STATS 3DA3

Homework Assignment 6

Instructor: Pratheepa Jeganathan

2025-03-21

Submission Deadline

- All submissions must be made before 10:00 PM on Wednesday, April 16, 2025.

Late Submissions

- Late Submission Penalty: A 15% deduction per day will be applied to assignments submitted after the deadline, including SAS accomdations.
- Late Submission Limit: Assignments submitted **more than 72 hours late** will receive a **grade of zero**, including SAS accomdations.

Submission Guidelines

- Format: Submit your work as a single PDF file via Avenue to Learn. You may submit individually or or as a group of up to three members.

Individual Submission

- Complete Questions 1–15.
- A GitHub repository is optional.

Group Submission

- Complete Questions 1–17.
- Group Size: Up to three members.
- Team Members' Contributions: For group submissions, you must complete Question 16, detailing each member's contributions. This should correspond with the commit history in the GitHub repository.
  - Note: While Question 16 is not graded, failure to include this information will result in the assignment not being graded.
  - Example:
    * Member A: Questions 1, 2, 4
    * Member B: Questions 3, 5, 6
- GitHub Repository: You must include a link to a public GitHub repository showing the assignment's version history.
  - Note: While Question 17 is not graded, failure to provide this information will result in the assignment not being graded.

Assignment Standards

Please ensure your assignment adheres to the following standards for submission:

- Title Page Requirements: Each submission must include a title page featuring your group members' names and student IDs. Assignments without a title page will not be considered for grading.
- Formatting Preferences: The use of Quarto Jupyter Notebook for document preparation is highly recommended.
- Font and Spacing: Submissions must utilize an eleven-point font (Times New Roman or a similar font) with 1.5 line spacing. Ensure margins of at least 1 inch on all sides.
- Individual Work: While discussing homework problems with peers and other groups is permitted, the final written submission must be your group work.

- Submission Content: Do not include the assignment questions within your PDF. Instead, clearly mark each response with the corresponding question number. Screenshots are not an acceptable form of submission under any circumstances.
- Academic Writing: Ensure that your writing and any references used are appropriate for an undergraduate level of study.
- Originality Checks: Be aware that the instructor may use various tools, including online resources or in-person meetings, to verify the originality of submitted work.

Assignment Policy on the Use of Generative AI

- The use of Generative AI is not permitted in assignments, except for using GitHub Copilot as a coding assistant.
  - If GitHub Copilot is used, you must clearly indicate this in the code comments.
- In alignment with McMaster academic integrity policy, it "shall be an offence knowingly to submit academic work for assessment that was purchased or acquired from another source". This includes work created by generative AI tools. Also state in the policy is the following, "Contract Cheating is the act of"outsourcing of student work to third parties" with or without payment." Using Generative AI tools is a form of contract cheating. Charges of academic dishonesty will be brought forward to the Office of Academic Integrity.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, f1_score, classification_report
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.feature_selection import SelectKBest, chi2
```

```
dataset_url = "https://raw.githubusercontent.com/PratheepaJ/datasets/refs/heads/master/ass6-dat
df = pd.read_csv(dataset_url)
print("Initial dataset head:")
print(df.head())
```

```
Initial dataset head:
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   63    1   1       145   233    1        2      150      0      2.3      3
1   67    1   4       160   286    0        2      108      1      1.5      2
2   67    1   4       120   229    0        2      129      1      2.6      2
3   37    1   3       130   250    0        0      187      0      3.5      3
4   41    0   2       130   204    0        2      172      0      1.4      1

    ca  thal  num
0  0.0   6.0    0
1  3.0   3.0    2
2  2.0   7.0    1
3  0.0   3.0    0
4  0.0   3.0    0
```

1 The goal is to predict whether a patient has heart disease (binary outcome) using various clinical and demographic features

```
from sklearn.preprocessing import StandardScaler
# checking missing values
print("\nMissing Values in Each Column:")
print(df.isnull().sum())

#   0 remains 0 (no heart disease) any value > 0 becomes 1 (heart disease present)
df['num'] = df['num'].apply(lambda x: 0 if x == 0 else 1)
print("\nValue Counts for Transformed Target 'num':")
print(df['num'].value_counts())

