HW3_complete_assignment

February 14, 2026

1 اعتماد قابل مصنوعی هوش درس | Trusted Artificial Intelligence

1.1 شماره تمرین ۳ | Homework 3

1.1.1 اجرا قابل و کامل نوتبوک (Q1 تا Q6)

دانشجو / Student: Taha Majlesi (810101504)
دانشگاه / University: University of Tehran, ECE Department
مدرس / Instructor: Dr. Mostafa Tavasolipour

یکسان برچسب‌های - تمرین صورت مطابق سوالات دقیق ترتیب - است: شده طراحی نهایی تصحیح برای نوتبوک این (seed) بازتولیدپذیر اجرای - گزارش خوانایی برای فارسی/انگلیسی دوزبانه متن -)زیربخش و سوال(تمرین قالب با ذخیره‌شده) خروجی‌های + ثابت

1.2 نمره‌دهی نقشه | Grading Map

Question	بخش	Score
اول سوال	Observational vs Interventional Probability	10
دوم سوال	Causal Recourse for Two Individuals	12
سوم سوال	Airline SCM Graph + Modeling + Variance Analysis	20
چهارم سوال	Insulin Causal Effect Estimation	22
پنجم سوال	Complete Causal Recourse Pipeline	20
ششم سوال	Theory from Robust Causal Recourse Paper	16

Total: 100

1.3 بازتولیدپذیری و اولیه تنظیمات | Setup and Reproducibility

```
from __future__ import annotations

import os
import sys
import math
import json
import random
import subprocess
from pathlib import Path
```

```
import numpy as np
import pandas as pd
import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt
import seaborn as sns
import torch
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.preprocessing import StandardScaler

SEED = 0
np.random.seed(SEED)
random.seed(SEED)
torch.manual_seed(SEED)

sns.set_theme(style='whitegrid')

# Resolve project root robustly.
ROOT = Path.cwd().resolve()
while ROOT != ROOT.parent and not (ROOT / 'description' / 'HW3_TAI.pdf').exists():
    ROOT = ROOT.parent
if not (ROOT / 'description' / 'HW3_TAI.pdf').exists():
    raise RuntimeError('Could not locate HW3 project root from current working directory.')

Q5_DIR_CANDIDATES = [ROOT / 'code' / 'q5_codes', ROOT / 'code' / 'Q5_codes']
Q5_DIR = next((p for p in Q5_DIR_CANDIDATES if p.exists()), None)
if Q5_DIR is None:
    raise RuntimeError('Could not locate q5_codes directory.')

if str(Q5_DIR) not in sys.path:
    sys.path.append(str(Q5_DIR))

import data_utils
import recourse
import trainers
import utils
import train_classifiers

DATASET_DIR = ROOT / 'dataset'
OUT_DIR = ROOT / 'output' / 'jupyter-notebook' / 'artifacts'
OUT_DIR.mkdir(parents=True, exist_ok=True)

print('ROOT:', ROOT)
print('Q5_DIR:', Q5_DIR)
print('DATASET_DIR:', DATASET_DIR)
print('Health source:', data_utils.get_health_source_path())
```

```
print('Health source tag:', data_utils.get_health_source_tag())
```

1.4 سوال اول (۱۰ نمره) | Question 1 (10 Points)

DAG: (S →A), (S →Y), (A →Y) with the exact probabilities from the assignment PDF.

Full theoretical note: / کامل نظری شرح این در DAG، متغیر S روی هم و A روی هم Y اثر؛ بنابراین دارد: کرد: استفاده بیز از باید مشاهده‌ای به‌صورت شود. تنظیم مداخله‌ای به‌صورت A اگر حتی هستند همبسته A و Y می‌شود قطع A به ورودی مسیرهای، $do(A = a)$ مداخله برای اما. $P(Y \mid A) = \sum_s P(Y \mid A, S = s)P(S \mid A)$ مخدوش‌گری اثر دقیقا کمیت دو این تفاوت. $P(Y \mid do(A = a)) = \sum_s P(Y \mid A = a, S = s)P(S = s)$ داریم و S می‌دهد. نشان را

1.4.1 سوال اول زیربخش (۵ نمره)

محاسبه‌ی:- (P_X(Y=1|A=N)) - (P_X(Y=1|A=O))

