Middle East Technical University Department of Computer Engineering

CENG 499

Introduction to Machine Learning

Fall 2020-2021

Homework 3 - Decision Trees, SVM

version 1

Due date: 08 01 2021, Friday, 23:59

1 Introduction

In this assignment you have two tasks. They are on decision trees and support vector machines (SVM). These tasks are independent from each other and have smaller related tasks in them which may have different data for you to work on. All the related files can be downloaded from `http://user.ceng.metu.edu.tr/~artun/ceng499/hw3_files.zip`.

2 Part 1: Decision Tree

2.1 Dataset

The dataset is created by modifying the Car Evaluation Dataset (`https://archive.ics.uci.edu/ml/datasets/Car+Evaluation`), so you need to use the version we provided. Its attributes (buying, maint, doors, persons, lug_boot, safety) are categorical, and it has 2 classes (unacc, acc). There are no missing values. In the given dataset file, the last value is for the class attribute. The values that attributes can take are:

- buying: vhigh, high, med, low
- maint: vhigh, high, med, low
- doors: 2, 3, 4, 5more
- persons: 2, 4, more
- lug_boot: small, med, big
- safety: low, med, high

The dataset is labeled and have separate train and test sets. It is pickled and you can load it by:

```
import pickle
with open('hw3_data/dt/data.pkl', 'rb') as f:
    train_data, test_data, attr_vals_list, attr_names = pickle.load(f)
```

2.2 Tasks

You are expected to try different attribute selection strategies using id3 algorithm to create decision trees and apply some pruning techniques. In this task, you are not allowed to use library functions that are decision tree related (including entropy etc.).

- Some empty functions are given in dt.py to guide you through the implementation. You need to fill them according to their descriptions. A mini code (dt_mini_test.py) is also given for you to test those functions whether they work as they should be. **Do not to change the signature of these functions as they will also be evaluated individually.**
- Grow your decision trees using your train data by employing ID3 algorithm. The pseudo code can be found in `https://en.wikipedia.org/wiki/ID3_algorithm`. Use **Information Gain**, **Gain Ratio**, and **Average Gini Index** to select attributes and report their test accuracies using test data. You can look up their definitions in the DecTrees.pdf in the lecture notes. Remember that we want to maximize information gain and gain ratio while we want to minimize the average gini index.
- **Gain Ratio with Chi-squared Pre-pruning.** You should 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 Gain Ratio and the class variable) using chi-squared test. The selection of the confidence value may change. In this homework, you can use, for example, 90% confidence. The critical values can be obtained from a chi-squared table.
- **Gain Ratio with Reduced error post-pruning.** After fully growing the tree using Gain Ratio, you should post-prune the tree. For this part, you should split the training set into train and validation sets to calculate the validation set accuracy to decide whether the pruning enhanced the performance. **You cannot use test set for this purpose.** You are expected to implement subtree replacement. You algorithm can be like this:

    1. while a useful replacing can be done:

        1.1. Replace a node with a leaf node (subtree replacement) whenever it increases the accuracy on the validation set and **it has no subtree with such property**.

- Draw the tree diagrams for each of the attribute selection heuristic and include them in your submission file and refer them in your reports. Don't put them directly in your reports since they won't fit in one page. In the leaves, proportion of each output class on that current node should be written. You may use any drawing library, tool, etc. for this purpose. A recommendation would be graphviz.

3 Part 2: Support Vector Machine

In this part, you are going to use svm implementation of scikit-learn `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html`.

3.1 First part

3.1.1 Dataset

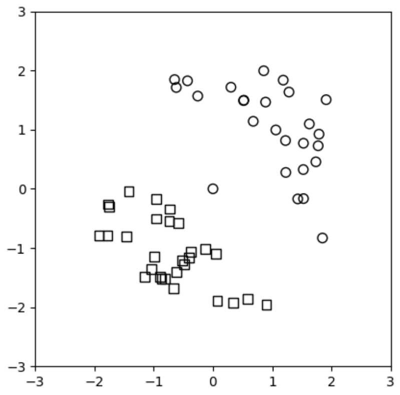

Figure 1: Linearly separable data.

The dataset is artificially created 2D and linearly separable. The dataset is labeled and you are only provided the train set because you won't need the test set in this part. It is saved as numpy arrays and you can load it by:

```
train_data = np.load('hw3_data/linsep/train_data.npy')
train_labels = np.load('hw3_data/linsep/train_labels.npy')
```

3.1.2 Tasks

You are expected to train a linear SVM (sklearn.svm.SVC) with different C values, draw the resulting classifier together with data points and comment on them.

- Train sklearn.svm.SVC with linear kernel and C values 0.01, 0.1, 1, 10, 100.
- Visualize each resultant classifier. You can use draw_svm function in draw.py provided to you. It takes an svm classifier and data points as input and draws them. The filled squares and circles are the support vectors. The dashed lines are the margins.
- Briefly comment on the effect of C on the SVM.

