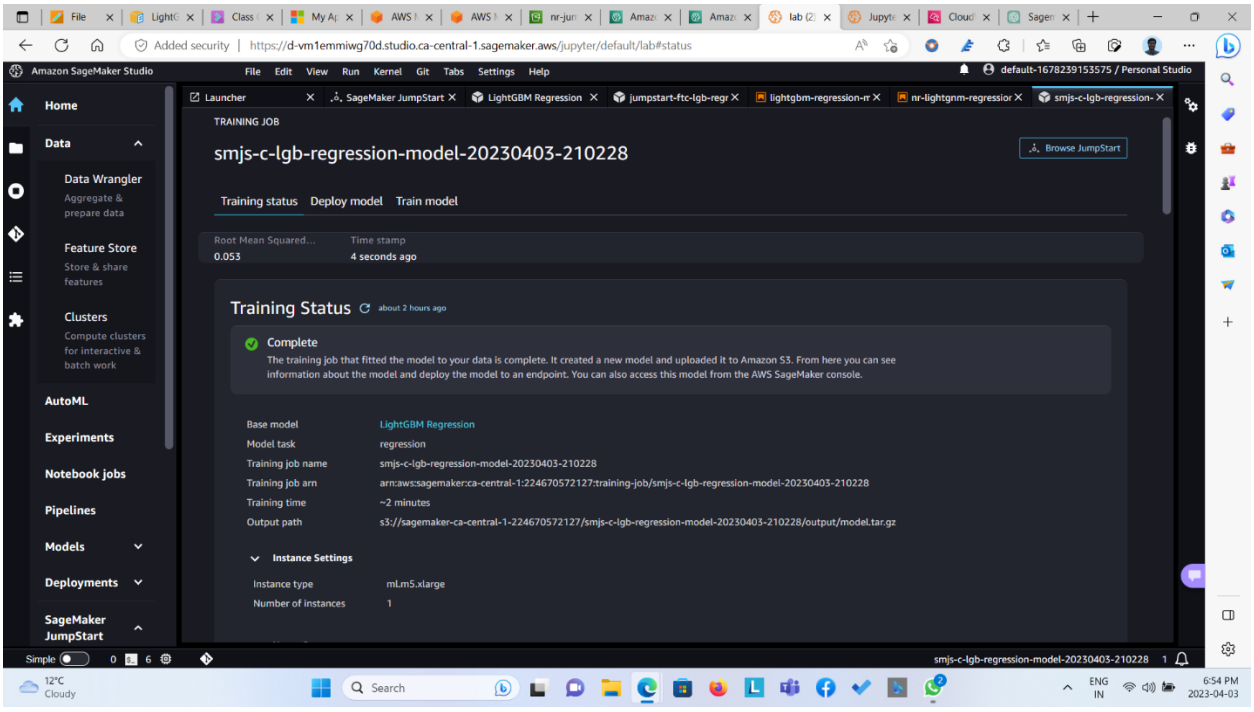
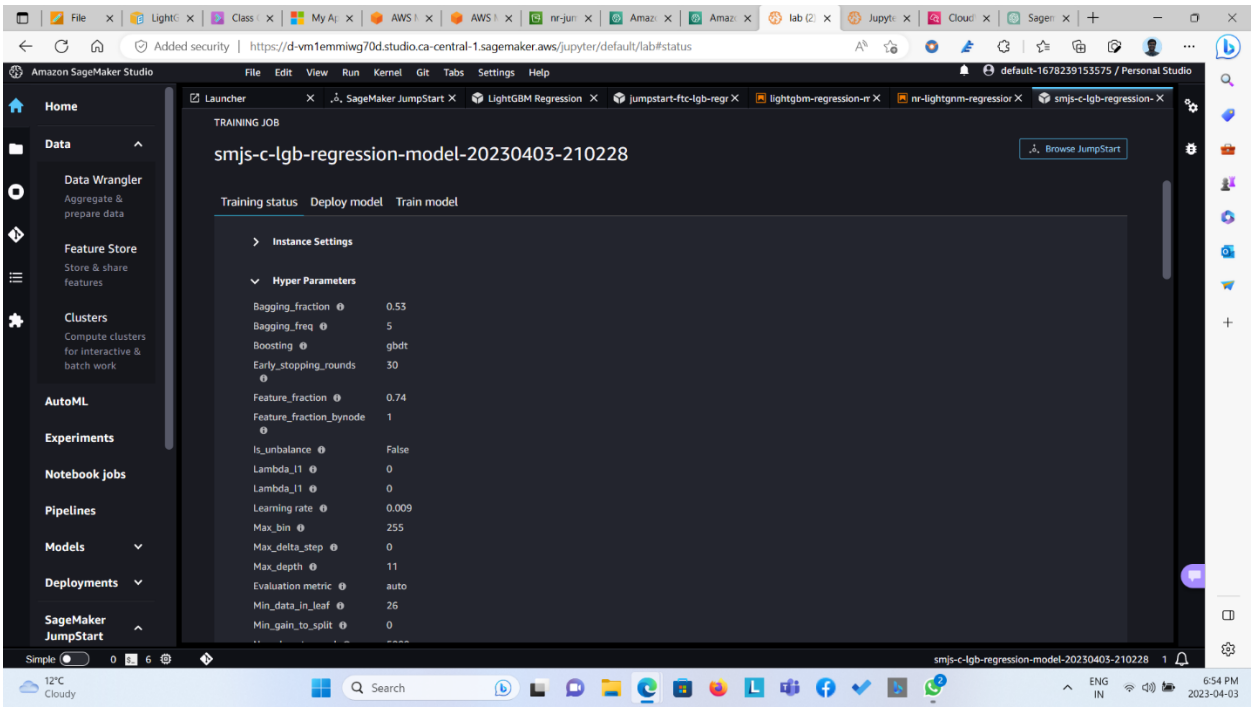
Assignment

