Practical 8 – Decision Tree, Bagging, Random Forest, and Support Vector Machines (Solutions)

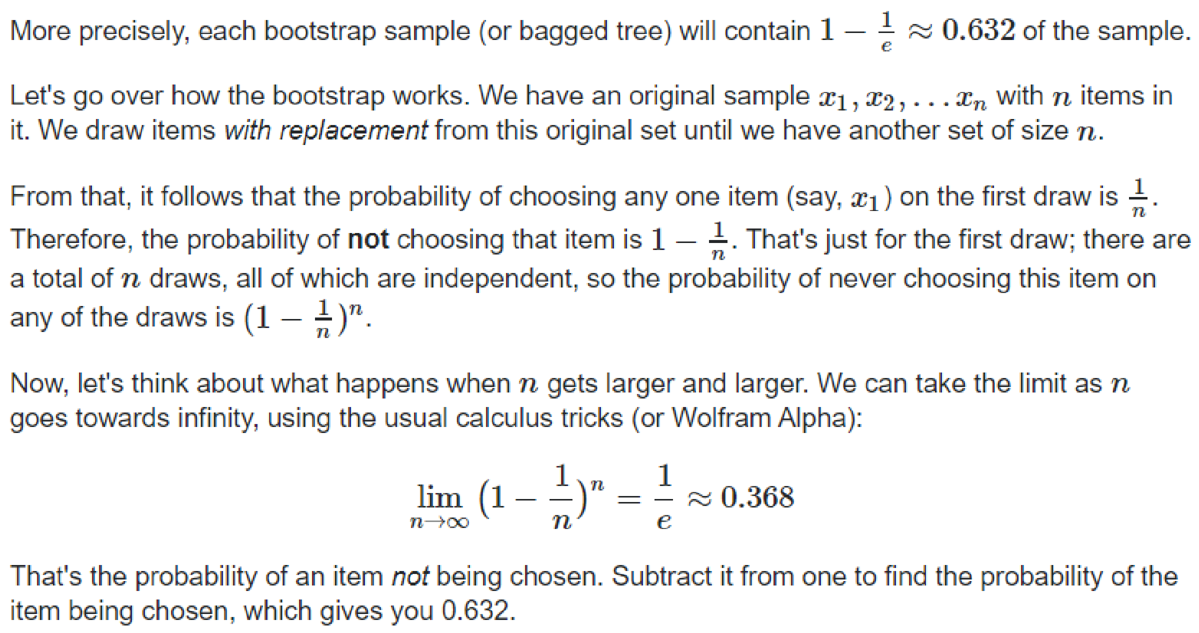
CMP9137M – MACHINE LEARNING

Overview

In this exercise, you are expected to understand the principle of support vector machines and use Python to train a SVM classifier.

Decision Tree and Random Forest

1. Why is decision tree learning a high variance and low bias algorithm?
   Solution: Decision trees makes almost no assumption about a target function, but it is highly susceptible to variance in data.
2. In bagging, roughly 1/3 of the examples do not appear in a particular bootstrap training set. Use probability theory to calculate the precise percentage.
   Solution:



More precisely, each bootstrap sample (or bagged tree) will contain $1 - \frac{1}{e} \approx 0.632$ of the sample.

Let's go over how the bootstrap works. We have an original sample $x_1, x_2, \ldots x_n$ with n items in it. We draw items *with replacement* from this original set until we have another set of size n.

From that, it follows that the probability of choosing any one item (say, x_1) on the first draw is $\frac{1}{n}$. Therefore, the probability of **not** choosing that item is $1 - \frac{1}{n}$. That's just for the first draw; there are a total of n draws, all of which are independent, so the probability of never choosing this item on any of the draws is $(1 - \frac{1}{n})^n$.

Now, let's think about what happens when n gets larger and larger. We can take the limit as n goes towards infinity, using the usual calculus tricks (or Wolfram Alpha):

$$\lim_{n\to\infty} (1 - \frac{1}{n})^n = \frac{1}{e} \approx 0.368$$

That's the probability of an item *not* being chosen. Subtract it from one to find the probability of the item being chosen, which gives you 0.632.

3. Explain why cross validation is not needed in Random Forest?
   Solution: In random forest, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error. It is estimated internally, during the run, as follows:

   Each tree is constructed using a different bootstrap sample from the original data. About one-third of the cases are left out of the bootstrap sample and not used in the construction of the kth tree.

