به نام خدا



**تمرین 3 (دسته بندی)**

**هوش مصنوعی و یادگیری ماشین**

**نام دانشجو:**

امیرعلی محمودزاده طوسی

810603142

**استاد درس:**

دکتر شریعت پناهی

اردیبهشت 1404

در این تمرین، با استفاده از داده‌های مربوط به شرایط عملکردی یک ماشین فرز (شامل دما، سرعت، گشتاور و سایش ابزار)، هدف آن بود که با روش‌های یادگیری ماشین، وضعیت ابزار از لحاظ خرابی پیش‌بینی شود. ابتدا بررسی شد که آیا ابزار دچار خرابی شده یا نه (مسئله دودسته‌ای)، سپس نوع خرابی مشخص شد (مسئله چندکلاسه). تمرکز اصلی بر روی پیش‌پردازش دقیق داده‌ها، انتخاب ویژگی‌های مؤثر، تنظیم پارامترهای مدل‌ها، و تحلیل عملکرد الگوریتم‌ها بود.

ابتدا کتابخانه های مورد نیاز مثل: numpy, pandas, seaborn, matplotlib, sklearn را فراخوانی میکنیم.

تنظیماتی را به جهت نمایش کامل سطر، ستون و سلول ها در جداول اعمال میکنیم تا از خلاصه شدن جدول ها در پایتون جلوگیری شود.

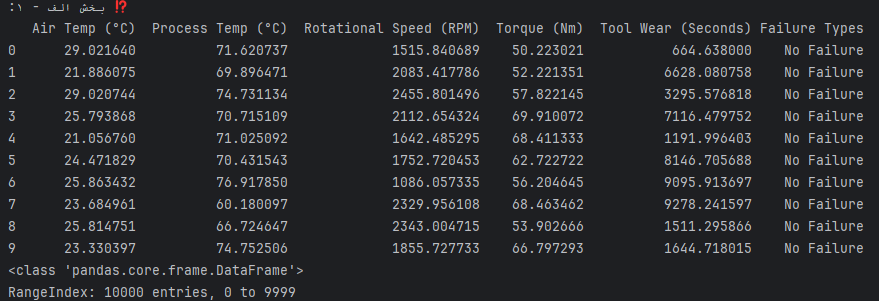
سپس به بخش اول تمرین یعنی بررسی داده های خام میپردازیم:

**الف) بررسی داده های خام:**

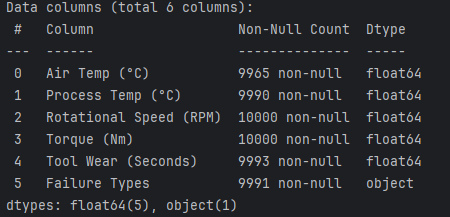
**الف-۱)**

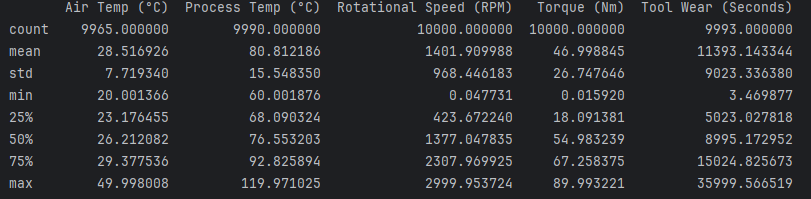
در ابتدا داده ها از فایل miling\_machine.csv بارگذاری شد، نمایش جدول داده ها، بررسی اولیه آماری و ساختار کلی داده‌ها با دستورات info, describe, head استخراج شد.

نمایش ده سطر اول جدول داده ها:



نمایش ساختار کلی و بررسی آماری داده ها :

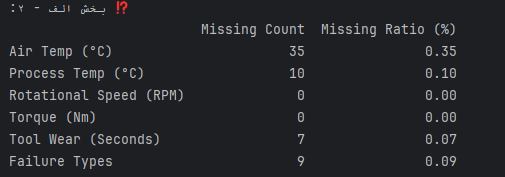




**الف-۲)**

در این بخش برای هر ویژگی تعداد و نسبت مقادیر ناموجود (missing value) استخراج و در جدولی نمایش داده شد:

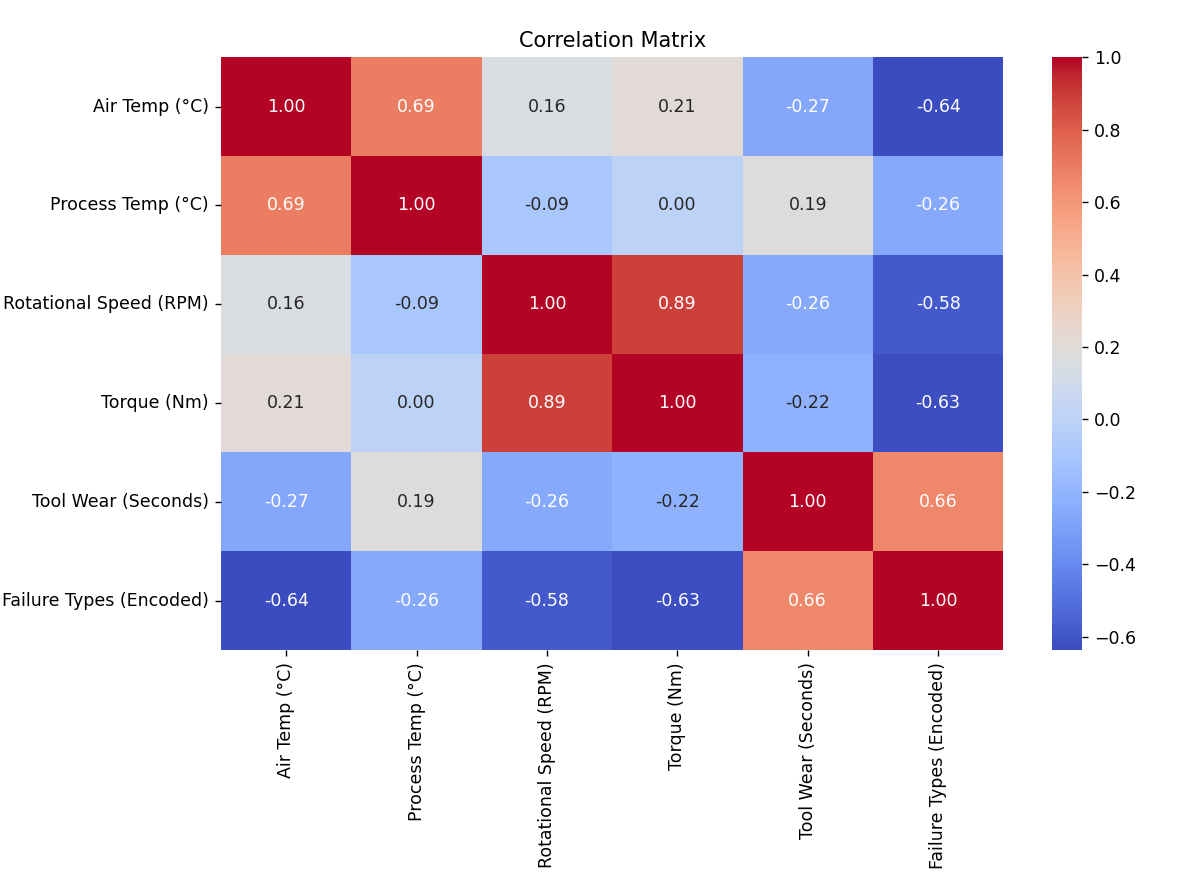
برای محاسبه تعداد داده های گمشده از دستور .isnull().sum() و برای محاسبه نسبت تعداد داده های گمشده بر طول کل دیتاست تقسیم و ضرب در صد شد.



**الف-۳)**

در این بخش correlation بین دو به دو ویژگی ها بررسی شد سپس با استفاده از تحلیل همبستگی عددی و تصویری (heatmap)، سه ویژگی زیر به عنوان مهم‌ترین عوامل مرتبط با خرابی ابزار انتخاب شدند:

