



دانشگاه صنعتی امیرکبیر

(پلی‌تکنیک تهران)

دانشکده مهندسی کامپیووتر

پروژه کارشناسی

گرایش معماری سیستم‌های کامپیووتری

پیاده‌سازی سیستم نگهداری و تعمیرات پیش‌بینانه  
تجهیزات بر بستر اینترنت اشیاء مبتنی بر تحلیل لرزش

نگارنده

میلاد اسرافیلیان نجف‌آبادی

استاد راهنما

دکتر حمیدرضا زرندی

۱۴۰۲ تیر

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِيْمِ

# صفحه فرم ارزیابی و تصویب پایان نامه- فرم تأیید اعضاء کمیته دفاع

در این صفحه فرم دفاع یا تایید و تصویب پایان نامه موسوم به فرم کمیته دفاع- موجود در پرونده آموزشی- را قرار دهید.

## نکات مهم:

- نگارش پایان نامه/رساله باید به **زبان فارسی** و بر اساس آخرین نسخه دستورالعمل و راهنمای تدوین پایان نامه های دانشگاه صنعتی امیرکبیر باشد.(دستورالعمل و راهنمای حاضر)
- رنگ جلد پایان نامه/رساله چاپی کارشناسی، کارشناسی ارشد و دکترا باید به ترتیب مشکی، طوسی و سفید رنگ باشد.
- چاپ و صحافی پایان نامه/رساله بصورت **پشت و رو(دورو)** بلامانع است و انجام آن توصیه می شود.



دانشگاه صنعتی امیرکبیر  
(پلی‌تکنیک تهران)

به نام خدا

تاریخ: تیر ۱۴۰۲

## تعهدنامه اصالت اثر

اینجانب میلاد اسرافیلیان نجف‌آبادی متعهد می‌شوم که مطالب مندرج در این پایان‌نامه حاصل کار پژوهشی اینجانب تحت نظرات و راهنمایی استادی دانشگاه صنعتی امیرکبیر بوده و به دستاوردهای دیگران که در این پژوهش از آنها استفاده شده است مطابق مقررات و روال متعارف ارجاع و در فهرست منابع و مأخذ ذکر گردیده است. این پایان‌نامه قبلًا برای احراز هیچ مدرک هم‌سطح یا بالاتر ارائه نگردیده است.

در صورت اثبات تخلف در هر زمان، مدرک تحصیلی صادر شده توسط دانشگاه از درجه اعتبار ساقط بوده و دانشگاه حق پیگیری قانونی خواهد داشت.

کلیه نتایج و حقوق حاصل از این پایان‌نامه متعلق به دانشگاه صنعتی امیرکبیر می‌باشد. هرگونه استفاده از نتایج علمی و عملی، واگذاری اطلاعات به دیگران یا چاپ و تکثیر، نسخه‌برداری، ترجمه و اقتباس از این پایان‌نامه بدون موافقت کتبی دانشگاه صنعتی امیرکبیر ممنوع است. نقل مطالب با ذکر مأخذ بلامانع است.

میلاد اسرافیلیان نجف‌آبادی

امضا

تھڈھم بہ مادر پر عزیزم

# پاسکزاری

از پدر و مادرم که همواره در مواجهه با سختی‌های این دنیا دلسوزانه همراهم بوده‌اند؛  
از استاد بزرگوارم جناب آقای دکتر حمیدرضا زرندی که با حسن خلق و گشاده‌رویی، رهنمودهای  
شبانه‌روزی خود را از من دریغ نکرده‌اند؛  
از آقای مهندس عاروان بابت زحمات، پیگیری، کمک، همراهی و نظارت بر این پروژه‌ی کارشناسی  
که بی‌شک بدون اینها انجام این پروژه میسر نمی‌بود؛  
و از سایر عزیزانی که در کنارشان این نتیجه حاصل آمد کمال تشکر و قدردانی را دارم.

میلاد اسرافیلیان بحق آبادی  
سیر ۱۴۰۲

## چکیده

موقفیت اینترنت اشیاء به توانایی ما در حل مشکلات چالش برانگیز که قبلاً غیرممکن بودند بستگی دارد. یکی از حیاتی‌ترین چالش‌ها در فضای اینترنت اشیاء، نگهداری پیش‌گیرانه در فرآیندهای تولید صنعتی برای به حداقل رساندن در دسترس بودن و دوام تجهیزات است. به طور سنتی، مدیریت پیش‌گیرانه فقط از استراتژی‌های کم‌هزینه‌تر، مثلاً مدیریت مبتنی بر زمان یا استفاده، پیروی می‌کند. با حجم زیادی از داده‌های حسگر جمع‌آوری شده از اینترنت اشیاء، می‌توانیم تعمیر و نگهداری پیش‌بینی‌کننده بسیار هوشمندتری را بر اساس پیش‌بینی دقیق خرابی ماشین‌آلات توسعه دهیم. در این پژوهش، ما سیستمی را متشکل از یک کارگزار اصلی و یک بسته‌یادگیری ماشین بر حسب داده‌های حسگرهای لرزش برای پیاده‌سازی مدیریت پیش‌بینانه توسعه داده‌ایم. به طور دقیق‌تر فرآیند ساخت مدل یادگیری ماشین را بر اساس پیش‌پردازش داده‌های حسگرهای استخراج ویژگی‌های مناسب و سپس تحلیل این ویژگی‌ها پیش بردیم. در نهایت با یادگیری مدلی خطی بر اساس تحلیل و مقایسه‌ی این ویژگی‌ها با دستگاه‌های سالم، زمان مفید باقی‌مانده برای هر گره موجود را پیش‌بینی کردیم. با استفاده از چنین سیستمی در صنعت، هزینه‌های نگهداری و همچنین ضررهای احتمالی متحمل شده بشدت کاهش خواهد یافت.

### واژه‌های کلیدی:

نگهداری پیش‌بینانه، تحلیل لرزش، یادگیری ماشین، اینترنت اشیاء

# فهرست مطالب

صفحه	عنوان
۱	۱ مقدمه
۲	۱-۱ مقدمه
۳	۲-۱ تعریف مسئله
۴	۳-۱ کارهای مشابه
۶	۲ تکنولوژی‌های استفاده شده
۷	۱-۲ زبان برنامه‌نویسی
۷	۱-۱-۲ زبان برنامه‌نویسی پایتون
۹	۲-۲ چارچوب‌ها و کتابخانه‌ها
۱۲	۳-۲ پایگاه داده‌ی رابطه‌ای
۱۳	۴-۲ داکر
۱۴	۵-۲ جمع‌بندی و نتیجه‌گیری
۱۶	۳ استقرار مدل یادگیری ماشین
۱۷	۱-۳ زیرساخت به عنوان خدمت
۱۷	۲-۳ روش استقرار مدل
۲۰	۳-۳ جمع‌بندی و نتیجه‌گیری
۲۱	۴ پیاده‌سازی سیستم
۲۲	۱-۴ معماری مایکروسرویس
۲۲	۱-۱-۴ داکر و داکر کامپوز
۲۳	۲-۱-۴ سرویس کارگزار اصلی
۲۳	۳-۱-۴ سرویس‌های پایگاه داده
۲۴	۲-۴ معماری سیستم
۲۷	۳-۴ پایگاه داده‌ی رابطه‌ای
۲۸	۴-۴ امنیت
۲۹	۱-۴-۴ احراز هویت

۳۰	۲-۴-۴ تایید مدیران و دروازه‌ها
۳۱	۵-۴ جمع‌بندی و نتیجه‌گیری
۳۲	۵ پیاده‌سازی و توسعه مدل یادگیری ماشین
۳۳	۱-۵ توضیح مسئله
۳۴	۲-۵ پیش‌پردازش
۳۴	۱-۲-۵ از بین بردن انحرافات
۳۵	۲-۲-۵ از بین بردن داده‌های پرت
۳۷	۳-۲-۵ استخراج ویژگی‌ها
۳۹	۳-۵ نحوه‌ی یادگیری مدل
۴۰	۱-۳-۵ محاسبه‌ی مشابهت بین وضعیت دستگاه‌ها
۴۰	۲-۳-۵ پیاده‌کردن مدل
۴۳	۴-۵ جمع‌بندی و نتیجه‌گیری
۴۴	۶ جمع‌بندی، نتیجه‌گیری و پیشنهادات
۴۵	۱-۶ جمع‌بندی و نتیجه‌گیری
۴۵	۲-۶ چالش‌ها
۴۶	۳-۶ محدودیت‌ها
۴۶	۴-۶ پیشنهادات
۴۹	منابع و مراجع
۵۳	پیوست

# فهرست اشکال

صفحه

شکل

۱-۱ مقایسه‌ی هزینه‌های انواع نگهداری‌ها	۲	
۱-۲ نمودار جریان کار	۴	
۱-۳ لوگوی FastAPI	۹	
۲-۱ گراف وابستگی کتابخانه‌های پایتون به NumPy	۱۱	[۱۲]
۲-۲ لогوی PostgreSQL، یکی از معروف‌ترین پایگاه داده‌های رابطه‌ای	۱۳	[۱۷]
۴-۱ معماری داکر	۱۴	[۱۹]
۵-۱ ماشین‌های مجازی در برابر کانتینرهای داکری	۱۵	[۲۰]
۱-۴ انواع سرویس‌های ارائه شده توسط شرکت‌های خدمات ابری	۱۸	[۲۲]
۲-۴ انواع روش‌های استقرار مدل‌های یادگیری ماشین	۱۹	[۲۳]
۱-۴ داکرفایل مربوط به کارگزار اصلی	۲۳	
۲-۴ داکر کامپوز مربوط به سیستم نهایی	۲۴	
۳-۴ معماری کلی سیستم طراحی شده	۲۵	
۴-۴ ساختار بسته‌های سیستم	۲۷	
۵-۴ تعدادی از نقاط انتهایی سامانه	۲۸	
۱-۵ ساختار کلی بسته‌ی هوش مصنوعی	۳۳	
۲-۵ روند خوشه‌بندی یک الگوریتم تراکم محور	۳۶	[۳۰]
۳-۵ مقادیر ویژگی مربع میانگین ریشه برای همه‌ی اندازه‌گیری‌های یک گره	۳۸	
۴-۵ مقادیر ویژگی چگالی طیفی توان و قله‌های موزون برای یک اندازه‌گیری از یک گره	۴۰	
۵-۵ دو مدل یادگیری ماشین برای مسئله‌ی پیش‌بینی میزان عمر مفید باقی‌مانده	۴۲	[۴]
۱-۶ نحوه‌ی طبقه‌بندی مدل یادگیری ماشین بر اساس پارامترهای مختلف	۴۷	[۳۲]

## فهرست جداول

صفحه

جدول

۲۸ .....	۱-۴ شمای جدول مدیران .....
۲۹ .....	۲-۴ شمای جدول دروازهها .....
۳۴ .....	۱-۵ توضیحات نشانه‌گذاری داده‌ها
۳۴ .....	۲-۵ برچسب‌های استفاده شده برای تعیین وضعیت دستگاهها

## فهرست نمادها

نماد	مفهوم
$N$	تعداد گره‌های موجود
$M$	تعداد اندازه‌گیری‌های لرزش مربوط به گره‌ها
$K$	تعداد همه‌ی نمونه‌های موجود در یک اندازه‌گیری
$n$	گره $n$ ام
$m$	اندازه‌گیری $m$ ام
$k$	نمونه‌ی $k$ ام در یک اندازه‌گیری
$a_{nmk}$	بردار سه‌بعدی مربوط به اندازی‌گیری لرزش
$\hat{a}$	بردار هنجارشده $a$
$r_{nm}^l$	ویژگی مربع میانگین ریشه برای یک اندازه‌گیری از یک گره در یکی از ابعاد
$p_{nm}$	ویژگی چگالی طیفی توان برای یک اندازه‌گیری از یک گره
$D_a$	میزان تفاوت دستگاه موجود از یک دستگاه در کلاس کاری نو

مقدمه

فصل اول

## ۱-۱ مقدمه

در صنعت، نگهداری و تعمیرات<sup>۱</sup> به تمام فعالیت‌های اطلاق می‌شود که بر روی ابزارهای صنعتی انجام می‌شود تا بهره‌وری و عمر این ابزارها افزایش یابد. در سال‌های اخیر، رویکردهای مختلفی برای انجام نگهداری مورد استفاده قرار گرفته است. روش‌های نگهداری زیر، از میان همه‌ی این رویکردها، بیشترین فراوانی استفاده در صنعت را دارند [۱]:

- **نگهداری و تعمیرات اصلاحی<sup>۲</sup>**: به جایگزینی قطعه خراب شده در سیستم می‌پردازد. در این رویکرد، تا زمانیکه فرایнд جایگزینی قطعه معیوب به اتمام نرسد، سیستم غیرقابل بهره‌برداری است و تعمیر قطعات بعد از خرابی هزینه‌های قابل توجهی برای صاحبان صنعت به همراه دارد [۲].
- **نگهداری و تعمیرات جلوگیرانه<sup>۳</sup>**: سعی در پیش‌گیری از اتلاف زمان ناشی از توقف اضطراری دارد، اما در عوض ممکن است تعدادی از قطعاتی که هنوز عمر مفید دارند، دور ریخته شوند و اسراف در هزینه و قطعات مصرفی صورت گیرد [۲].
- **نگهداری و تعمیرات پیش‌بینانه<sup>۴</sup>**: سعی می‌کند مشکلات دو نوع نگهداری و تعمیرات مذکور را حل کند. با استفاده از این روش، زمان عملیاتی هر قسمت دستگاه تخمین زده می‌شود و قطعاتی که توسط سیستم مشکوک به خرابی در آینده هستند تعویض می‌گردند و بنابراین ابزارهای موجود در سیستم به صورت بهینه مورد استفاده قرار می‌گیرند و هزینه‌های تعمیرات بشدت کاهش می‌یابد [۲، ۳].

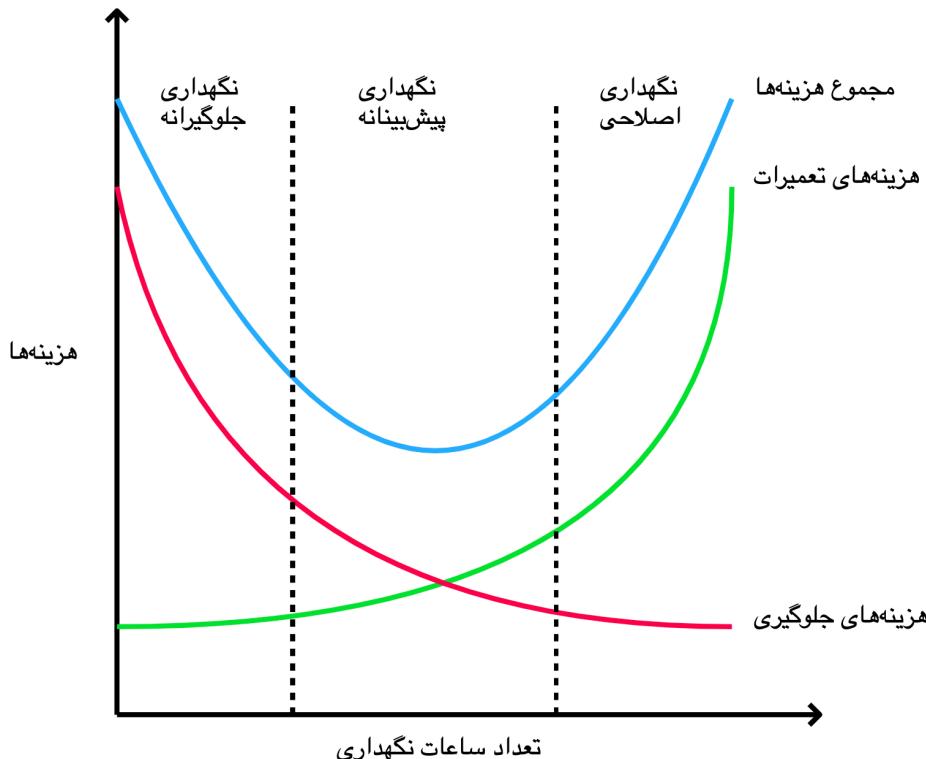
بدلیل اینکه در نگهداری پیش‌بینانه قطعات در حال خرابی، پیش از وقوع خرابی شناسایی می‌شوند و ناکارآمدی آن بخش به کل سیستم آسیب نمی‌رساند، همانطور که در **شکل ۱-۱** مشخص است، با استفاده از این نوع نگهداری، می‌توان مجموع هزینه‌های نگهداری و تعمیرات را به حداقل میزان ممکن رساند [۳].

