**Data Mining & Machine Learning – Assignment 5**

(20% of total grade)

\*\*\* REMINDER

a) Since this assignment is the last one, **NO late submission** will be accepted after the due date.

b) When the question says “**USING formula**”, you have to explain it in your own words USING the formula given in the slide.

c)

\* If two homework submissions are found to be similar to each other, both submissions will receive 0.

\* Homework solutions must be submitted through Canvas. If you have multiple files, please include all files as one zip file.

\* For coding assignments, it is strongly recommended to use **Jupytor notebook** and submit **.ipynb** file.

\* Answers with **math expressions** and **graphs** can be handwritten and scanned.

\* If you find any **typo/error** in the assignment, let me know.

\*\*\*\*\*

1. [3 pts/ea] In Bias/Variance formula,

1) Explain the meaning of and USING the formula.

is the prediction of the model trained on the D dataset. is the expected prediction of the model trained on all datasets

2) Explain why bagging can reduce the variance error USING its corresponding formula.

2. [5 pts/ea] In Q. 5 in Assignment 3, suppose we use only two lines as a classifier (e.g., L1+L2).

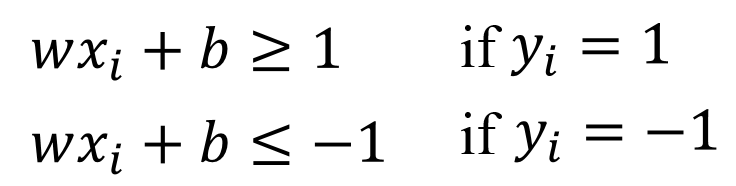
1) Explain whether this classifier shatters 3 data points.

2) Explain whether this classifier shatters 4 data points.

3. [5 pts] SVM linear classifier gives the global optimum. Explain the reason using the formula in p. 9 and p. 11 in svm-p2 slide.

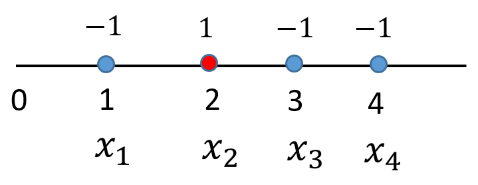
4. [5 pts/ea] Suppose class value ‘1’ should always be classified correctly, but it is okay to misclassify the class value ‘–1’.

1) How do we modify the following formula?



2) Show Lagrangian multiplier of this SVM problem

5. (Refer to p. 29-33 in ‘svm-p2’ slides.) With the following data,



1) [6 pts] Show the formula of dual problem of above problem. (We use the same kernel and C value.)

2) [6 pts] Show the process of computing alpha values of each data. What are support vectors?

3) [2 pts] (Refer to p. 32) After computing b values, show the final discriminant function.

6. Below is the formula of weighted knn classification.

Below is the formula of non-linear SVM.

1) [5 pts] Explain the similarities and differences between knn and non-linear svm USING above formula.

2) [3 pts] What is the meaning of in weighted knn formula and non-linear SVM formula, respectively.

7. In 1-nearest neighbor knn with Euclidean distance, the prediction of test data () is the target value of its single nearest neighbor (). In other words,

1) [6 pts] Change 1-nearest neighbor knn into kernelized 1-nearest neighbor knn. (You can use any kernel function)

2) [5 pts] Explain why kernelizing nearest neighbor algorithm is not a very good idea.

8. In linear regression, we compute the value of (in analytic method). In kernel linear regression, we don’t compute any more.

1) [4 pts] What is the advantage of computing (not ) ? Explain it USING formulas.

2) [8 pts] (Refer p. 12 in kernel slide) Suppose kernel function and all values are initialized to 0.1. Given X=[1,2,3] and Y=[1,4,9], update values.

9. [3 pts] We are going to classify **very long** text documents and consider each word as an independent feature. Explain why kernel method may not be necessary in this case.

10. [3 pts] Explain the picture in p. 31 in kernel slides.

11. [3 pts] Explain the meaning of formula in p. 24 in knn slide.

12. [3 pts] Explain the following formula in AdaBoost.



13. [3 pts] Explain why boosting is recommended in underfiting problem USING the formula in bias/variance.

14. Explain the formulas on p. 22 in boosting slide with your own words.

1) [4 pts] Explain why error function is approximated to

2) [4 pts] Explain why is approximated to the following.

+ const

3) [4 pts] Explain why new weak classifier should be the one that minimizes the residual error from previous ensemble. Explain it USING formulas (you can use the example of regression with squared error).

