دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)
دانشکده علوم کامپیوتر

تمرین پنجم

روش‌های پیشرفته در طبقه‌بندی

نگارش
ثمین مهدی پور
۹۸۳۹۰۳۹

استاد راهنما
دکتر قطعی

استادمشاور
دکتر یوسفی مهر

آذر ۱۴۰۲

چکیده

در این پروژه، ابتدا یک مجموعه داده شامل اطلاعات حسگرهای گاز در محیط تهیه شد. سپس، با استفاده از چندین الگوریتم مختلف مانند Random Forest، SVM و Naive Bayes، به دسته‌بندی داده‌ها پرداختیم. ارزیابی دقیقی بر روی هر دسته‌بندی‌کننده انجام شد که شامل مقایسه نتایج و تحلیل حساسیت به هایپرپارامترها بود.

سپس، به سمت دسته‌بندی به صورت مولتی‌مدل گام گرفتیم. از مفهوم Stacking استفاده شد که مجموعه‌ای از دسته‌بندی‌های پایه از جمله Random Forest، SVM و Naive Bayes را ترکیب کرده و با استفاده از یک دسته‌بندی‌کننده نهایی، کارایی بیشتری حاصل شد.

نتایج از نظر دقت و دیگر معیارهای ارزیابی برای هر دسته‌بندی‌کننده و همچنین مولتی‌مدل مورد بررسی قرار گرفت. علاوه بر این، حساسیت هایپرپارامترها نیز برای Random Forest، SVM و مولتی‌مدل Stacking به دقت مورد بررسی قرار گرفت. این تحلیل‌ها به محقق اطلاعات مفیدی ارائه می‌دهند تا بتواند بهبودهای لازم را در مدل‌ها اعمال کند.

واژه‌های کلیدی:

طبقه بندی، حسگرهای گاز، Random Forest, SVM, Multimodels

فهرست مطالب

صفحه

فهرست اشکال

صفحه

فصل اول

مقدمه

مقدمه

۱-۱ مجموعه داده

دیتاست Multimodal GasData دیتاستی جهت تشخیص انواع مختلف گازها و طبقه بندی آنها است. در این دیتاست که شامل ۸ ویژگی است برای هر نمونه به شکل همزمان ۷ ویژگی عددی با استفاده از هفت حسگر گاز مختلف و یک تصویر به کمک دوربین حرارتی ثبت شده است برای جمع آوری دیتاست دو گاز مختلف در نظر گرفته شده است تا چهار کلاس مختلف ایجاد شود این دیتاست شامل کلاسهای Smoke Perfume No Gas و Mixture of Perfume است که از هر کدام ۱۶۰۰ نمونـه در اختیـار داریم

در مجموع ۶۴۰۰ نمونه دیتاست را تشکیل میدهد.

۱-۲ اعمال الگوریتم های طبقه بندی

۱-۲-۱- Random Forest

در این روش، ما از مدل تصمیم‌گیری انبوهی با نام "Random Forest" استفاده کردیم. این مدل با ایجاد یک مجموعه از درخت‌های تصمیم، اطلاعات را تجمیع کرده و به دسته‌بندی نهایی می‌پردازد. این الگوریتم به دلیل قابلیت تعمیم بالا و مقاومت در برابر برخی از مشکلات از جمله برازش بیش‌اندازه (overfitting)، انتخاب متغیرهای مهم و سرعت بالا، گزینه مطلوبی برای طبقه‌بندی در داده‌های پیچیده به حساب می‌آید.

۱-۲-۲- Support Vector Machine (SVM)

الگوریتم SVM یک مدل قوی برای دسته‌بندی است که بر اساس ایجاد یک هایپرصفحه جداکننده بین دسته‌ها عمل می‌کند. در این تحقیق، ما از SVM با هسته‌های مختلف مانند خطی، RBF و polynomial بهره‌مند شدیم. این الگوریتم به خصوص برای مواردی که داده‌ها در یک فضای بلند بعدی قرار دارند، کارآمد است.

۱-۲-۳- Naive Bayes

از الگوریتم Naive Bayes نیز در این تحقیق استفاده کردیم. مدل Gaussian Naive Bayes بر پایه فرض استقلال شرطی ویژگی‌ها به شرط دانستن کلاس، دسته‌بندی انجام می‌دهد. این الگوریتم به دلیل سادگی و کارایی در مواجهه با مجموعه‌داده‌های کوچک و با توزیع‌های گوناگون، انتخاب موردی مناسب برای این تحقیق بود.

۱-۳- طبقه بندی با چند مدل

۱-۳-۱- VotingClassifier

در این بخش، از روش Multi-Modal Classification یا همان Voting Classifier استفاده کردیم. این روش به ما این امکان را می‌دهد که نتایج حاصل از چندین دسته‌بند را ترکیب کرده و با تصمیم گیری اکثریت، به یک پیش‌بینی نهایی برسیم. از این روش به دلیل سادگی پیاده‌سازی و عدم نیاز به تنظیمات پیچیده بهره‌مند شدیم.

دلایل استفاده:

- تنوع مدل‌ها:

با استفاده از چندین مدل مختلف از جمله Random Forest، SVM و Naive Bayes، تنوع بالایی در پیش‌بینی‌ها ایجاد کردیم. این تنوع می‌تواند به بهبود کارایی نهایی کلاسیفیکیشن کمک کند.

- استفاده از تجربه‌های مدل‌های مختلف:

هر مدل دارای قابلیت‌ها و محدودیت‌های خود است. با ترکیب این مدل‌ها، می‌توان از تجربیات و قوانین یادگرفته شده توسط هر یک استفاده کرد و بهبود مسائلی که ممکن است در یک مدل خاص وجود داشته باشد، انجام داد.

۱-۳-۲- Stacking Classifier

در این بخش، از روش Stacking Classifier استفاده کردیم که یک مرحله پیشرفته‌تر از Multi-Modal Classification است. این روش از چندین دسته‌بند پایه (Base Classifier) به عنوان یک لایه پایه استفاده می‌کند و خروجی این دسته‌بندهای پایه را به یک دسته‌بند نهایی (Meta-Classifier) منتقل می‌کند.

دلایل استفاده:

استفاده از قدرت مدل‌های مختلف: با انتخاب مدل‌های پایه متنوع از جمله Random Forest، SVM و Naive Bayes، می‌توانیم از قدرت هر کدام به نحو بهینه استفاده کنیم و این اطلاعات را در لایه نهایی بهبود بخشیم.

کنترل بیشتر بر روی هایپرپارامترها: این روش اجازه می‌دهد تا هایپرپارامترهای مختلف برای مدل‌های پایه و همچنین مدل نهایی تنظیم شوند، که می‌تواند به بهبود کلی کارایی سیستم کمک کند.

فصل دوم

پیاده‌سازی

۲-۱- کاوش و بررسی مجموعه داده

پس از تبدیل به دیتاست آن را مورد بررسی قرار دادیم. در این محموعه داده، داده گم شـده یـا تکـراری یافت نشد و دارای ۱۰ ویژگی و ۶۴۰۰ سطر بود.