Compute observational conditionals using Bayes + total probability.

```
# Q1 constants extracted from the assignment PDF.
pS_L = 0.49
pS_R = 1 - pS_L

pA_N_given_S = {'L': 0.77, 'R': 0.24}
pA_O_given_S = {'L': 1 - pA_N_given_S['L'], 'R': 1 - pA_N_given_S['R']}

pY1_given_SA = {
    ('L', 'N'): 0.73,
    ('L', 'O'): 0.69,
    ('R', 'N'): 0.93,
    ('R', 'O'): 0.87,
}

# Marginals for A
pA_N = pA_N_given_S['L'] * pS_L + pA_N_given_S['R'] * pS_R
pA_O = 1 - pA_N

# Bayes terms for observational conditionals
pS_L_given_A_N = (pA_N_given_S['L'] * pS_L) / pA_N
pS_R_given_A_N = 1 - pS_L_given_A_N

pS_L_given_A_O = (pA_O_given_S['L'] * pS_L) / pA_O
pS_R_given_A_O = 1 - pS_L_given_A_O

# Observational conditionals
pY1_given_A_N = (
    pY1_given_SA[('L', 'N')] * pS_L_given_A_N
    + pY1_given_SA[('R', 'N')] * pS_R_given_A_N
)
```

```
pY1_given_A_O = (
    pY1_given_SA[('L', 'O')] * pS_L_given_A_O
    + pY1_given_SA[('R', 'O')] * pS_R_given_A_O
)

# Interventional conditionals: cut incoming edges to A
pY1_given_do_A_N = (
    pY1_given_SA[('L', 'N')] * pS_L
    + pY1_given_SA[('R', 'N')] * pS_R
)
pY1_given_do_A_O = (
    pY1_given_SA[('L', 'O')] * pS_L
    + pY1_given_SA[('R', 'O')] * pS_R
)

q1_res = pd.DataFrame(
    [
        {'quantity': 'P(Y=1 | A=N)', 'value': pY1_given_A_N},
        {'quantity': 'P(Y=1 | A=O)', 'value': pY1_given_A_O},
        {'quantity': 'P(Y=1 | do(A=N))', 'value': pY1_given_do_A_N},
        {'quantity': 'P(Y=1 | do(A=O))', 'value': pY1_given_do_A_O},
    ]
)

q1_res
```

1.4.2 نمره) (۵ دوم زیربخش

محاسبه‌ی:- (P_X(Y=1|do(A=N))) - (P_X(Y=1|do(A=O)))

Interventional probabilities are computed with truncated factorization (cut incoming edges to (A)).

نمره‌ای جمع‌بندی / Grading note: جدول q1_res هر چهار کمیت خواسته‌شده را می‌کند. گزارش مستقیم

1.5 نمره) | Question 2 (12 Points) دوم سوال ۱۲)

Given: - $A = [75000, 25000]^T$, $B = [70000, 23800]^T$ - classifier: $h = \text{sgn}(X_1 + 5X_2 - 225000)$

مطلوب. حالت به تصمیم تغییر برای مداخله هزینه کمینه‌سازی هدف:

کامل نظری شرح / Full theoretical note: خطی تصمیم مرز برای $x_1 + 5x_2 = 225000$ و هزینه ℓ_1، کارآمدترین margin افزایش بیشترین تغییر، واحد یک ازای به چون (x_2 (اینجا است بزرگ‌تر ضریب با ویژگی روی مداخله جهت به پروژکشن به‌صورت عمل یعنی است؛ $w = (1, 5)$ نرمال بردار راستای در جهت بهترین، ℓ_2 هزینه برای می‌دهد. را می‌دهد. تغییر را مداخله هندسه مستقیما هزینه معیار که می‌دهد نشان تفاوت این می‌شود. انجام تصمیم مرز سمت

```
W = np.array([1.0, 5.0])
B_TH = 225000.0

individuals = {
```

```
    'A': np.array([75000.0, 25000.0]),
    'B': np.array([70000.0, 23800.0]),
}

def score(x: np.ndarray) -> float:
    return float(W @ x - B_TH)

def min_l1_nonneg_action(x: np.ndarray) -> np.ndarray:
    # minimize |d1|+|d2| subject to d>=0 and W^T(x+d) >= B_TH
    gap = max(0.0, -score(x))
    # best to allocate to feature with largest coefficient per unit L1 cost: X2
    return np.array([0.0, gap / W[1]])

def min_l2_nonneg_action(x: np.ndarray) -> np.ndarray:
    gap = max(0.0, -score(x))
    if gap == 0:
        return np.zeros_like(x)
    return (gap / float(W @ W)) * W

rows = []
for name, x in individuals.items():
    d1 = min_l1_nonneg_action(x)
    d2 = min_l2_nonneg_action(x)
    for metric, d in [('L1-opt', d1), ('L2-opt', d2)]:
        x_cf = x + d
        rows.append(
            {
                'individual': name,
                'metric': metric,
                'x1_old': x[0],
                'x2_old': x[1],
                'delta_x1': d[0],
                'delta_x2': d[1],
                'x1_new': x_cf[0],
                'x2_new': x_cf[1],
                'new_margin': score(x_cf),
                'L1_cost': float(np.abs(d).sum()),
                'L2_cost': float(np.sqrt((d**2).sum())),
            }
        )

q2_res = pd.DataFrame(rows)
q2_res
```

```
# Visual boundary and interventions
x1 = np.linspace(60000, 110000, 300)
x2_boundary = (B_TH - x1) / 5.0
```

```
fig, ax = plt.subplots(figsize=(8, 5))
ax.plot(x1, x2_boundary, 'k--', label='Decision boundary: x1 + 5x2 = 225000')

for name, x in individuals.items():
    ax.scatter(x[0], x[1], s=80, label=f'{name} original')
    d = min_l1_nonneg_action(x)
    x_cf = x + d
    ax.scatter(x_cf[0], x_cf[1], s=80, marker='x', label=f'{name} recourse (L1-opt)')
    ax.arrow(x[0], x[1], d[0], d[1], head_width=200, length_includes_head=True, alpha=0.6)

ax.set_xlabel('X1 (Annual Salary)')
ax.set_ylabel('X2 (Bank Balance)')
ax.set_title('Q2 Recourse Moves to Reach Loan Approval Boundary')
ax.legend(loc='best')
plt.tight_layout()
plt.show()
```

