به نام او که جان را فکرت آموخت

**تمرین سوم درس وب معنایی**

نفیسه عامری1

1 دانشجو، دانشکده مهندسی، دانشگاه فردوسی، مشهد

**صورت سوال**

هر یک از سوالات زیر را بر روی مجموعه داده Adults اجرا نمایید و تغییرات هر مرحله را با استفاده از دیتافریم نمایش دهید.

1. ویژگی‌های عددی این مجموعه داده را استخراج نمایید.
2. در کل مجموعه داده مقادیر خالی ویژگی‌های اسمی را با nan جایگزین نمایید.
3. در مجموعه داده فوق مقادیر عددی فاقد مقدار را با مقدار میانگین آن ویژگی جایگزین کنید.
4. با استفاده از یک روش نرمالسازی، مقادیر عددی را نرمال نمایید.
5. مقادیر عددی ویژگی سن را به 11 bins با عمق یکسان تقسیم کنید.
6. پیوستگی و correlation بین ویژگی‌ها را بدست آورید و ویژگی‌های با کواریانس بالا را حذف نمایید و بگویید در انتها چند ویژگی باقی خواهد ماند. این کار را یکبار برای ویژگی‌های عددی و یکبار برای کل ویژگی‌ها انجام دهید و در مورد نحوه تصمیم گیری خود برای حذف ویژگی‌ها توضیحات لازم را بنویسید.

# شرح تکنیکال

با استفاده از کتابخانه Pandas در پایتون، می‌توان ویژگی‌های عددی یک دیتاست را استخراج کرد. برای این کار ابتدا باید دیتاست را با استفاده از تابع read\_csv یا هر تابع مشابهی به پایتون بخوانیم. سپس با استفاده از تابع describe اطلاعات آماری ویژگی‌های عددی را مشاهده کنیم. و با columns عنوان ستون‌های عددی را به دست بیاوریم. در اینجا ویژگی‌های عددی دیتاست به صورت لیستی از نام‌های ستون‌ها (که عددی هستند) در متغیر numeric\_features قرار دادیم.

با توجه به پراکندگی داده‌های هر ویژگی، ما باید داده‌ها را بیشتر تجزیه و تحلیل کنیم تا محدوده، میانگین، میانه، حالت، انحراف استاندارد و سایر معیارهای آماری برای هر ویژگی تعیین شود. بدون این اطلاعات، تعیین پراکندگی داده‌ها دشوار است.

برای جایگزینی مقادیر خالی با NaN در ویژگی‌های اسمی می‌توان از دستور fillna استفاده کرد. در این دستور، fillna به ما اجازه می‌دهد تا مقادیر خالی را با NaN جایگزین کنیم. مقدار value=np.nan بیان می‌کند که می‌خواهیم مقادیر خالی را با NaN جایگزین کنیم. همچنین inplace=True به ما اجازه می‌دهد تا تغییرات را در محل انجام دهیم و دیتافریم اصلی را به‌روز رسانی کنیم.

برای جایگزینی مقادیر عددی فاقد مقدار با میانگین آن ویژگی، می‌توان از تابع fillna در pandas استفاده کرد. ابتدا میانگین هر ویژگی را با تابع mean محاسبه می‌کنیم، سپس با استفاده از تابع fillna مقادیر خالی را با میانگین آن ویژگی جایگزین می‌کنیم.

در این کد ابتدا داده‌ها از فایل خوانده شده و سپس با استفاده از تابع replace تمام مقادیر خالی ویژگی‌های اسمی با NaN جایگزین می‌شود. در ادامه برای هر ویژگی اگر نوع داده آن float64 یا int64 باشد، میانگین آن ویژگی با استفاده از تابع mean محاسبه شده و با استفاده از تابع fillna مقادیر فاقد مقدار با میانگین آن ویژگی جایگزین می‌شوند.