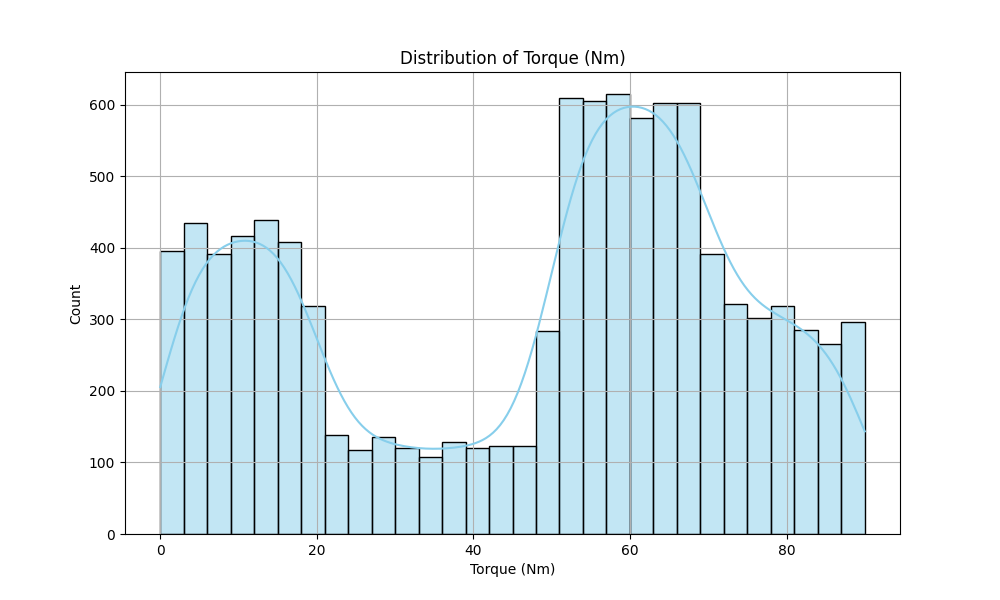
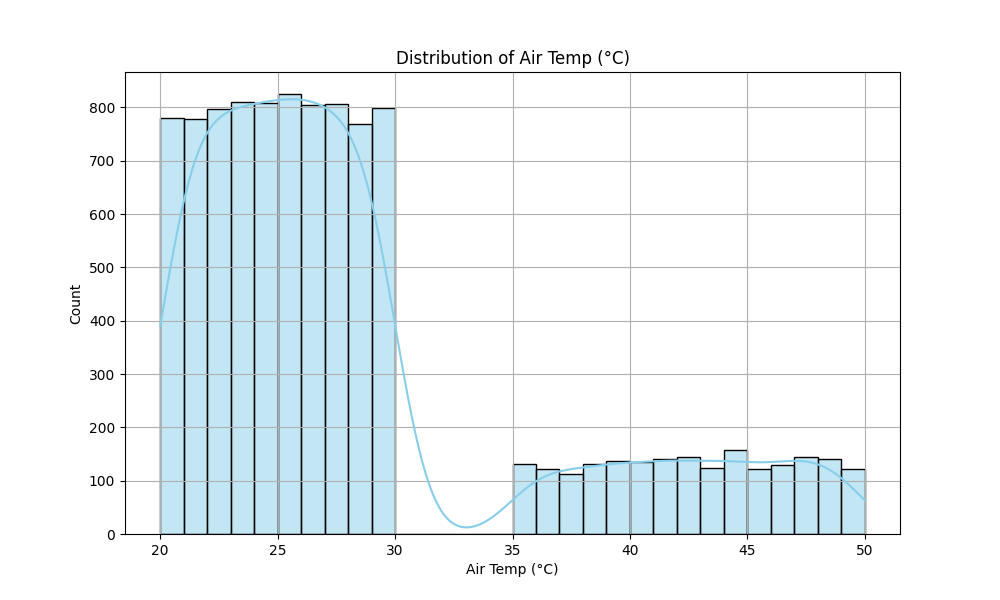
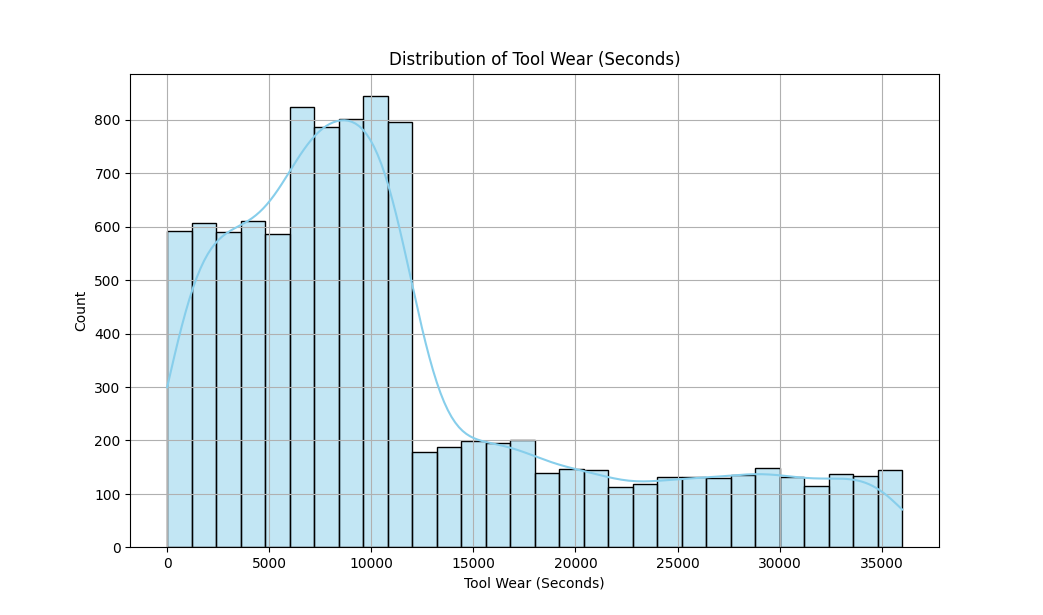
میزان سایش ابزار (Tool Wear)- دمای هوای اطراف (Air Temp)- گشتاور اعمال‌شده (Torque)

با توجه به اینکه ستون Failure Types متنی یا اسمی (categorical) هست (مثلاً No Failure, Tool Wear Failure, ...). برای محاسبه همبستگی عددی باید این ستون رو به مقادیر عددی تبدیل کنیم. چون نمی‌تواند مستقیم توی corr() استفاده شود و باید با LabelEncoder تبدیل به مقادیر عددی شود. (در این مرحله فقط برای بررسی همبستگی مجبور به انکد شدیم و چون هنوز داده ها دارای مقادیر گمشده هستند و این مقادیر فقط با نوع داده float64 قابل نمایش هستند مقادیر نسبت داده شده توسط انکدر اعشاری بوده و در مراحل بعدی برای تربیت مدل بعد از اتمام پیش پردازش این مسأله برطرف خواهد شد.)

**الف-۴)**

در این بخش برای ویژگی های انتخاب شده بخش قبل نمودار توزیع تعداد مشاهدات هر مقدار منحصر به فرد رسم شد.

