Introduction to Machine Learning: Assignment #3

Submission date: 18\07\2023, 23:55.
Please submit the assignment via Moodle.
For questions regarding the assignment please contact:
Sharon Rotgaizer (srotgaizer@gmail.com) or
Gili Goldin (gili.sommer@gmail.com).

Assignment Instructions:

- Submissions in pairs ***only***.
- The code must be reasonably documented.
- Please hand in one single ***.zip*** file containing all the python files and pdf file with the ***exact*** name id1_id2.zip.
- Don't change the given scripts names!

Assignment Topics:

- PAC, VCdim
- Decision Trees
- K means Clustering
- Cross validation
- Ensemble Methods

Good Luck.

Problem 1 – PAC, VC dimension

<u>Section 1</u>

A circle (r,c) is defined by its center c and its radius r. Look at the following classifiers family:

$$\mathcal{H} = \{h_{r,c}\colon\ r \in \mathbb{R}, c \in \mathbb{R}^2\} \text{ where } h_{r,c}(x) = 1 \text{ iff x inside the circle (r,c)}$$

Find the VCdim of this class with full proof.

<u>Section 2</u>

How many samples (at least) do we need for $\mathcal{H}$ to be Agnostic PAC learnable, such that the learning fill fail in probability at most e^{-2} and the error will be at most $\varepsilon/2$?

<u>Section 3</u>

Under the realizability assumption and assuming C_1=1 what is the minimal samples number needed?

Problem 2 – Decision Trees

Finally, we got a good job in some high-tech company, and we want to buy a car 😊

In order to decide how the car will be based on features, we want to train a decision tree model.

The cars database has the following features: Buying (buying price), maint (price of the maintenance), doors, persons (capacity), lug_boot (its size), safety.

Each car is mapped into one from four classes: unacc, acc, good, vgood.

Section a

You are given a python file with DecisionTree class, most of it is already implemented. You need to complete this missing implementation of the following functions:

`calculate_entropy(self, data)` – given data, compute the entropy, where the distribution is over its labels (target class).

`calculate_information_gain(self, data, feature)` – given data and specific feature, compute the information gain given by selecting that feature.

Note: To obtain the samples with specific feature value you may use `filter_data`. For example, when calculating the gain of 'persons' and subtracting the weighted entropy for 'persons=4', use `filter_data(data,'persons', '4')` to obtain only those samples.

Now, run the SECTION A in the code. The correct output is the tree.txt file. Pay attention that the order of each level in the tree might be different.

Section b

Now instead of fitting the entire data (as done in section a), we use train and test, as should be. Complete the missing line (in the SECTION B part) to get the predictions for the **training** data. What is the accuracy? Explain.

Section c

As seen in section a, the decision tree is extremely large (depth is 6). In order to solve this, lets modify our DecisionTree:

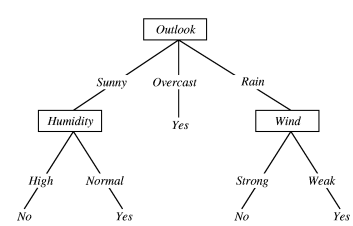
```
def __init__(self, maxdepth=np.inf):
        self.tree = {}
        self.maxdepth = maxdepth
```

Modify the rest of the code to stop growing after `maxdepth`. Hint: When reached to maxdepth, should we continue splitting? Which category will give the best accuracy?

Run SECTION C with `maxdepth=5` and `maxdepth=2` and explain the results (train vs test).

Finally – **attach** to the report drawing of the decision tree for maxdepth=2, for the **entire data** (like in section a). It should be in the format like the example:

Feel free to shrink the tree if possible.



Problem 3 – Ensemble methods, Cross Validation

Ensemble learning is a general technique to combat overfitting, by combining the predictions of many varied models into a single prediction based on majority vote or their average.

Section a

Consider a set of uncorrelated random variables $\{Y_i\}_{i=1}^{n}$ with mean μ and variance σ^2, where in our context, these Y_i's are analogous to the prediction made by classifier i.

Calculate the expectation and variance of their average ($Y = \frac{1}{n}\sum Y_i$). Explain why the answer implies that the average prediction (Y) is better than using single classifier (Y_i).

Note: of course, its relevant only to classification problem where the prediction is continuous.

Section b

Now, take the same data but implement RandomForest classifier, which uses the implementation from question 2.

Complete the missing implementation of the fit function – learn n_estimators trees where each has maxdepth and the features are randomly selected by select_features function.

Important: Even though random forest is bagging method, use the same entire data for all decision trees. It will keep the task simple.

Run the main part, which compares single decision tree with maxdepth=3 to random forest with 5 trees and maxdepth=3, on both train and test set. Explain the results using the following questions:

- Which model gives higher training error?
- Does the test error on both models differ much?
- What is the conclusion about the Random Forest model, in terms of generalization, overfitting and correlation?

