دانشگاه صنعتی امیرکبیر

(پلی‌تکنیک تهران)

دانشکده مهندسی کامپیوتر

پروژه کارشناسی

گرایش معماری سیستم‌های کامپیوتری

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

نگارنده

آریان بوکانی

استاد راهنما

دکتر حمیدرضا زرندی

مرداد ۱۴۰۲


بسم الله الرحمن الرحيم

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

تعهدنامه اصالت اثر

تاریخ: مرداد ۱۴۰۲

اینجانب **آریان بوکانی** متعهد می‌شوم که مطالب مندرج در این پایان‌نامه حاصل کار پژوهشی اینجانب تحت نظارت و راهنمایی اساتید 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 مآخذ ذکر گردیده است. این پایان‌نامه قبلاً 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 پایان نامه بدون موافقت کتبی دانشگاه صنعتی امیرکبیر ممنوع است. نقل مطالب با ذکر مآخذ بلامانع است.

آریان بوکانی

امضا

تقدیم به مادر و پدر عزیزم

سپاس‌گزاری

از استاد بزرگوارم جناب آقای دکتر حمیدرضا زرندی که با حسن خلق و گشاده‌رویی، رهنمودهای روشنگر خود را برای انجام این پروژه از من دریغ نکرده‌اند، از استاد مشاورم جناب آقای دکتر حامد فربه که راهنمایی‌های ایشان از بدو ورود به دانشگاه کمک بسیاری به من در طی کردن مسیر تحصیل بوده‌اند، از مادر و پدرم که همواره در مواجهه با سختی‌های این دنیا دلسوزانه همراهم بوده‌اند، و از سایر عزیزانی که در کنارشان این نتیجه حاصل آمد کمال تشکر و قدردانی را دارم.

آریان بوکانی

مرداد ۱۴۰۲

چکیده

در این قسمت چکیده پایان نامه نوشته می‌شود. چکیده باید جامع و بیان‌کننده خلاصه‌ای از اقدامات انجام‌شده باشد. در چکیده باید از ارجاع به مرجع و ذکر روابط ریاضی، بیان تاریخچه و تعریف مسئله خودداری شود.

واژه‌های کلیدی:

نگهداری پیش‌بینانه، تحلیل لرزش، یادگیری ماشین، اینترنت اشیاء

فهرست مطالب

عنوان صفحه

فهرست اشکال

شکل صفحه

فهرست جداول

جدول صفحه

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

فصل اول

مقدمه

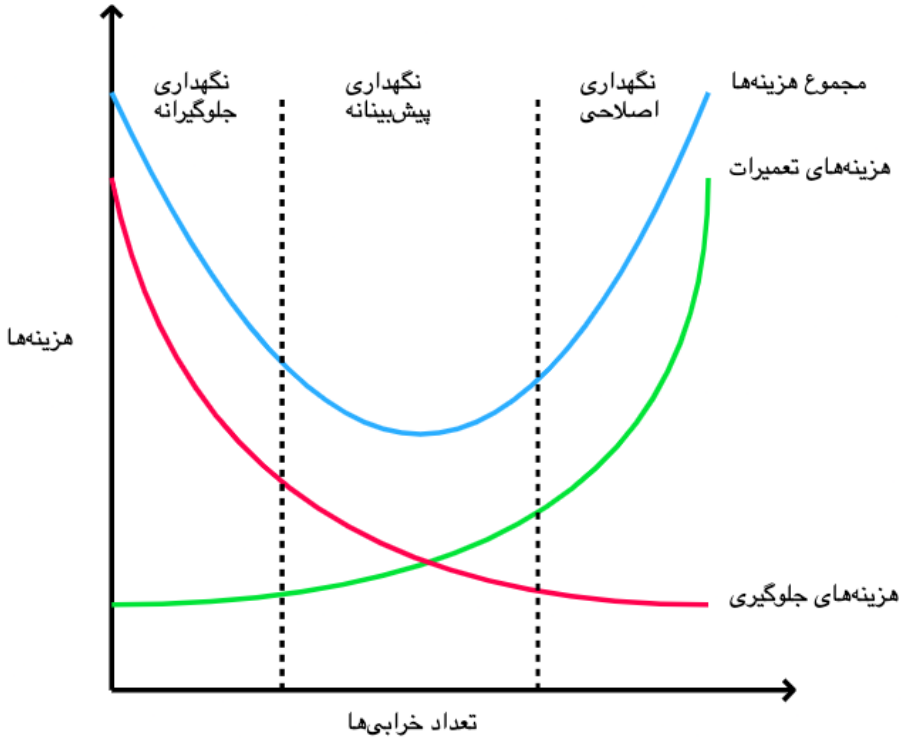
۱-۱ مقدمه

در صنعت، نگهداری و تعمیرات[۱] به تمام فعالیت‌هایی 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**[۲]: به جایگزینی قطعه خراب شده در سیستم می‌پردازد. در این رویکرد، تا زمانیکه فرایند 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**[۳]: 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 اصراف در هزینه و قطعات مصرفی صورت گیرد[۲].
- **نگهداری و تعمیرات پیش‌بینانه**[۴]: 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[۳، ۲].

