Assignment: Using MLflow for Experiment Tracking and Model Management

Objective: The objective of this task is to introduce you to MLflow for experiment tracking, model management, and reproducibility in machine learning projects for the Sentiment Analysis Project.

Task

1. Integrate MLflow into your existing machine learning projects.
2. Train machine learning models while logging relevant information with MLflow.
3. Demonstrate how to log parameters, metrics, and artifacts using MLflow tracking APIs.
4. Customizing MLflow UI with run names.
5. Demonstrate metric plots.
6. Demonstrate hyperparameter plots.
7. Demonstrate how to register models and manage by tagging them.
8. (**BONUS**) Build a Prefect Workflow and Auto Schedule it. Show the Prefect Dashboard with relevant outputs.

Task 1: Integrate MLflow into your existing machine learning projects.

Auto Logging Reviews Experiment Run Using MLFlow

Step 1 - Import MLFlow and set the experiment name

Step 1 - Import MLFlow and set the experiment name

```
import mlflow
mlflow.set_experiment("reviews_classification_ml_algos")
```

Step 2 - Start the auto logger

mlflow.sklearn.autolog()

 # Initialize the auto logger

 # max_tuning_runs=None will make sure that all the runs are recorded.

 # By default, top 5 runs will be recorded for each experiment

Step 3 - Start the experiment run

with mlflow.start_run() as run:

clf.fit(X_train, y_train)

```
mlflow.sklearn.autolog(max_tuning_runs=None)

with mlflow.start_run() as run:
    %time grid_search.fit(X_train['clean_text_lemma'], y_train_le)
```

Step 4 - # Stop the auto logger

mlflow.sklearn.autolog(disable=True)

```
# Stop the auto logger
mlflow.sklearn.autolog(disable=True)
```

Task 2: Train machine learning models while logging relevant information with MLflow

Random Forest

```
********** random_forest **********
2024/03/28 09:33:28 WARNING mlflow.sklearn: Unrecognized dataset type <class 'pandas.core.series.Series'>. Dataset logging skipped.
Fitting 5 folds for each of 48 candidates, totalling 240 fits
CPU times: total: 15min 3s
Wall time: 21min 3s
2024/03/28 09:54:32 INFO mlflow.utils.autologging_utils: Created MLflow autologging run with ID 'b96e24da48b74943a25ffd7e0a11f9c2', which will track hyperparameters, performance metrics, model artifacts, and lineage information for the current sklearn workflow
2024/03/28 09:54:32 WARNING mlflow.sklearn: Failed to log training dataset information to MLflow Tracking. Reason: 'numpy.ndarray' object has no attribute 'toarray'
Accuracy on Test Data:  0.9111842105263158
F1 Score (Positive Class) on Test Data:  0.9493704795070989
F1 Score (Negative Class) on Test Data:  0.638623326959847
Classification Report:
              precision    recall  f1-score   support

    Negative       0.62      0.65      0.64       255
    Positive       0.95      0.95      0.95      1873

    accuracy                           0.91      2128
   macro avg       0.79      0.80      0.79      2128
weighted avg       0.91      0.91      0.91      2128
```

Logistic Regression

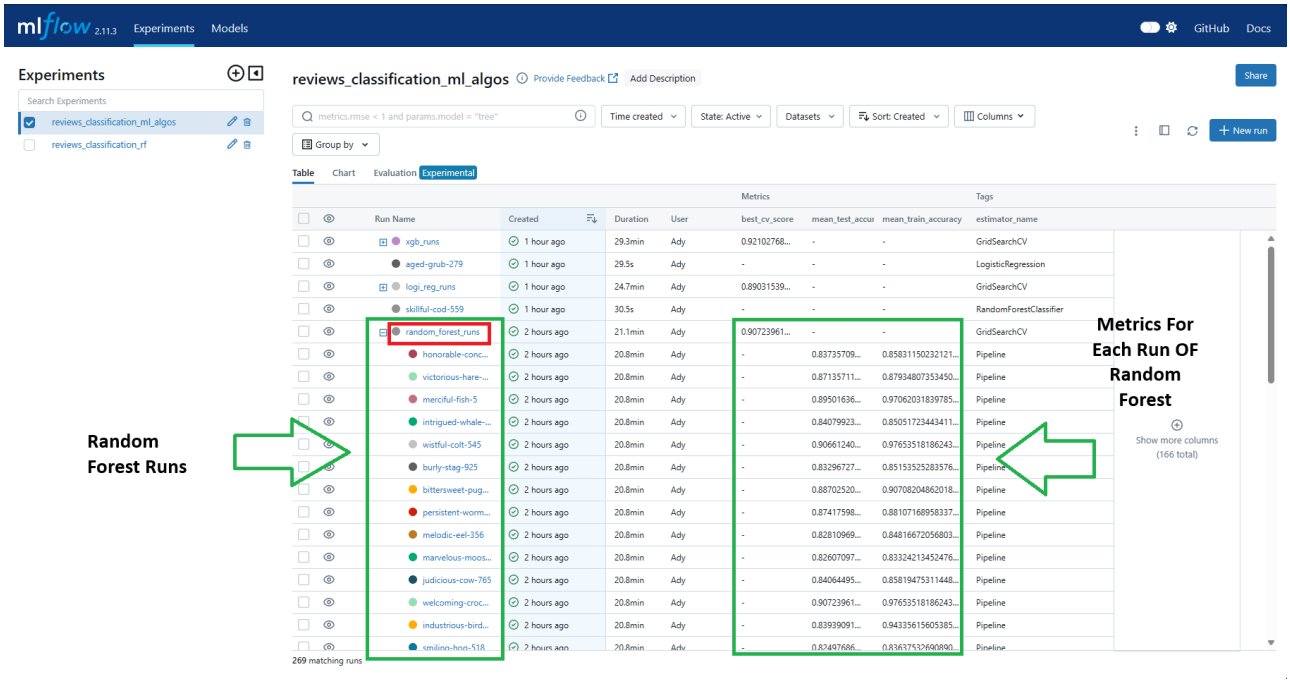
```
********** logistic_regression **********
2024/03/28 09:55:03 WARNING mlflow.sklearn: Unrecognized dataset type <class 'pandas.core.series.Series'>. Dataset logging skipped.
Fitting 5 folds for each of 72 candidates, totalling 360 fits
CPU times: total: 21min 9s
Wall time: 24min 44s
2024/03/28 10:19:49 INFO mlflow.utils.autologging_utils: Created MLflow autologging run with ID '3ed352f0c7714d34828f5988cdb1987c', which will track hyperparameters, performance metrics, model artifacts, and lineage information for the current sklearn workflow
2024/03/28 10:19:49 WARNING mlflow.sklearn: Failed to log training dataset information to MLflow Tracking. Reason: 'numpy.ndarray' object has no attribute 'toarray'
Accuracy on Test Data:  0.8952067669172933
F1 Score (Positive Class) on Test Data:  0.9394186362401521
F1 Score (Negative Class) on Test Data:  0.6121739130434782
Classification Report:
              precision    recall  f1-score   support

    Negative       0.55      0.69      0.61       255
    Positive       0.96      0.92      0.94      1873

    accuracy                           0.90      2128
   macro avg       0.75      0.81      0.78      2128
weighted avg       0.91      0.90      0.90      2128
```

