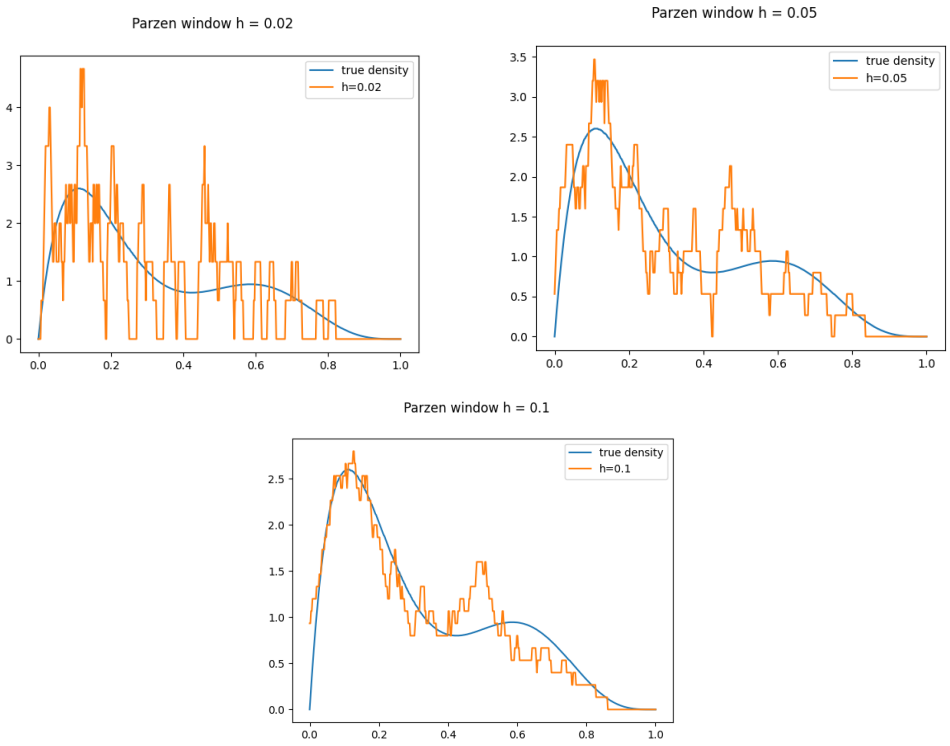
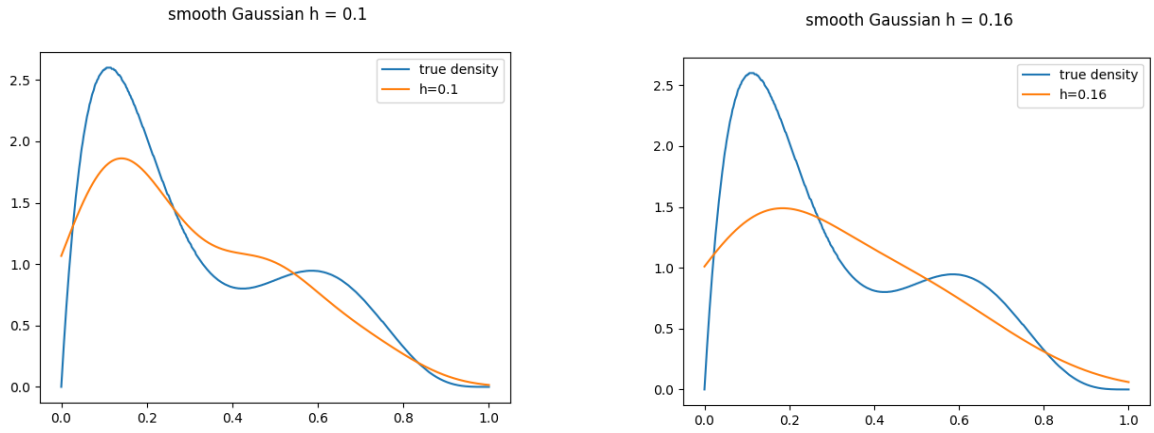
Problem 1

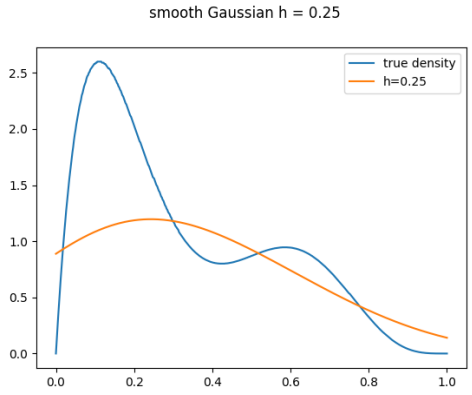
Part (c)



The 3 density estimations by Parzen window have the similar trends as the true density, such as increasing, decreasing, and peaks. Smaller h tends to give more spiky graphs with more discontinuities. As h increases, the density values display more smoothly (fewer sharp changes between points) and get closer to the true density.

