دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

تمرین پیاده‌سازی ۲

پیاده‌سازی درخت تصمیم برای تشخیص دیابت

محمدرضا دولتی

۹۷۱۱۰۴۱۱

استاد:

دکتر آرش عبدی هجراندوست

دی ۱۴۰۰

گزارش کد:

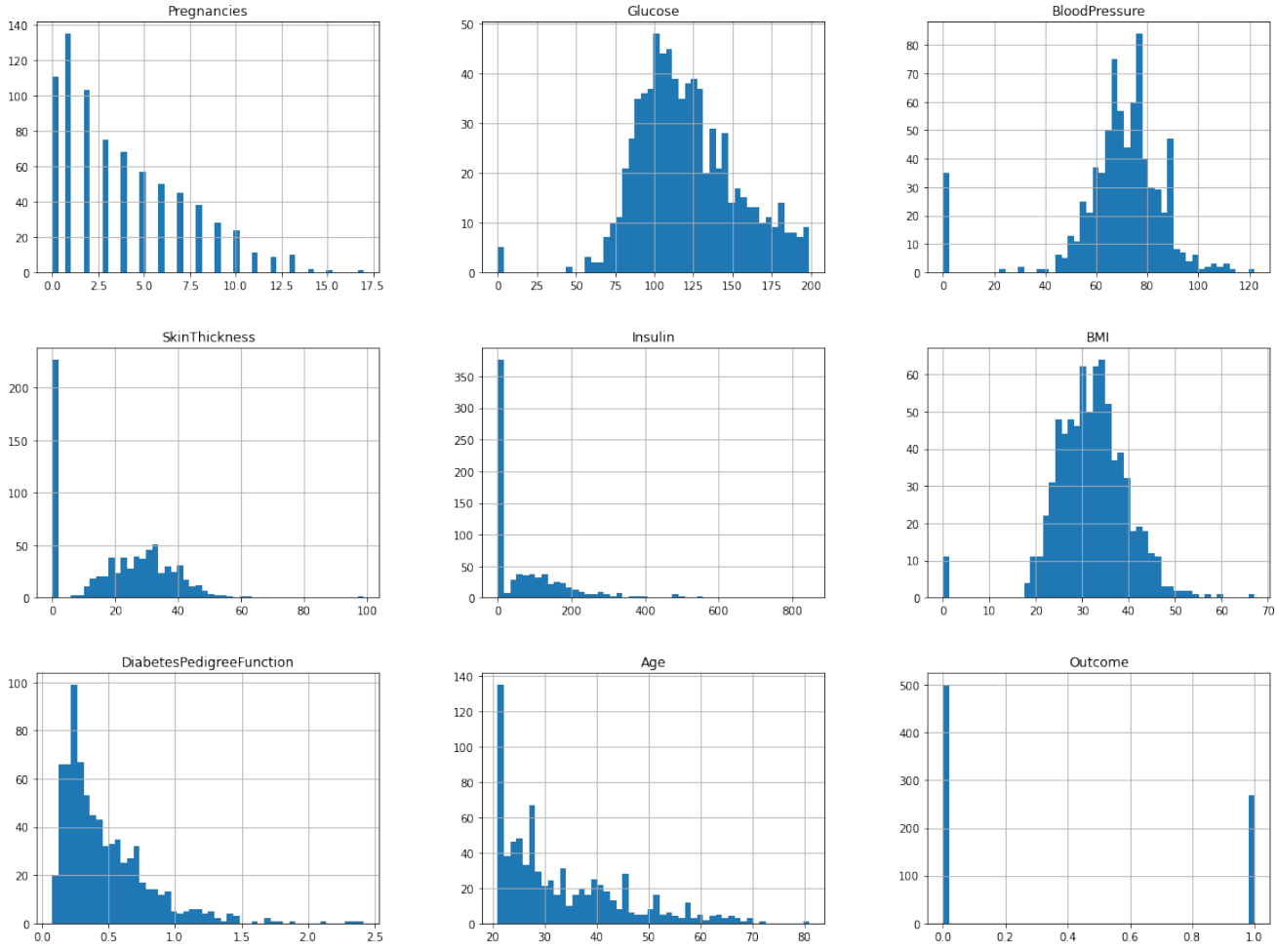
در ابتدا در قسمت اول پکیج های مورد نظر را ایمپورت کردیم(آخرین صفحه) و سپس فایل دیابت را خواندیم و این فایل را به صورت Text نیز در قسمت بعد آوردیم.

