ISTA 421/521 – Homework 3

Due: Friday, October 11, 10pm

20 pts total for Undergrads, 25 pts total for Grads

STUDENT NAME

Undergraduate / Graduate

Instructions

In this assignment you are required to modify/write 2 scripts in python. They will be submitted as 2 separate files, although you are free to copy chunks of code from any other script, especially the example scripts being provided in this homework. Details of what you are to do are specified in problems 2 and 7, below.

Included in the homework 3 release are following sample scripts:

- `approx_expected_value.py` - This script demonstrates how to approximate an expected value through sampling. You will modify this code and submit your solution for problem 2.
- `gauss_surf.py` - This is provided for fun – it is not required for any problem here. It generates a 2d multivariate Gaussian and plots it as both a contour and surface plots. In later work in this class, we will use contour plots.
- `predictive_variance_example.py` - This script demonstrates (a) generating and plotting error bars (predictive variance) and (b) sampling of model parameters from the $\text{cov}\{\widehat{\mathbf{w}}\}$ estimated from data. You will run this script in problem 6, and then use it as the basis for a script in problem 7.
- `w_variation_demo.py` - This script is also provided for fun and is not required for the assignment. However, it provides more python code as an example. This implements the simulated experiment demonstrating the theoretical and empirical bias in the estimate of variance, $\widehat{\sigma^2}$, of the model variance, σ^2, as a function of the sample size used for estimation.

All problems require that you provide some "written" answer (in some cases also figures), so you will also submit a .pdf of your written answers. (You can use LaTeX or any other system (including handwritten; plots, of course, must be program-generated) as long as the final version is in PDF.)

The final submission will include (minimally) the two scripts and a PDF version of your written part of the assignment. You are required to create either a .zip or tarball (.tar.gz / .tgz) archive of all of the files for your submission and submit your archive to the d2l dropbox by the date/time deadline above.

NOTE: Problems 4 and 8 are required for Graduate students only; Undergraduates may complete them for extra credit equal to the point value.

(FCMA refers to the course text: Rogers and Girolami (2012), *A First Course in Machine Learning*. For general notes on using LaTeX to typeset math, see: `http://en.wikibooks.org/wiki/LaTeX/Mathematics`)

1. [2 points] Adapted from **Exercise 2.3** of FCMA p.90:

   Let Y be a random variable that can take any positive integer value. The likelihood of these outcomes is given by the Poisson pdf:

   $$p(y) = \frac{\lambda^y}{y!} e^{-\lambda} \tag{1}$$

   By using the fact that for a discrete random variable the pdf gives the probabilities of the individual events occurring and the probabilities are additive...

   (a) Compute the probability that $Y \leq 5$ for $\lambda = 7$, i.e., $P(Y \leq 5)$. Write a (very!) short python script to compute this value, and include a listing of the code in your solution.

   (b) Using the result of (a) and the fact that one outcome has to happen, compute the probability that $Y > 5$.

   Solution.
   a)

```
numpy as np
import matplotlib.pyplot as plt

y = 5 ## value for parameter y as defined in assignment
lamb = 7 ## value for parameter lambda as defined in assignment
answer = 0  ## declaring what will become the answer to problem
for x in range(0,y+1):
    e = np.exp(-lamb)
    fact = 1
    for i in range(1,x+1):
        fact = fact * i

    num = lamb ** x
    den = fact

    frac = num/den
    inst = frac * e ##instance

    answer = answer + inst

print('The probability that y is less than or euqal to ' + str(y) + ' is ' +  str(answer))
print('The probability that y is greater than ' + str(y) + ' is ' + str(1-answer))
```

   $P(Y \leq 5) = 0.300009166667$

   b)
   $P(Y > 5) = [1 - P(Y \leq 5)] = 0.699990833333$

2. [3 points] Adapted from **Exercise 2.4** of FCMA p.90:

   Let Y be a random variable with uniform density, $p(y) = \mathcal{U}(a, b)$. Derive $\mathbf{E}_{p(y)}\{\sin(y)\}$. Note that $\int \sin(y)\, dy = -\cos(y)$. Compute $\mathbf{E}_{p(y)}\{\sin(y)\}$ for $a = 0$, $b = 1$ (show the steps).

   The script `approx_expected_value.py` demonstrates how you use random samples to approximate an expectation, as described in Section 2.5.1 of the book. In this case, the script estimates the expectation of the function x^2 when $X \sim \mathcal{U}(0, 1)$ (that is, y is uniformly distributed between 0 and 1). This script shows a plot of how the estimation improves as larger samples are considered, up to 100 samples.