نمودار توزیع ویژگی‌های مهم نشان می‌دهد که مقادیر Tool Wear نسبت به سایر ویژگی‌ها توزیع متمرکزتری دارد و از نظر آماری می‌تواند شاخص قوی‌تری برای تشخیص وضعیت ابزار باشد. در مقابل، توزیع دمای هوا و گشتاور کمی پراکنده‌تر است و نیاز به بررسی بیشتری دارد.



**ب)پیش پردازش داده ها:**

**ب-۱)**

با توجه به مقادیر و نسبت های داده های گمشده که در بخش های قبلی بدست آوردیم در این بخش مشکل این مقادیر ناموجود را رفع می کنیم.

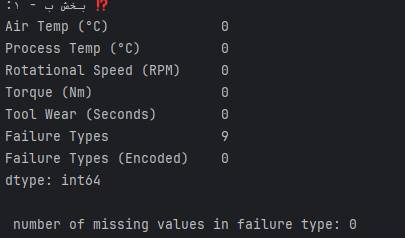
برای جایگزینی داده‌های گمشده، فرض شد که می‌توان از میانگین هر ویژگی در همان کلاس خرابی استفاده کرد.

این روش از جایگزینی میانگین کلی، دقیق‌تر است چون تفاوت بین کلاس‌ها را در نظر می گیردو به مدل کمک می‌کند تا ویژگی‌ها بهتر نمایانگر وضعیت واقعی باشد همچنین از بین بردن bias ناشی از مقداردهی ثابت برای همه‌ی کلاس‌ها را فراهم میکند.

ابتدا میانگین هر ویژگی در هر کلاس محاسبه شد سپس موقعیت داده های ناموجود پیدا شد و با میانگین همان کلاس جایگزین گردید و در نهایت دیتاست جدید جایگزین دیتاست قبلی شد.

تعداد داده های گمشده بررسی شد و مشاهده شد که نوع خرابی ۹ داده گمشده دارد پس ردیف‌هایی که مقدار خرابی در آن‌ها مشخص نبود، حذف شدند. چون داده‌ی بدون برچسب نمی‌تواند در آموزش مدل استفاده شود و اگر تعداد کم باشد، حذف آنها آسیبی به مدل وارد نمی کند.

نتایج نهایی این بخش:



**ب-۲)**

برای استفاده مؤثر از الگوریتم‌های یادگیری ماشین، مقیاس‌دهی ویژگی‌های عددی (کمی) ضروری است. در این تمرین، از روش استانداردسازی (Standardization) استفاده شد. در ادامه، تفاوت دو روش رایج توضیح داده می‌شود:

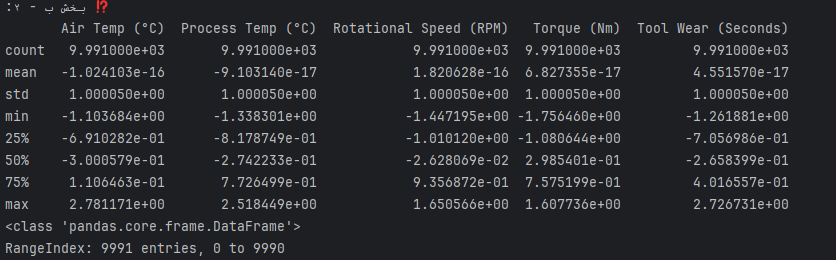
**استانداردسازی (Standardization)**   
در این روش، داده‌ها طوری تغییر می‌کنند که میانگین هر ویژگی صفر و انحراف معیار آن یک شود. این روش برای مدل‌هایی مانند رگرسیون لجستیک و SVM مناسب است، زیرا این الگوریتم‌ها به توزیع داده حساس هستند.

**نرمال‌سازی (Normalization)**   
این روش داده‌ها را در بازه‌ای مشخص، معمولاً [0, 1]، مقیاس‌بندی می‌کند. این روش بیشتر برای الگوریتم‌هایی مانند KNN یا شبکه‌های عصبی مفید است، زیرا این مدل‌ها بر اساس فاصله بین داده‌ها عمل می‌کنند.

در این تمرین چون ویژگی‌هایی مثل Tool Wear، Torque و ... مقیاس‌های بسیار متفاوت دارند و مدل‌هایی مثل KNN و SVM نسبت به مقیاس حساس هستند انجام استاندارد سازی لازم است.

پس همه ویژگی‌های عددی با استفاده از StandardScaler استاندارد شدند تا مقایسه بین ویژگی‌ها در مدل‌ها قابل‌قبول‌تر باشد.

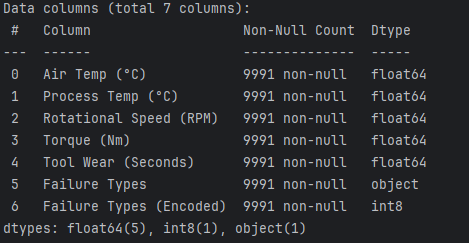
مشاهده می شود که داده ها تقریبا با میانگین صفر و انحراف معیار ۱ مقیاس دهی شدند:



اکنون با توجه به اتمام پیش پردازش داده ها انکد نهایی را ایجاد میکنیم

(چرا دوباره انکد کردیم؟ چون انکد قبلی روی نسخه‌ای از داده بود که دیگه استفاده نمی‌کنیم. برای مدل‌سازی باید روی نسخه نهایی (dataset\_scaled) دوباره انکد انجام بشه)

حال انکد ها دیگر اعشاری نیستند و به صورت int8 شدند.