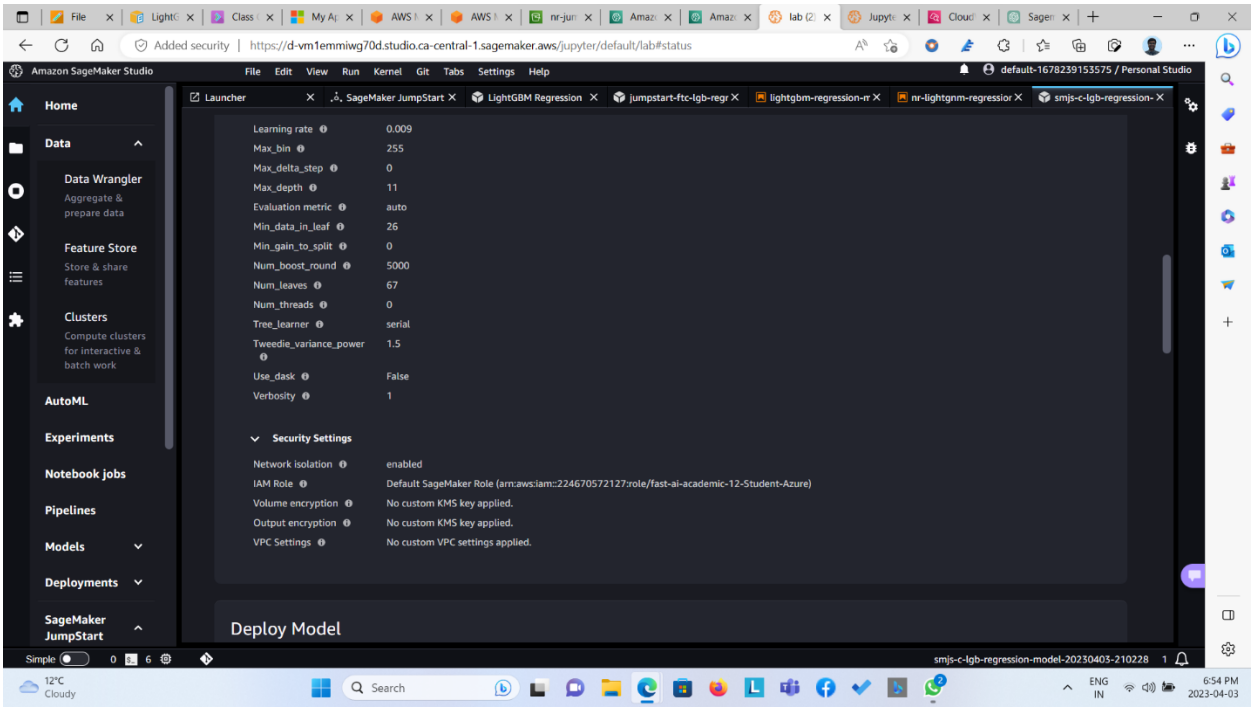
- The training job you used , we started based on a pre-trained model with Abalone data
- Did that really make a difference? What if you use the data I gave you and start a new training from scratch and use LighGBM algorithm to train a new mode? (https://docs.aws.amazon.com/sagemaker/latest/dg/lightgbm.html)
- You need to write code (not console) in the notebook to do this assignment
- Use the same hyperparameters you used during the class in JS to train from scratch and compare at least the following items:
  - Training time
  - Quality of model against validation data
  - Residual values
- Write a report to explain the steps you took and what you learned out of this activity
- Submit the completed notebook and report to blackboard. The notebook should have all the cells run already and I want to see the result of each cell
- You just open the file in the presentation day and explain what happens in each cell. You do not run those cells in the presentation day. We just talk about results and reports.