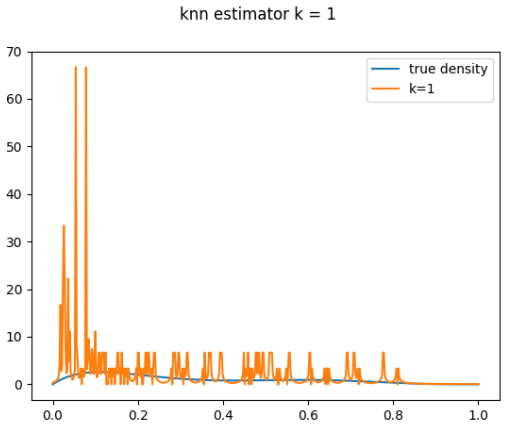
Part (d)

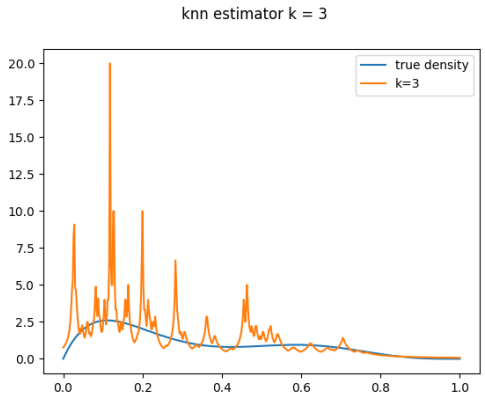


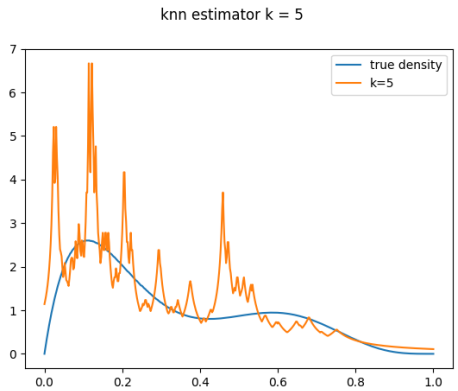


As h gets larger, the gaussian kernel estimation becomes smoother, but it gets farther away from the true density and some details may disappear and thus less accurate, like the peaks and the increasing/decreasing changes of the pdf values. For example, when h=0.1, there are still two small peaks in the kernel estimation similar to the true density; but when h=0.16, the second peak disappears, and the pdf decreases all the way after the peak. Compared with part c, this one is much smoother and has no discontinuities.

Part (e







As k increases, the estimation gets closer to the true density and shows more details, such as the peaks. Compared with the previous 2 methods, this one is spikier, has more discontinuities, and is overall farther away from the true density values.

Problem 2

Part 2.1

Mean squared training error: 22.68

Mean squared testing error: 33.55

Because the mean squared error on the train set is smaller, so the training error is better. In this case, I don't think we should worry about overfitting because 1) there is not a large discrepancy between the training and testing errors; 2) the training error is not very small compared with the testing error (training is not perfect); 3) the testing error is larger than the training error.

Part 2.2

(b) The new dimension of the vector is 91.

(c) Mean squared training error for quadratic model: 5.10

Mean squared testing error for quadratic model: 228.43

Compared with the linear model, the linear one in part 2.1 is better because its testing error is much smaller than the quadratic model one. Here, there is a large discrepancy between training and testing errors, and the training error is very small, so we should worry about overfitting problems.

Part 2.3

(b)

The weights: [[6.93430416 -0.85540777 0.13480408 -2.33579468 0.22280293 -0.43404051

2.46628645 0.84243118 2.3334114 -2.72416103 -2.49630149 -1.49561396

2.82062375 -1.88035981]]

Mean squared train error for online gradient descent: 250.56

Mean squared test error for online gradient descent: 752.00

Thus, it's worse than the results in part 2.1 because both the train and test errors are larger here.

(c)

1. change the learning rates when the number of steps is fixed at 1000

1) when fix learning rate to 0.01

Weights: [[2.01800377e+01 -1.71870018e+00 -1.96480933e-02 -1.32104303e-01
  1.13138222e+00 3.33317169e-01 3.25567265e+00 4.73367990e-01
 -6.73121206e-01 -3.02669647e+00 -1.92375819e+00 -9.12877285e-01
  1.74509174e+00 -6.05481066e+00]]

Mean squared train error for online gradient descent: 53.21

Mean squared test error for online gradient descent: 181.65

2) when fix learning rate to 0.05

Weights: [[17.94676445 -5.01771358 -1.35229637 0.04479351 -0.21812524 -2.25929296
  5.25738036 0.22031375 -1.06106387 -6.18399386 -3.23766006 0.14145789
  1.94782805 -4.03465317]]

Mean squared train error for online gradient descent: 153.74

Mean squared test error for online gradient descent: 592.01

3) when the learning rate is 2/sqrt(n)

Weights: [[4.30681424e+07 1.15620483e+08 -1.31976679e+06 -4.85462435e+06
 -3.32575900e+07 8.40664356e+06 -9.24126745e+06 -7.80777374e+06
  6.67736316e+06 1.13016754e+07 6.97308704e+06 3.51796112e+06
 -1.50332039e+07 -2.00129808e+07]]

Mean squared train error for online gradient descent: 11934962207434230.00

Mean squared test error for online gradient descent: 49036114424442272.00

2. change the number of steps when the learning rate is fixed at 0.01

1) when the number of steps is 500

Weights: [[1.75470534e+01 -1.96034035e+00 -6.94598316e-03 6.77590566e-01
  2.87987820e+00 1.75950556e+00 1.56330576e+00 4.56171395e-01
  1.44808306e-01 -3.95014505e+00 -2.42108832e+00 -5.61367805e-01

2.53710897e+00 -4.79293797e+00]]

Mean squared train error for online gradient descent: 65.72

Mean squared test error for online gradient descent: 264.06

2) when the number of steps is 3000

[[21.45634664 -1.09104248 0.32759367 0.30156091 0.34227307 -1.45569121

1.87045996 0.55118678 -1.70922412 0.59170678 -1.02614201 -0.83569021

0.30362281 -4.49141135]]

Mean squared train error for online gradient descent: 29.82

Mean squared test error for online gradient descent: 16.24

Observations:

When the number of steps is fixed, the larger the learning rate, the worse the errors will be. When the learning rate is fixed, the larger the number of steps, the better the errors will be. Between these 2 cases, changing learning rate when fixing number of steps has a more significant impact on the errors.