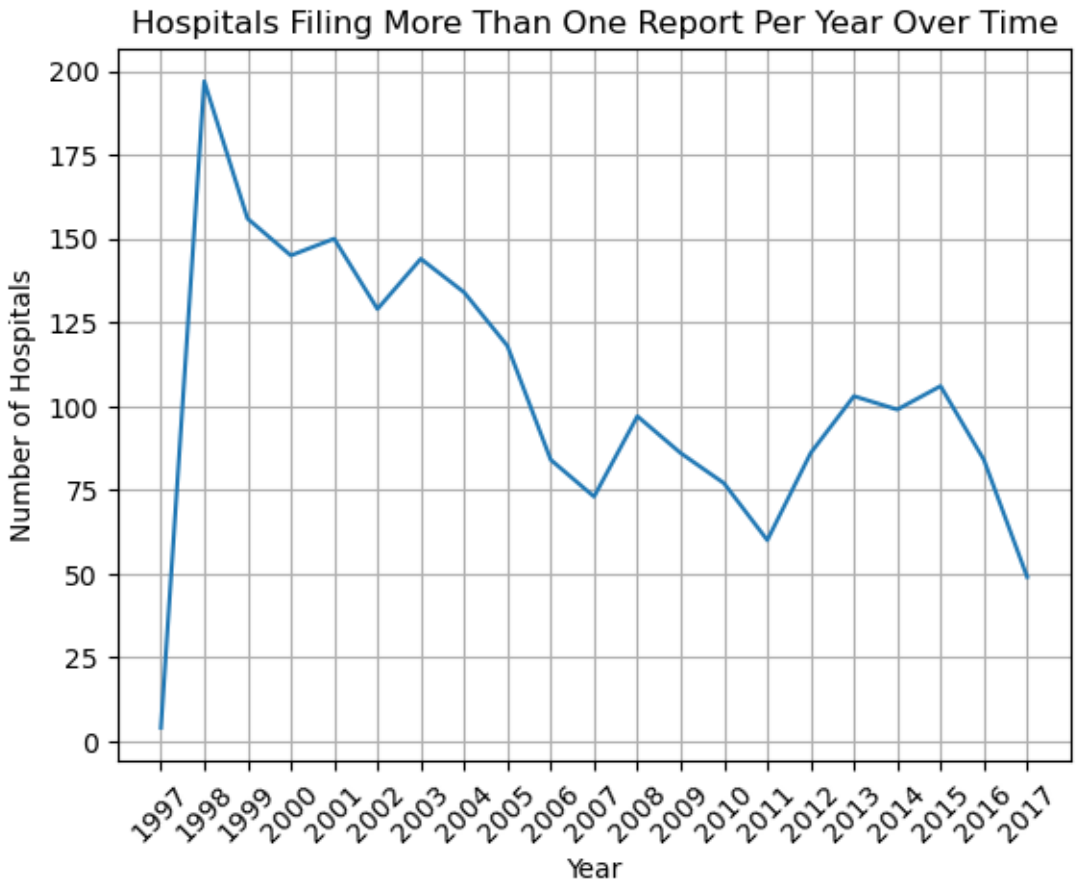
Question 1

	provider_number	year	report_count
0	10001	1997	1
1	10001	1998	1
2	10001	1999	1
3	10001	2000	1
4	10001	2001	1
...	...	...	...