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].

[1] Maintenance
[2] Corrective Maintenance
[3] Preventive Maintenance
[4] Predictive Maintenance



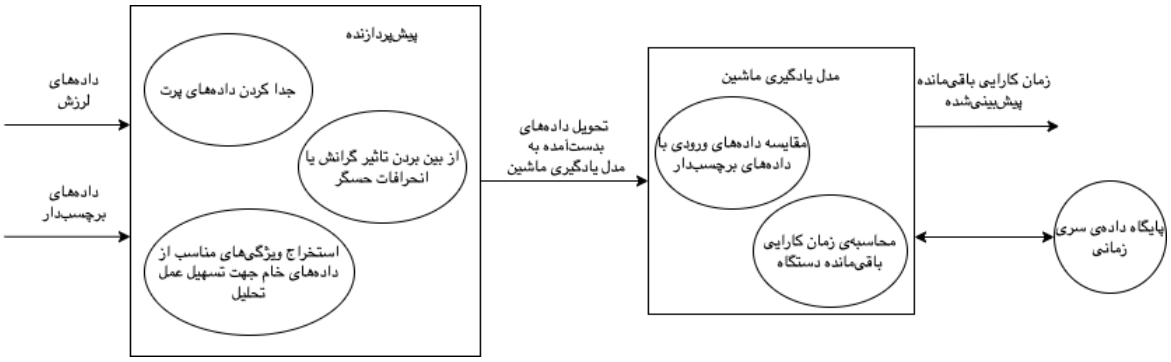
شکل ۱-۱: مقایسه‌ی هزینه‌های انواع نگهداری‌ها

۱-۲ تعریف مسئله

هدف از انجام این پروژه، پیاده‌سازی سیستمی برای اجرا کردن نگهداری پیش‌بینانه بر روی گره‌های موجود در یک اینترنت اشیاء[۵] به هم 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[۶] اشاره کرد. برای پیاده‌سازی این سیستم همانطور که در شکل ۱-۲ به تصویر آمده است، نیازمند آنیم که داده‌های لرزش مربوط به گره‌ها را که توسط یک سیستم قابل اتکا[۷] جمع‌آوری شده است، دریافت کرده و با جدا کردن داده‌های پرت[۸]، از بین بردن تاثیر اختلال[۹] ایجاد شده توسط گرانش و خرابی یا درست کار نکردن حسگر[۱۰] اندازه‌گیری لرزش، استخراج ویژگی[۱۱]های مناسب برای انجام تحلیل روی داده و

[5] Internet of Things
[6] Vibration Analysis
[7] Reliable
[8] Outlier Data
[9] Noise
[10] Sensor
[11] Feature Extraction

درنهایت پیشنهاد دادن مدلی برای نحوه‌ی یادگیری ماشین[۱۲] و تحلیل و مقایسه‌ی داده‌های بدست‌آمده با داده‌های برچسب‌دار[۱۳]، عمر باقی‌مانده[۱۴]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[۴]. برای راحتی استفاده از سیستم طراحی‌شده، مستقر ساختن[۱۵] سرویس توسعه‌یافته‌شده روی ابر[۱۶] و همچنین احراز هویّت مدیر[۱۷]ان و دروازه[۱۸]های ارسال‌کننده داده‌ی لرزش کارگزار[۱۹]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 این‌حال 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 خرابی قطعات مبتنی بر میزان استفاده از تجهیزات و بار داخلی آنها ارائه شده‌است اما مدل ارائه‌شده 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

[12] Machine Learning
[13] Labeled Data
[14] Remaining Useful Lifetime
[15] Deploy
[16] Cloud
[17] Admin
[18] Gateway
[19] Server

تجهیزات چرخشی ارائه شده‌است 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 شبیه‌سازی‌شده و کنترل‌شده آزمایش شده‌است و در هنگام استفاده در محیط واقعی غیرعملی است.

بطور کلی در پژوهش‌های یادشده روندهای در پیش‌گرفته‌شده برای پیش‌بینی خرابی و زمان آن، مختص نوع خاصی از تجهیزات است و در یک محیط آزمایشگاهی و کنترل‌شده ارزیابی شده‌اند. در حالی‌که در این پروژه با استفاده از داده‌های جمع‌آوری‌شده 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 طراحی‌شده روی ابر مستقر نشده‌اند و سرویس‌هایی همانند 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 استفاده‌شده

در این فصل تکنولوژی‌ها و چارچوب[1]های اصلی دخیل در توسعه این دستگاه را به طور دقیق مورد بررسی قرار می‌دهیم.

