هوش مصنوعی

تمرین شماره‌ی ۳

مدرسین: **دکتر فدایی و دکتر یعقوب‌زاده**

طراحان: حسام رمضانیان، جواد کاویان، محمد امانلو

مهلت تحویل: جمعه ۳۰ آذر ۱۴۰۳، ساعت ۲۳:۵۹

مقدمه

هدف این تمرین، آشنایی با روش‌های یادگیری ماشین[1] برای پیش‌بینی قیمت هتل‌ها بر اساس ویژگی‌های موجود در یک سایت رزرو آنلاین هتل است. این پروژه شامل سه فاز اصلی است: ابتدا در فاز اول به آماده‌سازی محیط و داده‌ها می‌پردازیم تا زیرساخت‌های لازم برای تحلیل و مدل‌سازی را فراهم کنیم. در فاز دوم، تحلیل اکتشافی داده[2] و مهندسی ویژگی‌ها[3] انجام می‌شود تا با بررسی داده‌ها، الگوهای مفید شناسایی شده و ویژگی‌های مناسب برای مدل‌سازی انتخاب و آماده‌سازی شوند. در نهایت، در فاز سوم، مدل‌های یادگیری ماشین توسعه داده شده، آموزش می‌بینند و با استفاده از معیارهای ارزیابی مناسب مورد سنجش قرار می‌گیرند تا دقت و عملکرد آن‌ها در پیش‌بینی قیمت هتل‌ها ارزیابی شود.

[1] Machine Learning

[2] Exploratory Data Analysis

[3] Feature Engineering

آشنایی با مجموعه داده

مجموعه داده‌ای که در اختیار شما قرار دارد شامل اطلاعاتی درباره‌ی قیمت هتل‌ها بر اساس تعداد افراد و شب‌های اقامت است. علاوه بر این، بخشی از اطلاعات مربوط به هر هتل که در صفحه‌ی آن در سایت نمایش داده می‌شود نیز در این مجموعه گنجانده شده است.

توضیحات مربوط به ستون‌های این مجموعه داده در جدول زیر ارائه شده است:

نام ستون	توضیح
name	نام هتل.
location	موقعیت مکانی هتل (ناحیه و شهر).
price	قیمت اقامت در هتل به ازای تعداد شب ها و افراد ذکر شده
rating	امتیاز کلی هتل بر اساس بازخورد کاربران (عددی بین 0 تا 10).
quality	توصیف کیفی امتیاز هتل (مانند "Very good"، "Good"، یا "Fabulous").
review	تعداد کل نظرات ثبت‌شده توسط کاربران درباره هتل.
bed	نوع تخت موجود در اتاق (مثلاً "double bed 1" یا "single bed 1").
size	اندازه اتاق (به مترمربع).
distance from center	فاصله هتل از مرکز شهر (به کیلومتر).
room type	نوع اتاق رزرو شده (مثلاً "Single Bed in 6-Bed Dormitory Room"، "Suite").
nights	تعداد شب‌های اقامت.
adults	تعداد بزرگسالان در رزرو.
free cancellation	امکان لغو رایگان رزرو .

آماده‌سازی محیط و تحلیل اکتشافی داده

در این بخش ابتدا باید مجموعه داده را بارگذاری کرده و آن را به فرمت DataFrame در Pandas تبدیل کنید تا امکان پردازش و تحلیل آسان‌تر و سریع‌تر فراهم شود.

پس از بارگذاری دادگان نیاز است تا در اولین قدم از ویژگی‌های مختلف دادگان خام مطلع شویم و به شناخت خوبی از دادگان خود برسیم تا با استفاده از این شناخت در مراحل بعدی بهترین تصمیم‌ها را بگیریم. به فاز اولیه تجزیه و تحلیل دادگان تحلیل اکتشافی داده یا Exploratory Data Analysis (به اختصار EDA) گفته می‌شود. یکی از مهم‌ترین روش‌های انجام EDA استفاده از مصورسازی دادگان[4] است.