	Serial Number	MQ2	MQ3	MQ5	MQ6	MQ7	MQ8	MQ135	Gas	Corresponding Image Name
0	0	555	515	377	338	666	451	416	NoGas	0_NoGas
1	1	555	516	377	339	666	451	416	NoGas	1_NoGas
2	2	556	517	376	337	666	451	416	NoGas	2_NoGas
3	3	556	516	376	336	665	451	416	NoGas	3_NoGas
4	4	556	516	376	337	665	451	416	NoGas	4_NoGas
...	...	...	...	...	...	...	...	...	...	...
6395	1595	658	445	455	414	491	321	436	Mixture	1595_Mixture
6396	1596	650	444	451	411	486	317	431	Mixture	1596_Mixture
6397	1597	630	443	446	407	474	312	429	Mixture	1597_Mixture
6398	1598	632	443	444	405	471	309	430	Mixture	1598_Mixture
6399	1599	633	442	442	402	468	306	434	Mixture	1599_Mixture

برای تحلیل بهتر ستونهای Serial Number, Corresponding image name حذف شدند چـون در روند تحلیل کمک کننده نبودند. ستون گاز هم برای طبقه بندی بهتر از حالت متن به عدد مپ شد پس در نهایت داشتیم:

	MQ2	MQ3	MQ5	MQ6	MQ7	MQ8	MQ135	Gas
0	555	515	377	338	666	451	416	0
1	555	516	377	339	666	451	416	0
2	556	517	376	337	666	451	416	0
3	556	516	376	336	665	451	416	0
4	556	516	376	337	665	451	416	0
...	...	...	...	...	...	...	...	...

۲-۲-پیش پردازش

در مرحله پیش‌پردازش داده، ابتدا داده‌ها را به دو بخش ورودی (X) و خروجی (y) تقسیم کردیم. ویژگی‌های ورودی (X) شامل تمامی ستون‌های داده با استثناء ستون مربوط به کلاس‌های گاز بود. ستون مربوط به کلاس‌ها به عنوان خروجی (y) تعیین شد.

سپس، داده‌ها به دو مجموعه آموزش و آزمون تقسیم شدند. ما ۸۰ درصد از داده‌ها را برای آموزش مدل‌ها (X_train و y_train) و ۲۰ درصد باقی‌مانده را برای ارزیابی مدل‌ها (X_test و y_test) در نظر گرفتیم.

سپس از یک ابزار مهم در پیش‌پردازش به نام `StandardScaler` استفاده کردیم. با استفاده از این ابزار، داده‌ها به گونه‌ای مقیاس‌دار (scaled) شدند که میانگین هر ویژگی صفر و انحراف معیار یک شود. این مرحله به ما کمک می‌کند تا تاثیرات مقیاس متفاوت ویژگی‌ها را از بین ببریم و مدل‌ها به‌طور بهینه‌تری آموزش ببینند.

در نهایت، داده‌های آموزش و آزمون پس از پیش‌پردازش حاصل شده و آماده برای استفاده در مدل‌های یادگیری ماشین شدند.

۲-۳- طبقه بندی

۲-۳-۱- Random Forest

در این بخش از کد، ما از الگوریتم Random Forest برای طبقه‌بندی داده‌ها استفاده کردیم. این الگوریتم یک مدل یادگیری ماشین انبوهی است که از چندین درخت تصمیم به نام "تصمیم گیر انبوهی" تشکیل شده است. در ادامه، توضیحاتی در مورد اقدامات انجام‌شده آمده است:

۱. تعریف پارامترها:

ما یک مجموعه از پارامترهای مهم برای Random Forest را تعریف کردیم که شامل تعداد درخت‌ها (n_estimators) و حداکثر عمق هر درخت (max_depth) می‌شود. این پارامترها به ما این امکان را می‌دهند که مدل را بهینه تنظیم کرده و از برازش بیش‌اندازه (overfitting) جلوگیری کنیم.

۲. ایجاد یک شی Random Forest:

با استفاده از `RandomForestClassifier` از کتابخانه scikit-learn، یک شیء از الگوریتم Random Forest ایجاد کردیم. این شیء برای بررسی تاثیر پارامترها و انجام تنظیمات لازم بر روی مدل استفاده خواهد شد.

۳. Grid Search برای تنظیم بهترین پارامترها:

از `GridSearchCV` برای جستجو در فضای پارامترها و انتخاب بهترین مقادیر استفاده کردیم. این ابزار انجام جستجو در فضای پارامترها را با کمک یک مدل انتخابی انجام می‌دهد و بهترین ترکیب پارامترها را با استفاده از معیارهای ارزیابی (مانند اعتبارسنجی متقاطع) انتخاب می‌کند.

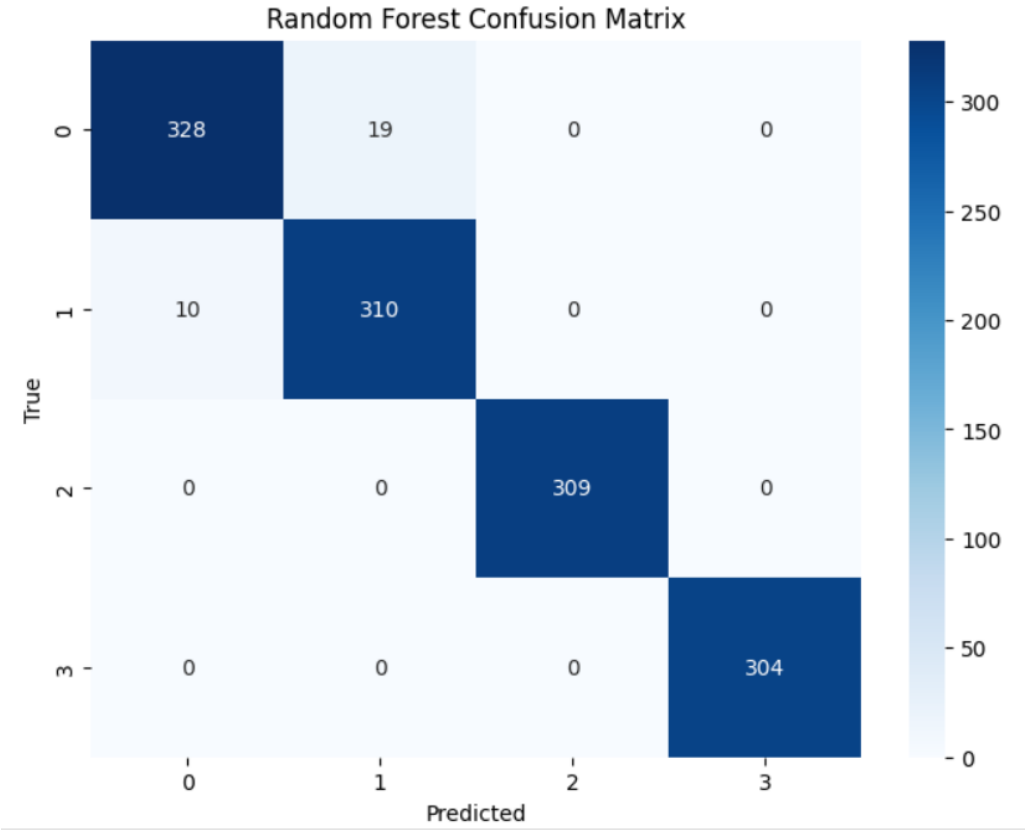
۴. آموزش مدل و ارزیابی:

ما مدل Random Forest را با استفاده از داده‌های آموزشی (`X_train_scaled` و `y_train`) آموزش دادیم و سپس بر روی داده‌های آزمون (`X_test_scaled`) ارزیابی کردیم. این ارزیابی شامل گزارش طبقه‌بندی (classification report) و دقت (accuracy) مدل بر روی داده‌های آزمون است.

