*****

Hotel Cancellations Productionisation

McGill-MMA-EnterpriseAnalytics/hotel_cancellation_ML2

Alisa Liu 18ZL107
Chiara Lu . . . . . LuChiara
Hao Duong . . . . . haohaoduong
Jaya Chaturvedi Jaya2404
Keani Schuller . . . . . keanischuller
Reo Paul Jackson reojackson31

INSY 695 Final Project Presentation

TABLE OF CONTENTS

TEAM & ROLES

Jaya Chaturvedi

Data Engineer

Keani Schuller

Business Analyst

Chiara Lu

Data Scientist / PM

Reo Paul Jackson

ML Engineer

Alisa Liu

Data Scientist

Hao Duong

Business Analyst

ABOUT HOTEL CANCELLATIONS

HOTEL CANCELLATIONS ML1 SUMMARY

CONTEXT

Managing hotel cancellations is crucial for maintaining revenue and enhancing guest experiences

MODEL

Classification model experimentation to predict the likelihood of cancellations

INTERPRETATION

Random Forest as final model and feature importance to get insights

BUSINESS CASE

ACCURACY

Provide hoteliers with the ability to forecast the likelihood of a reservation being canceled at the time of booking using the highest accuracy model

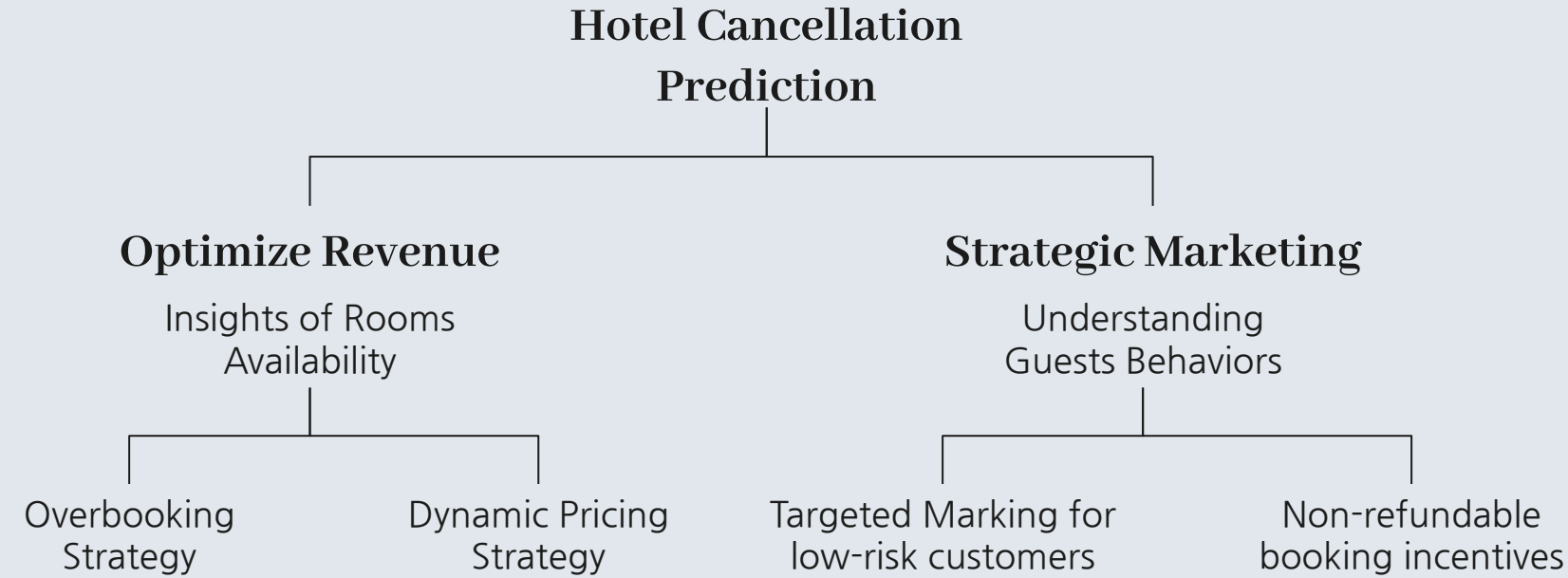
LONGEVITY

Audit the model to ensure it is easily interpreted, fair, and has longevity

DEPLOYMENT

Deployment is both local, using Docker containers, and on the cloud via Databricks, giving flexibility and scalability in operations