Comparison

After using the same hyper parameters and re-running a training from scratch by using LightGBM algorithm, it was seen that the both pretrained model and the model trained on notebook takes approximately same time. The pretrained model takes equivalent to 2 minutes as shown below –







Whereas the model trained on notebook also takes nearly 101 training seconds which is equivalent to 2 minutes.

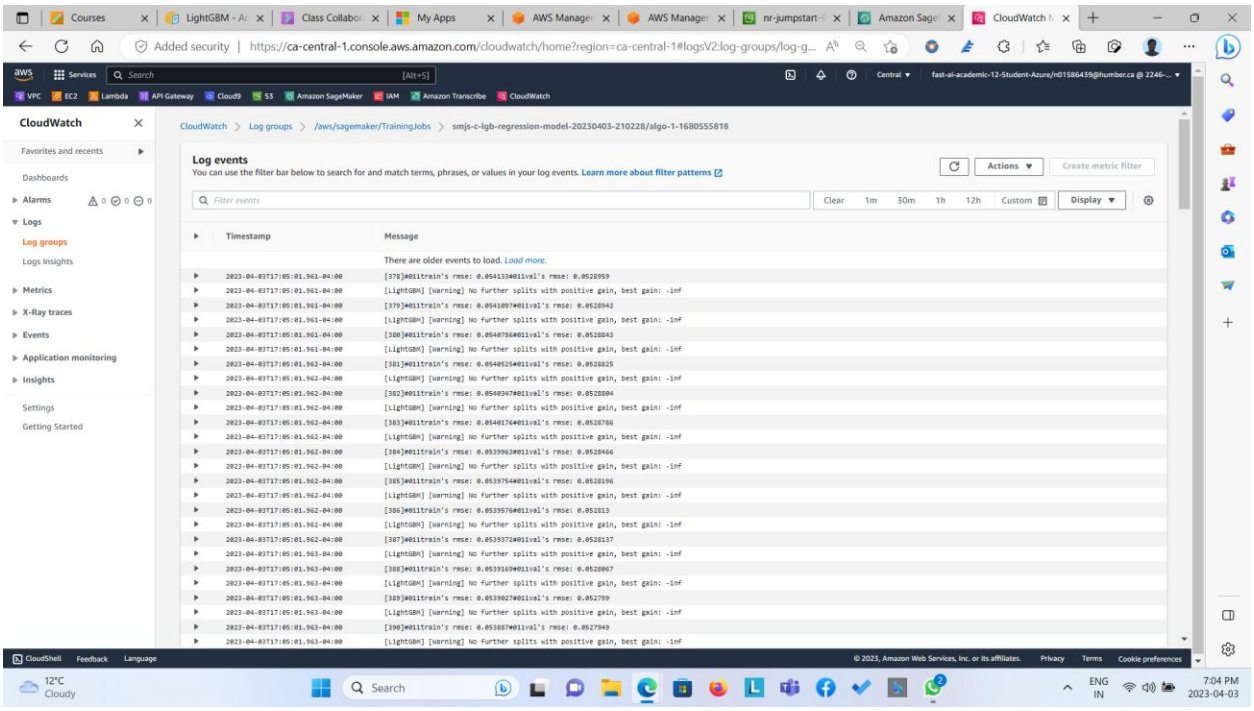
For algorithm specific hyper-parameters, we start by fetching python dictionary of the training hyper-parameters that the algorithm accepts with their default values.