<sup>1</sup>Maintenance

<sup>2</sup>Corrective Maintenance

<sup>3</sup>Preventive Maintenance

<sup>4</sup>Predictive Maintenance



شکل ۱-۱: مقایسه‌ی هزینه‌های انواع نگهداری‌ها

## ۲-۱ تعریف مسئله

هدف از انجام این پروژه، پیاده‌سازی سیستمی برای اجرا کردن نگهداری پیش‌بینانه بر روی گره‌های موجود در یک اینترنت اشیاء<sup>۵</sup> بهم پیوسته است. رویکردهای مختلفی برای این منظور تا کنون توسط محققان ابداع و مورد استفاده قرار گرفته شده است. از جمله‌ی این موارد می‌توان به تحلیل لرزش<sup>۶</sup> اشاره کرد. برای پیاده‌سازی این سیستم همانطور که در شکل ۲-۱ به تصویر آمده است، نیازمند آنیم که داده‌های لرزش مربوط به گره‌ها را که توسط یک سیستم قابل‌اتکا<sup>۷</sup> جمع‌آوری شده است، دریافت کرده و با جدا کردن داده‌های پرت<sup>۸</sup>، از بین بردن تاثیر اختلال<sup>۹</sup> ایجاد شده توسط گرانش و خرابی یا درست کار نکردن حسگر<sup>۱۰</sup> اندازه‌گیری لرزش، استخراج ویژگی<sup>۱۱</sup>‌های مناسب برای انجام تحلیل روی داده و درنهایت پیشنهاد دادن

<sup>5</sup>Internet of Things

<sup>6</sup>Vibration Analysis

<sup>7</sup>Reliable

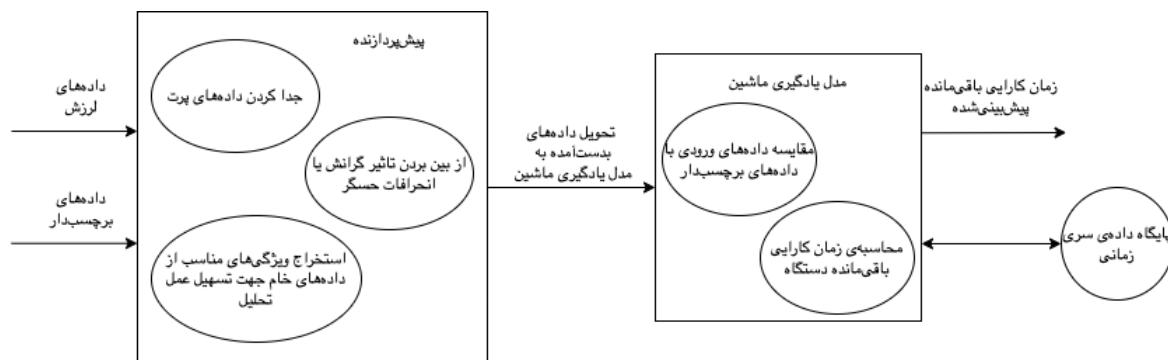
<sup>8</sup>Outlier Data

<sup>9</sup>Noise

<sup>10</sup>Sensor

<sup>11</sup>Feature Extraction

مدلی برای نحوه‌ی یادگیری ماشین<sup>۱۲</sup> و تحلیل و مقایسه‌ی داده‌های با داده‌های برچسب‌دار<sup>۱۳</sup>، عمر باقی‌مانده<sup>۱۴</sup>ی دستگاه‌های مختلف را پیش‌بینی کنیم و بر اساس اعداد بدست‌آمده، اقدامات مناسب را برای انجام مراقبت‌های دوره‌ای انجام دهیم و از تحمیل‌شدن هزینه‌های جانبی در آینده جلوگیری کنیم<sup>[۴]</sup>. برای راحتی استفاده از سیستم طراحی شده، مستقرساختن<sup>۱۵</sup> سرویس توسعه‌یافته شده روی ابر<sup>۱۶</sup> و همچنین احراز هویت مدیر<sup>۱۷</sup>ان و دروازه<sup>۱۸</sup>های ارسال‌کننده داده‌ی لرزش کارگزار<sup>۱۹</sup>ی نیز پیاده خواهد شد.



شکل ۱-۲: نمودار جریان کار

### ۱-۳ کارهای مشابه

نگهداری و تعمیرات پیش‌بینانه نسبتاً موضوع نو ظهوری است و عمر کمتری را نسبت به انواع دیگر نگهداری‌های موجود دارد. اما با این حال تا به امروز تلاش‌های قابل توجهی برای بکارگیری این نوع از نگهداری در سطح دنیا صورت گرفته است که در اینجا به مواردی که رویکردهای جالبی داشته‌اند اشاره خواهیم کرد. در [۵] مدلی برای خرایی قطعات مبتنی بر میزان استفاده از تجهیزات و بار داخلی آنها ارائه شده است اما مدل ارائه شده محدود به یک مدل تجهیزات است و برای استفاده در سایر مدل‌ها نظارت مجدد لازم است. در [۶] روندی مبتنی بر شبکه‌های عصبی جهت پیش‌بینی عمر مفید باقی‌مانده

<sup>12</sup>Machine Learning

<sup>13</sup>Labeled Data

<sup>14</sup>Remaining Useful Lifetime

<sup>15</sup>Deploy

<sup>16</sup>Cloud

<sup>17</sup>Admin

<sup>18</sup>Gateway

<sup>19</sup>Server

تجهیزات چرخشی ارائه شده است اما تنها مختص به این دسته از تجهیزات است. در [۷] رویکردی مبتنی بر شبکه عصبی جهت پیش‌بینی زمان نگهداری و تعمیرات تجهیزات بر اساس مدل خرابی و کارایی آنها ارائه می‌شود اما در یک محیط شبیه‌سازی شده و کنترل شده آزمایش شده است و در هنگام استفاده در محیط واقعی غیرعملی است.

بطور کلی در پژوهش‌های یادشده روندهای در پیش‌گرفته شده برای پیش‌بینی خرابی و زمان آن، مختص نوع خاصی از تجهیزات است و در یک محیط آزمایشگاهی و کنترل شده ارزیابی شده‌اند. در حالی که در این پروژه با استفاده از داده‌های جمع‌آوری شده از قطعات مختلف سعی کرده‌ایم رویکردی کلی و مناسب محیط واقعی و صنعتی ارائه دهیم. همچنین شایان ذکر است که در هیچ‌کدام از پروژه‌های یادشده، سیستم طراحی شده روی ابر مستقر نشده‌اند و سرویس‌هایی همانند احرار احراز هویت و برنامه‌ی تحت وب برای آنها طراحی نشده است. این در حالی است که در این پروژه قصد بر این بوده که سیستمی کلی برای مدیریت بهتر قطعات با جلوه‌ای مناسب طراحی گردد و توسعه یابد.

## فصل دوم

### تکنولوژی‌های استفاده شده

در این فصل تکنولوژی‌ها و چارچوب<sup>۱</sup>‌های اصلی دخیل در توسعه این دستگاه را بطور دقیق مورد بررسی قرار می‌دهیم.

## ۱-۲ زبان برنامه‌نویسی

برای انتخاب زبان برنامه‌نویسی مناسب برای توسعه مدل یادگیری ماشین شرح داده شده، باید معیارهای متفاوتی را در نظر گرفت. برای این منظور زبان پایتون<sup>۲</sup> را برگزیدیم. مواردی همچون داشتن چارچوب‌ها و کتابخانه‌های قدرتمند یادگیری ماشین، توسعه‌ی آسان و سریع و محبوبیت بالا از دلایل اصلی انتخاب پایتون بعنوان زبان اصلی برای توسعه‌ی سرویس یادگیری ماشین می‌باشد. همچنین شایان ذکر است که چون کارگزار اصلی جمع‌آوری اطلاعات لرزش به زبان پایتون نوشته شده است، استفاده از این زبان برای توسعه مدل یادگیری ماشین، باعث بهبود توسعه‌پذیری نیز می‌گردد.

### ۱-۱-۲ زبان برنامه‌نویسی پایتون

یک زبان برنامه‌نویسی عمومی و سطح بالا است که فلسفه طراحی آن بر روی خوانایی کد تأکید دارد. نحو<sup>۳</sup> پایتون به برنامه‌نویسان امکان می‌دهد تا مفاهیم را با تعداد کمتری خط کد نسبت به زبان‌هایی مانند سی<sup>۴</sup> بیان کنند و این زبان ساختارهایی را فراهم می‌کند که برنامه‌های واضح و قابل فهم را در هر دو مقیاس کوچک و بزرگ فراهم می‌سازد<sup>[۱]</sup>. یکی از مشخصه‌های مهم پایتون این است که از چندین الگو<sup>۵</sup> برنامه‌نویسی، از جمله شی‌گرای<sup>۶</sup> و تابعی یا روش‌های رویه‌ای، پشتیبانی می‌کند. پایتون سیستم نوع پویا و مدیریت خودکار حافظه را پشتیبانی می‌کند و کتابخانه‌های استاندارد و جانبی بزرگ و جامع دارد. مفسرهای پایتون برای بسیاری از سیستم‌عامل‌ها در دسترس هستند<sup>[۹]</sup>. از جمله مهم‌ترین ویژگی‌های پایتون می‌توان به موارد زیر اشاره کرد.

- **سادگی:** پایتون یک زبان برنامه‌نویسی بسیار سطح بالا است که منابع زیادی برای یادگیری آن وجود دارد. پایتون از ابزارهای شخص ثالث متنوعی پشتیبانی می‌کند که استفاده از آن را بسیار آسان‌تر می‌کند و کاربران را ترغیب می‌کند تا ادامه دهند<sup>[۹]</sup>, [۱۰].

<sup>1</sup>Framework

<sup>2</sup>Python

<sup>3</sup>Syntax

<sup>4</sup>C Programming Language

<sup>5</sup>Paradigm

<sup>6</sup>Object Oriented Programming (OOP)

- متن باز بودن<sup>۷</sup>: اگرچه تمام حقوق این زبان برنامه‌نویسی متعلق به سازمان پایتون است، اما در حال حاضر به عنوان یک نرم‌افزار متن باز وجود دارد و هیچ محدودیتی در استفاده، تغییر و توزیع آن وجود ندارد. می‌توان به آزادی از پایتون استفاده کرد و آن را برای استفاده شخصی و یا تجاری توزیع کرد. نه تنها می‌توان نرم‌افزاری که با آن نوشته شده است را استفاده و توزیع کرد، بلکه حتی می‌توان تغییراتی در خود کد منبع پایتون اعمال کرد. همچنین شایان ذکر است که پایتون یک جامعه بزرگ و پویا دارد که در هر نسخه آن را بهبود می‌بخشد<sup>[۹، ۱۰]</sup>.
- کتابخانه‌ها و چارچوب‌ها: پایتون دارای یک سری کتابخانه‌های استاندارد و چارچوب‌های متنوع است که کار برنامه‌نویسان را بشدت راحت می‌کند، زیرا نیازی نیست تمام کدنویسی را خود برنامه‌نویس انجام دهد. کتابخانه‌های استاندارد در پایتون بخوبی تست شده‌اند و توسط هزاران نفر استفاده می‌شوند. بنابراین، می‌توان اطمینان داشت که استفاده از این کتابخانه‌ها توانایی ایجاد خرابی در برنامه‌های شما را ندارند<sup>[۹، ۱۰]</sup>.
- حال به بررسی معایب پایتون می‌پردازیم. نکته‌ی قابل توجه در این قسمت این است که اگر معایب نامبرده شده تاثیر زیادی در کیفیت خدمت ارائه شده به کاربر بگذارند، استفاده از پایتون اصلاً توصیه نمی‌شود و باید به دنبال جایگزینی مناسب گشت. از جمله کاستی‌های پایتون عبارت‌اند از:
- کندی: بعنوان یک زبان با نوع پویا، پایتون به دلیل انعطاف‌پذیری بالا، کند عمل می‌کند، زیرا ماشین باید بسیاری از مراجعات را انجام دهد تا از تعریف چیزی مطمئن شود و این باعث کاهش عملکرد پایتون می‌شود<sup>[۹، ۱۰]</sup>.
- دشواری فرایند نگهداری<sup>۸</sup>: بدلیل اینکه پایتون یک زبان با نوع پویا است، یک چیز ممکن است براحتی به معنای متفاوتی در تکنمایی متفاوت تفسیر شود. با افزایش اندازه و پیچیدگی یک برنامه پایتون، نگهداری آن ممکن است دشوار شود. با کمک تست‌های واحد<sup>۹</sup> می‌توان تا حدی این از وقوع این مشکل جلوگیری کرد<sup>[۹، ۱۰]</sup>.

<sup>7</sup>Open Source

<sup>8</sup>Maintaining

<sup>9</sup>Unit Tests

## ۲-۲ چارچوب‌ها و کتابخانه‌ها

در این پروژه از چارچوب فست‌ای‌پی‌آی<sup>۱۰</sup> برای دریافت درخواست‌ها و ارسال نتایج پیش‌بینی استفاده شده است (لوگوی مربوط به این چارچوب در شکل ۲-۲<sup>۱۱</sup> آورده شده است). این سرویس بعنوان یک بسته<sup>۱۲</sup> پایتونی به کارگزار اصلی اضافه شده است. همچنین برای پیاده‌سازی مدل و انجام محاسبات ریاضی و ماتریسی از کتابخانه‌های نام‌پایی<sup>۱۳</sup> و سایکیت<sup>۱۴</sup> بهره برده شده است. در بخش‌های بعد به معرفی مختصر هر کدام از این موارد خواهیم پرداخت. لازم به ذکر است که جهت خوانایی بیشتر، از معادل انگلیسی این کتابخانه‌ها برای اشاره به اسم آنها استفاده خواهیم کرد.

### چارچوب FastAPI

یک چارچوب مدرن با عملکرد عالی برای طراحی وب است که برای پایتون توسعه داده شده است. از ویژگی‌های کلیدی FastAPI می‌توان به موارد زیر اشاره کرد<sup>۱۵</sup>.



شکل ۲-۱: لوگوی FastAPI [۱۱]

- **سریع بودن:** همانطور که در قسمت‌های قبل بدان اشاره شده، یکی از مزایای پایتون کند بودن می‌باشد. نکته‌ی قابل توجه در اینجا این است که با وجود اینکه یکی از چارچوب‌های پایتون است، اما FastAPI بسیار سریع است و کارایی و عملکرد بسیار بالایی را در اختیار می‌گذارد.

- **سادگی توسعه:** بدلیل اینکه این زبان از نحو پایتون برای توسعه بهره می‌برد، سرعت توسعه دهنده

<sup>10</sup>FastAPI

<sup>11</sup>Package

<sup>12</sup>NumPy

<sup>13</sup>Scikit-Learn

برای ایجاد برنامه را دو تا سه برابر نسبت به چارچوب‌های دیگر برای توسعه برنامه‌ی تحت وب افزایش می‌دهد.

- کوتاه‌بودن: این ویژگی باعث می‌شود که تکرار کد به حداقل میزان ممکن برسد و این خود منجر به این می‌شود که اشکالات<sup>۱۴</sup> کمتری که منشاء آن برنامه‌نویس هستند پیش بیایند.

### کتابخانه‌ی NumPy

NumPy یکی از معروف‌ترین کتابخانه‌های زبان پایتون برای پردازش علمی و عددی است و اکنون، ۱۸ سال پس از عرضه، همانطور که در شکل ۲-۲<sup>۱۲</sup> مشخص است، مبنای بسیاری از کتابخانه‌های دیگر پایتون است. این کتابخانه‌ی متن‌باز توسط جامعه‌ی پایتونی توسعه‌یافته است و یک شیء آرایه چندبعدی پایتون به همراه تابع‌هایی که روی آن عمل می‌کنند، ارائه می‌دهد. NumPy بدلیل سادگی ذاتی، به عنوان ساختار اصلی مبادله اطلاعات آرایه‌ای در پایتون مورد استفاده قرار می‌گیرد<sup>۱۳</sup>. آرایه‌ی نام‌پایی در واقع یک ساختمان داده است که به صورت بهینه آرایه‌های چندبعدی پایتون را ذخیره می‌کند و به آنها دسترسی پیدا می‌کند. همچنین توانایی انجام محاسبات علمی مختلف را بر روی این آرایه‌ها برای ما فراهم می‌کند. این ساختمان داده شامل یک اشاره‌گر<sup>۱۵</sup> به حافظه و تعدادی فرآداده<sup>۱۶</sup> برای تفسیر داده‌های موجود در آرایه است<sup>۱۲، ۱۳</sup>.