تصحیح برای توضیح / Grading interpretation: - جدول q2_res شامل state، جدید، نهایی margin و هزینه‌ها می‌دهد. نشان فرد هر برای را مداخله بردار و تصمیم مرز نمودار - است.

1.6 سوم سوال (۲۰ نمره) | Question 3 (20 Points)

دنبال ابتدا 1. نوتبوک: این بنابراین نیست؛ موجود تمرین صورت ستون‌های همان با airline خام فایل مخزن، این در اجرا قابل زیربخش‌ها کل تا می‌سازد را SCM با سازگار سنتتیکِ fallback نشود، پیدا اگر 2. می‌گردد. واقعی دیتاست بمانند.

This keeps the full methodology fully runnable for grading.

نویز و علّی والدهای از تابعی متغیر هر هواپیمایی، شرکت در SCM / Full theoretical note: کامل نظری شرح رگرسیون یا واریانس تجزیه با را متغیرها اثرگذاری هم و باشیم داشته علّی تفسیر هم می‌دهد اجازه فرم این است. حساسیت و کرد تفکیک را سود بر غیرمستقیم و مستقیم اثرات می‌توان شد، تعریف علّی ساختار وقتی بسنجیم. خطی نمود. تحلیل منسجم چارچوبی در را عملیاتی هزینه‌های و بازاریابی بودجه بلیت، قیمت به نسبت سود

1.6.1 اول زیربخش (۲ نمره)