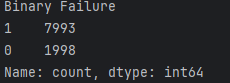
**ج) دسته بندی دوگانه:**

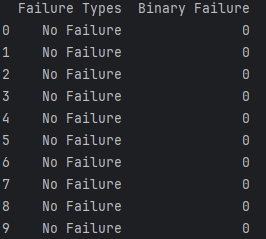
**ج-۱)**

در این بخش هدف ساخت یک ستون جدید است که در آن ابزارها را به دو دسته تقسیم کنیم:

یک ستون جدید با نامBinary Failure ساخته شد تا ابزارهای سالم و معیوب در قالب دودسته‌ای نمایش داده شوند. برچسب 0 به معنای "سالم" و برچسب 1 برای " ابزار معیوب" است.

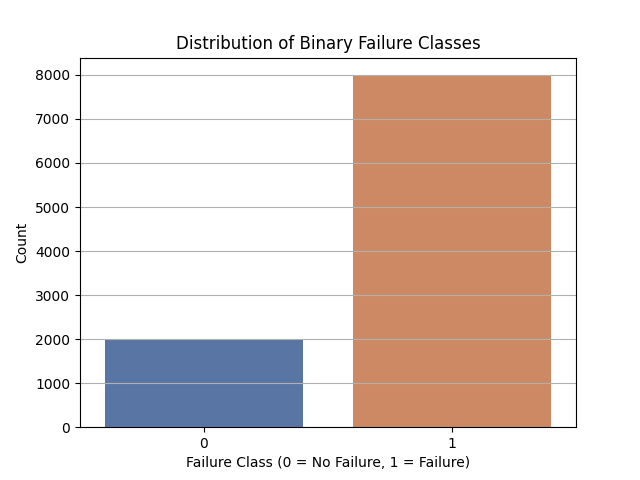
ده سطر اول جدول و مقدار های سالم و معیوب به شرح زیر است:





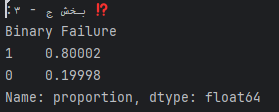
**ج-۲)**

در این بخش میخواهیم توازن داده ها را بررسی کنیم، نمودار میله ای را رسم میکنیم.



**ج-۳)**

مقدار نسبت هر کلاس را بررسی میکنیم:



80 درصد از داده‌ها مربوط به کلاس معیوب (1) هستند و فقط 20 درصد مربوط به کلاس سالم (0) هستند. برخلاف بسیاری از مسائل تشخیص خرابی، در اینجا کلاس "معیوب" در اکثریت قرار دارد. بنابراین، مدل ممکن است تمایل زیادی به پیش‌بینی خرابی داشته باشد و ابزارهای سالم را کمتر شناسایی کند. در این حالت نیز، نامتوازن بودن داده‌ها می‌تواند منجر به خطاهای نوع دوم (False Positive) شود.

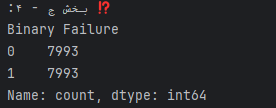
**ج-۴)**

در این قسمت میخواهیم عدم توازن مشاهده شد در قسمت قبلی را برطرف کنیم.

با استفاده از تکنیک SMOTE تعداد نمونه‌های کلاس اقلیت (کلاس سالم = 0) را با داده‌های مصنوعی ولی واقعی‌نما نمونه سازی میکنیم تا متعادل سازی کلاس ها صورت گیرد.

این روش کمک می‌کند مدل دید منصفانه‌تری نسبت به هر دو کلاس داشته باشد.

بعد از انتخاب ویژگی ها و برچسب ها smote را اعمال میکنیم (random\_satate=42) و داده های جدید را به یک دیتافریم جدید تبدی میکنیم، توزیع جدید به صورت زیر است مشاهده می شود که تعداد نمونه کلاس ها مساوی شده است:



**ج-۵)**

داده های پردازش شده را به دو بخش آموزش و تست با نسبت 80/20 تقسیم می کنیم.

پارامتر random\_state=42 برای تکرارپذیری نتایج تنظیم شد.

از پارامتر stratify برای حفظ نسبت کلاس‌ها در دو بخش استفاده شد تا ارزیابی منصفانه‌تر صورت گیرد.



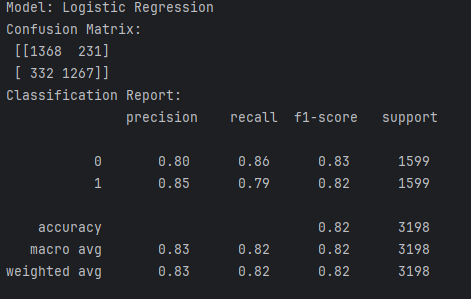
بعد از تقسیم داده ها مدل های زیر را با استفاده از کتابخانه scikit-learn آموزش می دهیم:

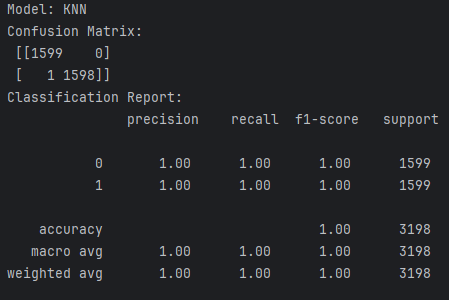
* Logistic Regression
* K-Nearest Neighbors (KNN)
* Support Vector Machine با هسته خطی
* SVM(RBF) با هسته غیر خطی