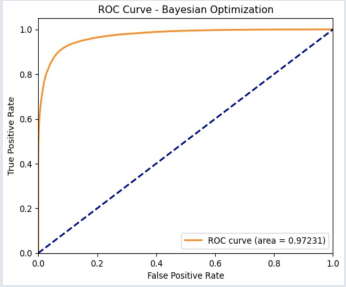
VALUE PROPOSITION



HYPERPARAMETER TUNING

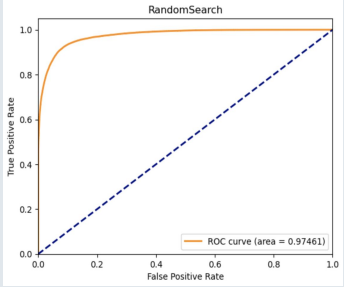
HYPERPARAMETER TUNING

Bayesian Optimization



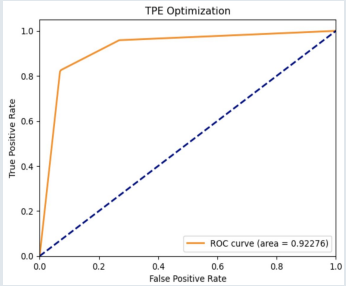
Best Params:
max_depth: 30
max_features: 1
min_samples_leaf: 1
min_samples_split: 2
n_estimators: 201

RandomizedSearchCV



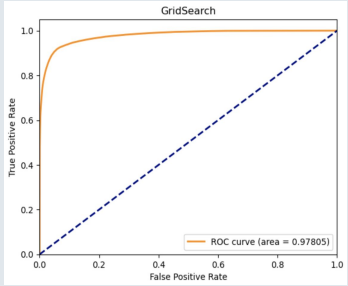
Best Params:
max_depth: None
max_features: 'sqrt'
min_samples_leaf: 1
min_samples_split: 5
n_estimators: 100

TPE Optimization (HyperOpt)



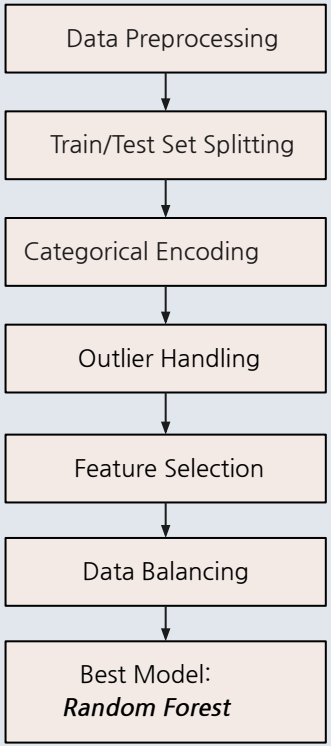
Best Params:
max_depth: None
max_features: 0.236
min_samples_leaf: 0
min_samples_split: 0
n_estimators: 2