networkx با علّی گراف رسم

```
import networkx as nx

AIRLINE_COLS = [
    'Booking_Mode',
    'Marketing_Budget',
    'Website_Visits',
    'Ticket_Price',
    'Tickets_Sold',
    'Sales_Revenue',
    'Operating_Expenses',
```

```
    'Profit',
]

def load_or_simulate_airline_df(seed: int = 0) -> tuple[pd.DataFrame, str, bool]:
    rng = np.random.default_rng(seed)

    candidates = [
        ROOT / 'dataset' / 'airline.csv',
        ROOT / 'dataset' / 'airline_operations.csv',
        ROOT / 'dataset' / 'out_data_2.csv',
        ROOT / 'code' / 'q5_codes' / 'data' / 'airline.csv',
        ROOT / 'code' / 'q5_codes' / 'data' / 'out_data_2.csv',
    ]

    for p in candidates:
        if p.exists():
            try:
                df = pd.read_csv(p)
                if set(AIRLINE_COLS).issubset(df.columns):
                    return df[AIRLINE_COLS].copy(), str(p), False
            except Exception:
                pass

    # Fallback synthetic SCM dataset
    n = 365
    booking = rng.binomial(1, 0.22, size=n)

    marketing = 1200 + 850 * booking + rng.normal(0, 120, size=n)
    website = 12000 + 2.4 * marketing + 2800 * booking + rng.normal(0, 900, size=n)
    ticket_price = 420 + 170 * booking + rng.normal(0, 35, size=n)
    tickets_sold = 1800 + 0.30 * website - 2.0 * ticket_price + 900 * booking + rng.normal(0,␣
    ↪300, size=n)
    tickets_sold = np.clip(tickets_sold, 100, None)
    sales = ticket_price * tickets_sold + rng.normal(0, 40000, size=n)
    op_exp = 900000 + 170 * marketing + 130 * tickets_sold + rng.normal(0, 30000, size=n)
    profit = sales - op_exp

    df = pd.DataFrame(
        {
            'Booking_Mode': booking.astype(bool),
            'Marketing_Budget': marketing,
            'Website_Visits': website,
            'Ticket_Price': ticket_price,
            'Tickets_Sold': tickets_sold,
            'Sales_Revenue': sales,
            'Operating_Expenses': op_exp,
            'Profit': profit,
```

```
        }
    )
    return df, 'synthetic_scm_fallback', True

air_df, air_source, used_fallback = load_or_simulate_airline_df(seed=SEED)
print('Airline source:', air_source)
print('Used synthetic fallback:', used_fallback)
air_df.head()
```

```
# Q3-A: draw the causal graph
G = nx.DiGraph()
G.add_edges_from(
    [
        ('Booking_Mode', 'Marketing_Budget'),
        ('Booking_Mode', 'Website_Visits'),
        ('Booking_Mode', 'Tickets_Sold'),
        ('Booking_Mode', 'Ticket_Price'),
        ('Marketing_Budget', 'Website_Visits'),
        ('Marketing_Budget', 'Operating_Expenses'),
        ('Website_Visits', 'Tickets_Sold'),
        ('Ticket_Price', 'Tickets_Sold'),
        ('Ticket_Price', 'Sales_Revenue'),
        ('Tickets_Sold', 'Sales_Revenue'),
        ('Tickets_Sold', 'Operating_Expenses'),
        ('Sales_Revenue', 'Profit'),
        ('Operating_Expenses', 'Profit'),
    ]
)

plt.figure(figsize=(11, 7))
pos = nx.spring_layout(G, seed=SEED, k=1.25)
nx.draw_networkx(G, pos=pos, arrows=True, node_size=2100, font_size=10)
plt.title('Q3-A Causal Graph (NetworkX)')
plt.axis('off')
plt.show()
```

1.6.2 نمره) (۵ دوم زیربخش

مدلسازی SCM: هر گره به‌صورت تابعی از والدها + نویز (linear structural equations)

```
# Q3-B: fit SCM equations (linear structural functions + additive noise)
parents = {
    'Marketing_Budget': ['Booking_Mode'],
    'Website_Visits': ['Booking_Mode', 'Marketing_Budget'],
    'Ticket_Price': ['Booking_Mode'],
    'Tickets_Sold': ['Booking_Mode', 'Website_Visits', 'Ticket_Price'],
    'Sales_Revenue': ['Ticket_Price', 'Tickets_Sold'],
```

```
    'Operating_Expenses': ['Marketing_Budget', 'Tickets_Sold'],
    'Profit': ['Sales_Revenue', 'Operating_Expenses'],
}

scm_models = {}
scm_noise_stats = []

work_df = air_df.copy()
work_df['Booking_Mode'] = work_df['Booking_Mode'].astype(int)

for node, pa in parents.items():
    X = work_df[pa].values
    y = work_df[node].values
    model = LinearRegression().fit(X, y)
    pred = model.predict(X)
    noise = y - pred

    scm_models[node] = model
    scm_noise_stats.append(
        {
            'node': node,
            'parents': ', '.join(pa),
            'r2': float(model.score(X, y)),
            'noise_mean': float(noise.mean()),
            'noise_std': float(noise.std(ddof=0)),
        }
    )

q3b_stats = pd.DataFrame(scm_noise_stats).sort_values('node')
q3b_stats
```

1.6.3 سوم زیربخش (۳ نمره)

واریانس Profit در Operating_Expenses و Sales_Revenue مستقیم سهم و سود واریانس

1.6.4 چهارم زیربخش (۵ نمره)

