LAB 6 – Random Forests

In this lab session, we are going to implement random forests with many different parameters, visualize the results and try to examine the differences between different parameters.

We are going to use the `RandomForestRegressor` class from the `sklearn` library for this task. You can examine its documentation here.

Random forests have many parameters which we can alter to control tree characteristics. We are going to investigate some of those in this lab. Please browse through the documentation link given above to clearly understand what each parameter refers to.

- (5 pts) Build X and y as we did previously. This time, since we aren't using a linear model, we don't need the ones column in the input, so exclude it. X consists of the columns "Age", "Height", "Mental" and "Skill". y equates to the "Salary" column.

- (5 pts) For number of trees n from 10 to 100, and for depth d from 3 to 12 (you need a nested loop for this), do the following:

    - (5 pts) Create a `RandomForestRegressor` object. Set its `n_estimators` and `max_depth` parameters to n and d.

    - (5 pts) Fit the tree using the train input and train output:

      ```
      reg.fit(X_train, y_train)
      ```

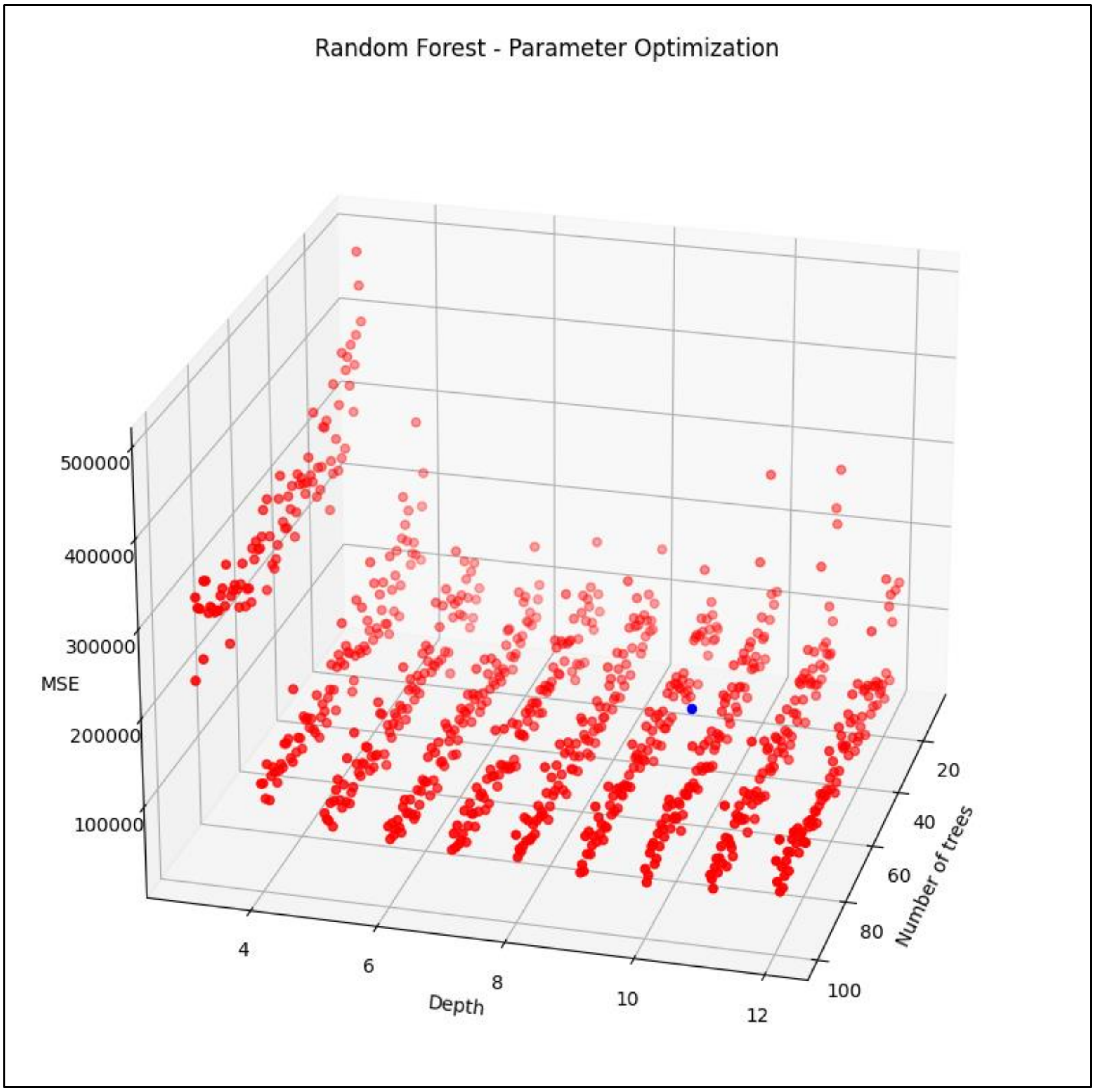
      where `reg` is the `RandomForestRegressor` object.