XGBoost

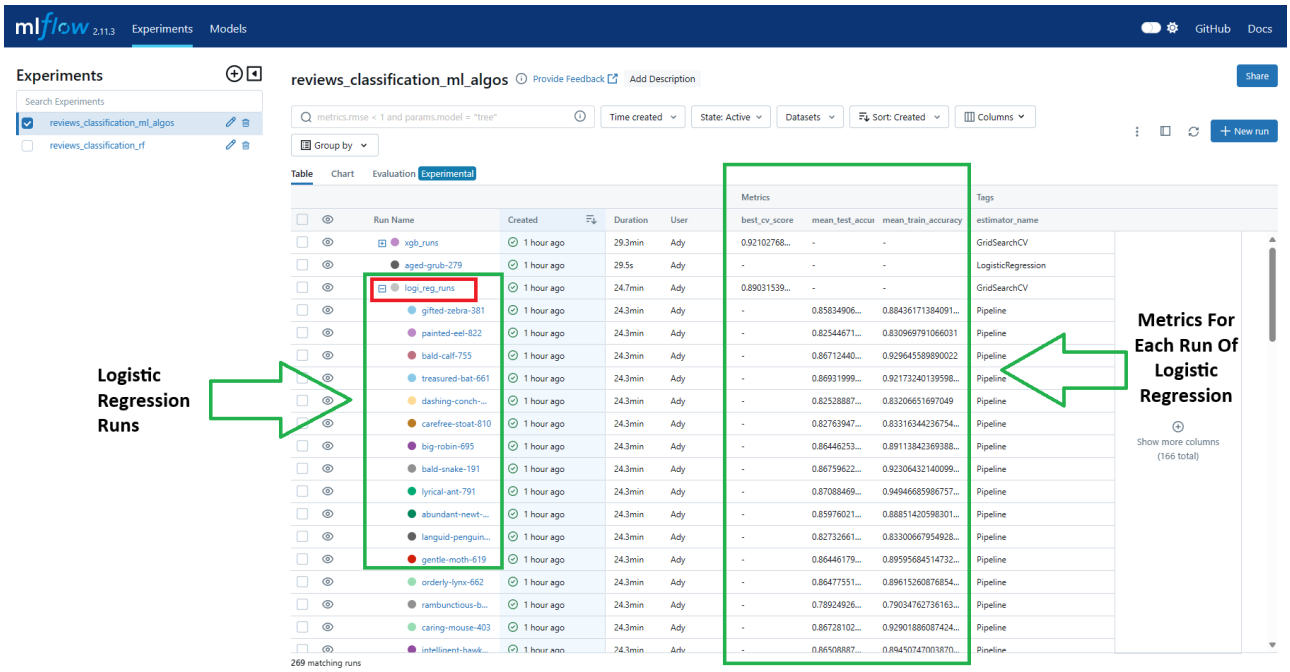
```
********** xgboost **********
2024/03/28 10:20:19 WARNING mlflow.sklearn: Unrecognized dataset type <class 'pandas.core.series.Series'>. Dataset logging skipped.
Fitting 5 folds for each of 144 candidates, totalling 720 fits
CPU times: total: 1h 22min
Wall time: 29min 16s
Accuracy on Test Data:  0.9172932330827067
F1 Score (Positive Class) on Test Data:  0.9534391534391534
F1 Score (Negative Class) on Test Data:  0.6302521008403361
Classification Report:
              precision    recall  f1-score   support

    Negative       0.68      0.59      0.63       255
    Positive       0.94      0.96      0.95      1873

    accuracy                           0.92      2128
   macro avg       0.81      0.78      0.79      2128
weighted avg       0.91      0.92      0.91      2128
```

Task 3: Demonstrate how to log parameters, metrics, and artifacts using MLflow tracking APIs.