۲-۱ زبان برنامه‌نویسی

برای انتخاب زبان برنامه‌نویسی مناسب برای توسعه مدل یادگیری ماشین شرح‌داده شده، باید معیارهای متفاوتی را در نظر گرفت. برای این منظور زبان پایتون[2] را برگزیدیم. مواردی همچون داشتن چارچوب‌ها و کتابخانه‌های قدرتمند یادگیری ماشین، توسعه‌ی آسان و سریع و محبوبیت بالا از دلایل اصلی انتخاب پایتون به عنوان 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.

۲-۱-۱ زبان برنامه‌نویسی پایتون

یک زبان برنامه‌نویسی عمومی و سطح بالا است که فلسفه طراحی آن بر روی خوانایی کد تأکید دارد. نحو[3] پایتون به برنامه‌نویسان امکان می‌دهد تا مفاهیم را با تعداد کمتری خط کد نسبت به زبان‌هایی مانند سی[4] 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[۸]. یکی از مشخصه‌های مهم پایتون این است که از چندین الگو[5]ی برنامه‌نویسی، از جمله شیءگرا[6] 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[۹]. 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[۹، ۱۰].

[1] Framework
[2] Python
[3] Syntax
[4] C Programming Language
[5] Paradigm
[6] Object Oriented Programming (OOP)

- **متن‌باز بودن**[۷]: اگرچه تمام حقوق این زبان برنامه‌نویسی متعلق به سازمان پایتون است، اما درحال حاضر به عنوان یک نرم‌افزار متن‌باز 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[۹، ۱۰].

- **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 به خوبی 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[۹، ۱۰].

حال به بررسی معایب پایتون می‌پردازیم. نکته‌ی قابل توجه در این قسمت این است که اگر معایب نام‌برده شده تاثیر زیادی در کیفیت خدمت ارائه‌شده به کاربر بگذارند، استفاده از پایتون اصلا توصیه نمی‌شود و باید به دنبال جایگزینی مناسب گشت. از جمله کاستی‌های پایتون عبارت‌اند از:

- **کندی**: به عنوان 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[۹، ۱۰].

- **دشواری فرایند نگهداری**[۸]: به دلیل اینکه پایتون یک زبان با نوع پویا است، یک چیز ممکن است به راحتی 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[۹] می‌توان تا حدی این از وقوع این مشکل جلوگیری کرد[۹، ۱۰].

[7] Open Source
[8] Maintaining
[9] Unit Tests

۲-۲ چارچوب‌ها و کتاب‌خانه‌ها

در این پروژه از چارچوب فست‌ای‌پی‌آی[۱۰] برای دریافت درخواست‌ها و ارسال نتایج پیش‌بینی استفاده‌شده است(لوگوی مربوط به این چارچوب در شکل ۲-۱[۱۱] آورده‌شده است). این سرویس به عنوان یک بسته[۱۱]ی پایتونی به کارگزار اصلی اضافه شده است. همچنین برای پیاده‌سازی مدل و انجام محاسبات ریاضی و ماتریسی از کتابخانه‌های نام‌پای[۱۲] و سایکیت[۱۳] 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.

۲-۲-۱ چارچوب FastAPI

یک چارچوب مدرن با عملکرد عالی برای طراحی وب است که برای پایتون توسعه داده‌شده است. از ویژگی‌های کلیدی FastAPI می‌توان به موارد زیر اشاره کرد[۱۱].

شکل ۲-۱: لوگوی FastAPI[۱۱]

- **سریع‌بودن**: همانطور که در قسمت‌های قبل بدان اشاره شده، یکی از معایب 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.

[10] FastAPI
[11] Package
[12] NumPy
[13] Scikit-Learn

- **سادگی توسعه**: بدلیل اینکه این زبان از نحو پایتون برای توسعه بهره می‌برد، سرعت توسعه‌دهنده 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[۱۴] کمتری که منشاء آن برنامه‌نویس هستند پیش بیایند.