سود تغییرپذیری در سیستم عامل مهم‌ترین شناسایی (feature importance)

```
# Q3-C and Q3-D: variance decomposition and dominant factors
# Direct parent decomposition for Profit = beta1*Sales + beta2*Operating + noise
profit_model = scm_models['Profit']
beta_sales, beta_op = profit_model.coef_

sales = work_df['Sales_Revenue'].to_numpy()
op = work_df['Operating_Expenses'].to_numpy()
profit = work_df['Profit'].to_numpy()
```

```
var_profit = float(np.var(profit, ddof=0))
var_sales = float(np.var(sales, ddof=0))
var_op = float(np.var(op, ddof=0))
cov_sales_op = float(np.cov(sales, op, ddof=0)[0, 1])

# Shapley-style split of covariance term equally
contrib_sales = beta_sales**2 * var_sales + beta_sales * beta_op * cov_sales_op
contrib_op = beta_op**2 * var_op + beta_sales * beta_op * cov_sales_op

q3c = pd.DataFrame(
    {
        'component': ['Var(Profit)', 'Sales contribution', 'Operating contribution'],
        'value': [var_profit, contrib_sales, contrib_op],
        'share_of_profit_var': [1.0, contrib_sales / var_profit, contrib_op / var_profit],
    }
)

# Q3-D: global factor ranking via standardized linear model to Profit
feature_cols = [
    'Booking_Mode', 'Marketing_Budget', 'Website_Visits',
    'Ticket_Price', 'Tickets_Sold', 'Sales_Revenue', 'Operating_Expenses'
]
Xf = work_df[feature_cols].astype(float)
yf = work_df['Profit'].astype(float)

Xf_std = (Xf - Xf.mean()) / Xf.std(ddof=0)
model_all = LinearRegression().fit(Xf_std, yf)
importance = pd.DataFrame({'feature': feature_cols, 'abs_std_coef': np.abs(model_all.coef_)})
importance = importance.sort_values('abs_std_coef', ascending=False)

print('Q3-C: Direct decomposition of profit variance')
display(q3c)
print('Q3-D: Dominant system factors (standardized effect magnitude)')
display(importance)
```

1.6.5 پنجم زیربخش (۵ نمره)

تمرین صورت در داده‌شده مقادیر با جدید سال اول روز تحلیل

```
# Q3-E: First day of new year analysis using provided table values
new_year_obs = {
    'Booking_Mode': True,
    'Marketing_Budget': 2079.01,
    'Website_Visits': 21110,
    'Ticket_Price': 700.47,
    'Tickets_Sold': 7987,
```

```
    'Sales_Revenue': 5594652.87,
    'Operating_Expenses': 4495588.74,
    'Profit': 1099064.13,
}

prev_first_day_profit = float(work_df.iloc[0]['Profit'])
delta_profit = new_year_obs['Profit'] - prev_first_day_profit
trend = 'increased' if delta_profit > 0 else 'decreased'

q3e = pd.DataFrame(
    [
        {'metric': 'Previous year first-day profit', 'value': prev_first_day_profit},
        {'metric': 'New year first-day observed profit', 'value': new_year_obs['Profit']},
        {'metric': 'Delta', 'value': delta_profit},
    ]
)

print(f'Profit {trend} compared to previous-year first day (delta={delta_profit:,.2f}).')
q3e
```

1.7 چهارم سوال ۲۲) (نمره | Question 4 (22 Points)

اثر تخمین هدف: insulin بر blood_glucose از استفاده با logistic regression.

Full theoretical note: / کامل نظری شرح هدف Q4 اثر برآورد علّی خون قند بر انسولین است. اگر Z و W از استفاده یا Z حذف است. علّی برآوردگر $E_{W,Z}E[Y \mid t, W, Z]$ آنگاه کنند، مسدود را backdoor مسیرهای همه اولویت دلیل و است علّی استنتاج در (adjustment) تنظیم‌گری منطق دقیقا این می‌شود. بایاس به منجر $E[Y \mid t]$ است. سؤال این در چندمتغیره برآوردگر