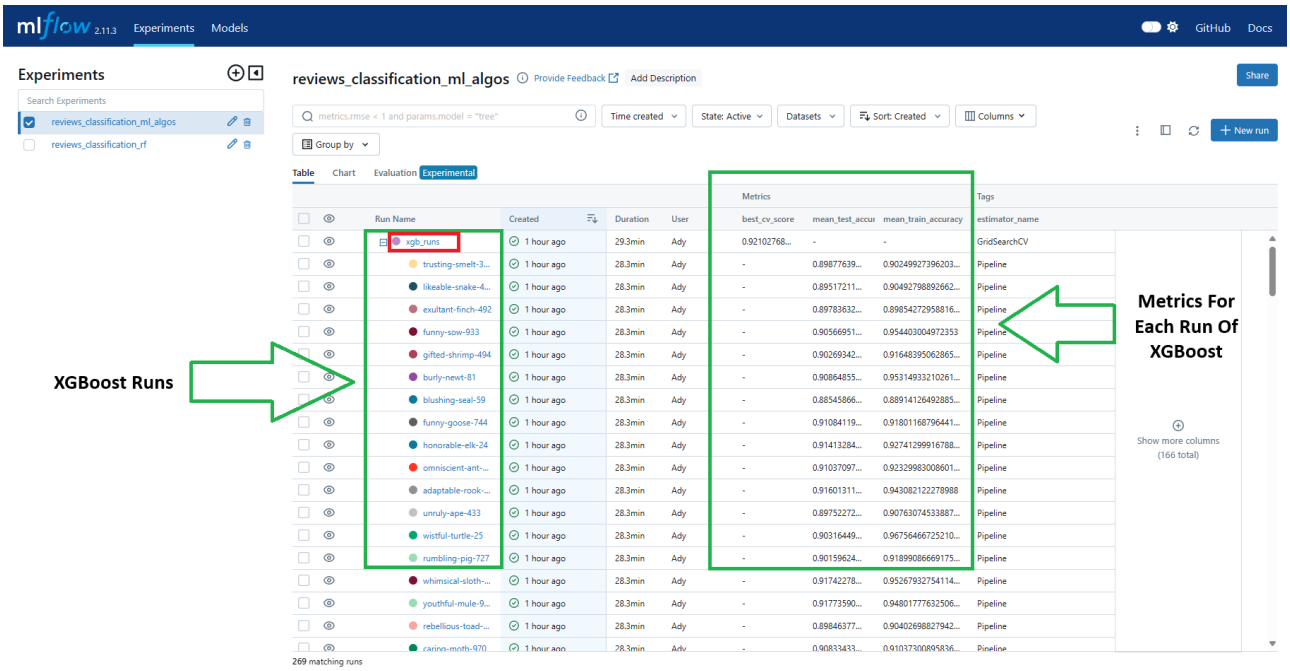
Random Forest



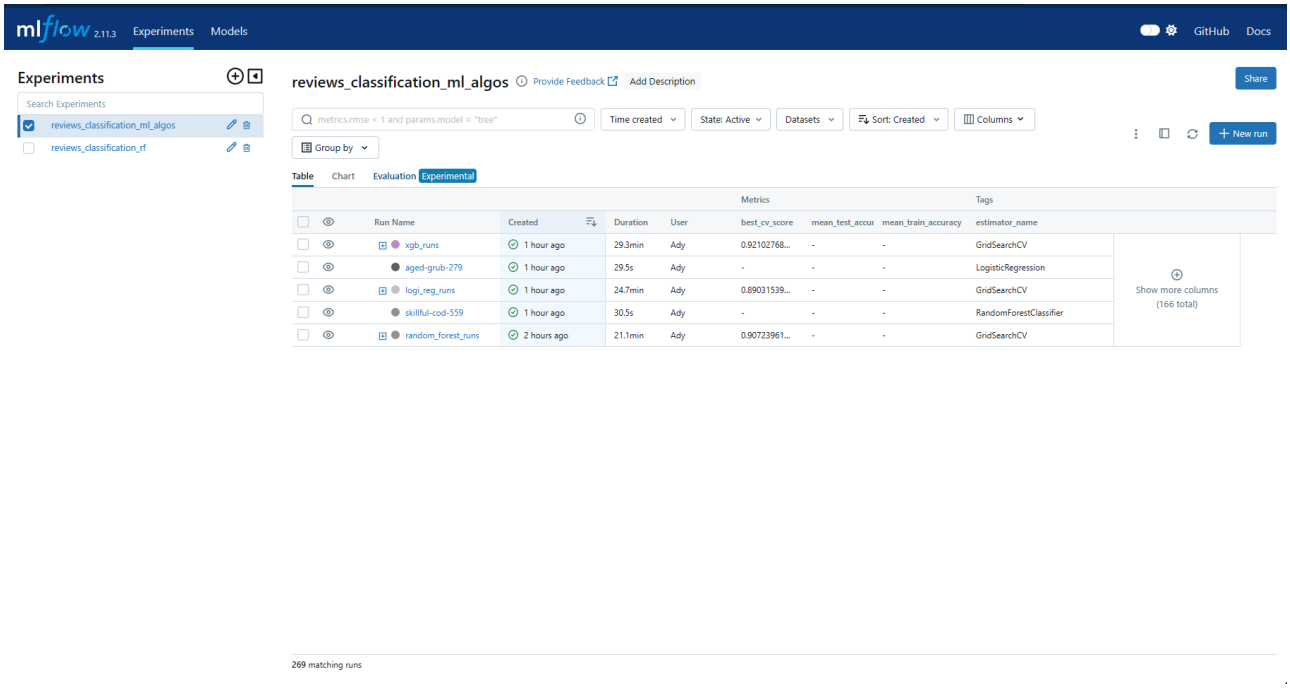
Logistic Regression



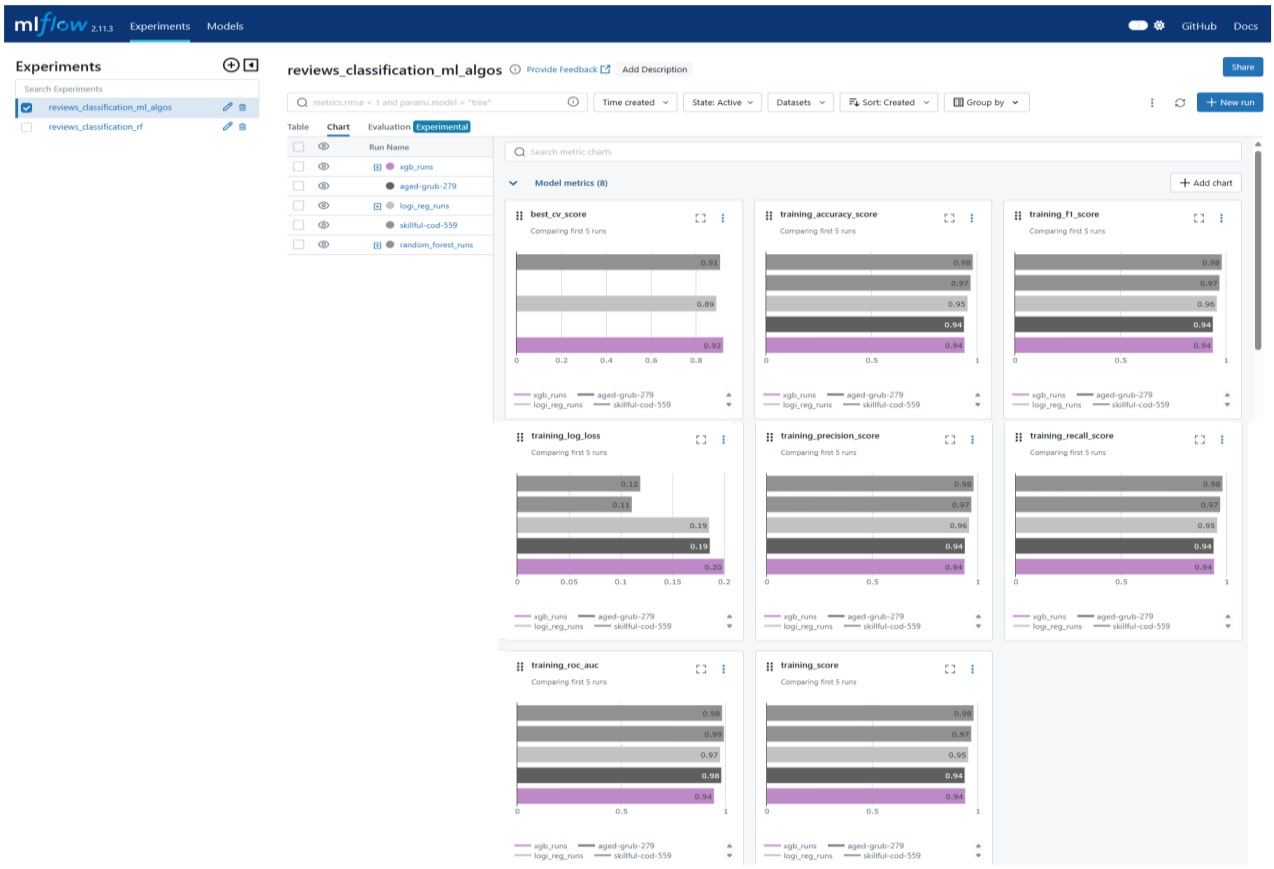
XGBoost



Task 4: Customizing MLflow UI with run names.



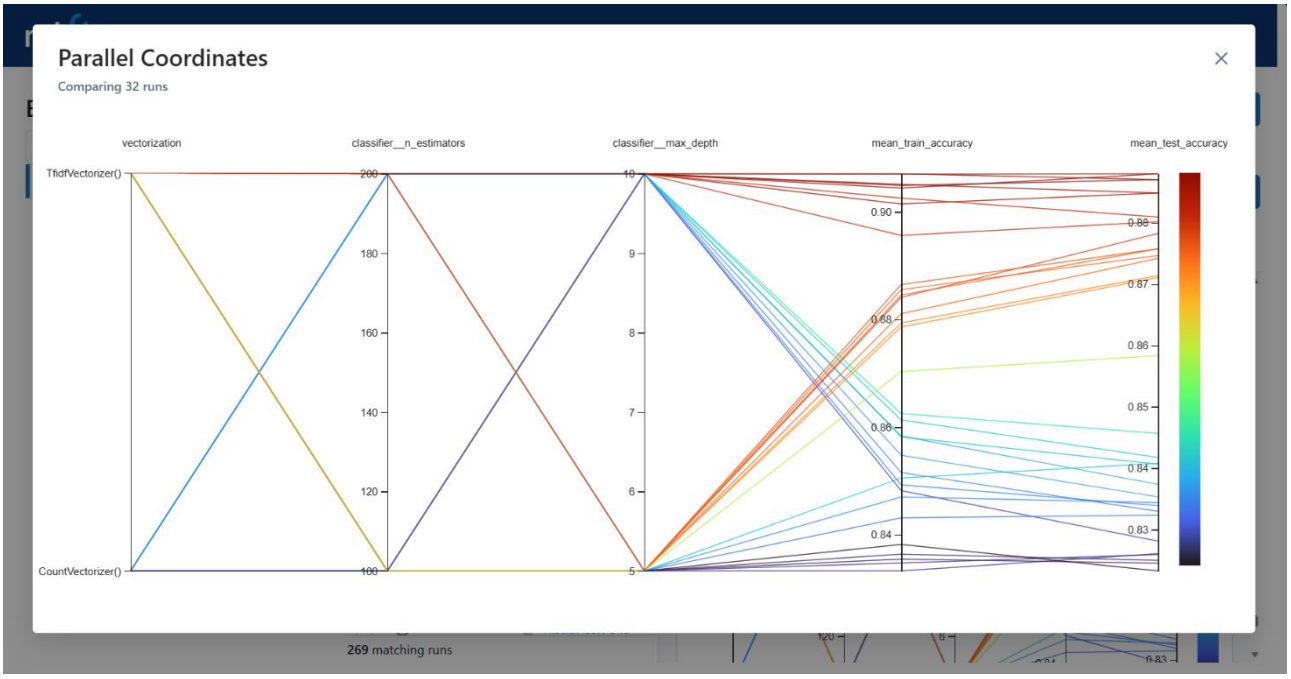
Task 5: Demonstrate metric plots.

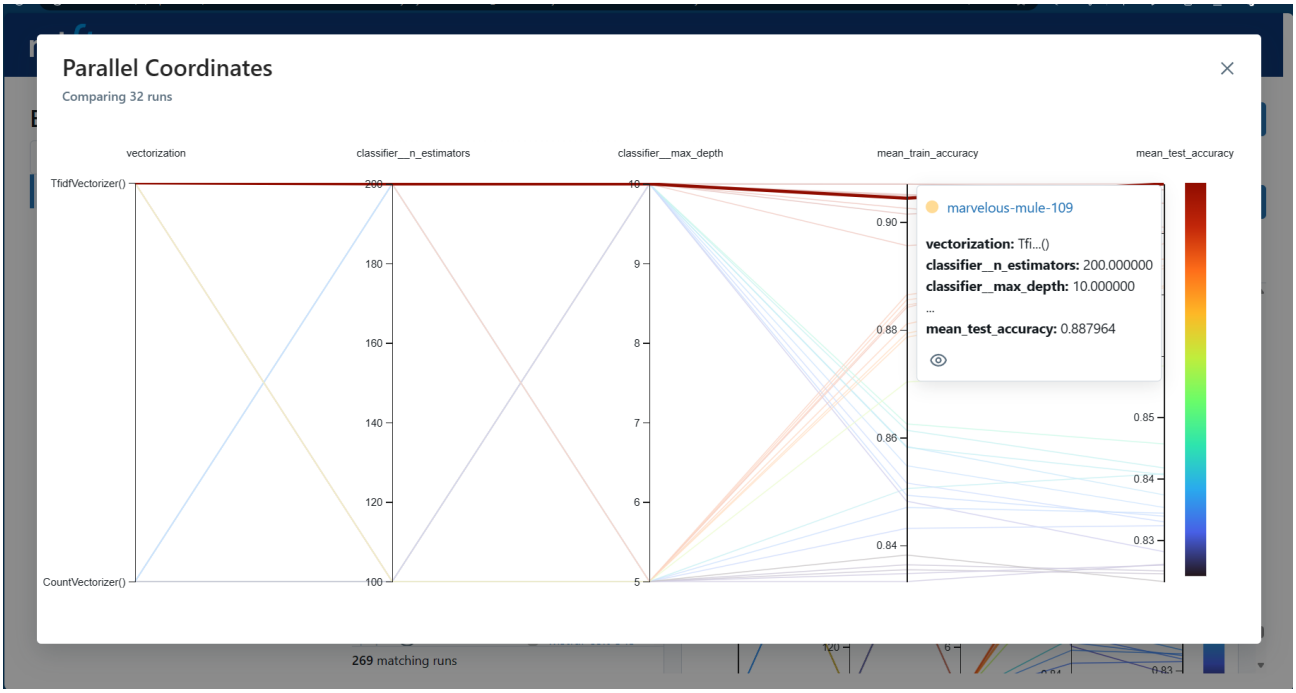


Task 6: Demonstrate hyperparameter plots.