برای نرمال‌سازی داده‌های عددی می‌توان از روش‌های مختلفی استفاده کرد. یکی از روش‌های معمول استفاده از نرمال‌سازی min-max است. با استفاده از کتابخانه pandas می‌توانیم به سادگی داده‌های عددی را نرمال‌سازی کنیم. برای این کار، ابتدا به داده‌های عددی دسترسی پیدا کرده و سپس با استفاده از تابع min و max، حداقل و حداکثر مقدار هر ستون را بدست آوریم. سپس، داده‌های نرمال‌شده را محاسبه کرده و جایگزین داده‌های اولیه کنیم. در این کد، ابتدا با استفاده از تابع MinMaxScaler که در کتابخانه sklearn قرار دارد، یک نمونه از کلاس MinMaxScaler ایجاد می‌شود. سپس با استفاده از تابع fit\_transform، داده‌های عددی نرمال می‌شوند و در متغیر normalized\_data ذخیره می‌کنیم.

برای تقسیم مقادیر عددی ویژگی سن به 11 بازه با عمق یکسان، می توان از دستور cut از کتابخانه pandas استفاده کرد. ابتدا با استفاده از دستور describe، ویژگی سن را بررسی می‌کنیم. در اینجا، ویژگی جدیدی با نام age\_group به داده اضافه شده است که مقادیر آن برای هر نمونه، شماره بازه‌ای است که مقدار ویژگی سن آن در آن بازه قرار دارد.

برای پیدا کردن پیوستگی بین ویژگی‌ها، ابتدا یک ماتریس کواریانس برای ویژگی‌های عددی ایجاد می‌کنیم و سپس ماتریس همبستگی را محاسبه می‌کنیم. با توجه به مقادیر کواریانس و همبستگی، می‌توانیم ویژگی‌های با کواریانس بالا را حذف کنیم. برای این کار ابتدا داده‌های عددی را از دیتافریم جدا می‌کنیم. سپس ماتریس کواریانس را با استفاده از تابع cov محاسبه می‌کنیم. سپس ماتریس همبستگی را با استفاده از تابع corr محاسبه می‌کنیم. حال برای حذف ویژگی‌های با کواریانس بالا، می‌توانیم به دو روش مختلف عمل کنیم:

1. حذف ویژگی‌هایی که مقدار کواریانس آن‌ها بیشتر از یک حد آستانه‌ای است.
2. حذف ویژگی‌هایی که همبستگی آن‌ها بیشتر از یک حد آستانه‌ای است.

برای این کار می‌توانیم از تابع heatmap کتابخانه Seaborn استفاده کنیم تا نمودار heatmap از ماتریس همبستگی را رسم کنیم. با توجه به مقدار همبستگی، می‌توانیم ویژگی‌هایی را که با همبستگی بالایی دارند، حذف کنیم. به طور مشابه، برای حذف ویژگی‌هایی که مقدار کواریانس آن‌ها بیشتر از یک حد آستانه‌ای است، می‌توانیم نمودار heatmap از ماتریس کواریانس را رسم کنیم و ویژگی‌هایی را که با کواریانس بالایی دارند حذف کنیم. برای حذف ویژگی‌های با کواریانس بالا، می‌توانیم مقدار آستانه‌ای برای کواریانس تعیین کنیم و تمامی ویژگی‌هایی که کواریانس آن با هر ویژگی دیگری بالاتر از این آستانه باشد را حذف کنیم. این کار را می‌توان با استفاده از یک حلقه for انجام داد. در این کد، مقدار آستانه‌ای که برای کواریانس تعیین کردیم، 0.8 بوده است. همچنین در صورت وجود هر ویژگی‌ای که کواریانس آن با ویژگی دیگری بالاتر از این آستانه باشد، آن ویژگی حذف می‌شود. بعد از حذف ویژگی‌های با کواریانس بالا، تعداد ویژگی‌های باقی‌مانده برای مجموعه داده را می‌توان با استفاده از دستور len(data.columns) بدست آورد.