121772	673063	2016	1
121773	673063	2017	1
121774	673064	2016	1
121775	673064	2017	1
121776	673065	2016	1

[121777 rows x 3 columns]



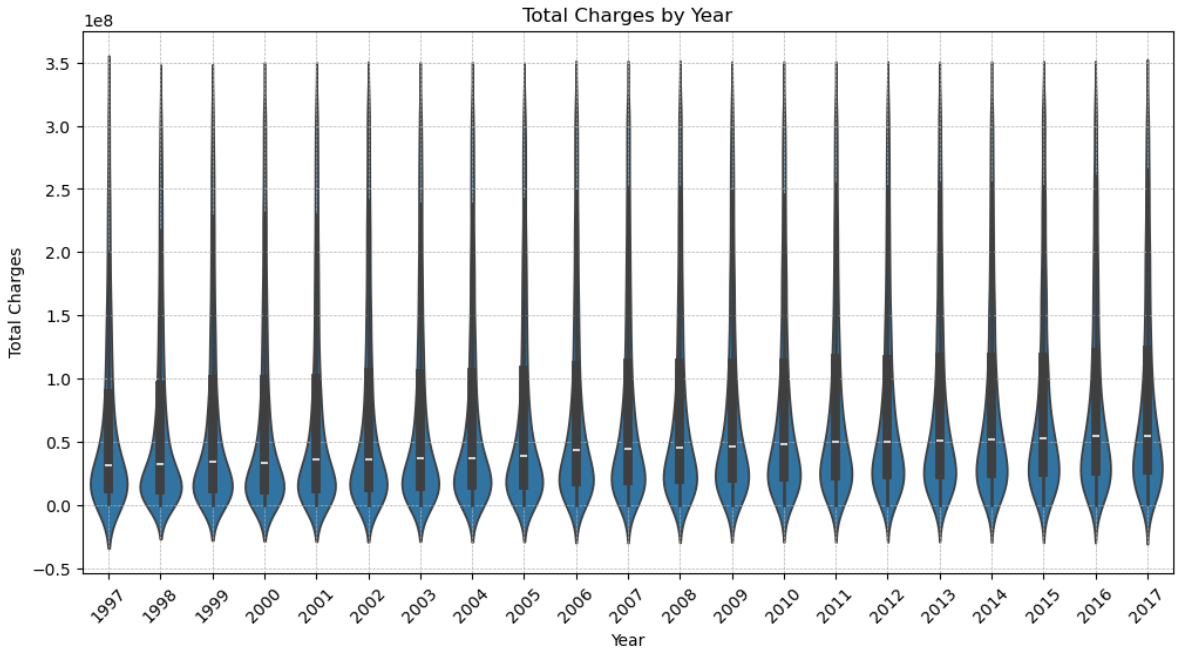

```
#Question 2

#Removing duplicate reports
unique_hospitals = HCRIS.drop_duplicates(subset=['provider_number', 'year'])

#Count of number of unique hospital IDs
unique_hospital_count = unique_hospitals['provider_number'].nunique()

print(f"Number of Unique Hospital IDs: {unique_hospital_count}")
```

```
Number of Unique Hospital IDs: 9323
```

```
#Question 3
HCRIS['tot_charges'] = pd.to_numeric(HCRIS['tot_charges'], errors='coerce')
```

```
# Convert tot_charges to numeric
HCRIS['tot_charges'] = pd.to_numeric(HCRIS['tot_charges'], errors='coerce')

#Remove rows with missing charges or years, negative values, and outliers
charges_by_year = HCRIS[['year', 'tot_charges']].dropna()
charges_by_year = charges_by_year[charges_by_year['tot_charges'] >= 0]

# Display summary statistics to find  cutoff values
summary_stats = charges_by_year['tot_charges'].describe()
print(summary_stats)

#creating upper bound limit
upper_bound = summary_stats['75%'] if '75%' in summary_stats else summary_stats['max']
charges_by_year = charges_by_year[charges_by_year['tot_charges'] <= upper_bound]

# Plot violin plot
plt.figure(figsize=(12, 6))
sns.violinplot(x='year', y='tot_charges', data=charges_by_year)
plt.title("Total Charges by Year")
plt.xlabel("Year")
plt.ylabel("Total Charges")
plt.xticks(rotation=45)
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.show()
```

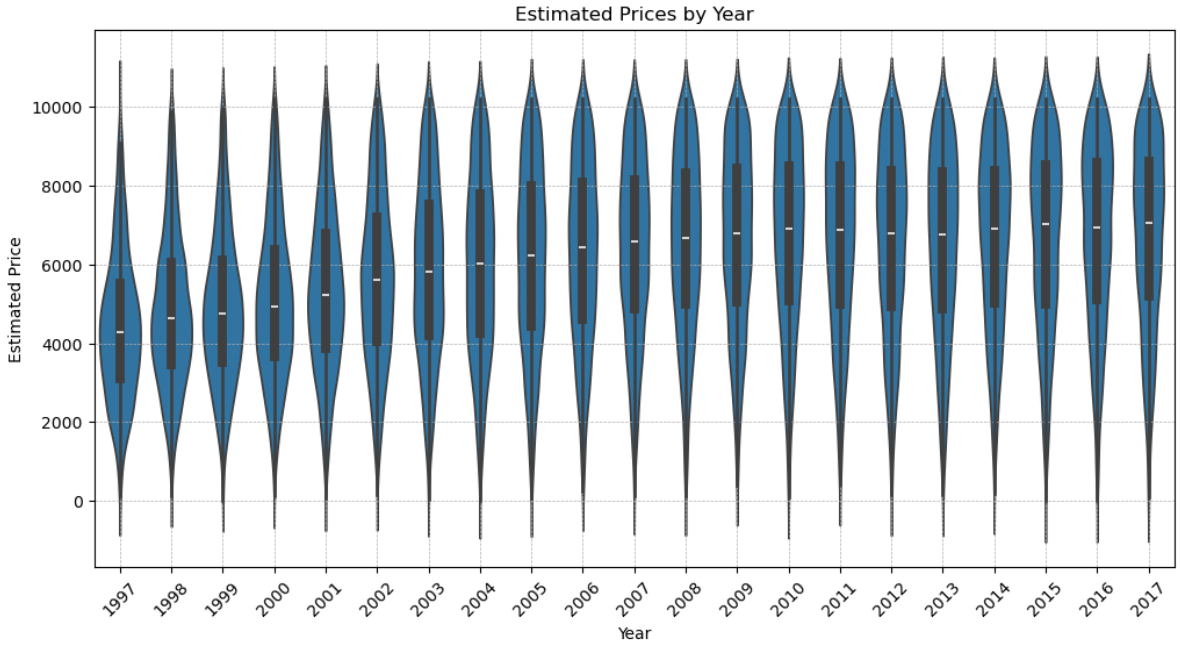
```
count    1.192340e+05
mean     3.226221e+08
std      6.867871e+08
min      1.000000e+00
25%      2.456732e+07
50%      7.782326e+07
75%      3.215973e+08
max      1.863371e+10
Name: tot_charges, dtype: float64
```




```
#Question 4

#Converting columns to numeric
numeric_columns = [
    'tot_discounts', 'tot_charges', 'ip_charges', 'icu_charges', 'ancillary_charges',
    'tot_mcare_payment', 'tot_discharges', 'mcare_discharges'
]
HCRIS[numeric_columns] = HCRIS[numeric_columns].apply(pd.to_numeric, errors='coerce')


# Remove missing values
hcris_clean = HCRIS[['year'] + numeric_columns].dropna()

# Calculate estimated price based on the formula
discount_factor = 1 - hcris_clean['tot_discounts'] / hcris_clean['tot_charges']
price_num = (hcris_clean['ip_charges'] + hcris_clean['icu_charges'] + hcris_clean['ancillary_
price_denom = hcris_clean['tot_discharges'] - hcris_clean['mcare_discharges']
hcris_clean['estimated_price'] = price_num / price_denom

#removing outliers and negatives
hcris_clean = hcris_clean[hcris_clean['estimated_price'] > 0]
summary_stats = hcris_clean['estimated_price'].describe()
print(summary_stats)
```

```
upper_bound = summary_stats['75%'] if '75%' in summary_stats else summary_stats['max']
hcris_clean = hcris_clean[hcris_clean['estimated_price'] <= upper_bound]

#Plot violin plot
plt.figure(figsize=(12, 6))
sns.violinplot(x='year', y='estimated_price', data=hcris_clean)
plt.title("Estimated Prices by Year")
plt.xlabel("Year")
plt.ylabel("Estimated Price")
plt.xticks(rotation=45)
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
plt.show()
```

```
count    5.882500e+04
mean              inf
std               NaN
min      1.953267e+00
25%      4.789842e+03
50%      7.109387e+03
75%      1.022331e+04
max               inf
Name: estimated_price, dtype: float64
```




```
#Q5
hcris_2012 = HCRIS[HCRIS['year'] == 2012]

#Calculating estimated price for 2012
hcris_2012['discount_factor'] = 1 - hcris_2012['tot_discounts'] / hcris_2012['tot_charges']
hcris_2012['price_num'] = (
    (hcris_2012['ip_charges'] + hcris_2012['icu_charges'] + hcris_2012['ancillary_charges'])
