

CS461 Homework 2

Due: Oct. 20 11:59 pm

1. Suppose someone asked you to identify a linear model used to generate the noise-free data below ($y = w_0 \cdot 1 + w_1x_1 + w_2x_2 + w_3x_3$). You plan to use MMSE regression to estimate the original model.

data num	(x_1, x_2, x_3)	y
d_1	(4, 1, 1)	16
d_2	(7, 0, 2)	23
d_3	(10, 1, 3)	36
d_4	(13, 0, 4)	43

1.1 Please write a data-matrix $\Phi(4 \times 4)$.

1.2 When the normal equation is shown as below, will it give you an exact solution or MMSE approximated solution?

$$\Phi^t \Phi \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} = \Phi^t \begin{bmatrix} 16 \\ 23 \\ 36 \\ 43 \end{bmatrix}$$

1.3 In the normal equation $\Phi^t \Phi$ is invertible? If not, how would you solve it? Please solve the equation and find \vec{w} .

1.4 The person showed you the original model as $y = 1 + 2 \cdot x_1 + 3 \cdot x_2 + 4 \cdot x_3$. Is it the same as yours? If not, why this happened?

1.5. The person suggested that, with four additional data points, you could obtain the same result as the original model. Please add four data points below to your normal equation and compute the new \vec{w} . Do you have a different answer from the previous one? What is the chance to obtain the original model using the regression method?

data num	(x_1, x_2, x_3)	y
d_5	(16, 1, 5)	56.04
d_6	(19, 0, 6)	62.77
d_7	(22, 1, 7)	76.04
d_8	(25, 0, 8)	82.96

1.6. Carefully examine your data matrix from 1.1. If you could delete a column to ensure a unique solution, which column would you remove?

2. [Lagrangian Function and KKT Conditions] Suppose you have two data points as below to estimate $y = w_0x_1 + w_1x_2$.

data num	(x_1, x_2)	y
d_1	(1 0)	1
d_2	(0,1)	1

2.1 Please set an MMSE objective function $J(\vec{w})$ with the data. What is the optimal solution (w_0, w_1) that minimizes the function?

2.2 We have only two data points, so the constrained optimization problem for regularization is formulated below. Please define the Lagrangian function for the problem.

$$\begin{aligned} \vec{w}^* &= \arg \min_{\vec{w}} J(\vec{w}) \\ &\text{subject to } \|\vec{w}\|^2 \leq C \end{aligned}$$

2.3 Based on KKT conditions, compute the optimal Lagrangian parameter and \vec{w}^* for the different $C = \{0.5, 1, 2, 3\}$. Hint: geometrical contours will help you to find the relation between w_{*0} and w_{*1} .

3. [Learning Sinusoidal Functions] You will recover a sinusoidal function $f(x)$ from noisy data by using MMSE regression. Data samples and required specifications are given below.

- train and test data files: train.npz, test.npz, and readme.txt
- use ten polynomial basis functions: $1, x, x^2, \dots, x^8, x^9$
- use MSE (Mean Square Error) for the performance metric: $\frac{1}{N} \sum_{i=1}^N (\vec{w}^t \phi(x_i) - y_i)^2$
- use five-folds cross-validation for the selection of the parameter of ridge regression λ

3.1 Write the code “ols_regression.py” to implement an MMSE regression model (five cross-validation and ordinary MMSE) and report one validation error averaging the five cross-validations.

3.2 Write the code “ridge_regression.py” to implement a ridge regression model (five cross-validation and ridge MMSE) and plot the averaged validation error for different ridge parameters: $0 \leq \lambda \leq 1$. Hint: possible λ s are $1e-8, 1e-7, \dots, 0.5, 1$ but the spacing is up to your design. Based on the plot, select the λ^* and save the corresponding models $w(\lambda^*)$. Additionally, save the five models for the ordinary MMSE $w(\lambda = 0)$.

3.3 Plot the two models: $w(\lambda = 0)$ and $w(\lambda^*)$ over the range $0 \leq x \leq 1$. The five models can be averaged: `w=np.mean(w, axis=0)`.

