Double ML & Heterogeneous Treatment Effects

Assignment 4

Zaheer Abbas

2025-07-02

Table of contents

1 Loading Dataset

```
print(dataset.head(6))
```

```
   age  gender region  sessions_per_day  past_engagement device_type  \
0   54    Male   Asia                 3        34.831284         iOS
1   18    Male     US                 5        77.758422         iOS
2   42    Male     US                 1        53.025246         iOS
3   27  Female     US                 2        19.752821     Android
4   53  Female     US                 1        79.309742         iOS
5   35  Female     EU                 7        71.962521     Android

   app_version  notification_sent  pop_up_rem  notification_opened  engagement
0            1                  1           0                    0   14.924706
1            2                  0           0                    0    5.229459
2            1                  1           1                    1   27.129607
3            1                  0           1                    1   17.782530
4            2                  0           0                    1   25.818239
5            1                  1           1                    0   23.795170
```

2 ATE of Notification Sent via Double ML (Partial-Linear Model)

```
# Final OLS regression: y_resid ~ t_resid
model = sm.OLS(y_resid_all, t_resid_all_with_const)
results = model.fit()

# Print the ATE estimate
print("Estimated ATE (DML):", results.params[1])
print("Standard Error:", results.bse[1])
print(results.summary())
```

```
Estimated ATE (DML): 8.739531269869078
Standard Error: 0.16458490838508583
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.220
Model:                            OLS   Adj. R-squared:                  0.220
Method:                 Least Squares   F-statistic:                     2820.
Date:                Wed, 02 Jul 2025   Prob (F-statistic):               0.00
Time:                        14:46:24   Log-Likelihood:                -34722.
```

```
No. Observations:               10000   AIC:                         6.945e+04
Df Residuals:                    9998   BIC:                         6.946e+04
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.0192      0.078      0.247      0.805      -0.134       0.172
x1             8.7395      0.165     53.100      0.000       8.417       9.062
==============================================================================
Omnibus:                        7.281   Durbin-Watson:                   2.010
Prob(Omnibus):                  0.026   Jarque-Bera (JB):                6.782
Skew:                           0.030   Prob(JB):                       0.0337
Kurtosis:                       2.887   Cond. No.                         2.11
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly speci
```

3 ATE of Notification Sent via Double ML (AIPW Model)

```python
# Compute AIPW scores
aipw_scores = (
    T * (Y - mu1_preds) / e_preds + mu1_preds
    - (1 - T) * (Y - mu0_preds) / (1 - e_preds) - mu0_preds
)

# Estimate ATE and standard error
ate = np.mean(aipw_scores)
se = np.std(aipw_scores) / np.sqrt(len(aipw_scores))

print(f"Estimated ATE (AIPW): {ate:.4f}")
print(f"Standard Error: {se:.4f}")
```

```
Estimated ATE (AIPW): 8.6743
Standard Error: 0.2725
```

4 Verification with DoubleMl package

```
# Create and fit DoubleMLPLR model
dml_plr_obj = dml.DoubleMLPLR(obj_dml_data, ml_l, ml_m)
dml_plr_obj.fit()

# Print results
print(dml_plr_obj.summary)
```

```
                       coef   std err          t  P>|t|    2.5 %   97.5 %
notification_sent  8.764015  0.166246  52.717287    0.0  8.43818  9.08985
```

