

به نام او که جان را فکرت آموخت

تمرین سوم درس وب معنایی

نقیسه عامری^۱

^۱ دانشجوی، 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.

- (۱) ویژگی‌های عددی این مجموعه داده را استخراج نمایید.
- (۲) در کل مجموعه داده مقادیر خالی ویژگی‌های اسمی را با nan جایگزین نمایید.
- (۳) در مجموعه داده فوق مقادیر عددی فاقد مقدار را با مقدار میانگین آن ویژگی جایگزین کنید.
- (۴) با استفاده از یک روش نرمالسازی، مقادیر عددی را نرمال نمایید.
- (۵) مقادیر عددی ویژگی سن را به ۱۱ bins با عمق یکسان تقسیم کنید.
- (۶) 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.

فصل ۱ – 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 ۱۱ بازه با عمق یکسان، می‌توان 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:

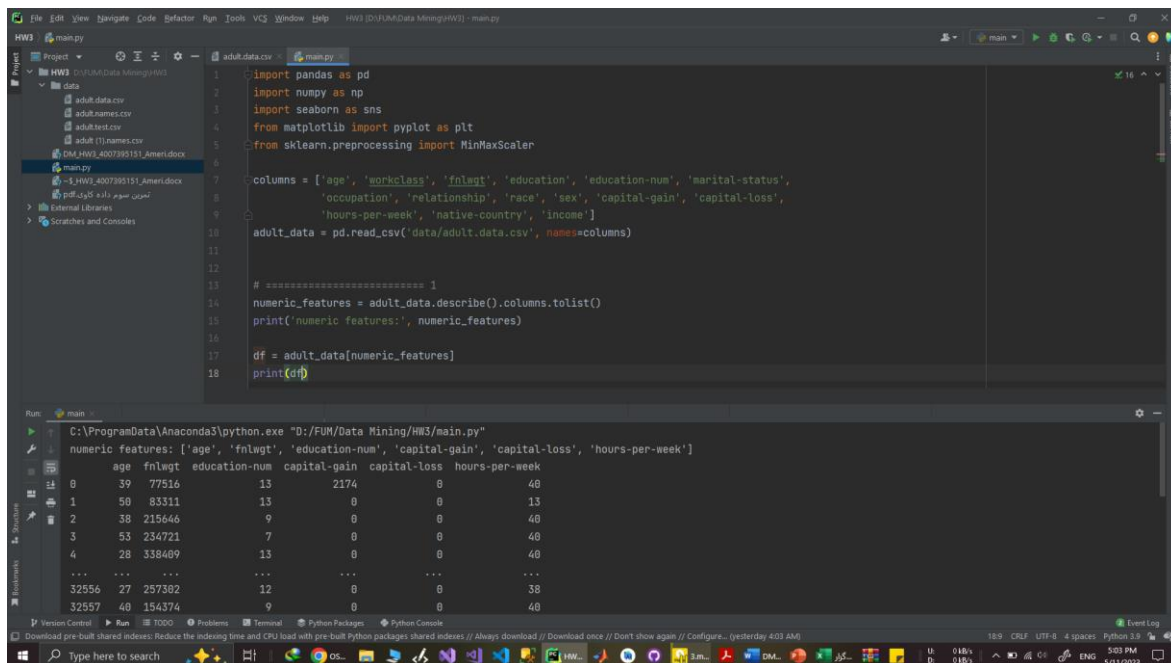
(۱) حذف ویژگی‌هایی که مقدار کواریانس آن‌ها بیشتر از یک حد آستانه‌ای است.

(۲) 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، ۰.۸ 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.

فصل ۲- 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.