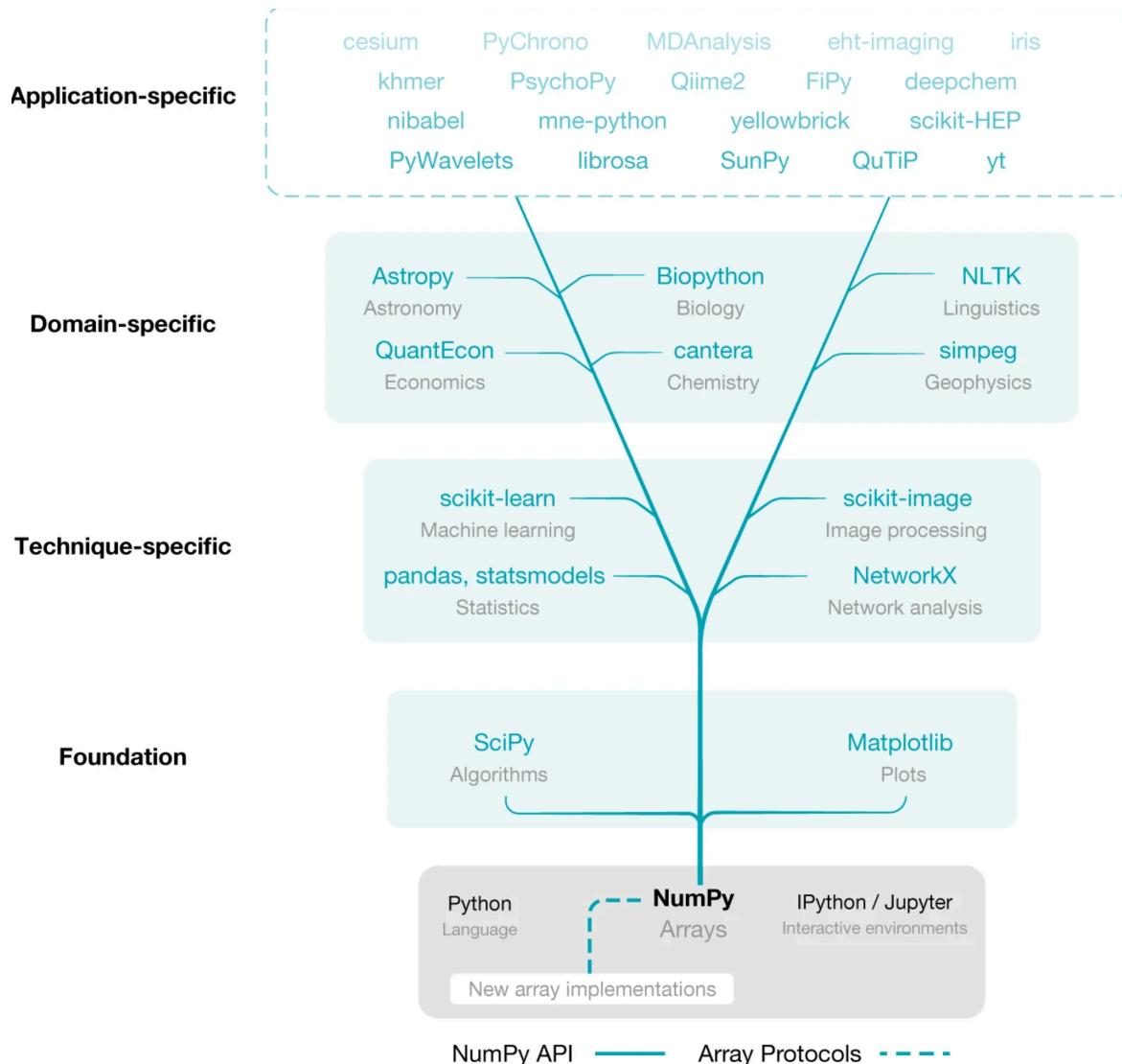
### کتابخانه‌ی Scikit-Learn

Scikit-Learn، جامع‌ترین و بزرگ‌ترین بسته یادگیری ماشین منبع‌باز در پایتون است. چون یادگیری ماشین اغلب عنوان یک جزء از یک برنامه عمومی‌تر (همانند پروژه‌ی کنونی که به عنوان یک سرویس در وب توسعه داده شده است) استفاده می‌شود، ایده‌آل است که از همان زبان برنامه‌نویسی استفاده شود تا بصورت یکپارچه با سایر بخش‌های برنامه هماهنگ شود. با استفاده از قابلیت‌های گسترده پایتون، Scikit-Learn بعنوان یک بسته محبوب برای برنامه‌های مرتبط با یادگیری ماشین در حال رشد است<sup>۱۴</sup>. این کتابخانه شامل توابع و اشیاء فراوانی برای مسائل طبقه‌بندی، رگرسیون، تقریب ماتریس کوواریانس، کاهش بعد و پیش‌پردازش داده‌ی خام می‌باشد<sup>۱۵</sup>. اگرچه پایتون یک زبان برنامه‌نویسی تفسیری است، اما بیشتر روش‌های یادگیری ماشین در Scikit-Learn بر پایه کتابخانه‌های دودویی کامپایل شده است

<sup>14</sup>Bugs

<sup>15</sup>Pointer

<sup>16</sup>Metadata



[۱۲] NumPy وابستگی کتابخانه‌های پایتون به

که در ابتدا با زبان‌های فورتران<sup>۱۷</sup>، سی یا سی‌پلاس‌پلاس<sup>۱۸</sup> برنامه‌نویسی شده‌اند. این پیاده‌سازی‌های مبتنی بر دودویی‌ها بطور قابل توجهی کارایی محاسبات را بهبود می‌بخشند[۱۴، ۱۵].

### کتابخانه‌های جانبی

در این پروژه برای توسعه‌ی بدون خطای سرویس‌های مختلف، از کتابخانه‌های جانبی دیگری نیز استفاده شده است که در اینجا به آنها و موارد کاربردشان اشاره می‌کنیم.

از این دو کتابخانه برای توسعه‌ی سرویس احراز هویت مدیران و

<sup>17</sup>Fortran

<sup>18</sup>C++

دروازه‌های ارسال کننده‌ی داده‌های مربوط به لرژش با کمک استاندارد JWT استفاده کرده‌ایم.

• برای کارهایی همانند خواندن مجموعه‌ی داده<sup>۱۹</sup> و پردازش روی آنها از این کتابخانه کمک گرفته شد.

• SQLAlchemy: این کتابخانه بدلیل در اختیار گذاشتن رابطه‌ای برنامه‌نویسی<sup>۲۰</sup> مناسب برای ارتباط با پایگاه‌داده‌های رابطه‌ای، بسیار محبوب است و در این پژوهه نیز از آن استفاده کرده‌ایم.

## ۳-۲ پایگاه داده‌ی رابطه‌ای

پایگاه داده برنامه‌ای برای ذخیره و بازیابی سریع حجم فراوانی از داده و بصورت مکرر می‌باشد. پایگاه داده‌ی رابطه‌ای<sup>۲۱</sup> در سال ۱۹۷۰ میلادی معرفی شد. داده در این نوع از پایگاه داده بشكل جداولی<sup>۲۲</sup> (از جدول به عنوان رابطه<sup>۲۳</sup> نیز یاد می‌شود) ذخیره می‌شود و می‌تواند به شکل‌های متفاوت به آن دسترسی پیدا کرد یا آن را تغییر داد. هر ستون این جدول نشان‌دهنده‌ی یک مشخصه و هر سطر نماینده‌ی یک موجودیت از آن رابطه می‌باشد. هر کدام از این جداول می‌توانند با هم‌دیگر ارتباط داشته باشند. از این‌رو به این نوع از پایگاه داده، پایگاه داده‌ی رابطه‌ای گفته می‌شود<sup>[۱۶]</sup>. از معروف‌ترین پایگاه‌داده‌های رابطه‌ای می‌توان به Microsoft SQL Server، PostgreSQL، MySQL و Oracle Database اشاره کرد.

اکثر پایگاه‌های داده رابطه‌ای از زبان پرسمان ساختاریافته<sup>۲۴</sup> برای دسترسی و اصلاح داده‌های ذخیره شده در پایگاه داده استفاده می‌کنند. برای انجام این پژوهه از PostgreSQL که یکی معروف‌ترین پایگاه داده‌ی رابطه‌ای متن‌باز است، استفاده کرده‌ایم (لوگوی آن در شکل ۳-۲ آورده شده است).

<sup>19</sup>Dataset

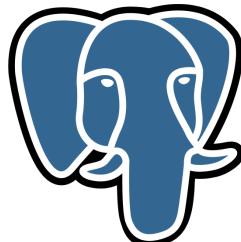
<sup>20</sup>Application Programming Interface (API)

<sup>21</sup>Relational Database

<sup>22</sup>Tables

<sup>23</sup>Relation

<sup>24</sup>Structured Query Language (SQL)



شکل ۲-۳: لوگوی PostgreSQL، یکی از معروف‌ترین پایگاه داده‌های رابطه‌ای [۱۷]

## ۴-۲ داکر

داکر<sup>۲۵</sup> در حال حاضر معروف‌ترین و پراستفاده‌ترین ابزار برای پیاده‌سازی معماری سرویس‌های کوچک<sup>۲۶</sup> و استقرار آنها روی ابر است. وظیفه‌ی اصلی و واحد داکر این است که با بسته‌بندی برنامه‌ها و متعلقات آن بصورت واحدی به نام کانتینر<sup>۲۷</sup>، امکان استفاده و کار کردن آنها را روی هر ماشینی که موتور داکر روی آن نصب شده است را تضمین کند. هر یک از کانتینرهای داکر در یک محیط مجزا و با منابع متفاوت تخصیص داده شده توسط موتور داکر، اجرا می‌شوند. این مدل اجرای ایزوله‌ی کانتینرهای داکر سبب قابل حمل‌بودن واحدهای مختلف اجرایی برنامه، مقیاس‌پذیری راحت برنامه‌های مختلف و همچنین انعطاف‌پذیری بالا نسبت به محیط اجرایی و شرایط وابسته به آن خواهد شد [۱۸، ۱۹].

سرویس داکر همانطور که در شکل ۴-۲<sup>۲۸</sup> مشخص است، خود از چندین بخش مجزا تشکیل شده است که در این قسمت آنها را شرح خواهیم داد. داکر از معماری مشتری-کارگزار<sup>۲۹</sup> استفاده می‌کند. در ساختار داکر، Docker Client با Docker Daemon که کارهایی همانند ساخت، اجرا و توزیع کانتینرهای داکری را انجام می‌دهد، صحبت می‌کند. سرویس‌های Docker Client و Docker Daemon می‌توانند روی یک سیستم اجرا شوند، یا می‌توانند روی دو ماشین متفاوت باشند و با یکدیگر ارتباط برقرار کنند. این دو سرویس با استفاده از رابط کاربری برنامه‌نویسی، با همدیگر تعامل می‌کنند. یکی دیگر از سرویس‌گیرندگان معروف داکر، Docker Compose است که امکان ایجاد و اجرای چند کانتینر داکری را بصورت همزمان برقرار می‌کند. همانطور که فهمیدیم، واحدهای اجرایی در داکر، کانتینر نامیده می‌شوند. نکته‌ی قابل توجه در اینجا این است که این واحدهای اجرایی از واحدهایی به نام ایمیج ساخته و اجرا می‌شوند. هنگامی که Docker Daemon درخواست جدیدی برای بالا آوردن یک کانتینر از Docker Client دریافت می‌کند، در صورتی که ایمیج داکری روی سیستم میزبان قرار داشت بلاfacسله شروع به

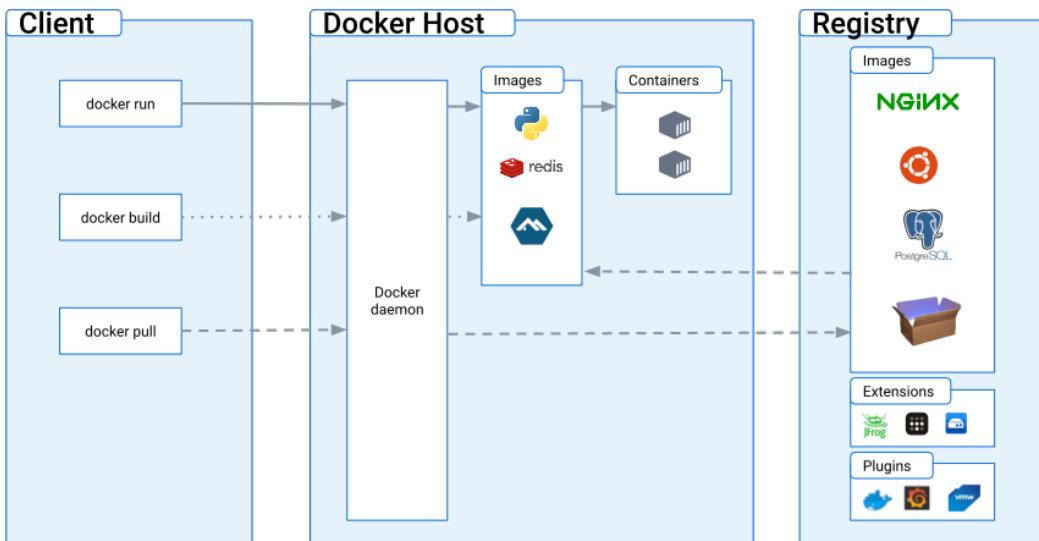
<sup>25</sup>Docker

<sup>26</sup>Microservices

<sup>27</sup>Container

<sup>28</sup>Client-Server

تخصیص منابع و شروع کانتینر مربوطه می‌کند. در غیر این صورت، از واحدی به نام Aymiqj مربوطه را دریافت و سپس همان روند قبلی را طی می‌کند [۱۹].



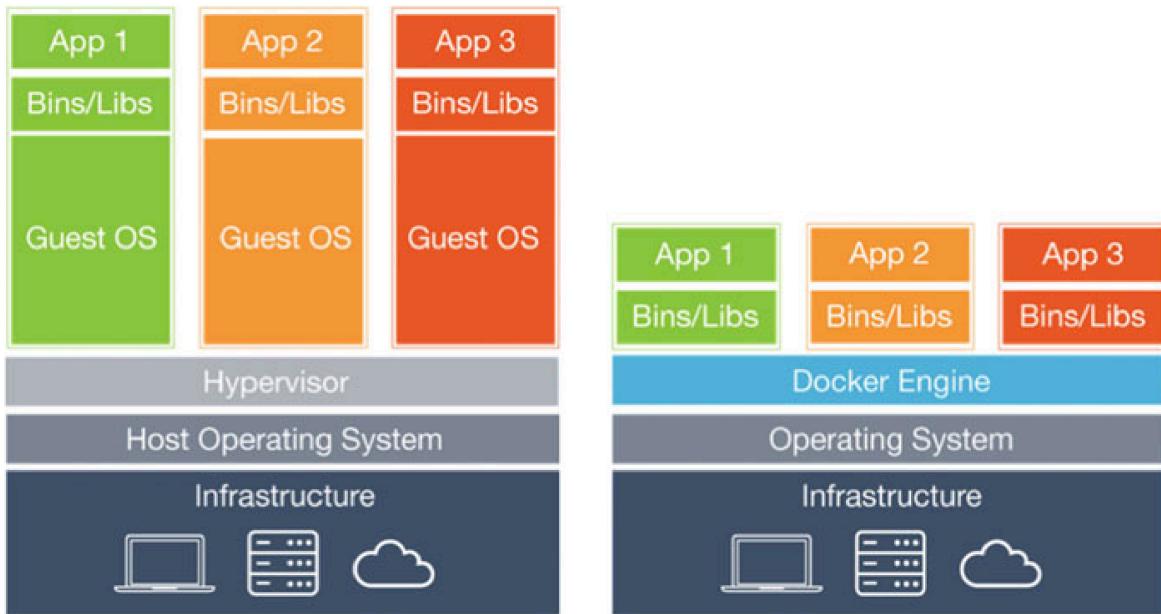
شکل ۴-۲: معماری داکر [۱۹]

توسعه‌ی داکر در واقع تلاشی برای رفع مشکلات استفاده از ماشین‌های مجازی<sup>۲۹</sup> بود. کانتینرهای داکر و ماشین‌های مجازی مطابق شکل ۵-۲ در نحوه مجازی‌سازی و استفاده از منابع کامپیوتر با یکدیگر متفاوت‌اند. ماشین‌های مجازی منابع سخت‌افزاری سیستم را بین خود تقسیم می‌کنند و بدلیل وجود لایه‌ای اضافی برای ایجاد دستورات قابل فهم برای لایه‌ی سخت‌افزار، نسبت به کانتینرهای داکر که منابع سیستم‌عامل را بین خود تقسیم می‌کنند و یک لایه کمتر دارند، کندر اجرا می‌شوند و سربار بشدت بالاتری را متحمل می‌شوند [۱۸، ۲۰].

## ۵-۲ جمع‌بندی و نتیجه‌گیری

در این بخش، به تکنولوژی‌ها و چارچوب‌های اصلی استفاده شده برای توسعه‌ی مدل یادگیری ماشین اشاره کردیم. همانطور که در طول فصل بدان اشاره شد، زبان پایتون بدلیل دارابودن غنی‌ترین کتابخانه‌های مربوط به محاسبات و یادگیری ماشین منطقی‌ترین انتخاب ممکن برای برگزیدن زبان توسعه‌ی مدل و سرویس هوش‌مصنوعی بود. دارابودن چارچوب‌های با کارایی بالا برای توسعه برنامه‌ی وب نیز دلیل

<sup>29</sup>Virtual Machines



شکل ۲-۵: ماشین‌های مجازی در برابر کانتینرهای داکری [۲۰]

مهم برای انتخاب پایتون است. در مرحله‌ی بعد پایگاه‌های داده‌ی رابطه‌ای را معرفی کردیم و تا حدودی با نحوه ذخیره و بازیابی داده در آنها آشنا شدیم و مشخص کردیم که از پایگاه داده‌ی PostgreSQL که یک پایگاه داده‌ی رابطه‌ای متن‌باز است، برای انجام این پروژه بهره بردیم. در انتهای این بخش نیز ابزار داکر را معرفی کردیم و اجزا و قسمت‌های متفاوت آن را بررسی کردیم و همچنین مزایای آن نسبت به ماشین‌های مجازی که روش سنتی استقرار برنامه‌ها روی ابر بود را بر شمردیم.

## فصل سوم

### استقرار مدل یادگیری ماشین

پس از پیاده‌سازی مدل یادگیری ماشین، نیاز است که به طریقی سیستم را در دسترس همگان قرار داد تا بتوان از مزایای آن استفاده کرد. شرکت‌های ارائه‌دهنده خدمات ابری<sup>۱</sup> یا به اختصار CSP، گزینه‌ی مناسبی برای این نیاز می‌باشد. برای این منظور، ما این پروژه را پس از پیاده‌سازی، توسط سرویس زیرساخت بعنوان خدمت<sup>۲</sup> مستقر کردیم.

### ۱-۳ زیرساخت به عنوان خدمت

در این مدل، سرویس‌دهنده ابر یا CSP مجموعه‌ای از منابع محاسباتی مجازی‌شده را در ابر فراهم می‌کند (مانند پهنای باند شبکه، ظرفیت ذخیره‌سازی، حافظه، قدرت پردازش). مسئولیت مشتری در این حالت این است که سیستم‌عامل و برنامه‌های نرم‌افزاری را روی این منابع مجازی اجرا و نگهداری کند. زیرساخت بعنوان خدمت یا IaaS از فناوری مجازی‌سازی استفاده می‌کند تا منابع فیزیکی را به منابع منطقی تبدیل کند که مشتریان می‌توانند بصورت پویا از آنها استفاده کنند و آنها را هنگام نیاز ایجاد و آزاد کنند<sup>۲۱</sup>. در شکل ۱-۳<sup>۲۲</sup> نمای کلی سرویس‌های مختلف موجود در یک سیستم ابری را مشاهده می‌کنیم. همانطور که مشخص است در IaaS، کاربر بیشترین کنترل را بر روی منابع در اختیار گذاشته شده دارد<sup>۲۳</sup>.