Modify the script `approx_expected_value.py` to compute a sample-based approximation to this expectation of the function $\sin(y)$ when $Y \sim \mathcal{U}(0,1)$ and observe how the approximation improves with the number of samples drawn. Include a plot showing the evolution of the approximation, relative to the true value, over 3,000 samples.

Solution.

$$E_{P(y)} = \int_0^1 \sin(y)P(y)\,dy = \frac{-\cos(y)}{b-a}\Big|_0^1 = \frac{-\cos(1)+\cos(0)}{1-0} = 0.45969769413186023$$

3. [3 points] Adapted from **Exercise 2.5** of FCMA p.91:

   Assume that $p(\mathbf{w})$ is the Gaussian pdf for a D-dimensional vector $\mathbf{w}$ given in

$$p(\mathbf{w}) = \frac{1}{(2\pi)^{D/2}|\mathbf{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{w}-\mu)^\top \mathbf{\Sigma}^{-1}(\mathbf{w}-\mu)\right\}.$$

   By expanding the vector notation and re-arranging, show that using $\mathbf{\Sigma} = \sigma^2\mathbf{I}$ as the covariance matrix, assumes independence of the D elements of $\mathbf{w}$. You will need to be aware that the determinant of a matrix that only has entries on the diagonal ($|\sigma^2\mathbf{I}|$) is the product of the diagonal values and that the inverse of the same matrix is constructed by simply inverting each element on the diagonal. (Hint, a product of exponentials can be expressed as an exponential of a sum. Also, just a reminder that $\exp\{x\}$ is e^x.)

   Solution.

$$p(\mathbf{w}) = \frac{1}{(2\pi)^{D/2}\begin{bmatrix}\sigma^2 & 0\\ 0 & \sigma^2\end{bmatrix}^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{w}-\mu)^\top \begin{bmatrix}\frac{1}{\sigma^2} & 0\\ 0 & \frac{1}{\sigma^2}\end{bmatrix}(\mathbf{w}-\mu)\right\}.$$

4. [2 points; **Required only for Graduates**] Adapted from **Exercise 2.6** of FCMA p.91:

   Using the same setup as in Problem 4, see what happens if we use a diagonal covariance matrix with different elements on the diagonal, i.e.,

$$\mathbf{\Sigma} = \begin{bmatrix}\sigma_1^2 & 0 & \cdots & 0\\ 0 & \sigma_2^2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \sigma_D^2\end{bmatrix}$$

   Solution. <Solution goes here>

5. [4 points] Adapted from **Exercise 2.9** of FCMA p.91:

   Assume that a dataset of N binary values, $x_1, ..., x_n$, was sampled from a Bernoulli distribution, and each sample x_i is independent of any other sample. Explain why this is *not* a Binomial distribution. Derive the maximum likelihood estimate for the Bernoulli parameter.

   Solution.
   A binomial distribution is a distribution of the probability of a number of successes, that is to say what is the probability of x successes out of N instances given a probability p of success in each instance. Under the given circumstances one could approximate a binomial distribution by finding the expected value of the Bernoulli Distribution (with sufficient data). But by simply having a series results from a

Bernoulli distribution does not make binomial distribution, but merely an instance found on a possible binomial distribution.

Maximum likelihood of a Bernoulli is:

$$\frac{\sum_{i=1}^{N} x_i}{N}$$

Where x is the outcome of a Bernoulli Distribution.

6. [3 points] Adapted from **Exercise 2.12** of FCMA p.91:

   Familiarize yourself with the provided script `predictive_variance_example.py`. When you run it, it will generate a dataset and then remove all values for which $-0.5 \leq x \leq 2.5$. Observe the effect this has on the predictive variance in this range. Plot (a) the data, (b) the error bar plots for model orders 1, 3, 5 and 9, and (c) the sampled functions for model orders 1, 3, 5 and 9. You will plot a total of 9 figures. Include a caption for each figure that qualitatively describes what the figure shows. Also, clearly explain what removing the points has done in contrast to when they're left in.

   Solution.