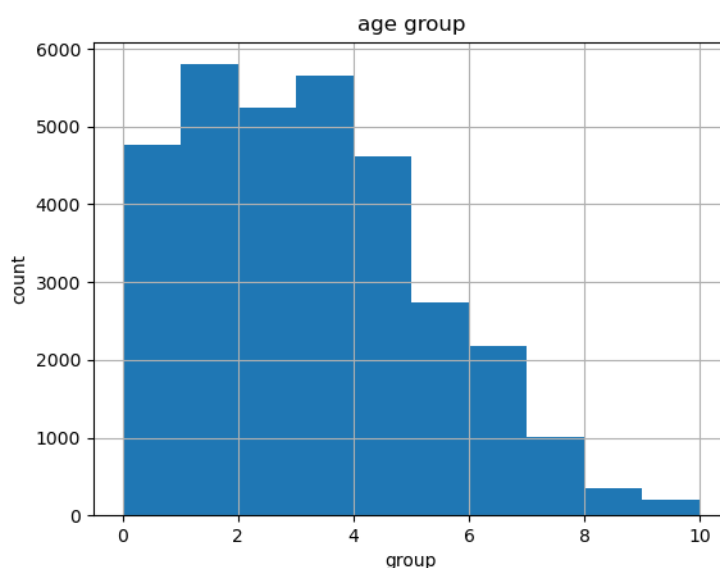
```
1 import pandas as pd
2 import numpy as np
3 import seaborn as sns
4 from matplotlib import pyplot as plt
5 from sklearn.preprocessing import MinMaxScaler
6
7 columns = ['age', 'workclass', 'fnlwgt', 'education', 'education-num', 'marital-status',
8           'occupation', 'relationship', 'race', 'sex', 'capital-gain', 'capital-loss',
9           'hours-per-week', 'native-country', 'income']
10
11 adult_data = pd.read_csv('data/adult.data.csv', names=columns)
12
13 # ===== 1
14 numeric_features = adult_data.describe().columns.tolist()
15 print('numeric features:', numeric_features)
16
17 df = adult_data[numeric_features]
18 print(df)
```

numeric features: ['age', 'fnlwgt', 'education-num', 'capital-gain', 'capital-loss', 'hours-per-week']

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week
0	39	77516	13	2174	0	40
1	50	83311	13	0	0	13
2	38	215646	9	0	0	40
3	53	234721	7	0	0	40
4	28	338409	13	0	0	40
...
32556	27	257302	12	0	0	38
32557	40	154374	9	0	0	40

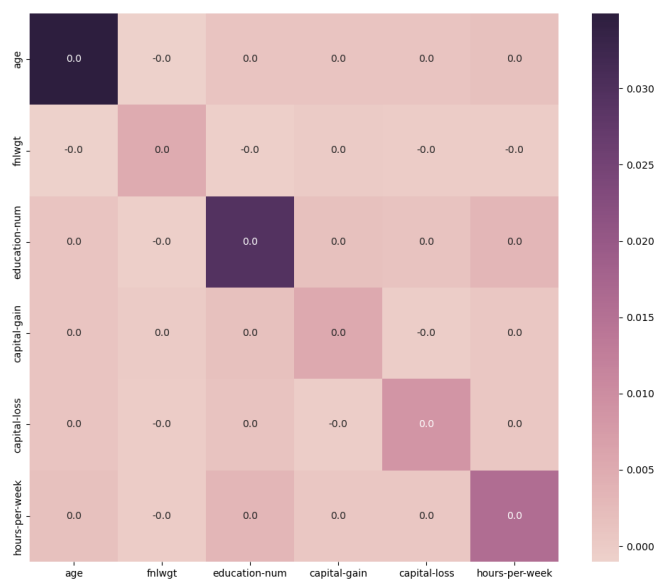
شکل ۲-۱- خروجی قسمت اول برنامه

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

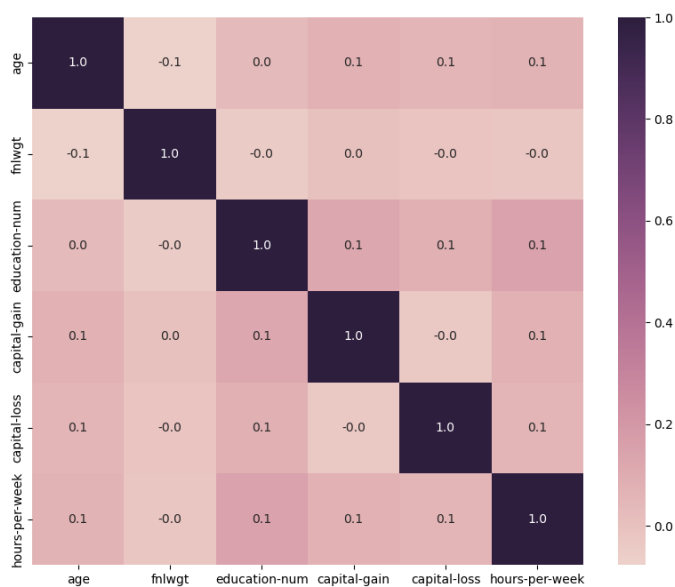


شکل ۲-۲: نمودار هیستوگرام مقادیر عددی ویژگی سن را به ۱۱ bins با عمق یکسان تقسیم کردیم.

همانطور که در تصویر بالا مشاهده می‌کنید، بیشترین گروه سنی مربوط به گروه یک است که در بازه (۲۰-۳۰) سال است.



شکل ۳-۲: نمودار heatmap کواریانس مجموعه داده



شکل ۴-۲: نمودار correlation heatmap مجموعه داده

```
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')
```