# Feature Scaling
# Identify numeric feature columns (excluding the target 'num')
numeric_features = df.select_dtypes(include=['float64', 'int64']).columns.tolist()
if 'num' in numeric_features:
    numeric_features.remove('num')

print("\nNumeric Feature Columns:")
print(numeric_features)

# Apply Standard Scaling to numeric features
scaler = StandardScaler()
df[numeric_features] = scaler.fit_transform(df[numeric_features])

print("\nDataset after Scaling (first 5 rows):")
print(df.head())
```

```
Missing Values in Each Column:
age         0
```

```
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          4
thal        2
num         0
dtype: int64

Value Counts for Transformed Target 'num':
num
0    164
1    139
Name: count, dtype: int64

Numeric Feature Columns:
['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slo

Dataset after Scaling (first 5 rows):
        age       sex        cp  trestbps      chol       fbs   restecg  \
0  0.948726  0.686202 -2.251775  0.757525 -0.264900  2.394438  1.016684
1  1.392002  0.686202  0.877985  1.611220  0.760415 -0.417635  1.016684
2  1.392002  0.686202  0.877985 -0.665300 -0.342283 -0.417635  1.016684
3 -1.932564  0.686202 -0.165268 -0.096170  0.063974 -0.417635 -0.996749
4 -1.489288 -1.457296 -1.208521 -0.096170 -0.825922 -0.417635  1.016684
```

```
    thalach     exang   oldpeak     slope        ca      thal  num
0  0.017197 -0.696631  1.087338  2.274579 -0.718306  0.653650    0
1 -1.821905  1.435481  0.397182  0.649113  2.487269 -0.895552    1
2 -0.902354  1.435481  1.346147  0.649113  1.418744  1.170051    1
3  1.637359 -0.696631  2.122573  2.274579 -0.718306 -0.895552    0
4  0.980537 -0.696631  0.310912 -0.976352 -0.718306 -0.895552    0
```

```
# 3
print("\n--- Dataset Information ---")
print(df.info())
print("\n--- Statistical Summary ---")
print(df.describe())
```

```
--- Dataset Information ---
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    float64
 1   sex       303 non-null    float64
 2   cp        303 non-null    float64
 3   trestbps  303 non-null    float64
 4   chol      303 non-null    float64
 5   fbs       303 non-null    float64
 6   restecg   303 non-null    float64
 7   thalach   303 non-null    float64
 8   exang     303 non-null    float64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    float64
 11  ca        299 non-null    float64
```

```
 12  thal      301 non-null    float64
 13  num       303 non-null    int64
dtypes: float64(13), int64(1)
memory usage: 33.3 KB
None

--- Statistical Summary ---
                age           sex            cp      trestbps          chol  \
count  3.030000e+02  3.030000e+02  3.030000e+02  3.030000e+02  3.030000e+02
mean  -1.465641e-18 -2.931282e-17 -1.670831e-16  4.426236e-16  2.345026e-16
std    1.001654e+00  1.001654e+00  1.001654e+00  1.001654e+00  1.001654e+00
min   -2.819115e+00 -1.457296e+00 -2.251775e+00 -2.145037e+00 -2.334877e+00
25%   -7.135564e-01 -1.457296e+00 -1.652679e-01 -6.652997e-01 -6.905030e-01
50%    1.729945e-01  6.862024e-01 -1.652679e-01 -9.616980e-02 -1.101357e-01
75%    7.270888e-01  6.862024e-01  8.779855e-01  4.729601e-01  5.476139e-01
max    2.500191e+00  6.862024e-01  8.779855e-01  3.887739e+00  6.138485e+00

                fbs       restecg       thalach         exang       oldpeak  \
count  3.030000e+02  3.030000e+02  3.030000e+02  3.030000e+02  3.030000e+02
mean  -1.172513e-17 -1.172513e-17 -1.172513e-16 -9.086974e-17  2.345026e-17
std    1.001654e+00  1.001654e+00  1.001654e+00  1.001654e+00  1.001654e+00
min   -4.176345e-01 -9.967493e-01 -3.442067e+00 -6.966305e-01 -8.968617e-01
25%   -4.176345e-01 -9.967493e-01 -7.053073e-01 -6.966305e-01 -8.968617e-01
50%   -4.176345e-01  9.967493e-03  1.485618e-01 -6.966305e-01 -2.067053e-01
75%   -4.176345e-01  1.016684e+00  7.178079e-01  1.435481e+00  4.834512e-01
max    2.394438e+00  1.016684e+00  2.294182e+00  1.435481e+00  4.451851e+00