Random Forest

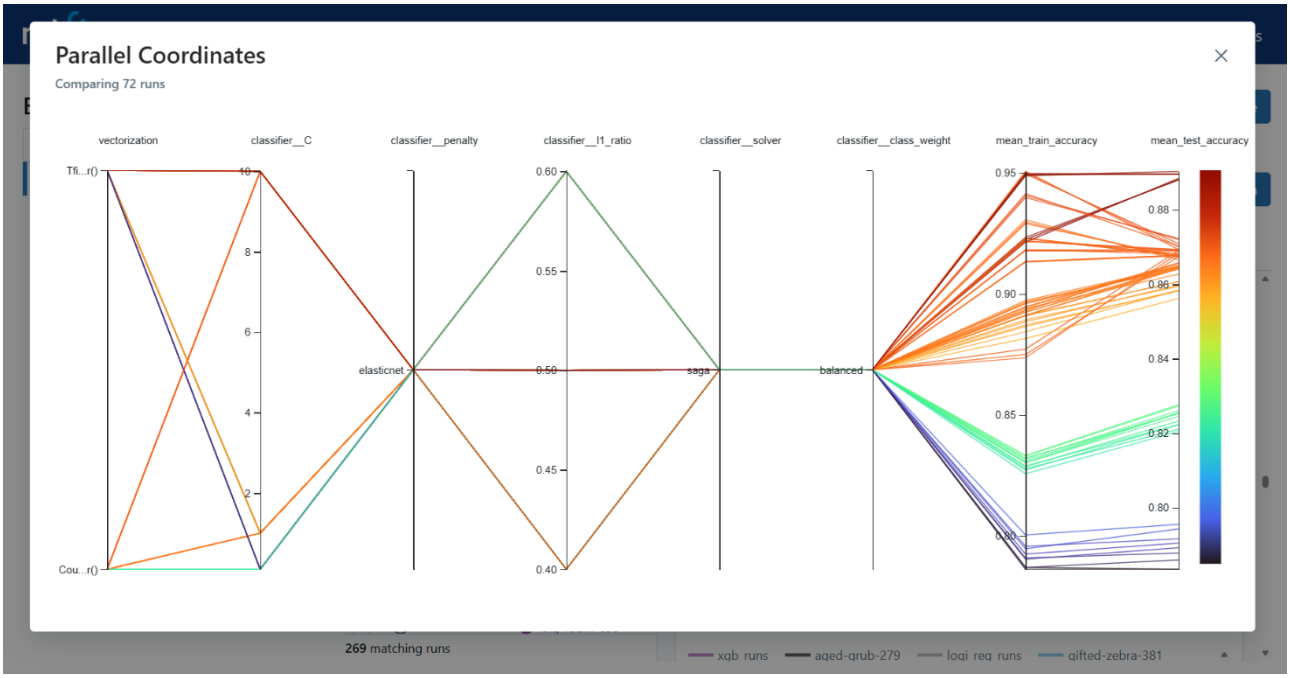
```
random_forest
Pipeline(memory=Memory(location=.cache\joblib),
         steps=[('vectorization', TfidfVectorizer(max_features=5000)),
                ('smote', SMOTE(random_state=42)),
                ('classifier', RandomForestClassifier())])
```

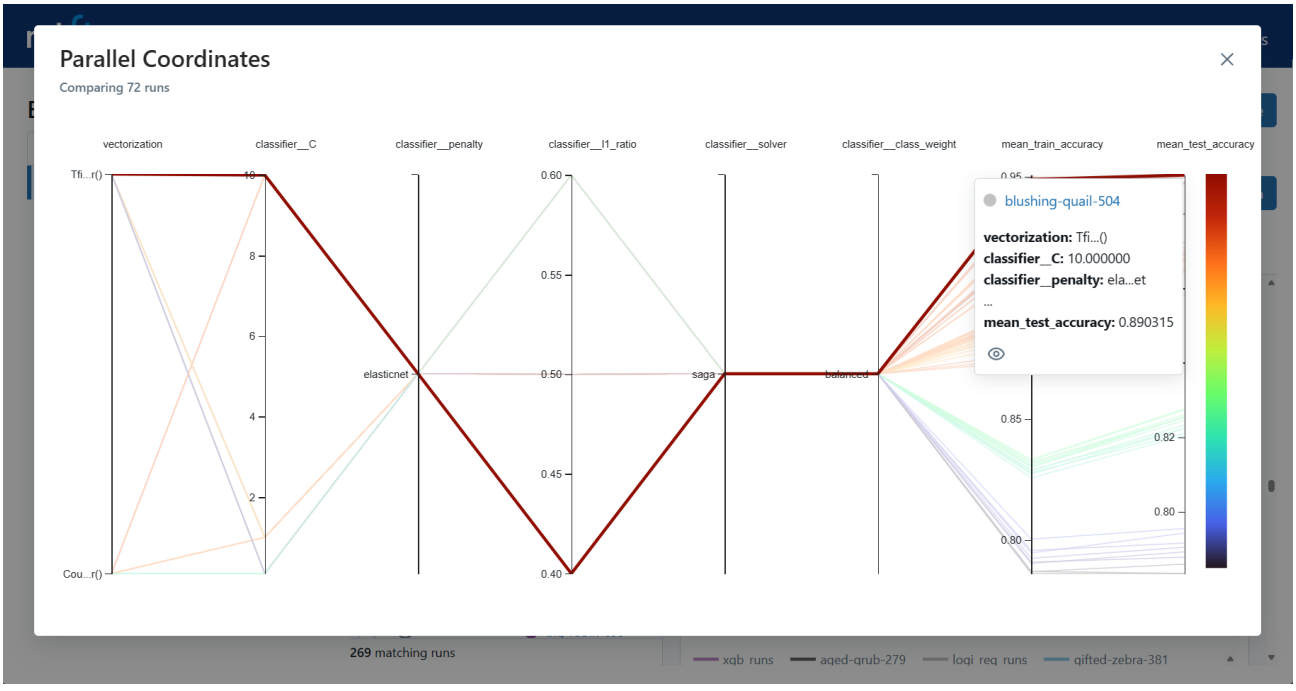




Logistic Regression

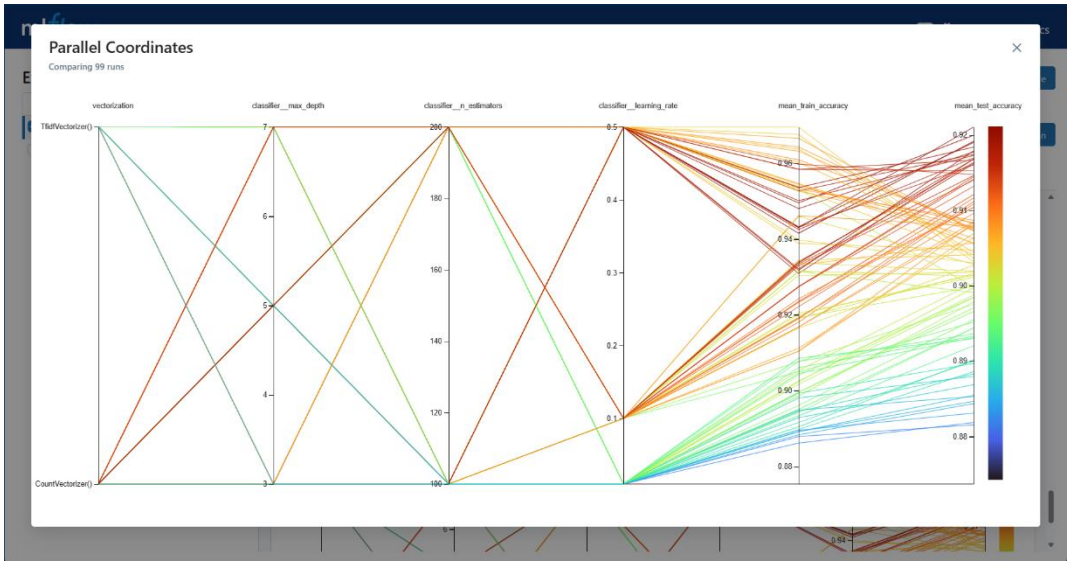
```
logistic_regression
Pipeline(memory=Memory(location=.cache\joblib),
         steps=[('vectorization', TfidfVectorizer(max_features=5000)),
                ('smote', SMOTE(random_state=42)),
                ('classifier',
                 LogisticRegression(C=10, class_weight='balanced', l1_ratio=0.4,
                                    penalty='elasticnet', solver='saga'))])
```

