LAB 6 - Cubic Spline Regression

In this lab, we are going to implement a cubic spline regression. A cubic spline, like any other non-linear regression technique, can be implemented using standard multiple linear regression.

IMPORTANT NOTE: For the last time in this course, no outside packages for regression fitting are allowed! The packages allowed are limited to numpy, matplotlib and pandas. Starting from our next lab session, we will start using other packages which make regression fitting much easier.

Our input file is now a different and a bigger one. It is a .csv file with various statistics taken from Wimbledon and Roland Garros grand slam matches (men category) from 2013. We are going to investigate the effects of a player's first serve percentage to the number of points won of that player during his first serve In this case, our "x" is going to be the "FSP.1" column in the .csv file, and our "y" is going to be the "FSW.1" column.

- (60 pts) Implement the cubic spline regression in a function with three parameters: your x and y (which can be taken from the data sheet), and your *knot* values in a separate array. Inside this function:

    - (35 pts) Create your X matrix:

      Just like multiple linear regression, we need an input matrix. But this time, aside from our x, our other independent variables are going to be some function of x. See the image below for details:

$$\begin{aligned} b_1(x_i) &= x_i \\ b_2(x_i) &= x_i^2 \\ b_3(x_i) &= x_i^3 \\ b_{k+3}(x_i) &= (x_i - \xi_k)^3_+, \end{aligned} \quad \rightarrow \quad \mathbf{X} = \begin{array}{c} \begin{array}{ccccc} & \mathrm{x_i} & \mathrm{x_i}^2 & & (\mathrm{x_i}-\xi_k)^3_+ \\ & \downarrow & \downarrow & & \downarrow \end{array} \\ \begin{bmatrix} 1 & x_{21} & x_{31} & \cdots & x_{k1} \\ 1 & x_{22} & x_{32} & \cdots & x_{k2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{2n} & x_{3n} & \cdots & x_{kn} \end{bmatrix} \end{array}$$

Note: The function $(f(x))_+$ evaluate to:

- 0, if $f(x) < 0$, and
- $f(x)$, if $f(x) \geq 0$

Where x_i is our input data, $\mathcal{E}_k$ is the knot value at the k^{th} knot, and $k = 1,2,3,\ldots,K$. This means that in your input matrix X, you will have $3 + k + 1$ columns (including the ones column).

- (15 pts) Sort the rows of your matrix using the second column (the column where x_i is). **You must also rearrange y so that it correctly corresponds to the matrix.**

- (10 pts) Calculate coefficients, and return the regression line:

  You can calculate the coefficients using:

  $$\hat{\boldsymbol{\beta}} = \left[\mathbf{X}'\mathbf{X}\right]^{-1}\mathbf{X}'\mathbf{y}$$

  After calculating the coefficients, calculate the regression line as your prediction and return it. You can calculate the line using:

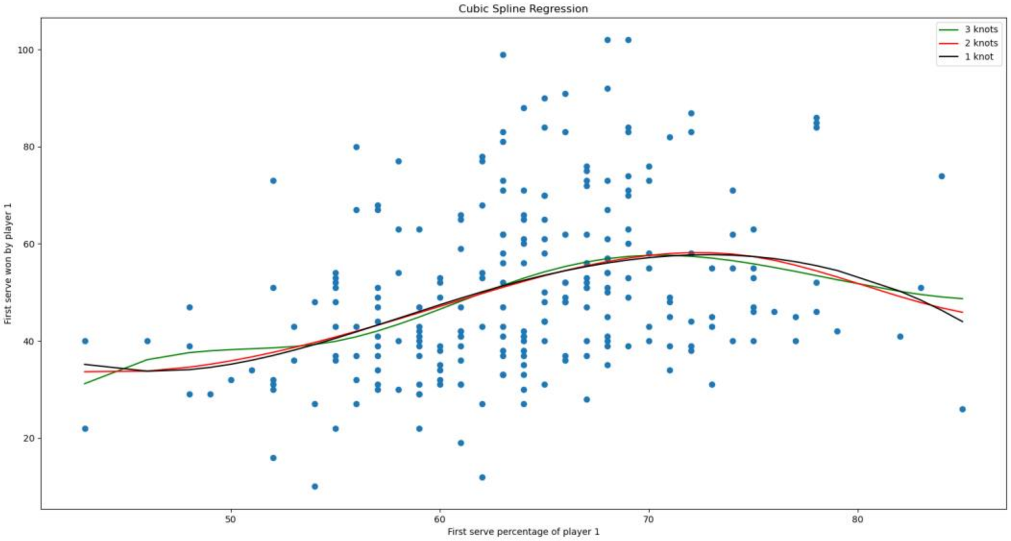
  $$\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}}$$

  Also, extract and return the column containing your sorted x values (the column where x_i is).

- (10 pts) In the main part of the script, read your data sheet, extract your x and y vectors. Call the function you implemented three times, using the same x and y, but different knot values. The knot values you should pass to the function are:

  1) 3 knots at 55, 65 and 70
  2) 2 knots at 60 and 75
  3) 1 knot at 62

- (30 pts) Since we have a single independent variable x, we can easily plot our regression lines. Plot the data points (x and y, the values you initially extracted from the .csv file) as a scatter plot, then plot all three regression lines, all in different colors, *on the same window*. In all 3 line plots, the x-axis is the x column you extracted in the function, and the y-axis is the predictions you calculated in the function. You should get a result like this:


Cubic Spline Regression
3 knots
2 knots
1 knot
First serve won by player 1
100
80
60
40
20
50
60
70
80
First serve percentage of player 1

Continuing with the insight section:

We implemented a step-wise model in the form of cubic splines. A step-wise function is one which changes within certain intervals. The intersection points of these intervals are called *knots*.

A cubic spline function enforces that each step has a third degree polynomial (hence the name *cubic*). Each step has a different polynomial, but the overall function should be *continuous*. To enforce this continuity, we simply make it so that each step's polynomial function evaluates to the same number at the knot values. The equations for the cubic spline are formed according to these constraints.

The polynomial functions for each step are determined according to our data, using the regression coefficients. But the knot values are *arbitrarily* chosen, so please feel free to experiment with different knot values to see the effect of knot placement!

We can use cross-validation or other techniques to choose approximately optimal knot values.