دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده ریاضی و علوم کامپیوتر

درس هوش مصنوعی و کارگاه

گزارش ۵: حل و تحلیل مسئله بیماران دیابتی به روش خوشه‌بندی و طبقه بندی

نگارش

کیارش مختاری دیزجی

۹۸۳۰۰۳۲

استاد اول

دکتر مهدی قطعی

استاد دوم

بهنام یوسفی مهر

اردیبهشت ۱۴۰۲

چکیده

در این گزارش سعی شده است دیتاست دیابت ۱۳۰ که از بیمارستان‌های ایالات متحده برای سال‌های ۱۹۹۹ – ۲۰۰۸ جمع آوری شده است را ابتدا پیش‌پردازش کنیم و سپس با استفاده از بصری سازی ویژگی‌ها، درکی از ارتباط داده‌ها با یکدیگر برسیم و در نهایت نیز با استفاده از الگوریتم درخت تصمیم برای مدل دسته‌بندی و همینطور الگوریتم k-means برای مدل خوشه بندی، یک مدل پیش‌بینی برای این که آیا بیمار دوباره بستری می‌شود یا خیر بسازیم .

واژه‌های کلیدی:

پیش‌پردازش، درخت تصمیم، K-means، بصری‌سازی

لینک پروژه:

github project 5

فهرست مطالب

صفحه

فهرست اشکال

صفحه

۱. فصل اول

مقدمه

در این بخش توضیح مختصری از هر ویژگی که در دیتاست وجود دارد می‌دهیم:

- **Encounter ID:** شناسه منحصر به فردی که برای هر بار ویزیت در بیمارستان به بیمار اختصاص داده می‌شود.
- **Patient number:** شناسه منحصر به فرد برای هر بیمار
- **Race Values:** Caucasian, Asian, African American, Hispanic, and other
- **Gender Values:** male, female, and **unknown/invalid**
- **Age** Grouped in 10-year intervals: (0, 10), (10, 20), …, (90, 100)
- **Weight:** وزن بیمار که به پوند می‌باشد.
- **Admission type:** شناسه عدد صحیح مربوط به ۹ مقدار متمایز مانند emergency، urgent،...
- **Discharge disposition:** شناسه عدد صحیح مربوط به ۲۹ مقدار متمایز مانند، discharged to home، expired، .....
- **Admission source:** شناسه عدد صحیح مربوط به ۲۱ مقدار متمایز مانند physician referral، emergency room، .....
- **Time in hospital:** تعداد روزهای بین پذیرش و ترخیص بیمار
- **Payer code:** شناسه عدد صحیح مربوط به ۲۱ مقدار متمایز مانند Blue Cross/Blue Shield Medicare
- **Medical specialty:** شناسه عدد صحیح متناظر با تخصص پزشک پذیرنده بیمار با ۸۴ مقدار متمایز مانند cardiology، internal medicine، .....

- **Number of lab procedures:** تعداد آزمایشات آزمایشگاهی انجام شده در طول بستری بودن بیمار.
- **Number of procedures**: تعداد رویه‌ها (به غیر از تست های آزمایشگاهی) انجام شده در طول بستری بودن بیمار.
- **Number of medications:** تعداد نام‌های ژنریک دارو متمایز که در طول زمان بستری مصرف شده است.
- **Number of outpatient visits**: تعداد ویزیت‌های سرپایی بیمار در سال قبل از بستری شدن.
- **Number of emergency visits**: تعداد ویزیت‌های اورژانسی بیمار در سال قبل از بستری شدن.
- **Number of inpatient visits**: تعداد ویزیت های بستری بیمار در سال قبل از بستری شدن.
- **Diagnosis 1**: تشخیص اولیه (کد شده به عنوان سه رقم اول ICD9) ؛ ۸۴۸ مقدار متمایز
- **Diagnosis 2**: تشخیص ثانویه (کد شده به عنوان سه رقم اول ICD9) ؛ ۹۲۳ مقدار متمایز
- **Diagnosis 3**: تشخیص ثانویه اضافی (کد شده به عنوان سه رقم اول ICD9) ؛ ۹۵۴ مقدار متمایز
- **Number of diagnoses**: تعداد تشخیص های وارد شده به سیستم
- **Glucose serum test result** محدوده نتیجه یا عدم انجام آزمایش را نشان می‌دهد. مقادیر: «> ۲۰۰»، «>۳۰۰»، «normal» و «none» اگر اندازه‌گیری نشود.

