Part-1 of Exam-1 - Fall 2024

ECE 469/568 – Machine Learning

SIU Southern Illinois University CARBONDALE

Name (Print): _____

Date: 10/14/2024
Due date: By 11.59 PM on 10/23/2024
Number of pages: 3
Number of problems: 2
Policy: No make-up exams will be given

This exam contains 3 pages (including this cover page) and 2 problems. Check to see if any pages are missing. Enter all requested information on the top of this page, and put your initials on the top of every page, in case the pages become separated.

You are required to show your work on each problem on this exam. The following rules apply:

- **Organize your work**, in a reasonably neat and coherent way, in the space provided. Work scattered all over the page without a clear ordering will receive very little credit.
- **Mysterious or unsupported answers will not receive full credit**. A correct answer, unsupported by calculations, explanation, or algebraic work will receive no credit; an incorrect answer supported by substantially correct calculations and explanations might still receive partial credit.
- Include all codes of your Matlab programs together with all plots as necessary.

Do not write in the table to the right.

Problem	Points	Score
1	50	
2	50	
Total:	100	

1. **Regression in machine learning**

   In this question, you will use the California Housing Prices data-set from the StatLib repository. This data-set is based on data from the 1990 California census.

   Note that the original data-set appeared in R. Kelley Pace and Ronald Barry, "Sparse Spatial Autoregressions", Statistics & Probability Letters 33, no. 3 (1997): 291–297.

   Hint: The following code segment will fetch and load the data.

```
from pathlib import Path
import pandas as pd
import tarfile
import urllib.request
def load_housing_data():
tarball_path = Path("datasets/housing.tgz")
if not tarball_path.is_file():
Path("datasets").mkdir(parents=True, exist_ok=True)
url = "https://github.com/ageron/data/raw/main/housing.tgz"
urllib.request.urlretrieve(url, tarball_path)
with tarfile.open(tarball_path) as housing_tarball:
housing_tarball.extractall(path="datasets")
return pd.read_csv(Path("datasets/housing/housing.csv"))
housing = load_housing_data()
```

   (a) [2 points] Use info() method to identify the attributes of this data-set.

   (b) [3 points] Use describe() method to identify and peek at a summary of the numerical attributes.

   (c) [5 points] Use hist() method on the whole dataset and plot a histogram for each numerical attribute. Notice that many histograms are skewed right.

   (d) [5 points] Clean and normalize/standardize the data-set to make it appropriate for training a regression model. Creating training and test sets. Create a copy of the data with only the numerical attributes by excluding the text attribute ocean_proximity from the data-set.

   (e) [5 points] Because this data-set includes geographical information (latitude and longitude), you are asked to create a scatterplot of all the districts to visualize the geographical data in 2D space.

   (f) [5 points] Compute the standard correlation coefficient between every pair of attributes using the corr() method.

   Hint: use the following code to see how much each attribute correlates with the median house value, for instance.

```
corr_matrix = housing.corr()
corr_matrix["median_house_value"].sort_values(ascending=False)
```

   (g) [5 points] Use scatter matrix (i.e., scatter_matrix() method) to plot every numerical attribute against every other numerical attribute, plus a histogram of each numerical attribute's values on the main diagonal.

(h) [5 points] Add three new attributes; (i) rooms per house = total rooms/households, (ii) bedrooms ratio = total bedrooms/total rooms, and (iii) people per house = population/households.

(i) [5 points] Noticed that the "total bedrooms" attribute has some missing values. You are asked to fix this issue via one of these three options: (i) get rid of the corresponding districts, (ii) get rid of the whole attribute, and (iii) set the missing values to some value (zero, the mean, the median, etc.).

(j) [10 points] Use the above preprocessed data-set to train a linear regression model to predict "median_house_value". Plot the training and test errors against the data-set size. Justify your results as much as possible.

2. **Classification in machine learning**

In this question, you will be using the MNIST data-set, which is a set of 70,000 small images of digits handwritten by high school students and employees of the US Census Bureau. Each image is labeled with the digit it represents.

(a) [15 points] You are asked to build a probabilistic classifier to classify handwritten digits (0-9) in the MNIST dataset that achieves over 97% accuracy on the test set.

Hints: The following code can be used to fetch the MNIST data-set from OpenML.org.

```
from sklearn.datasets import fetch_openml
mnist = fetch_openml('mnist_784', as_frame=False)
```

(b) [10 points] Write a function that can shift an MNIST image in any direction (left, right, up, or down) by one pixel. Then, for each image in the training set, create four shifted copies (one per direction) and add them to the training set. Finally, train your best model on this expanded training set and measure its accuracy on the test set.

You should observe that your model performs even better now. This technique of artificially growing the training set is called data augmentation or training set expansion.

Hint: You can use the shift() function from the scipy.ndimage.interpolation module.

```
shift(image, [2, 1], cval=0)
```

The above code-line shifts the image two pixels down and one pixel to the right.

(c) [10 points] KNN-based algorithms belong to the class of non-probabilistic classifiers. You are asked to design a KNN-based classifier to classify handwritten digits (0-9) in MNIST data-set.

(d) [15 points] Determine the confusion matrix and compute precision, recall and F1 scores for both probabilistic and non-probabilistic classifiers that you designed in Part (a) and Part (c). Discuss your results in terms of performance, computational complexity, and their applicability to mission critical engineering applications.