DF_Creation

October 20, 2025

[ ]:
```
import pandas as pd
pd.set_option('display.max_columns', None)

# Load CSV
df = pd.read_csv('mimic_cohort.csv')

# Save as Parquet
df.to_parquet('mimic_cohort.parquet')
```

[ ]:
```
mimic_cohort = pd.read_parquet('mimic_cohort.parquet')
mimic_cohort.head()
```

[ ]:
```
one_df = pd.read_csv("elixhauser_onehot.csv")

one_df.to_parquet("elixhauser_onehot.parquet")
```

[ ]:
```
onehot_cohort = pd.read_parquet("elixhauser_onehot.parquet")
onehot_cohort.head()
```

[ ]:
```
first_comorbidity_col = 'chf'
comorbidity_cols = onehot_cohort.columns[onehot_cohort.columns.
 ↪get_loc(first_comorbidity_col):].tolist()
X = onehot_cohort[comorbidity_cols].values
X
```

[ ]:
```
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans

# Standardize, since some papers do (binary columns usually don't need, but␣
 ↪follow paper's method if specified)

n_clusters = 6  # Set to number of clusters used in the paper
kmeans = KMeans(n_clusters=n_clusters, random_state=42)
cluster_labels = kmeans.fit_predict(X)
onehot_cohort['cluster'] = cluster_labels
```

[ ]:
```
# Compute the mean presence of each comorbidity feature in each cluster
cluster_summary = onehot_cohort.groupby('cluster')[comorbidity_cols].mean()
print(cluster_summary)
```

[ ]:
```
import networkx as nx
import matplotlib.pyplot as plt

# Example: Build a simple co-occurrence graph for one cluster
G = nx.Graph()
for comorb in comorbidity_cols:
    G.add_node(comorb)
# Add edges based on co-occurrence rates (customize logic to match paper)
# Example: link every pair of comorbidities if they co-occur in >10% of␣
 ↪patients in the cluster
threshold = 0.1
cluster = 0  # Example: focus on first cluster
df_cluster = onehot_cohort[onehot_cohort['cluster'] == cluster]

for i, c1 in enumerate(comorbidity_cols):
    for c2 in comorbidity_cols[i+1:]:
        co_occur_rate = ((df_cluster[c1] == 1) & (df_cluster[c2] == 1)).mean()
        if co_occur_rate > threshold:
            G.add_edge(c1, c2, weight=co_occur_rate)

nx.draw(G, with_labels=True)
plt.show()
```

[ ]:
```

```