در این بخش، هدف استفاده از تکنیک‌های مختلف تجسم داده‌ها برای تحلیل بهتر داده‌ها و شناسایی الگوهای موجود است. با استفاده از نمودارها و گراف‌ها، می‌توان به راحتی روندها، روابط و الگوهای پنهان در داده‌ها را مشاهده کرده و این اطلاعات را برای تصمیم‌گیری‌های بهینه در مراحل بعدی تحلیل یا مدل‌سازی استفاده کرد. با توجه به این موضوع در هر یک از مراحل این بخش، بیان استنتاج مربوطه که بتوان دانشی از آن برای پیشبرد مراحل بعدی پروژه داشت، حائز اهمیت است.

برای انجام کامل این بخش، مراحل زیر توصیه می‌شود:

- ساختار کلی داده‌ها را بدست آورید. (برای این کار می‌توانید از متدهای info و describe استفاده کنید)
- نمودار تعداد مقادیر منحصر به فرد برای هر ویژگی را رسم کنید.
- نمودار وابستگی بین ویژگی‌ها را رسم کرده و بگویید که کدام ویژگی بیشترین وابستگی را با ستون هدف دارد.
- نمودارهای Scatter و Hexbin معمولا برای بررسی ارتباطات استفاده می‌شوند. از این نمودارها برای بررسی ارتباط متغیرهای مستقل با متغیر وابسته استفاده کنید.
- درباره بررسی‌های دیگری که در این بخش حائز اهمیت است تحقیق کنید و یکی از آن‌ها را انجام دهید.

پیش‌پردازش دادگان و مهندسی ویژگی‌ها

مهم‌ترین بخش هر پروژه یادگیری ماشین، بخش پیش‌پردازش داده‌ها می‌باشد. در این فاز فرمت داده‌ها را تغییر داده، اصلاح یا خلاصه می‌کنیم. چرا که در دنیای واقعی اطلاعات جمع آوری شده به راحتی کنترل

[4] Data Visualization

نمی‌شوند و در نتیجه مقادیر خارج از محدوده، ناممکن، از دست رفته و به طور کلی گمراه کننده در دادگان وجود دارند. این فاز باعث می‌شود مدل کارا تری را توسعه دهیم.

برای انجام کامل این بخش، مراحل زیر توصیه می‌شود.

- ابتدا داده‌های از دست رفته را باید تکمیل کنیم، برای انجام این کار روش‌های مختلفی وجود دارد. با توجه به شناختی که در بخش قبلی بدست آوردید 3 روش که به نظر شما برای هر ویژگی مناسب تر است را انتخاب و اجرا کنید.
- در صورتی که امکان حذف برخی از ستون‌ها وجود دارد، آن‌ها را حذف کنید. دلیل آن را توضیح دهید.
- در صورت نیاز، داده‌های عددی را از ستون‌های متنی استخراج کنید.
- با استفاده از ستون‌هایی که از پیش در دادگان موجود است، ستون‌های اضافی که برای تحلیل‌های بعدی یا مدل‌سازی مورد نیاز است، به داده‌ها اضافه کنید. اصطلاحا به این کار مهندسی ویژگی می‌گویند.
- و داده‌های دسته‌بندی‌شده[5] را به مقادیر عددی تبدیل کنید تا برای مدل‌های یادگیری ماشین قابل استفاده باشند.

در قدم بعدی هدف این است که هتل‌ها را بر اساس قیمت آن‌ها به دو گروه تقسیم کنیم. دلیل تبدیل ستون "قیمت" به یک ستون دسته‌بندی‌شده این است که قرار است مدل‌های دسته‌بندی[6] آموزش داده شوند. مدل‌های دسته‌بندی برای پیش‌بینی متغیرهای دسته‌ای طراحی شده‌اند و قادر به پردازش داده‌های عددی پیوسته مانند قیمت نیستند. بنابراین، برای استفاده از این مدل‌ها، باید داده‌های پیوسته را به گروه‌های مشخص تقسیم کرد. در این حالت، قیمت‌ها بر اساس یک آستانه مشخص به دو دسته "هتل‌های ارزان‌تر" (برچسب ۰) و "هتل‌های گران‌تر" (برچسب ۱) تقسیم می‌شوند. این آستانه از طریق محاسبه میانه قیمت‌ها تعیین می‌شود. به این ترتیب، داده‌ها به صورت دسته‌بندی‌شده تبدیل می‌شوند که برای مدل‌های classification مناسب است.