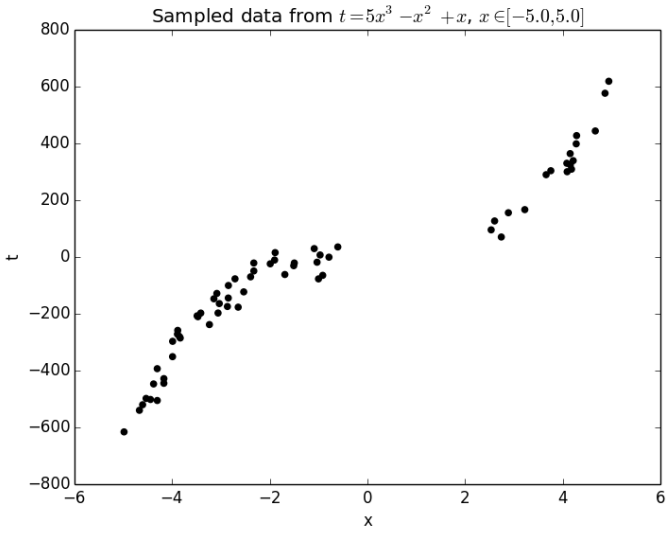
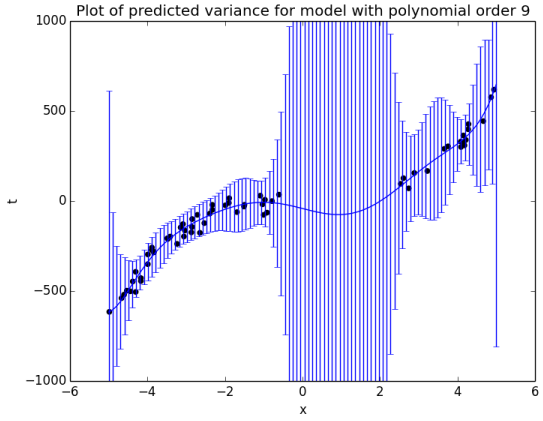
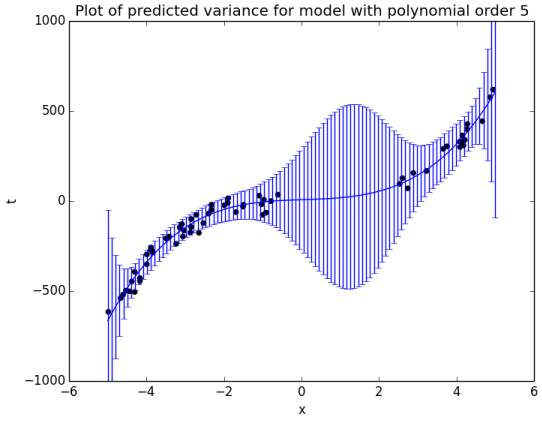


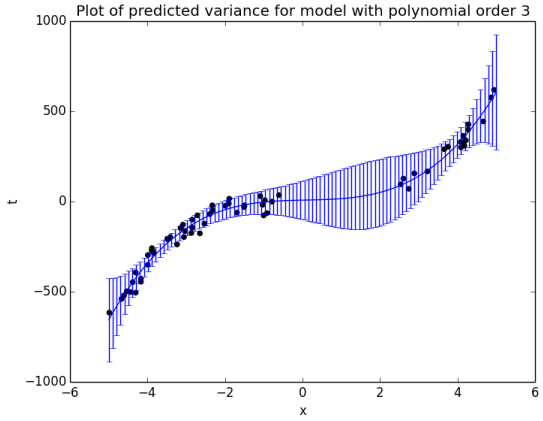
Figure 1: This plot is of the data after the data found between $-0.5 \leq x \leq 2.5$ is removed



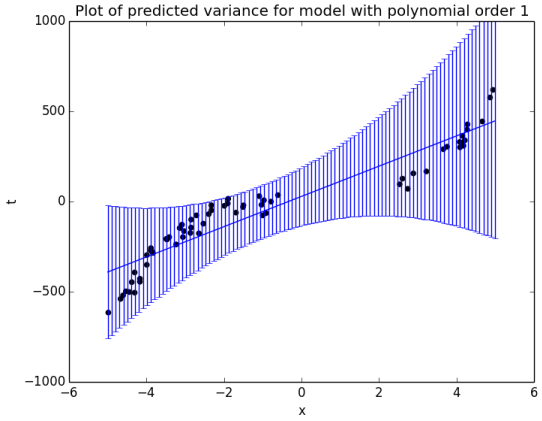
(a)



(b)