GridSearchCV



Best Params:
max_features: None
min_samples_leaf: 1
n_estimators: 200

Part I Base Model



AutoML

1. Import train, test, and validation datasets
2. Library installations

- Light
- Neural Network
- Multifactor Dimensionality Reduction

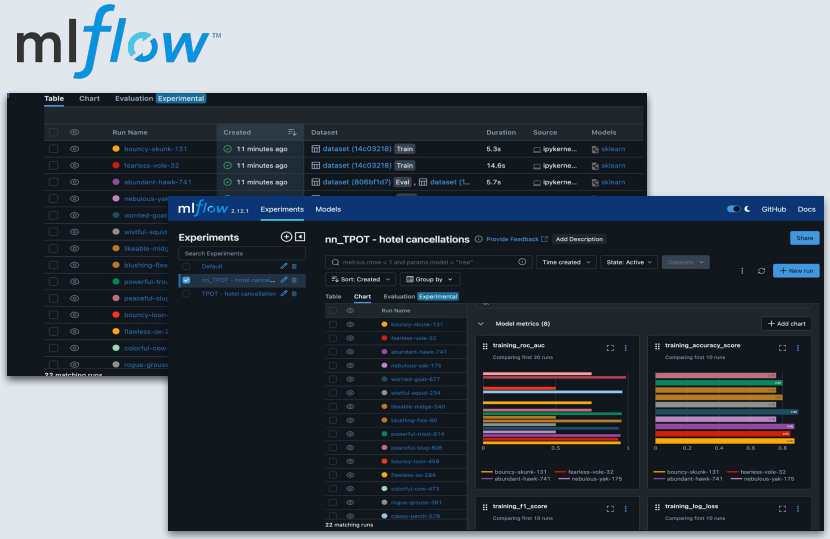
3. Initiate Model
4. Train
5. Export Pipeline

Key Benefits

Efficient

Accessible

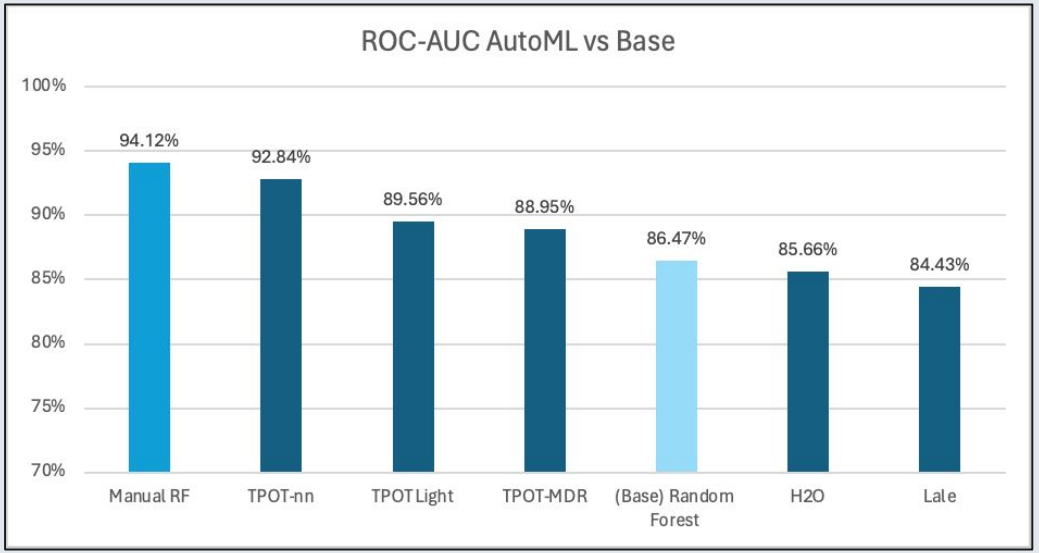
Reproducible



MODELING RESULTS

- ROC-AUC was used as the comparison metric because it discriminates how well a model is able to perform a binary classification
- The best pipeline from the AutoML approaches were the TPOT variations
- However, pipelines from AutoML did not prove to be better than the manually fine-tuned pipeline

Model	ROC-AUC
Manual RF	94.12%
TPOT-nn	92.84%
TPOT Light	89.56%
TPOT-MDR	88.95%
(Base) Random Forest	86.47%
H2O	85.66%
Lale	84.43%



DOCKER DEPLOYMENT

FastAPI & Gradio

MODEL TRAINING

Data Preprocessing

Model Training & Tuning

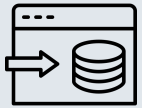
Model Registry (pickle file)