    * hcris_2012['discount_factor'] - hcris_2012['tot_mcare_payment'])
hcris_2012['price_denom'] = hcris_2012['tot_discharges'] - hcris_2012['mcare_discharges']
hcris_2012['price'] = hcris_2012['price_num'] / hcris_2012['price_denom']

#NA payments
hcris_2012['hvbp_payment'] = hcris_2012['hvbp_payment'].fillna(0)
hcris_2012['hrrp_payment'] = hcris_2012['hrrp_payment'].fillna(0).abs()

#Defining penalty
hcris_2012['penalty'] = (hcris_2012['hvbp_payment'] - hcris_2012['hrrp_payment'] < 0).astype

# Calculate average price for penalized vs non-penalized hospitals
mean_penalized = round(hcris_2012.loc[hcris_2012['penalty'] == 1, 'price'].mean(), 2)
mean_non_penalized = round(hcris_2012.loc[hcris_2012['penalty'] == 0, 'price'].mean(), 2)

print(f"Average price for penalized hospitals: {mean_penalized}")
```

```
print(f"Average price for non-penalized hospitals: {mean_non_penalized}")
```

```
Average price for penalized hospitals: 9393.53
Average price for non-penalized hospitals: 9679.17

/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1354654028.py:5: SettingWitl
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
  hcris_2012['discount_factor'] = 1 - hcris_2012['tot_discounts'] / hcris_2012['tot_charges']
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1354654028.py:6: SettingWitl
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
  hcris_2012['price_num'] = (
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1354654028.py:9: SettingWitl
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
  hcris_2012['price_denom'] = hcris_2012['tot_discharges'] - hcris_2012['mcare_discharges']
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1354654028.py:10: SettingWit
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
  hcris_2012['price'] = hcris_2012['price_num'] / hcris_2012['price_denom']
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1354654028.py:13: SettingWit
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
  hcris_2012['hvbp_payment'] = hcris_2012['hvbp_payment'].fillna(0)
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1354654028.py:14: SettingWit
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
```

```
  hcris_2012['hrrp_payment'] = hcris_2012['hrrp_payment'].fillna(0).abs()
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1354654028.py:17: SettingWi
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
  hcris_2012['penalty'] = (hcris_2012['hvbp_payment'] - hcris_2012['hrrp_payment'] < 0).asty
```

```
#Question 6
hcris_2012['beds_quartile'] = pd.qcut(hcris_2012['beds'], 4, labels=[1, 2, 3, 4])

# Create indicator variables for each quartile
for i in range(1, 5):
    hcris_2012[f'quartile_{i}'] = (hcris_2012['beds_quartile'] == i).astype(int)


# Calculate average price for treated and control groups within each quartile
Avg_per_group = []
for i in range(1, 5):
    treated_mean = hcris_2012.loc[(hcris_2012[f'quartile_{i}'] == 1) & (hcris_2012['penalty']
    control_mean = hcris_2012.loc[(hcris_2012[f'quartile_{i}'] == 1) & (hcris_2012['penalty']
    Avg_per_group.append({'Quartile': i, 'Treated_Mean_Price': round(treated_mean, 2), 'Cont

results_df = pd.DataFrame(Avg_per_group)
print(results_df)
```

```
   Quartile  Treated_Mean_Price  Control_Mean_Price
0         1             4891.28             8288.78
1         2             6347.81             8482.43
2         3             8268.98             8855.91
3         4            11218.04            10992.34

/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1606227692.py:2: SettingWitl
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
```

```
  hcris_2012['beds_quartile'] = pd.qcut(hcris_2012['beds'], 4, labels=[1, 2, 3, 4])
