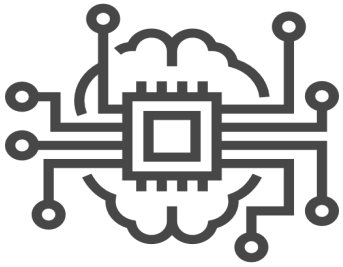
HomeWork7 AI

Milad Barooni
9632459



این تکلیف با استفاده از ابزار jupyter notebook و ipython انجام شده است. در هر قسمت سعی شده که با قرار دادن header های مناسب و markdown در کد به کارکرد هر قسمت اشاره کنیم. مشخص است که برای گرفتن نتیجه‌ی هر قسمت باید قسمت اول اجرا شود تا dataframe آماده برای استفاده شود.

قسمت اول: آماده سازی داده ها برای یادگیری

در این قسمت همه‌ی کارهای خواسته شده در سوال اول را به ترتیب پیاده سازی شده است. تمام این کارها با استفاده از کتابخانه‌ي مهم pandas انجام شده است. میدانیم که وقتی csv را میخوانیم pandas به آن به چشم یک dataframe نگاه میکند و تمام کارهایی که انجام میدهیم در واقع داریم این dataframe را عوض میکنیم. در انتها ما یک قسمت train_x و یک train_y داریم. از این دو قسمت برای یادگیری classifier هایمان استفاده میکنیم. قسمت test_x را به مدل میدهیم که روی آن آنچه را یاد گرفته است پیاده سازی کند. و test_y در واقع مقادیر واقعی ای است که در ازای test_x ما باید به دست آوریم.

قسمت دوم: یادگیری با شیوه‌ی gini index

در این قسمت با استفاده از مثال های ذکر شده در خود سایت sklearn یک مدل decision tree پیاده سازی شده است. ابتدا مدل با پارامتر criterion ساخته میشود. سپس train_x و train_y را میدهیم. سپس یک predict_y با استفاده از مدل به دست می آوریم.سپس با استفاده از کتابخانه‌ی موجود در sklearn به اسم metrics مقادیر خواسته شده چاپ میشوند. که در تصویر زیر در اجرای انجام شده نشان داده شده است. مشخص است که این مقادیر در هر اجرا متفاوت است زیرا که داریم data ها را shuffle میکنیم. سپس تصویر خواسته شده را رسم کرده‌ایم که در ضمیمه‌ی پروژه با اسم decistion_tree_gini.png وجود دارد.

```
Accuracy: 0.7046979865771812
Confusion matrix: [[57 24]
 [20 48]]
Precision: [[57 24]
 [20 48]]
Recall: 0.7058823529411765
F1: 0.6857142857142857
```

قسمت سوم: یادگیری با شیوه‌ی Information gain

در این قسمت مدل خواسته شده در سوال سوم پیاده سازی شده است. تمامی کارهای انجام شده مطابق قسمت دوم است، تنها تفاوت آن در پارامتر criterion است که طبق مستند موجود در سایت sklearn نوشته شده است که برای استفاده از مدل information gain ما باید این پارامتر را مطابق کد قرار دهیم. نتایج به دست آمده در زیر موجود است و هم اینکه تصویر برای این مدل با استفاده از کتابخانه‌ی matplotlib ساخته شده است مثل قسمت قبل. این تصویر با نام decistion_tree_info_gain.png موجود است.

```
Accuracy: 0.6845637583892618
Confusion matrix: [[56 25]
 [22 46]]
Precision: [[56 25]
 [22 46]]
Recall: 0.6764705882352942
F1: 0.6618705035971224
```

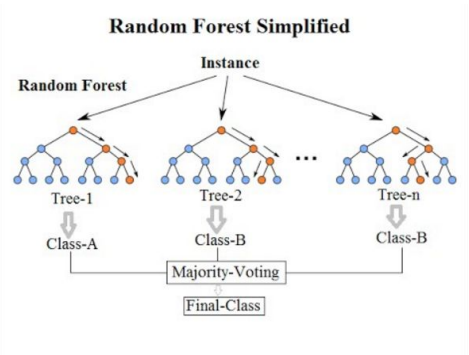
قسمت چهارم: پیدا کردن عمق مناسب برای یادگیری با شیوه‌ی Cross Validation