توسعه، آموزش و ارزیابی مدل‌ها

در این بخش، هدف اصلی طراحی، آموزش و ارزیابی مدل‌های یادگیری ماشین، برای حل مسئله Classification است. این فرآیند به‌گونه‌ای تنظیم شده است که شما تمامی مراحل، از آماده‌سازی داده‌ها

[5] Categorical
[6] Classification

گرفته تا توسعه و ارزیابی مدل‌های پیشرفته، را به‌صورت عملی تجربه کنید. در ادامه، توضیحاتی کلی درباره این مراحل ارائه شده است:

1. **Train-Test Split**

Train-Test Split روشی برای تقسیم داده‌ها به دو مجموعه آموزش (train) و تست (test) است. این تقسیم معمولاً برای ارزیابی عملکرد مدل‌ها استفاده می‌شود و از ایجاد overfitting جلوگیری می‌کند، زیرا مدل فقط روی داده‌های آموزش آموزش می‌بیند و عملکرد آن روی داده‌های تست ارزیابی می‌شود.

داده‌ها باید به دو بخش تقسیم شوند: ۸۰٪ برای آموزش و ۲۰٪ برای آزمون. این کار برای ارزیابی عملکرد مدل‌ها ضروری است و به جلوگیری از Overfitting کمک می‌کند.

2. **Normalization/Standardization**

Normalization/Standardization نقش مهمی در بهبود عملکرد مدل‌ها دارند. در این بخش، شما باید اهمیت این مرحله را بررسی کرده و روش‌های مختلف آن را بررسی کنید. دلیل استفاده از روش مورد نظر برای این مرحله را طبق EDA اولیه بیان کنید.

همچنین در صورت نیاز از Transformer ها نیز می‌توانید استفاده کنید.

3. **Sklearn Models**

- **Naive Bayes**

الگوریتم Naive Bayes یک الگوریتم یادگیری ماشین ساده و قدرتمند است که مبتنی بر قانون بیز است. این الگوریتم برای مسائل دسته‌بندی مورد استفاده قرار می‌گیرد که بر اساس آن فرض می‌شود ویژگی‌ها مستقل از هم هستند، حتی اگر در واقع اینگونه نباشند. به همین دلیل نام این الگوریتم Naive یا ساده است.
با استفاده از کتابخانه‌های موجود این الگوریتم را پیاده سازی اجرا و مطابق بخش ارزیابی، ارزیابی کنید.

- **Decision Tree**

درخت تصمیم یک مدل پیش‌بینی است که از ساختار درختی برای تصمیم‌گیری در مورد مقدار یک متغیر هدف استفاده می‌کند. این درخت از گره‌ها و لیستی از تقسیم‌ها تشکیل شده است که به ازای هر گره، یک متغیر و یک مقدار تقسیم‌بندی انتخاب می‌شود تا داده‌ها به گره‌های فرزند تقسیم

شوند. این فرآیند ادامه پیدا می‌کند تا ویژگی‌های مهم مجموعه داده درخت تصمیم را تشکیل دهند. هدف نهایی این است که با استفاده از این درخت، می‌توان پیش‌بینی‌هایی در مورد داده‌های جدید انجام داد. درخت تصمیم به دلیل قابل فهم بودن ساختار و نتایج آن، یکی از محبوب‌ترین روش‌های یادگیری ماشین است.

با استفاده از کتابخانه‌های موجود این الگوریتم را پیاده سازی اجرا و مطابق بخش ارزیابی، ارزیابی کنید. در صورت نیاز از Prune کردن درخت استفاده کنید. سعی کنید فراپارامترهای[7] درخت را بهینه کنید و در نهایت درخت بدست آمده رسم کنید. (می‌توانید از کتابخانه Plot_tree استفاده کنید.)

- **Random Forest**

روش‌های Ensemble در یادگیری ماشین به مجموعه‌ای از مدل‌ها اشاره دارند که به صورت همکاری برای بهبود دقت پیش‌بینی‌ها کار می‌کنند. این روش‌ها معمولاً با ترکیب چندین مدل ساده‌تر، مدل نهایی را می‌سازند که در مجموع از هر یک از مدل‌های تکی بهتر عمل می‌کند. دو روش اصلی در متدهای Ensemble وجود دارد: Bagging و Boosting به منظور کاهش واریانس مدل‌ها استفاده می‌شود و در آن چندین نمونه از داده‌ها به طور تصادفی انتخاب شده و برای هر نمونه یک مدل ساخته می‌شود. این مدل‌ها سپس ترکیب می‌شوند تا نتیجه نهایی حاصل شود.