۵. تحلیل حساسیت به هایپرپارامترها:

با استفاده از `permutation_importance`، حساسیت مدل به ویژگی‌ها و پارامترها را تحلیل کردیم. این تحلیل به ما اطلاعاتی در مورد اهمیت ویژگی‌ها در تصمیم‌گیری مدل ارائه می‌دهد.

```
Random Forest Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.95      0.96       347
           1       0.94      0.97      0.96       320
           2       1.00      1.00      1.00       309
           3       1.00      1.00      1.00       304

    accuracy                           0.98      1280
   macro avg       0.98      0.98      0.98      1280
weighted avg       0.98      0.98      0.98      1280

Random Forest Accuracy: 0.97734375
Random Forest Feature Importances:
[0.040625   0.271875   0.01703125 0.074375   0.05546875 0.1478125
 0.0696875 ]
```



در ادامه، ما حساسیت مدل Random Forest به ویژگی‌ها و پارامترها را تحلیل کرده و سپس این حساسیت را تصویرسازی کرده‌ایم. زیرا این بخش از کد نیز به صورت تحلیلی و تصویری اطلاعات در مورد اهمیت ویژگی‌ها در مدل Random Forest ارائه می‌دهد.

۱. تحلیل حساسیت به ویژگی‌ها:

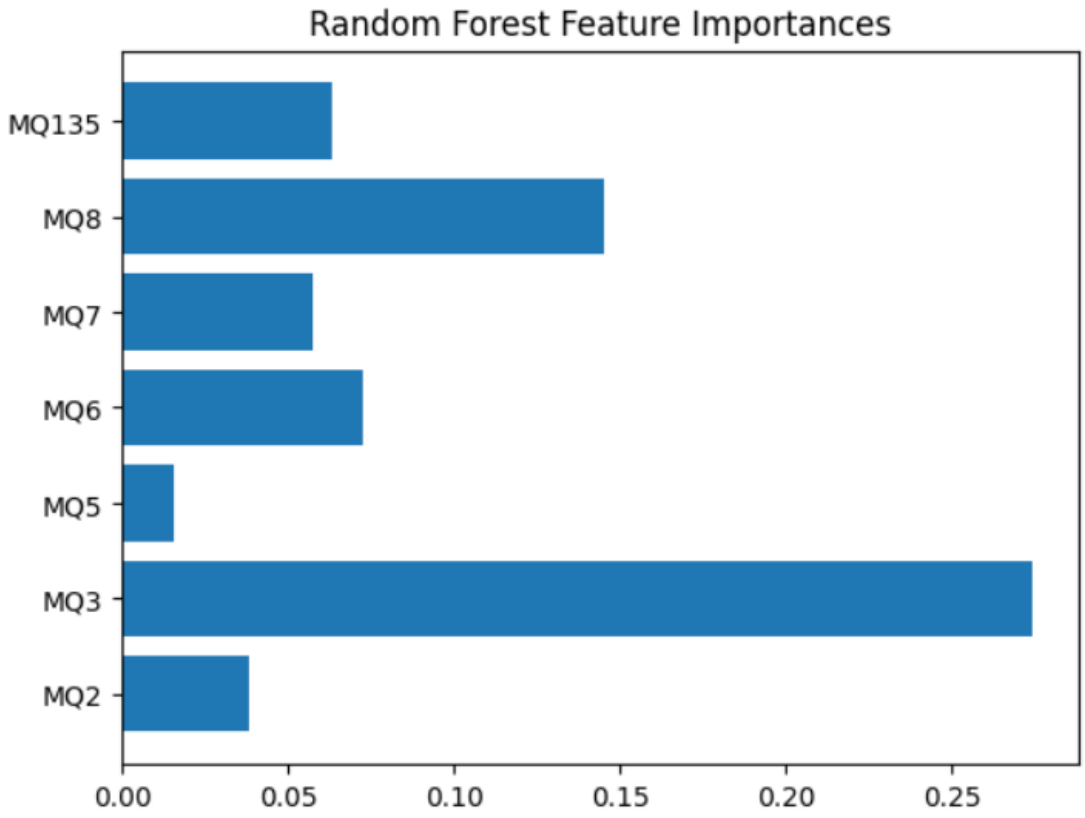
از `permutation_importance` برای تحلیل حساسیت مدل به ویژگی‌ها استفاده کردیم. این ابزار اهمیت هر ویژگی را با تغییر مقادیر آن و انجام پرمیوتیشن بر روی داده‌های آزمون، ارزیابی می‌کند. نتایج این تحلیل حاوی اطلاعاتی در مورد اهمیت نسبی هر ویژگی در تصمیم‌گیری مدل است.

۲. تصویرسازی حساسیت به ویژگی‌ها:

سپس از یک نمودار ملودی برای نمایش اهمیت متوسط ویژگی‌ها استفاده کردیم. این نمودار ملودی با استفاده از `plt.barh` ایجاد شده است و اهمیت متوسط هر ویژگی در تصمیم‌گیری مدل را به صورت واضح و زیبا نمایش می‌دهد.

در کل، این بخش از کد به ما این امکان را می‌دهد که ببینیم کدام ویژگی‌ها برای مدل Random Forest در تصمیم‌گیری مهم‌تر بوده‌اند و به ما در فهم بهتر اطلاعاتی از مدل ارائه می‌دهد.

```
Random Forest Feature Importances:
[0.0384375  0.27453125 0.015625   0.0725     0.05734375 0.1453125
 0.063125  ]
```



۲-۳-۲- SVM

در این بخش از کد، ما از الگوریتم Support Vector Machine (SVM) برای طبقه‌بندی داده‌ها استفاده کردیم و بهینه‌سازی پارامترها با استفاده از Grid Search انجام دادیم. زیرا SVM یک الگوریتم مهم در یادگیری ماشین است که برای مسائل طبقه‌بندی و رگرسیون کارآیی بالایی دارد. در ادامه، توضیحاتی در مورد اقدامات انجام‌شده آمده است:

۱. تعریف پارامترها:

ما یک مجموعه از پارامترهای مهم برای SVM تعریف کردیم که شامل پارامتر میان‌گین خطاها (C) و نوع هسته (kernel) می‌شود. این پارامترها به ما این امکان را می‌دهند که مدل را بهینه تنظیم کرده و از برازش بیش‌اندازه جلوگیری کنیم.

۲. ایجاد یک شی SVM:

با استفاده از `SVC` از کتابخانه scikit-learn، یک شیء از الگوریتم SVM ایجاد کردیم. این شیء برای بررسی تاثیر پارامترها و انجام تنظیمات لازم بر روی مدل استفاده خواهد شد.