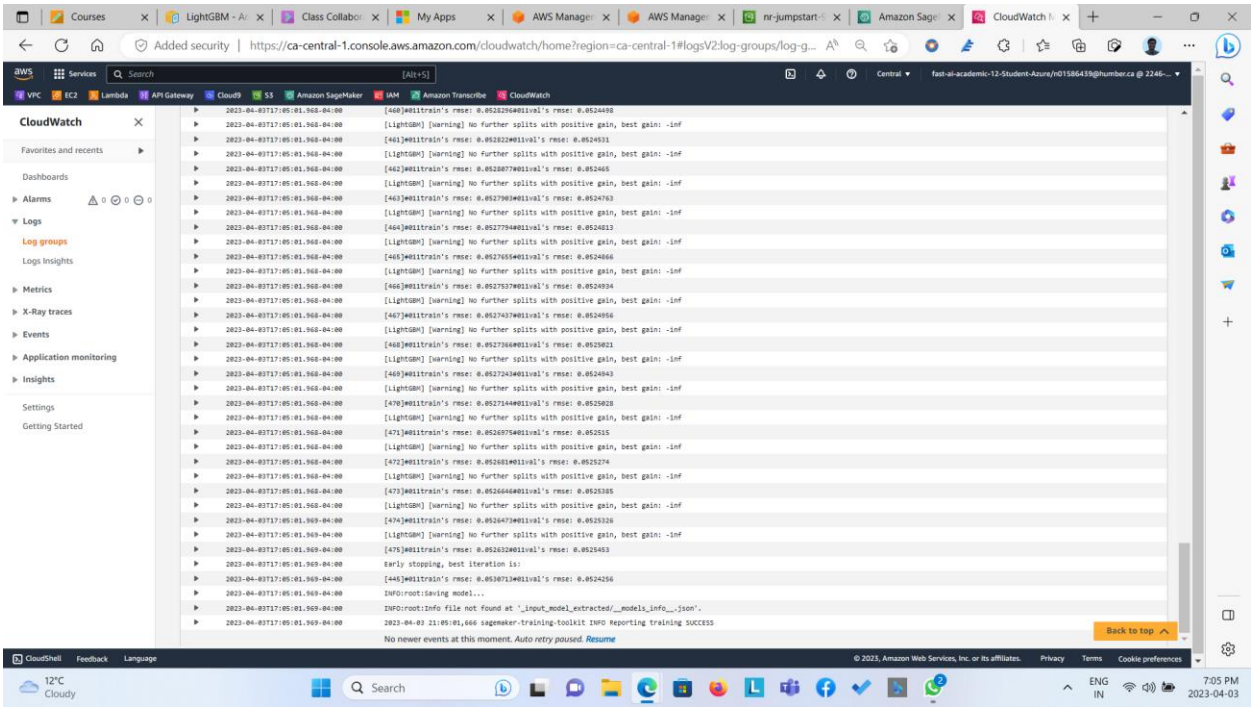
In [8]:
```
from sagemaker import hyperparameters

# Retrieve the default hyper-parameters for fine-tuning the model
hyperparameters = hyperparameters.retrieve_default(
    model_id=train_model_id, model_version=train_model_version
)
print(hyperparameters)
```

```
{'num_boost_round': '5000', 'early_stopping_rounds': '30', 'metric': 'auto', 'learning_rate': '0.009', 'num_leaves': '67', 'feature_fraction': '0.74', 'bagging_fraction': '0.53', 'bagging_fre
q': '5', 'max_depth': '11', 'min_data_in_leaf': '26', 'max_delta_step': '0.0', 'lambda_l1': '0.0', 'lambda_l2': '0.0', 'boosting': 'gbdt', 'min_gain_to_split': '0.0', 'tree_learner': 'serial',
'feature_fraction_bynode': '1.0', 'is_unbalance': 'False', 'max_bin': '255', 'tweedie_variance_power': '1.5', 'num_threads': '0', 'verbosity': '1', 'use_dask': 'False'}
```

```
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[473]#011train's rmse: 0.0526646#011val's rmse: 0.0525385
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[474]#011train's rmse: 0.0526473#011val's rmse: 0.0525326
[LightGBM] [Warning] No further splits with positive gain, best gain: -inf
[475]#011train's rmse: 0.052632#011val's rmse: 0.0525453
Early stopping, best iteration is:
[445]#011train's rmse: 0.0530713#011val's rmse: 0.0524256
INFO:root:Saving model...
INFO:root:Info file not found at '_input_model_extracted/__models_info__.json'.
2023-04-03 22:27:53,836 sagemaker-training-toolkit INFO     Reporting training SUCCESS

2023-04-03 22:28:38 Uploading - Uploading generated training model
2023-04-03 22:28:38 Completed - Training job completed
Training seconds: 101
Billable seconds: 101
```