              slope            ca          thal         num
count  3.030000e+02  2.990000e+02  3.010000e+02  303.000000
mean   1.436328e-16 -2.970496e-17 -8.262125e-17    0.458746
std    1.001654e+00  1.001676e+00  1.001665e+00    0.499120
min   -9.763521e-01 -7.183062e-01 -8.955519e-01    0.000000
```

```
25%   -9.763521e-01 -7.183062e-01 -8.955519e-01    0.000000
50%    6.491132e-01 -7.183062e-01 -8.955519e-01    0.000000
75%    6.491132e-01  3.502190e-01  1.170051e+00    1.000000
max    2.274579e+00  2.487269e+00  1.170051e+00    1.000000
```

Three descriptive statements:

The dataset consists of clinical and demographic variables (e.g., age, sex, chest pain type, blood pressure, cholesterol) that are typical in a heart disease dataset. The continuous variables (such as age, trestbps, chol, and thalach) show a broad range and their distributions may be normal or slightly skewed. The dataset initially contains a number of observations (df.shape[0]) and includes some missing values.

```python
# 4
df['num'] = df['num'].apply(lambda x: 0 if x == 0 else 1)
print("\nValue counts for transformed binary target 'num':")
print(df['num'].value_counts())
```

```
Value counts for transformed binary target 'num':
num
0    164
1    139
Name: count, dtype: int64
```

```python
# 5(need further explaining)
corr_matrix = df.corr()
print("\nCorrelation matrix:")
print(corr_matrix)
```

```
Correlation matrix:
```

	age	sex	cp	trestbps	chol	fbs	\
age	1.000000	-0.097542	0.104139	0.284946	0.208950	0.118530	
sex	-0.097542	1.000000	0.010084	-0.064456	-0.199915	0.047862	
cp	0.104139	0.010084	1.000000	-0.036077	0.072319	-0.039975	
trestbps	0.284946	-0.064456	-0.036077	1.000000	0.130120	0.175340	
chol	0.208950	-0.199915	0.072319	0.130120	1.000000	0.009841	
fbs	0.118530	0.047862	-0.039975	0.175340	0.009841	1.000000	
restecg	0.148868	0.021647	0.067505	0.146560	0.171043	0.069564	
thalach	-0.393806	-0.048663	-0.334422	-0.045351	-0.003432	-0.007854	
exang	0.091661	0.146201	0.384060	0.064762	0.061310	0.025665	
oldpeak	0.203805	0.102173	0.202277	0.189171	0.046564	0.005747	
slope	0.161770	0.037533	0.152050	0.117382	-0.004062	0.059894	
ca	0.362605	0.093185	0.233214	0.098773	0.119000	0.145478	
thal	0.127389	0.380936	0.265246	0.133554	0.014214	0.071358	
num	0.223120	0.276816	0.414446	0.150825	0.085164	0.025264	

	restecg	thalach	exang	oldpeak	slope	ca	\
age	0.148868	-0.393806	0.091661	0.203805	0.161770	0.362605	
sex	0.021647	-0.048663	0.146201	0.102173	0.037533	0.093185	
cp	0.067505	-0.334422	0.384060	0.202277	0.152050	0.233214	
trestbps	0.146560	-0.045351	0.064762	0.189171	0.117382	0.098773	
chol	0.171043	-0.003432	0.061310	0.046564	-0.004062	0.119000	
fbs	0.069564	-0.007854	0.025665	0.005747	0.059894	0.145478	
restecg	1.000000	-0.083389	0.084867	0.114133	0.133946	0.128343	
thalach	-0.083389	1.000000	-0.378103	-0.343085	-0.385601	-0.264246	
exang	0.084867	-0.378103	1.000000	0.288223	0.257748	0.145570	
oldpeak	0.114133	-0.343085	0.288223	1.000000	0.577537	0.295832	
slope	0.133946	-0.385601	0.257748	0.577537	1.000000	0.110119	
ca	0.128343	-0.264246	0.145570	0.295832	0.110119	1.000000	
thal	0.024531	-0.279631	0.329680	0.341004	0.287232	0.256382	
num	0.169202	-0.417167	0.431894	0.424510	0.339213	0.460442	