   Put each case left out in the construction of the kth tree down the kth tree to get a classification. In this way, a test set classification is obtained for each case in about one-third of the trees. At the end of the run, take j to be the class that got most of

the votes every time case n was oob. The proportion of times that j is not equal to the true class of n averaged over all cases is the oob error estimate. This has proven to be unbiased in many tests.

Support Vector Machines

Problem 1:

You are given a data set in a CSV file with the first two columns being features and the third column being the class labels. Work on the following tasks using Python:

1) Randomly split the data to training (80%) and test (20%) sets.
2) Plot the scatter plots of the training data: positive examples in red and negative examples in blue circles.
3) Train a SVM classifier using the training data; you could use the "scikit-learn" module for Python or any other modules. You should plot the trained decision boundary. Can you point out which are the support vectors? What are the prediction results on the test data?
   Solution: see the code demo_svm.py

Problem 2 (Optional):

Assume you are given data points: $x_1 = 1, y_1 = +1; x_2 = 2, y_2 = +1; x_3 = 4, y_3 = -1, x_4 = 5, y_4 = -1; x_5 = 6, y_5 = +1$. Work on the following tasks:

1) Consider a hard margin SVM model; write down the original and dual optimization problems explicitly, i.e., inserting the data points into the optimization problems.
2) Consider a SVM model with the kernel function $K(x_i, x_j) = \left(1 + x_i^T x_j\right)^2$ in your hard margin SVM model. From a quadratic optimization solver, we have the solutions for the dual optimization problem: $\alpha_1 = 0, \alpha_2 = 2.5, \alpha_3 = 0, \alpha_4 = 7.333, \alpha_5 = 4.833$. For a new data point, the discriminant function (the classifier) is
   $\mathrm{f(x_*)} = \sum_i \alpha_i y_i \boldsymbol{K}(\boldsymbol{x_i}, \boldsymbol{x_*}) + b$

   a) Find b
   b) Write the discriminant function with those training data point explicitly
   c) Classify the new data points: $x_* = 4.5, 1.5$.

Solutions:

1) **For this particularly problem, SVM is to solve the following original optimization problem by inserting those data points into the max margin optimization problem:**

$$\boldsymbol{min_w}\ ||\boldsymbol{w}||\text{^}\boldsymbol{2}$$
$$\boldsymbol{s.t.}, \boldsymbol{w} + \boldsymbol{w_0} \geq \boldsymbol{1}$$
$$\boldsymbol{2w} + \boldsymbol{w_0} \geq \boldsymbol{1}$$
$$\boldsymbol{-1} \times (\boldsymbol{-w} + \boldsymbol{w_0}) \geq \boldsymbol{1}$$
$$\boldsymbol{-1} \times (\boldsymbol{5w} + \boldsymbol{w_0}) \geq \boldsymbol{1}$$
$$\boldsymbol{6w} + \boldsymbol{w_0} \geq \boldsymbol{1}$$

The dual formulation for this problem is:

$$max_{\alpha}\sum_{i=1}^{5}\alpha_i-\frac{1}{2}\times 2\times(\alpha_1\alpha_1+4\alpha_2\alpha_2+16\alpha_3\alpha_3+25\alpha_4\alpha_4+36\alpha_5\alpha_5+2\alpha_1\alpha_2 - 4\alpha_1\alpha_3-5\alpha_1\alpha_4+6\alpha_1\alpha_5-8\alpha_2\alpha_3-10\alpha_2\alpha_4+12\alpha_2\alpha_5+20\alpha_3\alpha_4 - 24\alpha_3\alpha_5-30\alpha_4\alpha_5)$$

$$s.t., \alpha_1+\alpha_2-\alpha_3-\alpha_4+\alpha_5=0, \alpha_i\geq 0$$

2) Insert all the training data points into the discriminant function:

$$f(x)=\sum_i \alpha_i y_i K(x_i,x)+b$$
$$=2.5\times 1\times K(x,x_2)+7.333\times(-1)\times K(x,x_4)+4.833\times 1\times K(x,x_5)+b$$
$$=2.5\times(2x+1)^2-7.333\times(5x+1)^2+4.833\times(6x+1)^2+b$$
$$=0.665x^2-5.33x+b$$

Since $\alpha_2,\alpha_4,\alpha_5$ are not zero, these training data points must lie on the margin. This means that for each of these data points, the following conditions are satisfied:

For x_2: $y_2(0.665x_2^2-5.33x_2+b)=1$; we obtain $b=9$

For x_4: $y_4(0.665x_4^2-5.33x_4+b)=1$; we obtain $b=9.03$

For x_5: $y_5(0.665x_5^2-5.33x_5+b)=1$; we obtain $b=9.04$

Therefore $b=\frac{9+9.03+9.04}{3}=9.02$.