۳. Grid Search برای تنظیم بهترین پارامترها:

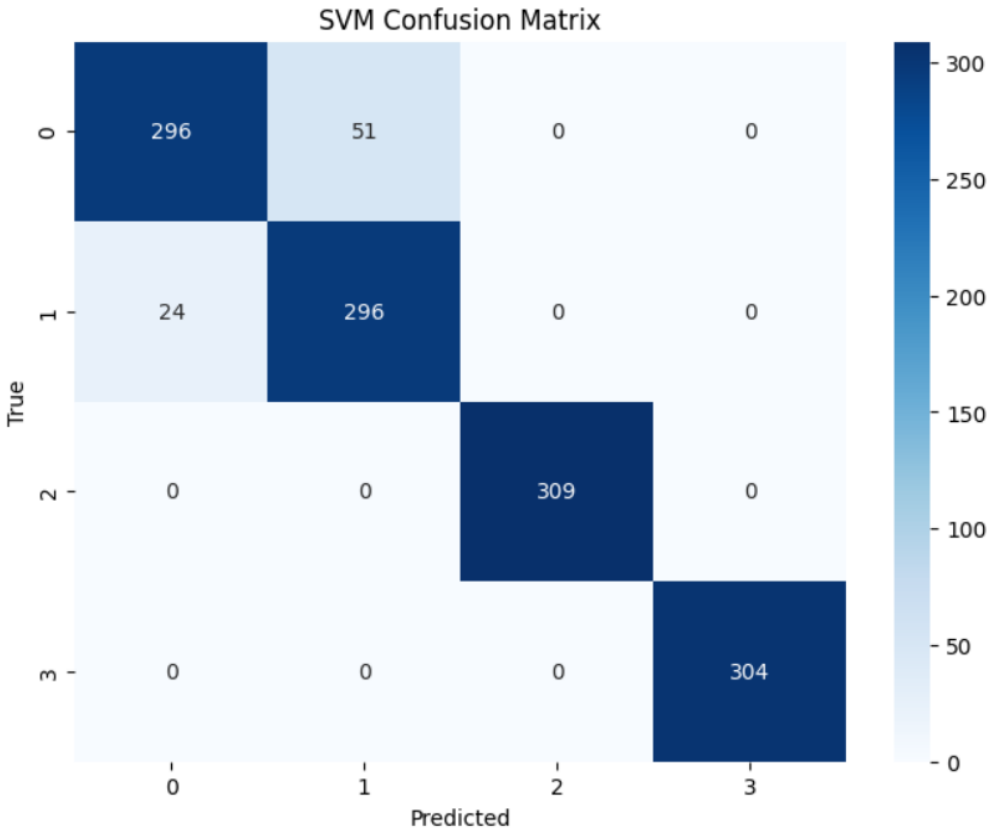
از `GridSearchCV` برای جستجو در فضای پارامترها و انتخاب بهترین مقادیر استفاده کردیم. این ابزار انجام جستجو در فضای پارامترها را با کمک یک مدل انتخابی انجام می‌دهد و بهترین ترکیب پارامترها را با استفاده از معیارهای ارزیابی انتخاب می‌کند.

۴. آموزش مدل و ارزیابی:

ما مدل SVM را با استفاده از داده‌های آموزشی (`X_train_scaled` و `y_train`) آموزش دادیم و سپس بر روی داده‌های آزمون (`X_test_scaled`) ارزیابی کردیم. این ارزیابی شامل گزارش طبقه‌بندی (classification report) و دقت (accuracy) مدل بر روی داده‌های آزمون است.

در نهایت، اطلاعات ارزیابی برای مدل SVM به صورت گزارش طبقه‌بندی و دقت چاپ شده‌اند. این اطلاعات به ما کمک می‌کنند تا درک بهتری از عملکرد مدل SVM در طبقه‌بندی داده‌ها پیدا کنیم.

```
SVM Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.85      0.89       347
           1       0.85      0.93      0.89       320
           2       1.00      1.00      1.00       309
           3       1.00      1.00      1.00       304

    accuracy                           0.94      1280
   macro avg       0.94      0.94      0.94      1280
weighted avg       0.94      0.94      0.94      1280

SVM Accuracy: 0.94140625
```



در ادامه، ما حساسیت مدل SVM به یکی از پارامترهای مهم آن یعنی C را بررسی کرده‌ایم. چرا که C یک پارامتر مهم در SVM است که نشان‌دهنده میزان مجازی از عدم قطعیت در تصمیم‌گیری مدل است. این کد به ما امکان می‌دهد تا تاثیر مقادیر مختلف پارامتر C بر دقت مدل SVM را بررسی کنیم.

۱. تعریف مقادیر مختلف برای C:

ما چندین مقدار مختلف برای پارامتر C انتخاب کردیم. در اینجا، مقادیر `[۰.۱, ۱, ۱۰]` برای C انتخاب شده‌اند.

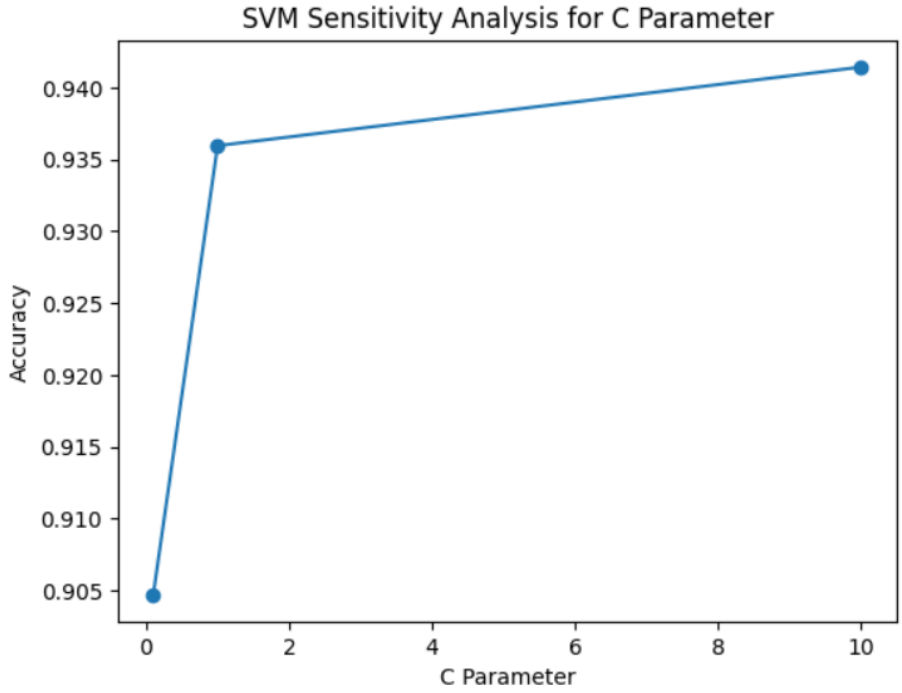
۲. بررسی حساسیت:

از یک حلقه `for` برای آموزش مدل SVM با هر یک از مقادیر C و ارزیابی دقت مدل بر روی داده‌های آزمون استفاده کردیم. دقت هر مدل برای هر مقدار C ذخیره شده و در یک لیست به نام `svm_sensitivity_scores` اضافه شده است.