```
              thal       num
age       0.127389  0.223120
sex       0.380936  0.276816
cp        0.265246  0.414446
trestbps  0.133554  0.150825
chol      0.014214  0.085164
fbs       0.071358  0.025264
restecg   0.024531  0.169202
thalach  -0.279631 -0.417167
exang     0.329680  0.431894
oldpeak   0.341004  0.424510
slope     0.287232  0.339213
ca        0.256382  0.460442
thal      1.000000  0.525689
num       0.525689  1.000000
```

```python
# 6
df_clean = df.dropna()
print("\nNumber of observations after dropping missing values:", df_clean.shape[0])
```

```
Number of observations after dropping missing values: 297
```

```python
numeric_cols = df_clean.select_dtypes(include=[np.number]).columns.tolist()
if 'num' in numeric_cols:
    numeric_cols.remove('num')
print("\nContinuous variables used for subgroup analysis:", numeric_cols)
```

```
Continuous variables used for subgroup analysis: ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs
```

```
pca = PCA(n_components=2, random_state=1)
pca_results = pca.fit_transform(df_clean[numeric_cols])
df_clean['pca1'] = pca_results[:, 0]
df_clean['pca2'] = pca_results[:, 1]
```

```
/var/folders/bz/_xbkcp397bb676k7lcghv7b40000gn/T/ipykernel_16862/2647241709.py:3: SettingWithC
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/
  df_clean['pca1'] = pca_results[:, 0]
/var/folders/bz/_xbkcp397bb676k7lcghv7b40000gn/T/ipykernel_16862/2647241709.py:4: SettingWithC
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/
  df_clean['pca2'] = pca_results[:, 1]
```

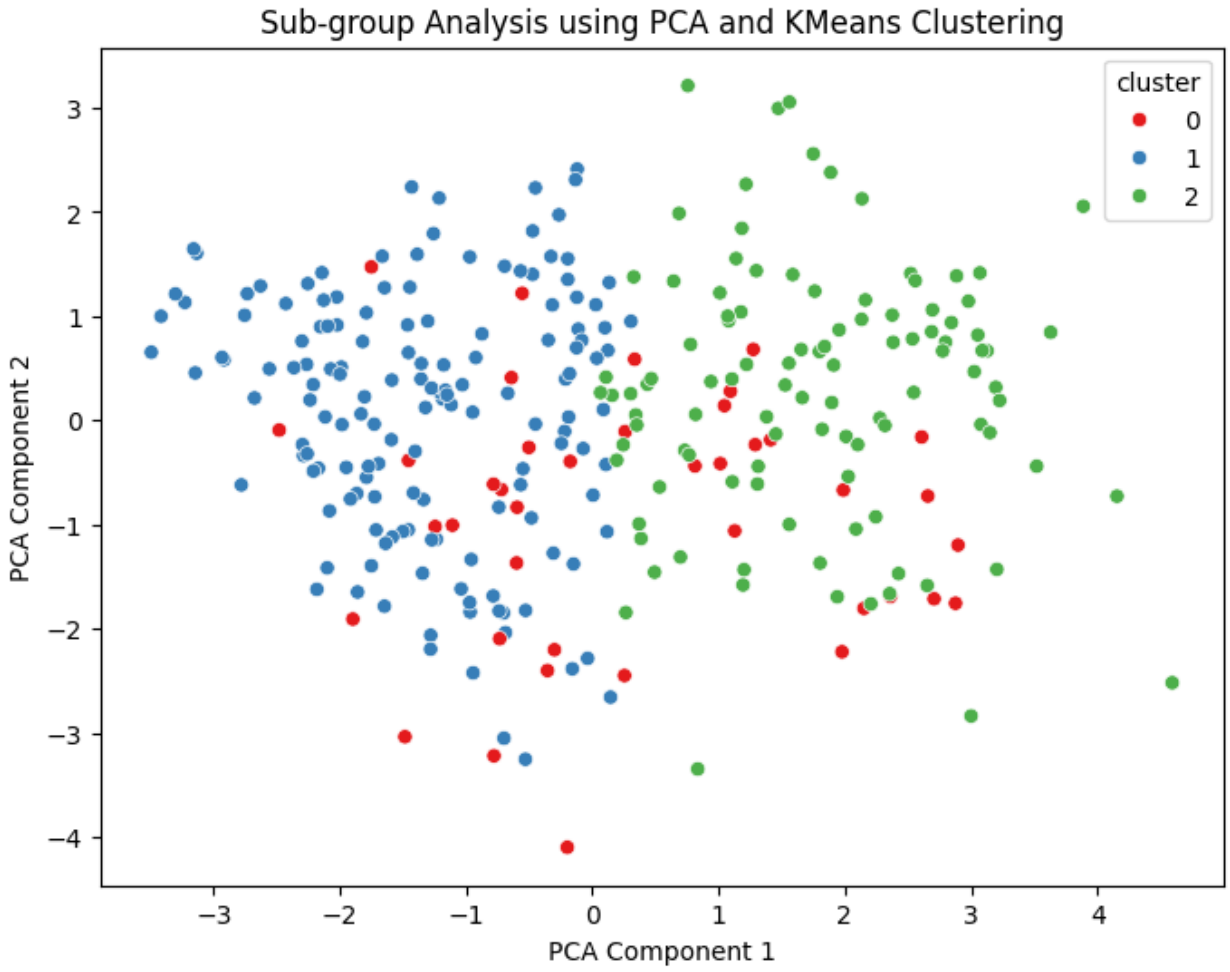
```
kmeans = KMeans(n_clusters=3, random_state=1)
df_clean['cluster'] = kmeans.fit_predict(df_clean[numeric_cols])
```

