(x, x') = e^{K_1(x, x')}$

   We begin by writing the Taylor expansion of the right hand side and then using the identities we proved above. Recall the Taylor expansion of the exponential function:

$$e^x = \sum_{n=1}^{\infty} \frac{x^n}{n!}$$

$$e^{K_1(x,x')} = 1 + K_1(x, x') + \frac{K_1(x, x')^2}{2} + \frac{K_1(x, x')^2}{2} + ... + \frac{K_1(x, x')^n}{n!} + ...$$

Each term in the above summation is a kernel because it is a product of valid kernels (and constants), which we showed in example 2 above. Therefore, the entire summation is a kernel because a sum of valid kernels is a valid kernel, which we showed in example 1.