در دو قسمت بعدی آمدیم کد محاسبه دو تابع آنتروپی و gain را زدیم که تصاویر آن نیز در ادامه آمده است:

```
# Define the calculate entropy function
def calculate_entropy(df_label):
    classes,class_counts = np.unique(df_label,return_counts = True)
    entropy_value = np.sum([(-class_counts[i]/np.sum(class_counts))*np.log2(class_counts[i]/np.sum(class_counts))
                      for i in range(len(classes))])
    return entropy_value
```

```
# Define the calculate information gain function
def calculate_information_gain(dataset,feature,label):
    # Calculate the dataset entropy
    dataset_entropy = calculate_entropy(dataset[label])
    values,feat_counts= np.unique(dataset[feature],return_counts=True)

    # Calculate the weighted feature entropy                                    # Call the calculate_entropy function
    weighted_feature_entropy = np.sum([(feat_counts[i]/np.sum(feat_counts))*calculate_entropy(dataset.where(dataset[feature]
                           ==values[i]).dropna()[label]) for i in range(len(values))])
    feature_info_gain = dataset_entropy - weighted_feature_entropy
    return feature_info_gain
```

در قسمت بعد نیز تابع توزیع تمامی ۸ ویژگی‌ای که داریم را آوردیم تا یک دید کلی از نحوه توزیع مقادیر داخل جدول داشته باشیم که تصویر آن نیز در ادامه آمده است:



همچنین برای هندل کردن مقادیر صفر و NULL باید توجه کرد که صفرهای نشان داده شده در جدول صفر نیستند بلکه مقادیر NULL هستند و این درحالی است که بعضی از ویژگی ها مانند skin thickness insulin, IBM نمی‌توانند مقادیر NULL داشته باشند:

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0
0	137	40	35	168	43.1	2.288	33	1
5	116	74	0	0	25.6	0.201	30	0
3	78	50	32	88	31	0.248	26	1
10	115	0	0	0	35.3	0.134	29	0
2	197	70	45	543	30.5	0.158	53	1
8	125	96	0	0	0	0.232	54	1
4	110	92	0	0	37.6	0.191	30	0
10	168	74	0	0	38	0.537	34	1
10	139	80	0	0	27.1	1.441	57	0
1	189	60	23	846	30.1	0.398	59	1

مقادیر NULL هر ستون با میانگین همان ستون جایگزین می‌شود:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	\
0	6	148.0	72.000000	35.00000	95.674746	33.600000	
1	1	85.0	66.000000	29.00000	95.674746	26.600000	
2	8	183.0	64.000000	25.09192	95.674746	23.300000	
3	1	89.0	66.000000	23.00000	94.000000	28.100000	
4	0	137.0	40.000000	35.00000	168.000000	43.100000	
5	5	116.0	74.000000	25.09192	95.674746	25.600000	
6	3	78.0	50.000000	32.00000	88.000000	31.000000	
7	10	115.0	72.533517	25.09192	95.674746	35.300000	
9	8	125.0	96.000000	25.09192	95.674746	32.056663	
10	4	110.0	92.000000	25.09192	95.674746	37.600000	

	DiabetesPedigreeFunction	Age	Outcome
0	0.627	50	1
1	0.351	31	0
2	0.672	32	1
3	0.167	21	0
4	2.288	33	1
5	0.201	30	0
6	0.248	26	1
7	0.134	29	0
9	0.232	54	1
10	0.191	30	0

در قسمت های بعد برای محاسبه راحتتر و پیش‌بینی افرادی که دیابت دارند میانه و میانگین هرکدام از ۸ ویژگی گفته شده را محاسبه کردیم تا مقدار آن و مقایسه آن با دیگر ویژگی‌ها را بدست آوریم.

به به عنوان مثال برای ویژگی BMI داریم:

```
median_bmi = df['BMI'].median()
mean_bmi = df['BMI'].mean()
print("The median BMI is :",median_bmi)
print("The mean BMI is :",mean_bmi)
```

```
The median BMI is : 32.0
The mean BMI is : 31.992578124999977
```

در مرحله بعد الگوریتم اصلی درخت تصمیم را نوشتیم که داده‌ها نیز به دو مجموعه آموزشی و آزمایشی ۶۰ و ۴۰ درصد تقسیم شده‌اند و هدف نیز ستون outcome است که وابسته به حاصل ۸ ویژگی قبل است. خروجی outcome به صورت باینری و صفر و یکی است.