1.7.1 محاسباتی زیربخش ۱۲) (نمره

کمیت: سه محاسبه - (E_{W,Z}E[Y|t,W,Z]) - (E_W E[Y|t,W]) - (E[Y|t])

```
health_df = data_utils.load_health_dataframe().copy()
health_df['high_glucose'] = (health_df['blood_glucose'] >= health_df['blood_glucose'].median()).
    ↪astype(int)

# Define insulin intervention grid in observed range
q = np.linspace(0.1, 0.9, 9)
t_grid = np.quantile(health_df['insulin'].to_numpy(), q)

# Fit logistic models for the three expressions
m1 = LogisticRegression(max_iter=2000).fit(health_df[['insulin', 'age', 'blood_pressure']],␣
    ↪health_df['high_glucose'])
m2 = LogisticRegression(max_iter=2000).fit(health_df[['insulin', 'age']],␣
    ↪health_df['high_glucose'])
m3 = LogisticRegression(max_iter=2000).fit(health_df[['insulin']], health_df['high_glucose'])
```

```
def avg_prob_m1(t: float) -> float:
    X = health_df[['insulin', 'age', 'blood_pressure']].copy()
    X['insulin'] = t
    return float(m1.predict_proba(X)[:, 1].mean())

def avg_prob_m2(t: float) -> float:
    X = health_df[['insulin', 'age']].copy()
    X['insulin'] = t
    return float(m2.predict_proba(X)[:, 1].mean())

def avg_prob_m3(t: float) -> float:
    X = pd.DataFrame({'insulin': np.full(len(health_df), t)})
    return float(m3.predict_proba(X)[:, 1].mean())

q4_res = pd.DataFrame(
    {
        'insulin_t': t_grid,
        'E_WZ_E[Y|t,W,Z]': [avg_prob_m1(t) for t in t_grid],
        'E_W_E[Y|t,W]': [avg_prob_m2(t) for t in t_grid],
        'E[Y|t]': [avg_prob_m3(t) for t in t_grid],
    }
)
q4_res
```

```
plt.figure(figsize=(8.5, 5))
plt.plot(q4_res['insulin_t'], q4_res['E_WZ_E[Y|t,W,Z]'], marker='o', label='E_WZ E[Y|t,W,Z]')
plt.plot(q4_res['insulin_t'], q4_res['E_W_E[Y|t,W]'], marker='o', label='E_W E[Y|t,W]  (causal␣
↪estimator)')
plt.plot(q4_res['insulin_t'], q4_res['E[Y|t]'], marker='o', label='E[Y|t]')
plt.xlabel('Insulin intervention level t')
plt.ylabel('Predicted P(high_glucose=1)')
plt.title('Q4 Estimators vs Insulin')
plt.legend()
plt.tight_layout()
plt.show()

q4_summary = pd.DataFrame(
    [
        {
            'estimator': 'E_WZ E[Y|t,W,Z]',
            'approx_effect (last-first)': float(q4_res['E_WZ_E[Y|t,W,Z]'].iloc[-1] -␣
↪q4_res['E_WZ_E[Y|t,W,Z]'].iloc[0]),
        },
        {
            'estimator': 'E_W E[Y|t,W] (causal)',
```

```
        'approx_effect (last-first)': float(q4_res['E_W_E[Y|t,W]'].iloc[-1] - q4_res['E_W_E[Y|t,W]'].
↪iloc[0]),
    },
    {
        'estimator': 'E[Y|t]',
        'approx_effect (last-first)': float(q4_res['E[Y|t]'].iloc[-1] - q4_res['E[Y|t]'].iloc[0]),
    },
]
)
q4_summary
```

1.7.2 نمره) (۱۰ علّی اثر اثبات زیربخش

(Z) روی شرط‌گذاری - است. لازم confounding کنترل برای (سن) (W) روی تنظیم - ۴: سوال DAG به توجه با کند. وارد bias می‌تواند پس‌مداخله‌ای) (فرزند

پس estimator علّی مناسب: (E_W E[Y|t,W]).

1.8 پنجم سوال ۲۰) (نمره | Question 5 (20 Points)

شده داده قرار دیتاست با Causal Algorithmic Recourse و Nearest Counterfactual Explanation مقایسه‌ی پوشه: در /Users/tahamajs/Documents/uni/truthlyAI/HomeWorks/HW3/dataset

1.8.1 اول زیربخش

ویژگی‌های تنها 1. که: نمایید تکمیل طوری را process_health_data تابع و نموده مراجعه data-utils.py فایل به از blood_pressure, insulin, blood_glucose مقادیر 2. باشند. actionable، blood_glucose و insulin نروند. فراتر دیتاست حداقل/حداکثر

```
# Q5-A: verify actionable features and feasible limits
X_health, Y_health, constraints = data_utils.process_health_data()

q5a = {
    'n_samples': int(X_health.shape[0]),
    'n_features': int(X_health.shape[1]),
    'actionable_indices': constraints['actionable'],
    'feature_order': ['age', 'insulin', 'blood_glucose', 'blood_pressure'],
    'limits_shape': tuple(constraints['limits'].shape),
}

q5a
```

1.8.2 دوم زیربخش

کنید. گزارش را محاسبه‌شده هزینه‌ی و کنید اجرا ناسالم فرد ۱۰ به‌ازای را main.py

می‌شود. گزارش فرد ۱۰ همان روی baseline SCM-off / nearest style با بخش این معادل نوتبوک، این در

```
# Q5-B and Q5-E: run matched SCM-off vs SCM-on for 10 unhealthy individuals
cmd = [sys.executable, str(Q5_DIR / 'run_q5_assignment.py'), '--seed', '0', '--nexplain', '10']
subprocess.run(cmd, cwd=str(Q5_DIR), check=True)

summary_path = Q5_DIR / 'results' / 'q5_diabetes_summary.csv'
per_inst_path = Q5_DIR / 'results' / 'q5_diabetes_per_instance.csv'
example_path = Q5_DIR / 'results' / 'q5_diabetes_example.csv'

q5_summary = pd.read_csv(summary_path)
q5_per_instance = pd.read_csv(per_inst_path)
q5_example = pd.read_csv(example_path)

print('Q5 summary (SCM off vs on):')
display(q5_summary)
print('Q5 one-instance comparison:')
display(q5_example)
print('Q5 per-instance comparison (first 10 rows):')
display(q5_per_instance.head(10))
```

1.8.3 سوم زیربخش

به مراجعه با scm.py، کلاس Health_SCM را کامل کنید به‌طوری که: - insulin و blood_glucose actionable باشند. و Age - blood_pressure constant features باشند.

