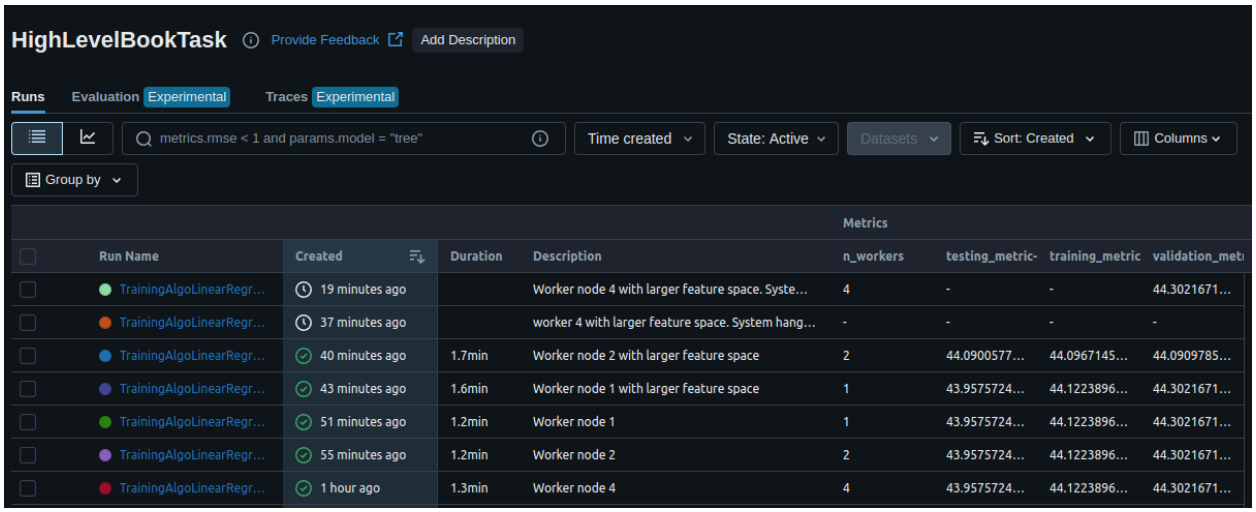
Summary

Overall Experiment Report as tracked on Mlflow



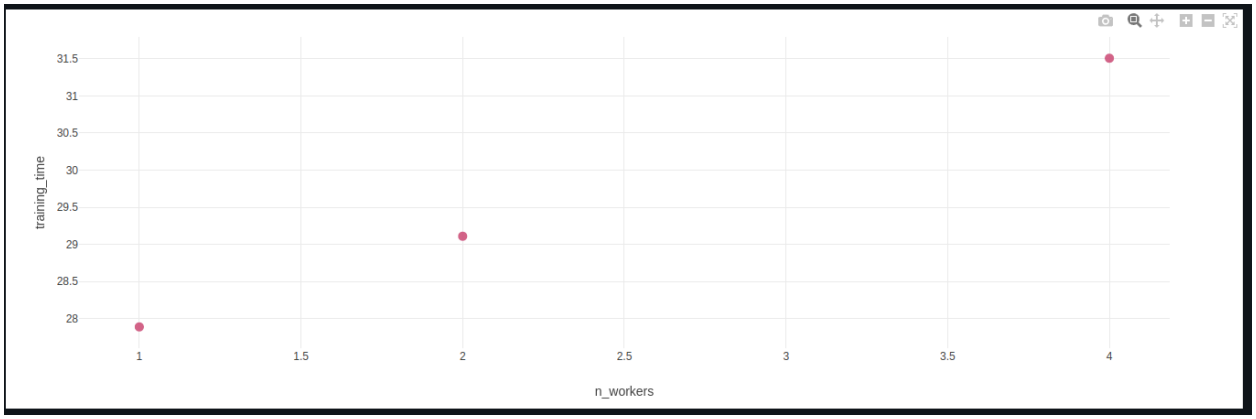
HighLevelBookTask | Provide Feedback | Add Description

Runs | Evaluation Experimental | Traces Experimental

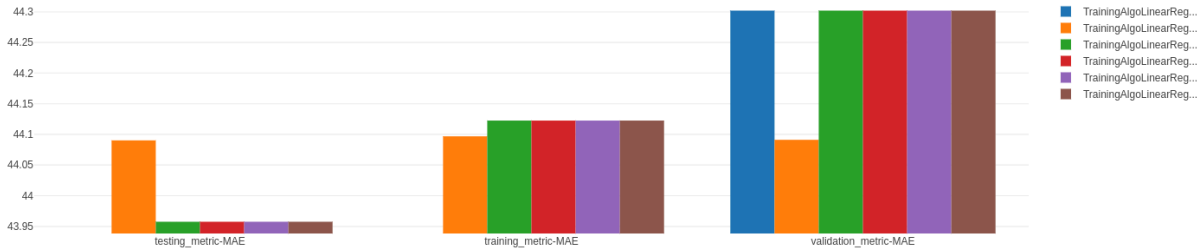
metrics.rmse < 1 and params.model = "tree" | Time created | State: Active | Datasets | Sort: Created | Columns | Group by