```
/var/folders/bz/_xbkcp397bb676k7lcghv7b40000gn/T/ipykernel_16862/4096796358.py:2: SettingWithC
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/
  df_clean['cluster'] = kmeans.fit_predict(df_clean[numeric_cols])
```

```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='pca1', y='pca2', hue='cluster', data=df_clean, palette='Set1')
plt.title('Sub-group Analysis using PCA and KMeans Clustering')
plt.xlabel('PCA Component 1')
```

```
plt.ylabel('PCA Component 2')
plt.show()
```




```
# 8
# Prepare the features and target variables while excluding the additional subgroup columns.
features = df_clean.drop(columns=['num', 'cluster', 'pca1', 'pca2'])
target = df_clean['num']

# Split the data using 30% for testing and 70% for training (random seed = 1).
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.3, random_sta
print("\nTraining set size:", X_train.shape[0])
print("Test set size:", X_test.shape[0])
```

```
Training set size: 207
Test set size: 90
```

9

Clusters-logistic, random forest, GridSearchCV

Grading scheme

1.	Answer [1]
2.	Codes (variable type transformation, etc.) [1] OR rationale for no transformation [1]
3.	Codes [3] and three statements [2]
4.	Codes for transforming the response variable [1]
5.	Codes for association [2] and interpretation of figures or tables [2]
6.	Codes [1] answer [1]
7.	Codes to identify sub groups [3] and Plot the sub groups [1]
8.	Codes [1]
9.	classifiers and justification [2]
10.	Describe the two metrics [2]
11.	Codes for training two classifiers [2] Codes for tuning parameters (if any) [1]
12.	Codes for feature selection or feature extraction [1] Codes for training the third classifier with the selected or extracted features [1] Codes for tuning parameters (if any) [1]
13.	Codes for evaluating three classifiers on the test set using two metrics in (10) [3] Two statements for the findings [2] One statement for the impact of feature selection or extraction [1]
14.	Codes finding the important variables [1] Two statements for the analysis and interpretation of the most important predictor variables [2]

15.	Codes for the sub-group improvement strategy (training and tuning parameters, if any) [Bonus 2] Comparison of the performance with the results from (13) [Bonus 1] **Bonus 3 points will be added to the final grade**
16.	Document each team member's specific contributions
17.	Link to the public GitHub repository

The maximum point for this assignment is 39. We will convert this to 100%. The bonus 3 points will be added to the final grade.

All group members will receive the same grade if they contribute to the same.