# شرح نتایج

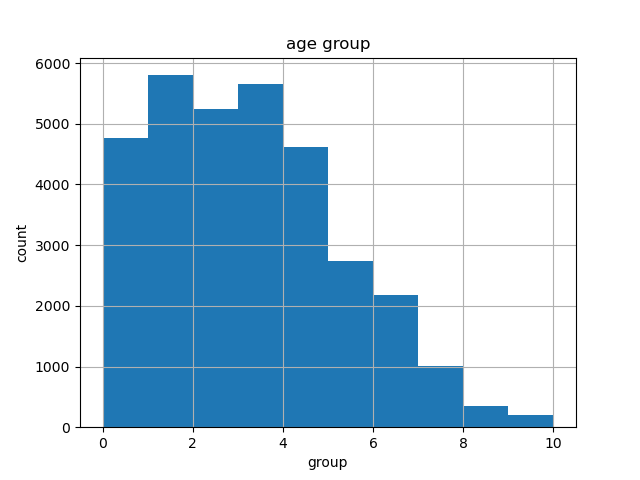
خروجی نتایج مسئله به شرح زیر است. همچنین به ترتیب سوال، فایل‌های numeric features.csv و adult\_data nan.csv و adult\_data mean.csv و adult\_data normalized.csv که در پوشه results قرار دارد نتایج دیتافریم هر مرحله ذخیره شده است.