فیچری که تعیین شده است نیز مقادیر تمامی این ۸ ویژگی را در خود ذخیره می‌کند.

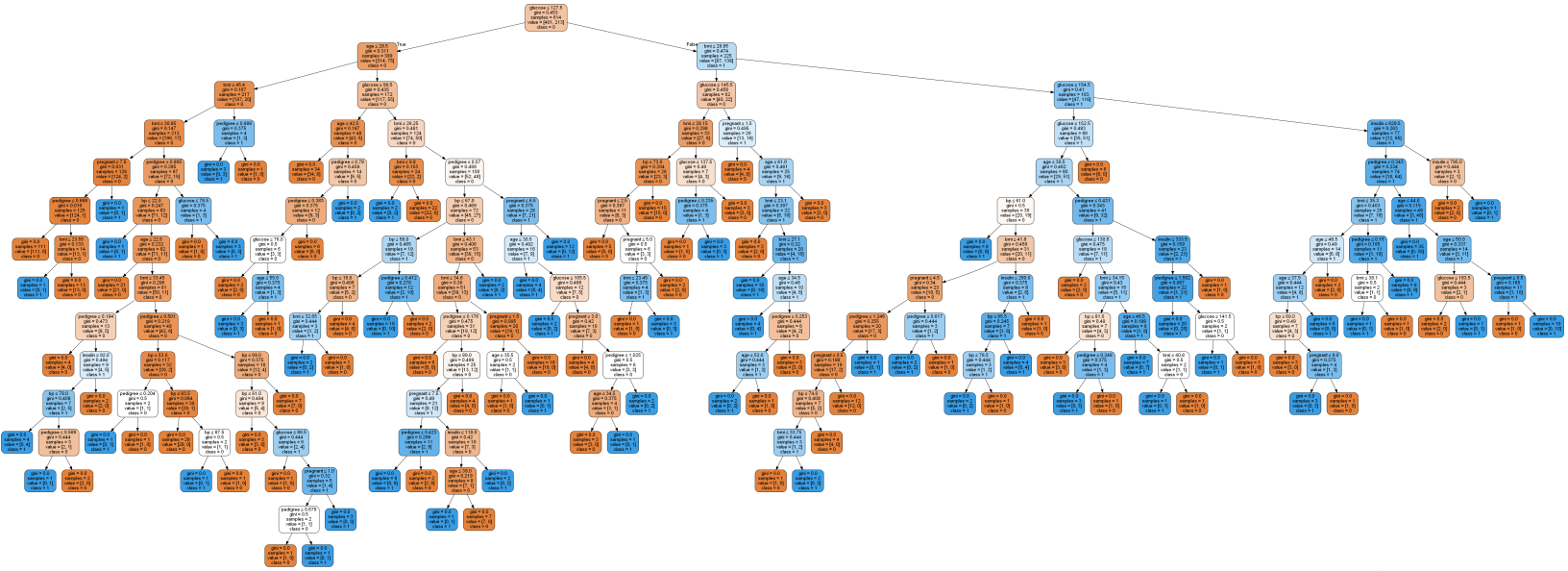
در سلول بعدی آمدیم کد پیش بینی تصمیم درست را زدیم و درصد آن را محاسبه کردیم تا بفهمیم به احتمال چه درصدی تصمیممان درست بوده است.

که برای این الگوریتم ۷۴ درصد دقت درستی داریم:

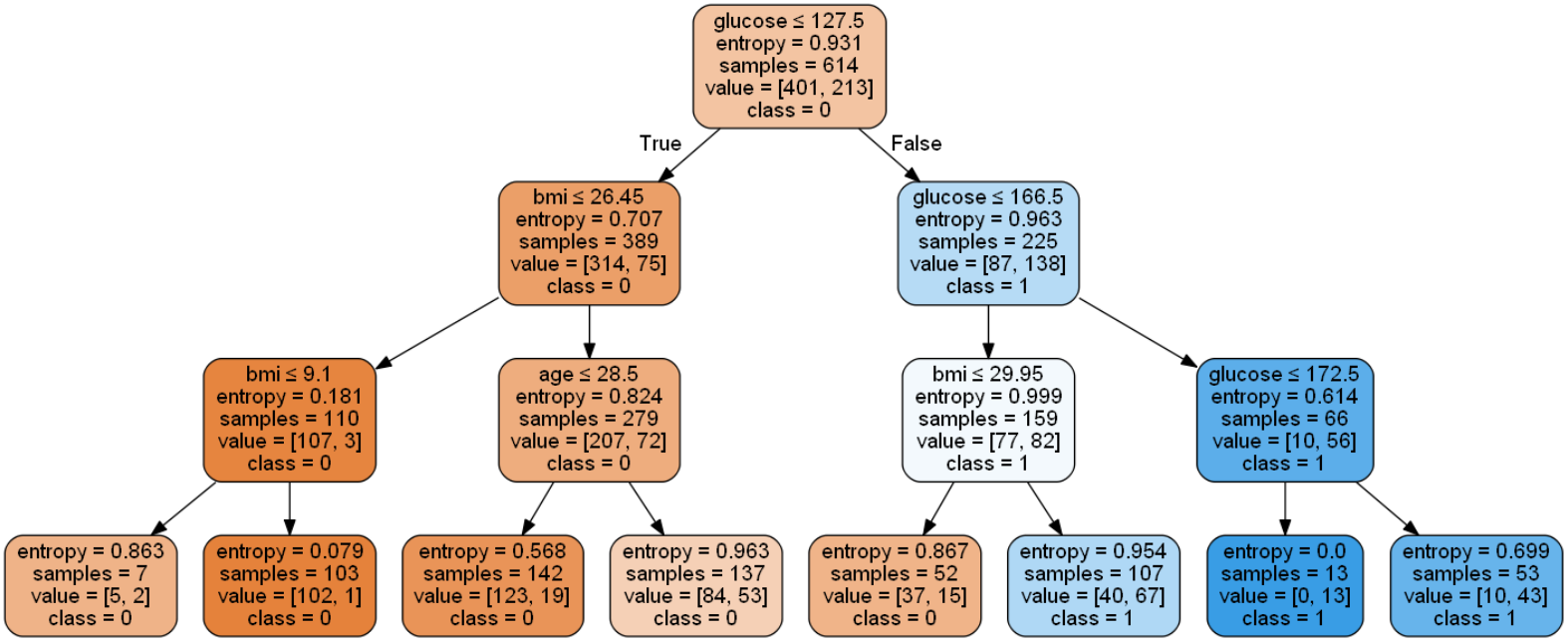
```
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=21)
neigh.fit(feature,target)
TrueDecisionpredicted = neigh.predict(test_input)
print(metrics.classification_report(expected, TrueDecisionpredicted))
print(metrics.confusion_matrix(expected, knnpredicted))
print("TrueDecision accuracy: ",neigh.score(test_input,expected))
TrueDecisionscore=neigh.score(test_input,expected)
```

```
              precision    recall  f1-score   support

           0       0.80      0.82      0.81       206
           1       0.61      0.59      0.60       102

    accuracy                           0.74       308
   macro avg       0.71      0.70      0.70       308
weighted avg       0.74      0.74      0.74       308

[[168  38]
 [ 42  60]]
TrueDecision accuracy:  0.7402597402597403
```

در آخر هم درخت تصمیم را آورده‌ایم به این صورت که اولین گره این درخت با ویژگی گلوکز شروع می‌شود و همچنین outcome که صورت باینری و صفر و یکی و نتیجه ما نیز می‌باشد به این صورت که اگر ۰ باشد، دیابت نداریم و اگر ۱ باشد دیابت داریم با نام class در درخت آورده شده است Sample در این درخت تعداد تعداد کسانی است که به طور مثال بعد از گره اول اگر گلوکز کمتر از ۱۲۷.۵ داشته‌اند True بوده‌اند و اگر گلوکز بیش‌تر از ۱۲۷.۵ داشته‌اند False شده‌اند و همچنین مقدار دست‌آورد اطلاعات و آنتروپی در تمامی گره‌ها مشخص شده است:



اگر عمق درخت ۴ باشد:



پکیج‌ها و ایمپورتشان:

NumPy: Base n-dimensional array package
SciPy: Fundamental library for scientific computing
Matplotlib: Comprehensive 2D/3D plotting
IPython: Enhanced interactive console
Sympy: Symbolic mathematics
Pandas: Data structures and analysis