					Metrics			
	Run Name	Created	Duration	Description	n_workers	testing_metric-	training_metric	validation_met
	TrainingAlgoLinearRegr...	19 minutes ago		Worker node 4 with larger feature space. Syste...	4	-	-	44.3021671...
	TrainingAlgoLinearRegr...	37 minutes ago		worker 4 with larger feature space. System hang...	-	-	-	-
	TrainingAlgoLinearRegr...	40 minutes ago	1.7min	Worker node 2 with larger feature space	2	44.0900577...	44.0967145...	44.0909785...
	TrainingAlgoLinearRegr...	43 minutes ago	1.6min	Worker node 1 with larger feature space	1	43.9575724...	44.1223896...	44.3021671...
	TrainingAlgoLinearRegr...	51 minutes ago	1.2min	Worker node 1	1	43.9575724...	44.1223896...	44.3021671...
	TrainingAlgoLinearRegr...	55 minutes ago	1.2min	Worker node 2	2	43.9575724...	44.1223896...	44.3021671...
	TrainingAlgoLinearRegr...	1 hour ago	1.3min	Worker node 4	4	43.9575724...	44.1223896...	44.3021671...

Training Time wrt worker nodes



Validation Score wrt modeling



Engineering

1. Deployed spark cluster on local, documented approach. Shared on github (No previous experience in spark.)
2. Used MlFLow to track experiments. (No previous experience)
3. Developed the app in spark using pyspark. (No previous experience)

Data Science

1. Performed EDA. Shared on github.
2. Could have done better feature engineering.
3. Hyperparameter tuning.

Result:

As can be seen above, the model achieved a good MAE score and generalizes well. This was achieved without much dedicated feature engineering or tuning. But the model fails to capture variance as R2 score is very less. However as claimed in the EDA, the distribution of Impact is very skewed and for such data R2 is not a good metric to evaluate. There is enough literature to support this.

Approach

Listing down the thought process below.
List down the challenges and what I did to tackle them

1. I didn't know about spark. Started from scratch. Followed a few tutorials on how to deploy clusters locally. Completed this in ~5 hours.
2. Data was huge with textual features. I knew upon MLP based data mining, the feature space would explode. So, I decided to develop the app in spark. Again, I followed a few tutorials to learn pyspark. Found it to be similar to sklearn pipelines and grid search routines. Completed in ~2.5 hours.
3. Tried deploying kubeflow for experiment tracking on local. Failed to deploy kubernetes cluster locally. Tried for ~2 hours.
4. Decided to go with Mlflow as it was much easier. ~1 hour
5. Tested all the deployment. Couldn't run the training or data pipeline from jupyter. Coded test pipelines for spark as well as mlflow. ~1.5 hours

6. Started with EDA. I tried finding some structure from data particularly from date, authors, categories and publisher fields. Sharing the EDA notebook in github. ~3 hours
    a. publishedDate was very noisy.
        i. No structure
        ii. Missing entries
        iii. Dates back to 1016 to 2030. Not possible.
        iv. Unwanted chars such as *, ??
    b. Tried extracting the year at least, but in interest of time proceeded ahead.
    c. Tried to find linear relation between other fields and impact only to find no relationship.
7. Designed the data preprocessing and feature engineering pipeline. ~3 hours
8. Started running experiments and here things started jumping off the cliff. ~ 6+ hours and running
    a. Jobs started failing with 4 worker nodes as well.
    b. Increased memory per node from 1GB to 2 GB.
    c. Yet, it fails
    d. Instead of processing all the field at once, started running one field at a time and it worked.
    e. With all the fields and all the records, the training failed.
    f. Trimmed the records. Dropped the missing values. Still fails. Realized that the feature space is exploding.
    g. Trimmed the fields for feature engineering. Removed Title and description, the pipeline runs end to end.
    h. Multiple system crashes and iteration performed.
9. Choose Linear Regression for training the model. Reasons:
    a. Purely engineering decision. Since the jobs were failing, I thought of starting with simplest model. Although I had tried Random Forest as well and it failed. But with reduced feature space it could have trained.
    b. Certainly not a statistical decision, the target and the data doesn't satisfy the linear model assumption.
    c. Still it performs good on the MAE metric. There must be some non linear features in the feature space which is explaining the distribution better. Could not figure out which features and how much.
    d. In the interest of deadline, could not try out other models.

What Next?

Almost every stage from a data science perspective can be re-looked for better performance.

Model performance can be improved by

1. Can try much advance NLP feature extractors. Lack of knowledge of complicated transformer based embedders.
2. Can try parameter tuning, local resource failing but learnt how to do on spark. Very similar to sklearn.
3. Can try other modeling techniques. Ensembles tend to perform better.

Engineering performance can be improved by

4. The code can be more modular.
5. More compute don't hurt. But the worker nodes v/s time graph is completely different to my assumption. Don't know how?