CNG 409

Introduction to Machine Learning

Fall 2021-2022

Homework 3: Decision Trees and Support Vector Machines

version 1.0

Due date: 11 January 2022, 23:59

1 Introduction

In this assignment, you will have a chance to get hands-on experience with decision trees and support vector machines (SVMs). Like in the previous homework, you will write a report called **report.pdf**, the structure of which is up to you. The related material can be found in **hw3_material.zip**.

2 Decision Trees

In this section, you are using the seeds dataset. Each instance has 7 numerical features, and it can belong one of 3 categories. However, we have removed the first two features so there are 5 numerical features.

```
import numpy as np
train_set = np.load('hw3_material/dt/train_set.npy')
train_labels = np.load('hw3_material/dt/train_labels.npy')

test_set = np.load('hw3_material/dt/test_set.npy')
test_labels = np.load('hw3_material/dt/test_labels.npy')
```

You should do:

- Implement the empty functions in **dt.py**.
- Using the training set and ID3 algorithm, grow two trees with the following attribute selection strategies respectively: Information Gain and Average Gini Index. For each tree, report its test accuracy and visualization.

- You will implement binary split for the decision trees. In other words, at each node, the data is going to be split into 2 parts if a split is going to happen. Check numerical attributes section of the lecture slide DecTrees.pdf.
- You are not allowed to use libraries implementing decision trees and related functions such as entropy and gini index.
- Apply prepruning on the previous trees by using chi-squared test of independence (refer to your statistics books or this link). For each tree, report its test accuracy and visualization. You are going to stop growing the tree when you cannot reject the null hypothesis which states that there is no association between the two variables (in our case, the attribute selected by the heuristic function, i.e., information gain and average gain index, and the class variable). In this homework, you will use 90% confidence. The critical values can be obtained from a chi-squared table.
- You do not have to directly put the (four) tree diagrams in your report. You can refer to them in the report. At each node, the number of examples that belong to each class in the training set that has survived until that node should be written. You may use any drawing library and tool for this purpose such as graphviz.

3 Support Vector Machines

This section consists of four tasks, and you will use scikit-learn's SVM implementation to complete them.

3.1 Task1

In this section, you are given a 2D linearly separable dataset.

```
import numpy as np
train_set = np.load('hw3_material/svm/task1/train_set.npy')
train_lbs = np.load('hw3_material/svm/task1/train_labels.npy')
```

- Using linear kernel, train 5 SVMs with the following C values respectively: 0.01, 0.1, 1, 10, 100.
- Visualize the decision boundary of each classifier and comment on the effects of C values. You can use the code snippet in the recitation notebook for that purpose.

3.2 Task2

In this section, you are given a 2D non-linearly separable dataset.

```
import numpy as np
train_set = np.load('hw3_material/svm/task2/train_set.npy')
train_lbs = np.load('hw3_material/svm/task2/train_labels.npy')
```

- Fixing C as 1, train 4 SVMs with following kernels respectively: linear, rbf, polynomial, and sigmoid.
- Visualize the decision boundary of each classifier and comment on the effects of kernels.

3.3 Task3

In this section, you are given 32 x 32 grayscale images generated synthetically, where each image can belong to one of two classes.

```
import numpy as np
train_set = np.load('hw3_material/svm/task3/train_set.npy')
train_lbs = np.load('hw3_material/svm/task3/train_labels.npy')

test_set = np.load('hw3_material/svm/task3/test_set.npy')
test_labels = np.load('hw3_material/svm/task3/test_labels.npy')
```

- Perform a grid search on hyperparameters of SVM (required hyperparameters are kernel, C, and gamma). For each hyperparameter configuration, report the validation accuracy in the form of tables. And, for the best hyperparameter configuration, use the whole training set to train an SVM and report its test accuracy. Try at least 15 different hyperparameter configurations. Remember that you do reserve some part of your training set as validation set and not use the test set in hyperparameter optimization process.

3.4 Task4

In this section, you will use an imbalanced version of the Task3's dataset.

```
import numpy as np
train_set = np.load('hw3_material/svm/task4/train_set.npy')
train_lbs = np.load('hw3_material/svm/task4/train_labels.npy')

test_set = np.load('hw3_material/svm/task4/test_set.npy')
test_labels = np.load('hw3_material/svm/task4/test_labels.npy')
```

- Using rbf kernel and C=1, train an SVM classifier. Report its test accuracy and the confusion matrix (using the test set). Comment on whether accuracy is a good performance metric or not.
- Oversample the minority class of the training set. You can do that by simply copying the examples of the minority class so that the number of examples in both classes become somewhat close. Then, repeat the same procedure in the first item.

- Undersample the majority class of the training set. You can do that by simply deleting some of the examples in the majority class so that the number of examples in both classes become somewhat close. Then, repeat the same procedure in the first item.
- Set class_weight parameter of sklearn.svm.SVC to "balanced" which adjust the class weights inversely proportional to the class frequencies. Then, repeat the same procedure in the first item.

4 Specifications

- The code must be in Python.
- You are also provided with simple testers. Note that passing a tester does not guarantee that your code works perfectly.
- Falsifying results, changing the composition of training and test data are strictly forbidden, and you will receive 0 if this is the case. Your programs will be examined to see if you have actually reached the results and if they are working correctly.
- Using any piece of code that is not your own is strictly forbidden and constitutes as cheating. This includes friends, previous homeworks, or the internet. The violators will be punished according to the department regulations.
- Follow the course page on ODTUClass for any updates and clarifications. Please ask your questions on the discussion section of ODTUClass instead of e-mailing.
- You have a total of 3 late days for **all** homeworks. The late submission penalty will be calculated using $5d^2$, that is, 1 day late submission will cost you 5 points, 2 days will cost you 20 points, and 3 days will cost you 45 points. No late submission is accepted after reaching a total of 3 late days.

5 Submission

Submission will be done via ODTUClass. You will submit a zip file called **hw3.zip** that contains all your **source code**, **report.pdf**, and **decision tree diagrams**. Also, please provide a **README** file describing how to run your code and perform the experiments.

6 Resources

Homework3's recitation notebook can be accessed through the link.