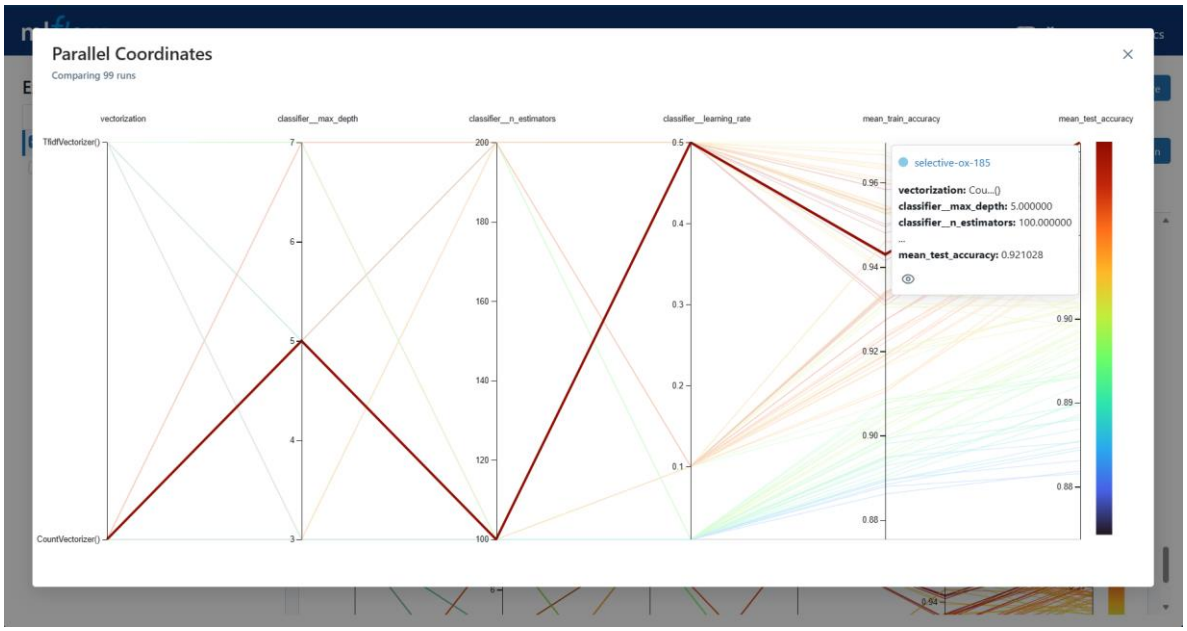




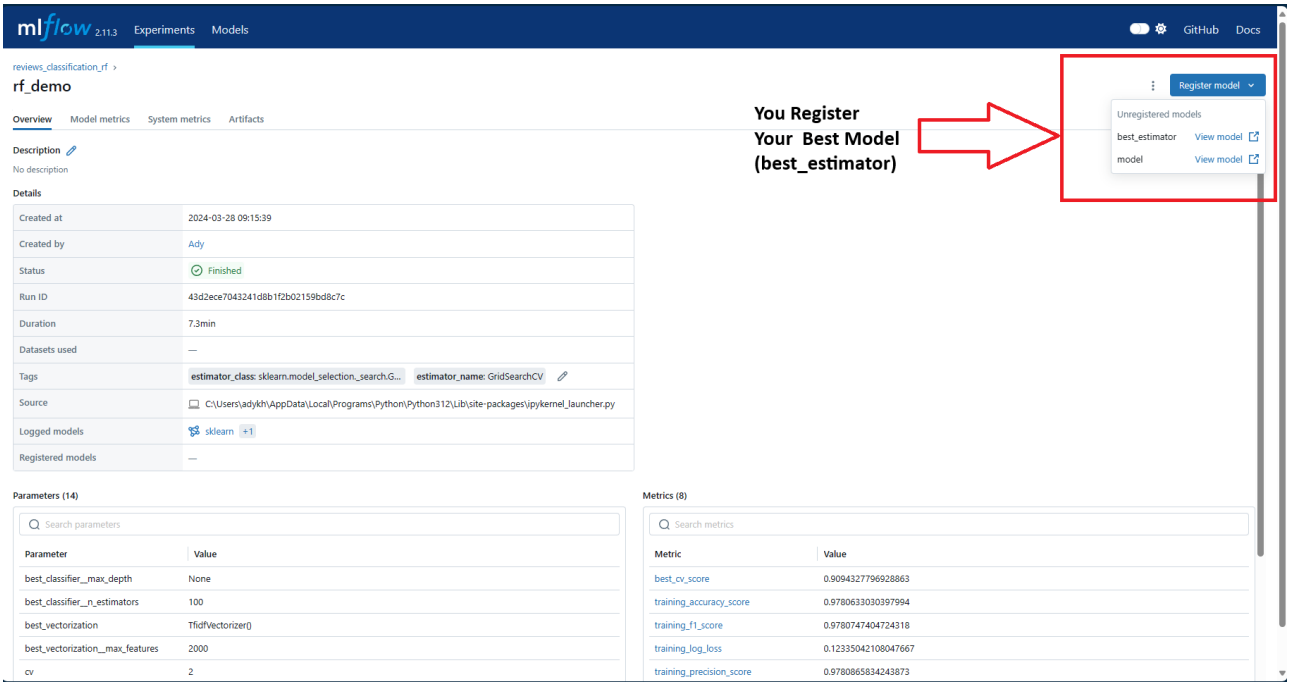
XGBoost

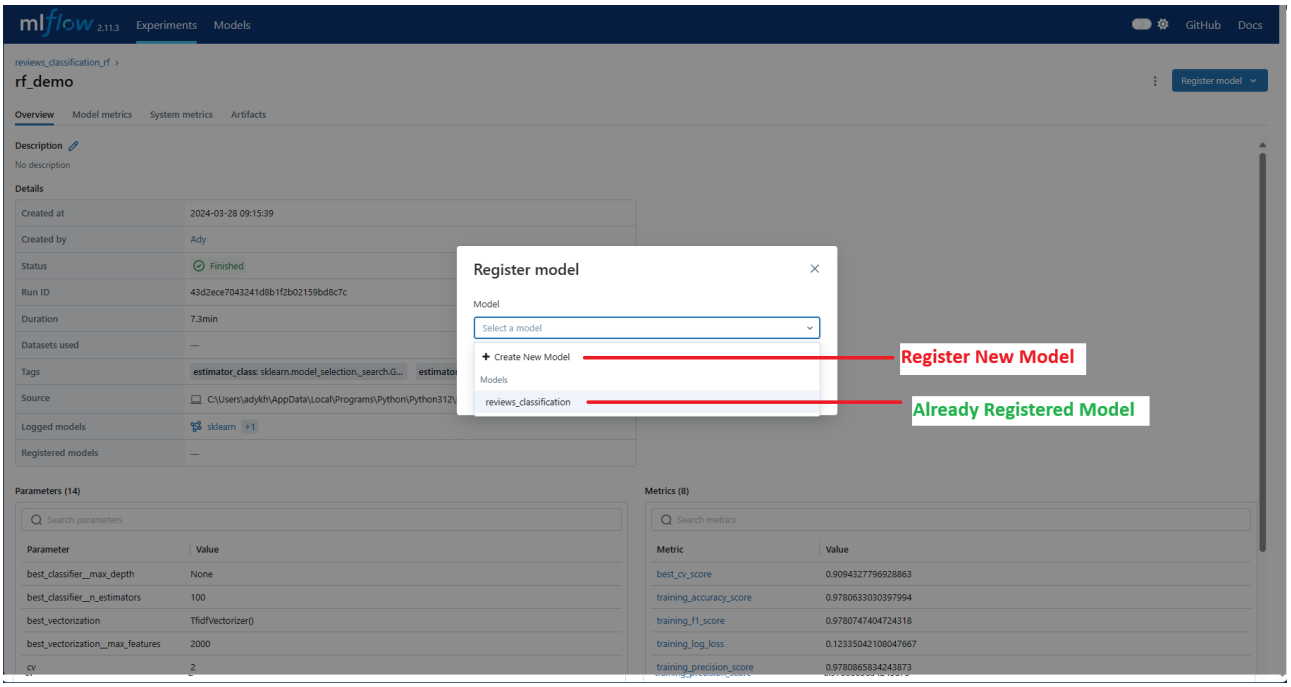
```
xgboost
Pipeline(memory=Memory(location=.cache\joblib),
         steps=[('vectorization', CountVectorizer(max_features=5000)),
                ('smote', SMOTE(random_state=42)),
                ('classifier',
                 XGBClassifier(base_score=None, booster=None, callbacks=None,
                               colsample_bylevel=None, colsample_bynode=None,
                               colsample_bytree=None, device=None,
                               early_stopping_rounds=None,
                               enable_categorical=False, eval_me...
                               feature_types=None, gamma=None, grow_policy=None,
                               importance_type=None,
                               interaction_constraints=None, learning_rate=0.5,
                               max_bin=None, max_cat_threshold=None,
                               max_cat_to_onehot=None, max_delta_step=None,
                               max_depth=5, max_leaves=None,
                               min_child_weight=None, missing=nan,
                               monotone_constraints=None, multi_strategy=None,
                               n_estimators=100, n_jobs=None,
                               num_parallel_tree=None, random_state=None, ...))])
```

