AutoGluon Assignment

❖ Part 2

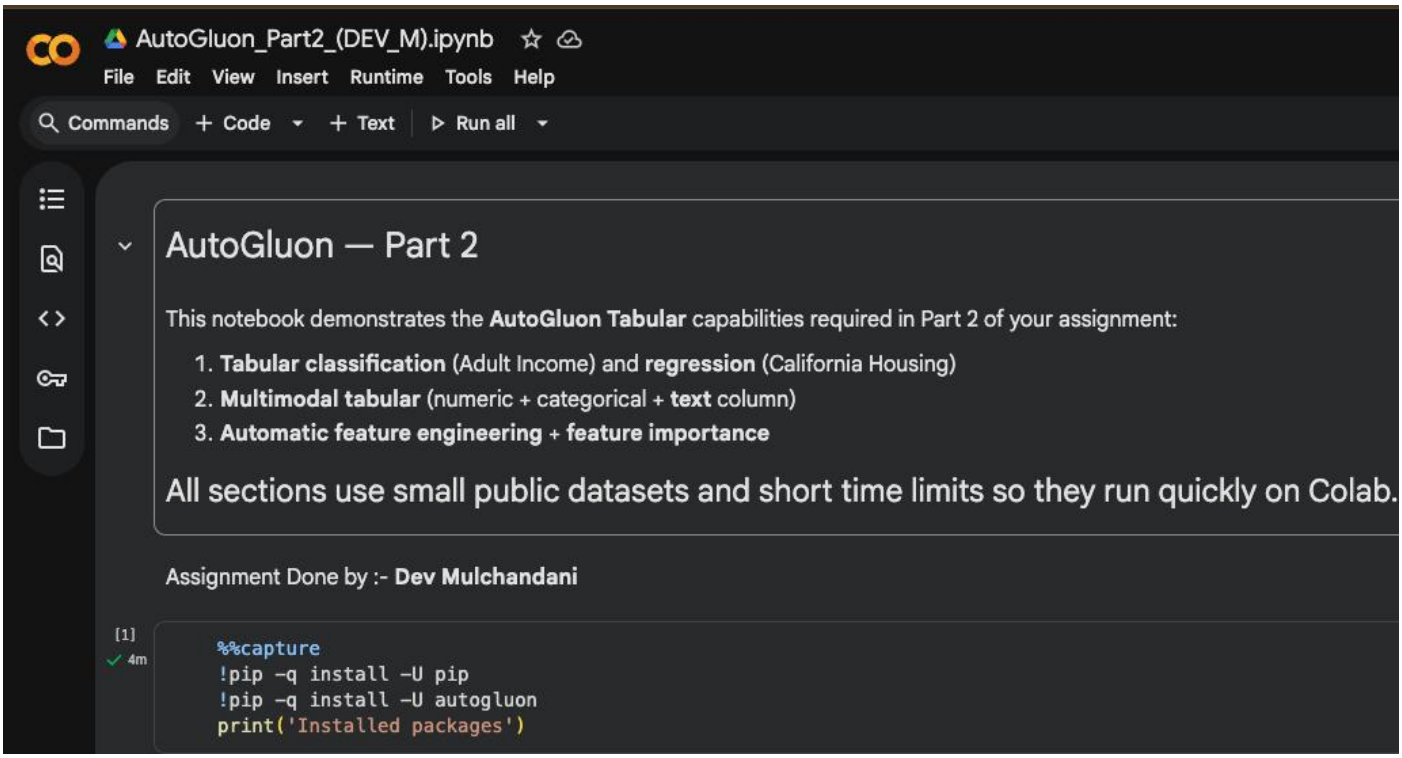
❖ Colab Notebook :-Link

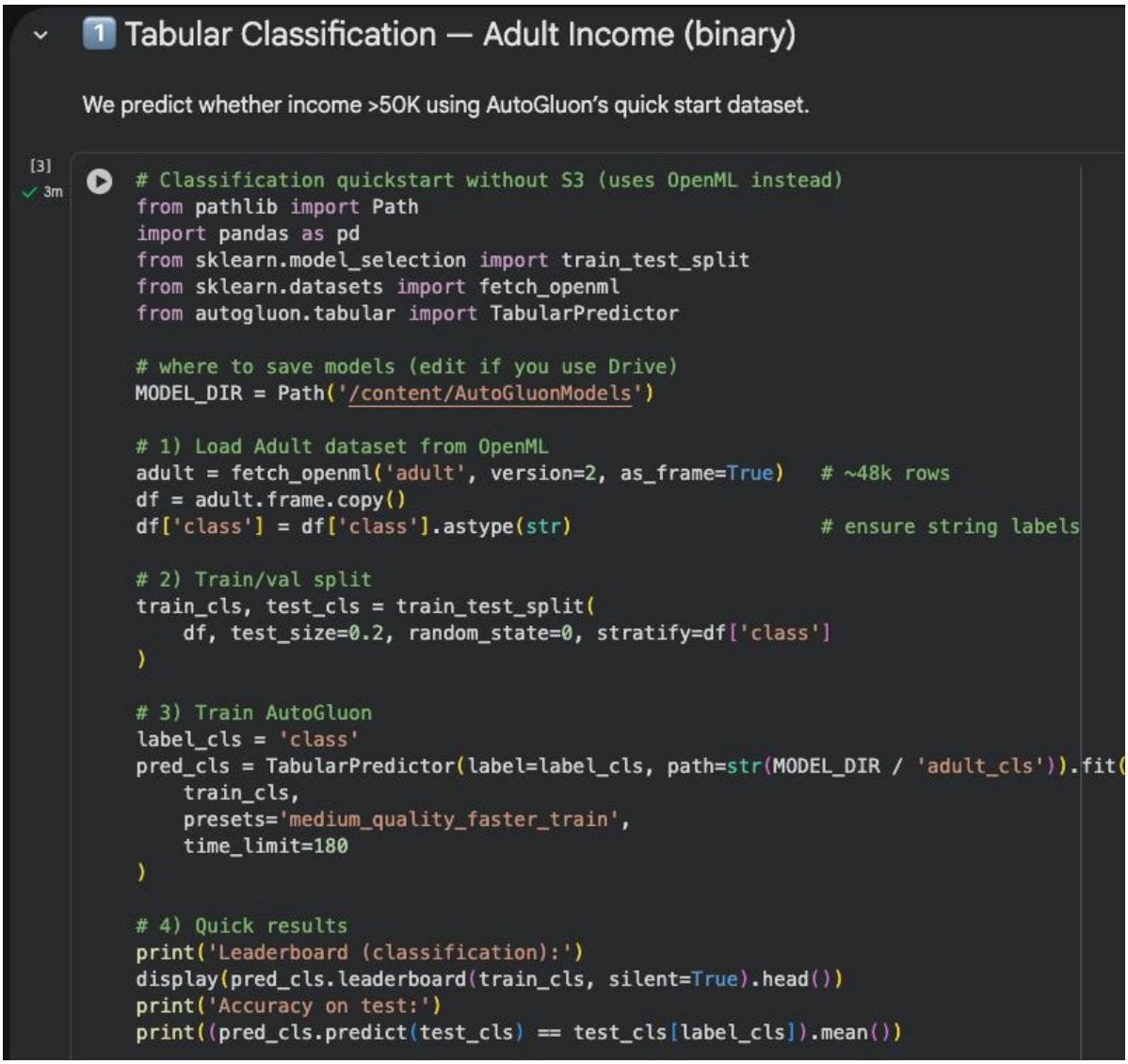
❖ Submitted By :- Dev Mulchandani

❖ Overview :-

In Part 2 of the AutoGluon assignment, I demonstrated how AutoGluon can be applied to different machine learning problems using three quick examples. First, I performed tabular classification on the Adult Income dataset from OpenML, where the goal was to predict whether an individual earns more than $50K a year. Next, I built a regression model using the California Housing dataset to predict median house values based on location and demographic features. Finally, I created a small multimodal tabular example that combined numeric and text features to predict spending behavior. In each case, I used AutoGluon's TabularPredictor with medium-quality presets and short time limits, allowing the models to automatically handle preprocessing, feature engineering, and training with minimal code. This part showed how easily AutoGluon adapts to various data types and tasks, providing quick and accurate results with very little manual effort.