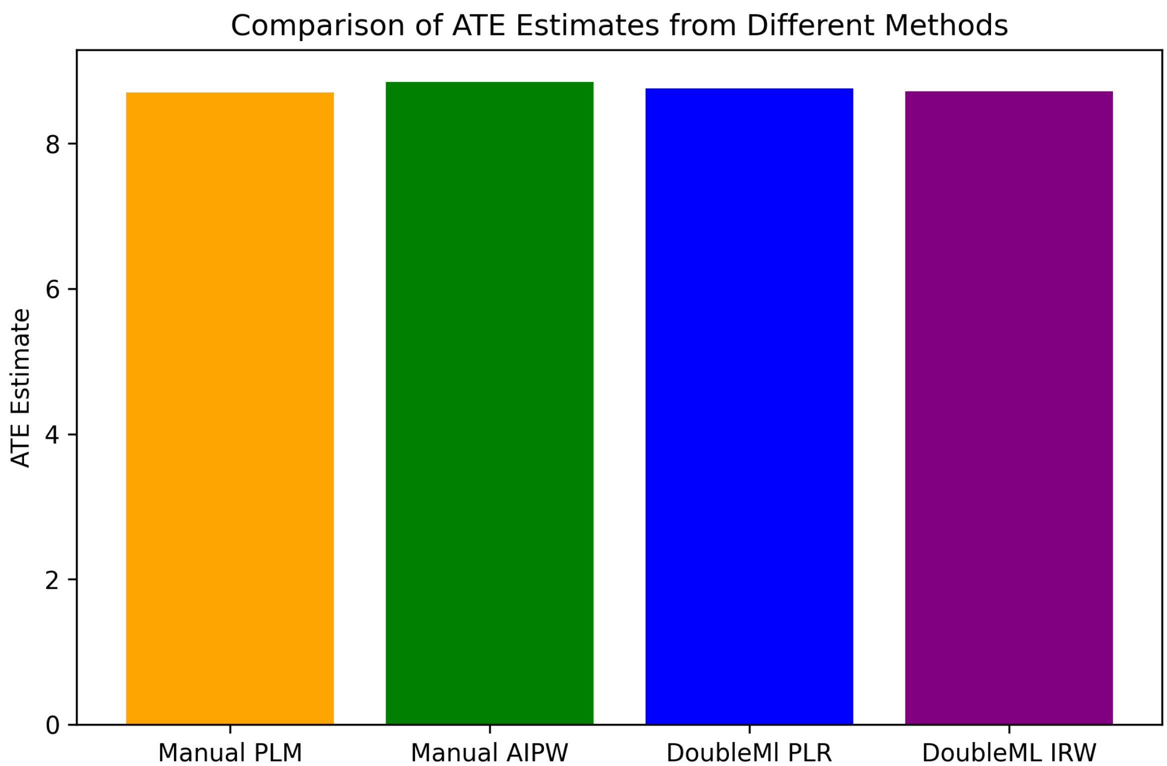
```
#  DoubleMLIRM model
dml_irm_obj = dml.DoubleMLIRM(obj_dml_data, ml_g=ml_g_cloned, ml_m=ml_m_cloned)
dml_irm_obj.fit()

# Print results
print(dml_irm_obj.summary)
```

```
                       coef   std err          t  P>|t|     2.5 %    97.5 %
notification_sent  8.722585  0.164558  53.006115    0.0  8.400057  9.045113
```

```
methods= ["Manual PLM", "Manual AIPW", "DoubleMl PLR", "DoubleML IRW"]
estimates = [manual_plm_ate, manual_aipw_ate, dml_plr_ate, dml_irm_ate]

#plot
plt.figure(figsize=(8,5))
plt.bar(methods, estimates, color=['orange', 'green', 'blue', 'purple'])
plt.ylabel('ATE Estimate')
plt.title('Comparison of ATE Estimates from Different Methods')
plt.show()
```



4.0.1 Interpretation,

All four methods yield very similar ATE estimates, around 8.7 to 8.9, indicating consistency across both manual and automated (DoubleML-based) implementations. This suggests that the estimated treatment effect is robust to the choice of method, providing confidence in the reliability of the causal inference results. # ATE of Notification Opened Via DoubleML

```
dml_iivm_obj = dml.DoubleMLIIVM(obj_dml_data, ml_g, ml_m, ml_r)
dml_iivm_obj.fit()


ate_estimate = dml_iivm_obj.coef[0]
# Display
print("\nATE Estimate:", ate_estimate)

print(dml_iivm_obj.summary)
```

```
ATE Estimate: 8.39439199546623
                          coef   std err         t          P>|t|    2.5 %  \
notification_opened   8.394392  0.847088  9.909707  3.777532e-23  6.73413

                         97.5 %
notification_opened   10.054654
```

5 Heterogeneous Treatment Effects of Notification Sent

- Estimate group ATEs by gender (M/F) and by tertiles of past_engagement.

```
print("\nGATE by Gender:")
print(gate_gender)
print("\nGATE by Past Engagement:")
print(gate_engagement)
```

```

GATE by Gender:
================== DoubleMLBLP Object ==================

------------------ Fit summary ------------------
                   coef   std err          t          P>|t|    [0.025  \
Group_Female  10.053458  0.246148  40.843166   0.000000e+00  9.571017
Group_Male     7.358292  0.215963  34.072013  1.916407e-254  6.935012

                 0.975]
Group_Female  10.535899
Group_Male     7.781571

GATE by Past Engagement:
================== DoubleMLBLP Object ==================

------------------ Fit summary ------------------
                   coef   std err          t          P>|t|    [0.025  \
Group_Low      7.324093  0.301686  24.277244  3.410174e-130  6.732800
Group_Medium   8.448609  0.271020  31.173416  2.443172e-213  7.917420
Group_High    10.394889  0.278911  37.269595  5.102941e-304  9.848234
```

```
                 0.975]
Group_Low      7.915385
Group_Medium   8.979797
Group_High    10.941543
```

