Assignment 1

Due on October 21, 2021 (23:59:59)

Click here to accept your Assignment 1

Instructions. There are two parts in this assignment. The first part involves a series of theory questions and the second part involves coding. The goal of this problem set is to make you understand and familiarize with K-Nearest Neighbor Regression Algorithm.

Part I: Theory Questions

k-Nearest Neighbor Classification

1. Assume that you have a large training dataset. Specify a disadvantage of the k-Nearest Neighbor method when using it during testing. State also your reason about your answer.
2. Create a 1-Dimensional classification dataset in which the 1-Nearest Neighbors method always gives a leave-one out cross validation error value of 1 (In other words, the method can't guess correct class for a specific point in the dataset). State also a proper explanation about your reasoning.
3. Assume that you have the following training set of positive (+), negative (-) instances and a single test instance (o) in the figure below (Figure 1). Assume also that the Euclidean metric is used for measuring the distance between instances. Finally consider that every nearest neighbor instance affects the final vote equally.

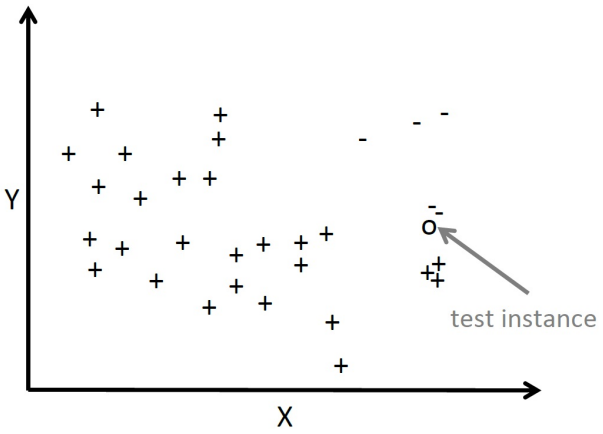


Figure 1: Example KNN Case

- What is the class appointed to the test instance for K=1? State also reason behind your answer.
- What is the class appointed to the test instance for K=3? State also reason behind your answer.
- What is the class appointed to the test instance for K=5? State also reason behind your answer.

4. Fill the blanks with T (True) or F (False) for the statements below:
   - If all instances of the data have the same scale then k-Nearest Neighbor's performance increases drastically. (_)
   - While k-Nearest Neighbor performs well with a small number of input variables, it's performance decreases when the number of inputs becomes large. (_)
   - k-Nearest Neighbor makes no assumption about the functional form of the problem it handles. (_)

Linear Regression

1. Assume that you have five students have registered to a class and the class have a midterm and the final exam. You have obtained a set of their marks on two exams, which is in the table below:

Student	**Midterm Exam**	**Midterm exam (Squared)**	**Final Exam**
$x^{(1)}$	87	7569	94
$x^{(2)}$	70	4900	72
$x^{(3)}$	92	8464	85
$x^{(4)}$	67	4489	76
$x^{(5)}$	45	2025	51

   You plan to a model which form's is $f_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$ for fitting the data above. The x_1 shows midterm exam score while x_2 shows square of the midterm score. Besides you plan to use feature scaling (using divide operation by the "max-min", or range, of a feature) and mean normalization. What is the normalized value of the feature $x_2^{(4)}$?

2. Considering the figure below (Figure 2), which of the offsets used in linear regression's least square line fit? Assume that horizontal axis represents independent variable and vertical axis represents dependent variable. State your answer with your proper explanation.

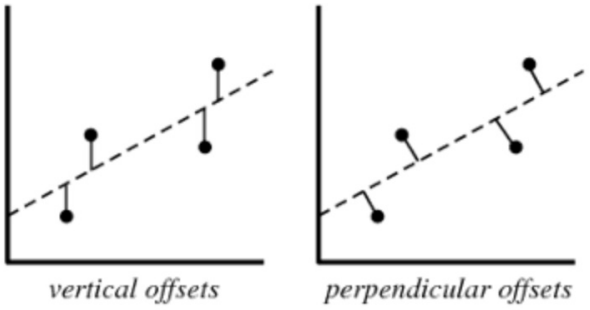


Figure 2: Different Alternative Offsets

3. Considering the table below (Table 1), consisting of four training examples:

x	y
1	0.5
2	1
4	2
0	0

Figure 1

Assume that you are trying to fit the data above to the linear regression model $f_\theta(x) = \theta_0 + \theta_1 x_1$. Find the θ_0 and θ_1 values by using closed form solution ($\theta = (X^T X)^{-1} X^T y$). Also state dimension values of X, y and θ matrices. Finally show your calculations step by step.

4. State a valid reason for feature scaling and explain why it is a valid reason with respect to your reasoning.

PART II: Age Estimation from Face Images

In this part of the assignment, you will implement a nearest neighbor algorithm to estimate age values of people from different face images. You will also extend your implementation as weighted KNN algorithm.