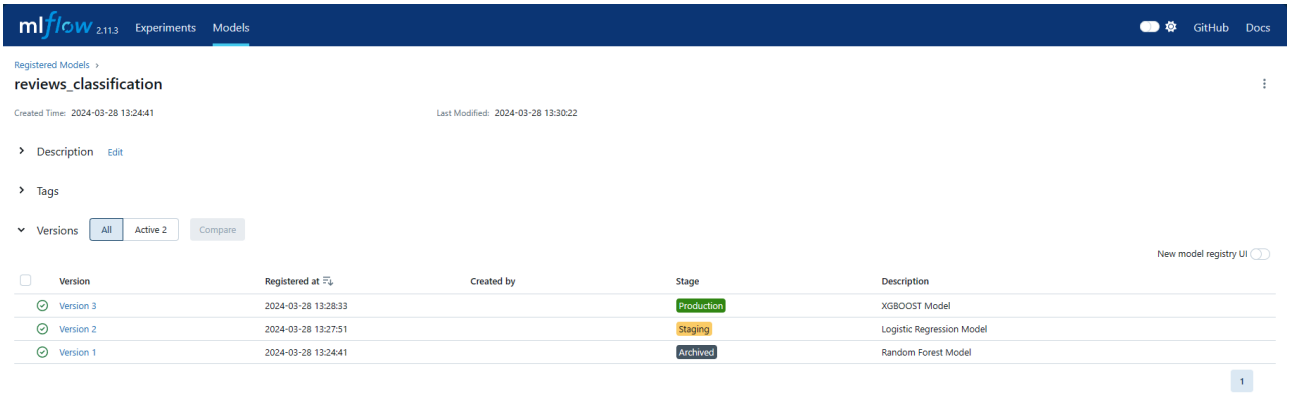




Task 7: Demonstrate how to register models and manage by tagging them.

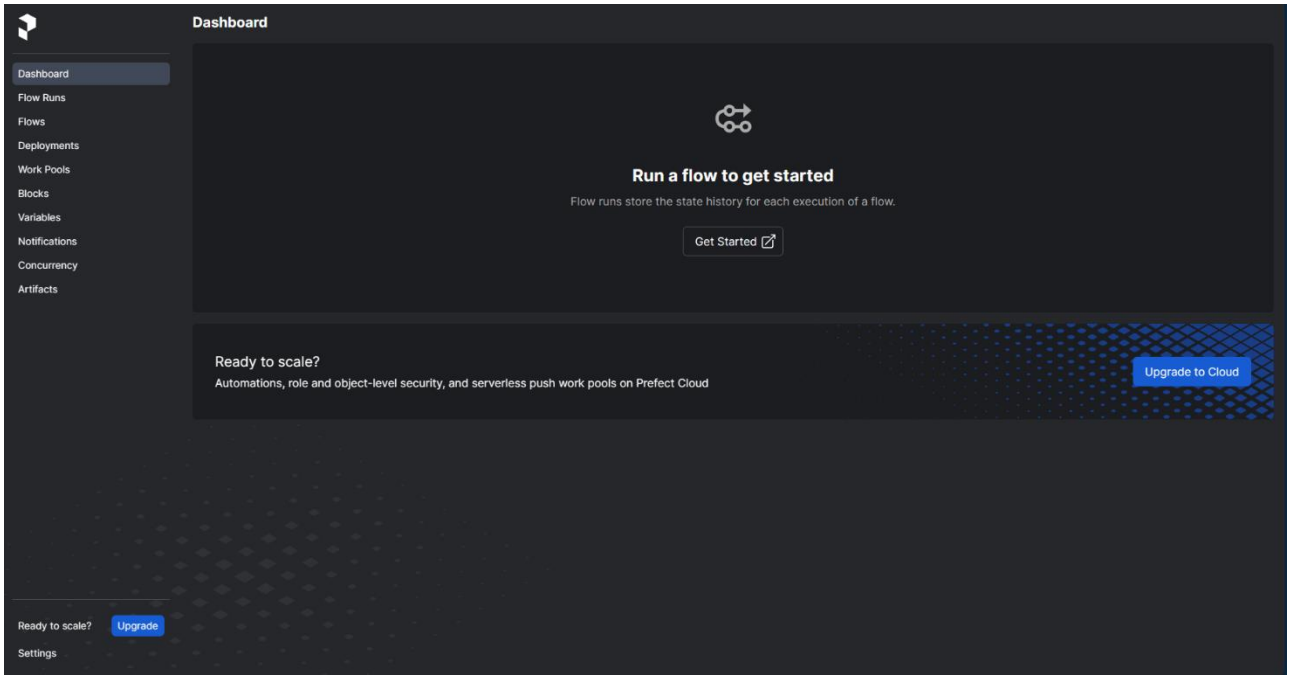




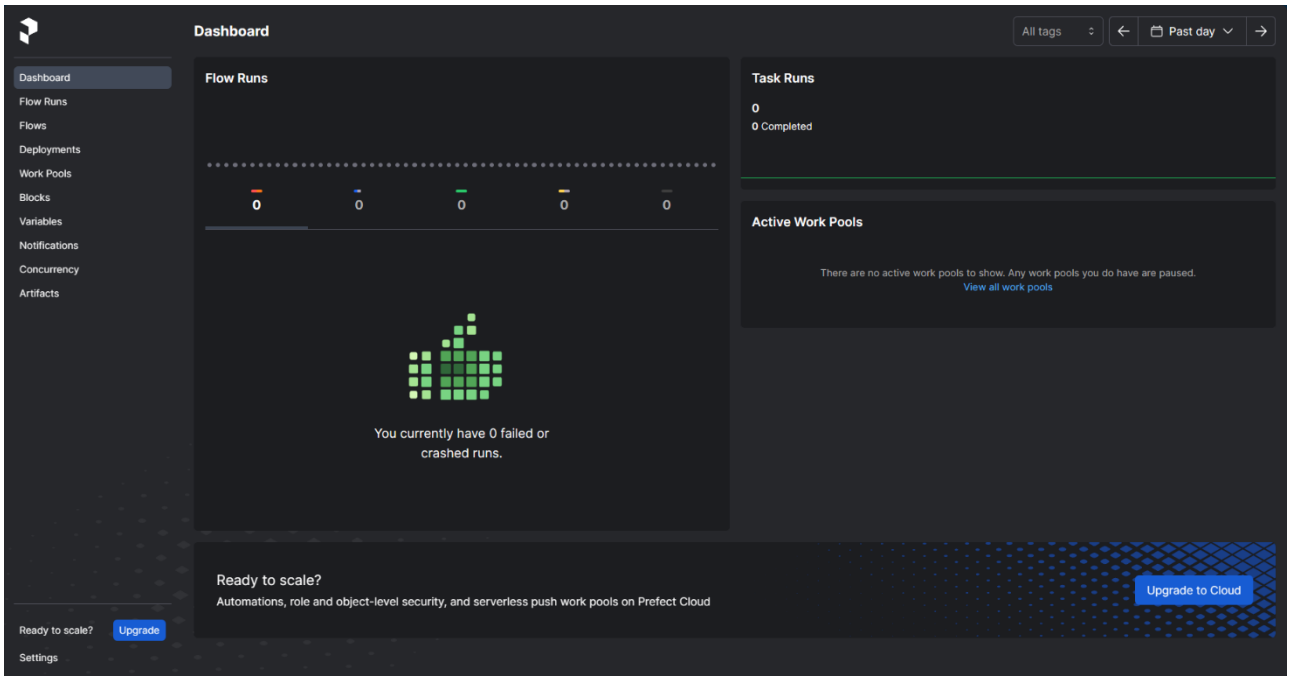


Task 8: (BONUS) Build a Prefect Workflow and Auto Schedule it. Show the Prefect Dashboard with relevant outputs.