1.8.4 چهارم زیربخش

ضرایب به توجه با SCM، تابع get_Jacobian را کامل کنید تا ژاکوبین SCM خروجی داده شود.

```
# Q5-C and Q5-D: inspect Health_SCM and Jacobian
scmm = utils.get_scm('lin', 'health')
J = scmm.get_Jacobian()

print('Actionable features in Health_SCM:', scmm.actionable)
print('Soft-intervention flags:', scmm.soft_interv)
print('SCM coefficients:')
print('  w21=', scmm.w21, 'w31=', scmm.w31, 'w32=', scmm.w32, 'w42=', scmm.w42, 'w43=',␣
↪scmm.w43)
print('Jacobian:')
print(J)

plt.figure(figsize=(5.5, 4.5))
sns.heatmap(J, annot=True, fmt='.3f', cmap='Blues',
            xticklabels=['age','insulin','blood_glucose','blood_pressure'],
            yticklabels=['age','insulin','blood_glucose','blood_pressure'])
plt.title('Q5-D Health_SCM Jacobian')
plt.tight_layout()
plt.show()
```

1.8.5 پنجم زیربخش

می‌شود. انجام SCM-on با مجدد اجرای و شده حذف utils.py در get_scm بخش های Comment

باشد. مستقیم مقایسه تا شده گزارش فرد ۱۰ همان روی SCM-off کنار SCM-on خروجی نوتبوک، این در

1.8.6 ششم زیربخش

کنید. مقایسه را (SCM-on) پنجم بخش و (SCM-off) دوم بخش هزینه‌های

Grading note: - q5_summary: روش دو هزینه/اعتبار کلّی مقایسه‌ی - q5_example: فرد یک کامل مقایسه‌ی نمونه (features + actions + costs) - q5_per_instance: فرد ۱۰ همه‌ی برای سطری مقایسه

1.8.7 کامل) نمره برای (اختیاری تکمیلی تحلیل

حسب بر پایداری تحلیل و بهینه‌سازی robustness radius (epsilon)

```
# Q5 optimization: epsilon sweep for linear recourse (SCM off/on)
np.random.seed(0)
torch.manual_seed(0)

X, Y, cons = data_utils.process_data('health')
X_train, Y_train, X_test, Y_test = data_utils.train_test_split(X, Y)

model_path = Q5_DIR / 'models' / 'health_ERM_lin_s0.pth'
if not model_path.exists():
    _ = train_classifiers.train('health', 'ERM', 'lin', utils.get_train_epochs('health', 'lin', 'ERM'), 0,␣
    ↪0, save_model=True)

model = trainers.LogisticRegression(X_train.shape[-1], actionable_features=cons['actionable'],␣
    ↪actionable_mask=False)
model.load_state_dict(torch.load(model_path, map_location='cpu'))
model.set_max_mcc_threshold(X_train, Y_train)

id_neg = model.predict(X_test) == 0
X_neg = X_test[id_neg]
idx = np.random.choice(np.arange(X_neg.shape[0]), size=min(10, X_neg.shape[0]),␣
    ↪replace=False)
X_exp = X_neg[idx]

def eval_eps(eps: float, scm_on: bool):
    w, b = model.get_weights()
    scm_obj = utils.get_scm('lin', 'health') if scm_on else None
    Jw = w if scm_obj is None else scm_obj.get_Jacobian().T @ w
    dual_norm = np.sqrt(Jw.T @ Jw)
    explainer = recourse.LinearRecourse(w, b + dual_norm * eps)
    _, valids, costs, _, _ = recourse.causal_recourse(X_exp, explainer, cons, scm=scm_obj,␣
    ↪verbose=False)
```

```
    valids = np.asarray(valids).astype(bool)
    costs = np.asarray(costs)
    return float(valids.mean()), float(costs[valids].mean()) if valids.any() else np.nan

rows = []
for eps in [0.0, 0.1, 0.2]:
    for scm_on in [False, True]:
        vr, vc = eval_eps(eps, scm_on)
        rows.append({'epsilon': eps, 'method': 'SCM-on' if scm_on else 'SCM-off', 'valid_rate': vr,
    ↪'valid_cost': vc})

q5_eps = pd.DataFrame(rows)

fig, axes = plt.subplots(1, 2, figsize=(11, 4))
sns.lineplot(data=q5_eps, x='epsilon', y='valid_rate', hue='method', marker='o', ax=axes[0])
sns.lineplot(data=q5_eps, x='epsilon', y='valid_cost', hue='method', marker='o', ax=axes[1])
axes[0].set_title('Validity vs epsilon')
axes[1].set_title('Valid cost vs epsilon')
fig.tight_layout()
plt.show()

q5_eps
```