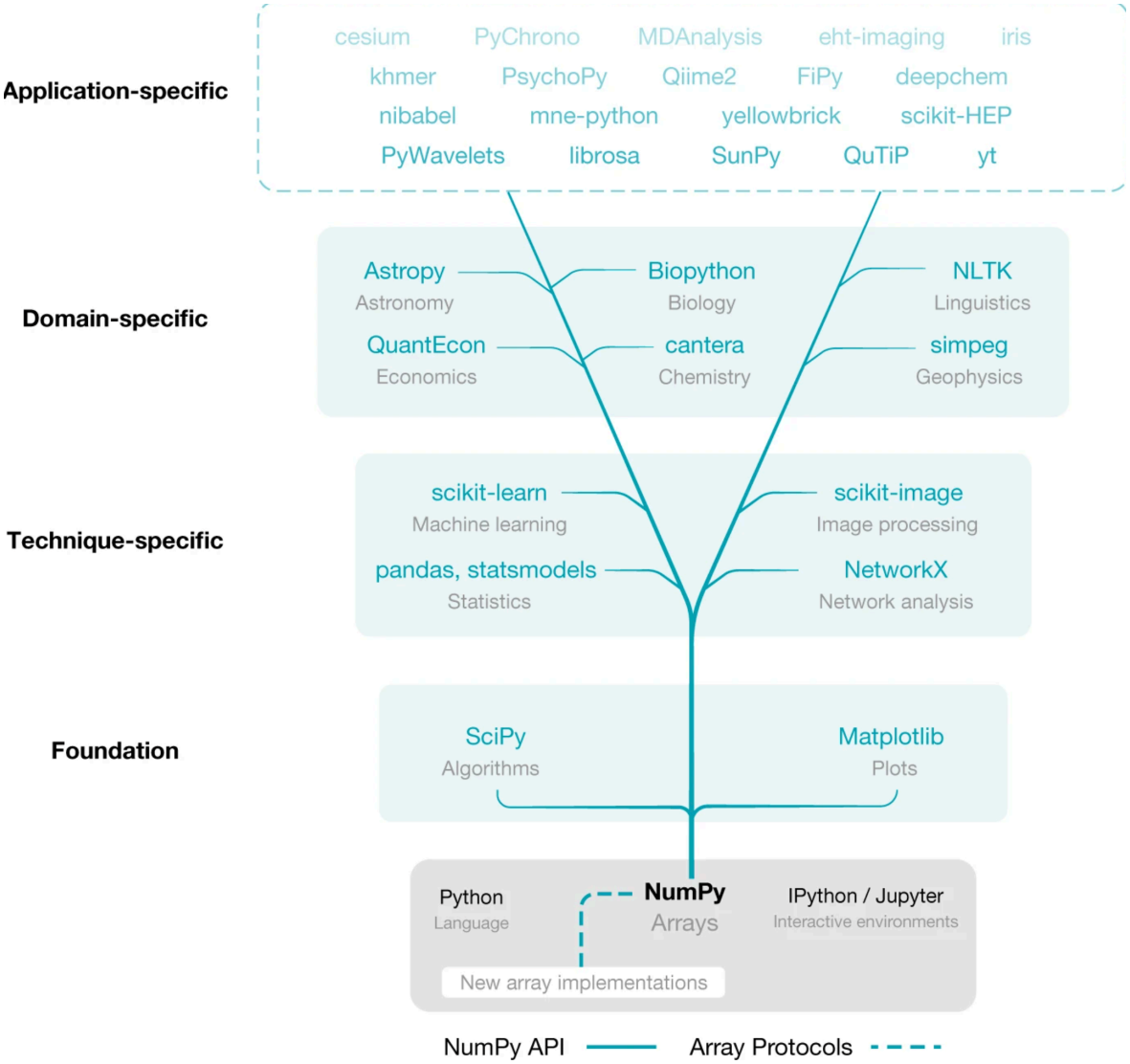
۲-۲-۲ کتاب‌خانه‌ی NumPy

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[۱۲] 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[۱۳]. آرایه‌ی نام‌پای 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[۱۵] به حافظه و تعدادی فراداده[۱۶] برای تفسیر داده‌های موجود در آرایه است[۱۳، ۱۲].

۲-۲-۳ کتاب‌خانه‌ی Scikit-Learn

Scikit-Learn، جامع‌ترین و بزرگ‌ترین بسته یادگیری ماشین منبع‌باز در پایتون است. چون یادگیری ماشین اغلب به عنوان یک جزء از یک برنامه عمومی‌تر (همانند پروژه‌ی کنونی که به عنوان یک سرویس در وب توسعه داده‌شده است) استفاده می‌شود، ایده‌آل است که از همان زبان برنامه‌نویسی استفاده شود تا به‌صورت یکپارچه با سایر بخش‌های برنامه هماهنگ شود. با استفاده از قابلیت‌های گسترده پایتون، Scikit-Learn به عنوان یک بسته محبوب برای برنامه‌های مرتبط با یادگیری ماشین در حال رشد است[۱۴]. این کتاب‌خانه شامل توابع و اشیاء فراوانی برای مسائل طبقه‌بندی، رگرسیون، تقریب ماتریس کوواریانس، کاهش بعد و پیش‌پردازش داده‌ی خام می‌باشد[۱۵]. اگرچه پایتون یک زبان برنامه‌نویسی

[14] Bugs
[15] Pointer
[16] Metadata



شکل ۲-۲: گراف وابستگی کتاب‌خانه‌های پایتون به NumPy[۱۲]

تفسیری است، اما بیشتر روش‌های یادگیری ماشین در Scikit-Learn بر پایه کتابخانه‌های دودویی کامپایل شده است که در ابتدا با زبان‌های فورتران[۱۷]، سی یا سی‌پلاس‌پلاس[۱۸] برنامه‌نویسی شده‌اند. این پیاده‌سازی‌های مبتنی بر دودویی‌ها به طور قابل توجهی کارایی محاسبات را بهبود می‌بخشند[۱۴، ۱۵].

