CS 1219: Introduction to Machine Learning

HW-1 (Given: Sep 15, 2021. Due on: Sep 22, 12 pm IST)

Total Points: 100 (tentative)

Logistics:

We will be using Google Colaboratory for this assignment. You are required to MAKE A COPY of the assignment notebook attached to this document, and then play around with it. **Any changes made within the notebook authored by us will not be saved and you will lose your progress**. You do not have to submit your notebook for this assignment, please take a look at the submission guidelines as given below. You do not have to code anything up from scratch in this assignment, just specify some values of your choice as prompted, and judiciously use functions provided to you, and you are good to go!

Submission Guidelines:

1. This is an exploratory assignment. **We expect you to submit a PDF file** with your observations on running experiments with the set-up that we have provided. You can add all your answers and relevant plots from your notebook in this PDF doc itself.
2. You can use any editor of your choice, but you must convert it to the PDF format before submission. **Make sure to mention your name at the top of the document. Mention if you have collaborated with anyone.**
3. Filename convention: **FirstName_LastName_HW#_CS1219.pdf.** Replace # with the corresponding assignment number.
4. Submit your answer scripts **through DropBox**. Click here for the link.
5. Lastly, in case you decide to use your own scripts/notebooks, please submit all your files as a zipped folder. Use the same naming convention.

Late Submission Policy:

The deadline for submission of this assignment is : **22 Sep, 12 pm IST**.

Late submissions **will not be accepted.**

Exceptions will be made only if you obtain prior approval from the professor, and let the TAs know of the same.

Assignment Overview and Goals:

The goal of this homework is to help you get familiar with the Bias-Variance tradeoff. Currently, we are working in a supervised learning set-up, where all our data is explicitly labelled by *some* oracle function. Our task is to build models that approximate the function as closely as possible.

As a designer of machine learning models, your aim should be to build a model that best fits not just the data available to your machine (called "training data"), but also to unseen data (typically called "test data") that may be made available to your machine in the future.

We saw that as the model complexity increases, so does its tendency to "overfit" the training data (Ref. Fig. 1). We understand bias as the tendency of a learner to consistently learn the wrong thing by not taking into account all the information available to it, in some senses, underfitting the data. We understand variance as the difference between the model's performance on the training data and the test data, and in effect, a high variance indicates that the model has not performed well in some cases, which becomes evident during test-set evaluation; typically, this indicates a problem in the direction of overfitting to training data.

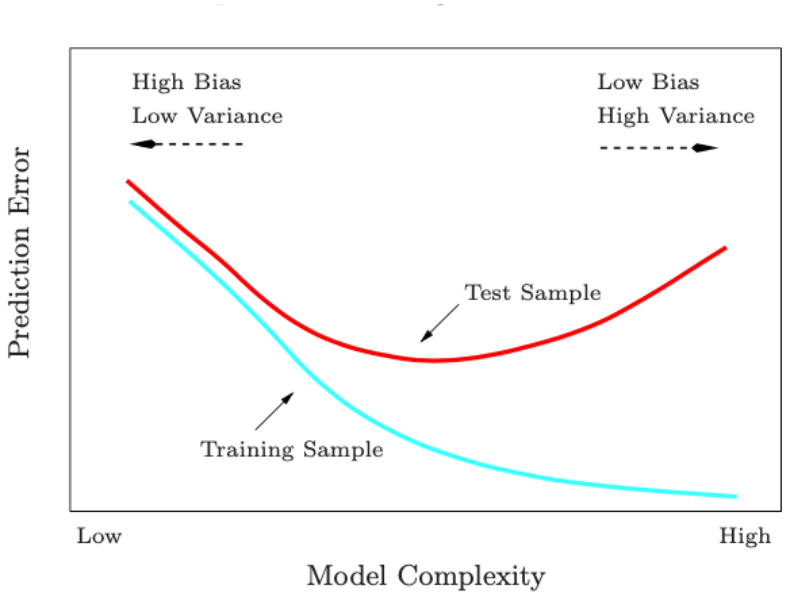

Figure:1

(Source: A screenshot from the text, "An Introduction to Statistical Learning", by Trevor Hastie, Robert Tibshirani and Jerome Friedman)

Thus bias and variance are somewhat complementary, i.e. reducing one (most often) increases the other. The trick is to allow one to increase if the other decreases more than the increase. [Read More]

As a designer one looks for that sweet spot where you get best of both, the least bias and the least variance.

Your work starts here:

We have provided a notebook here for you to experiment with and see the tradeoff for yourself. Read instructions on the notebook carefully.

You must answer the following questions:-

Regression Analyses:

1. You have been provided with a collection of functions which you can generate data and sample from. Pick a function and generate enough samples so that you can plot them. You must do this for the **linear** (`"line"`)**, exponential** (`"exponential"`) **and polynomial functions** (`"polynomial-n"`). **Bonus points** if you play around with the other functions or incorporate your own.
2. For each of the datasets that you generate, you must **fit a regression model**. You must vary the parameters of the regression model and figure out what the optimal parameter will be. In other words, vary the degree of the polynomial used by your regression model.
3. Plot the prediction error vs complexity of the model (in our case, this simply translates to the degree of the polynomial used to regress) in each case. This should look similar to Fig. 1. You will have to implement this on your own.

K-Nearest Neighbours:

1. You are provided with the **Iris Dataset**. Repeat above steps. Except, this time, **fit a KNN classification model.**
2. Plot the prediction error vs complexity of the model (think about what this translates to in this case). This should look similar to Fig. 1. You will have to implement this on your own.
3. Bonus questions are embedded in the code cells.

Don't worry about how pre-written libraries work for the moment. Your goal is to simply be able to use them.