۳. تصویرسازی حساسیت به C:

با استفاده از `plt.plot`، نمودار خطی از تغییرات دقت بر حسب مقادیر مختلف C ایجاد شده است. این نمودار به ما نشان می‌دهد که با افزایش یا کاهش مقدار C، چگونه دقت مدل SVM تغییر می‌کند.

در نتیجه، این بخش از کد به ما اطلاعاتی از تحلیل حساسیت مدل SVM به پارامتر C ارائه می‌دهد و می‌تواند بهترین مقدار برای این پارامتر را مشخص کند.



۲-۳-۳- Gaussian Naive Bayes

در این بخش از کد، از الگوریتم Naive Bayes برای طبقه‌بندی داده‌ها استفاده کردیم. برای Naive Bayes، معمولاً نیازی به تنظیم هایپرپارامترها نداریم. در اینجا، از یک نوع خاص از Naive Bayes یعنی `GaussianNB` استفاده شده است که برای متغیرهای پیوسته مناسب است.

۱. ایجاد شی Naive Bayes:

ما از `GaussianNB` که یک نمونه از الگوریتم Naive Bayes برای داده‌های پیوسته است، یک شیء از طبقه‌بند Naive Bayes ایجاد کردیم.

۲. آموزش مدل:

مدل Naive Bayes را با استفاده از داده‌های آموزش (`X_train_scaled` و `y_train`) آموزش دادیم.

۳. ارزیابی مدل:

ما مدل را بر روی داده‌های آزمون (`X_test_scaled`) ارزیابی کردیم. این ارزیابی شامل گزارش طبقه‌بندی (classification report) و دقت (accuracy) مدل بر روی داده‌های آزمون است.

نتیجه این بخش از کد، اطلاعاتی در مورد عملکرد مدل Naive Bayes بر روی داده‌های آزمون و دقت طبقه‌بندی ارائه می‌دهد. از آنجا که Naive Bayes یک الگوریتم ساده است و بر اساس اصل نیایی (Naive) عمل می‌کند، معمولاً به سرعت و با دقت مناسب برای داده‌های مختلف کارآمد است.

```
Naive Bayes Classification Report:
              precision    recall  f1-score   support

           0       0.68      0.80      0.74       347
           1       0.73      0.60      0.66       320
           2       1.00      1.00      1.00       309
           3       1.00      1.00      1.00       304

    accuracy                           0.84      1280
   macro avg       0.85      0.85      0.85      1280
weighted avg       0.85      0.84      0.84      1280

Naive Bayes Accuracy: 0.84453125
```

در ادامه ما مدل Naive Bayes را با استفاده از متغیرهای ورودی مختلف و گزینه‌های پیش‌پردازش مختلف با استفاده از اعتبارسنجی متقاطع (Cross-Validation) ارزیابی کرده‌ایم. این بخش از کد به ما اطلاعاتی از کارایی مدل Naive Bayes با تغییرات در ویژگی‌ها و گزینه‌های پیش‌پردازش می‌دهد.

۱. انتخاب گزینه‌های پیش‌پردازش و ویژگی‌ها:

برای هر گزینه پیش‌پردازش (preprocessing_options) و هر مجموعه ویژگی (feature_sets)، ابتدا گزینه‌های پیش‌پردازش اعمال می‌شوند (اگر گزینه مشخص شده باشد) و سپس مدل Naive Bayes ایجاد می‌شود.

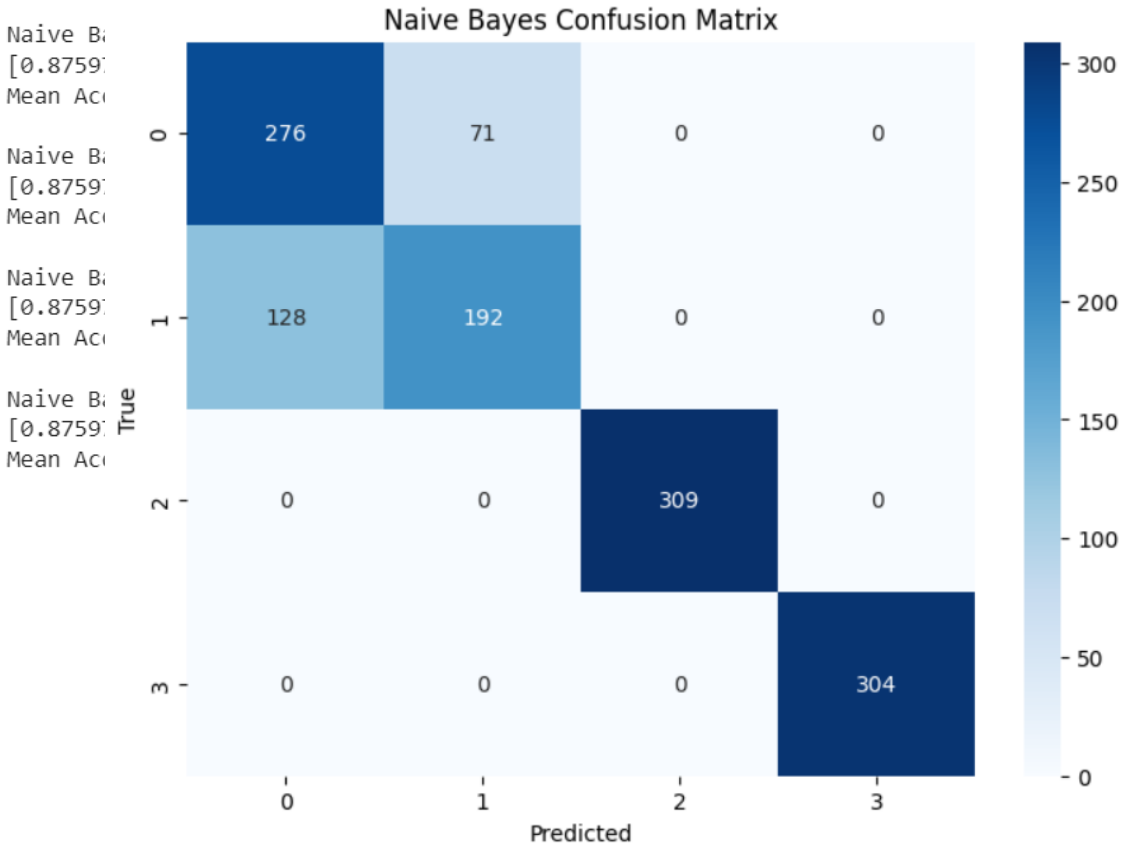
۲. اعتبارسنجی متقاطع:

برای هر ترکیب از گزینه‌های پیش‌پردازش و ویژگی‌ها، از `cross_val_score` برای انجام اعتبارسنجی متقاطع با ۵ تا اجرا استفاده کردیم.