۲-۲-۴ کتاب‌خانه‌های جانبی

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

[17] Fortran
[18] C++

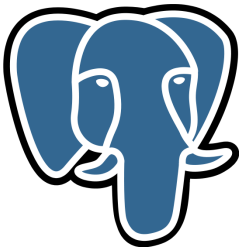
- **PyJWT و Werkzeug**: از این دو کتاب‌خانه برای توسعه‌ی سرویس احراز هویت مدیران و دروازه‌های ارسال‌کننده‌ی داده‌های مربوط به لرزش با کمک استاندارد JWT استفاده کرده‌ایم.
- **Pandas**: برای کارهایی همانند خواندن مجموعه‌ی داده[۱۹] و پردازش روی آنها از این کتاب‌خانه کمک گرفته شد.
- **SQLAlchemy**: این کتاب‌خانه به دلیل در اختیار گذاشتن رابط‌های برنامه‌نویسی[۲۰] مناسب برای ارتباط با پایگاه‌داده‌های رابطه‌ای، بسیار محبوب است و در این پروژه نیز از آن استفاده کرده‌ایم.

۲-۳ پایگاه داده‌ی رابطه‌ای

پایگاه داده برنامه‌ای برای ذخیره و بازیابی سریع حجم فراوانی از داده و به صورت مکرر می‌باشد. پایگاه داده‌ی رابطه‌ای[۲۱] در سال ۱۹۷۰ میلادی معرفی شد. داده در این نوع از پایگاه داده به شکل جداولی[۲۲] (از جدول به عنوان رابطه[۲۳] 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 موجودیّت 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[۱۶]. از معروف‌ترین پایگاه‌داده‌های رابطه‌ای می‌توان به MySQL، PostgreSQL، Oracle Database و Microsoft SQL Server اشاره کرد.

اکثر پایگاه‌های داده رابطه‌ای از زبان پرسمان ساختاریافته[۲۴] برای دسترسی و اصلاح داده‌های ذخیره شده در پایگاه داده استفاده می‌کنند. برای انجام این پروژه از PostgreSQL که یکی معروف‌ترین پایگاه داده‌ی رابطه‌ای متن‌باز است، استفاده کرده‌ایم(لوگوی آن در شکل ۲-۳[۱۷] آورده شده است).

[19] Dataset
[20] Application Programming Interface (API)
[21] Relational Database
[22] Tables
[23] Relation
[24] Structured Query Language (SQL)