❖ Screenshots :-





	model	score_test	score_val	eval_metric	pred_time_test	pred_time_val	fit_time	pred_time_test_marginal	pred_time_val_marginal	fit_time_marginal	stack_level	can_infer	fit_order
0	RandomForestEntr	0.990479	0.8528	accuracy	3.879886	0.208280	15.436682	3.879886	0.208280	15.436682	1	True	4
1	RandomForestGini	0.990428	0.8520	accuracy	3.352239	0.351482	14.402138	3.352239	0.351482	14.402138	1	True	3
2	ExtraTreesGini	0.989788	0.8420	accuracy	2.914752	0.234976	9.171849	2.914752	0.234976	9.171849	1	True	6
3	ExtraTreesEntr	0.989686	0.8404	accuracy	3.791875	0.276714	8.114681	3.791875	0.276714	8.114681	1	True	7
4	WeightedEnsemble_L2	0.885368	0.8728	accuracy	0.562812	0.048370	55.397773	0.004806	0.001364	0.138440	2	True	10

```
Accuracy on test:
0.8724536800081891
```

1 Tabular Regression — California Housing (sklearn)

We load the classic California housing dataset from scikit-learn and train a regression model.

[4] ✓ 3m

```
from sklearn.datasets import fetch_california_housing
import pandas as pd
from autogluon.tabular import TabularPredictor

cal = fetch_california_housing(as_frame=True)
df_reg = cal.frame.copy()
df_reg.rename(columns={'MedHouseVal':'target'}, inplace=True)
label_reg = 'target'

pred_reg = TabularPredictor(label=label_reg, path=str(MODEL_DIR / 'cal_housing_reg'),
                            eval_metric='root_mean_squared_error') \
    .fit(df_reg, presets='medium_quality_faster_train', time_limit=180)

print('Leaderboard (regression):')
display(pred_reg.leaderboard(df_reg, silent=True).head())
```

```
AutoGluon training complete, total runtime = 180.17s ... Best model: WeightedEnsemble_L2 | Estimated inference throughput: 4349.6 rows/s (2064 batch size)
TabularPredictor saved. To load, use: predictor = TabularPredictor.load("/content/AutoGluonModels/cal_housing_reg")
Leaderboard (regression):
```

	model	score_test	score_val	eval_metric	pred_time_test	pred_time_val	fit_time	pred_time_test_marginal	pred_time_val_marginal	fit_time_marginal	stack_level	can_infer
0	WeightedEnsemble_L2	-0.223297	-0.408002	root_mean_squared_error	4.219149	0.474523	100.801907	0.005034	0.000513	0.010566	2	True
1	CatBoost	-0.225790	-0.409260	root_mean_squared_error	0.162361	0.015482	95.735993	0.162361	0.015482	95.735993	1	True
2	LightGBM	-0.230212	-0.423584	root_mean_squared_error	4.051753	0.458528	5.055348	4.051753	0.458528	5.055348	1	True
3	RandomForestMSE	-0.236308	-0.500920	root_mean_squared_error	1.675405	0.274807	55.218094	1.675405	0.274807	55.218094	1	True
4	LightGBMXT	-0.271515	-0.443633	root_mean_squared_error	18.417991	1.696069	12.683245	18.417991	1.696069	12.683245	1	True

2 Multimodal Tabular — add a text column

We create a small dataset containing numeric, categorical, and **text** features to show AutoGluon's multimodal handling with the same API.

[5] ✓ 7s

```
import pandas as pd
from autogluon.tabular import TabularPredictor

mm = pd.DataFrame({
    'age': [25,45,33,52,41,22,61,37,29,48],
    'income': [35000,120000,70000,150000,90000,28000,200000,80000,60000,110000],
    'role': ['entry','exec','engineer','director','pm','intern','c-suite','analyst','engineer','exec'],
    'desc': ['entry-level role','senior executive','mid-level engineer','director position','project manager','intern new grad'
})
mm['high_spender'] = [0,1,0,1,0,0,1,0,0,1]

train_mm = mm.sample(frac=0.8, random_state=1)
test_mm  = mm.drop(train_mm.index)

pred_mm = TabularPredictor(label='high_spender', path=str(MODEL_DIR / 'mm')) \
    .fit(train_mm, presets='medium_quality_faster_train', time_limit=60)

print('Predictions on holdout rows:')
display(pred_mm.predict(test_mm))
```

```
AutoGluon training complete, total runtime = 5.3s ... Best model: WeightedEnsemble_L2 | Estimated inference throughput: 471.3 rows/s (2 batch size)
Disabling decision threshold calibration for metric `accuracy` due to having fewer than 10000 rows of validation data for calibration, to avoid overfitting (2 rows).
    `accuracy` is generally not improved through threshold calibration. Force calibration via specifying `calibrate_decision_threshold=True`.
TabularPredictor saved. To load, use: predictor = TabularPredictor.load("/content/AutoGluonModels/mm")
Predictions on holdout rows:
```