15. [3 pts/ea] Suppose, in your algorithm, training error is low, but test error is high. You are going to fix this problem. Explain the following cases.

1) Increasing the size of training data will help?

2) Feature selection methods will help?

3) Using cross-validation will help? Explain using the formula of bias/variance.

4) Boosting will help? If boosting is helpful, how exactly are you going to use boosting in this scenario?

16. (coding) knn

1) Implement the following program

def euclidean\_dist(data1, data2):

‘’’

16. 1)-1 [2 pts]

YOUR WORK HERE

Given data1 and data2, compute Euclidean distance

‘’’

def manhattan(data1, data2):

‘’’

16. 1)-2 [2 pts]

YOUR WORK HERE

Given data1 and data2, compute Euclidean distance

‘’’

# create sample data

# n\_features and n\_informative should be same (when n\_redundant=0)

# you can change ‘n\_samples’ and ‘n\_features’ if you want.

X\_train, y\_train = make\_classification(n\_samples = 50, n\_features = 2, n\_informative = 2,

n\_redundant = 0, n\_classes = 2)

# predict X\_test data

‘’’

16. 1)-3 [5 pts]

YOUR WORK HERE

give your own X\_test

compute k neighbors of X\_test data using Euclidean distance

(show the distance of each neighbor)

compute the target value of X\_test

‘’’

2) [4 pts] Show a plot of accuracy vs number of neighbors

3) [bonus 5 pts] Use Manhattan instead of Euclidean and repeat Q. 1)-2).

17. (coding) bagging

1) Implement the following program.

import numpy as np

from sklearn.datasets import load\_iris

from sklearn.tree import DecisionTreeClassifier

from sklearn.model\_selection import train\_test\_split

# Load dataset

data = load\_iris()

X, y = data.data, data.target

# Split data into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3)

# size of each sample

# you can choose your own sample\_size

sample\_size = 20% of length of X\_train

# Store each trained model

list\_models = []

# number of decision trees

no\_classifiers = 10

# Train each model on a bootstrap sample

for \_ in range(no\_classifiers):

‘’’

17. 1)-1 [6 pts]

YOUR WORK HERE

Create a bootstrap sample (with replacement)

Train a decision tree per each sample

(For this purpose, you can use DecisionTreeClassifier in sklearn)

Add the trained decision tree to list\_models

‘’’

‘’’

17. 1)-2 [6 pts]

YOUR WORK HERE

Each model in ‘list\_models’ makes prediction on X\_test

Compute final prediction using majority voting

Show the final predictions of X\_test

‘’’

2) [4 pts] Plot a graph accuracy vs number of classifiers

3) [bonus 5 pts] Instead of decision tree, use another classifier and repeat q. 1)-2).

18. (coding) Gradient Boosting

1) Implement the following Gradient Boosting program: Regression using squared error

from sklearn.tree import DecisionTreeRegressor

# Decision tree visualization

# You can use other tree visualization methods

from sklearn.tree import export\_text

# you can change no of classifiers, if necessary

# show the first 3 steps of boosting only. (You can change this, if necessary)

# T: total number of iterations(classifiers)

T=3

‘’’

18. 1)-1 [3 pts]

YOUR WORK HERE

Create your own data, or you can use an existing data(e.g., iris.csv).

X\_train=

Y\_train=

‘’’

‘’’

18. 1)-2 [3 pts]

YOUR WORK HERE

compute mean of , which is the first weak classifier

: the first classifier

: ensemble of classifiers at step

= {}

‘’’

# add weak classifier

for :

‘’’

18. 1)-3 [6 pts]

YOUR WORK HERE

for each data i, compute the predicted value of

: predicted value of ensemble = { for data i.

compute residual using

create a new Y\_train (Y\_train is replaced by values)

show the revised Y\_train

‘’’

‘’’

18. 1)-4 [3+3+3=9 pts, given as follows]

YOUR WORK HERE

create a new weak classifier using DecisionTreeRegressor [3 pts]

(we assume to use decision trees as weak classifiers)

(For this purpose, you can use DecisionTreeRegressor in sklearn)

Show (visualize) using export\_text or other tools [3 pts]

show values [3 pts]

‘’’

‘’’

18. 1)-5 [2 pts]

YOUR WORK HERE

**if then**

continue for loop

**else**

is the final ensemble set.

**end if**

‘’’

end for