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

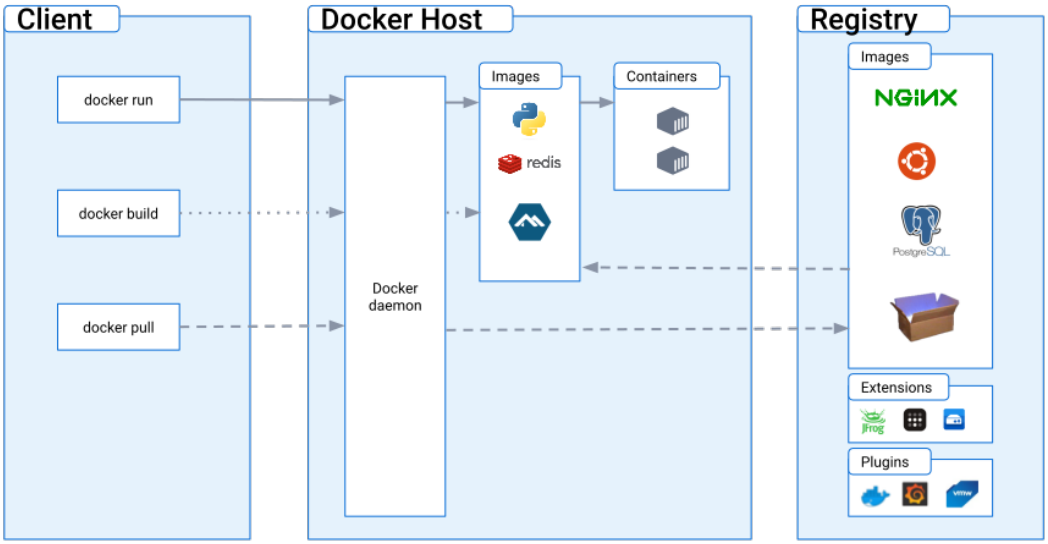
۲-۴ داکر

داکر[۲۵] در حال حاضر معروف‌ترین و پراستفاده‌ترین ابزار برای پیاده‌سازی معماری سرویس‌های کوچک[۲۶] 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[۲۷]، 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 داده‌شده 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 ۲-۴[۱۹] مشخص است، خود از چندین بخش مجزا تشکیل‌شده است که در این قسمت آنها را شرح خواهیم داد. داکر از معماری مشتری-کارگزار[۲۸] استفاده می کند. در ساختار داکر، Docker Client با Docker Daemon که کارهایی همانند ساخت، اجرا و توزیع کانتینرهای داکری را انجام می دهد، صحبت می کند. سرویس‌های Docker Client و Docker Daemon می توانند روی یک سیستم اجرا شوند، یا می توانند روی دوی 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 به صورت همزمان را 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 بلافاصله

[25] Docker
[26] Microservices
[27] Container
[28] Client-Server

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



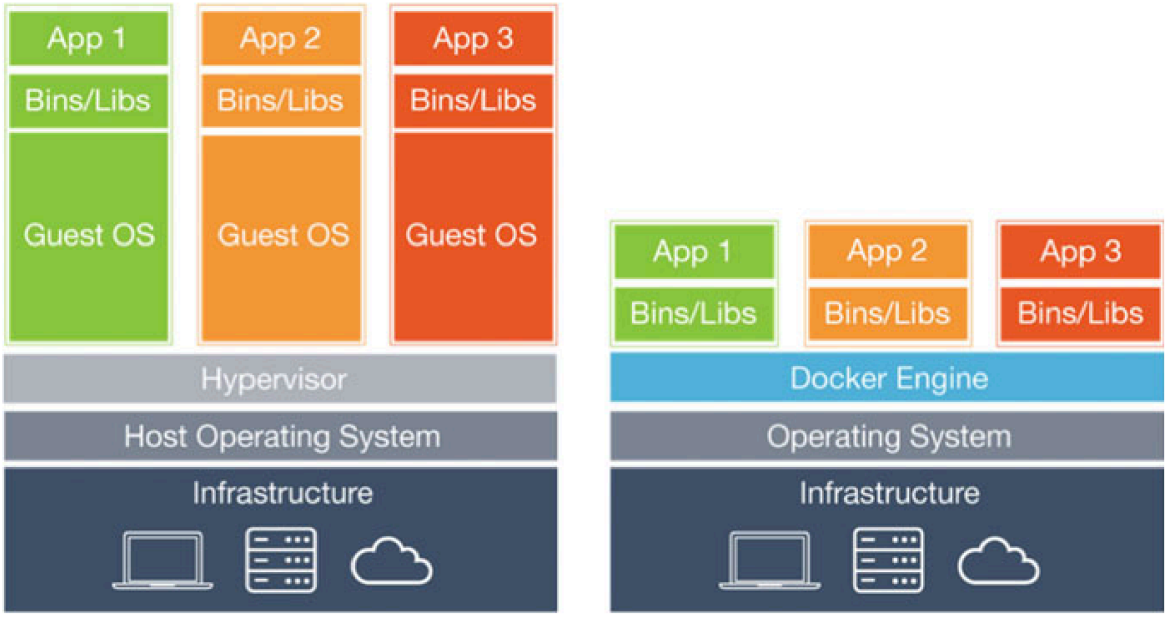
شکل ۲-۴: معماری داکر[۱۹]

توسعه‌ی داکر در واقع تلاشی برای رفع مشکلات استفاده از ماشین‌های مجازی[۲۹] بود. کانتینرهای داکر و ماشین‌های مجازی مطابق شکل ۲-۵ در نحوه‌ی مجازی‌سازی و استفاده از منابع کامپیوتر با همدیگر متفاوت‌اند. ماشین‌های مجازی منابع سخت‌افزاری سیستم را بین خود تقسیم می‌کنند و بدلیل وجود لایه‌ای اضافی برای ایجاد دستورات قابل‌فهم برای لایه‌ی سخت افزار، نسبت به کانتینرهای داکر که منابع سیستم عامل را بین خود تقسیم می‌کنند و یک لایه کمتر دارند، کندتر اجرا می‌شوند و سربار بشدت بالاتری را متحمل می‌شوند[۱۸، ۲۰].

۲-۵ جمع‌بندی و نتیجه‌گیری

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

[29] 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 نحوه‌ی 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 ارائه‌دهنده‌ی خدمات ابری[1] یا به اختصار CSP، گزینه‌ی مناسبی برای این نیاز می‌باشد. برای این منظور، ما این پروژه را پس از پیاده‌سازی، توسط سرویس زیرساخت به عنوان خدمت[2] مستقر کردیم.