- **A1c test:** محدوده نتیجه یا اگر آزمایش انجام نشده باشد. مقادیر: "> ۸" اگر نتیجه بزرگتر از ۸٪ بود، "> ۷" اگر نتیجه بیشتر از ۷٪ بود اما کمتر از ۸٪، "normal " اگر نتیجه کمتر از ۷٪ بود، "none" اگر نتیجه اندازهگیری نشده باشد.

- **Change of medications**: نشان می دهد که آیا تغییری در داروهای دیابتی (اعم از دوز یا نام ژنریک) وجود داشته است. مقادیر: "change" و "no change"

- **Diabetes medications**: نشان می دهد که آیا داروی دیابتی تجویز شده است یا خیر. مقادیر: "بله" و "نه"

- **Readmitted**: چند روز تا بستری مجدد بیمار، دارای مقادیر: "< ۳۰" اگر بیمار در کمتر از ۳۰ روز بستری مجدد شده است، "> ۳۰" اگر بیمار در بیش از ۳۰ روز بستری مجدد شده است، "no" برای عدم سابقه بستری مجدد.

۲. فصل دوم

مصورسازی و پیش پردازش

در ابتدا یک هستوگرام از هر ویژگی و ویژگی هدف میسازیم تا درک کلی از داده‌ها بدست بیاوریم و سپس با استفاده از این شهود سعی می‌کنیم دیتاست را برای ساخت مدل آماده کنیم.

در این بخش توضیحاتی درباره نحوه پیش‌پردازش دیتاست می‌دهیم.

در ابتدا با استفاده از متد head در پانداس ۵ سطر ابتدایی دیتاست را نمایش داده‌ایم و یک شهود کلی از ویژگی‌های موجود در آن پیدا می‌کنیم.

اولین کاری که برای پیش پردازش لازم است باید انجام دهیم این است که داده های گمشده که در این دیتاست یا ? نمایش داده شده‌اند را یا حذف یا جایگذاری کنیم بنابراین تعداد داده‌های گمشده را با روش زیر بدست می‌آوریم:

```
# Replace missing values "?" with nan
df.replace('?', float('nan'), inplace=True)

# Count the number of missing values in each column
missing_counts = df.isna().sum()
print(missing_counts)
```

بعد از این کار متوجه می‌شویم که داده‌های weight، payer_code، medical_speciality بیشترین میزان داده‌های گمشده را دارند بنابراین کل این ستون ها را حذف می‌کنیم.

برای ویژگی race با توجه به ماهیت داده هایش و همینطور کم بودن تعداد داده‌های گمشده آن از روش مد گرفتن استفاده کردیم و داده‌های گمشده را با مد جایگذاری کرده‌ایم:

```
# Using mode to replace missing values for race attribute
mode_race = df['race'].mode()[0]
df['race'].fillna(mode_race, inplace=True)
print(df['race'].head(25))
```

و همین روش را برای diag_1، diag_2، diag_3، نیز به طور مشابه انجام داده‌ایم:

```
diag_1_mode = df['diag_1'].mode()[0]
diag_2_mode = df['diag_2'].mode()[0]
diag_3_mode = df['diag_3'].mode()[0]

df['diag_1'] = df['diag_1'].fillna(diag_1_mode)
df['diag_2'] = df['diag_2'].fillna(diag_2_mode)
df['diag_3'] = df['diag_3'].fillna(diag_3_mode)
```

با توجه به توضیحاتی که درباره هر ویژگی در بخش مقدمه داده شد متوجه شدیم که داده‌های گمشده gender به صورت unknown/invalid است بنابراین ابتدا تعدادشان را چک و سپس با توجه به اینکه تعدادشان در حد ۳ عدد است، آن سه سطر را از دیتاست حذف می‌کنیم:

```
print('gender missing value before droping', df['gender'][df['gender'] ==
'Unknown/Invalid'].count())
df = df[df['gender'] != 'Unknown/Invalid']
print('gender missing value after droping', df['gender'][df['gender'] ==
'Unknown/Invalid'].count())
```

حال که همه داده‌های گمشده را حذف کردیم باید با توجه به وجود دو ویژگی patient_nbr و encounter_id می توان احتمال داد که برخی از سطرها داده‌های یکسان باشند بنابراین با استفاده از کد زیر این داده‌های مشابه را از دیتاست حذف می‌کنیم:

```
df.drop_duplicates(['patient_nbr'], keep = 'first', inplace = True)
```

و همینطور داده‌های patient_nbr و encounter_id را در انتها به دلیل بی ارزش بودن در مرحله تولید مدل، از دیتاست حذف می‌کنیم.