### ۲-۳ روش استقرار مدل

روش‌های مختلفی برای استقرار و استفاده از مدل‌های یادگیری هوش مصنوعی مورد استفاده قرار می‌گیرند که از بین اینها چهار روش نشان داده شده در شکل ۲-۳<sup>۲۴</sup> مرسوم‌تر هستند:

- **پیاده‌سازی دسته‌ای<sup>۳</sup>:** پیش‌بینی‌ها به فاصله‌های زمانی مشخص محاسبه می‌شوند و پیش‌بینی‌های حاصل در پایگاه داده ذخیره می‌شوند و براحتی می‌توان آنها را در صورت نیاز بازیابی کرد. با این حال، نمی‌توان از داده‌های بروزتر استفاده کرد و پیش‌بینی‌ها می‌توانند به سرعت منسخ شوند<sup>۲۵</sup>.

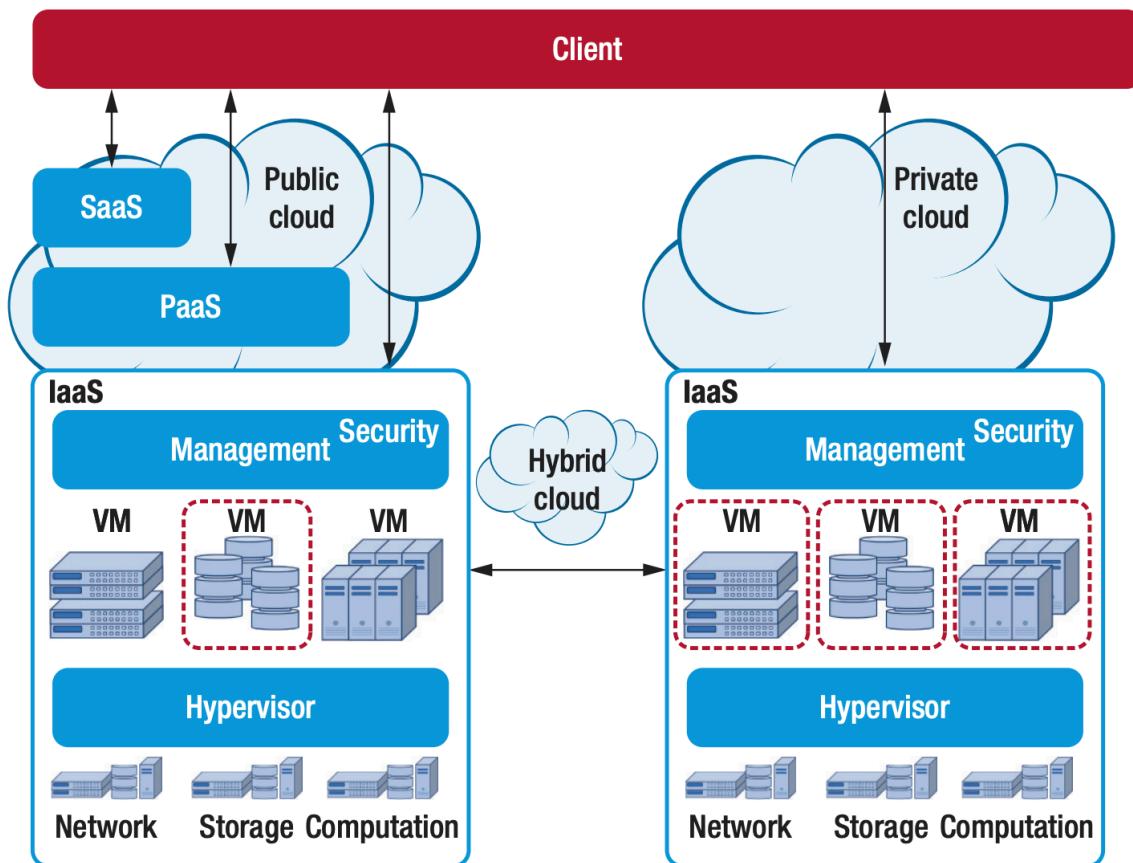
- **پیاده‌سازی بی‌درنگ<sup>۴</sup>:** در این نوع از استقرار، درخواست کاربر برای گرفتن جدیدترین پیش‌بینی‌ها

<sup>1</sup>Cloud Services Providers

<sup>2</sup>Infrastructure as a Service (IaaS)

<sup>3</sup>Batch Deployment

<sup>4</sup>Real-Time Deployment



شکل ۳-۱: انواع سرویس‌های ارائه شده توسط شرکت‌های خدمات ابری [۲۲]

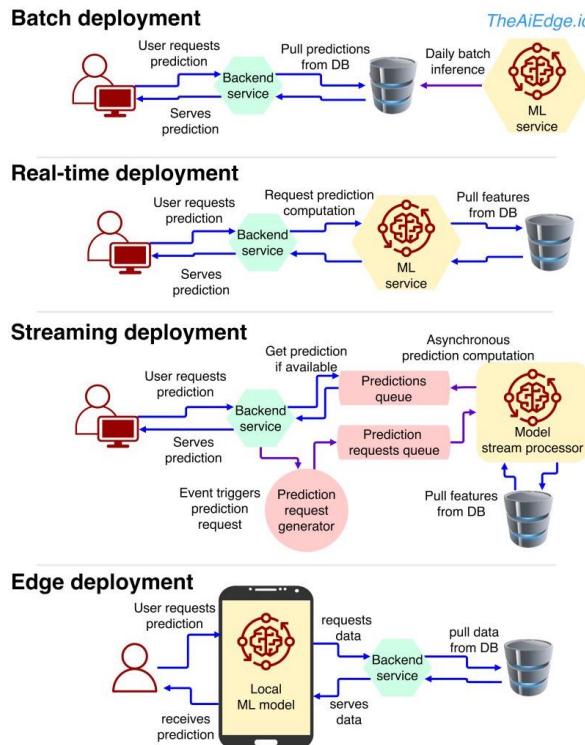
عنوان یک راهانداز<sup>۴</sup> توسط رابط برنامه‌نویسی اج‌تی‌پی<sup>۵</sup> به کارگزار ارسال می‌شود. سپس سرویس یادگیری ماشین که به عنوان افزونه‌ای در سمت کارگزار توسعه یافته است، شروع به کار می‌کند و جدیدترین نتایج پیش‌بینی را تولید و ذخیره می‌کند و به سمت کاربر عنوان نتیجه ارسال می‌کند. مشکل اصلی این روش قرارگیری مدل یادگیری ماشین، کنبدودن روند یادگیری و پیش‌بینی است که منجر به منتظرماندن کاربر می‌گردد. می‌توان با بهره‌گیری از فرآیند<sup>۶</sup> چندریسمانی<sup>۷</sup> برای دریافت درخواست‌های کاربر و انجام مرحله‌ی یادگیری و پیش‌بینی مدل، تا حد زیادی این مشکل را برطرف کرد [۲۴، ۲۵].

<sup>5</sup>Trigger

<sup>6</sup>Hypertext Transfer Protocol (HTTP)

<sup>7</sup>Process

<sup>8</sup>Multi-Threaded



شکل ۳-۳: انواع روش‌های استقرار مدل‌های یادگیری ماشین [۲۳]

• **پیاده‌سازی جریانی<sup>۹</sup>:** این امکان را می‌دهد تا فرآیند ناهمزمان<sup>۱۰</sup> تری ایجاد شود. یک رویداد می‌تواند شروع فرآیند استنتاج را فراهم کند. این فرآیند در صف یک واسط پیام<sup>۱۱</sup> مانند کافکا<sup>۱۲</sup> قرار داده می‌شود و مدل یادگیری ماشینی در هنگام آماده‌شدن برای انجام درخواست، آن را انجام می‌دهد. این کار به سرویس پشتیبانی فرصت می‌دهد و با فرآیند صف بهینه، قدرت محاسباتی بسیاری را صرفه‌جویی می‌کند. پیش‌بینی‌های حاصل شده نیز در صف قرار گرفته و در صورت نیاز توسط سرویس‌های پشتیبانی مصرف می‌شوند. از مزیت‌های این روش نسبت به روش بی‌درنگ، می‌توان به کم شدن تاخیر پاسخ‌دهی به کاربران اشاره کرد[۲۴، ۲۵].

• **پیاده‌سازی لبه‌ای<sup>۱۳</sup>:** در این روش استقرار، مدل مستقیماً بر روی کلاینت نصب می‌شود، مانند مرورگر وب، یک تلفن همراه یا محصولات اینترنت اشیاء. این کار باعث رسیدن به سریع‌ترین استنتاج می‌شود، اما معمولاً مدل‌ها باید به اندازه کافی کوچک باشند تا بتوانند در سخت‌افزارهای

<sup>9</sup>Streaming Deployment

<sup>10</sup>Asynchronous

<sup>11</sup>Message Broker

<sup>12</sup>Apache Kafka

<sup>13</sup>Edge Deployment

کوچکتر نصب شوند [۲۳].

بدلیل اینکه گره‌های موجود در شبکه اشیاء دارای توان پردازشی محدود هستند و اینکه ماهیت مدل هوش مصنوعی مربوط به حوزه‌ی کاری پیش‌بینی عمر دستگاه‌ها بدین‌گونه است که حتماً باید از داده‌های مربوط به همه‌ی گره‌های موجود استفاده کرد، با توجه به گزینه‌های مطرح شده برای استقرار مدل هوش مصنوعی توسعه‌داده شده و همچنین مزایا و معایب هر کدام، از روش استقرار بی‌درنگ برای ارائه و بکارگیری مدل هوش مصنوعی در این پروژه استفاده شده است. بطور دقیق‌تر، مدل هوش مصنوعی بعنوان یک سرویس اضافی برای کارگزار اصلی توسعه داده شده، تعییه شده است.

### ۳-۳ جمع‌بندی و نتیجه‌گیری

در این فصل نحوه‌ی استقرار مدل یادگیری ماشین را بصورت دقیق بررسی کردیم. همانطور که گفته شد، از میان سرویس‌های مختلف ارائه‌شده توسط شرکت‌های ارائه‌دهنده‌ی سرویس‌های ابری، از زیرساخت بعنوان خدمت برای مستقر کردن سرویس استفاده شد. این سرویس بعنوان یک کارگزار برای دریافت درخواست‌های کاربر و همچنین انجام فرایند یادگیری و ارسال نتایج پیش‌بینی به کاربر عمل می‌کند و از میان روش‌های مختلف برای استقرار و یادگیری مدل، روش استقرار بی‌درنگ بدلیل ماهیت اصلی مدل، پیچیده‌نبودن فرایند یادگیری در این مورد بخصوص و همچنین نیاز به بررسی همه‌ی داده‌های ارسال شده توسط گره‌های مختلف برای داشتن دقیق‌ترین پیش‌بینی‌ها انتخاب شد.

# فصل چهارم

## پیاده‌سازی سیستم

در جهت اعمال نگهداری و تعمیرات پیش‌بینانه روی گره‌های موجود در اینترنت اشیاء، توسعه و پیاده‌سازی یک کارگزار برای دریافت درخواست‌ها و ارسال نتایج پیش‌بینی بسیار مفید است. با کمک این کارگزار همچنین فرایندهای احراز هویت<sup>۱</sup> و تایید<sup>۲</sup> مدیران و دروازه‌ها نیز پیش‌برده خواهند شد. همچنین این کارگزار با بسته‌ی هوش مصنوعی که در فصل بعد به آن اشاره می‌کنیم، تعامل خواهد کرد.

## ۱-۴ معماری مايكروسوسي

یکی از معماری‌هایی که در سال‌های اخیر مورد توجه مهندسین نرم‌افزار قرار گرفته‌است، این معماری است که در آن هر کدام از بخش‌های سیستم در فضایی جدا شده و منزوی قرار می‌گيرند و بصورت سرویس‌های کوچکی در می‌آيند و هر کدام از اين سرویس‌ها با يكديگر در ارتباط مي‌باشند. در اين حالت نياز به يك سرویس اصلی مرکзи برای مدیریت سایر سرویس‌ها و ايجاد ارتباط ميان آنها نیز می‌باشيم. از اصلی‌ترین نقاط مثبت اين سیستم اين است که می‌توان مشکل تضاد داشتن کتابخانه‌ها و نيازمندی‌های قسمت‌های مختلف را بسادگی رفع کرد. بدین صورت که هر سرویس دارای کتابخانه‌ها و نيازمندی‌هایی که دارد می‌باشد و ديگر نيازی ندارد تا با بقیه‌ی سرویس‌ها تراز باشد. از مزاياي ديگر اين سیستم عيب‌يابي ساده‌تر آن به نسبت سیستم يكپارچه می‌باشد<sup>[۲۶]</sup>. در اين پروژه سعى بر اين بوده که تا حد ممکن، بخش‌های مختلف پروژه از هم جدا شوند تا بتوان از مزاياي معماری مايكروسوسي استفاده کنیم.

### ۱-۱-۴ داکر و داکر کامپوز

اصلی‌ترین ابزار برای اعمال و پیاده‌سازی معماری مايكروسوسي، داکر می‌باشد. برای استفاده از آن پس از توسعه‌ی سرویس مورد نظر با کمک داکرفایل<sup>۳</sup>، ايمیج مربوط به سرویس را درست کرد و پس از آن بين سرویس‌های مختلف با کمک داکر کامپوز ارتباط برقرار کرد. داکر کامپوز به ما کمک می‌کند تا چندين کانتينر داکري را براحتی مدیریت و اجرا کرد.

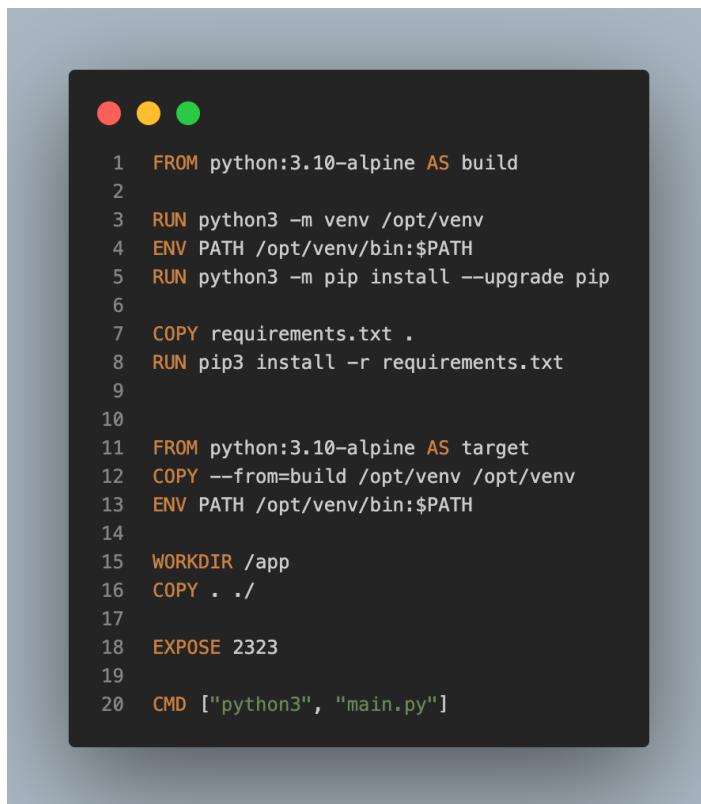
<sup>1</sup>Authetication

<sup>2</sup>Verification

<sup>3</sup>Dockerfile

## ۲-۱-۴ سرویس کارگزار اصلی

کارگزار اصلی به صورت بسته‌هایی شامل احراز هویت، مدل یادگیری ماشین و تحلیل و آنالیز می‌باشد. پس از توسعه‌ی این سرور با کمک داکرفایل نمایش‌داده شده در [شکل ۲-۴](#)، ایمیج مربوط به این کارگزار را تولید کرده و از این پس براحتی با کمک داکر و ایجاد کانتینر از این ایمیج، از آن استفاده می‌کنیم.



```

1  FROM python:3.10-alpine AS build
2
3  RUN python3 -m venv /opt/venv
4  ENV PATH /opt/venv/bin:$PATH
5  RUN python3 -m pip install --upgrade pip
6
7  COPY requirements.txt .
8  RUN pip3 install -r requirements.txt
9
10
11 FROM python:3.10-alpine AS target
12 COPY --from=build /opt/venv /opt/venv
13 ENV PATH /opt/venv/bin:$PATH
14
15 WORKDIR /app
16 COPY . .
17
18 EXPOSE 2323
19
20 CMD ["python3", "main.py"]