۳-۱ زیرساخت به عنوان خدمت

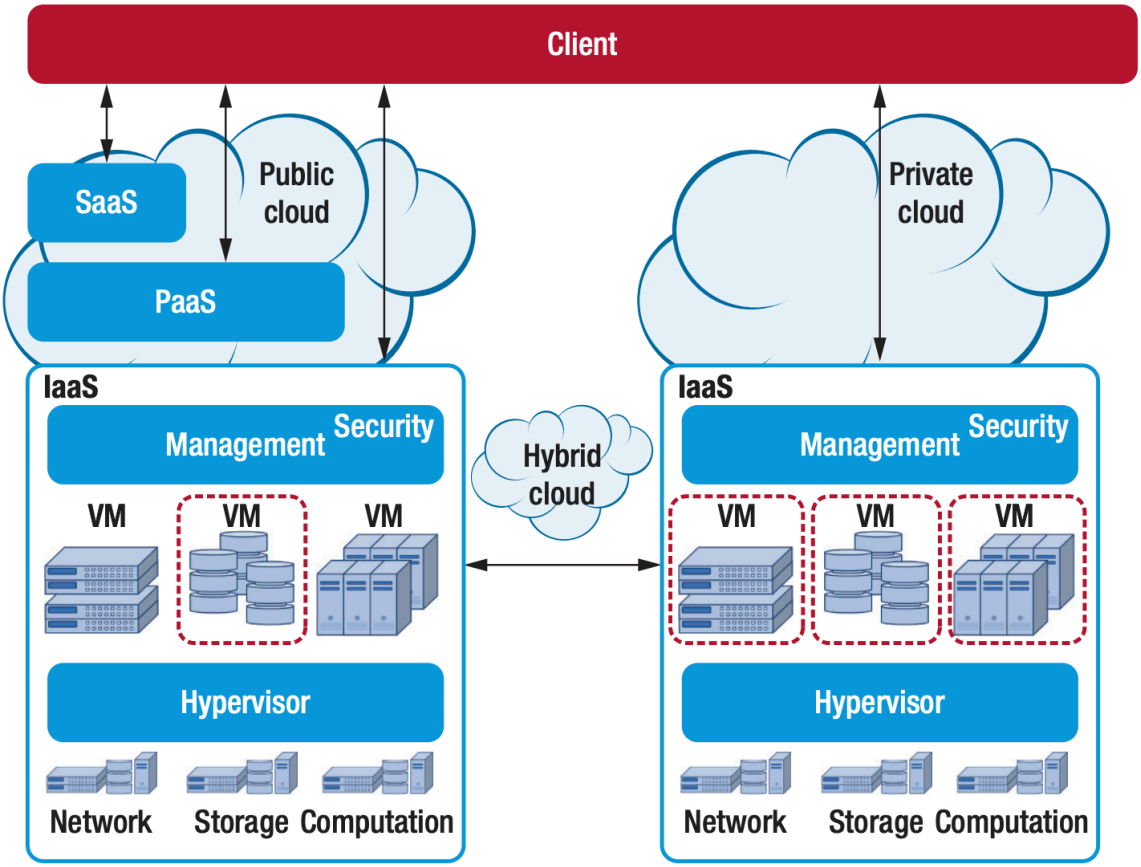
در این مدل، سرویس‌دهنده ابر یا CSP مجموعه‌ای از منابع محاسباتی مجازی شده را در ابر فراهم می‌کند (مانند پهنای باند شبکه، ظرفیت ذخیره‌سازی، حافظه، قدرت پردازش). مسئولیت مشتری در این حالت این است که سیستم‌عامل و برنامه‌های نرم‌افزاری را روی این منابع مجازی اجرا و نگهداری کند. زیرساخت به عنوان خدمت یا IaaS از فناوری مجازی‌سازی استفاده می‌کند تا منابع فیزیکی را به منابع منطقی تبدیل کند که مشتریان می‌توانند به صورت پویا از آنها استفاده کنند و آنها را هنگام نیاز ایجاد و آزاد کنند[۲۱]. در شکل ۳-۱[۲۲] 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 گذاشته‌شده دارد[۲۲].