سپس اگر به ستون دو ویژگی citoglipton و examide توجه کنیم می‌بینیم مقدار همه داده‌ها no می‌باشد بنابراین این دو ویژگی بی ارزشی است و آن‌ها را حذف می‌کینم:

```
print("citoglipton: ", df['citoglipton'].value_counts())
print("examide: ", df['examide'].value_counts())
df = df.drop(['citoglipton', 'examide'], axis = 1)
```

حال که داده‌های گمشده و تکراری و بی ارزش از دیتاست حذف شده‌اند لازم است که مقادیر برخی از ویژگی‌ها را به صورت دلخواه دربیاوریم تا بتوان از داده‌ها در ساخت مدل استفاده کرد. لازم به ذکر است که این بخش با کمک [1] انجام شده است.

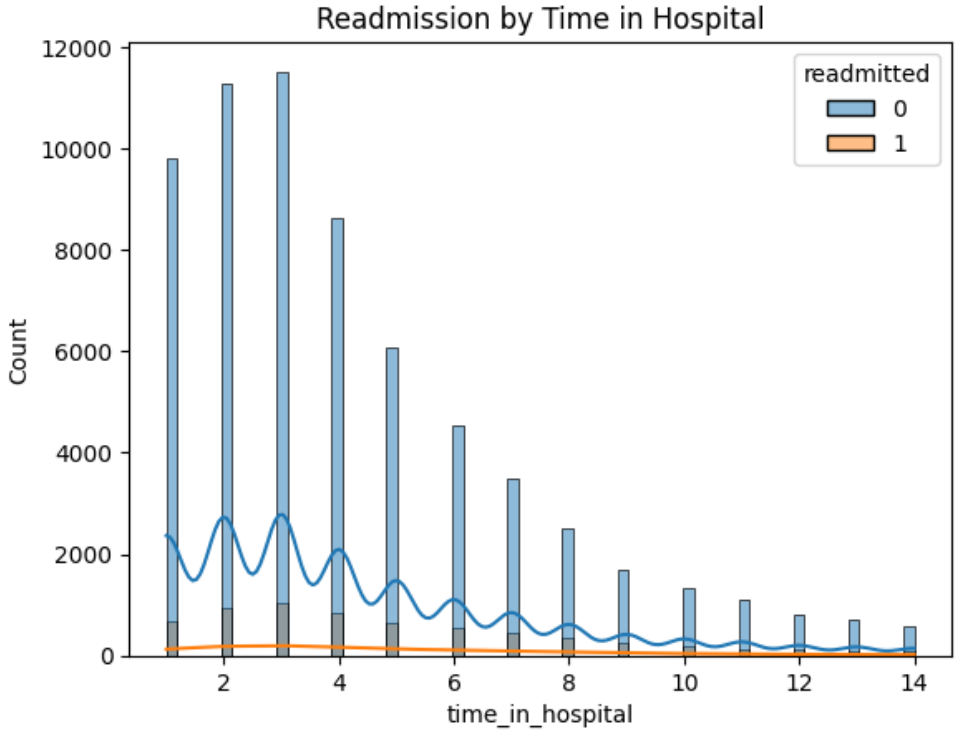
در ابتدای کار لازم است که ویژگی age که به صورت بازه است به گسسته سازی کنیم و برای همین وسط هر بازه را نماینده هر بازه در نظر و به آن مپ می‌کنیم:

```
replaceDict = {'[0-10)' : 5, '[10-20)' : 15, '[20-30)' : 25,  '[30-40)' :
35,  '[40-50)' : 45,  '[50-60)' : 55, '[60-70)' : 65,
               '[70-80)' : 75, '[80-90)' : 85, '[90-100)' : 95}

df['age'] = df['age'].apply(lambda x : replaceDict[x])
print(df['age'].head())
```

مشابه همین کار را برای ویژگی‌های diag_1، diag_2، diag_3، discharge_disposition_id، admission_id_type، admission_source_id، max_glu_serum، A1Cresult، medication_cols، change، diabetesMed انجام می‌دهیم و دسته‌ای از مقادیر را به یک مقدار کلی‌تر اختصاص می‌دهیم تا از تعداد زیاد این مقادیر کمی کاسته شود.