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1606227692.py:6: SettingWit
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
  hcris_2012[f'quartile_{i}'] = (hcris_2012['beds_quartile'] == i).astype(int)
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1606227692.py:6: SettingWit
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
  hcris_2012[f'quartile_{i}'] = (hcris_2012['beds_quartile'] == i).astype(int)
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1606227692.py:6: SettingWit
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
  hcris_2012[f'quartile_{i}'] = (hcris_2012['beds_quartile'] == i).astype(int)
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/1606227692.py:6: SettingWit
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
  hcris_2012[f'quartile_{i}'] = (hcris_2012['beds_quartile'] == i).astype(int)
```

```python
from causalinference import CausalModel
# Step 1: Define treatment (penalized) and control groups
hcris_2012['treated'] = (hcris_2012['penalty'] > 0).astype(int)
treated_df = hcris_2012[hcris_2012['treated'] == 1]
control_df = hcris_2012[hcris_2012['treated'] == 0]

# Step 2: Select bed quartiles for matching
covariate = 'beds_quartile'

# Step 3: Calculate inverse variance weights
variance_quartiles = control_df.groupby(covariate)['beds'].var().fillna(1)
inverse_weights = 1 / variance_quartiles

# Step 4: Perform nearest neighbor matching using inverse variance distance
matched_pairs = []
for _, treated_row in treated_df.iterrows():
```

```
    quartile = treated_row[covariate]
    control_candidates = control_df[control_df[covariate] == quartile]

    if not control_candidates.empty:
        # Compute distance: absolute difference in beds * inverse variance weight
        distances = np.abs(control_candidates['beds'] - treated_row['beds']) * inverse_weight

        # Get the best match (hospital with minimum distance)
        best_match_idx = distances.idxmin()
        best_match = control_candidates.loc[best_match_idx]

        # Store matched pair (treated price, control price)
        matched_pairs.append((treated_row['price'], best_match['price']))

#Calculate ATE
treated_prices, control_prices = zip(*matched_pairs)
ate_nn_inverse_variance = np.mean(np.array(treated_prices) - np.array(control_prices))

X = hcris_2012[[covariate]].values
y = hcris_2012['price'].values
treatment = hcris_2012['treated'].values

causal_model = CausalModel(Y=y, D=treatment, X=X)
causal_model.est_via_matching(matches=1, bias_adj=True)

#Printing
print(f"ATE using Nearest Neighbor Matching (Inverse Variance Distance): {ate_nn_inverse_vari
print(causal_model.estimates)
```

```
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/4238294782.py:3: SettingWitl
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
  hcris_2012['treated'] = (hcris_2012['penalty'] > 0).astype(int)
/var/folders/2q/wzjp_2kd355b8clhzqwmytb40000gn/T/ipykernel_45724/4238294782.py:11: FutureWarn
  variance_quartiles = control_df.groupby(covariate)['beds'].var().fillna(1)

ValueError: zero-size array to reduction operation minimum which has no identity
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[57], line 40
```

```
     37 treatment = hcris_2012['treated'].values
     39 causal_model = CausalModel(Y=y, D=treatment, X=X)
---> 40 causal_model.est_via_matching(matches=1, bias_adj=True)
     42 #Printing
     43 print(f"ATE using Nearest Neighbor Matching (Inverse Variance Distance): ate_nn_inve
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/causal.py:331, in CausalMode
    328 else:
    329  W = weights
--> 331 self.estimates['matching'] = Matching(self.raw_data, W,