جنگل تصادفی یکی دیگر از روش های یادگیری جمعی است که بر اساس ایده ای از تجمع از قوانین یا الگوریتم های ساده تر، به صورت تصادفی، تعدادی از مدل های یادگیری خود را اجرا می کند و سپس از ترکیب نتایج حاصل از این مدل ها برای پیش بینی مقادیر جدید استفاده می کند. در واقع، جنگل تصادفی یک مجموعه از درخت های تصمیم است که هر کدام به صورت مستقل از دیگری آموزش داده می شوند و سپس نتایج آن ها ترکیب می شوند تا یک پیش بینی نهایی برای داده های ورودی انجام شود. این روش برای حل مسائل پیچیده و تعداد زیادی داده بسیار موثر و کارآمد است.

با استفاده از کتابخانه‌های موجود این الگوریتم را پیاده سازی اجرا و مطابق بخش ارزیابی، ارزیابی کنید. برای بهینه کردن فراپارامترهای درخت از RandomizedSearchCV استفاده کنید.

- **Adaptive Boosting**

AdaBoost یک الگوریتم یادگیری ماشین است که به عنوان یک روش Boosting شناخته می‌شود. این الگوریتم به طور خاص برای بهبود دقت مدل‌های پیش‌بینی طراحی شده است و معمولاً از ترکیب چندین مدل ضعیف برای ایجاد یک مدل قوی‌تر استفاده می‌کند. در هر تکرار، AdaBoost یک مدل ضعیف را آموزش می‌دهد و وزن نمونه‌های اشتباه شناسایی شده را افزایش می‌دهد. این فرآیند ادامه می‌یابد تا مدل‌های بیشتری ایجاد شود که در نهایت ترکیب می‌شوند تا پیش‌بینی نهایی را ارائه

[7] HyperParameters

دهند.

با استفاده از کتابخانه‌های موجود این الگوریتم را پیاده سازی اجرا و مطابق بخش ارزیابی، ارزیابی کنید. بهترین فراپارامترهای را با آزمون و خطا مشخص کنید و سپس با تغییر مقدار فراپارامتر n_estimator تاثیر این فراپارامتر را نشان دهید. (دست کم 3 مقدار مختلف برای این فراپارامتر تنظیم کرده و نتایج را نگه‌دارید.)

- **XGBoost**

XGBoost یک الگوریتم یادگیری ماشین است که بر پایه روش های گرادیان کاهشی است. این الگوریتم برای حل مسائل مختلف یادگیری ماشین از جمله طبقه بندی، پیش بینی و رتبه بندی مورد استفاده قرار می گیرد. XGBoost قابلیت اجرای سریع، کارآیی بالا و افزایش دقت در پیش بینی ها را دارا می باشد.

با استفاده از کتابخانه‌های موجود این الگوریتم را پیاده سازی اجرا و مطابق بخش ارزیابی، ارزیابی کنید. برای بهینه کردن فراپارامترهای درخت از GridSearchCV استفاده کنید. برای سادگی بیشتر تنها به بهینه کردن فراپارامترهای زیر بپردازید:

learning_rate: نرخ تغییر وزن‌ها در هر گام.

n_estimators: تعداد مدل‌های پایه.

min_samples_split: حداقل تعداد نمونه‌ها برای تقسیم یک گره.

min_samples_leaf: حداقل تعداد نمونه‌ها برای برگ‌ها.

max_depth: حداکثر عمق درخت‌های تصمیم.

max_features: حداکثر تعداد ویژگی‌ها برای هر تقسیم گره.

4. Boosting from Scratch

در این بخش، شما وظیفه دارید Boosting from Scratch را با استفاده از الگوریتم SAMME پیاده‌سازی کنید. برای این کار، کافی است بخش‌های مشخص شده از کلاسی که در اختیارتان قرار داده شده است را تکمیل کنید. هدف از این تمرین این است که با مفاهیم پایه‌ای Boosting آشنا شوید و فرآیند تقویت مدل‌ها را به‌صورت عملی یاد بگیرید. با تکمیل این بخش، درک عمیق‌تری از عملکرد الگوریتم‌های تقویتی خواهید داشت.