3.4 Evaluate the two models with “test.npz” and report the two test MSEs.

3.5 Write the code “ols_regression_largeset.py” to implement a regression model with “train_1000.npz” without any regularization and cross validation. Plot the model over the range $0 \leq x \leq 1$.

3.6. Based on your answers in 3.3, 3.4, and 3.5, please provide two possible solutions to control the effective complexity in machine learning.

4. [Eigenface] Spectral decomposition is applied to a large set of images to extract the most dominant correlations between images. You will approximate one test facial image by using $\vec{x} \simeq \vec{x}' = \bar{x} + E_M E_M^t (x - \bar{x})$. E_M is the eigenvectors of $COV(X, X)$ that correspond to the M largest eigenvalues where random vector X represents the whole facial data.

4.1 Please compute $COV(X, X)$ and $E[X]$ and spectral decomposition $COV(X, X) = E \Lambda E^t$ by using the gif images in “train” folder. You can use numpy and PIL and no need to submit your computations.

4.2 Approximate the test image in “test” folder for different M values: 2, 10, 100, 1,000, 4,000. Present the five images in your report. Select one representative image for each M.

4.3 Represent the eigenvectors corresponding to the 10 largest eigenvalues in grayscale images. Please include the ten images in your report and explain how the eigenimages capture the various features and aspects of human faces. Hint: you may need this formula for visualization:

$$\frac{x - \min(x)}{\max(x)} \times 256$$

5. [Extra 25 Points, Estimation of Year of Made] You will build a predictor of the year of made for art by using the embedding collected from a deep-CNN style classifier (VGG-16). It is a high dimensional embedding as 512-D. After conducting dimensional reduction to 2-D and whitening, a polynomial feature map will be constructed for regression. Data samples and required specifications are given below.

- train and test data files: vgg16_train.npz, vgg16_test.npz, readme.txt
- use any order polynomial basis functions as you want: for example, $1, x_1, x_1^2, x_2, x_1x_2, x_2^2$
- use MSE (Mean Square Error) for the performance metric

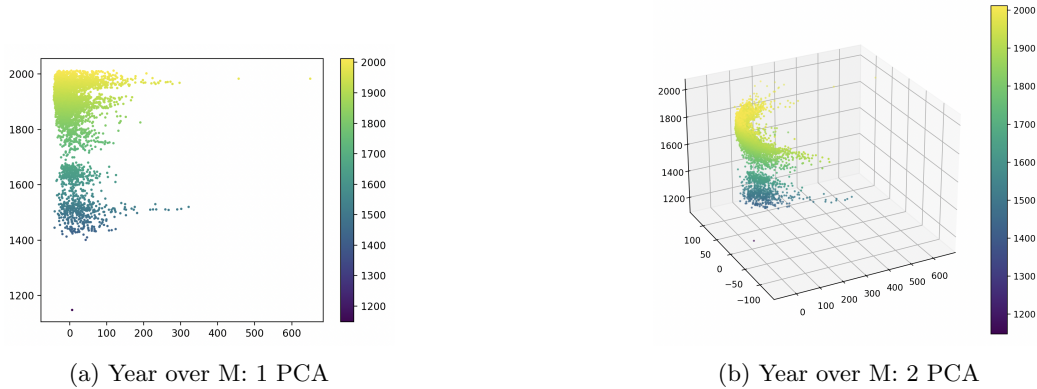


Figure 1: Year Information over PCA Space

5.1 Dimensional Reduction and Whitening: Please use PCA formula $\Lambda^{-1/2} E_M^t (x - \bar{x})$. Scatter plots for the case of $M=1$ and $M=2$ are shown above. Please reproduce the plots. Based on the figures, please reason why 2-D projection will be more promising in year prediction than 1-D case.

5.2 Train: Write the code “year_train.py” to implement a 2-D regression model to predict the year of made using vgg16_train.npz. Try any order of polynomial basis as you want and test your models on a validation set. For simplicity, please use a subset of the training set for validation to finalize your model (no cross-validation). Hint: you could increase the order until observing overfitting or performance stagnation.

5.3 Test: Write the code “year_test.py” to evaluate your model on vgg16_test.npz. Report a test MSE score and identify the image where your model shows the most accurate prediction and the image where it shows the least accurate prediction (report image filenames).