

CENG 499

Introduction to Machine Learning

Spring 2020-2021

Homework 3 - Decision Trees, SVM version 1

Due date: June 11, 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



Figure 1: Iris dataset explanatory images. Image taken from <https://rpubs.com/yao33/611068>.

2.1 Dataset

The dataset is the Iris Dataset [1]; however, you need to use the version we provided since we split it into train test sets. The task is to classify the iris flower species: Iris Setosa, Iris Versicolor, and Iris Virginica. The data contains the measurements of sepal length, sepal width, petal length, petal width in cm. Therefore, the data is four dimensional. An explanatory image can be seen at Fig. 1. The values are all numerical and there are no missing values. The dataset is labeled and have separate train and test sets. It is in the form of saved numpy array and you can load it using the following code:

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

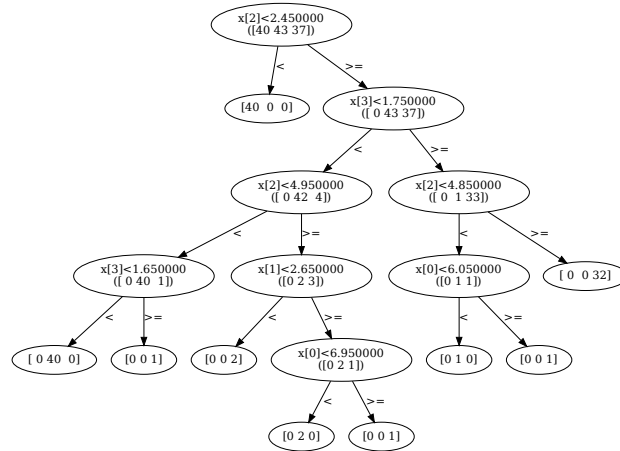


Figure 2: The expected result of the decision tree using information gain without prepruning. You can make changes in the design of the tree. However, which attribute you are using to divide the data, the splitting value and the class content of each each class should be interpretable.

2.2 Tasks

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

- You are going to implement a decision tree for the case of **binary** split which means at each node the data is going to be split into 2 parts if a split is going to happen. To decide where to split according to specific attribute, you need to sort the data according to that specific attribute and try every possible intermediate value and select the best one according to your heuristic function (information gain or average gini index). For more explanation, you can look at the Numerical Attributes part of the "DecTrees.pdf" in Lecture Notes.
- 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 or in Lecture Notes. Note that the pseudo code in wikipedia is for 2-class classification while in the homework, you are going to implement multi-class version of it. Use **Information Gain** 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 the information gain while we want to minimize the average gini index.

- **Chi-squared Pre-pruning.** You are going to implement a prepruning strategy using chi-squared test of independence. (You can use your statistics book or [this link](#) to remember the subject. We are also planning to cover it in the recitation.) 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 (information gain or average gini index) and the class variable – using chi-squared test. You are going to apply it with both information gain and average gini index. 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.

- Draw the tree diagrams for each of the attribute selection heuristic with and without pruning and include them in your submission file and refer them in your reports. You don't have to put them directly in your reports. At each node, the number of examples that belongs to each class in the training set that survived until that node should be written. The result for the information gain without pruning is given in Fig. 2. 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

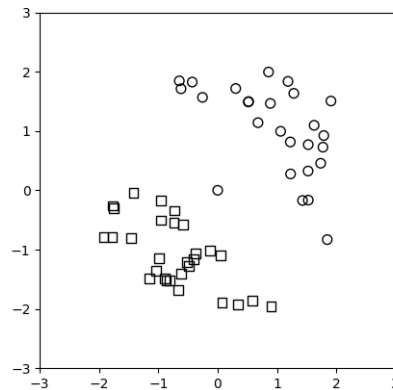


Figure 3: Linearly separable data.

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.

👉 By default, it uses RBF kernel. So, to use the linear kernel, you need to explicitly specify `kernel='linear'`.

- 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 kernels on the SVM using the visualizations.