حال با الگوریتم SAMME بیشتر آشنا می‌شویم.این الگوریتم نیز مانند دیگر الگوریتم های boosting، قصد دارد تا تعدادی طبقه‌بند ضعیف را با یکدیگر ترکیب کرده و تبدیل به یک مدل قوی شود.

ابتدا به صورت یکنواخت، به همه نمونه های آموزشی وزن می‌دهیم. بدین صورت که اگر n سمپل داشته باشیم، وزن هر سمپل به صورت $\frac{1}{n}$ خواهد بود.از این وزن ها در آموزش مدل، هنگامی که متد fit آن را فراخوانی می‌کنیم، استفاده می‌شود.بدین صورت که مدل بر اساس اهمیتی که به هر نمونه داده شده آموزش داده می‌شود.

حال مدل را طبق وزن‌های یکنواخت اولیه روی نمونه‌ها آموزش می‌دهیم. سپس می‌بینیم که مدل بعد از آموزش دیدن، کدام نمونه‌ها را اشتباه طبقه‌بندی می‌کند و طبق رابطه‌ای که در الگوریتم آورده شده، وزن آن نمونه را بروزرسانی می‌کنیم. در این مرحله وزن این نمونه باید زیاد شود. حال که وزن این نمونه‌ها زیاد شد، طبقه‌بند بعدی را با این اوزان جدید آموزش می‌دهیم و این پروسه را تا زمانی که تمامی طبقه‌بندها آموزش ببینند، تکرار می‌کنیم.

افزون بر موارد ذکر شده در پاراگراف فوق، هنگام آموزش مدل به طبقه‌بندها، بر حسب عملکرد آن‌ها امتیازدهی می‌کنیم. برای این کار، می‌بایست یک متریک به نام error محاسبه کنیم که در واقع نسبت مجموع اوزان نمونه‌های اشتباه طبقه‌بندی شده، بر جمع وزن کل نمونه‌هاست(بخش compute error در الگوریتم). بر اساس این error، یک امتیاز مطابق قسمت compute learner weight به طبقه‌بند تخصیص می‌دهیم و ذخیره می‌کنیم. توجه کنید که اگر error این طبقه‌بند از حدی بیشتر باشد، آن طبقه‌بند را دور می‌اندازیم. مثلا اگر مسئله طبقه بندی دو کلاسه است، مدلی که خطای آن 70% است برای ما ارزشی ندارد، چرا که امید ریاضی خطای یک طبقه بند که حتی آموزش ندیده، 50% است.

پس از اینکه آموزش مدل به پایان رسید، فرض کنید می‌خواهیم کلاس یک نمونه جدید را پیش‌بینی کنیم. در اینجا باید عملیاتی به نام weighted vote انجام شود. بدین صورت که به ازای هر کلاس، باید یک عدد نگه داریم و آن عدد، جمع امتیاز مدل هایی است که پیش بینی آنها این کلاس است. برای مثال، فرض کنید جدول زیر، وزن و پیش بینی هر یک از مدل‌ها را بدهد:

learner weight	learner prediction
0.5	1
0.2	2
0.7	1

2	0.4
3	0.8

در اینصورت، جمع وزن ها برای کلاس 1، به صورت $0.5 + 0.7 = 1.2$ است و برای کلاس 2 به صورت $0.2 + 0.4 = 0.6$ است و برای کلاس 3، 0.8 است.با توجه به اینکه weighted vote مربوط به کلاس 1 بیشتر از بقیه است، پیش بینی این مدل برای کوئری، کلاس 1 خواهد بود.