A screenshot of a computer program

Description automatically generated with medium confidence

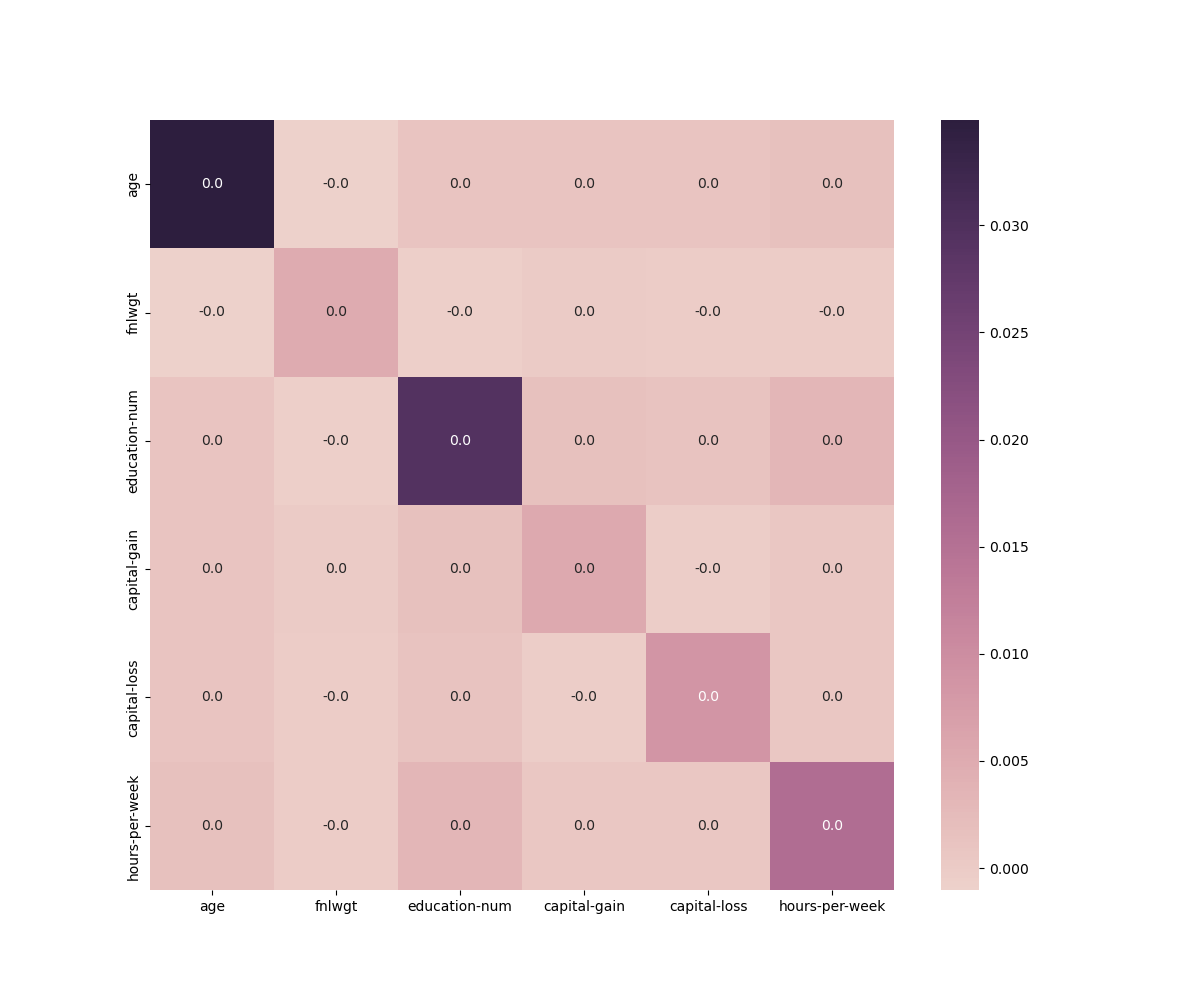
شکل ‏2‑1- خروجی قسمت اول برنامه

همانطور که در تصویر بالا مشاهده می‌کنید، ویژگی‌های عددی این مجموعه داده را استخراج کردیم.

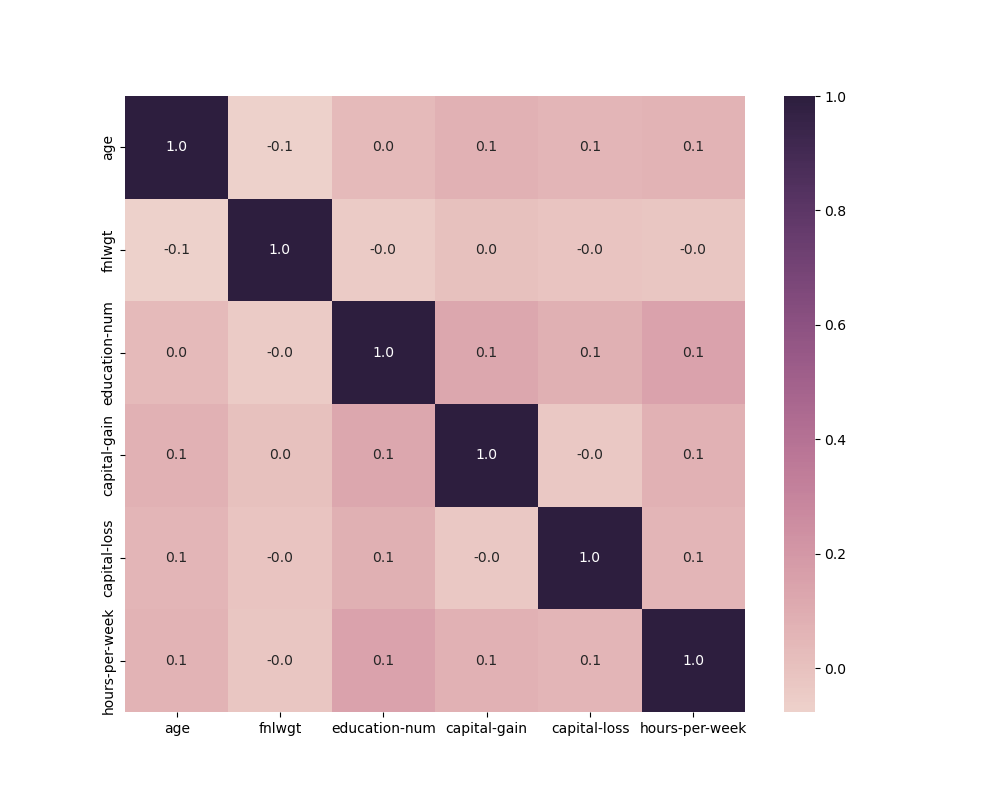


شکل ‏2‑2: نمودار هیستوگرام مقادیر عددی ویژگی سن را به 11 bins با عمق یکسان تقسیم کردیم.

