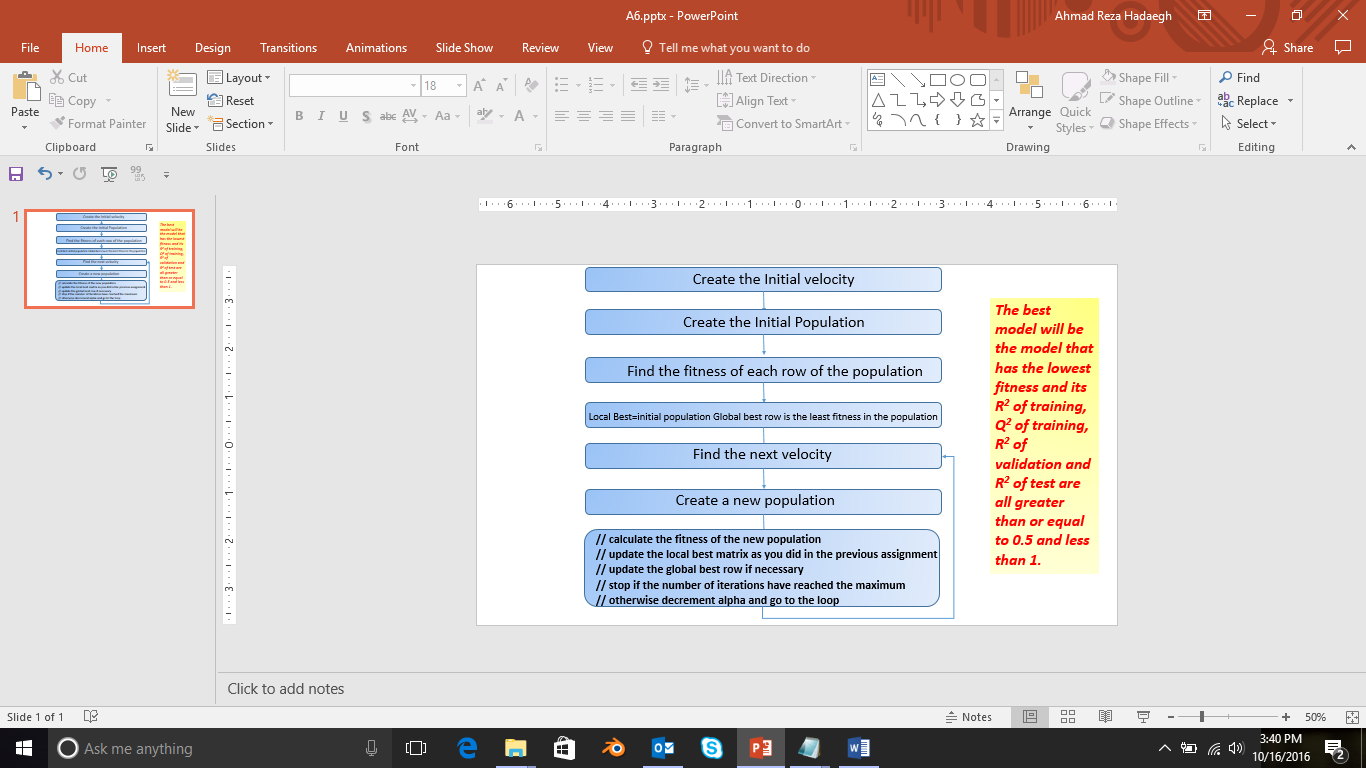
**CS512 - Assignment 6:**

Write a Python program to use the scheme of DE-BPSO (combination of Differential evolution and Binary Particle Swarm Optimization) to predict a linear model for an HIV inhibitor (Pill). You can either choose a linear model such as “Multiple Linear Regression”, “Support Vector Machine”, or “Partial Least Square Regression”, or a none-linear model such “Artificial Neural Network”. The algorithm of DE-BPSO is:



**Step 1:**

**// Create the initial velocity (call it matrix V) that is between 0 and 1 (not binary)**

**for (i=0; i<50; i++)**

**for (j=0; j<385; j++)**

**{**

**V[i,j] = random number between 0 and 1;**

**}**

**---------------------------------------------------------------------------------**

**Step 2:**

**// Create the initial population (call it matrix X) based on the values of the initial velocity**

**for (i=0; i<50; i++)**

**for (j=0; j<385; j++)**

**{**

**if (V[i,j] <= Lambda) // Note: The value of Lambda is 0.01**

**X[i,j] = 1;**

**else**

**X[i, j] = 0;**

**}**

**---------------------------------------------------------------------------------**

**Step 3:**

**// Find the fitness of each row of the 50 models in the first population**

**---------------------------------------------------------------------------------**

**Step 4:**

**// Since you only have the first population at this time, your local best**

**// matrix (call it matrix P) become the same as the first population**

**// Therefore:**

**P = X**

**// The row with the best fitness in P becomes the global best. Let’s call that row Row "G"**

**// Therefore:**

**G = the row in P with the best fitness**

**---------------------------------------------------------------------------------**

**Step 5:**

**// This section is going to be repeated until the end of the program**

**// In this section we need to find the next velocity matrix**

**// We are using Differential Evolution (DE) algorithm to find the new velocity matrix**

**for (i=0; i<50; i++)**

**for (j=0; j<385; j++)**

**{**

**Randomly select 3 rows from the populations and call them as r1, r2, and r3**

**Let r = r3 + F \* (r2 - r1) // the value of F should be set to 0.7**

**// Do the cross mutation of row "i" and "r"**

**if ((random between 0 and 1) < CR) // not binary, CR = 0.7**

**V[i,j] = r[j]**

**else**

**V[i,j] = V[i,j] // remains unchanged**

**}**

**---------------------------------------------------------------------------------**

**Step 6:**

**// This section creates the new population. The value of Alpha**

**// starts from 0.5 and is decremented to 0.33. The value of beta is 0.004**

**for (i=0; i<50; i++)**

**for (j=0; j<385; j++)**

**{**

**if ( (alpha < V[i,j]) && (V[i,j] <= 0.5\*(1+alpha))**

**X[i, j] = P[i,j];**

**else if ( (0.5\*(1+alpha)) < V[i,j]) && (V[i,j] <= (1-beta))**

**X[i,j] = G[j] // the global vector value**

**else if (1-beta) < V[i,j]) && (V[i,j] <=1))**

**X[i.j] = 1 - X[i,j]**

**else**

**X[i,j] = X[i,j]; // remains unchanged**

**}**

**---------------------------------------------------------------------------------**

**Step 7:**

**// calculate the fitness of the new population**

**// update the local best matrix as you did in the previous assignment**

**// update the global best row if necessary**

**// stop if the number of iterations have reached the maximum**

**// otherwise decrement alpha and go to step 5**

**---------------------------------------------------------------------------------**