الگوریتم آن را در تصویر زیر مشاهده می‌کنید:

Algorithm 1 SAMME algorithm

Initialize the observation weights uniformly

for $m = 1 \to M$ **do**

Fit classifier $T^m(x)$ to the training data using weights w_i

compute error : $err^m = \frac{\sum_{i=1}^{n} w_i I(c_i \neq T^m(x_i))}{\sum_{i=1}^{n} w_i}$

compute learner weight : $\alpha^m = \log(\frac{1-err^m}{err^m}) + \log(K-1)$ in which K is number of classes

update weights: $w_i \leftarrow w_i . \exp(\alpha^m . I(c_i \neq T^m(x_i)))$ and renormalize w afterwards.

end for

Output : $C(x) = \underset{k}{argmax} \sum_{m=1}^{M} \alpha^m . I(T^m(x) = k)$

5. Comparison with Library Implementation:

در این بخش عملکرد الگوریتم Boosting from Scratch با نسخه آماده آن در کتابخانه Scikit-learn مقایسه می‌کنید. این مقایسه به شما کمک می‌کند تا درک بهتری از مفاهیم و جزئیات الگوریتم‌های Boosting پیدا کرده همچنین تفاوت‌ها و شباهت‌های بین پیاده‌سازی دستی و نسخه کتابخانه‌ای را بهتر درک کنید.

ارزیابی مدل‌ها

معیارهای زیادی برای سنجش و ارزیابی عملکرد مدلها وجود دارد. ارزیابی مدل های دسته بندی در یادگیری ماشینی به معنای ارزیابی عملکرد و کارایی مدل های مختلف است که برای دسته بندی داده ها استفاده می شوند. ارزیابی مدل های دسته بندی از اهمیت بسیاری برخوردار است زیرا به ما کمک می کند تا بتوانیم مدلی که می‌سازیم را با دقت بیشتری پیشرفت دهیم و اطمینان حاصل کنیم که عملکرد آن بهینه است.

با استفاده از این معیارها و ارزیابی کننده های دیگر می توان مدل های دسته بندی را مقایسه کرده و انتخاب بهترین مدل را برای مسئله خاص خود انجام داد.

برای ارزیابی مناسب از معیارهای زیر استفاده نمائید:

- ماتریس درهم‌ریختگی[8]
- Recall
- F1-Score
- Precision
- Accuracy
- میانگین‌گیری Macro و Micro و Weighted

مطالعه این دو لینک (لینک۱ و لینک۲) برای درک معیارهای فوق به شما کمک خواهد کرد.

[8] Confusion matrix

نکات پایانی

- دقت کنید که کد شما باید به نحوی زده شده باشد که نتایج قابلیت بازتولید داشته باشند.
- توضیحات مربوط به هر بخش از پروژه را بطور خلاصه و در عین حال مفید در گزارش خود ذکر کنید. از ابزارهای تحلیل داده مانند نمودارها استفاده کنید. حجم توضیحات گزارش شما هیچ گونه تاثیری در نمره نخواهد داشت و تحلیل و نمودارهای شما بیشترین ارزش را دارد.
- درباره هر بخش از مراحل پروژه می‌بایست علل استفاده یا عدم استفاده از هر الگوریتم، مزایا و معایب، عملکرد، فراپارامترها و وضعیت خروجی‌ها را بطور دقیق مطالعه کنید. از این موضوعات در زمان تحویل پرسیده خواهد شد.
- سعی کنید از پاسخ‌های روشن در گزارش خود استفاده کنید و اگر پیش‌فرضی در حل سوال در ذهن خود دارید، حتما در گزارش خود آن را ذکر نمایید.
- پس از مطالعه کامل و دقیق صورت پروژه، در صورت وجود هرگونه ابهام یا سوال با طراحان پروژه در ارتباط باشید.
- نتایج، گزارش و کدهای خود را در قالب یک فایل فشرده با فرمت AI_CA3_[stdNumber].zip در سامانه ایلرن بارگذاری کنید. به طور مثال AI_CA3_810101999.zip
- محتویات پوشه باید شامل فایل پاسخ‌های شما به سوالات کتبی، فایل jupyter-notebook، خروجی html و فایل‌های مورد نیاز برای اجرای آن باشد. از نمایش درست خروجی‌های مورد نیاز در فایل html مطمئن شوید.
- توجه کنید این تمرین باید به صورت تک‌نفره انجام شود و پاسخ‌های ارائه شده باید نتیجه فعالیت فرد نویسنده باشد. در صورت مشاهده تقلب به همه افراد مشارکت‌کننده، نمره تمرین 100- و به استاد نیز گزارش می‌گردد. همچنین نوشته نشدن کدها توسط هوش مصنوعی نیز بررسی می‌شود!

موفق باشید