```

شکل ۲-۴: داکرفایل مربوط به کارگزار اصلی

## ۳-۱-۴ سرویس‌های پایگاه داده

پس از پیادهسازی کارگزار اصلی، هر کدام از پایگاه داده‌های رابطه‌ای و غیر رابطه‌ای را بعنوان یک سرویس اضافه و مستقل از کارگزار، توسط داکر کامپوز نشان داده شده در [شکل ۲-۴](#) توسعه دادیم و به کارگزار اصلی در یک شبکه‌ی داخلی متصل کردیم.

```

version: '3.9'
services:
  app:
    build: .
    container_name: pdm-api-server
    image: pdm-api-server
    ports:
      - ${SERVER_PORT}:${SERVER_PORT}
    env_file:
      - .env
    depends_on:
      - postgres
      - influxdb
  postgres:
    image: postgres:15.1
    container_name: postgres
    environment:
      POSTGRES_USER: ${POSTGRES_USER}
      POSTGRES_PASSWORD: ${POSTGRES_PASSWORD}
      PGDATA: /data/postgres
    volumes:
      - postgres-data:/data/postgres
  influxdb:
    image: influxdb:2.6.1
    container_name: influxdb
    environment:
      DOCKER_INFLUXDB_INIT_MODE: setup
      DOCKER_INFLUXDB_INIT_USERNAME: ${INFLUXDB_USER}
      DOCKER_INFLUXDB_INIT_PASSWORD: ${INFLUXDB_PASSWORD}
      DOCKER_INFLUXDB_INIT_ORG: ${INFLUXDB_ORG}
      DOCKER_INFLUXDB_INIT_BUCKET: ${INFLUXDB_BUCKET}
      DOCKER_INFLUXDB_INIT_ADMIN_TOKEN: ${INFLUXDB_TOKEN}
    volumes:
      - influxdb-data:/data/influxdb
    ports:
      - ${INFLUXDB_PORT}:${INFLUXDB_PORT}
  volumes:
    postgres-data:
    influxdb-data:

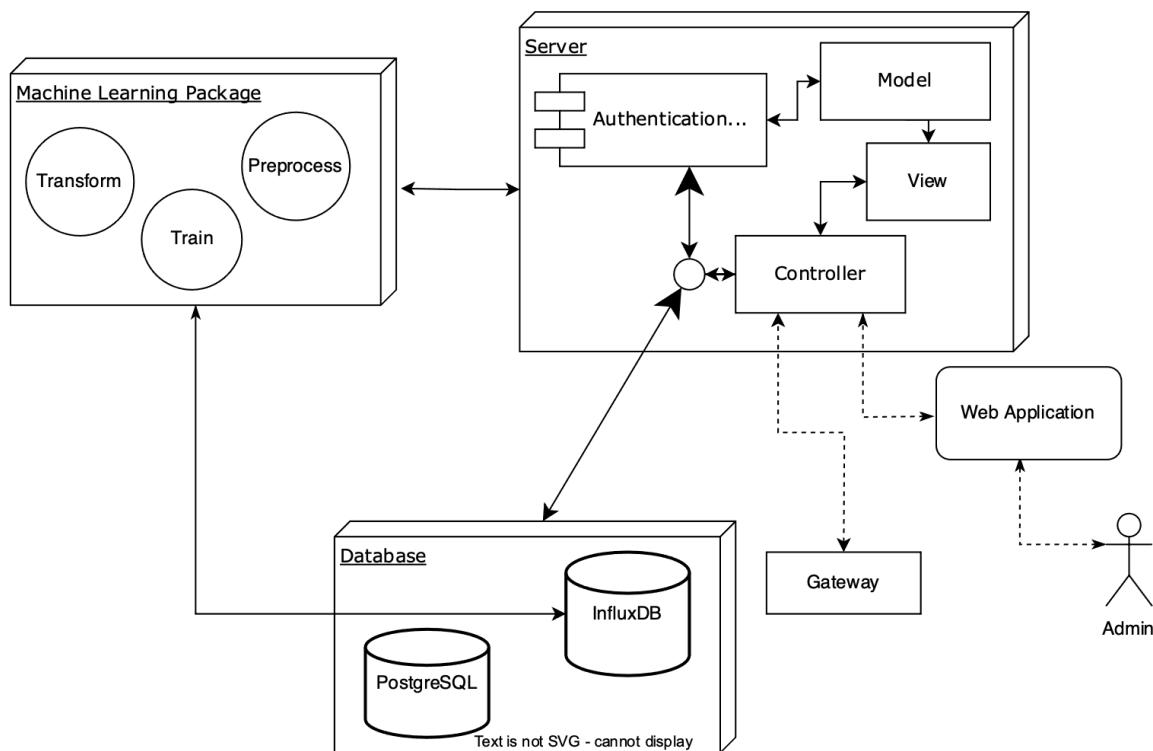
```

شکل ۴-۲: داکر کامپوز مربوط به سیستم نهایی

## ۲-۴ معماری سیستم

در این قسمت ابتدا بخش‌های مختلف سیستم را نشان می‌دهیم سپس به توضیح هر کدام از قسمت‌ها خواهیم پرداخت. همانطور که در شکل ۳-۴ می‌بینیم، سیستم نگهداری پیش‌بینانه طراحی شده بطور کلی از چهار بخش تقسیم‌شده است. کارگزار اصلی که رابط بین مدیران و دروازه‌ها با مدل یادگیری ماشین است و وظیفه دارد بطور مشخص و دقیق درخواست مربوط به هر کدام از کاربران را پاسخ دهد. مهم‌ترین قسمت این سیستم، بسته‌ی هوش مصنوعی می‌باشد که با کارگزار اصلی و پایگاه داده‌ی سری زمانی در ارتباط است و نتایج تحلیل‌ها و پیش‌بینی‌ها را به این دو ابلاغ می‌کند. نکته‌ی شایان توجه در

این قسمت این است که به ازای هر بار آموزش جدید مدل، داده‌ها به مدت ۱۵ دقیقه در پایگاه داده‌ی سری زمانی ذخیره می‌شوند تا سرعت پاسخ‌دهی به کاربران برای درخواست‌های تکراری افزایش یابد.



شکل ۴-۳: معماری کلی سیستم طراحی شده

در [شکل ۴-۴](#) انواع بسته‌های پایتونی مجزا از هم توسعه‌داده شده برای سیستم نگهداری پیش‌بینانه را مشاهده می‌کنیم. در این قسمت، هر کدام از این بخش‌ها را عنوان و تشریح می‌کنیم.

#### بسته‌ی auth

وظیفه‌ی این بخش انجام همه‌ی احراز هویت و تایید کاربران است. همچنین توابع تولید هش رمز عبور کاربران جدید نیز در این بخش پیاده شده است.

#### بسته‌ی influxdb

همه‌ی فایل‌ها و کلاس‌های لازم برای ارتباط با و استفاده از پایگاه داده‌ی سری زمانی، در این بسته قرار دارند. تمام کارهای خواندن و نوشتن داده‌ها و نتایج مدل یادگیری ماشین در این بخش مدیریت می‌شوند. این بخش از آنجا که هم با کارگزار اصلی و هم با بسته‌ی یادگیری ماشین در تعامل است، گلوگاه سیستم حساب می‌شود و ممکن است درخواست‌های زیاد را بخوبی نتواند مدیریت کند. از این رو

در طی پیاده‌سازی این پروژه سعی بر این بوده که این بخش کارایی و سرعت لازم را داشته باشد و در نوشتن پرس‌وجوهای پایگاه داده در این بخش، نهایت دقت به عمل آمده است.

#### بسته‌ی models

این بسته شامل شمای کلی شیء‌های درخواست‌های کاربران و پاسخ‌های مناسب به آنها است.<sup>۲۷</sup> تمامی درخواست‌هایی که به سمت کارگزار اصلی ارسال می‌شوند باید اشیائی طبق شمای بیان‌شده در این بخش را داشته باشند. همچنین ساختار پاسخ‌های کارگزار به کاربران در این قسمت قرار داده شده است.

#### بسته‌ی postgresdb

این بسته نیز شامل همه‌ی فایل‌های لازم برای ایجاد ارتباط با پایگاه داده‌ی رابطه‌ای است. در قسمت‌های بعد بیشتر با این بسته آشنا خواهیم شد.

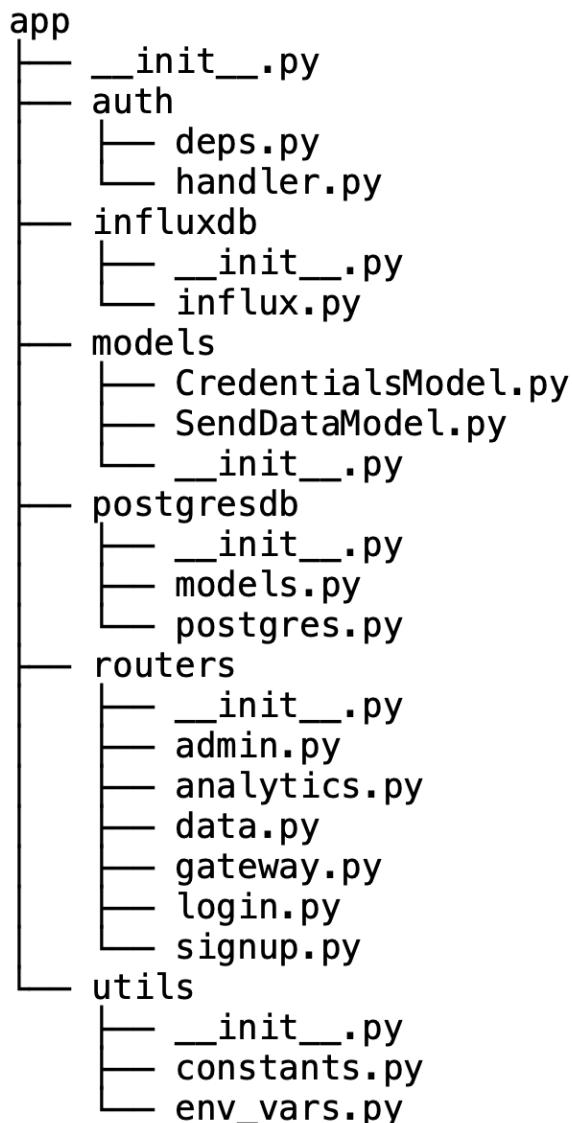
#### بسته‌ی routers

همه‌ی نقاط انتهایی‌ای که در دسترس کاربران قرار گرفته‌اند، در این بخش پیاده و توصیف شده‌اند. همانطور که در [شکل ۴-۴](#) می‌بینیم، برای هر کدام از کارهای ورود، ثبت‌نام، تایید، آنالیز و تحلیل و همچنین ارسال و دریافت داده، نقاط نهایی مشخصی تعییه شده‌اند. در [شکل ۵-۴](#) تعدادی از این نقاط انتهایی را برای تایید مدیران و دروازه‌ها، دریافت و ارسال داده‌های لرزش و ورود کاربران، مشاهده می‌کنیم. نکته‌ی قابل توجه در این بخش این است که الگوی پیاده‌شده در این قسمت، بر اساس الگوی مدل-نمای-کنترل‌گر<sup>۲۸</sup> می‌باشد و این بسته شامل دو بخش نما و کنترل‌گر است.

#### بسته‌ی utils

این بسته نیز شامل مواردی است که در کل سامانه مورد استفاده قرار می‌گیرند. این موارد شامل متغیرهای محیطی<sup>۴</sup>، متغیرهای ثابت و پارامترهای ثابت مدل یادگیری ماشین هستند.

<sup>4</sup>Environment Variables



شکل ۴-۴: ساختار بسته‌های سیستم

### ۳-۴ پایگاه داده‌ی رابطه‌ای

برای مدیریت و نگهداری اطلاعات مدیران و دروازه‌ها در این پروژه از پایگاه داده‌ی رابطه‌ای متن‌باز PostgreSQL استفاده کرده‌ایم. تنها، بخش نما از کارگزار اصلی به این پایگاه داده دسترسی دارد و عملیات احراز هویت و همچنین مدیریت دسترسی کاربران برای دیدن یا درخواست انجام پیش‌بینی‌ها و تحلیل‌ها و صدور مجوز برای ارسال داده‌های حسگرهای لرزش از سمت دروازه‌ها به کارگزار اصلی با کمک این بخش انجام خواهد شد. در جدول ۱-۴ و جدول ۲-۴ به ترتیب شمای مربوط به جداول مدیران و دروازه‌ها را مشاهده می‌کنیم.

admin		
GET	/api/v1/admin/me	Current Admin Info
GET	/api/v1/admin/all	All Admins Info
PUT	/api/v1/admin/verify	Verify Admin
gateway		
GET	/api/v1/gateway/all	All Gateways Info
PUT	/api/v1/gateway/verify	Verify Admin
data		
GET	/api/v1/data	Get All Data
POST	/api/v1/data	Send Data
DELETE	/api/v1/data	Delete Vibration Data
GET	/api/v1/data/node/{nodeId}	Get Node Data
GET	/api/v1/data/measurement/{measurementId}	Get Measurement Data
GET	/api/v1/data/allNodes	Get Nodes Id
GET	/api/v1/data/allMeasurements	Get Measurements Id
login		
POST	/api/v1/login/gateway	Gateway Login
POST	/api/v1/login/admin	Admin Login

شکل ۴-۵: تعدادی از نقاط انتهایی سامانه

جدول ۱-۴: شمای جدول مدیران

مدیر	توضیحات
ستون	
شناسه	عددی یکتا برای هر مدیر
ایمیل	رشته‌ای یکتا برای هر مدیر
رمز عبور	رشته‌ای هش‌شده برای ورود هر مدیر به سیستم
نام	به صورت رشته
تاییدشده	متغیری صفر و یکی
ثبتنام در	زمان ثبتنام مدیر
آخرین ورود در	زمان آخرین ورود مدیر به سیستم
تایید شده در	زمان تایید مدیر توسط یک مدیر تاییدشده

## ۴-۴ امنیت

یکی از مهم‌ترین ویژگی‌های سیستم طراحی شده که وجه تمایز بین این پروژه را با دیگر پروژه‌های مشابه ایجاد می‌کند، پیاده‌شدن لایه‌ها مختلف امنیت در این پروژه است. همه مدیران و دروازه‌ها پس از ثبتنام و ایجاد حساب کاربری باید توسط مدیران تاییدشده تایید شوند تا دسترسی‌های خود را پیدا کنند. همچنین برای احراز هویت هر کدام از مدیران و دروازه‌ها تدبیری اندیشیده‌ایم که در قسمت‌های

جدول ۴-۲: شمای جدول دروازه‌ها

توضیحات	دروازه
ستون	شناسه
عددی یکتا برای هر دروازه	شناسه
آدرس فیزیکی یکتا منسوب به هر دروازه	آدرس فیزیکی
رشته‌ای هش‌شده برای ورود هر دروازه به سیستم	رمز عبور
متغیری صفر و یکی	تاییدشده
زمان ثبتنام دروازه	ثبتنام در
زمان آخرین ورود دروازه به سیستم	آخرین ورود در
زمان تایید دروازه توسط یک مدیر تاییدشده	تایید شده در

زیر به آنها اشاره خواهیم کرد.

#### ۱-۴-۴ احراز هویّت

برای شناسایی و احراز هویّت کاربران این سیستم از استاندارد JWT<sup>۵</sup> بهره برده‌ایم. از این استاندارد برای ایجاد داده‌ی رمزشده با یک امضا<sup>۶</sup> مشخص استفاده می‌شود. این نشانه‌ها یا توسط یک رمز خصوصی یا با کلید عمومی خصوصی امضا می‌شوند. هر نشانه شامل سه بخش سرآمد<sup>۷</sup>، بدنی اصلی<sup>۸</sup> و امضا می‌باشد<sup>۹</sup>[۲۸]. نکته‌ی قابل توجه این است که هر گونه تغییر در بدنی اصلی نشانه، امضا نشانه را بهم می‌زند و پیام ارزش قانونی خود را از دست خواهد داد. در واقع کارگزار اصلی پس از هر بار دریافت نشانه از سمت کاربران، بررسی می‌کند که آیا خود این پیام را امضا کرده است یا خیر. اگر این پیام توسط خودش امضا نشده باشد، درخواست کاربر را بررسی نخواهد کرد.

#### سرآمد

در سرآمد نشانه، اطلاعات مربوط به الگوریتم استفاده شده برای ایجاد امضا آورده می‌شود. در قسمت زیر نمونه‌ای از سرآمد یک نشانه‌ی JWT را مشاهده می‌کنیم. همانطور که مشخص است، از الگوریتم HMAC-SHA256 برای امضا بدنی اصلی این پیام استفاده شده است.

```

1 {  

2   "alg": "HS256",

```

<sup>5</sup>JSON Web Token

<sup>6</sup>Signature

<sup>7</sup>Header

<sup>8</sup>Payload

```

3   "typ" : "JWT"
4 }
```

### بدنه‌ی اصلی

این قسمت شامل تعدادی از ادعاهاست که فرد دارنده‌ی نشانه می‌تواند از آن استفاده کند. در قسمت زیر نمونه‌ای از یک نشانه‌ی صادر شده برای یکی از مدیران و زمان انقضای این نشانه را مشاهده می‌کنیم. نکته‌ی قابل توجه این است که پس از زمان نشانداده شده، نشانه دیگر کارایی ندارد و درخواست‌هایی که شامل این نشانه هستند، بررسی نخواهند شد.

```

1 {
2   "email" : "user@example.com",
3   "exp" : 1689956613
4 }
```

### امضا

این قسمت بصورت امن، قانونی بودن نشانه‌ی صادر شده را تضمین می‌کند. این امضا، توسط رمزشدن رشته‌ی تغییریافته توسط Base64 سرآمد و بدنه‌ی اصلی با کمک الگوریتم مشخص شده در سرآمد و با یک رمز خصوصی در سمت کارگزار تولید می‌شود. هنگامی که کارگزار نشانه‌ای را دریافت می‌کند، ابتدا امضای آن را با امضای صادرشده برای آن مقایسه می‌کند و در صورت یکی بودن آنها درخواست بررسی خواهد شد.

## ۴-۴-۲ تایید مدیران و دروازه‌ها

هر یک از مدیران و دروازه‌های تازه ثبت‌نام کرده باید توسط یکی از مدیران تایید و احرار هویت‌شده، تایید شوند. پس از تایید، مدیران می‌توانند به نتایج تحلیل‌ها و داده‌های بدست‌آمده توسط مدل یادگیری ماشین دسترسی پیدا کنند. از طرفی دروازه‌ها نیز پس از این عمل، توانایی ارسال داده‌های جمع‌آوری شده توسط خود را پیدا خواهند کرد.