در این قسمت باید دقت هر کدام از عمق های یادگیری با روش ذکر شده پیدا کرد. پس یک حلقه روی هر کدام از عمق ها میزنیم. روی هر کدام یک مدل میسازیم و داده ها به آن میدهیم. سپس طبق مستند ارائه شده در سایت sklearn باید امتیاز را با متد cross_val_score بدست آوریم و سپس باید با استفاده از متد mean میانگین را پیدا کنیم. برای رسم کردن نمودار آن مقادیر را در یک آرایه ذخیره کردیم و با استفاده از کتابخانه‌ی matplotlib در کد این نمودار رسم شده است. همچنین نتایج به دست آماده در اجرای صورت گرفته در کامپیوتر شخصی در تصویر زیر آماده است. واضح است که با استفاده از این متد میتوانیم به دقت بهتری نسبت به قسمت های قبلی برسیم. این یعنی میتوانیم با پیدا کردن یک عمق مشخص به جواب و در واقع مدل بهتری برای تشخصی خبر های صحیح و جعلی برسیم.

```
8 :  71.14406779661017
9 :  70.63276836158192
10 :  70.29378531073446
11 :  70.9858757062147
12 :  70.48022598870057
13 :  71.4858757062147
14 :  72.15819209039547
15 :  71.64406779661017
16 :  72.65254237288136
17 :  70.9689265536723
```

قسمت پنجم: توضیح الگوریتم Random Forest

طبق مطالب گفته شده در سایت ویکی پدیا داریم:



یک روش یادگیری ترکیبی برای دسته‌بندی، رگرسیون می‌باشد، که بر اساس ساختاری متشکل از شمار بسیاری درخت تصمیم، بر روی زمان آموزش و خروجی کلاس‌ها (کلاس‌بندی) یا برای پیش‌بینی‌های هر درخت به شکل مجزا، کار می‌کنند. جنگل‌های تصادفی برای درختان تصمیم که در مجموعهٔ آموزشی دچار بیش برازش می‌شوند، مناسب هستند. عملکرد جنگل تصادفی معمولا بهتر از درخت تصمیم است، اما این بهبود عملکرد تا حدی به نوع داده هم بستگی دارد.

بنابر تعریف گفته شده یعنی از چند درخت تصمیم(decision tree) استفاده می کند و روی زمان یادگیری روی هر کدام از درخت های تصمیم و ... توجه می کند و از اینها استفاده میکند برای اینکه ببیند کدام بهتر عمل میکند. این روش سعی میکند که بر خلاف روش درخت تصمیم واریانس خروجی را کاهش دهد. به این ترتیب که بهترین عمق موجود در روش درخت تصمیم را استفاده میکند. سعی میکند که عمق را تا میتواند کم نگه دارد. برای انجام این کار این مدل سعی در میانگین گیری روی درخت های تصمیم متفاوت با عمق های متفاوت دارد.

قسمت ششم: پیاده سازی الگوریتم Random Forest

تقریبا مشابه کاری که در قسمت های دوم و سوم انجام شد ما با استفاده از کتابخانه‌ی random forest توانستیم داده های را دسته بندی یا classify کنیم. در واقع اخبار صحیح را از اخبار جعلی جدا کنیم. در اجرای انجام شده نتیجه‌ی زیر بدست آمد. در قسمت های قبل ما دقت را در ۱۰۰ ضرب نکردیم اما در این قسمت در ۱۰۰ ضرب شده است.

```
Accuracy: 78.52348993288591
```

نکته‌ی جالب این است که این دقت از تمام روش های قبل و حتی عوض شدن عمق هم بهتر عمل میکند و این نشان میدهد که این روش چقدر موثر است.