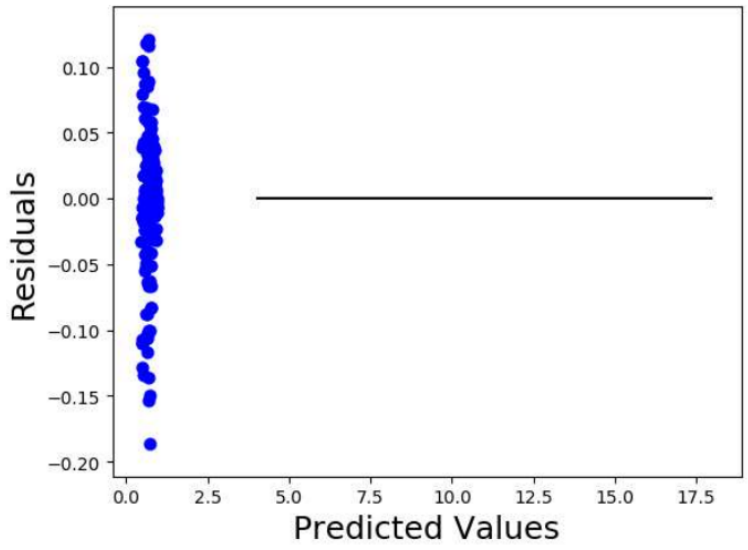
When we compare with the quality of model with respect to the validation data as well as with the residual values, it is noticed that with pretrained model, the best training rmse value is 0.0530 and best validation rmse is 0.0524. The pretrained model's R2 Score is 0.83478, mean squared error is 0.00327 and mean absolute error is 0.04144, which is as shown below.


CloudWatch
Log events
You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns
Timestamp
Message
There are older events to load. Load more.
No newer events at this moment. Auto retry paused. Resume
Back to top


CloudWatch
Log events
You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns
Timestamp
Message
There are older events to load. Load more.
No newer events at this moment. Auto retry paused. Resume
Back to top

In [6]:
```
# Visualization: a residual plot to compare the model predictions and ground truth targets. For each example, the residual value
# is the subtraction between the prediction and ground truth target.
# We can see that the points in the residual plot are randomly dispersed around the horizontal axis y = 0,
# which indicates the fitted regression model is appropriate for the ABALONE data

residuals =  ground_truth_label.values[:, 0] - model_predictions
plt.scatter(model_predictions, residuals, color="blue", s=40)
plt.hlines(y=0, xmin=4, xmax=18)
```

```
plt.xlabel('Predicted Values', fontsize=18)
plt.ylabel('Residuals', fontsize=18)
plt.show()
```