    332                                         matches, bias_adj)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:23, i
     20 Y_c, Y_t = data['Y_c'], data['Y_t']
     21 X_c, X_t = data['X_c'], data['X_t']
---> 23 matches_c = [match(X_i, X_t, W, m) for X_i in X_c]
     24 matches_t = [match(X_i, X_c, W, m) for X_i in X_t]
     25 Yhat_c = np.array([Y_t[idx].mean() for idx in matches_c])
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:83, i
     79 def match(X_i, X_m, W, m):
     81  d = norm(X_i, X_m, W)
---> 83  return smallestm(d, m)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:76, i
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
---> 76  return smallestm(d, m+2)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:76, i
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
---> 76  return smallestm(d, m+2)
    [... skipping similar frames: smallestm at line 76 (1001 times)]
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:76, i
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
---> 76  return smallestm(d, m+2)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:73, i
     71 if d[par_idx[:m]].max() < d[par_idx[m]]:  # m < (m+1)th
     72  return par_idx[:m]
---> 73 elif d[par_idx[m]] < d[par_idx[m+1:]].min():  # m+1 < (m+2)th
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
File /opt/anaconda3/lib/python3.12/site-packages/numpy/core/_methods.py:45, in _amin(a, axis
     43 def _amin(a, axis=None, out=None, keepdims=False,
     44           initial=_NoValue, where=True):
---> 45     return umr_minimum(a, axis, None, out, keepdims, initial, where)
```

```
ValueError: zero-size array to reduction operation minimum which has no identity
```

```
from causalinference import CausalModel
#Question 7 - Nearest Neighbor Match - Mahalanobis

# Select relevant variables
X = hcris_2012[['beds_quartile']].values
y = hcris_2012['price'].values
treatment = hcris_2012['penalty'].values

# Create Causal Model
causal_model = CausalModel(Y=y, D=treatment, X=X)

# Perform Nearest Neighbor Matching (1-to-1) with Inverse Variance Distance
causal_model.est_via_matching(matches=1, bias_adj=True)

# Print the Average Treatment Effect (ATE)
print("ATE using Nearest Neighbor Matching (Inverse Variance Distance):")
print(causal_model.estimates)
```

```
ValueError: zero-size array to reduction operation minimum which has no identity
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[31], line 14
     11 causal_model = CausalModel(Y=y, D=treatment, X=X)
     13 # Perform Nearest Neighbor Matching (1-to-1) with Inverse Variance Distance
---> 14 causal_model.est_via_matching(matches=1, bias_adj=True)
     16 # Print the Average Treatment Effect (ATE)
     17 print("ATE using Nearest Neighbor Matching (Inverse Variance Distance):")
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/causal.py:331, in CausalMode
    328 else:
    329  W = weights
--> 331 self.estimates['matching'] = Matching(self.raw_data, W,
    332                                         matches, bias_adj)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:23, i
     20 Y_c, Y_t = data['Y_c'], data['Y_t']
     21 X_c, X_t = data['X_c'], data['X_t']
---> 23 matches_c = [match(X_i, X_t, W, m) for X_i in X_c]
     24 matches_t = [match(X_i, X_c, W, m) for X_i in X_t]
     25 Yhat_c = np.array([Y_t[idx].mean() for idx in matches_c])
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:83, i
     79 def match(X_i, X_m, W, m):
     81  d = norm(X_i, X_m, W)
```

```
---> 83  return smallestm(d, m)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:76, i
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
---> 76  return smallestm(d, m+2)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:76, i