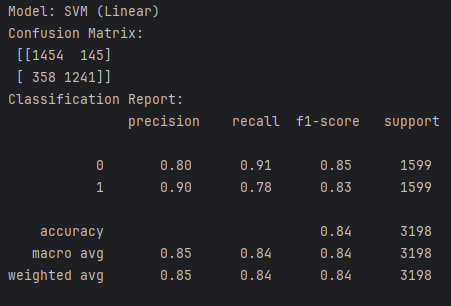
برای مدل‌ها از .values.ravel() استفاده می‌کنیم چون y\_train به‌صورت DataFrame است و بعضی مدل‌ها با Series راحت‌تر کار می‌کنند.

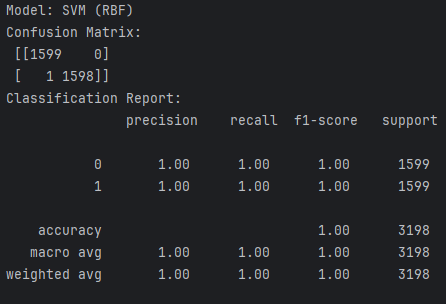
**ج-۶)**

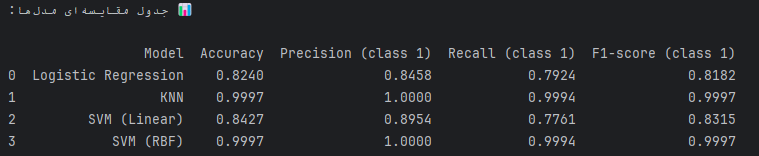
برای هر مدل، ماتریس آشفتگی و گزارش ارزیابی استخراج شد. نتایج نشان می‌دهد که مدل هایKNN و SVM(RBF) عملکرد بهتری در شناسایی خرابی ابزار (class 1) دارد. عملکرد هر مدل و جدول خلاصه عملکرد مدل‌ها:



****

****

****

****

مقادیر 1یا نزدیک به 1 در Recall و Precision معمولاً نشانگر زیادی تمیز یا ساده بودن داده‌هاست، با وجود دقت بسیار بالا در مدل‌های KNN و SVM (RBF)، احتمال Overfitting وجود دارد و نیاز به تنظیم دقیق‌تر هایپرپارامترها در مرحله بعد احساس می‌شود.

**ج-۷)**

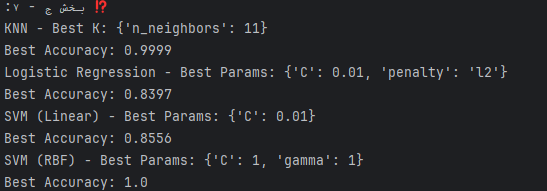
در این بخش برای بهبود عملکرد مدل‌ها، با استفاده از GridSearchCV پارامترهای مختلف مورد آزمون قرار می دهیم، برای همه مدل ها دو پارامتر را انتخاب میکنیم (برای KNN فقط ‌k) و مقدار بهینه (بهترین دقت) را پیدا میکنیم.

|  |  |  |
| --- | --- | --- |
| مدل | پارامترها | مقدار پارامترها |
| KNN | n\_neighbors | [3, 5, 7, 9, 11] |
| Logistic Regression | C, penalty | 'C': [0.01, 0.1, 1, 10] 'penalty': ['l1', 'l2'] |
| SVM (Linear) | C, kernel=’linear’ | 'C': [0.01, 0.1, 1, 10] |
| SVM (RBF) | C, gamma | 'C': [0.1, 1, 10] 'gamma': [0.01, 0.1, 1] |

پیش از تنظیم پارامترها با GridSearchCV، مدل‌ها با پارامترهای پیش‌فرض زیر آموزش داده شدند:

* KNN با n\_neighbors=5
* Logistic Regression با c=1.0 , penalty=l2
* SVM(linear) با c=1.0
* SVM(RBF) با c=1.0, gamma=scale

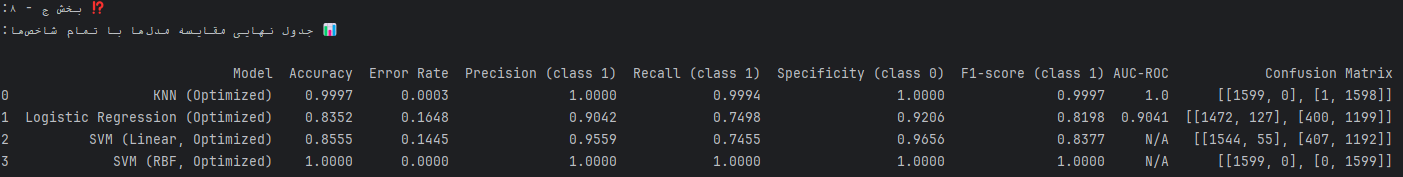
سپس با استفاده از GridSearchCV، مقادیر مختلف برای این پارامترها تست شده و بهترین ترکیب انتخاب شد. مقادیر پارامتر های بهینه و بهترین دقت به شرح زیر است:



**ج-۸)**

در این بخش به بررسی مقایسه عملکرد مدل ها می پردازیم، از شاخص های زیر که در درس معرفی شده‌اند استفاده میکنیم:

* Accuracy (درصد پیشبینی درست)
* Precision (دقت برای کلاس مثبت) (کلاس ۱)
* Recall (Sensitivity) (پوشش کلاس مثبت) (کلاس ۱)
* Error Rate (1 - Accuracy)
* Specificity (پوشش کلاس منفی) (کلاس 0)
* F1-score (میانگین هماهنگ Precision و Recall) (کلاس ۱)
* AUC-ROC (توانایی مدل در جدا کردن دو کلاس)
* Confusion Matirx

جدول زیر تمام معیارهای ارزیابی عملکرد مدل‌ها را با استفاده از داده‌ی آزمون نشان می‌دهد. این مقایسه بر اساس شاخص‌هایی است که در طول درس معرفی شده‌اند، و عملکرد کلی مدل‌ها از جنبه‌های مختلف بررسی شده است:

**د) دسته بندی چندگانه :**

**د-۱)**

در این بخش مدل‌سازی مستقیم برای پیش‌بینی مقدار ستون Failure Types (Encoded) صورت می گیرد.

داده به دو بخش آموزش و تست با نسبت 80/20 و random\_state=42 تقسیم می شوند.

سپس مدل های زیر را آموزش می دهیم:

K-Nearest Neighbors

Decision Tree

Random Forest

Support Vector Machine با روش یکی در برابر همه

برای SVM نمی‌توان مستقیماً چندکلاسه آموزش داد زیرا SVC فقط دودسته‌ای هست، برای همین از OneVsRestClassifier استفاده کردیم.

از probability=True استفاده کردیم چون بعدا AUC-ROC حساب میکنیم.

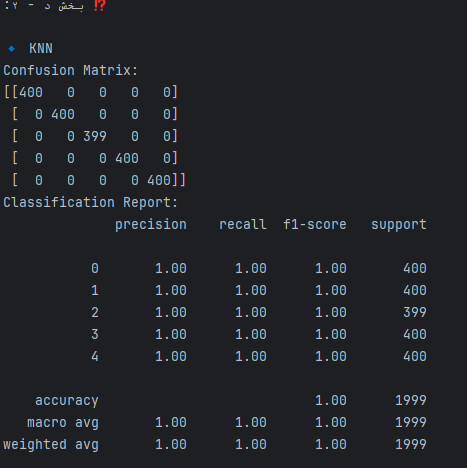
همه‌ی مدل‌ها از داده‌های X\_train\_mc و y\_train\_mc استفاده می‌کنند.

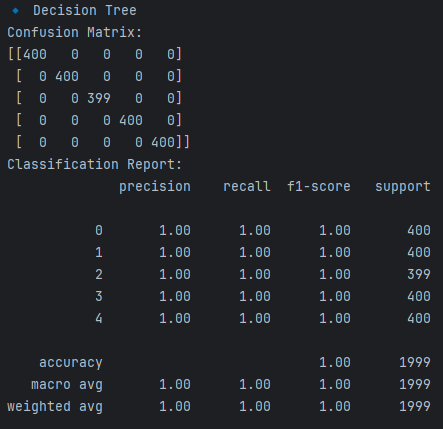
از پارامتر stratify برای حفظ نسبت کلاس‌ها استفاده شد تا ارزیابی منصفانه‌تر صورت گیرد.

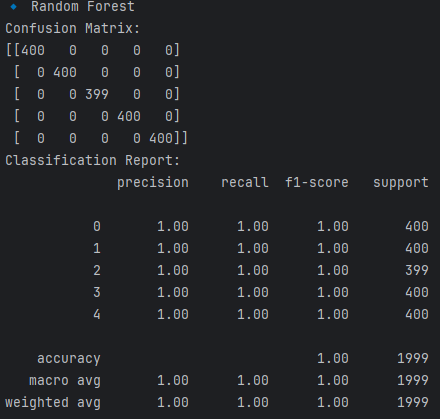
**د-۲)**

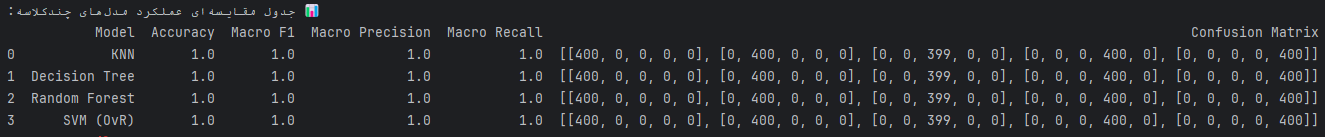
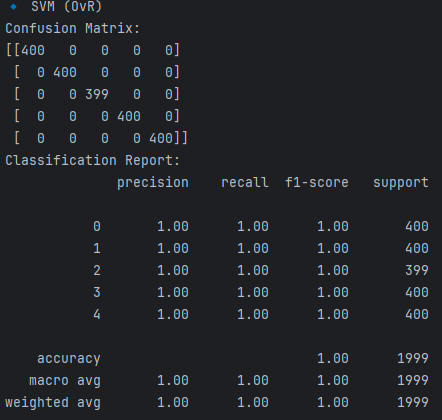
در این بخش مانند بخش ج-۶ با تشکیل ماتریس آشفتگی دقت و عملکرد مدل ها را بررسی میکنیم.