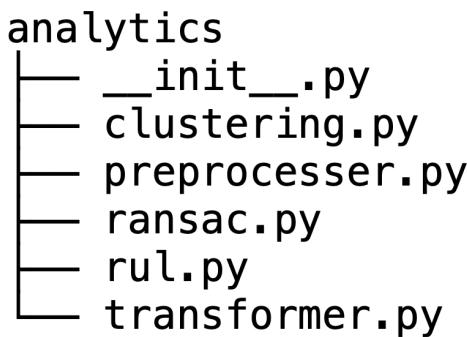
## ۵-۴ جمع‌بندی و نتیجه‌گیری

در این بخش از نوشتة، با معماری کلی سامانه‌ی طراحی شده برای اعمال نگهداری پیش‌بینانه روی گره‌های اینترنت اشیاء آشنا شدیم. همانطور که دیدیم، این سیستم طبق معماری مایکروسرویس، شامل چندین بخش مجزا پیاده‌شده است. همچنین دیدیم که الگوی بکاررفته برای طراحی کارگزار الگوی مدل-نماینترل گر بود و نتایج درخواست‌ها طبق این الگو تحويل کاربران داده می‌شوند. در مرحله‌ی آخر نیز دو سرویس پایگاه داده‌ی رابطه‌ای و امنیت را تشریح کردیم. در فصل بعد، مهم‌ترین مؤلفه‌ی سیستم کنونی، یعنی سرویس یادگیری ماشین را تجزیه و تحلیل خواهیم کرد.

## فصل پنجم

### پیاده‌سازی و توسعه مدل یادگیری ماشین

در این فصل روش پیاده‌سازی مدل هوش مصنوعی را به تفصیل شرح خواهیم داد. ابتدا فرضیات و داده‌های ورودی و آماده‌ی تحلیل را مشخص کرده و نماد هر کدام را که تا انتهای این نوشه از آنها استفاده خواهیم کرد، مشخص می‌کنیم. در قسمت بعد مراحل پیش‌پردازش<sup>۱</sup> را که روی این داده‌ها انجام می‌شود به ترتیب توضیح می‌دهیم و سپس ویژگی‌هایی که نیاز داریم از این داده‌های خام در بیاوریم را توضیح می‌دهیم و نحوه‌ی استخراج این ویژگی‌ها را نمایان می‌کنیم. در مرحله‌ی نهایی نحوه‌ی یادگیری مدل هوش مصنوعی و پیش‌بینی عمر باقی‌مانده‌ی دستگاه‌ها را بر اساس این ویژگی‌ها شرح می‌دهیم. در **شکل ۱-۵**، قالب بسته‌ی هوش مصنوعی توسعه داده‌شده برای کارگزار اصلی را مشاهده می‌کنید که در این فصل به توضیح بخش‌های مختلف آن می‌پردازیم.



شکل ۱-۵: ساختار کلی بسته‌ی هوش مصنوعی

## ۱-۵ توضیح مسئله

با توجه به اینکه سیستم یادگیری ماشین بر اساس اطلاعات حسگرهای لرزش عمل می‌کند، برای داشتن کمترین خطا در عملیات پیش‌بینی باید فرضیاتی را پیش از طراحی و پیاده‌سازی سیستم در نظر داشته باشیم. اولاً نمونه‌های بدبخت آمده برای حسگرهای متفاوت بازه‌های زمانی مختلف را در بر می‌گیرند و همگن نیستند. ثانیاً این داده‌های دارای انحرافاتی در اندازه‌گیری بدليل وجود گرانش یا خرابی حسگر هستند. ثالثاً وضعیت ابتدایی هر یک از گره‌هایی که می‌خواهیم اطلاعات لرزش آنها را جمع‌آوری و تحلیل کنیم یکی نیستند<sup>۴</sup>. با توجه به نکاتی که مطرح کردیم، پیاده‌کردن یک سیستم پیش‌پردازش و استخراج‌کننده‌ی ویژگی‌های مناسب، الزامی است.

در **جدول ۱-۵** توضیحات نشانه‌گذاری داده‌ی مربوط به این مسئله را می‌بینیم. همچنین در جهت مشخص‌کردن محدوده‌ی کاری این مسئله، از سه برچسب که در **جدول ۲-۵** مشخص شده‌اند، برای

<sup>1</sup>Preprocess

جدول ۵-۱: توضیحات نشانه‌گذاری داده‌ها

نشانه	توضیحات
$N$	تعداد کل گره‌ها
$M$	تعداد کل اندازه‌گیری‌ها
$K$	تعداد کل نمونه‌های یک اندازه‌گیری
$n$	گره $n$ ام
$m$	اندازه‌گیری $m$ ام
$k$	نمونه‌ی $k$ ام یک اندازه‌گیری
$a_{nmk}$	بردار سه‌بعدی مربوط به اندازه‌گیری لرزش
$a_{nm}^l$	بردار $k$ بعدی مربوط به لرزش در محور $\{x, y, z\}$

جدول ۵-۲: برچسب‌های استفاده شده برای تعیین وضعیت دستگاه‌ها

برچسب	توضیحات
$A$	دستگاه‌های نو که تازه تولید شده‌اند و آماده استفاده‌اند
$B, C$	دستگاه‌هایی که نیستند ولی هنوز مشغول کارکردن هستند
$D$	دستگاه‌هایی خراب شده‌اند یا در حال خرابی‌اند

تعیین کردن وضعیت گره‌های موجود استفاده می‌کنیم.

## ۲-۵ پیش‌پردازش

این بخش وظیفه دارد قبل از انجام تحلیل داده، در ابتدا انحرافات و داده‌های پرت<sup>۲</sup> را از داده‌ی خام جدا کرده و داده‌ی قابل پردازش را به لایه‌ی بعد که لایه‌ی استخراج ویژگی است تحويل دهد. در نهایت خروجی بخش پیش‌پردازنه، ویژگی‌هایی هستند که دستگاه یادگیری ماشین با تحلیل و بررسی آنها عملیات یادگیری و پیش‌بینی را انجام خواهد داد.

## ۱-۲-۵ از بین بردن انحرافات

حسگرهای کم‌هزینه MEMS که داده‌های جمع‌آوری‌شده برای این پروژه توسط این نوع از حسگرهای تأمین شده است، غالباً با گذشت زمان دچار انحرافاتی در اندازه‌گیری خواهند شد که منجر به اضافه یا کم‌شدن یک مقدار ثابت غیر صفر در اندازه‌گیری‌هایشان خواهد شد. از طرفی وجود گرانش، تاثیراتی روی اندازه‌گیری‌ها خواهد داشت و موجب ایجاد انحرافاتی رو به بالا یا پایین در این مقادیر خواهد شد.<sup>[۴]</sup>

<sup>2</sup>Outlier Data

برای از بین بردن این مشکل همانطور که در برابری (۱-۵) آورده شده است [۲۹]، از هنجار کردن<sup>۳</sup> داده با کم کردن میانگین مقادیر شتاب اندازه گیری شده در هر کدام از سه محور از مقادیر اندازه گیری شده استفاده کردہ ایم. لازم به ذکر است همانطور که مشخص است،  $\hat{a}_{nm}^l$  نماد ماتریس هنجار شده است.

$$\hat{a}_{nm}^l = a_{nm}^l - \sum_{k=1}^K \frac{a_{nmk}^l}{K} \quad (1-5)$$

## ۲-۲-۵ از بین داده های پرت

در سیستم طراحی شده برای جمع آوری اطلاعات، ممکن است که تعدادی از حسگرهای دچار مشکل شده باشند و داده ای که تحويل دروازه می دهند دقیق و در راستای داده های از قبل جمع آوری شده نباشد. بطور کلی، پایدار بودن میانگین شتاب دریافت شده از هر اندازه گیری، معیار خوبی برای تشخیص صحت و درستی اطلاعات جمع آوری شده است [۴]. به عبارتی دیگر، میانگین لرزش های اندازه گرفته شده نباید به طور ناگهانی بالا یا پایین روند و باید در همان حدود اندازه گیری های قبل باشند. برای جدا کردن این داده ها که اصطلاحاً به آنها داده های پرت می گوییم، پیش از شروع یادگیری مدل، ابتدا میانگین شتاب جمع آوری شده برای هر اندازه گیری موجود را در هر سه بعد حساب کرده و سپس با کمک یک الگوریتم خوش بندی<sup>۴</sup>، این داده ها را جدا می کنیم.

برای انجام عملیات تشخیص داده های پرت، از الگوریتم خوش بندی میانگین تغییر<sup>۵</sup> استفاده کردہ ایم که الگوریتمی بر اساس تخمین تراکم هسته<sup>۶</sup> می باشد. در شکل ۲-۵ [۳۰] نمونه ای از یک الگوریتم خوش بندی تراکم محور آورده شده است. روند کار این الگوریتم بدین صورت است که به ازای ورودی بصورت نقاط و پارامتر ورودی پهنه ای باند<sup>۷</sup>، الگوریتم بطور مکرر هر نقطه داده را به نزدیک ترین مرکز خوش اختصاص می دهد و جهت نزدیک ترین مرکز خوش بر اساس جایی که اکثر نقاط نزدیک در آن قرار دارند تعیین می شود. در هر بار تکرار، هر نقطه داده به جایی که بیشترین نقاط در آن قرار دارد، نزدیک تر می شود، که در نهایت به مرکز خوش منجر خواهد شد. هنگامی که الگوریتم متوقف می شود، هر نقطه به یک خوش اختصاص داده می شود. همانطور که گفته شد، این الگوریتم بغیر از پهنه ای باند به پارامتر دیگری نیاز ندارد. این امر سبب می شود که مواردی همانند تعداد و مراکز هر خوش، توسط خود

<sup>3</sup>Normalizing

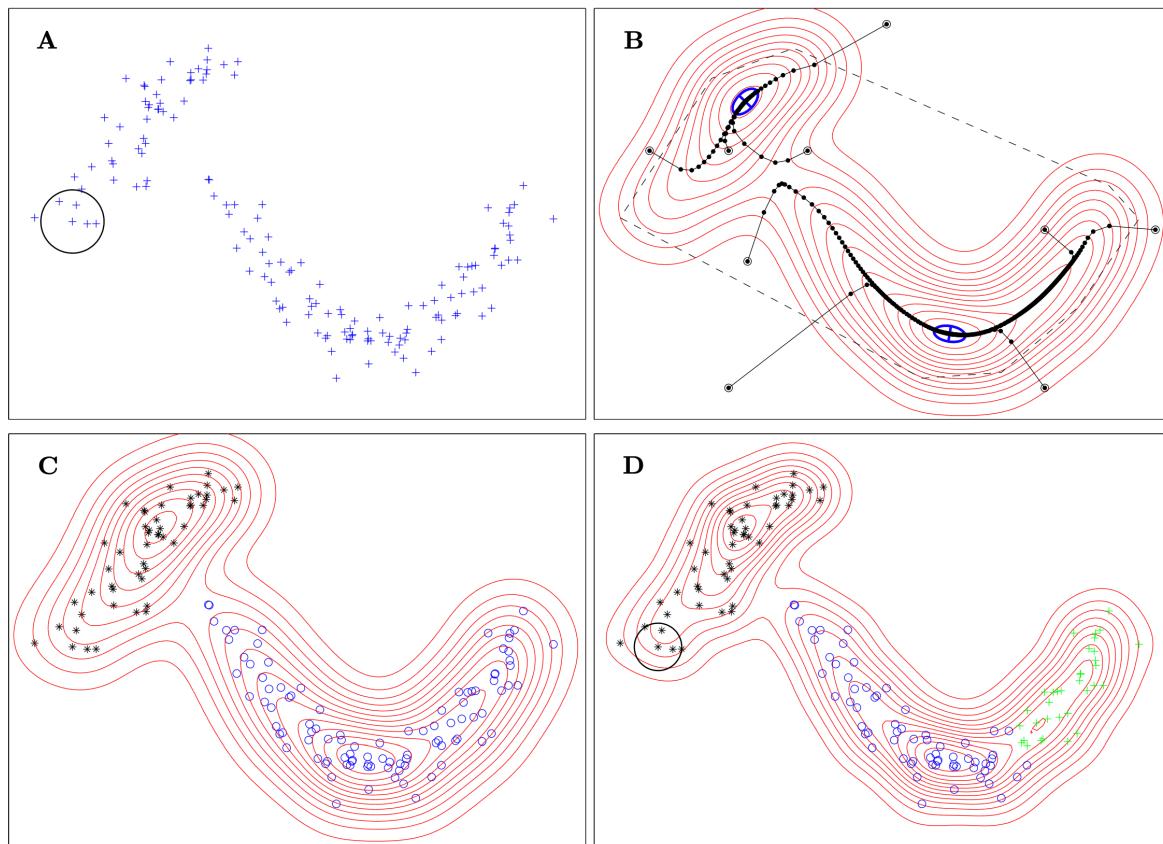
<sup>4</sup>Clustering

<sup>5</sup>Mean Shift

<sup>6</sup>Kernel Density Estimate

<sup>7</sup>Bandwidth

الگوریتم مشخص شوند [۳۰].



شکل ۲-۵: روند خوشبندی یک الگوریتم تراکم‌محور [۳۰]

نحوه‌ی استفاده از این الگوریتم در این مسئله بدین گونه است که پیش از عملیات استخراج ویژگی‌ها و یادگیری ماشین، میانگین مقادیر لرزش در هر سه بعد برای هر اندازه‌گیری محاسبه می‌شود. این مقادیر نباید خیلی با هم اختلاف داشته باشند. برای تشخیص داده‌های پرت، خوشبندی میانگین تغییر سه‌بعدی استفاده می‌کنیم و در نهایت داده‌هایی که در خوشبندی اقلیت قرار دارند را برای محاسبات و یادگیری ماشین در نظر نمی‌گیریم [۴]. برای پیاده‌سازی این الگوریتم کلاس MeanShiftClustering در فایل clustering.py پیاده‌شده است. این کلاس از کلاس MeanShift از کتابخانه Scikit-Learn از کتابخانه ارثبری می‌کند.

### ۳-۲-۵ استخراج ویژگی‌ها

تا اینجای کار، اثرهای انحرافات ممکن را از بین بردیم و داده‌های پرت را از میان کل داده‌ها جدا کردیم. از آنجا که داده‌ی خام لرزش گره‌های موجود در شبکه‌ی اشیاء، در دامنه‌ی زمانی<sup>۸</sup> بوده و حالت شروع به کار و وضعیت فعلی هر کدام از آنها در حال حاضر با همدیگر متفاوت است، نیازمند آنیم که از این داده‌های خام، ویژگی‌هایی مناسب را جهت انجام تحلیل و یادگیری ماشین، استخراج کنیم؛ برای این منظور بردن داده‌های موجود در دامنه زمانی به دامنه فرکانسی با کمک تبدیل فوریه<sup>۹</sup>، شروع خوبی است.

#### Root Mean Square (RMS) ویژگی

ویژگی مربع میانگین ریشه یا به اختصار RMS تنها بزرگی اندازه‌ی لرزش‌های اندازه‌گرفته شده را نمایان می‌کند و برای بردارهای لرزش هر اندازه‌گیری منسوب به هر دستگاه اینترنت اشیاء موجود، به صورت برابر (۳-۵) محاسبه می‌شود. در شکل ۳-۵ مقادیر محاسبه شده‌ی این ویژگی را برای همه‌ی اندازه‌گیری‌های یک گره موجود حساب کرده‌ایم. محور افقی شناسه‌ی هر یک از اندازه‌گیری‌ها می‌باشد.

$$\begin{aligned} r_{nm}^l &= \frac{1}{\sqrt{K}} \|a_{nm}^l\| \\ r_{nm}^2 &= \sum_{l \in \{x,y,z\}} (r_{nm}^l)^2 \end{aligned} \quad (3-5)$$

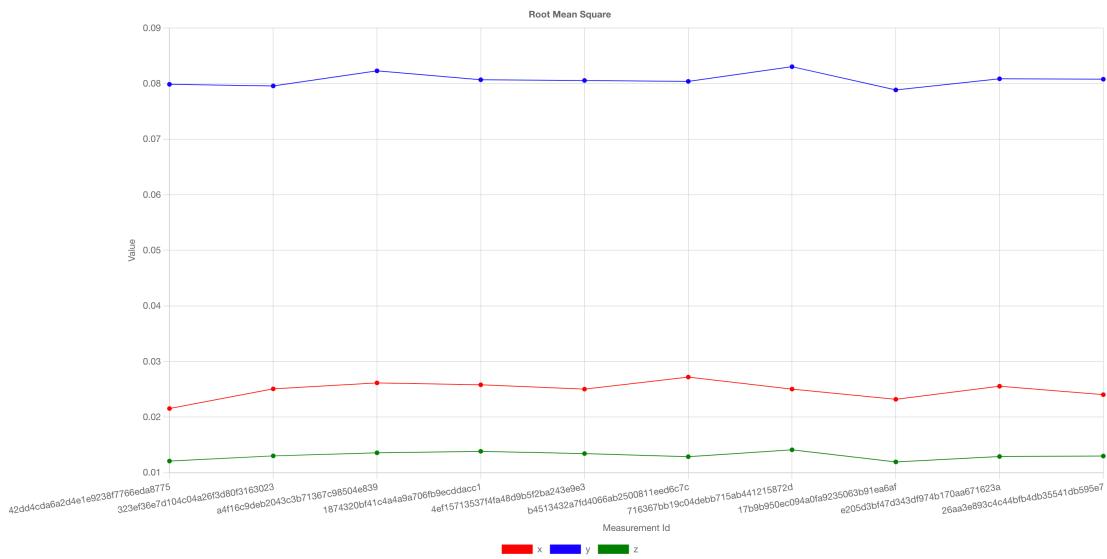
همانطور که گفته شد این ویژگی اطلاعات زیادی را در اختیار ما قرار نمی‌دهد. نکته‌ی قابل توجه دیگر در مورد این ویژگی این است که مقادیر بدست آمده در این ویژگی منسوب به دامنه زمانی می‌باشد و با توجه به مسئله‌ی کنونی، این ویژگی کاربردی در تحلیل اطلاعات لرزش برای ما نخواهد داشت.