همانطور که در تصویر بالا مشاهده می‌کنید، بیشترین گروه سنی مربوط به گروه یک است که در بازه (20-30) سال است.



شکل ‏2‑3: نمودار heatmap کواریانس مجموعه داده



شکل ‏2‑4: نمودار heatmap correlation مجموعه داده

**پیوست**

**کد برنامه**

import pandas as pd

import numpy as np

import seaborn as sns

from matplotlib import pyplot as plt

from sklearn.preprocessing import MinMaxScaler

columns = ['age', 'workclass', 'fnlwgt', 'education', 'education-num', 'marital-status',

           'occupation', 'relationship', 'race', 'sex', 'capital-gain', 'capital-loss',

           'hours-per-week', 'native-country', 'income']

adult\_data = pd.read\_csv('data/adult.data.csv', names=columns)

# ========================== 1

numeric\_features = adult\_data.describe().columns.tolist()

print('numeric features:', numeric\_features)

df = adult\_data[numeric\_features]

df.to\_csv('numeric features.csv')

# print(df.to\_markdown())

# ========================== 2

adult\_data.fillna(value=np.nan, inplace=True)

# print(adult\_data.to\_markdown())

adult\_data.to\_csv('adult\_data nan.csv')

# ========================== 3

numeric\_data = adult\_data[numeric\_features]

for col in numeric\_data.columns:

    if numeric\_data[col].dtype == 'float64' or numeric\_data[col].dtype == 'int64':

        mean\_value = numeric\_data[col].mean()

        numeric\_data[col].fillna(value=mean\_value, inplace=True)

adult\_data[numeric\_features] = numeric\_data

adult\_data.to\_csv('adult\_data mean.csv')

# ========================== 4

min\_vals = numeric\_data.min()

max\_vals = numeric\_data.max()

scaler = MinMaxScaler()

normalized\_data = scaler.fit\_transform(numeric\_data)

adult\_data[numeric\_features] = normalized\_data

adult\_data.to\_csv('adult\_data normalized.csv')

# ========================== 5

print(adult\_data['age'].describe())

age\_group = pd.cut(adult\_data['age'], bins=11, labels=False)

age\_group.hist()

plt.title('age group')

plt.xlabel('group')

plt.ylabel('count')

plt.savefig('age\_group.png')

# ========================== 6

cov\_matrix = adult\_data.cov()

corr\_matrix = adult\_data.corr()

plt.figure(figsize=(12, 10))

sns.heatmap(cov\_matrix, cmap=sns.cubehelix\_palette(as\_cmap=True), annot=True, fmt=".1f")

plt.savefig('cov\_matrix.png')

plt.figure(figsize=(10, 8))

sns.heatmap(corr\_matrix, cmap=sns.cubehelix\_palette(as\_cmap=True),annot=True, fmt=".1f")

plt.savefig('corr\_matrix.png')

print(len(adult\_data.columns))

threshold = 0.8

corr\_features = set()

for i in range(len(corr\_matrix.columns)):

    for j in range(i):

        if abs(corr\_matrix.iloc[i, j]) > threshold:

            colname = corr\_matrix.columns[i]

            corr\_features.add(colname)

            if colname in adult\_data.columns:

                print(colname)

                del adult\_data[colname]

print(len(adult\_data.columns))

corr\_matrix = numeric\_data.corr()

print(len(numeric\_data.columns))

threshold = 0.8

corr\_features = set()

for i in range(len(corr\_matrix.columns)):

    for j in range(i):

        if abs(corr\_matrix.iloc[i, j]) > threshold:

            colname = corr\_matrix.columns[i]

            corr\_features.add(colname)

            if colname in numeric\_data.columns:

                print(colname)

                del numeric\_data[colname]

print(len(numeric\_data.columns))

plt.figure(figsize=(10, 8))

sns.heatmap(corr\_matrix, cmap=sns.cubehelix\_palette(as\_cmap=True), annot=True, fmt=".1f")

plt.savefig('corr\_matrix\_num.png')