۳. نمایش نتایج:

نتایج اعتبارسنجی متقاطع برای هر ترکیب از گزینه‌ها نمایش داده شده‌اند. این نتایج شامل امتیازهای هر تکرار اعتبارسنجی و میانگین دقت این امتیازها است.

به این ترتیب، این بخش اطلاعاتی از کارایی مدل Naive Bayes با تغییرات در ویژگی‌ها و گزینه‌های پیش‌پردازش در شرایط مختلف را ارائه می‌دهد.



در ادامه با استفاده از مدل های طبقه بندی شده طبقه بندی را انجام میدهیم:

۲-٤-طبقه بندی با مدل های چند لایه ای

۲-۴-۱- Voting Classifier

در این بخش از کد، ما از مدل Voting Classifier برای ترکیب نتایج از دو مدل مختلف یعنی Random Forest و SVM به منظور بهبود کارایی طبقه‌بندی استفاده کردیم.

۱. ایجاد Voting Classifier:

با استفاده از `VotingClassifier` از scikit-learn، یک مدل ترکیبی از دو مدل Random Forest و SVM ایجاد شده است. در اینجا، از روش "hard" برای انتخاب بیشینه تعداد آرا در طبقه‌بندی‌ها استفاده شده است.

۲. آموزش Voting Classifier:

ما Voting Classifier را با استفاده از داده‌های آموزش (`X_train_scaled` و `y_train`) آموزش دادیم.

۳. ارزیابی Voting Classifier:

مدل Voting Classifier را بر روی داده‌های آزمون (`X_test_scaled`) ارزیابی کردیم. این ارزیابی شامل گزارش طبقه‌بندی (classification report) و دقت (accuracy) مدل بر روی داده‌های آزمون است.

نتیجه این بخش از کد به ما اطلاعاتی از کارایی مدل Voting Classifier با استفاده از دو مدل مختلف در طبقه‌بندی داده‌ها را ارائه می‌دهد. استفاده از Voting Classifier می‌تواند بهبود در کارایی مدل نسبت به استفاده از هر یک از مدل‌ها به تنهایی ایجاد کند.

```
Voting Classifier Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.95      0.94       347
           1       0.95      0.92      0.93       320
           2       1.00      1.00      1.00       309
           3       1.00      1.00      1.00       304

    accuracy                           0.97      1280
   macro avg       0.97      0.97      0.97      1280
weighted avg       0.97      0.97      0.97      1280

Voting Classifier Accuracy: 0.96640625
```

در ادامه، ما از تمام ترکیب‌های مختلف از مدل‌های پایه (Random Forest و SVM) برای ایجاد Voting Classifier استفاده کردیم. این امکان به ما می‌دهد تا تأثیر ترکیب‌های مختلف از مدل‌ها بر کارایی مدل Voting Classifier را بررسی کنیم.

۱. تعریف مدل‌های پایه:

مدل‌های پایه که در اینجا از Random Forest و SVM استفاده شده‌اند، به عنوان `base_classifiers` تعریف شده‌اند.

۲. استفاده از ترکیب‌های مختلف:

با استفاده از `combinations` از ماژول `itertools`، تمام ترکیب‌های مختلف از مدل‌های پایه با اندیس‌های ۱ تا تعداد کل مدل‌ها ایجاد شده‌اند.

۳. ایجاد Voting Classifier با هر ترکیب:

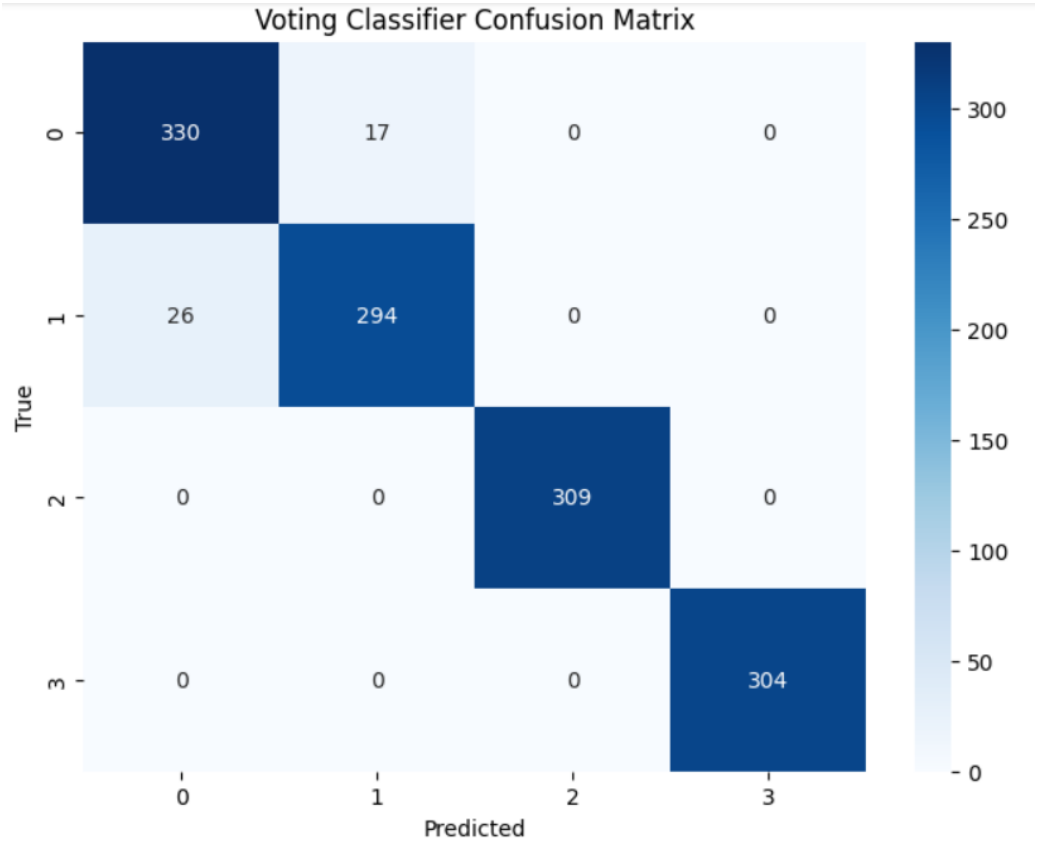
برای هر ترکیب مدل‌های پایه، یک Voting Classifier با این ترکیب ایجاد شده است و بر روی داده‌های آموزش آموزش داده شده است.

۴. ارزیابی Voting Classifier با هر ترکیب:

Voting Classifier مربوطه بر روی داده‌های آزمون ارزیابی شده و گزارش طبقه‌بندی و دقت آن چاپ شده است.