INTERFACE

FastAPI endpoints for predictions (single & batch predictions)

Gradio web app for user interface to make prediction on new input

DEPLOYMENT



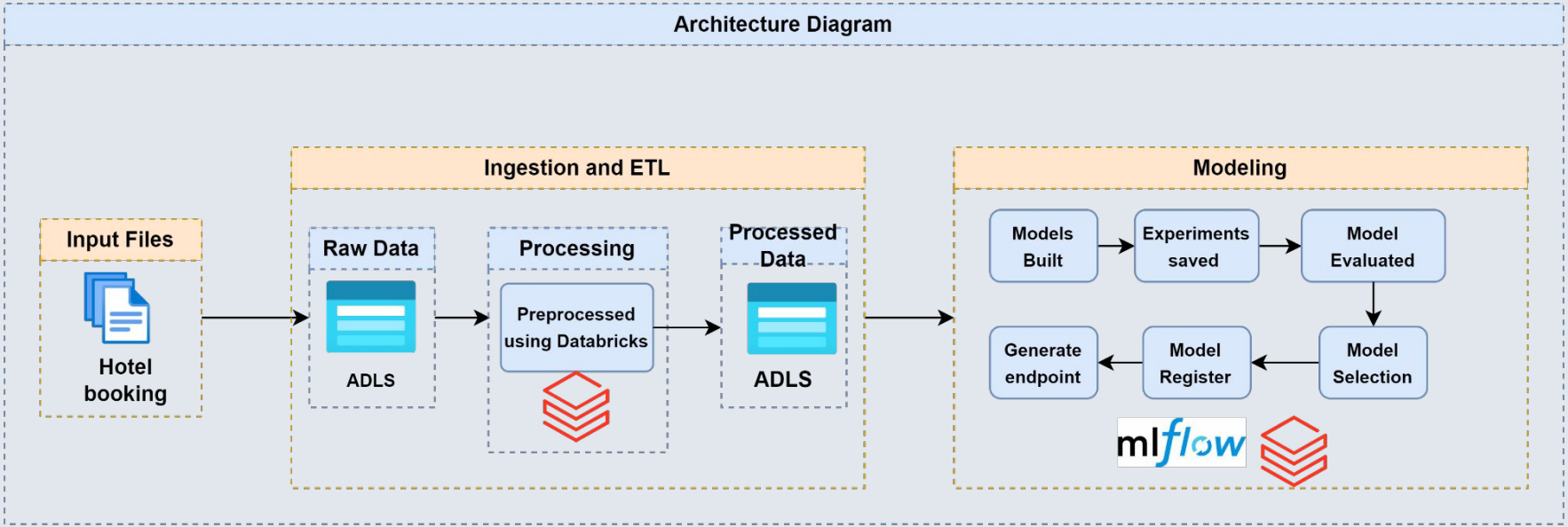
Docker Image for files and dependencies

Docker container to run the FastAPI & Gradio applications

DEMO

ARCHITECTURE

USING DATABRICKS



USING DATABRICKS

RandomForest_Champion	1 day ago		11.0min	best_mo...	RandomForest_Champion ...	-	0.978017118...	0.870560643...	0.941280977...
fearless-finch-111	1 day ago		11.3min	best_mo...	sklearn	0.978017118...	-	-	-
adorable-grub-283	1 day ago		7.6min	best_mo...	-	0.881166351...	-	-	-
TPOT_model	1 day ago		5.0min	best_mo...	-	0.919666934...	-	-	-
RandomForest_BestParam...	1 day ago		12.2min	best_mo...	RandomForest_BestParam...	0.978017118...	-	-	-

Serving endpoints >

BestParameters

Permissions | Edit endpoint | Query endpoint

Serving endpoint state: Ready

Created by: jaya.chaturvedi@mail.mcgill.ca

URL: https://adb-4634864076059286.6.azuredatabricks.net/serving-endpoints/BestParameters/invocations

