Problem Statement (Medium)

This is a regression task that will predict the percentage of marks that a student is expected to score based upon the number of hours they studied. This is a simple linear regression task as it involves just two variables. Please follow the steps and capture the results on each step.

- **Importing Libraries**
  To import necessary libraries for the first task, so execute the following import statements:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

- **Use Dataset**
  Download csv file (https://drive.google.com/file/d/1nQ837oT0eMFnNT5ncZRJSQhA19_R9YdT) and use dataset from current directory of the jupyter notebook. The following command imports the CSV dataset using pandas:

```
dataset = pd.read_csv('student-scores.csv')
dataset.shape
dataset.head()
dataset.describe()
```

- **Plot data points**

```
dataset.plot(x='Hours', y='Scores', style='o')
plt.title('Hours vs Percentage')
plt.xlabel('Hours Studied')
plt.ylabel('Percentage Score')
plt.show()
```

- **Preparing the Data**
  The next step is to divide the data into "attributes" and "labels". Attributes are the independent variables while labels are dependent variables whose values are to be predicted. To extract the attributes and labels, execute the following script:

```
X = dataset.iloc[:, :-1].values
Y = dataset.iloc[:, 1].values
```

The next step is to split this data into training and test sets into 80:20 ratio.

```
trainX, testX, trainY, testY = train_test_split(X, Y, test_size=0.2,
random_state=0)
```

- **Training the Algorithm**
  Split data into training and testing sets and now is finally the time to train the algorithm.

```
regressor = LinearRegression()
regressor.fit(trainX, trainY)
```

- **Retrieve the intercept**

```
print(regressor.intercept_)
```

- **Retrieve the slope** (coefficient of x)

```
print(regressor.coef_)
```

- **Making Predictions**
  Now the algorithm is trained, it's time to make some predictions.

```
predY = regressor.predict(testX)
df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
df
```

- **Evaluating the Algorithm**
  The final step is to evaluate the performance of the algorithm. This step is particularly important to compare how well different algorithms perform on a particular dataset. For regression algorithms, three evaluation metrics are commonly used:
  - **Mean Absolute Error (MAE)** is the mean of the absolute value of the errors.
  - **Mean Squared Error (MSE)** is the mean of the squared errors.
  - **Root Mean Squared Error (RMSE)** is the square root of the mean of the squared errors.

```
print('Mean Absolute Error:', metrics.mean_absolute_error(testY, predY))
print('Mean Squared Error:', metrics.mean_squared_error(testY, predY))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(testY,
predY)))
```