1.9 شش سوال ۱۶) (نمره | Question 6 (16 Points)

Paper: On the Adversarial Robustness of Causal Algorithmic Recourse

Full theoretical note: حضور در که است این مقاله کلیدی نتیجه SCM، ورودی‌ها عدم‌قطعیت / کامل نظری شرح margin shift یک به robust recourse شرط بنابراین می‌شود؛ منتقل مداخله فضای به علّی Jacobian طریق از (در تجربی هم و اثبات) قابل خطی مدل (برای است نظری هم موضوع این می‌گردد. تبدیل $\|J^T w\|$ به وابسته شود: پرهزینه‌تر می‌تواند غیرعلّی recourse چرا می‌گوید ما به همچنین می‌شود). مشاهده هزینه-اعتبار نمودارهای می‌گیرد. نادیده را مداخله زنجیره‌ای اثرات چون

1.9.1 اول زیربخش ۸) (نمره

می‌شود؟ تضمین SCM و classifier در robustness شرایطی چه تحت

- classifier خطی یا locally linear
- SCM مشتق‌پذیر و درست‌مشخص‌شده
- صریح قیود و محدب مداخله مجموعه
- uncertainty bounded مثل) ||delta||_2 <= epsilon)
- حل robust با margin shift مناسب دوگان

1.9.2 دوم زیربخش ۸) (نمره

شهود Proposition 4 و معادله :(5)

برای score و خطی SCM Jacobian :داریم w^T(x+Ja) >= b + epsilon * ||J^T w||_*

شود. عبور علّی propagation و اغتشاش بدترین با متناسب حاشیه‌ای باید نیست؛ کافی nominal تصمیم مرز یعنی

1.10 تصحیح برای نهایی خروجی‌های | Export Artifacts for Grading

```
q1_res.to_csv(OUT_DIR / 'q1_results.csv', index=False)
q2_res.to_csv(OUT_DIR / 'q2_results.csv', index=False)
q3b_stats.to_csv(OUT_DIR / 'q3_scm_fit_stats.csv', index=False)
q3c.to_csv(OUT_DIR / 'q3_variance_decomposition.csv', index=False)
importance.to_csv(OUT_DIR / 'q3_factor_importance.csv', index=False)
q4_res.to_csv(OUT_DIR / 'q4_estimators_curve.csv', index=False)
q4_summary.to_csv(OUT_DIR / 'q4_estimators_summary.csv', index=False)
q5_summary.to_csv(OUT_DIR / 'q5_summary.csv', index=False)
q5_per_instance.to_csv(OUT_DIR / 'q5_per_instance.csv', index=False)
q5_example.to_csv(OUT_DIR / 'q5_example.csv', index=False)
q5_eps.to_csv(OUT_DIR / 'q5_epsilon_sweep.csv', index=False)

print('Saved notebook artifacts under:', OUT_DIR)
for p in sorted(OUT_DIR.glob('*.csv')):
    print('-', p.name)
```

1.11 تحویل نهایی چک‌لیست | Final Grading Checklist

- ☒ Q1 احتمالاتی کمیت چهار (obs + do)
- ☒ Q2 برای هزینه و بهینه حالت A و B
- ☒ Q3 زیربخش پنج (graph, SCM, variance, dominant factor, first-day analysis)
- ☒ Q4 سه estimator + تشخیص estimator علّی
- ☒ Q5 نمونه‌محور مقایسه + ۶ تا ۱ زیربخش‌های
- ☒ Q6 کامل نظری پاسخ دو
- ☒ CSV artifacts exported under output/jupyter-notebook/artifacts

1.12 سریع اجرای دستورات | Quick Re-run Commands

```
source /Users/tahamajs/Documents/uni/venv/bin/activate
cd /Users/tahamajs/Documents/uni/truthlyAI/HomeWorks/HW3
jupyter lab code/HW3_complete_assignment.ipynb
```