Tags:

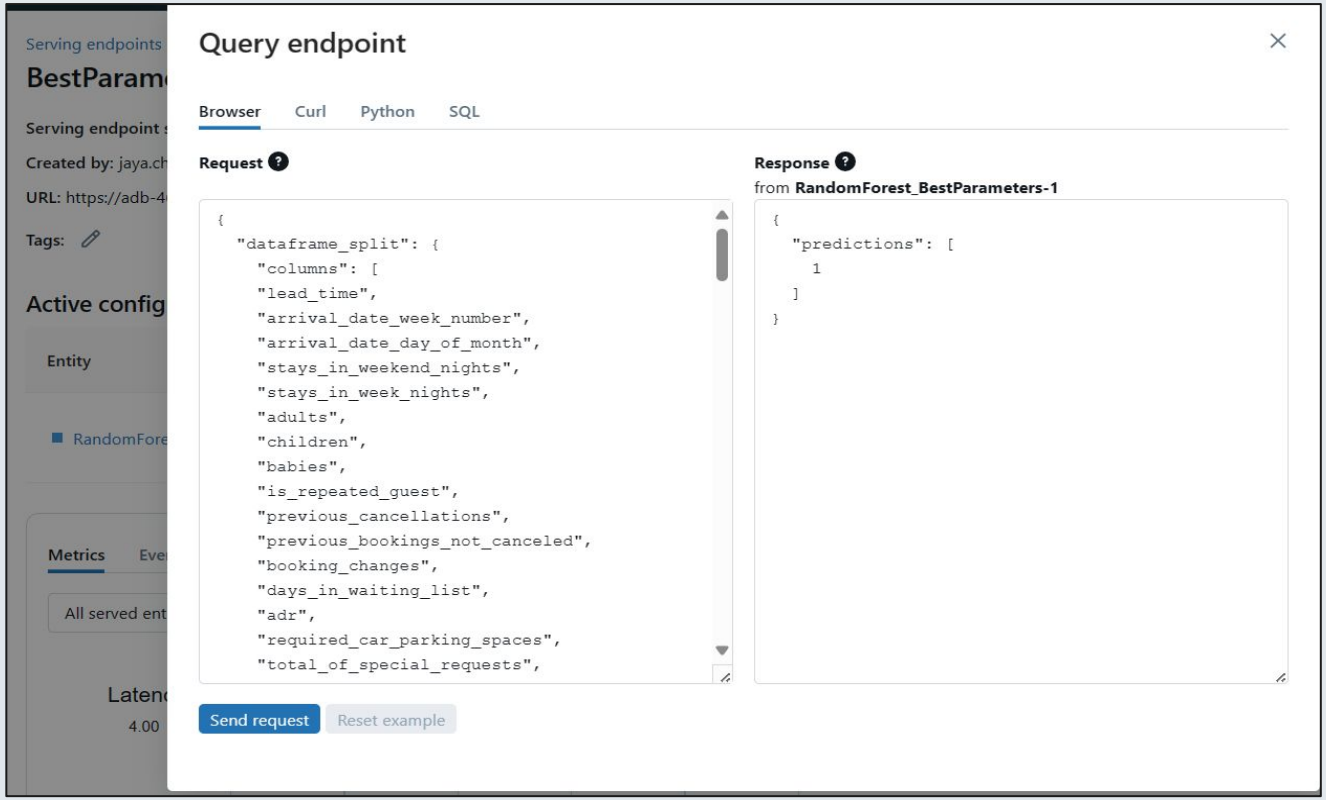
Inference table: Not enabled

Route optimization: Not enabled

Active configuration

Entity	Version	Name	State	Compute	Traffic (%)
RandomForest_BestParameters	Version 1	RandomForest_BestParameters-1	Ready	**CPU**, **Small** 0-4 concurrency (0-4 DBU)	100