#### Power Spectral Density (PSD) ویژگی

ویژگی چگالی طیفی توان یا PSD به ما در یافتن مشخصه‌های مبهم موجود در ویژگی‌های مربوط به دامنه زمانی کمک می‌کند. با ضرب ماتریسی سیگنال لرزش اندازه‌گرفته شده در بعد زمان در یک

<sup>8</sup>Time Domain

<sup>9</sup>Fourier Transform



شکل ۳-۵: مقادیر ویژگی مربع میانگین ریشه برای همه اندازه‌گیری‌های یک گره

ماتریس تبدیل کسینوسی گستته<sup>۱۰</sup> به ابعاد  $K \times K$ , می‌توان بردار ویژگی سیگنال مربوطه را در حوزه‌ی فرکانس بدست آورد. در برابری (۳-۵) طرز محاسبه‌ی این ویژگی را مشاهده می‌کنیم. نکته‌ی قابل توجه در این قسمت این است که بنابر قضیه‌ی پارسوال<sup>۱۱</sup>, برابری  $(r_{nm}^l)^2 = \sum_{k=1}^K s_{nmk}^l$  برقرار می‌باشد.

$$s_{nm}^l = \frac{1}{2K} (a_{nm}^l \times W_K)^2 \quad (3-5)$$

$$s_{nm} = \sum_{l \in \{x,y,z\}} s_{nm}^l$$

پس از این مرحله باید توجه کرد که ویژگی چگالی طیفی توان، یک ویژگی با ابعاد بالا (به اندازه‌ی تعداد نمونه‌ها در یک اندازه‌گیری که در مسئله‌ی ما این عدد برابر با ۶۰ می‌باشد) است که عموماً در هنگام محاسبات ( $s^T s$ ) منجر به ایجاد ماتریس منفرد<sup>۱۲</sup> خواهد شد و در نتیجه برای مسائل رگرسیون دچار مشکل خواهیم شد. از طرف دیگر، این ویژگی به دلیل نوسانات تصادفی زیاد در دامنه آنها در فرکانس بدلیل انحراف اندازه‌گیری ذاتی در سنسور MEMS غیرقابل اتکا است.

<sup>10</sup>Discrete Cosine Transform (DCT)

<sup>11</sup>Parseval's Theorem

<sup>12</sup>Singular Matrix

### Harmonic Peaks Feature ویژگی

برای رفع این مشکلات، ویژگی قله‌های موزون را معرفی می‌کنیم. برای هر اندازه‌گیری با ۶۰ نمونه، این ویژگی عبارت است از ۲۰ تا از بیشترین مقادیر ویژگی PSD به همراه فرکانس‌های متناظر با آنها. به عبارت دیگر، این ویژگی به صورت  $p_m = \{(f_{mk}, p_{mk})\}_{k=1, \dots, n_p}$  تعریف می‌شود که  $n_p$  در این مسئله برابر با ۲۰ است.

برای استخراج ویژگی قله‌های موزون، باید دو مرحله را طی کنیم.

- از بین بردن اثر انحرافات در اندازه‌گیری‌های ویژگی PSD با استفاده از عملیات پیچیدگی<sup>۱۳</sup> با پنجره‌ی هان<sup>۱۴</sup> که در برابری (۴-۵) آورده شده است (در این برابری،  $n_h$  اندازه‌ی پنچره است که در مسئله‌ی ما ۱۶ انتخاب شده است). پس از انجام این عملیات، سیگنال ویژگی‌ها هموار<sup>۱۵</sup> می‌شود و از این پس می‌توان عملیات جستجو برای بیشترین مقادیر را شروع کرد.

$$w_h(n) = 0.5(1 - \cos \frac{2\pi n}{n_h - 1}) \quad (4-5)$$

- پیدا کردن ۲۰ تا از بیشترین مقادیر ویژگی‌های هموارشده از طریق شناسایی نقاطی که مشتق اول سیگنال در آنها از مثبت به منفی، تغییر علامت می‌دهد در شکل ۴-۵ مقادیر ویژگی چگالی طیفی توان و ۲۰ تا از بیشترین قله‌های موزون را برای یکی از اندازه‌گیری‌های منسوب به یک گره، نمایش داده‌ایم.

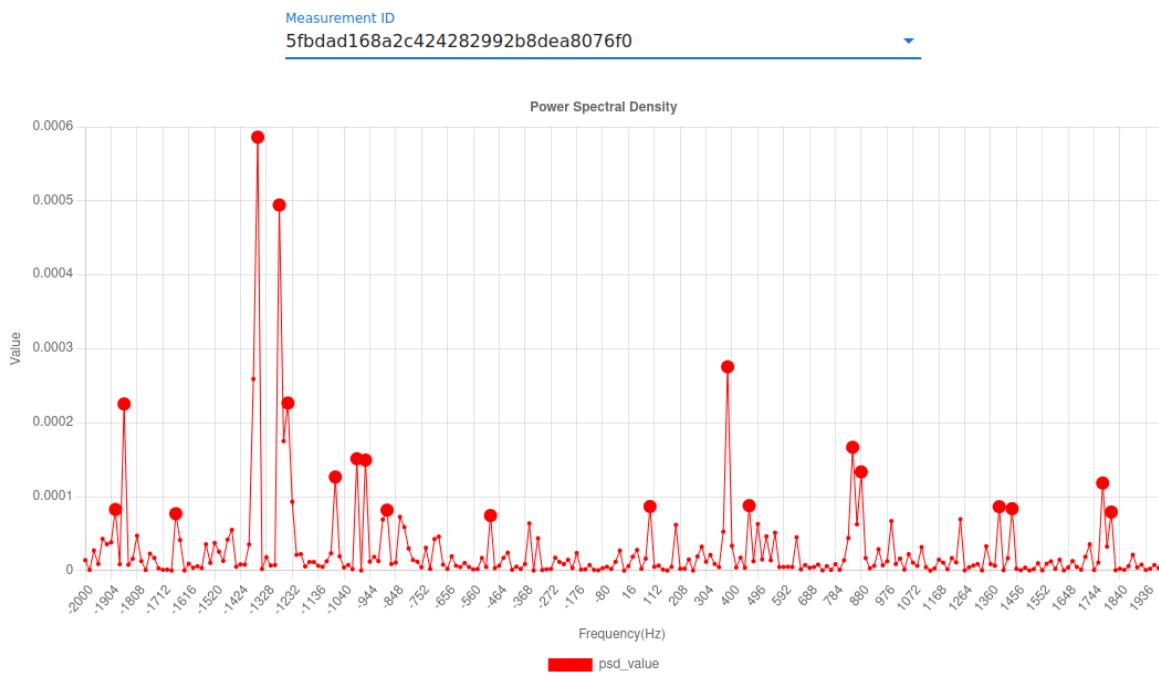
## ۳-۵ نحوه یادگیری مدل

پس از استخراج ویژگی‌ها مناسب، نیازمند آنیم که با تحلیل این ویژگی‌ها به آموزش مدل بپردازیم. برای رسیدن به این مهم باید ابتدا روش مناسبی را برای مقایسه‌ی کمی بین ویژگی‌های مختلف پیدا کرد.

<sup>13</sup>Convolution

<sup>14</sup>Hann Window

<sup>15</sup>Smooth



شکل ۵-۴: مقادیر ویژگی چگالی طیفی توان و قله‌های موزون برای یک اندازه‌گیری از یک گره

### ۱-۳-۵ محاسبه‌ی مشابهت بین وضعیت دستگاه‌ها

در این مرحله، برای مشخص کردن میزان تفاوت بین دو ویژگی قله‌های موزون  $p_j$  و  $p_i$  الگوریتم ۱ را ارائه می‌دهیم. لازم به ذکر است که خروجی الگوریتم،  $D_{ij}$ ، میزان عددی تفاوت بین ویژگی‌ها است و هرچه این عدد از صفر دورتر باشد، ویژگی‌ها متفاوت‌تر هستند [۴].

### ۲-۳-۵ پیاده‌کردن مدل

پس از طراحی الگوریتم ۱ تحلیل ویژگی‌های بدست آمده برای هر اندازه‌گیری را شروع می‌کنیم. نکته‌ی قابل توجه در این الگوریتم این است که این رویه، میزان جریمه‌ی بیشتری را برای مقادیر بیشینه در فرکانس‌های بالاتر در نظر می‌گیرد و این دقیقاً همان چیزی است که به ما در شناسایی دستگاه‌های با کارکرد غیرعادی کمک می‌کند. از آنجا که این نوع دستگاه‌ها در فرکانس‌های بالاتر دچار انحرافات شدیدی هستند. در مقابل، دستگاه‌های با کارکرد عادی دارای مقادیر بیشینه‌ی بیشتر در فرکانس‌های پایین‌تر هستند.

با یک فرض پایه، توضیح نحوه‌ی یادگیری مدل را شروع می‌کنیم و آن این است که کلیه‌ی دستگاه‌ها به مرور زمان و با کار بیشتر، از حالت نویی دور می‌شوند. به عبارت دیگر، با گذشت زمان، هر دستگاه از وضعیت دستگاه تازه و آماده بکار دورتر می‌شود. برای این منظور، پارامتر  $D_a$  را برای تحلیل دستگاه‌ها

### الگوریتم ۱ تشخیص کمی میزان تفاوت دو ویژگی

**Require:**  $p_{max} \leftarrow max(p_n, p_m), f_{max} \leftarrow max(f_n, f_m) \forall (p_n, f_n) \in p_i, \forall (p_m, f_m) \in p_j$

**Ensure:**  $n_h = 16, n_p = 20$

$sum \leftarrow 0, cnt \leftarrow 0$

$p_n \leftarrow p_n/p_{max}, f_n \leftarrow f_n/f_{max}, \forall (p_n, f_n) \in p_i$

$p_m \leftarrow p_m/p_{max}, f_m \leftarrow f_m/f_{max}, \forall (p_m, f_m) \in p_j$

$queue_i \leftarrow copy(p_i)$

$queue_j \leftarrow copy(p_j)$

**while**  $queue_i$  is not empty **do**

$(f_i, p_i) \leftarrow queue_i.pop()$

$f_* \leftarrow$  do binary search for  $f_i$  in  $[f_{j1} \dots f_{j20}]$

**if**  $|f_i - f_{j*}| \times f_{max} < n_h$  **then**

$(f_{j*}, p_{j*}) \leftarrow queue_j.pop(f_{j*})$

$dist \leftarrow dist + \|(f_i, p_i) - (f_{j*}, p_{j*})\|$

**else**

$dist \leftarrow \|(f_i, p_i)\|$

**end if**

$sum \leftarrow sum + dist$

$cnt \leftarrow cnt + 1$

**end while**

$D_{ij} \leftarrow (sum + \sum_{k=1}^{20} p_{jk}) / (cnt + len(p_j))$

معرفی می‌کنیم که عبارت است از میزان متفاوت بودن دستگاه مورد نظر با یک دستگاه نو (کلاس کاری در جدول ۲-۵). تا زمانیکه هیچ نگهداری ای برای دستگاه صورت نگیرد،  $D_a$  به مرور زمان بصورت یکنواخت افزایش می‌یابد.

### RANSAC

برای یادگیری این مدل خطی و پیش‌بینی زمان سرویس دستگاه‌ها از رویکرد اجماع نمونه‌ی تصادفی<sup>۱۶</sup> یا به اختصار RANSAC استفاده می‌کنیم. این الگوریتم، بصورت بازگشتی، مدل خطی‌ای بر اساس داده‌های آموزشی درست می‌کند و نکته‌ی قابل توجه این الگوریتم این است که توانایی تشخیص و در نظرنگرفتن داده‌های پرت را دارد و از این رو بسیار مناسب استفاده در این مسئله است. بطور کلی روند ایجاد مدل در این الگوریتم به صورت زیر است [۳۱]:

۱. بصورت تصادفی حداقل تعداد نقاط لازم، برای پیدا کردن پارامترهای مدل مشخص می‌شود.

۲. پارامترهای مدل بدست آورده می‌شوند.

<sup>16</sup>RANdom SAmples Consensus (RANSAC)

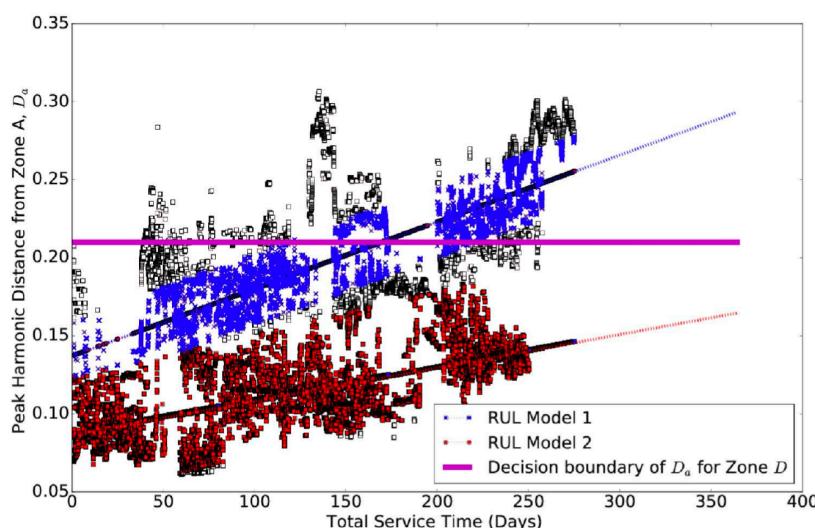
۳. از میان کل نقاط، تعداد نقاطی که با میزان تحمل از قبل تعیین شده‌ی  $\epsilon$  شامل مدل هستند، حساب می‌شوند.

۴. اگر نسبت نقاط مشمول در مدل به تعداد کل نقاط از یک مقدار از پیش تعیین شده‌ی  $\tau$  بیشتر شد، پارامترهای مدل دوباره با کمک نقاط مشمول حدس زده می‌شوند و الگوریتم پایان می‌یابد.

۵. در غیر این صورت، مرحله‌ی ۱ تا ۴ تا حداً  $N$  بار تکرار می‌شود.

جهت پیش‌بینی میزان عمر مفید باقی‌مانده، الگوریتم ما ابتدا آستانه‌ی  $D_a$  را برای دو کلاس کاری  $B, C$  و  $D$  حساب می‌کند. این آستانه از طریق کمینه‌کردن میزان خطا در طبقه‌بندی دستگاه‌های این دو کلاس مشخص می‌شود. پس از تعیین این آستانه، الگوریتم با بررسی مدل خطی بدست‌آمده از RANSAC، میزان عمر مفید دستگاه مربوطه را خروجی می‌دهد.

در شکل ۵-۵ [۴] دو مدل برای پیش‌بینی میزان عمر مفید باقی‌مانده، آموزش داده شده است. محور افقی زمان اندازه‌گیری‌ها بر حسب روز و محور عمودی فاصله‌ی ویژگی‌های دستگاه‌ها با ویژگی دستگاه‌های سالم و نو است. همچنین خط بنفسرنگ، آستانه‌ی ورود دستگاه از کلاس دستگاه‌های سالم به کلاس دستگاه‌های معیوب می‌باشد (این مقدار آستانه در شکل برابر با  $21/0$  است). برای پیش‌بینی عمر باقی‌مانده‌ی هر دستگاه ابتدا میزان تعلق مقادیر  $D_a$  برای آن دستگاه در دو مدل محاسبه می‌شود. سپس این حد آستانه با مدلی که دستگاه بیشترین تعلق را به آن دارد تلاقی داده می‌شود. در نهایت فاصله‌ی زمان اندازه‌گیری با نقطه‌ی تلاقی، برابر با عمر مفید باقی‌مانده‌ی دستگاه مورد نظر بر حسب روز است.



شکل ۵-۵: دو مدل یادگیری ماشین برای مسئله‌ی پیش‌بینی میزان عمر مفید باقی‌مانده [۴]

## ۴-۵ جمع‌بندی و نتیجه‌گیری

این بخش به طور دقیق روند یادگیری ماشین از اطلاعات لرزش گره‌ها را نشان داد. مشخص شد که داده‌های جمع‌آوری شده ابتدا به کمک پیش‌پردازنده که شامل هنجارکننده، تشخیص‌دهنده‌ی داده‌ی پرت و استخراج‌کننده‌ی ویژگی‌های آماده پردازش است، پالایش شده و سپس تحویل واحد یادگیری ماشین می‌شود. در این مرحله این واحد ابتدا با هموارکردن و سپس با انتخاب‌کردن ۲۰ تا از بیشترین مقادیر موزون ویژگی PSD برای هر اندازه‌گیری، اقدام به محاسبه‌ی میزان شباهت این اندازه‌گیری‌ها به اندازه‌گیری‌های یک دستگاه سالم می‌کند و بنا به میزان شباهت یا تفاوت با آن، پیش‌بینی مربوط به طول عمر باقی‌مانده‌ی گره را خروجی خواهد داد.

## فصل ششم

### جمع‌بندی، نتیجه‌گیری و پیشنهادات

در بخش انتهایی این نوشه، ابتدا به توضیح کلی پروژه و نکاتی که در هر بخش به آن پرداختیم، اشاره می‌کنیم. در قسمت بعد، چالش‌هایی که در مسیر توسعه‌ی سیستم نگهداری پیش‌بینانه به آنها برخورده‌یم را عنوان می‌کنیم. سپس، محدودیت‌هایی که خواهانخواه بر روند تکمیل پروژه تاثیراتی گذاشتند را بیان می‌کنیم. در نهایت پیشنهاداتی را بهبود پروژه کنونی نام می‌بریم.