- R-Learner

```
RandomForestRegressor()
```

```
ite_rl = tau_model.predict(X_eval)
eval_data['ite_rl'] = ite_rl
print(eval_data[['ite_rl']].head())
```

```
        ite_rl
6252  5.848759
4684  6.275323
1731  0.974028
4742  4.754528
4521  2.290479
```

- DR-Learner model

```
X_eval = pd.get_dummies(eval_data[X_cols], drop_first=True)
X_eval = X_eval.reindex(columns=X_train.columns, fill_value=0)

ite_dr_eval = tau_model.predict(X_eval)  # tau_model is DR-Learner model
eval_data['ite_dr'] = ite_dr_eval

print(eval_data[['ite_dr']].head())
```

```
        ite_dr
6252  5.848759
4684  6.275323
1731  0.974028
4742  4.754528
4521  2.290479
```

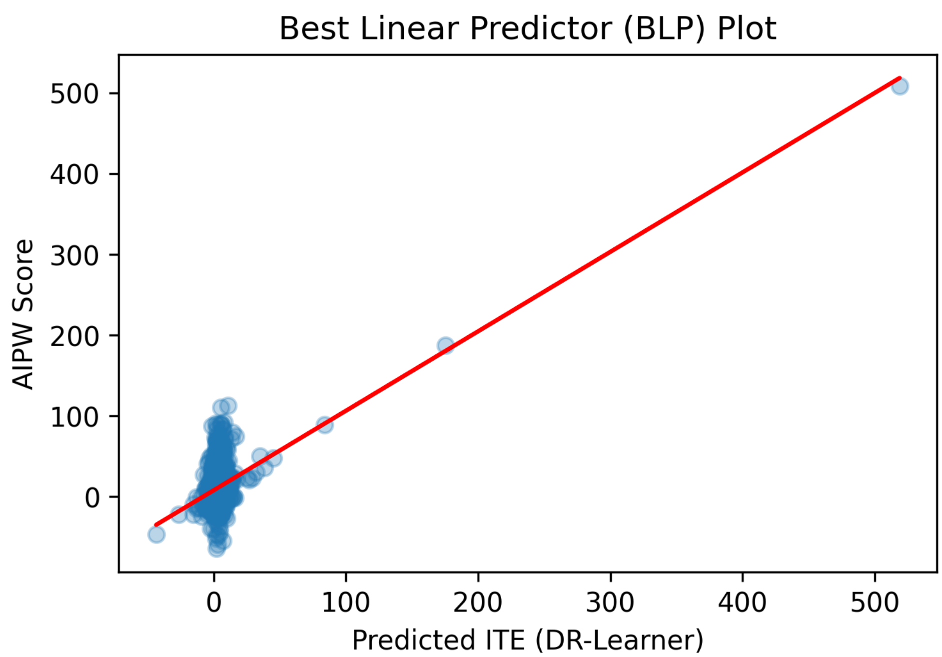
6 HTE Evaluation

```
LinearRegression()
```

```
print("BLP coefficient:", blp_model.coef_[0])
print("BLP intercept:", blp_model.intercept_)

plt.scatter(ite_dr_eval, aipw_scores, alpha=0.3)
plt.plot(ite_dr_eval, blp_model.predict(ite_dr_eval.reshape(-1,1)), color='red')
plt.xlabel("Predicted ITE (DR-Learner)")
plt.ylabel("AIPW Score")
plt.title("Best Linear Predictor (BLP) Plot")
plt.show()
```

```
BLP coefficient: 0.9843571937512781
BLP intercept: 7.944396402807552
```