نتیجه این بخش از کد، اطلاعاتی از کارایی Voting Classifier با ترکیب‌های مختلف از مدل‌های پایه (Random Forest و SVM) را ارائه می‌دهد. این کار می‌تواند به شناخت بهتری از تأثیر ترکیب‌های مختلف مدل‌ها در کارایی نهایی کمک کند.

```
Voting Classifier Subset (rf) Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.95      0.96       347
           1       0.94      0.97      0.96       320
           2       1.00      1.00      1.00       309
           3       1.00      1.00      1.00       304

    accuracy                           0.98      1280
   macro avg       0.98      0.98      0.98      1280
weighted avg       0.98      0.98      0.98      1280

Subset Accuracy (rf): 0.97734375


Voting Classifier Subset (svm) Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.85      0.89       347
           1       0.85      0.93      0.89       320
           2       1.00      1.00      1.00       309
           3       1.00      1.00      1.00       304

    accuracy                           0.94      1280
   macro avg       0.94      0.94      0.94      1280
weighted avg       0.94      0.94      0.94      1280

Subset Accuracy (svm): 0.94140625


Voting Classifier Subset (rf, svm) Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.95      0.94       347
           1       0.95      0.92      0.93       320
           2       1.00      1.00      1.00       309
           3       1.00      1.00      1.00       304

    accuracy                           0.97      1280
   macro avg       0.97      0.97      0.97      1280
weighted avg       0.97      0.97      0.97      1280

Subset Accuracy (rf, svm): 0.96640625
```



Stacking Classifier -۱-۴-۲

در این بخش از کد، از مدل Stacking Classifier برای ترکیب نتایج از سه مدل مختلف یعنی Random Forest، SVM و Naive Bayes به منظور بهبود کارایی طبقه‌بندی استفاده کردیم.

۱. تعریف مدل‌های پایه:

مدل‌های پایه شامل Random Forest، SVM و Naive Bayes با نام‌های مختلف به عنوان `base_classifiers` تعریف شده‌اند.

۲. تعریف مدل فراطبقه‌بندی (Meta-Classifier):

یک مدل فراطبقه‌بندی نیز به عنوان `meta_classifier` تعریف شده است. در اینجا از یک Random Forest به عنوان مدل فراطبقه‌بندی استفاده شده است.

۳. ایجاد Stacking Classifier:

با استفاده از `StackingClassifier` از scikit-learn، یک مدل Stacking Classifier با مدل‌های پایه و مدل فراطبقه‌بندی ایجاد شده است. این مدل ترکیبی از مدل‌های پایه است که با استفاده از یک مدل فراطبقه‌بندی (Meta-Classifier)، نتایج این مدل‌های پایه را ترکیب می‌کند.

۴. آموزش Stacking Classifier:

ما Stacking Classifier را با استفاده از داده‌های آموزش (`X_train_scaled` و `y_train`) آموزش دادیم.

۵. ارزیابی Stacking Classifier:

مدل Stacking Classifier را بر روی داده‌های آزمون (`X_test_scaled`) ارزیابی کردیم. این ارزیابی شامل گزارش طبقه‌بندی (classification report) و دقت (accuracy) مدل بر روی داده‌های آزمون است.

نتیجه این بخش از کد، اطلاعاتی از کارایی مدل Stacking Classifier با استفاده از مدل‌های مختلف پایه (Random Forest، SVM و Naive Bayes) و مدل فراطبقه‌بندی را ارائه می‌دهد. Stacking Classifier می‌تواند بهبود در کارایی مدل‌ها در کنار یکدیگر ایجاد کند.

```
Stacking Classifier Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.96      0.96       347
           1       0.96      0.96      0.96       320
           2       1.00      1.00      1.00       309
           3       1.00      1.00      1.00       304

    accuracy                           0.98      1280
   macro avg       0.98      0.98      0.98      1280
weighted avg       0.98      0.98      0.98      1280

Stacking Classifier Accuracy: 0.9796875
```

در این بخش از کد، ما از تنظیم هایپرپارامتر برای مدل‌های پایه و فراطبقه‌بندی (Meta-Classifier) در مدل Stacking Classifier استفاده کرده‌ایم. این امکان به ما می‌دهد تا بهبود کارایی مدل Stacking Classifier با انتخاب بهترین تنظیمات هایپرپارامترها را بررسی کنیم.

۱. تعریف هایپرپارامترها برای مدل‌های پایه:

برای هر یک از مدل‌های پایه (Random Forest، SVM، Naive Bayes)، هایپرپارامترهای مربوط به آن تعریف شده‌اند. برای مدل‌های Random Forest و SVM از یک Grid Search برای تنظیم بهترین هایپرپارامترها استفاده شده است.

۲. ایجاد مدل‌های پایه با تنظیم هایپرپارامتر:

ما از Grid Search برای تنظیم هایپرپارامترهای مدل‌های Random Forest و SVM استفاده کردیم و بهترین مدل‌ها را از نتایج Grid Search برگرفتیم.

۳. تعریف هایپرپارامترها برای فراطبقه‌بندی (Meta-Classifier):

برای مدل فراطبقه‌بندی (Meta-Classifier) که یک Random Forest است، هایپرپارامترهای مربوط به آن نیز تعریف شده‌اند.

۴. ایجاد Stacking Classifier با تنظیم هایپرپارامتر:

ما یک مدل Stacking Classifier با استفاده از بهترین مدل‌های پایه و مدل فراطبقه‌بندی تعریف شده با تنظیم هایپرپارامترها ایجاد کرده‌ایم.

۵. آموزش Stacking Classifier با تنظیم هایپرپارامتر:

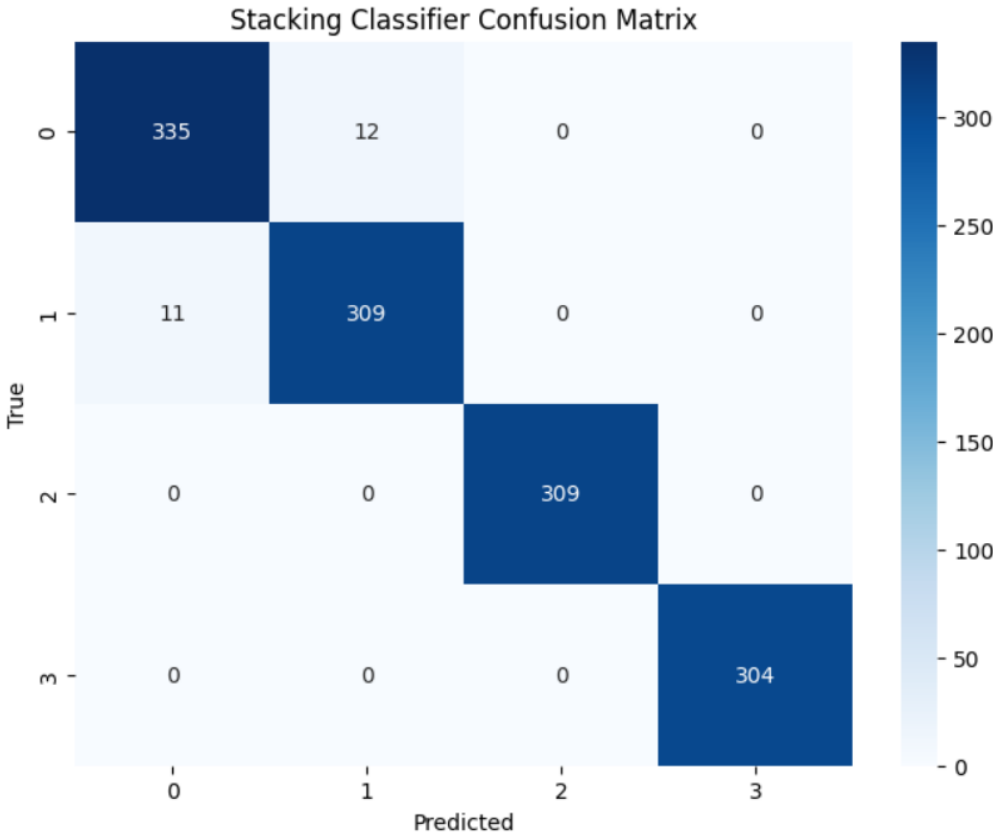
مدل Stacking Classifier را با استفاده از داده‌های آموزش (`X_train_scaled` و `y_train`) و با تنظیم هایپرپارامترها آموزش دادیم.