	high_spender
5	0
8	0

dtype: int64

3 Automatic Feature Engineering & Importances

AutoGluon performs preprocessing and feature generation internally. We visualize feature importances and transformed features using the Adult dataset.

[8] ✓ 3m

```python
# Feature engineering & importances using the Adult dataset (no S3)

from sklearn.datasets import fetch_openml
import pandas as pd
from autogluon.tabular import TabularPredictor

# 1) Load data
adult = fetch_openml('adult', version=2, as_frame=True)   # public & reliable
train_fe = adult.frame.copy()
train_fe['class'] = train_fe['class'].astype(str)         # ensure string labels
label_fe = 'class'

# 2) Train a quick model
pred_fe = TabularPredictor(label=label_fe, path='AutoGluonModels/feat_eng').fit(
    train_fe, presets='medium_quality_faster_train', time_limit=120
)

# 3) Feature importances
fi = pred_fe.feature_importance(train_fe)
print('Top feature importances:')
display(fi.sort_values('importance', ascending=False).head(10))

# 4) View transformed features produced by AutoGluon's pipeline
sample_batch = train_fe.head(200)
X_transformed = pred_fe.transform_features(sample_batch)
print('Transformed feature columns (preview):', list(X_transformed.columns)[:12])
display(X_transformed.head())
```

```
AutoGluon training complete, total runtime = 167.67s ... Best model: WeightedEnsemble_L2 | Estimated inference throughput: 14958.2 rows/s (2500 batch size)
Disabling decision threshold calibration for metric `accuracy` due to having fewer than 10000 rows of validation data for calibration, to avoid overfitting (2500 rows).
    `accuracy` is generally not improved through threshold calibration. Force calibration via specifying `calibrate_decision_threshold=True`.
TabularPredictor saved. To load, use: predictor = TabularPredictor.load("/content/AutoGluonModels/feat_eng")
Computing feature importance via permutation shuffling for 14 features using 5000 rows with 5 shuffle sets...
    26.94s = Expected runtime (5.39s per shuffle set)
    26.66s = Actual runtime (Completed 5 of 5 shuffle sets)
Top feature importances:
```

	importance	stddev	p_value	n	p99_high	p99_low
capital-gain	0.04676	0.003448	0.000004	5	0.053859	0.039661
occupation	0.02304	0.003968	0.000102	5	0.031211	0.014869
marital-status	0.02176	0.003129	0.000050	5	0.028202	0.015318
age	0.01968	0.003022	0.000065	5	0.025902	0.013458
relationship	0.01812	0.002556	0.000046	5	0.023382	0.012858
education	0.01380	0.003353	0.000387	5	0.020703	0.006897
capital-loss	0.01364	0.002355	0.000103	5	0.018490	0.008790
hours-per-week	0.01108	0.002941	0.000544	5	0.017136	0.005024
workclass	0.00792	0.001425	0.000121	5	0.010855	0.004985
fnlwgt	0.00772	0.002081	0.000577	5	0.012006	0.003434

```
Transformed feature columns (preview): ['age', 'fnlwgt', 'education-num', 'sex', 'capital-gain', 'capital-loss', 'hours-per-week', 'workclass', 'education', 'marital-status', 'occupation', 'relationship'
```

	age	fnlwgt	education-num	sex	capital-gain	capital-loss	hours-per-week	workclass	education	marital-status	occupation	relationship	race	native-country
0	25	226802	7	0	0	0	40	3	1	4	6	3	2	37
1	38	89814	9	0	0	0	50	3	11	2	4	0	4	37
2	28	336951	12	0	0	0	40	1	7	2	10	0	4	37
3	44	160323	10	0	7688	0	40	3	15	2	6	0	2	37
4	18	103497	10	1	0	0	30	NaN	15	4	NaN	3	4	37