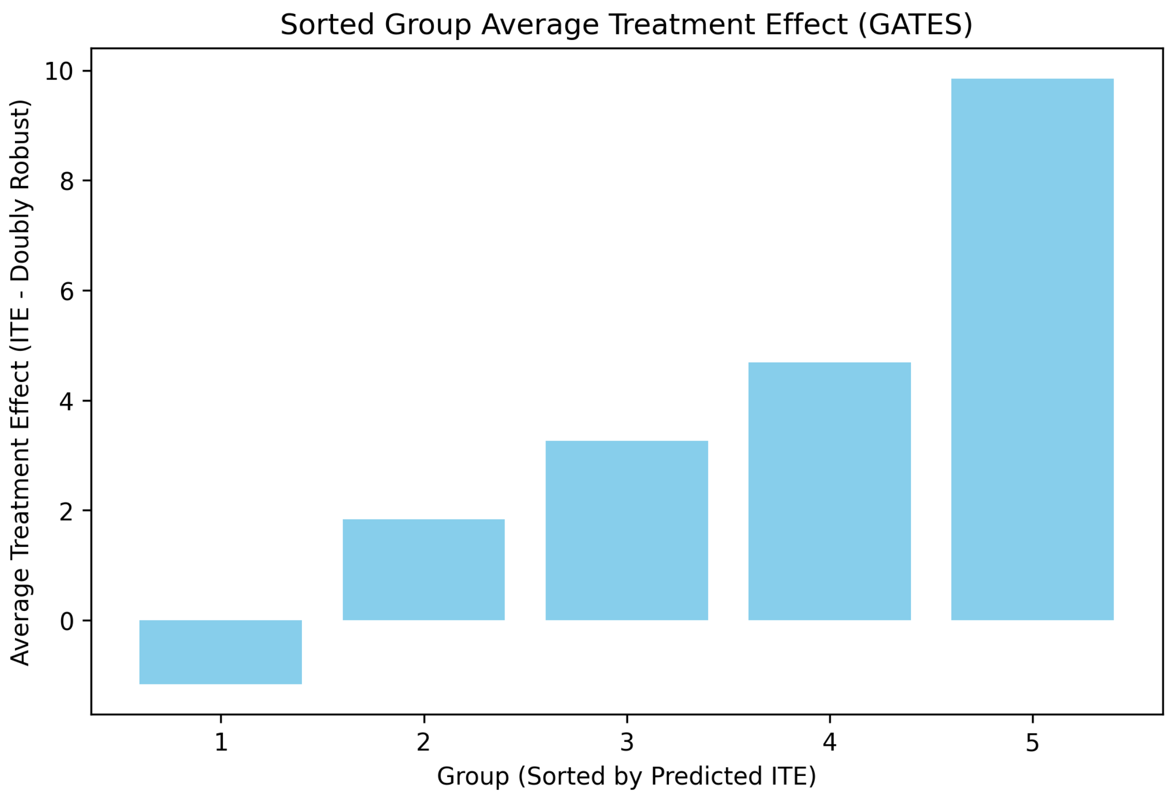

```
print("Samples per group:")
print(counts)
```

```python
print("\nSorted Group Average Treatment Effects (GATES):")
print(gates)

# Plotting the GATES
plt.figure(figsize=(8, 5))
plt.bar(gates.index.astype(str), gates.values, color='skyblue')
plt.xlabel('Group (Sorted by Predicted ITE)')
plt.ylabel('Average Treatment Effect (ITE - Doubly Robust)')
plt.title('Sorted Group Average Treatment Effect (GATES)')
plt.show()
```

```
Samples per group:
group
1    400
2    400
3    400
4    400
5    400
Name: count, dtype: int64

Sorted Group Average Treatment Effects (GATES):
group
1   -1.158407
2    1.832626
3    3.261341
4    4.693296
5    9.852490
Name: ite_dr, dtype: float64
```



7 Optimal Policy Learning

```
# Confusion matrix and report
cm = confusion_matrix(eval_data['optimal_treatment'], policy_preds)
print("Confusion Matrix:")
print(cm)

print("\nClassification Report:")
print(classification_report(eval_data['optimal_treatment'], policy_preds, zero_divisio

# Plot the decision tree
plt.figure(figsize=(10, 7))
plot_tree(policy_tree, feature_names=X_eval.columns, class_names=["No", "Yes"], filled
plt.title("Optimal Treatment Policy Tree")
plt.show()
```

```
Confusion Matrix:
[[ 165   64]
 [ 593 1178]]

Classification Report:
              precision    recall  f1-score   support

           0       0.22      0.72      0.33       229
           1       0.95      0.67      0.78      1771

    accuracy                           0.67      2000
   macro avg       0.58      0.69      0.56      2000
weighted avg       0.86      0.67      0.73      2000
```