3.2 Second part

3.2.1 Dataset

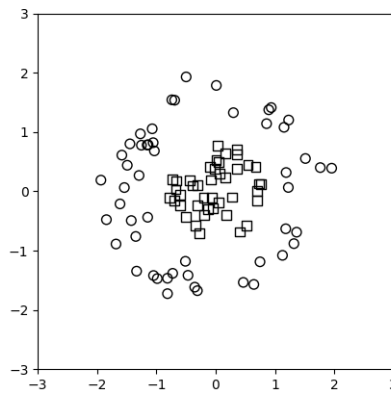


Figure 4: Not linearly separable data.

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. You can use `C=1`.
- 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 using the visualizations.

3.3 Third part

3.3.1 Dataset

The dataset is created by taking the T-shirt/top and Trouser classes, modifying them and reducing the number of examples of the [Fashion-MNIST Dataset](#) [2]. Therefore, you need to use the one we provided.

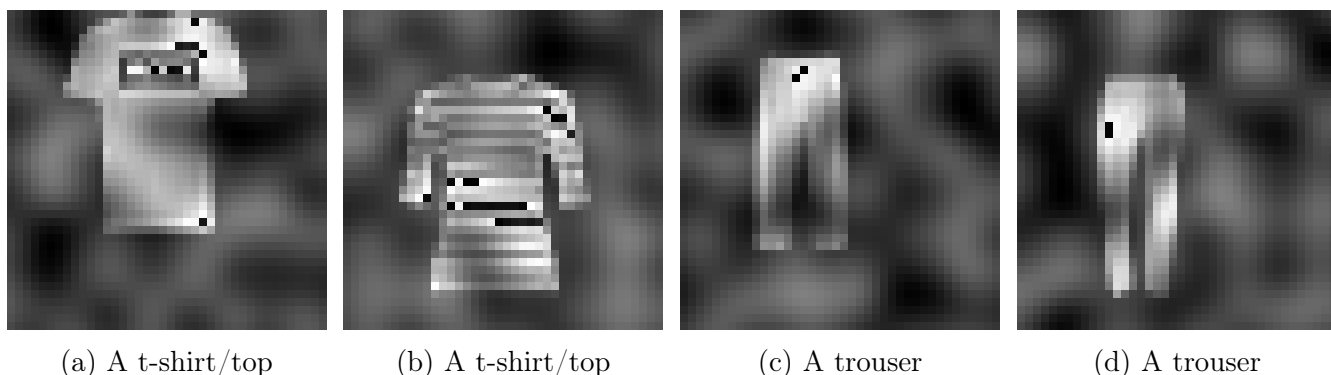


Figure 5: Some examples from the modified Fashion-MNIST dataset.

You can see some examples from the dataset at Fig. 5. 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/fashion_mnist/train_data.npy')
train_labels = np.load('hw3_data/fashion_mnist/train_labels.npy')
test_data = np.load('hw3_data/fashion_mnist/test_data.npy')
test_labels = np.load('hw3_data/fashion_mnist/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. You can simply divide it by 255 since the pixels take values between 0 and 255.

👉 Don't forget to apply the same procedure to the test data.

- Do a grid search on hyperparameters kernel, C, gamma. You can also change the degree for the polynomial kernel but it is not a requirement in this homework. Example hyperparameters are given in the report template; however, feel free to change them. Write the best scores in bold font at each table so that they are easier to spot. Remember that you should fill the whole grid, so increase their numbers carefully. Details of the 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, train your model with those parameters using whole training data and report its test accuracy.

3.4 Fourth part

3.4.1 Dataset


The dataset is also created by taking the T-shirt/top and Trouser classes, modifying them and reducing the number of examples of the [Fashion-MNIST Dataset](#) but in an imbalanced way. 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/fashion_mnist_imba/train_data.npy')
train_labels = np.load('hw3_data/fashion_mnist_imba/train_labels.npy')
test_data = np.load('hw3_data/fashion_mnist_imba/test_data.npy')
test_labels = np.load('hw3_data/fashion_mnist_imba/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 effects.

- Normalize the data. You can simply divide it by 256 since the pixels take values between 0 and 255.

 Don't forget to apply the same procedure to the test data.

- Using rbf kernel with C=1 (the default values), 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.

 Be careful about which number is what in your confusion matrix. If you are using a library; then, its convention may not match with of the course material.

- 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 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.

References

- [1] R. Fisher, "Iris flower dataset," 1936.
- [2] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.