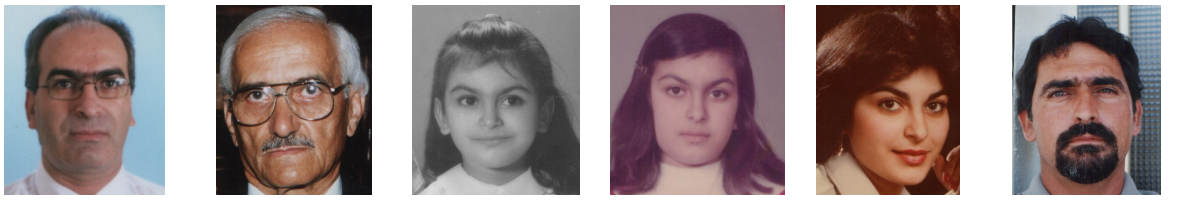
A dataset is provided for your training phase. Test images will be provided later and announced from Piazza group. Since test images will be provided later, you should use a subset of the training set to validate the performance of your model. In other words, you should split your training dataset into two set; training set which will be used to learn model, and validation set which will be used to measure the success of your model. You can use k-fold cross-validation method which is explained in the class. **You should also state your details about each model's accuracy, model accuracies with**

respect to different k parameters you chosen and how you choose the final accuracy in your report.

FGNET Dataset [1]

- You can download the dataset from given **link**.
- Dataset consists of 1002 face images of 82 people with age value range from 0 to 69 values.
- For each face image, the ground-truth age information is embedded within it's name with a format this respectively:
  "First Three Digit": **ID Number of Image**
  "A letter": **Age Delimiter**
  "Two Digits After Age Delimiter": **Ground-Truth Age Value belongs to Image**

- For example, in Figure 3, "001A43a.jpg" represents that "001" is the id number of image, "A" letter represents age delitimier and finally the two digit number after "A" letter represents the ground-truth age value for the related face image, which is 43.
- Another example,"004A62.jpg", 004 is id number and 62 is the age value for the second image from the left.
- The face images in the dataset have various height and width values, so you should resize them to same specific height and width values firstly before using KNN algorithm on them.



001A43a.jpg **004A62.jpg** **008A06.jpg** **008A12.jpg** **008A18.jpg** **013A44.jpg**

Figure 3: Different face images from various age values. The ground-truth age information is embedded in each image's name.

Features

- There is no limitation about features. You can use any feature that you think it's proper for your age regression/estimation assignment. Some features are listed below:
  1. Tiny images: You can resize images to a very small size.

2. Shape features: Shape is an important and powerful feature for regression process on images. You can use shape information extracted using histogram of edge detection. Edge information in the image is obtained by using the Canny edge detection.
3. Texture features: The texture feature is extracted usually using filter based method. The Gabor filter is a frequently used filter in texture extraction.

- You can use more than one feature by concatenating them.

Steps you need to follow:

1. Extract features for each face image in training set (Canny, Gabor, etc.).
2. For each given test face image/sample,
   - predict it's age value using k-NN.
   - predict it's age value using weighted k-NN.
3. Finally compute performance of your model to measure the success of your KNN-Regression method for each setting you have used:

   Mean Absolute Error (MAE) $= \frac{1}{n}\sum_{i=1}^{n}|d_i - \hat{d}_i|$
   where,
   d_i is the actual (ground-truth) age value of face image
   $\hat{d}_i$ is the predicted/estimated age value of face image
   n is the number of images

   You will report mean error by averaging your MAE results for k folds (cross-validation).
4. **Error Analysis**
   - Find a few misestimated images and comment on why you think they were hard to estimate.
   - Compare performance of different feature choices and investigate the effect of important system parameters (number of training images used, k in k-NN, etc.). Wherever relevant, feel free to discuss computation time in addition to regression/estimation rate.
5. **(Bonus) Deep Image Features** Extract deep image features of VGG-19 net for age estimation process of images.

Submit

You are required to submit all your code (*all your code should be written in Jupyter notebook* long with a report in ipynb format (should be prepared using Jupyter notebook). The codes you will submit should be well commented. Your report should be self-contained and should contain a brief overview of the problem and the details of

your implemented solution. You can include pseudocode or figures to highlight or clarify certain aspects of your solution. You can also include a table to report your results like:

Feature	MAE
Gabor	0.5
Canny	0.08
Attribute	0.3

Finally, prepare a ZIP file named **name-surname-pset1.zip** containing

- report.ipynb (PDF file containing your report)
- code/ (directory containing all your codes as Python file .py)

The ZIP file will be submitted via Github Classroom. Click here to accept your Assignment 1

NOTE: To enter the competition, you have to register kaggle in Class with your department email account. The webpage of the competition will be announced later. Top 2 assignment will earn extra points.

Grading

- Code (70): k-NN: 20, Weighted k-NN: 40, Part 5 (Bonus): 10
- Report(40): Theory part: 12 points, Analysis of the results for prediction: 28 points.

  Note: Preparing a good report is important as well as the correctness of your solutions! You should explain your choices and their effects to the results. You can create a table to report your results.

Late Policy

You may use up to four extension days (in total) over the course of the semester for the three problem sets you will take. Any additional unapproved late submission will be weighted by 0.5. You have to submit your solution in (rest of your late submission days + 4 days), otherwise it will not be evaluated.

Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode)

will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.

References

[1] Fu, Yanwei, et al. "Interestingness Prediction by Robust Learning to Rank." European conference on computer vision. Springer, Cham, 2014.