با این تفاوت که در این بخش پیش بینی نوع کلاس خرابی و ارزیابی همه کلاس ها به یک اندازه اهمیت دارد و مثل بخش قبل دوگانه نیست، در مسئله چندکلاسه، اگر کلاس‌ها نامتوازن باشند، accuracy به تنهایی گمراه‌کننده‌ست. پس ما برای مقایسه بهتر از macro avg (وزن برابر برای همه کلاس‌ها (حتی کم‌تعدادها)) استفاده کردیم (به‌جای فقط کلاس ۱).







****

نتایج نشان می‌دهد که همه‌ی مدل‌ها عملکرد بسیار بالایی دارند، که می‌تواند ناشی از قابل‌تفکیک بودن ویژگی‌ها برای دسته‌بندی کلاس‌ها باشد. با این حال، ممکن است چنین عملکردی در داده‌های واقعی‌تری که نویز یا همپوشانی دارند، به دست نیاید.

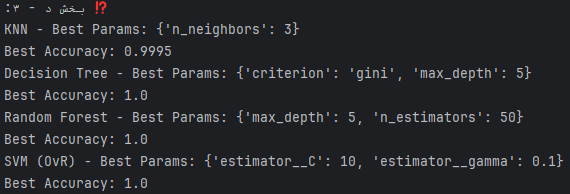
با توجه به جدا سازی داده های آموزش و تست بعد از پیش پردازش (طبق ترتیب تمرین) داده‌های آموزش و آزمون خیلی شبیه‌ یا به‌شکل کامل از همدیگه قابل تفکیک‌ هستند و چون از داده‌های متوازن‌شده (SMOTE) و ویژگی‌های تمیز و استانداردشده استفاده کردیم، احتمالاً مدل‌ها خیلی راحت توانستند کلاس‌ها رو یاد بگیرند و تفکیک کنند.

با توجه به اینکه دقت مدل ها ۱ است نیازی به بهینه سازی بیشتر نیست، اما به جهت تکمیل تمرین ادامه مراحل نیز انجام می شود.

**د-۳)**

در این بخش برای بهبود عملکرد مدل‌ها، با استفاده از GridSearchCV پارامترهای مختلف مورد آزمون قرار می دهیم، برای همه مدل ها دو پارامتر را انتخاب میکنیم (برای KNN فقط ‌k) و مقدار بهینه (بهترین دقت) را پیدا میکنیم.

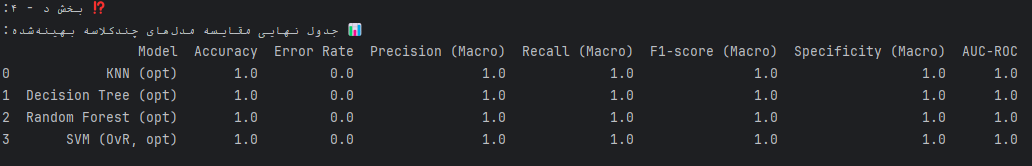
|  |  |  |
| --- | --- | --- |
| مدل | پارامترها | مقدار پارامترها |
| KNN | n\_neighbors | [3, 5, 7, 9, 11] |
| Decision Tree | max\_depth, criterion | 'max\_depth': [5, 10, 20, None], 'criterion': ['gini', 'entropy'] |
| RandomForest | n\_estimators, max\_depth | 'n\_estimators': [50, 100, 200], 'max\_depth': [5, 10, 20, None] |
| SVM (OvR) | C, gamma | 'C': [0.1, 1, 10] 'gamma': [0.01, 0.1, 1] |



نتایج نشان داد که مدل‌ها تقریباً به دقت کامل دست یافتند، حتی قبل از بهینه‌سازی، و GridSearch تغییر چشم‌گیری در عملکرد نهایی ایجاد نکرد.  
این نشان می‌دهد که ویژگی‌های انتخاب‌شده، داده‌ی آموزشی تمیز و قابل‌تفکیک را فراهم کرده‌اند.

**د-۴)**

در این بخش به بررسی مقایسه عملکرد مدل ها می پردازیم، از شاخص های زیر که در درس معرفی شده‌اند استفاده میکنیم:

* Accuracy (درصد پیشبینی درست)
* Precision (دقت برای کلاس مثبت) (Macro Average)
* Recall (Sensitivity) (پوشش کلاس مثبت) (Macro Average)
* Error Rate (1 - Accuracy)
* Specificity (پوشش کلاس منفی) (برای چندکلاسه: میانگین TN / (TN+FP))
* F1-score (میانگین هماهنگ Precision و Recall) (Macro Average)
* AUC-ROC (توانایی مدل در جدا کردن دو کلاس – اگر ممکن باشد - فقط اگر مدل predict\_proba داشته باشد)
* Confusion Matirx

