Assignment 3

Homework assignments will be done individually: each student must hand in their own answers. Use of partial or entire solutions obtained from others or online is strictly prohibited. Electronic submission on Canvas is mandatory.

Submission Instructions You shall submit a zip file named Assignment3_LastName_FirstName.zip which contains:

- python files (.ipynb or .py) including all the code, comments, plots, and result analysis. You need to provide detailed comments in English.
- pdf file including (a) solutions for questions 1 and 2 (b) descriptions of observations for questions 3,4,5.

Table 1: Data set

data	x_{i1}	x_{i2}	y_i	α_i
$\mathbf{x}_1$	4	2.9	1	0.414
$\mathbf{x}_2$	4	4	1	0
$\mathbf{x}_3$	1	2.5	-1	0
$\mathbf{x}_4$	2.5	1	-1	0.018
$\mathbf{x}_5$	4.9	4.5	1	0
$\mathbf{x}_6$	1.9	1.9	-1	0
$\mathbf{x}_7$	3.5	4	1	0.018
$\mathbf{x}_8$	0.5	1.5	-1	0
$\mathbf{x}_9$	2	2.1	-1	0.414
$\mathbf{x}_{10}$	4.5	2.5	1	0

1. **Support Vector Machines** (20 points) Given 10 points in Table 1, along with their classes and their Lagranian multipliers (α_i), answer the following questions:
   (a) (7 pts) What is the equation of the SVM hyperplane $h(x)$? Draw the hyperplane with the 10 points.

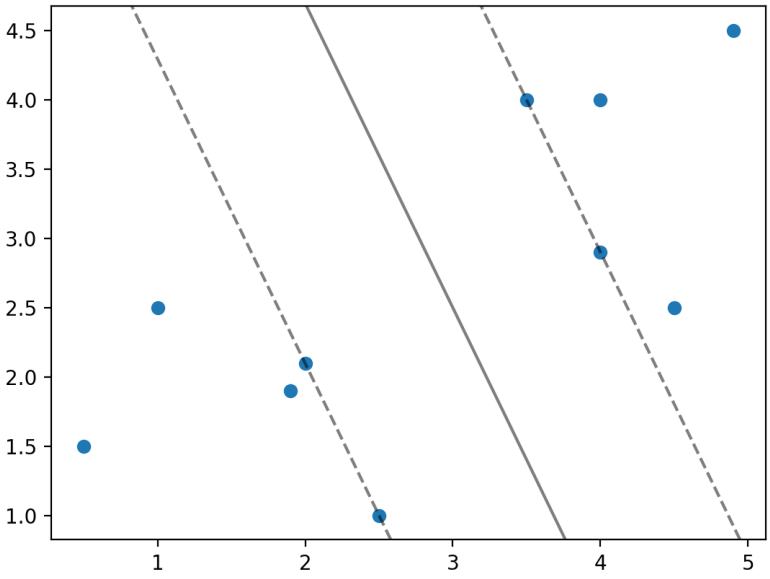
$$w = \sum_{i=i}^{N} \alpha_i y_i x_i = (0.414 * \begin{pmatrix} 4 \\ 2.9 \end{pmatrix}) - (0.018 * \begin{pmatrix} 2.5 \\ 1 \end{pmatrix}) + (0.018 * \begin{pmatrix} 3.5 \\ 4 \end{pmatrix}) - (0.414 * \begin{pmatrix} 2 \\ 2.1 \end{pmatrix}) = \begin{pmatrix} 0.8460 \\ 0.3852 \end{pmatrix}$$

$$\mathbf{b} = \mathbf{y} - \mathbf{w}^T \mathbf{x} = \frac{1}{4} \sum_{i=1}^{N} y_i - \mathbf{w}^T x_i =$$

$$(1 - \begin{pmatrix} 0.8460 \\ 0.3852 \end{pmatrix}^T \begin{pmatrix} 4 \\ 2.9 \end{pmatrix}) + (-1 - \begin{pmatrix} 0.8460 \\ 0.3852 \end{pmatrix}^T \begin{pmatrix} 2.5 \\ 1 \end{pmatrix}) + (1 - \begin{pmatrix} 0.8460 \\ 0.3852 \end{pmatrix}^T \begin{pmatrix} 3.5 \\ 4 \end{pmatrix}) + (-1 - \begin{pmatrix} 0.8460 \\ 0.3852 \end{pmatrix}^T \begin{pmatrix} 2 \\ 2.1 \end{pmatrix})$$

$$= -3.427$$

$$h(\mathbf{x}) = \begin{pmatrix} 0.8460 \\ 0.3852 \end{pmatrix}^T \mathbf{x} - 3.427$$



(b) (8 pts)What is the distance of x_6 from the hyperplane? Is it within the margin of the classifier?

$$\frac{1}{||w||}(w^T x + b) = \frac{1}{\sqrt{0.715716 + 0.14837904}}(\begin{pmatrix}0.8460\\0.3852\end{pmatrix}^T \begin{pmatrix}1.9\\1.9\end{pmatrix} - 3.427) = -1.17$$

No it is not within the margin of the classifier.