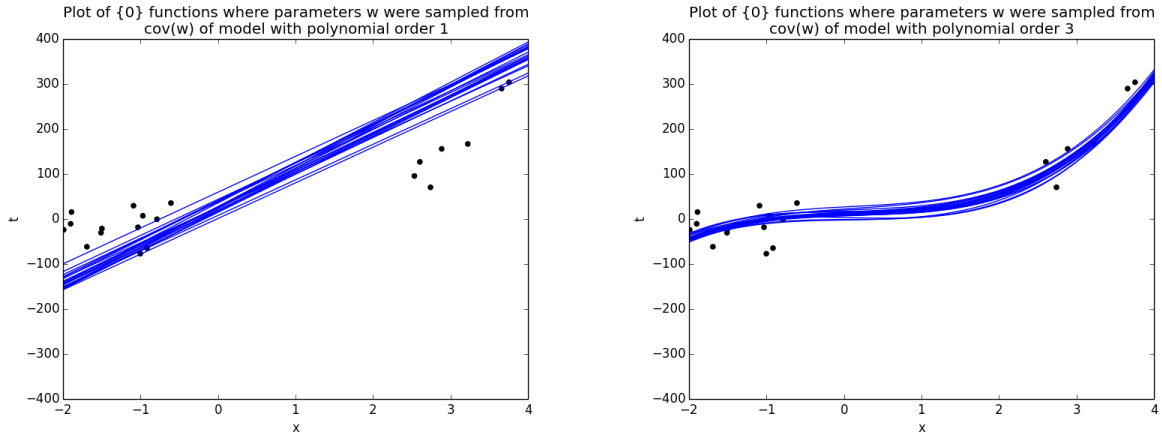


(c)



(d)

Figure 2: Fig a shows the error bars of a 9^{th} order regression, note that the error is greatest in the areas at the edge and in the middle, where the data is missing. Fig b shows similar tendencies but is a 5^{th} order regression and more accurate than fig a. Fig c is a 3^{rd} order regression and shows the smallest error space, thought the error is still greatest around the missing data. Fig d is a 1^{st} order regression and has the smallest error around the missing data relative to the overall error, but is still less accurate than that of figure c and even fig b.



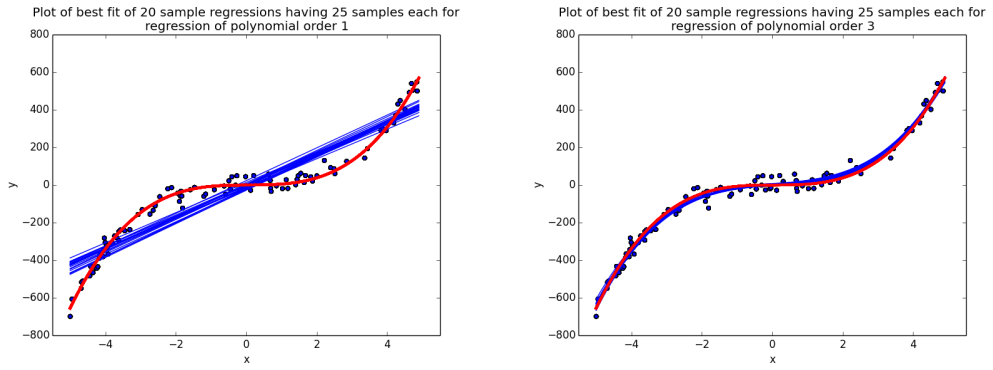
(a) (b) (c) (d)

Figure 3: Each diagram shows several instances of possible models created by sampling the data, each conform to the error bars seen in the previous diagrams.

7. [5 points]

In this exercise, you will create a simple demonstration of how model bias impacts variance, similar to the demonstration in class. Using the same true model in the script `predictive_variance_example.py`, that is $t = 5x^3 - x^2 + x$, generate 20 data sets, each consisting of 25 samples from the true function (using the same range of $x \in [-5.0, 5.0]$). Then, create a separate plot for each of the model polynomial orders 1, 3, 5 and 9, in which you plot the true function in red and each of the best fit functions of that model order to the 20 data sets. You will therefore produce four plots. The first will be for model order 1 and will include the true model plotted in red and then 20 curves, one each for an order 1 best fit model for each of the 20 data set, for all data sets. The second plot will repeat this for model order 3, and so on. You can use any of the code in the script `predictive_variance_example.py` as a guide. Describe what happens to the variance in the functions as the model order is changed. (tips: plot the true function curve last, so it is plotted on top of the others; also, use `linewidth=3` in the plot fn to increase the line width to make the curve stand out more.)

Solution.



(a) (b)

(c) (d)

Figure 4: As seen, the data shows that the regression models conform to a would be predicted variance for each curve with the areas in the center of the distributions more tightly spaced than that of the extremes. The ideal fit is the 3^{rd} order polynomial wight he least variance, with 5^{th} order model coming in a close second. The 1^{st} order regression conforms to the mean values of the sample points, giving it perhaps the poorest fit for prediction, while the 9^{th} order polynomial over fits the center of the samples, but diverges wildly at the extremes. Given the plots provided, I would say that as the order increases to the best model, the models better conform to the data, however as the model's order progresses past the best fit, the potential of for error at the extremes increases with each successive order and moves closer to the center of the sample.

8. [3 points; **Required only for Graduates**] Adapted from **Exercise 2.13** of FCMA p.92: Compute the Fisher Information Matrix for the parameter of a Bernoulli distribution.

   Solution. <Solution goes here>