در ادامه برای درک ارتباط دقیق‌تر برخی از ویژگی‌هایی که تصور می‌شد ارتباط زیادی با ویژگی هدف داشته باشند انجام دادیم. به عنوان مثال زمان ماندن در بیمارستان بیشتر بوده احتمال اینکه بیمار دوباره بستری شود کاهش پیدا کرده است.



۱- مقایسه زمان بستری ماندن در بیمارستان با دوباره بستری شدن بیمار

۳. فصل سوم

ساخت مدل‌های دسته بندی و خوشه بندی و ارزیابی آنها

در این مرحله با استفاده از تابع آماده درخت تصمیم پکیج Scikit-learn یک مدل دسته بندی از این دیتاست ایجاد کرده‌ایم:

```
X = df.drop('readmitted', axis=1) # features
y = df['readmitted'] # target feature
# split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Get the categorical features
cat_features = ['race', 'gender', 'diag_1', 'diag_2', 'diag_3']

# Create a label encoder object
encoder = LabelEncoder()

# Encode the categorical features in the training and testing data
for feature in cat_features:
    X_train[feature] = encoder.fit_transform(X_train[feature])
    X_test[feature] = encoder.transform(X_test[feature])

# Create a decision tree classifier object
dt = DecisionTreeClassifier(random_state=42)

# Train the model on the training data
dt.fit(X_train, y_train)

y_pred = dt.predict(X_test)

# Evaluate the model's accuracy and classification report
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
print(classification_report(y_test, y_pred))
```

لازم به ذکر است که ابتدا برخی از داده‌ها که مقادیرشان عددی نبود با استفاده از آبجکت LabelEncoder آن‌ها را به مقادیر عددی تبدیل کرده‌ایم و سپس با جدا کردن داده‌ها به داده‌های تست و ترین مدل را روی داده‌ی ترین فیت و در آخر با استفاده از داده‌ی تست پریدیکت کرده‌ایم.

در ادامه نیز یک مدل خوسه بندی با استفاده از الگوریتم K-means ایجاد کرده‌ایم:

```
# select the data
X = df[features]

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# create a k-means object with 3 clusters
kmeans = KMeans(n_clusters=2, random_state=42)

# fit the k-means model to the data
kmeans.fit(X_scaled)

# get the cluster labels for each data point
labels = kmeans.labels_
df['cluster_labels'] = labels
print(df['cluster_labels'].value_counts())
```

در اینجا هم پس از انکد کردن داده‌هایی که مقادیر عددی ندارند مقادیر را به یک اسکیل آورده‌ایم و سپس با استفاده از آبجکت KMeans روی آن فیت و لیبل‌های ایجاد شده را استخراج کرده ایم.

```
Accuracy: 0.8220344998935188
              precision    recall  f1-score   support

           0       0.91      0.89      0.90     12822
           1       0.12      0.15      0.13      1265

    accuracy                           0.82     14087
   macro avg       0.52      0.52      0.52     14087
weighted avg       0.84      0.82      0.83     14087
```

با توجه به مقادیر با نشان میدهد که مدل درخت تصمیم ما با دقت ۰.۸۲ می‌تواند تشخیص دهد آیا احتمال دارد بیمار دوباره بستری شود یا خیر. همینطور در ادامه هم ارزیابی های precision، recall، f1-score را در جدول میتوان دید و دقت کلی مدل ۰.۸۲ است، به این معنی که ۸۲ درصد از نمونه‌های مجموعه داده به درستی توسط مدل طبقه بندی شده اند. میانگین precision، recall f1-score ۰.۵۲ است که میانگین نمرات هر دو کلاس است. میانگین وزنی precision ، recall و f1-score ۰.۸۴است که عدم تعادل در مجموعه داده را در نظر می‌گیرد و وزن بیشتری به عملکرد مدل در کلاس اکثریت می‌دهد.

منابع

[1] https://medium.com/analytics-vidhya/diabetes-130-us-hospitals-for-years-1999-2008-e18d69beea4d

[2] https://www.kaggle.com/code/iabhishekofficial/prediction-on-hospital readmission

[3] https://www.kaggle.com/code/alibaris/eda-vis-on-diabetes-data

لینک پروژه:

github project 5