

# HW3\_complete\_assignment

February 13, 2026

## 1 Trusted Artificial Intelligence | اعتماد قابل مصنوعی هوش درس 1

### 1.1 Homework 3 | ۳ شماره تمرین

#### 1.1.1 (Q1 تا Q6) اجرا قابل و کامل نوتبوک

Student: Taha Majlesi (810101504) / دانشجو

University: University of Tehran, ECE Department / دانشگاه

Instructor: Dr. Mostafa Tavasolipour / مدرس

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 Grading Map | نمره‌دهی نقشه

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 Setup and Reproducibility | بازتولیدپذیری و اولیه تنظیمات

```
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 (۱۰ اول سوال | Question 1 (10 Points))

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

##### 1.4.1 (۵ اول زیربخش | Question 1.1 (5 Points))

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

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)).

می‌کند. گزارش مستقیم را خواسته‌شده کمیت چهار هر q1\_res جدول / Grading note: نمره‌ای جمع‌بندی

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

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:

```
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:  $x_1 + 5x_2 = 225000$ ')

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()

```

نهایی margin و هزینه‌ها، جدید، state شامل q2\_res جدول - / Grading interpretation: توضیح می‌دهد. نشان فرد هر برای را مداخله بردار و تصمیم مرز نمودار - است.

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

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 سستیک fallback نشود، پیدا اگر 2. می‌گردد. واقعی دیتاست بمانند.

This keeps the full methodology fully runnable for grading.

#### 1.6.1 نمره (۲) اول زیربخش

networkx با علی 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

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

```

# 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 (نمره ۳) سوم زیربخش

Profit واریانس در Operating\_Expenses و Sales\_Revenue مستقیم سهم و سود واریانس

### 1.6.4 (نمره ۵) چهارم زیربخش

(feature importance) سود تغییرپذیری در سیستم عامل مهم‌ترین شناسایی

```

# 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)

```

#### نمره (۵) پنجم زیربخش 1.6.5

تمرین صورت در داده شده 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 Question 4 (22 Points) | نمره (۲۲) چهارم سوال

logistic regression. از استفاده با blood\_glucose بر insulin اثر تخمین هدف:

#### 1.7.1 نمره (۱۲) محاسباتی زیربخش

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

```

### نمره (۱۰) علی اثر اثبات زیربخش 1.7.2

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

(E\_W E[Y|t,W]). مناسب: علی estimator پس

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

شده داده قرار دیتاست با Causal Algorithmic Recourse و Nearest Counterfactual Explanation مقایسه‌ی  
در پوشه: /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 insulin, blood\_glucose, actionable مقادیر 2. باشند. blood\_pressure, insulin, blood\_glucose  
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

insulin و blood\_glucose actionable - که: به طوری کنید کامل را Health\_SCM کلاس، scm.py، به مراجعه با باشند. Age و blood\_pressure constant features - باشند.

### 1.8.4 چهارم زیربخش

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

```
# 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

می شود. انجام SCM-on با مجدد اجرای و شده حذف در 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 هزینه های

Grading note: - q5\_summary: روش دو هزینه/اعتبار کلی مقایسه ی - q5\_example: فرد یک کامل مقایسه ی  
فرد ۱۰ همه ی برای سطری مقایسه: q5\_per\_instance - (features + actions + costs) نمونه

### 1.8.7 کامل (نمره برای (اختیاری تکمیلی تحلیل

(epsilon) robustness radius حسب بر پایداری تحلیل و بهینه سازی

```
# 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 Question 6 (16 Points) | نمره (۱۶ ششم سوال)

Paper: On the Adversarial Robustness of Causal Algorithmic Recourse

#### 1.9.1 نمره (۸ اول زیربخش)

می‌شود؟ تضمین SCM و classifier در robustness شرایطی چه تحت

- classifier یا locally linear خطی
- مشتق‌پذیر و درست مشخص شده SCM
- صریح قیود و محدب مداخله مجموعه
- uncertainty bounded (مثل  $||\delta||_2 \leq \epsilon$ )
- مناسب دوگان margin shift با robust حل

#### 1.9.2 نمره (۸ دوم زیربخش)

(5): معادله و Proposition 4 شهود

برای score خطی و SCM Jacobian داریم:  $w^T(x+J_a) \geq b + \epsilon * ||J^T w||_*$

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

### 1.10 Export Artifacts for Grading | تصحیح برای نهایی خروجی‌های

```
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 Final Grading Checklist | تحویل نهایی چک‌لیست

- ☑ Q1 (obs + do) کمیت چهار
- ☑ Q2 A و B هزینه و بهینه حالت



- ☒ Q3 زیربخش پنج (graph, SCM, variance, dominant factor, first-day analysis)
- ☒ Q4 علی estimator + تشخیص سه
- ☒ Q5 نمونه محور مقایسه + ۶ تا ۱ زیربخش های
- ☒ Q6 کامل نظری پاسخ دو
- ☒ CSV artifacts exported under output/jupyter-notebook/artifacts

### 1.12 Quick Re-run Commands | سریع اجرای دستورات

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