Section c

We think that Sharon cheated in section b and maybe by using another number of estimators we can get better accuracy. In order to find the best number we have to use cross validation. Section c is mostly implemented in the code, your task is to complete the missing code in the KFold function.

Based on the graph, what is the best number of estimators?

Problem 4 – K means clustering

We learned in the tutorials about partitional clustering and specifically – k means algorithm. In this question we will implement it and see some nice applications.

Section a

Recall the objective function of k-means (which we want to minimize):

$$J_{SSE} = \sum_{i=1}^{k} \sum_{x \in D_i} \|x - \mu_i\|^2$$

Show that the new assignment μ_i (for each cluster) which brings the cost function to (local) minimum is the center of the ith cluster, $\mu_i = \frac{1}{n_i} \sum_{x \in D_i} x$.

Section b

In q4.py, complete the missing implementation of Kmeans. Since there are k clusters, we will label each point with {0,..,k-1}. Attach the plotting of the clustered exams data for k=2.

Section c

In the same file q4.py, use the Elbow Method (seen on the tutorials) to choose another number of centroids (the k of k-means) between 1-10. Explain why you chose it and attach both elbow plotting and the clustering with the k you chose.

Section d

Now, we will compress some image using k-means. Here, we are given image from size 400x600x3. The last parameter is the number of channels. 3 channels means that the image is colored (unlike 1 in hw2, which meant grayscale).

Our goal is to reduce the number of colors to 20 and represent (compress) the photo using those 20 colors only.

Motivation: the original image requires 400x600x3x8 bits, while the new image will require only 400x600x5 + 20x24 bits, almost 5 times smaller!

To really do this, we will take the image and treat every pixel as a data point, where each data point is in 3d space (r,g,b). Then, we cluster into 20 centroids, and we assign each pixel to a centroid. This will allow us to represent the image using only 20 colors.

Use q4_D.py, complete the missing code and attach the output.

Problem 5 – AdaBoost (10 points bonus)

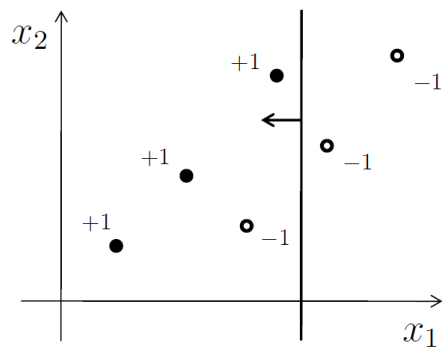
AdaBoost (Adaptive Boosting) is another approach to the ensemble method field.

It always uses the entire data and features (unlike before) and aims to create T weighted classifiers (unlike before, where each classifier had same influence). The new classification will be decided by linear combination of all the classifiers, by:

$$g(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t f_t(x)\right), \alpha_t \geq 0$$

In the assignment files, you will find the tutorial that talks about AdaBoost (which is not for the exam this year). Read & understand it.

Consider the following dataset in $\mathbb{R}^2$:



1) The first decision stump is already drawn, the arrow points in the positive direction. Calculate the classifier error (ε_1) and weight (α_1).

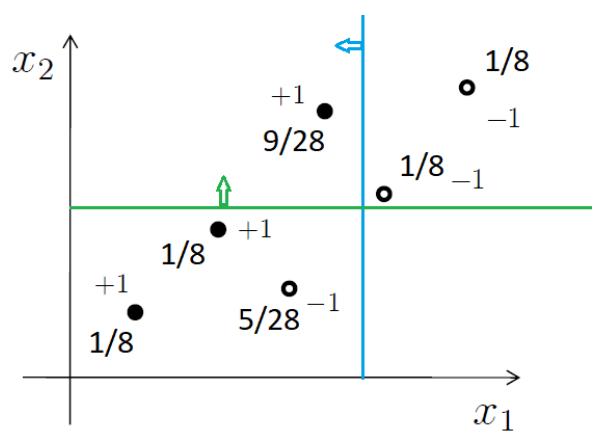
2) Calculate the new weights of the samples (and normalize them to get valid distribution).

3) Draw the second decision stump. Reminder: the decision stump (our classifiers) are parallel to x/y axis.

4) Without calculations, which classifier's weight is larger, α_1 or α_2? Explain why.

5) In the right image, there is the dataset and the weights for each point, after finding the <u>third</u> decision stump and calculating the new weights. Which of the following (green or blue) is the correct third decision stump?

   Hint: you the part of the questions in the tutorial.



6) Given $\alpha_2 = 1.1$, $\alpha_3 = 0.62$, draw the full classifier, like in slide 24.

   What is the train accuracy?

Good Luck.