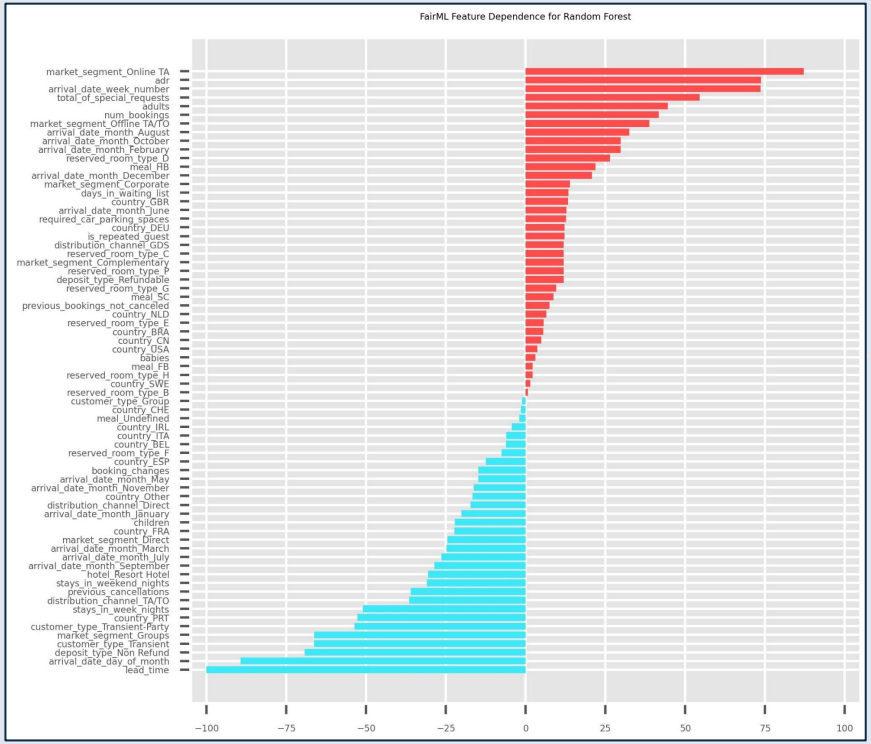
USING DATABRICKS



FAIRNESS & EXPLAINABILITY

MODEL FAIRNESS

- Used FairML to evaluate the fairness of our final model
- market_segment_Online TA (Online Travel Agent) has the highest positive influence
- Reservations made through online travel agents significantly increase the likelihood that the model predicts a cancellation
- Conversely, lead_time has the highest negative influence
- Reservations made well in advance decrease the likelihood it predicts a cancellation



MODEL EXPLAINABILITY WITH LIME

PREDICTION

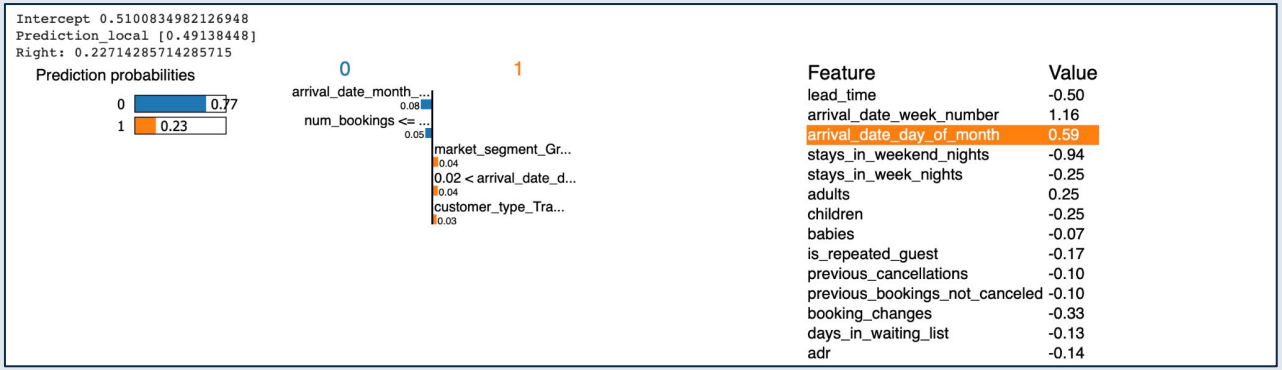
Model predicts that there's a 77% chance that the reservation will not be cancelled and a 23% chance that it will