## ۱-۶ جمع‌بندی و نتیجه‌گیری

در این پروژه سعی بر این شد که سیستمی برای نگهداری و تشخیص خرابی گره‌های موجود در یک شبکه‌ی اشیاء بر اساس [۴] توسعه یابد. هدف این سیستم پیش‌بینی زمان خرابی دستگاه‌های مختلف بر اساس تحلیل نحوه‌ی لرزش موتور آنها بود.

ابتدا در فصل ۲ تمام تکنولوژی‌ها و ابزارهایی که برای پیاده‌سازی این سیستم از آنها استفاده شده است را عنوان کردیم و همه‌ی آنها را با جزئیات توضیح دادیم.

در فصل ۳ انواع روش‌های موجود برای استقرار مدل یادگیری ماشین و استفاده از آن شرح داده شدند و بنابر نیازمندی‌های پروژه و ماهیت استفاده از مدل یادگیری ماشین برای مسئله‌ی پیش‌بینی میزان عمر مفید باقی‌مانده، استقرار بی‌درنگ را انتخاب کردیم.

در فصل ۴ نیز نحوه‌ی توسعه و معماری کارگزاری که برای دریافت درخواست‌های کاربران و همچنین ارسال پاسخ‌های تحلیل‌های سیستم پیاده شده بود را شرح دادیم. سپس در فصل ۵ نحوه‌ی توسعه‌ی مدل یادگیری ماشین بر اساس ویژگی‌های بدست‌آمده از داده‌های خام لرزش گره‌های موجود در شبکه‌ی اشیاء توضیح داده شد. در نهایت یک مدل خطی برای پیش‌بینی میزان عمر مفید باقی‌مانده بر اساس مقدار متفاوت‌بودن آن دستگاه نسبت به یک دستگاه سالم پیشنهاد داده شد.

## ۲-۶ چالش‌ها

در مراحل مختلف این پروژه، چالش‌هایی سر راه ما قرار گرفتند که برای رفع هر کدام تدبیری اندیشیدیم. در این قسمت به تعدادی از این موارد اشاره می‌کنیم و روش حل هر کدام را توضیح می‌دهیم.

- اولین مورد انتخاب روش مناسب استقرار سیستم یادگیری ماشین بود. همانطور که در فصل سوم توضیح دادیم، روش‌های مختلفی برای استقرار مدل‌های یادگیری ماشین وجود دارند که بنا به انتخاب هر کدام، نحوه‌ی شروع یادگیری، نحوه‌ی ذخیره و ارسال نتایج پیش‌بینی و روش

بروزرسانی مدل یادگیری ماشین متفاوت است. این مورد باید حتماً پیش از شروع توسعه‌ی مدل یادگیری ماشین مشخص می‌شد تا برای مواردی که بیان شد، بهترین روش را برگزینیم.

- نحوه‌ی هموار کردن و استفاده از پارامترهای مناسب پنجره‌ی هان نیز یکی از موارد چالش‌برانگیز دیگر بود. این پارامترها باید با توجه به تعداد نمونه‌های موجود در یک اندازه‌گیری طوری انتخاب شود که سیگنال خیلی هموار نشود به نحوی که ویژگی ذاتی خود را از دست بدهد. با انتخاب حالت‌های مختلف، مناسب‌ترین گزینه انتخاب شد.
- پیاده‌سازی الگوریتم مقایسه‌ی بین ویژگی‌ها یکی از مسائل دشوار بود. زیرا باید هنگام پیاده‌سازی به ماهیت سیگنال ویژگی PSD و همچنین آهنگ لرزش دستگاه‌ها توجه می‌شد و بر این اساس، الگوریتمی پیشنهاد می‌شد تا به درست‌ترین نحو ممکن اختلاف بین ویژگی‌ها را شناسایی کند.

## ۳-۶ محدودیت‌ها

یکی از بزرگ‌ترین محدودیت‌ها در طی انجام این پروژه نبود مجموعه‌ی داده‌ی مناسب و برچسب‌دار برای استفاده‌ی مدل یادگیری ماشین بود. از آنجا که مسئله‌ی پیش بینی عمر مفید باقی‌مانده‌ی هر دستگاه، هم یک مسئله‌ی طبقه‌بندی (باید کلاس کاری دستگاه مشخص شود) و هم یک مسئله‌ی رگرسیون (عمر باقی‌مانده‌ی دستگاه باید مشخص شود) می‌باشد، نیازمند دو مجموعه داده‌ی کامل برچسب‌دار بودیم که فراهم نشد. یکی از مهم‌ترین دلایلی که مجموعه داده‌ی آماده برای این نوع مسئله در دسترس عموم قرار ندارد این است که اکثر پروژه‌هایی که در این باب انجام شده‌اند صنعتی بوده و رویکرد هر کدام و نحوه‌ی طبقه‌بندی و پارامترهای دخیل (لرزش، دما، رطوبت، صدا، ارتفاع از سطح دریا) در تحلیل داده برای هر کدام متفاوت است.

## ۴-۶ پیشنهادات

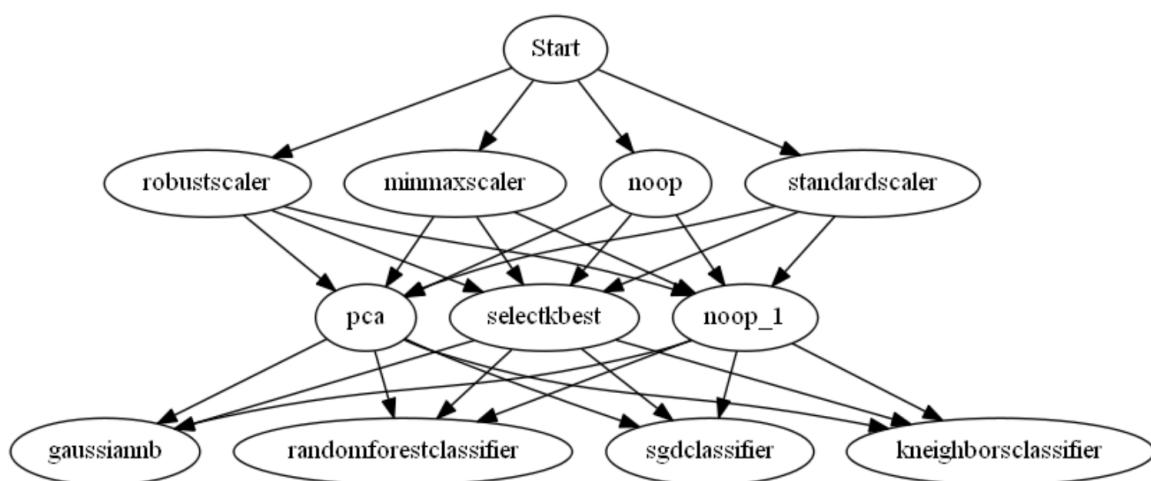
پروژه‌ی کنونی قابلیت توسعه و افزایش کارایی را در بخش‌های مختلفی اعم از میکروسرویس‌های توسعه‌ی یافته برای کارگزار اصلی، مدل هوش مصنوعی هم از لحاظ عملکرد و هم از لحاظ سرعت و همچنین اضافه کردن قابلیت تشخیص نوع خرابی را دارد. در بخش‌های زیر پیشنهاداتی را برای کارهای آینده و مرتبط با این پروژه آورده‌ایم.

## پیاده‌کردن سیستم ارسال گزارش و هشدار به مدیران

با افزایش تعداد گره‌های موجود در سیستم، کار نظارت درست بر این گره‌ها برای مدیران سخت می‌شود. از طرفی ارائه‌ی گزارشی دقیق از عملکرد دستگاه‌های مختلف هم به مدیران و هم به سازندگان قطعات می‌تواند مفید باشد. همچنین پیاده‌سازی بخشی برای ارسال هشدار به مدیران جهت اطلاع دادن از خرابی قریب‌الوقوع یک دستگاه می‌تواند مدیر را در پیش‌بینی تدابیر مناسب کمک کند.

## تشخیص نوع خرابی

یکی دیگر از جنبه‌های نگهداری که می‌تواند سود بسیار زیادی را برای صنعت‌گران داشته باشد، تشخیص نوع خرابی در کنار پیش‌بینی زمان خرابی دستگاه‌ها است. سیستمی موسوم به دکتر ماشین‌ها در [۳۲] پیاده‌شده است که بسته به سیستم‌های مختلف پارامترهای مختلفی را در نظر می‌گیرد و در یادگیری مدل بر اساس ویژگی‌های دستگاه‌ها، وزن پارامترها برای هر کدام از دستگاه‌ها فرق می‌کند. پیاده‌سازی چنین سیستمی قاعده‌تا نیازمند قدرت پردازشی بسیار بالا در کنار مجموعه داده‌های دقیق و مورد تایید متخصصان است. نمونه‌ای از نحوه‌ی طبقه‌بندی چنین سیستمی در [شکل ۱-۶](#) [۳۲] آورده شده است.



شکل ۱-۶: نحوه‌ی طبقه‌بندی مدل یادگیری ماشین بر اساس پارامترهای مختلف [۳۲]

## افزایش دقت پیش‌بینی

رویکردی که در این پژوهه از آن استفاده شد تنها لرزش موتور سیستم را در سه بعد در نظر می‌گیرد. یکی از بهبودهایی که می‌توان برای نتایج پیش‌بینی در این پژوهه متصور شد، در نظر گرفتن پارامترهای

دیگری اعم از دما، شرکت سازنده و رطوبت در کنار لرزش است تا دقیق پیش‌بینی‌ها برای میزان عمر مفید باقی‌مانده افزایش یابد.

## منابع و مراجع

- [1] Zhao, Jingyi, Gao, Chunhai, and Tang, Tao. A review of sustainable maintenance strategies for single component and multicomponent equipment. *Sustainability*, 14(5):2992, 2022.
- [2] Ran, Yongyi, Zhou, Xin, Lin, Pengfeng, Wen, Yonggang, and Deng, Ruilong. A survey of predictive maintenance: Systems, purposes and approaches. *arXiv preprint arXiv:1912.07383*, 2019.
- [3] Zonta, Tiago, Da Costa, Cristiano André, da Rosa Righi, Rodrigo, de Lima, Miromar Jose, da Trindade, Eduardo Silveira, and Li, Guann Pyng. Predictive maintenance in the industry 4.0: A systematic literature review. *Computers & Industrial Engineering*, 150:106889, 2020.
- [4] Jung, Deokwoo, Zhang, Zhenjie, and Winslett, Marianne. Vibration analysis for iot enabled predictive maintenance. in *2017 ieee 33rd international conference on data engineering (icde)*, pp. 1271–1282. IEEE, 2017.
- [5] Tinga, Tiedo. Application of physical failure models to enable usage and load based maintenance. *Reliability engineering & system safety*, 95(10):1061–1075, 2010.
- [6] Wu, Sze-jung, Gebraeel, Nagi, Lawley, Mark A, and Yih, Yuehwern. A neural network integrated decision support system for condition-based optimal predic-

- tive maintenance policy. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37(2):226–236, 2007.
- [7] Kaiser, Kevin A and Gebraeel, Nagi Z. Predictive maintenance management using sensor-based degradation models. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 39(4):840–849, 2009.
- [8] Van Rossum, Guido et al. Python programming language. in *USENIX annual technical conference*, vol. 41, pp. 1–36. Santa Clara, CA, 2007.
- [9] Srinath, KR. Python—the fastest growing programming language. *International Research Journal of Engineering and Technology*, 4(12):354–357, 2017.
- [10] Sharma, Akshansh, Khan, Firoj, Sharma, Deepak, Gupta, Sunil, and Student, FY. Python: the programming language of future. *Int. J. Innovative Res. Technol*, 6(2):115–118, 2020.
- [11] FastAPI — fastapi.tiangolo.com. <https://fastapi.tiangolo.com/>. [Accessed 07-Apr-2023].
- [12] Van Der Walt, Stefan, Colbert, S Chris, and Varoquaux, Gael. The numpy array: a structure for efficient numerical computation. *Computing in science & engineering*, 13(2):22–30, 2011.
- [13] Harris, Charles R, Millman, K Jarrod, Van Der Walt, Stéfan J, Gommers, Ralf, Virtanen, Pauli, Cournapeau, David, Wieser, Eric, Taylor, Julian, Berg, Sebastian, Smith, Nathaniel J, et al. Array programming with numpy. *Nature*, 585(7825):357–362, 2020.
- [14] Hao, Jiangang and Ho, Tin Kam. Machine learning made easy: a review of scikit-learn package in python programming language. *Journal of Educational and Behavioral Statistics*, 44(3):348–361, 2019.

- [15] Kramer, Oliver and Kramer, Oliver. Scikit-learn. *Machine learning for evolution strategies*, pp. 45–53, 2016.
- [16] Jatana, Nishtha, Puri, Sahil, Ahuja, Mehak, Kathuria, Ishita, and Gosain, Disphant. A survey and comparison of relational and non-relational database. *International Journal of Engineering Research & Technology*, 1(6):1–5, 2012.
- [17] Group, PostgreSQL Global Development. PostgreSQL — postgresql.org. <https://www.postgresql.org/>. [Accessed 12-Apr-2023].
- [18] Anderson, Charles. Docker [software engineering]. *Ieee Software*, 32(3):102–c3, 2015.
- [19] Docker overview — docs.docker.com. <https://docs.docker.com/get-started/overview/>. [Accessed 12-Apr-2023].
- [20] Yadav, Anuj Kumar, Garg, ML, and Ritika. Docker containers versus virtual machine-based virtualization. in *Emerging Technologies in Data Mining and Information Security: Proceedings of IEMIS 2018, Volume 3*, pp. 141–150. Springer, 2019.
- [21] Youssef, Ahmed E. Exploring cloud computing services and applications. *Journal of Emerging Trends in Computing and Information Sciences*, 3(6):838–847, 2012.
- [22] Serrano, Nicolas, Gallardo, Gorka, and Hernantes, Josune. Infrastructure as a service and cloud technologies. *IEEE Software*, 32(2):30–36, 2015.
- [23] Deploying your Machine Learning models | Kaggle — kaggle.com. <https://www.kaggle.com/discussions/getting-started/382794>. [Accessed 03-Apr-2023].

- [24] Singh, Pramod. Deploy machine learning models to production. *Cham, Switzerland: Springer*, 2021.
- [25] Pacheco, Fannia, Exposito, Ernesto, Gineste, Mathieu, Baudoin, Cedric, and Aguilar, Jose. Towards the deployment of machine learning solutions in network traffic classification: A systematic survey. *IEEE Communications Surveys & Tutorials*, 21(2):1988–2014, 2018.
- [26] Thönes, Johannes. Microservices. *IEEE software*, 32(1):116–116, 2015.
- [27] Deacon, John. Model-view-controller (mvc) architecture. *Online]/Citado em: 10 de março de 2006.] http://www. jdl. co. uk/briefings/MVC. pdf*, 28, 2009.
- [28] Jones, Michael, Bradley, John, and Sakimura, Nat. Json web token (jwt). tech. rep., 2015.
- [29] García, Salvador, Luengo, Julián, and Herrera, Francisco. *Data preprocessing in data mining*, vol. 72. Springer, 2015.
- [30] Carreira-Perpinán, Miguel A. A review of mean-shift algorithms for clustering. *arXiv preprint arXiv:1503.00687*, 2015.
- [31] Derpanis, Konstantinos G. Overview of the ransac algorithm. *Image Rochester NY*, 4(1):2–3, 2010.
- [32] Patel, Dhaval, Zhou, Nianjun, Shrivastava, Shrey, and Kalagnanam, Jayant. Doctor for machines: a failure pattern analysis solution for industry 4.0. in *2020 IEEE International Conference on Big Data (Big Data)*, pp. 1614–1623. IEEE, 2020.

## پیوست

تمامی کدها و مستندات پروژه‌ی انجامشده به همراه همه‌ی فایل‌های مورد نیاز برای نوشتن مجموعه‌ی داده در پایگاه داده و همچنین استقرار پروژه در ابر و آموزش نحوه‌ی راهاندازی پروژه در مخزن زیر در گیتهاب قابل دسترسی است.

<https://github.com/mies47/Predictive-Maintenance-Server>

# **Abstract**

The success of the Internet of Things depends on our ability to solve challenging problems that were previously impossible. One of the most critical challenges in the IoT space is preventive maintenance in industrial manufacturing processes to maximize equipment availability and durability. Traditionally, preventive management has followed only lower-cost strategies, such as time-based or utilization-based management. With the large amount of sensor data collected from the Internet of Things, we can develop much smarter predictive maintenance based on the accurate prediction of machinery failure. In this project, we have developed a system consisting of a core server and a machine learning package based on vibration sensor data to implement predictive maintenance. More precisely, we advanced the process of building a machine learning model based on the pre-processing of sensor data, extracting suitable features and then analyzing these features. Finally, by learning a linear model based on the analysis and comparison of these features with healthy devices, we predicted the remaining useful time for each existing node. By using such a system in the industry, maintenance costs as well as possible losses incurred will be greatly reduced.

## **Key Words:**

Predictive Maintenance, Vibration Analysis, Machine Learning, Internet of Things



**Amirkabir University of Technology  
(Tehran Polytechnic)**

**Department of Computer Engineering**

**B. Sc. Thesis**

# **IoT-Enabled Predictive Maintenance System Based on Vibration Analysis**

**Author**

**Milad Esrafilian Najafabadi**

**Supervisor**

**Dr. Hamidreza Zarandi**

**July 2023**