3.2 Second part

3.2.1 Dataset

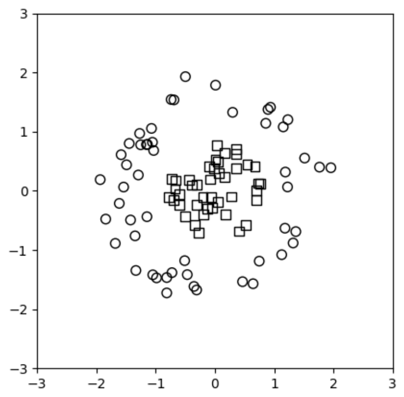

Figure 2: Not linearly separable data.

The dataset is artificially created 2D and not linearly separable. The dataset is labeled and you are only provided the train set because you won't need the test set in this part. It is saved as numpy arrays and you can load it by:

```
train_data = np.load('hw3_data/nonlinsep/train_data.npy')
train_labels = np.load('hw3_data/nonlinsep/train_labels.npy')
```

3.2.2 Tasks

You are expected to train an SVM (sklearn.svm.SVC) with different kernels, plot the resulting classifier and comment on them.

- Train sklearn.svm.SVC with linear, rbf, polynomial, and sigmoid kernels.
- Visualize each resultant classifier. You can use the draw_svm function in draw.py provided to you as it is described above.
- Briefly comment on the effect of kernels on the SVM.

3.3 Third part

3.3.1 Dataset

The dataset is created by taking the cat and dog classes and additionally shrinking the size of CIFAR10 dataset (`https://www.cs.toronto.edu/~kriz/cifar.html`). Therefore, you need to use the one we provided. The dataset is labeled and it is separated as train and test sets. It is saved as numpy arrays and you can load it by:

```
train_data = np.load('hw3_data/catdog/train_data.npy')
train_labels = np.load('hw3_data/catdog/train_labels.npy')
test_data = np.load('hw3_data/catdog/test_data.npy')
test_labels = np.load('hw3_data/catdog/test_labels.npy')
```

3.3.2 Tasks

You are expected to do grid search on hyperparameters of SVM and test your best combination on test set.

- Normalize the data. Even simple normalization like mapping it between -1 and 1 would help extremely. Don't forget to normalize also the test set in the same way.
- Do a grid search on hyperparameters kernel, C, gamma. You can also change the degree for the polynomial kernel but it is not required in this homework. Example hyperparameters are given in the report template; however, feel free to change them. Remember that you should fill the whole grid, so increase their numbers carefully. Details of grid search are given in the item below.
- To do the grid search, you are expected to do cross validation. It can be 5-fold. You can directly use sklearn.model_selection.GridSearchCV or sklearn.model_selection.cross_validate if you want to use your own loops. The validation accuracy you are going to use in the table will be the average of the 5 trained models (in 5-fold cross validation). Remember that you cannot use test set to optimize your hyperparameters.
- Report the hyperparameters of your best model and its test accuracy.

3.4 Fourth part

3.4.1 Dataset

The dataset is created by taking the cat and dog classes and shrinking the size of CIFAR10 dataset in an imbalanced way (`https://www.cs.toronto.edu/~kriz/cifar.html`). Therefore, you need to use the one we provided. The dataset is labeled and it is separated as train and test sets. It is saved as numpy arrays and you can load it by:

```
train_data = np.load('hw3_data/catdogimba/train_data.npy')
train_labels = np.load('hw3_data/catdogimba/train_labels.npy')
test_data = np.load('hw3_data/catdogimba/test_data.npy')
test_labels = np.load('hw3_data/catdogimba/test_labels.npy')
```

3.4.2 Tasks

You are expected to work on an imbalanced dataset, report your observations and apply some techniques to reduce its effect.

- Normalize the data. Even simple normalization like mapping it between -1 and 1 would help extremely. Don't forget to normalize also the test set in the same way.
- Using rbf kernel with C=1, train an SVM classifier.
- Report its test accuracy. Can accuracy be a good performance metric? Write your answer and reasoning in the report.
- Calculate the confusion matrix using test set, report it and comment on the performance using it. You can calculate other performance metrics to strengthen your arguments if you want.
- Oversample the minority class. 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. Report your test accuracy, confusion matrix and comment on them.
- Undersample the majority class. 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. Report your test accuracy, confusion matrix and comment on them.

- Set class_weight parameter of sklearn.svm.SVC to "balanced" which adjust the class weights inversely proportional to the class frequencies. Report your test accuracy, confusion matrix and comment on them.

4 Specifications

- Please provide a README file that describes how one can run your code do the experiments.
- The codes must be in Python3. You are not allowed to use the part of libraries that are related to decision trees, entropy, or information gain etc.
- 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 it is working correctly.
- The late policy given in the syllabus applies here. You have total of 6 late days for **all** your homeworks but you can spend at most 3 for a specific homework. The late submission penalty will be calculated using $5n^2$, 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.
- 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. You can use the codes given in the recitation.
- Follow the course page on ODTUCLASS for any updates and clarifications. Please ask your questions on Homework 3 Discussion section of ODTUCLASS instead of e-mailing if the question does not contain code or solution.

5 Submission

Submission will be done via ODTUCLASS. If you do not have access to ODTUCLASS send send an email to "artun@ceng.metu.edu.tr" as soon as possible. You will submit a zip file called "hw3.zip" that contains all the source code (including dt.py, draw.py, dt_mini_test.py), README file, **decision tree diagrams**, and your report in a pdf format compiled from the given latex file.