۶. ارزیابی Stacking Classifier با تنظیم هایپرپارامتر:

مدل Stacking Classifier را بر روی داده‌های آزمون (`X_test_scaled`) ارزیابی کردیم و گزارش طبقه‌بندی و دقت آن چاپ شده است.

نتیجه این بخش از کد، اطلاعاتی از کارایی مدل Stacking Classifier با استفاده از بهترین تنظیمات هایپرپارامترها برای مدل‌های پایه و مدل فراطبقه‌بندی را ارائه می‌دهد.

```
Stacking Classifier Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.97      0.97       347
           1       0.96      0.97      0.96       320
           2       1.00      1.00      1.00       309
           3       1.00      1.00      1.00       304

    accuracy                           0.98      1280
   macro avg       0.98      0.98      0.98      1280
weighted avg       0.98      0.98      0.98      1280

Stacking Classifier Accuracy: 0.98203125
```



فصل سوم

جمع‌بندی

جمع‌بندی

۳-۱ طبقه بندی

تجزیه و تحلیل گزارش‌های طبقه‌بندی بر اساس خروجی برای هر یک از روش‌های طبقه‌بندی انجام شده است:

۱. Random Forest:

- دقت (Accuracy): 97.73%

- گزارش طبقه‌بندی:

- دقت بسیار بالا برای تمام کلاس‌ها (۰.۹۶ تا ۱.۰۰).

- دقت و بازخوانی (precision و recall) به خوبی برای هر کلاس متوازن است.

- میانگین دقت (macro avg) و میانگین دقت و بازخوانی (weighted avg) همگی به خوبی نزدیک به ۱ هستند.

- ویژگی‌های مهم:

- میزان اهمیت ویژگی‌ها نیز نشان می‌دهد که ویژگی‌های مختلفی در انجام طبقه‌بندی مؤثر بوده‌اند.

۲. SVM:

- دقت (Accuracy): 94.14%

- گزارش طبقه‌بندی:

- دقت بالا برای کلاس‌های ۲ و ۳، اما کمی پایین‌تر برای کلاس‌های ۰ و ۱.

- بازخوانی نسبتاً متوازن برای تمام کلاس‌ها.

- میانگین دقت و بازخوانی به خوبی نزدیک به ۱ هستند.

- تحلیل:

- SVM به نسبت دقت بالا دارد اما برخی از کلاس‌ها ممکن است دقت پایین‌تری داشته باشند.

۳. Naive Bayes:

- دقت (Accuracy): 84.45%

- گزارش طبقه‌بندی:

- دقت متوسط برای کلاس‌ها با توجه به دقت پایین برای کلاس‌های ۰ و ۱.

- بازخوانی برای کلاس‌های ۰ و ۱ کمی پایین‌تر از دیگر کلاس‌ها است.

- میانگین دقت و بازخوانی هم به نسبت کمی پایین‌تر از دیگر دو روش است.

- تحلیل:

- Naive Bayes با دقت کمتری همراه است و برای کلاس‌های ۰ و ۱ نتایج کمی ضعیف‌تری دارد.

جمعبندی:

- Random Forest با دقت بالا و عملکرد خوب برای تمام کلاس‌ها بهترین عملکرد را از نظر دقت نشان داده است.

- SVM نیز عملکرد خوبی داشته و می‌تواند یک گزینه معقول باشد.

- Naive Bayes با دقت کمتر، به خصوص برای کلاس‌های ۰ و ۱، کمتر مورد توجه قرار می‌گیرد.

۳-۲- طبقه بندی با روش های چند لایه

مقایسه و تحلیل گزارش‌های طبقه‌بندی بر اساس خروجی برای روش‌های چند لایه (Voting Classifier و Stacking Classifier) انجام شده است:

۱. Voting Classifier:

- دقت (Accuracy): 96.64%

- گزارش طبقه‌بندی:

- دقت بسیار بالا برای تمام کلاس‌ها (۰.۹۳ تا ۱.۰۰).

- میانگین دقت (macro avg) و میانگین دقت و بازخوانی (weighted avg) به خوبی نزدیک به ۱ هستند.

- تحلیل:

- Voting Classifier عملکرد خوبی داشته و دقت بالایی در تشخیص کلاس‌ها نشان داده است.

۲. Stacking Classifier:

- دقت (Accuracy): 97.97%

- گزارش طبقه‌بندی:

- دقت بسیار بالا برای تمام کلاس‌ها (۰.۹۶ تا ۱.۰۰).

- میانگین دقت (macro avg) و میانگین دقت و بازخوانی (weighted avg) به خوبی نزدیک به ۱ هستند.

- تحلیل:

- Stacking Classifier عملکرد بسیار بالایی داشته و دقت بیشتری نسبت به Voting Classifier دارد.

جمعبندی:

- هر دو روش چند لایه، Voting Classifier و Stacking Classifier، دقت بسیار بالایی در تشخیص کلاس‌ها دارند.

- Stacking Classifier با دقت بالاتر نسبت به Voting Classifier به نظر می‌آید و ممکن است برای مسائل پیچیده‌تر و با داده‌های بزرگتر، عملکرد بهتری داشته باشد.

منابع و مراجع

[1] https://www.kaggle.com/datasets/heeraldedhia/groceries-dataset

- پیوست‌ها

- **https://colab.research.google.com/drive/1zfcsaLsrfIxK6BE63S13biQuJCcpvyNj?usp=sharing**

Abstract

In this project, first, a data set containing the information of gas sensors in the environment was prepared. Then, using several different algorithms such as Random Forest, SVM and Naive Bayes, we classified the data. A detailed evaluation was performed on each classifier, which included comparison of results and sensitivity analysis to hyperparameters.

Then, we moved towards multi-model classification. The concept of Stacking was used, which combines a set of basic classifiers including Random Forest, SVM and Naive Bayes, and more efficiency was achieved by using a final classifier.

The results were analyzed in terms of accuracy and other evaluation criteria for each classifier as well as multimodel. In addition, the sensitivity of hyperparameters was also carefully examined for Random Forest, SVM and Multimodel Stacking. These analyzes provide the researcher with useful information to make the necessary improvements in the models.

Keywords:

Classification, gas sensors, Random Forest, SVM, Multimodels

Amirkabir University of Technology
(Tehran Polytechnic)

Department of Computer Science

Project 5

Advanced methods in Classification

By
Samin Mahdipour

Supervisor
Dr.Ghatee

Advisor
Dr.Yousofi Mehr

November 2023