(c) (5 pts)Classify the point $z = (3, 3)^\top$ using $h(x)$ from above.

$$h(x) = \begin{pmatrix}0.8460\\0.3852\end{pmatrix}^T x - 3.427 = \begin{pmatrix}0.8460\\0.3852\end{pmatrix}^T \begin{pmatrix}3\\3\end{pmatrix} - 3.427 = 0.2666$$

2. **Support Vector Machines** (20 points) The SVM loss function with slack variables can be viewed as:

$$\min_{\mathbf{w},b} \frac{||\mathbf{w}||^2}{2} + \gamma \sum_{i=1}^{N} \underbrace{\max(0, 1 - y_i f(\mathbf{x}_i))}_{\text{Hinge loss}} \tag{1}$$

The hinge loss provides a way of dealing with datasets that are not separable.

(a) (8 pts)Argue that $l = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})$ is convex as a function of $\mathbf{w}$.
Since the max of two convex functions is convex, we need to prove that both are convex. The first part of the max is 0, which is a constant and constants are convex. Next we take the second dervative of $1 - y\mathbf{w}^\top \mathbf{x}$ which is 0. Since constants are convex, $1 - y\mathbf{w}^\top \mathbf{x}$ is convex. Therefore, $l = \max(0, 1 - y\mathbf{w}^\top \mathbf{x})$ is a convex function.

(b) (5 pts) Suppose that for some $\mathbf{w}$ we have a correct prediction of f with $\mathbf{x}_i$, i.e. $f(\mathbf{x}_i) = \text{sgn}(\mathbf{w}^\top \mathbf{x}_i)$. For binary classifications ($y_i = +1/-1$), what range of values can the hinge loss, l, take on this correctly classified example? Points which are classified correctly and which have non-zero hinge loss are referred to as margin mistakes.
Hinge loss for the correctly classified examples is given by the following loss function:

$$l((x, y), w) = \max\left[0, 1 - y\mathbf{w}^\top \mathbf{x}\right]$$

For all correctly classified examples x_i where $\bar{x_i}$ is the projetion on the boundry,

$$y_i \mathbf{w}^\top x_i \in \left[y_i \mathbf{w}^\top \bar{x_i}, \infty\right]$$

$$y_i \mathbf{w}^\top x_i \in [0, \infty]$$

$$-y_i \mathbf{w}^\top x_i \in [-\infty, 0]$$

$$1 - y_i \mathbf{w}^\top x_i \in [-\infty, 1]$$

$$\max(0, 1 - y_i \mathbf{w}^\top x_i) \in [0, 1]$$

Therefore,

$$loss((x_i, y_i), w) \in [0, 1]$$

(c) (7 pts) Let $M(\mathbf{w})$ be the number of mistakes made by $\mathbf{w}$ on our dataset (in terms of classification loss). Show that:

$$\frac{1}{n} M(\mathbf{w}) \leq \frac{1}{n} \sum_{i=1}^{n} \max(0, 1 - y_i \mathbf{w}^\top \mathbf{x}_i)$$

In other words, the average hinge loss on our dataset is an upper bound on the average number of mistakes we make on our dataset.
Let $M(w)$ be the numner of mistakes made by our model in our dataset of N items. So the total loss could be written as,

$$\sum_{i=1}^{N} \max(0, 1 - y\mathbf{w}^\top x_i) = A + B$$

Let the total loss made by mistakes be A and total loss made by correct predictions be B

$$A = \sum_{i \in mistakes} \max(0, 1 - y\mathbf{w}^\top x_i)$$

$$B = \sum_{i \notin mistakes} \max(0, 1 - y\mathbf{w}^\top x_i)$$

Since we know that

$$\forall_{i \in mistakes} \max(0, 1 - y\mathbf{w}^\top x_i) \geq 1$$

And we know,

$$\forall_{i \notin mistakes} \max(0, 1 - y\mathbf{w}^\top x_i) \in [0, 1)$$
$$B \in [0, N - M(w))$$

Therefore,

$$A + B \in [M(w), \infty)$$
$$\frac{A+B}{N} \in [\frac{1}{N}M(w), \infty)$$

Therefore,

$$\frac{1}{N}M(w) \leq \frac{1}{N}\sum_{i=1}^{N} \max(0, 1 - y\mathbf{w}^\top x_i)$$

3. **Decision Trees** (20 points) Implement a Decision Tree model for the Titanic data set (only use the train file).