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
---> 76  return smallestm(d, m+2)
    [... skipping similar frames: smallestm at line 76 (1001 times)]
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:76, i
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
---> 76  return smallestm(d, m+2)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:73, i
     71 if d[par_idx[:m]].max() < d[par_idx[m]]:  # m < (m+1)th
     72  return par_idx[:m]
---> 73 elif d[par_idx[m]] < d[par_idx[m+1:]].min():  # m+1 < (m+2)th
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
File /opt/anaconda3/lib/python3.12/site-packages/numpy/core/_methods.py:45, in _amin(a, axis
     43 def _amin(a, axis=None, out=None, keepdims=False,
     44           initial=_NoValue, where=True):
---> 45     return umr_minimum(a, axis, None, out, keepdims, initial, where)
ValueError: zero-size array to reduction operation minimum which has no identity
```

```
# Question 7 - Inverse Propensity Weighting
X = hcris_2012[['beds_quartile']].values
y = hcris_2012['price'].values
treatment = hcris_2012['penalty'].values

pw_model = CausalModel(Y=y, D=treatment, X=X)
pw_model.est_via_matching(matches=1, bias_adj=True)

# Print
print("ATE using Propensity Score Matching:")
print(pw_model.estimates)
```

```
ValueError: zero-size array to reduction operation minimum which has no identity
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[44], line 7
      4 treatment = hcris_2012['penalty'].values
      6 pw_model = CausalModel(Y=y, D=treatment, X=X)
----> 7 pw_model.est_via_matching(matches=1, bias_adj=True)
      9 # Print
     10 print("ATE using Propensity Score Matching:")
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/causal.py:331, in CausalMode
    328 else:
    329  W = weights
--> 331 self.estimates['matching'] = Matching(self.raw_data, W,
    332                                         matches, bias_adj)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:23, i
     20 Y_c, Y_t = data['Y_c'], data['Y_t']
     21 X_c, X_t = data['X_c'], data['X_t']
---> 23 matches_c = [match(X_i, X_t, W, m) for X_i in X_c]
     24 matches_t = [match(X_i, X_c, W, m) for X_i in X_t]
     25 Yhat_c = np.array([Y_t[idx].mean() for idx in matches_c])
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:83, i
     79 def match(X_i, X_m, W, m):
     81  d = norm(X_i, X_m, W)
---> 83  return smallestm(d, m)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:76, i
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
---> 76  return smallestm(d, m+2)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:76, i
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
---> 76  return smallestm(d, m+2)
    [... skipping similar frames: smallestm at line 76 (1001 times)]
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:76, i
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
---> 76  return smallestm(d, m+2)
File /opt/anaconda3/lib/python3.12/site-packages/causalinference/estimators/matching.py:73, i
     71 if d[par_idx[:m]].max() < d[par_idx[m]]:  # m < (m+1)th
     72  return par_idx[:m]
---> 73 elif d[par_idx[m]] < d[par_idx[m+1:]].min():  # m+1 < (m+2)th
     74  return par_idx[:m+1]
     75 else:  # mth = (m+1)th = (m+2)th, so increment and recurse
```

```
File /opt/anaconda3/lib/python3.12/site-packages/numpy/core/_methods.py:45, in _amin(a, axis
     43 def _amin(a, axis=None, out=None, keepdims=False,
     44           initial=_NoValue, where=True):
---> 45     return umr_minimum(a, axis, None, out, keepdims, initial, where)
ValueError: zero-size array to reduction operation minimum which has no identity
```

```
#Question 7 - Simple Liner Regression
```