    - (20 pts) We need a reliable way of evaluating this model. Instead of cross-validating, random forests have a convenient feature called *out-of-bag predictions.* Find where it is stored, calculate the MSE using these predictions and y. Store the corresponding n, d and MSE.
        - (10 pts) (Bonus) When n is small, you will encounter some warnings on the console when running, because some of the samples will not have out-of-bag predictions. You can solve this by converting these warnings into exceptions using the `warnings` module and implementing the section above within a try-except block.

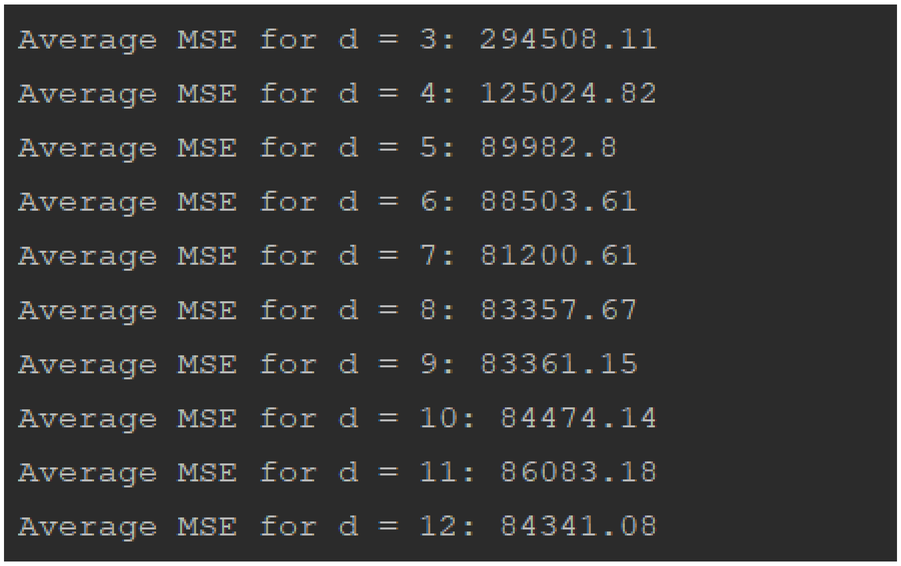
    - (5 pts) This loop will have a very high running time. To observe results during runtime, print n, d and MSE on the console.

- Plotting:
  - (25 pts) Plot the n, d and MSE values you have stored previously as a 3D scatter plot. Make the plot red.
    - (5 pts) (Bonus) As an exception, get the n, d and MSE where MSE is the minimum, and plot it blue.
  - (5 pts) Set axis labels and plot title accordingly.

Here is how the plot looks like (it will look slightly different at each run):



- (25 pts) Depth analysis:
    - Find the average MSE where $d = 3$, and display the result on the console. Repeat for all other d values. Here is an example console output:



```
Average MSE for d = 3: 294508.11
Average MSE for d = 4: 125024.82
Average MSE for d = 5: 89982.8
Average MSE for d = 6: 88503.61
Average MSE for d = 7: 81200.61
Average MSE for d = 8: 83357.67
Average MSE for d = 9: 83361.15
Average MSE for d = 10: 84474.14
Average MSE for d = 11: 86083.18
Average MSE for d = 12: 84341.08
```