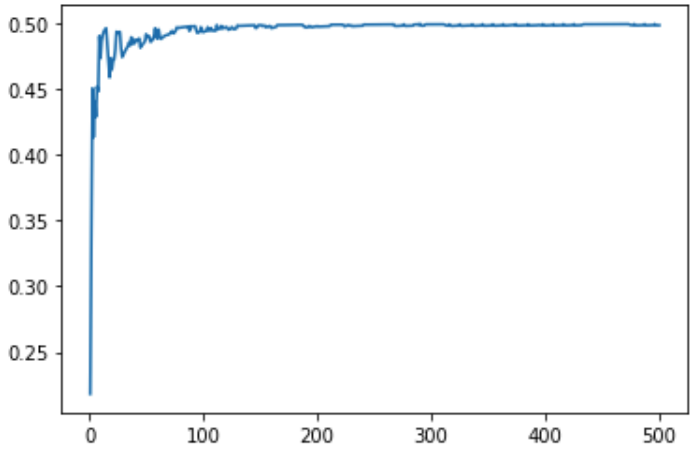
   - Explain how you preprocess the features.
   - Divide the train file into training data and testing data.
   - Build a tree on the training data and evaluate the classification performance on the test data.
   - Compare Gini index and Information Gain.
   - Report your best accuracy on the test data set.
   - Give a brief description of your observations.

   To preprocess the features, I changed the Cabin column to a Has_Cabin column with 0 meaning they did not have a cabin and 1 meaning they did. I next made a FamilySize columm by combining SibSp and Parch columns. Based on the new field, I created a new feature IsAlone with 0 meaning alone and 1 meaning not alone. I then removed all the nulls from embarked, fare, and age column. Then I grouped all non-common titles into one single grouping "Rare". Next, I mapped sex with female being 0 and male being 1 along with titles with "Mr" being 1, "Master" being 2, "Mrs" being 3, "Miss" being 4, and "Rare" being 5, Embarked with "S" being 0, "C" being 1, and "Q" being 2, Fare with any less then or equal to 7.91 being 0, greater then 7.91 and less then or equal to 14.454 being 1, greater then 14.454 and less then or equal to 31 being 2, and greater than 31 being 3, Age with less than or equal to 16 being 0, greater than and less than or equal to 32 being 1, greater than 32 and less than or equal to 48 being 2, greater than 48 and less than or equal to 64 being 3, and anyone greater than 64 was not mapped. I then removed PassengerId, Name, Ticket, Cabin, and SibSp since they were no longer relevant.
   When comparing Gini index and Information Gain we see that they both give the same result. When looking at the difference between the starting value and sex vs. title we see in both that title should be the first split. The best accuracy was 81.69. This was found by using K-Fold crossvalidation to find the best tree depth which was 3. The split was on title and then the next import split was on FamilySize. If you did not have the "Mr." title and your FamilySize was less than or equal to 4.5 then you survived.

4. **Boosting** (20 points) Implement AdaBoost for the Titanic data set. You can use package/tools to implement your decision tree classifiers. The fit function of DecisionTreeClassifier in sklearn has a parameter: sample weight, which you can use to weigh training examples differently during various rounds of AdaBoost.

   - Plot the train and test errors as a function of the number of rounds from 1 through 500.
   - Report your best accuracy on the test data set.
   - Give a brief description of your observations.

   Using AdaBoost, we are able to acheive an accuracy of 83%. When observating the error rate as the rounds progress I saw that it shot up and leveled out at about the 100th round. Since this accuracy is higher then the previous decision tree question, this means that by using AdaBoost we can build a more accurate model.

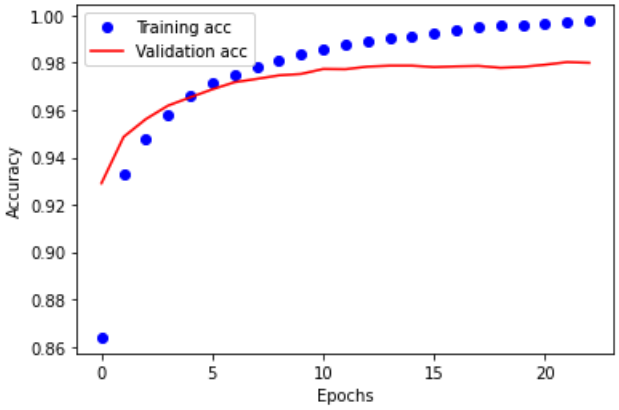


5. **Neural Networks** (20 points) Apply a Neural Network (NN) model to predict a handwritten digit images into 0 to 9. The pickled file represents a tuple of 3 lists : the training set, the validation set and the testing set. Each of the three lists is a pair formed from a list of images and a list of class labels for each of the images. An image is represented as numpy 1-dimensional array of 784 (28 x 28) float values between 0 and 1 (0 stands for black, 1 for white). The labels are numbers between 0 and 9 indicating which digit the image represents. The code block below shows how to load the dataset.

```
import cPickle, gzip, numpy

# Load the dataset
f = gzip.open('mnist.pkl.gz', 'rb')
train_set, valid_set, test_set = cPickle.load(f)
f.close()
```

- Plot the train, validation, and test errors as a function of the epoches.
- Report the best accuracy on the validation and test data sets.
- Apply early stopping using the validation set to avoid overfitting.
- Give a brief description of your observations.



The best accuracy on the validation set was 98% and the best accuracy on the test set was 98%. By using early stopping, I was able to prevent overfitting. I know this because the test set accuracy is the same as the validation accuracy. Also by the graph we can see that the validation accuracy line is not hugging the training accuracy but it is not too far away that we would have underfitting.