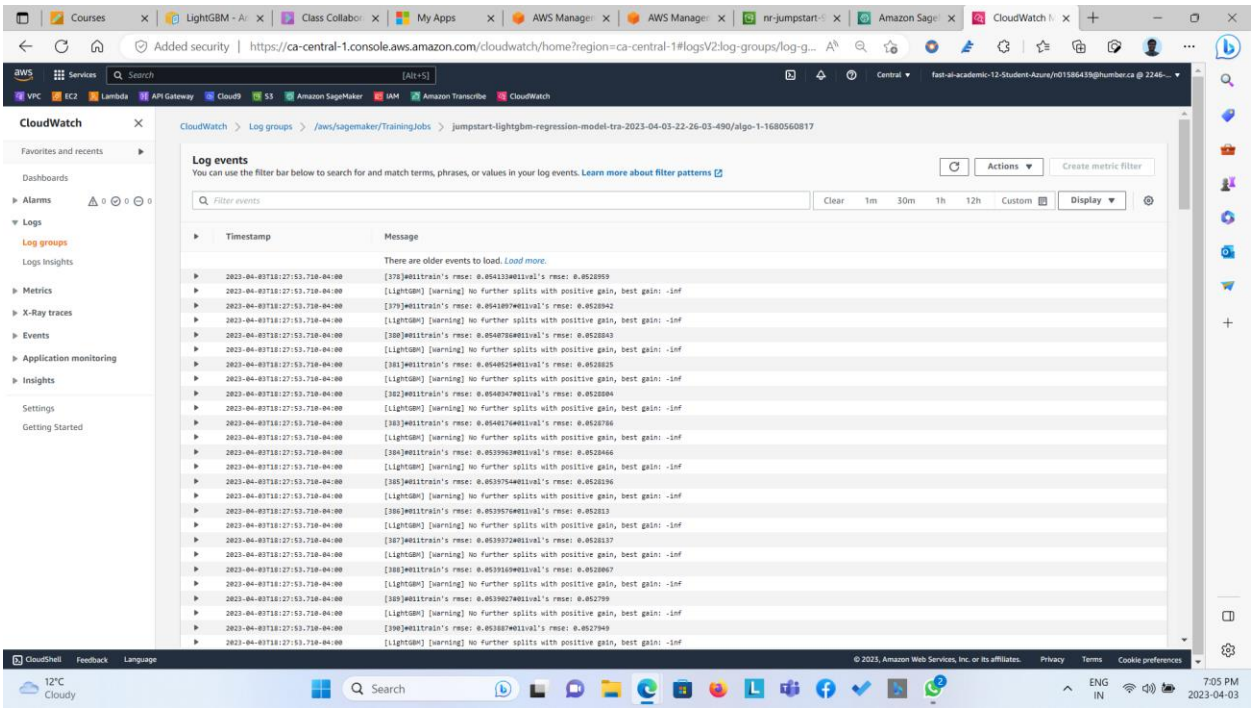


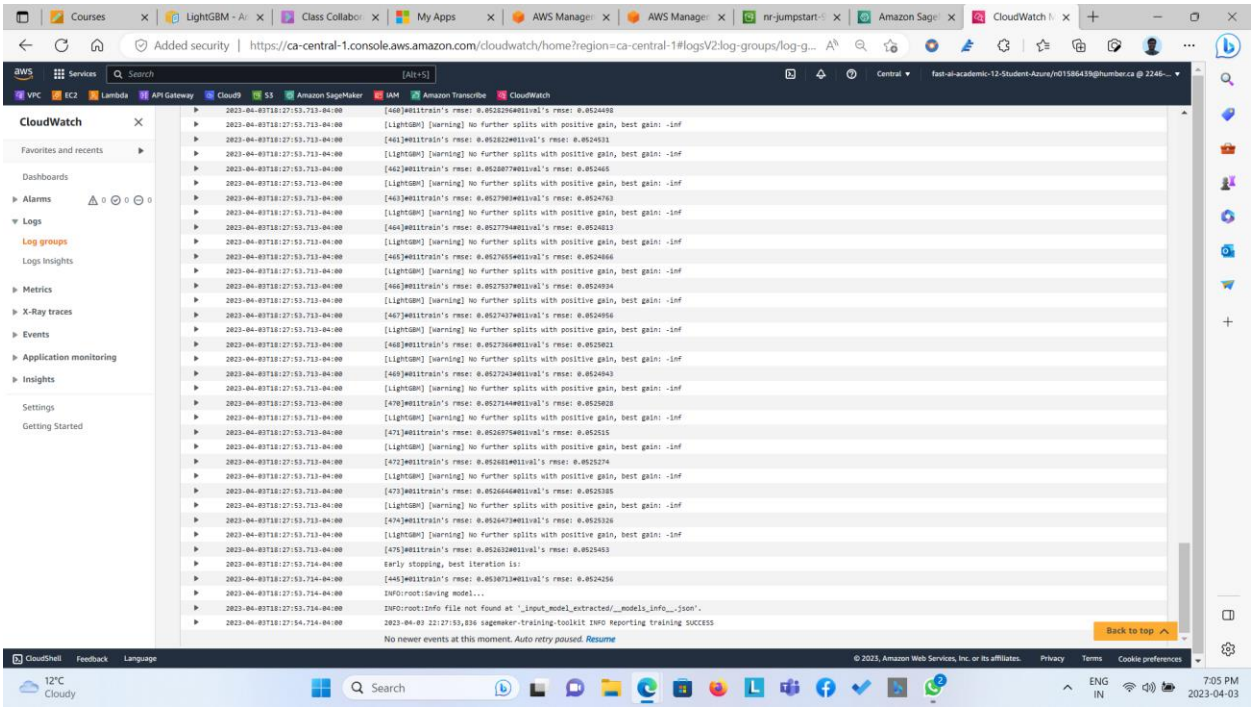
In [7]:
```
# Evaluate the model predictions quantitatively.
eval_r2_score = r2_score(ground_truth_label.values, model_predictions)
eval_mse_score = mean_squared_error(ground_truth_label.values, model_predictions)
eval_mae_score = mean_absolute_error(ground_truth_label.values, model_predictions)
print (
    f"{bold}Evaluation result on test data{unbold}:{newline}"
    f"{bold}{r2_score.__name__}{unbold}: {eval_r2_score}{newline}"
    f"{bold}{mean_squared_error.__name__}{unbold}: {eval_mse_score}{newline}"
    f"{bold}{mean_absolute_error.__name__}{unbold}: {eval_mae_score}{newline}"
)
```

```
Evaluation result on test data:
r2_score: 0.8347803463244985
mean_squared_error: 0.0032756969897500972
mean_absolute_error: 0.041440139984582014
```