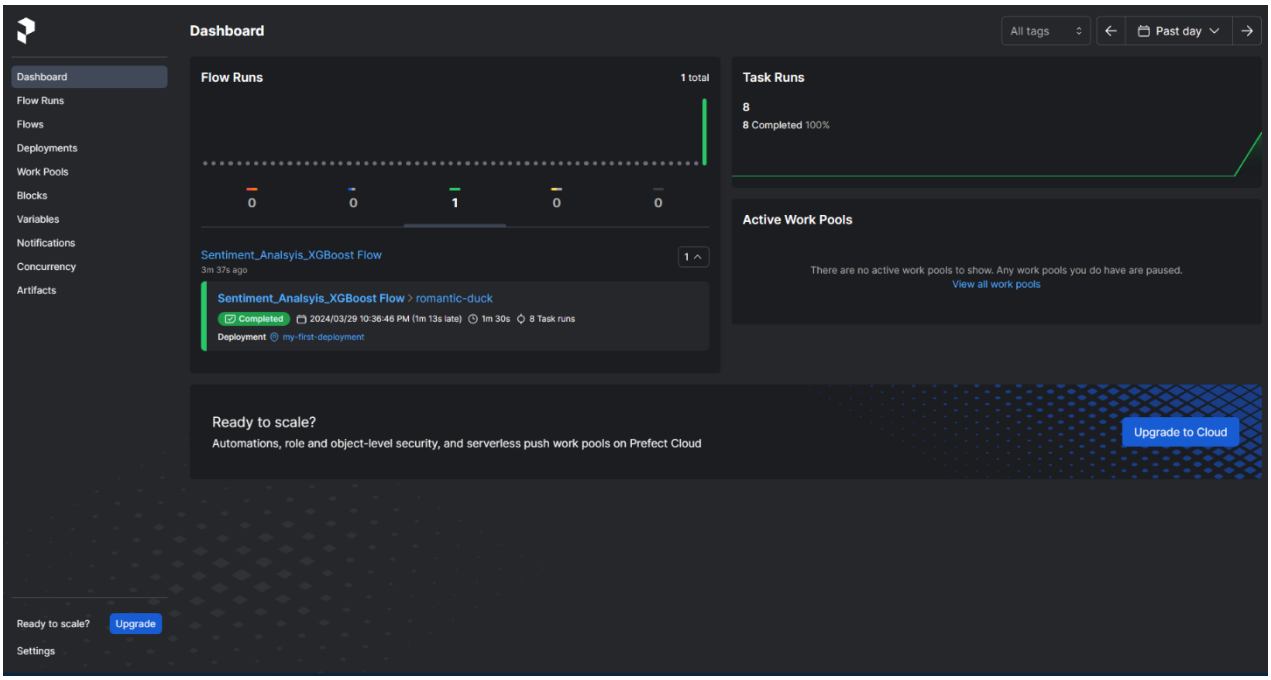
Prefect Dashboard



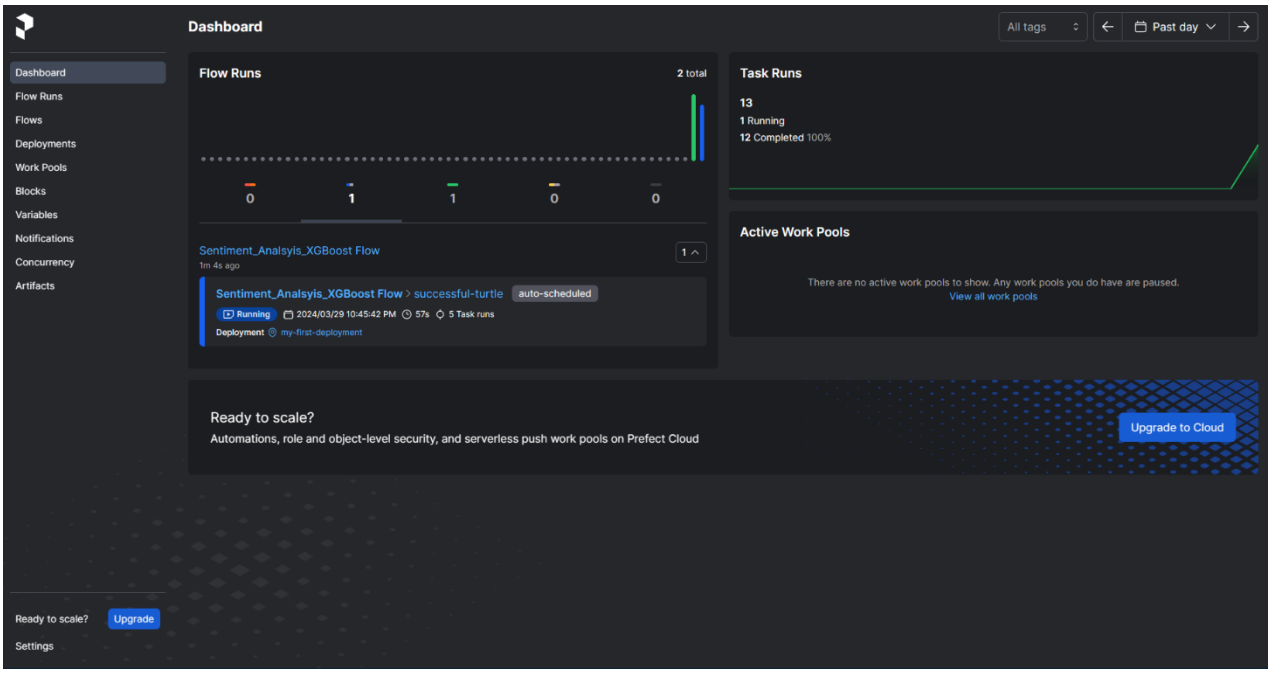
Prefect Dashboard After 1st Deployment



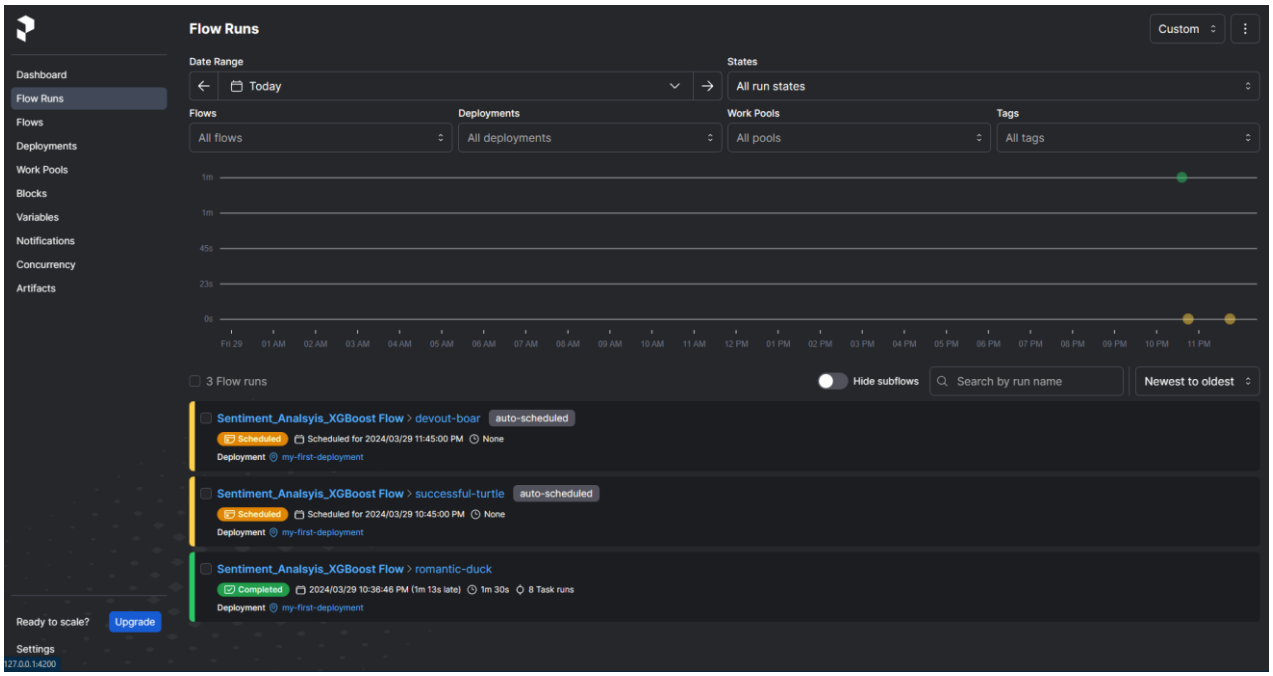
Prefect Dashboard (After Completing) 1st Task