POSITIVE INFLUENCE

Arrival month's value of 0.59 suggests that the month of the arrival date contributes towards a higher chance of cancellation

NEGATIVE INFLUENCE

Variables like children, babies and is_repeated_guest all have negative values meaning they contribute to a lesser likelihood of cancellation



MODEL DRIFT

MODEL DRIFT

Adversarial Drift

Adversarial Drift Test Setup

```python
def detect_adversarial_examples(clf_rf_best, X, threshold=0.5):

    # Get prediction probabilities for each class
    probs = clf_rf_best.predict_proba(X)
    # Get the maximum prediction probability for each sample
    max_probs = np.max(probs, axis=1)
    # Flag as potential adversarial if max probability is below the threshold
    return max_probs < threshold

# Example usage
adversarial_flags = detect_adversarial_examples(clf_rf_best, X_test)

print("Potential adversarial detected:", adversarial_flags)
```

```
Potential adversarial detected: [False False False ... False False False]
```

There is no adversarial drift detected in the model

KSDrift

KSDrift Setup

```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder


# Train a Random Forest classifier
clf = RandomForestClassifier()
clf.fit(X_ref, y_ref)

# Generate predictions for the reference and new dataset
y_ref_pred = clf.predict_proba(X_ref)
y_new_pred = clf.predict_proba(X_new)

ohe = OneHotEncoder(sparse=False, categories='auto', handle_unknown='ignore')
y_ref_ohe = ohe.fit_transform(y_ref_pred.reshape(-1, 1))
y_new_ohe = ohe.transform(y_new_pred.reshape(-1, 1))

# Initialize KSDrift detector
cd = KSDrift(p_val=0.05, x_ref=y_ref_ohe)

# Check for drift in the new predictions
drift_results = cd.predict(y_new_ohe)

# Output
print(f'Drift detected: {drift_results["data"]["is_drift"]}')
```

```
Drift detected: 0
```

CI/CD

GitHub ACTIONS

The goal of GitHub Actions is to automate build, test, and deployment pipeline. There can be multiple action workflows that start manually or on a schedule.

```yaml
name: Docker Image CI

on:
  push:
    branches: [ "main" ]
  pull_request:
    branches: [ "main" ]

jobs:

  build:

    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@main
    - name: Build the Docker image
      run: docker build . --file Dockerfile --tag my-image-name:$(date +%s)
```

What this yml file does:

1. Triggers on push and pull requests to the main branch.
1. Job named "build" runs on Ubuntu-latest.
2. Checkout the repository using the latest version of the checkout action.
3. Build the Docker image using the provided Dockerfile.
4. Tag the image with a unique identifier based on the current timestamp, i.e. “my-image-name:1620173371”.

For CICD at enterprise level, Jenkins can be used to continually build, test, and deploy pipelines. GitHub Actions works well with GitHub-based workflows.

THANK YOU!

Q&A

APPENDIX

EXPORT TPOT-NN

```
# export best nn_tpot
nn_tpot.export('nn_tpot_pipeline.py')
```