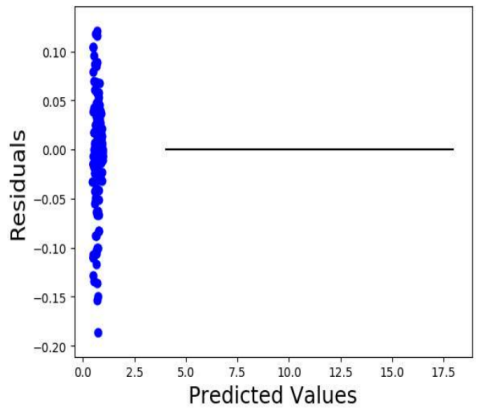
When we compare with the quality of model with respect to the validation data as well as with the residual values, it is noticed that with model trained on notebook gives identical values with respect to pretrained model. The best training rmse value is 0.05307 and best validation rmse is 0.05242. The pretrained model's R2 Score is 0.83478, mean squared error is 0.00327 and mean absolute error is 0.04144, which is as shown below.

```
Early stopping, best iteration is:
[445]#011train's rmse: 0.0530713#011val's rmse: 0.0524256
INFO:root:Saving model...
INFO:root:Info file not found at '_input_model_extracted/__models_info__.json'.
2023-04-03 22:27:53,836 sagemaker-training-toolkit INFO     Reporting training SUCCESS
```





In [22]:
```
# Visualization: a residual plot to compare the model predictions and ground truth targets. For each example, the residual value
# is the subtraction between the prediction and ground truth target.
# We can see that the points in the residual plot are randomly dispersed around the horizontal axis y = 0,
# which indicates the fitted regression model is appropriate for the ABALONE data

residuals = ground_truth_label.values[:, 0] - model_predictions
plt.scatter(model_predictions, residuals, color="blue", s=40)
plt.hlines(y=0, xmin=4, xmax=18)
plt.xlabel("Predicted Values", fontsize=18)
plt.ylabel("Residuals", fontsize=18)
plt.show()
```



In [23]:
```
# Evaluate the model predictions quantitatively.
eval_r2_score = r2_score(ground_truth_label.values, model_predictions)
eval_mse_score = mean_squared_error(ground_truth_label.values, model_predictions)
eval_mae_score = mean_absolute_error(ground_truth_label.values, model_predictions)
print(
    f"{bold}Evaluation result on test data{unbold}:{newline}"
    f"{bold}{r2_score.__name__}{unbold}: {eval_r2_score}{newline}"
    f"{bold}{mean_squared_error.__name__}{unbold}: {eval_mse_score}{newline}"
    f"{bold}{mean_absolute_error.__name__}{unbold}: {eval_mae_score}{newline}"
)
```

```
Evaluation result on test data:
r2_score: 0.8347803463244985
mean_squared_error: 0.0032756969897500972
mean_absolute_error: 0.041440139984582014
```