Group C: Neptune Experiment's Tasks

The python file you will be provided with contains the solution to a regression problem, predicting housing prices and has been done using SK-Learn. You will perform the following instructions and tasks where you will alter certain data such as hyperparameter and dataset features, and run the regression using two different supervised learning algorithms: Linear_regression and Random Forest Regressor algorithms. This will then lead us to see how neptune.ai helps you track the associated assets. You will perform the required steps to track, retrieve and query relevant data from the experiment' runs using Neptune.

Preparation

1. You are required to track the following assets/data using Neptune where applicable while running the experiment:
    - The datasets and features used for each run
    - Relevant parameters and hyper-parameters are used in each run.
    - The model generated in each run
    - The evaluation metrics obtained from generated model

    For each run, you are to log the relevant data (parameters, datasets, model files, and metrics) for the regression task, while training and evaluating the regression model.

2. Log in to Neptune.ai with your credentials.
3. Create a new project, with the name "experiment"
4. Open the file neptune-c.py
5. Initialize Neptune and link it to the project you created, (Below Line 18).
6. In script neptune-c.py, a data frame has been created for you in line 28. Uncomment line 37 to ensure the data and its features are tracked with Neptune.
7. The parameters used for the RFR algorithm are defined in line 42.(**Don't uncomment just yet**). However, uncomment line 50 to ensure the split_param is tracked, which determines the ratio of training to test data size, is also tracked.
8. Lines 60-66 train a model using a simple linear regression algorithm from sci-kit learn. Ensure the generated model is converted to a binary file (using *get_pickled_model(*)) and tracked by uncommenting and updating line 71.
9. Lines 77 - 80, code evaluates the model created from step 8. The calculated evaluation metrics are RMSE, mean_absolute_error and r2.
    a. **Uncomment** below **lines 82** to track the metrics values.

Experimenting with algorithm 1 (Linear Regression)

Carry out the following experimental runs using different sets of dataset features and learning parameters as described below. Execute the python script at each run to train and evaluate the model performance. Open the terminal window, and navigate to the *neptune-c.py.*

10. **Run 1**: Use the default dataset features and parameter values and execute the python scripts with the following command
    $ python neptune-c.py

11. **Run 2**: Change the normalize parameter to False and execute the script again:
    Run the script again: *$ python neptune-c.py*

12. **Run 3**: In the Diabetes dataset we have a total of 10 different features as shown in the image below:

    - age age in years
    - sex
    - bmi body mass index
    - bp average blood pressure
    - s1 tc, total serum cholesterol
    - s2 ldl, low-density lipoproteins
    - s3 hdl, high-density lipoproteins
    - s4 tch, total cholesterol / HDL
    - s5 ltg, possibly log of serum triglycerides level
    - s6 glu, blood sugar level

    Now we will train the model with the following 3 features: 'age', 'sex', 'bmi'. To do this, uncomment line 32 (The line which has "# Feature 2" next to it).

    Run the file again: *$ python neptune-c.py*

13. **Run 4**: We have used 80% (0.20) of the dataset as a training set, change the test_size to 70% (0.30). That's in split_param, for a change.

```
# Splitting the data and setting test_size
split_param = {'test_size': 0.20, 'random_state': 28750}

X_train, X_test, y_train, y_test = train_test_split(data, y, **split_param)
```

    Run the script again: *$ python neptune-c.py*

14. **Run 5:** We will now replace one of the features to train the model. In line 32 (The line which has “# Feature 2” next to it), change the "age" to "s5". Also, set the “normalize” parameter to True.

    Run the file again: *$ python neptune-c.py*

15. **Run 6:** Use all the features from the dataset to train the model. Do this by commenting line 32 (line with “# Feature 2” next to it).

    Run the file again: *$ python neptune-c.py*

16. **Run 7:** Retrieve the model generated in EX-2 (i.e, **Run 2**) (uncomment line 54). Test the model using the dataset prepared in line 31 (uncomment line 31 with “# Feature 1” next to it).

    - Retrieve the model by uncommenting line 54 this downloads it from **Run 2**
    - Uncomment line 56-57.
    - Add run attribute in neptune.init(`run='<RUN_ID>'`) `# in this case 'EX-2'`

    Run the file again: *$ python neptune-c.py*

Experimenting with algorithm 2 (RFR algorithm)

We have carried out a series of runs using the Linear Regression algorithm, and now we will use the RFR algorithm. This will give us different results compared to the previous algorithm.

First, do the following to prepare your script.

- Below line 18, where you initialise neptune, remove the run attribute from neptune.init().
- Comment out line 54 (Where you download the model)
- Comment out lines 56-57
- Uncomment line 60 (RFR algorithm)
- Comment out the current model, LinearRegression (line 61)
- Uncomment out line 31 (The line with “# Feature 1”)

17. **Run 8:** Comment out the Linear regression(Line 61) model with (#), and uncomment the RandomForestRegressor(Line 60), as shown below:

```
model = RandomForestRegressor(**parameters)
#model = LinearRegression(normalize=True)
```

    Run the file again: *$ python neptune-c.py*

18. **Run 9**: The parameter we are using for the RFR is in line 42, uncomment it.

```
# The parameters for Random Forest Regressor
parameters = {'n_estimators': 50, 'max_depth': 5, 'min_samples_split': 6, 'ccp_alpha'=0.1}
```

Run the script again: *$ python neptune-c.py*

19. **Run 10**: To avoid overfitting change the ***ccp_alpha*** and ***max_depth*** to ***0.01*** and ***10*** respectively.
    Run the script again: *$ python neptune-c.py*

20. **Run 11**: Now we will add more features to train the model. Add '**s3**' in feature selection as shown below:
    ```
    df = df[['sg5', 'sex', 'bmi', 's3']]
    ```
    Run the script again: *$ python neptune-c.py*

21. **Run 12**: Let's try to improve the model. The default value of max_features is None. Add the following to the parameter dictionary in line 42:
    Copy this:
    'max_features': 'log2'

    Run the file again: *$ python neptune-c.py*

22. **Run 13**: Change the test_size back to 0.20.

    Run the file again: *$ python neptune-c.py*

Now return to the experiment's questionnaire