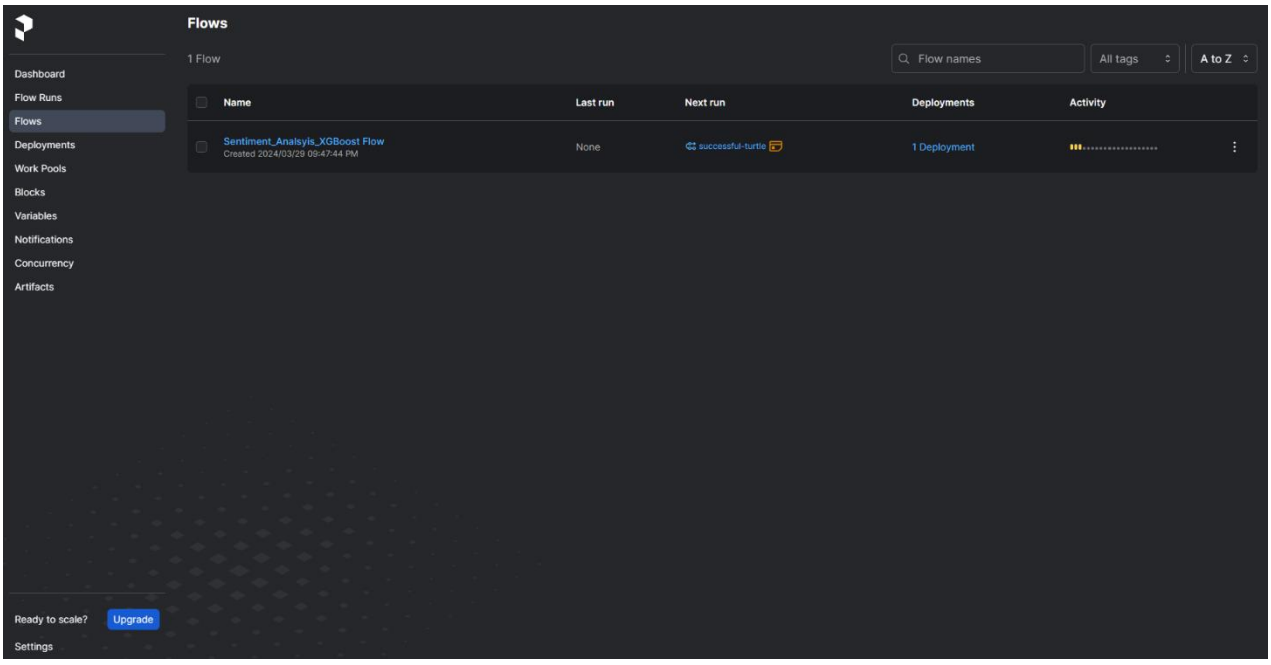
Prefect Dashboard (2nd Task Pending)



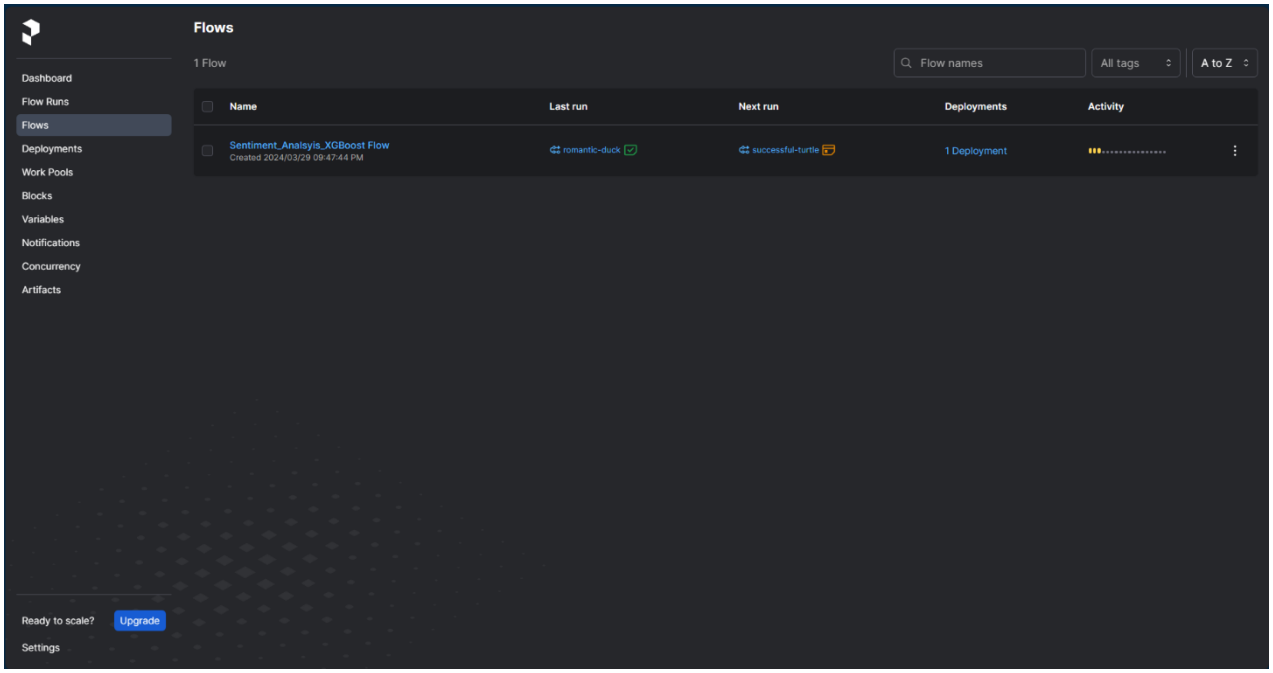
Prefect Flow Runs



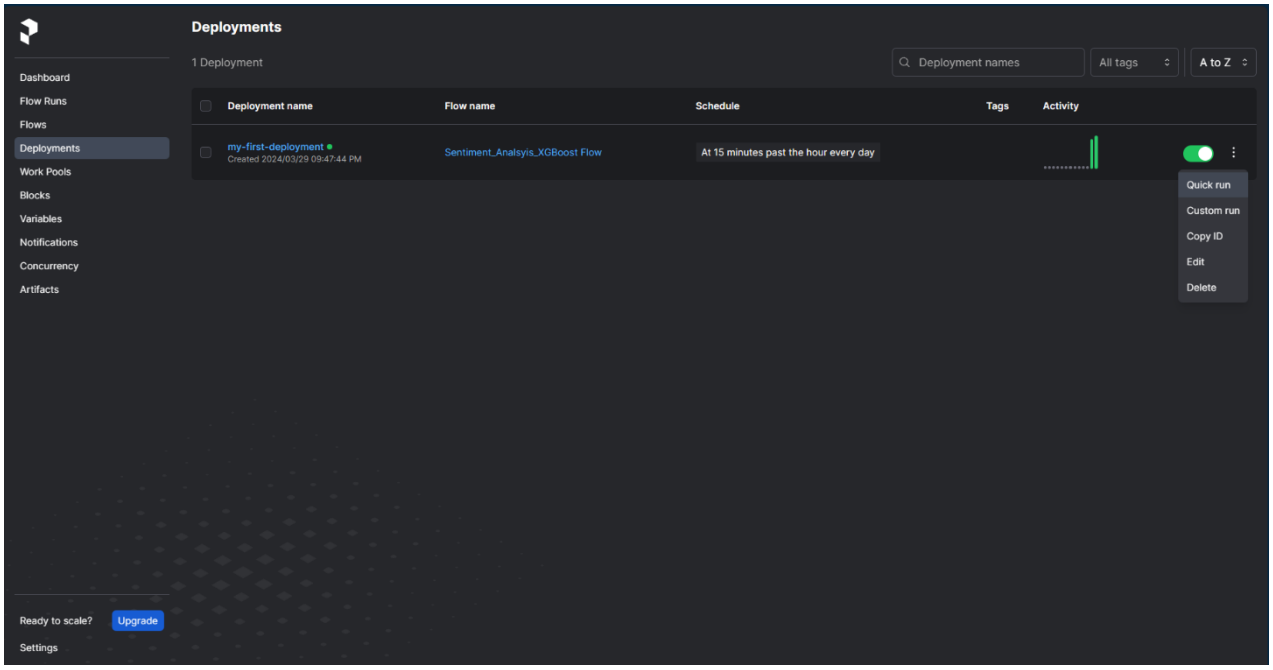
Prefect Flows (Last Run: None)



Prefect Flows (With Last & Next Run)



Prefect Deployments (Quick Run) (I Use Quick Run(name): romantic-duck



Prefect Flows Runs (successful-turtle)(logs)