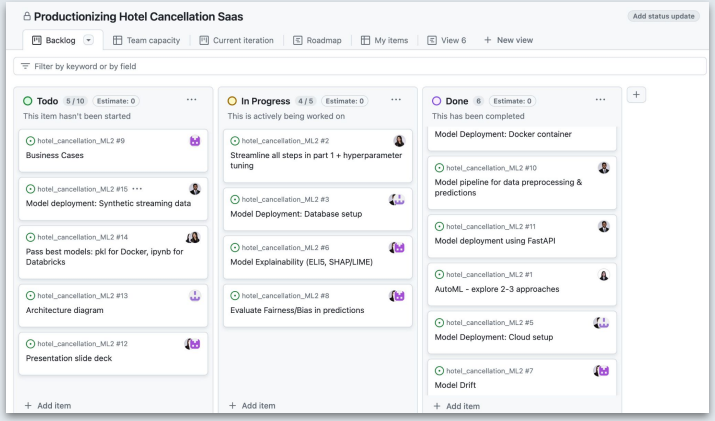
✓ 0.0s

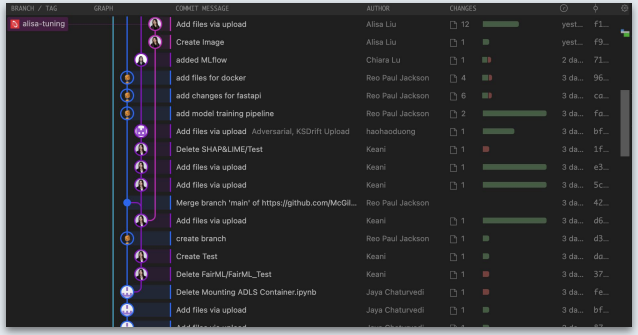
nn_tpot_pipeline.py > ...

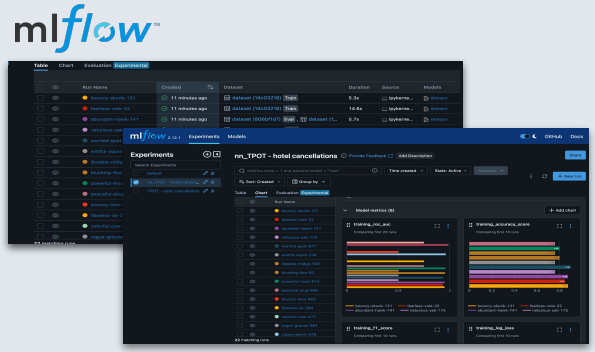
```python
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline, make_union
from sklearn.tree import DecisionTreeClassifier
from tpot.builtins import StackingEstimator
from xgboost import XGBClassifier

# NOTE: Make sure that the outcome column is labeled 'target' in the data file
tpot_data = pd.read_csv('PATH/TO/DATA/FILE', sep='COLUMN_SEPARATOR', dtype=np.float64)
features = tpot_data.drop('target', axis=1)
training_features, testing_features, training_target, testing_target = \
            train_test_split(features, tpot_data['target'], random_state=None)

# Average CV score on the training set was: 0.9310294569353934
exported_pipeline = make_pipeline(
    StackingEstimator(estimator=DecisionTreeClassifier(criterion="gini", max_depth=4, min_samples_leaf=4, min_samples_split=9)),
    XGBClassifier(learning_rate=0.5, max_depth=5, min_child_weight=11, n_estimators=100, n_jobs=1, subsample=0.6000000000000001, verbosity=0)
)

exported_pipeline.fit(training_features, training_target)
results = exported_pipeline.predict(testing_features)
```

MLOps BEST PRACTICES

- Project Management using Kanban
- Keeping experiment logs on MLflow
- Working in squads to attack a major problem in small steps
- Use GitLens to view changes on GitHub branches
- Saving failed tests







DOCKER IMAGE (FOR FASTAPI)

```
1  FROM python:3.11
2
3  WORKDIR /code
4
5  COPY ./requirements.txt /code/requirements.txt
6
7  RUN pip install --no-cache-dir --upgrade -r /code/requirements.txt
8
9  #
10 COPY ./preprocessing.py /code/
11 COPY ./bestModel_hpo.pkl /code/
12 COPY ./bestModel_tpot.pkl /code/
13 COPY ./fastapi_app.py /code/
14
15 #
16 CMD ["uvicorn", "fastapi_app:app", "--host", "0.0.0.0", "--port", "8000"]
```