We now can use the discriminant function to classify the new data points:

For $x_*=4.5, f(4.5)=0.665*4.5^2-5.33*4.5+9.02=-1.499$.
Therefore, this data point is classified to negative class.

For $x_*=1.5, f(1.5)=0.665*1.5^2-5.33*1.5+9.02=2.521$.
Therefore, this data point is classified to positive class.

We can see that after using the polynomial kernel function, the 1-dimensional data (Figure 1) is mapped to 2-dimensional space (Figure 2), and in this example, the map defined by the kernel function is $K: x\rightarrow(x,x^2)$. We can see that in the original space, the data is not linearly separable, but after the mapping the data is linearly separable in feature space.

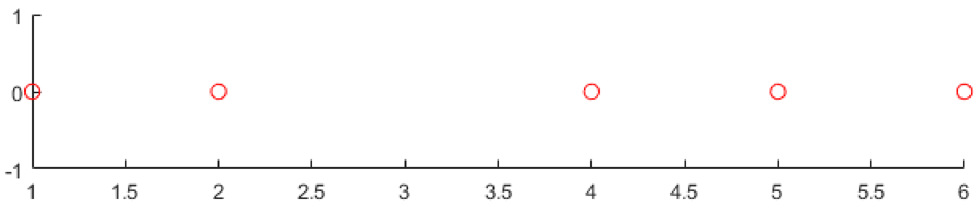


Figure 1: Training data in original space

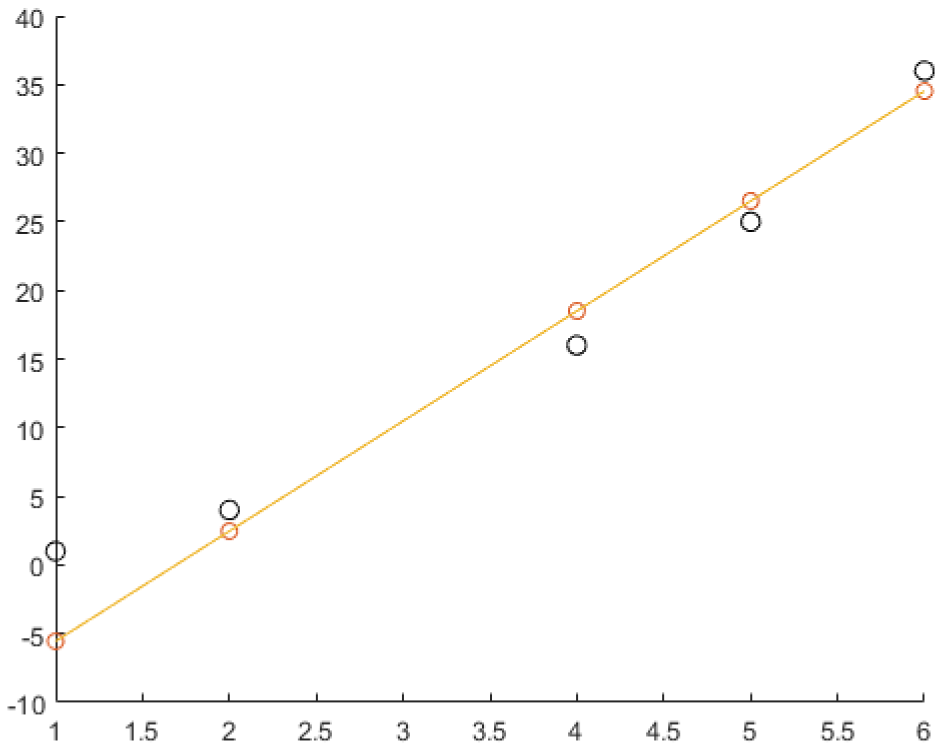


Figure 2: Training data in feature space; the yellow line is the decision boundary