Homework #1

MTH 9899 Baruch College
DATA SCIENCE II: Machine Learning

Due: April 6, 2016 - 18:00

Notes

- Code for this **MUST** be written in Python.
- Do NOT use 3^{rd} Party Packages for the regression functions.
- One thing not mentioned in class is the relationship between λ and the size of the dataset. As the dataset grows in size, you will generallly need to include higher λ values.
- For this assignment, please submit the relevant graphs and a short paragraph, as asked for below. Also, please submit a copy of your python code.

Problem 1 In our first lecture, we spoke at length about Ridge Regression and the tradeoff between bias and variance. In this question, we'll run some simulations to see how λ can affect the variance of β. Below, you will find python code to generate 4 different datasets for testing. Based on this code, you need to:

i For datasets 1-4 in the code, generate each dataset 1000 times. For each of these 1000 times, perform simple OLS regression and record the β values. Plot a histogram of the β values and report the μ_β and σ^2_β.

ii Repeat the above trials with Ridge Regression instead, using reasonable λ values. Prepare a graph of how μ_β and σ^2_β change as a function of λ for each of the datasets - you do NOT need to include histograms of all of your distributions. Also, please calculate the effective degrees of freedom, to make sure that the λ values you are using are reasonable, you should see effective DOFs from 2 down to less than 1.

iii Calculate the expected $\mathrm{v}ar(\beta^R)$ using the formulas from class. How do these compare to the simulated distributions for β? Please explain any differences.

```
import numpy as np

def generate_data(num_rows, num_features, true_betas, sigma_2 = 1, seed = None):
    ”””
    Args:
      num_rows (int): The number of sample rows of data
      num_features (int): The number of features
      true_betas (array): The true beta values used to generate y
      sigma_2 (float): The multiplier for the random noise
    """
    if seed:
        np.random.seed(seed)
    X = np.random.randn(num_rows, num_features)
    Y = X.dot(true_betas) + np.random.randn(num_rows) * sigma_2
    return X,Y


def get_dataset(set_num):
    if set_num == 1:
        return generate_data(1000, 2, np.array([1,1]))
    elif set_num == 2:
        return generate_data(50, 2, np.array([1,0]));
    elif set_num == 3:
        return generate_data(250, 2, np.array([1,0]));
    elif set_num == 4:
        return generate_data(100000, 2, np.array([1,0]));
    assert "Shouldn't be here"
```