۳-۲ 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 ۳-۲[۲۳] مرسوم‌تر هستند:

- **پیاده‌سازی دسته‌ای**[3]: پیش‌بینی‌ها به فاصله‌های زمانی مشخص محاسبه می‌شوند و پیش‌بینی‌های حاصل در پایگاه داده ذخیره می‌شوند و به راحتی می‌توان آنها را در صورت نیاز بازیابی کرد. با این حال، نمی‌توان از داده‌های بروزتر استفاده کرد و پیش‌بینی‌ها می‌توانند به سرعت منسوخ شوند[۲۴، ۲۵].
- **پیاده‌سازی بی‌درنگ**[4]: در این نوع از استقرار، درخواست کاربر برای گرفتن جدیدترین پیش‌بینی‌ها

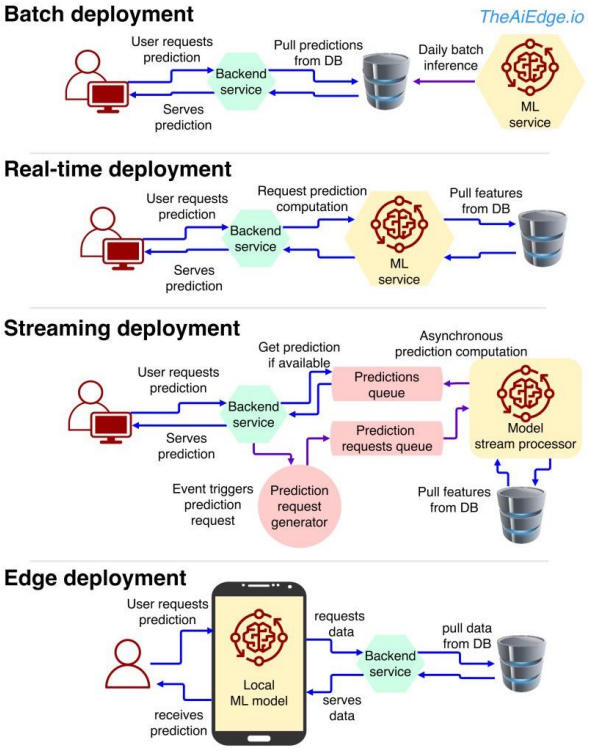
[1] Cloud Services Providers
[2] Infrastructure as a Service (IaaS)
[3] Batch Deployment
[4] Real-Time Deployment



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

به عنوان یک راه‌انداز[۵] توسط رابط برنامه‌نویسی اچ‌تی‌تی‌پی[۶] 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 به عنوان نتیجه ارسال می‌کند. مشکل اصلی این روش قرارگیری مدل یادگیری ماشین، کند بودن روند یادگیری و پیش‌بینی است که منجر به منتظر ماندن کاربر می‌گردد. می‌توان با بهره‌گیری از فرآیند[۷]های چندریسمانی[۸] برای دریافت درخواست‌های کاربر و انجام مرحله‌ی یادگیری و پیش‌بینی مدل، تا حد زیادی این مشکل را برطرف کرد[۲۴، ۲۵].

[5] Trigger
[6] Hypertext Transfer Protocol (HTTP)
[7] Process
[8] Multi-Threaded



شکل ۳-۲: انواع روش‌های استقرار مدل‌های یادگیری ماشین[۲۳]

- **پیاده‌سازی جریانی**[۹]: این امکان را می‌دهد تا فرآیند ناهمزمان[۱۰]تری ایجاد شود. یک رویداد می‌تواند شروع فرآیند استنتاج را فراهم کند. این فرآیند در صف یک واسط پیام[۱۱] مانند کافکا[۱۲] قرار داده می‌شود و مدل یادگیری ماشینی در هنگام آماده شدن 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 کم‌شدن تاخیر پاسخ‌دهی به کاربران اشاره کرد[۲۴، ۲۵].

- **پیاده‌سازی لبه‌ای**[۱۳]: 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

[9] Streaming Deployment
[10] Asynchronous
[11] Message Broker
[12] Apache Kafka
[13] Edge Deployment

کوچکتر نصب شوند[۲۳].

بدلیل اینکه گره‌های موجود در شبکه‌ی اشیاء دارای توان پردازشی محدود هستند و اینکه ماهیت مدل هوش مصنوعی مربوط به حوزه‌ی کاری پیش‌بینی عمر دستگاه‌ها بدین‌گونه است که حتما باید از داده‌های مربوط به همه‌ی گره‌های موجود استفاده کرد، با توجه به گزینه‌های مطرح‌شده 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. به طور دقیقتر، مدل هوش مصنوعی به عنوان یک سرویس اضافی برای کارگزار اصلی توسعه داده‌شده، تعبیه شده است.

۳-۳ جمع‌بندی و نتیجه‌گیری

در این فصل نحوه‌ی استقرار مدل یادگیری ماشین را به صورت دقیق بررسی کردیم. همانطور که گفته شد، از میان سرویس‌های مختلف ارائه‌شده توسط شرکت‌های ارائه‌دهنده‌ی سرویس‌های ابری، از زیرساخت به عنوان خدمت برای مستقر کردن سرویس استفاده شد. این سرویس به عنوان 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، پیچیده نبودن فرایند یادگیری در این مورد بخصوص و همچنین نیاز به بررسی همه‌ی داده‌های ارسال‌شده توسط گره‌های مختلف برای داشتن دقیق‌ترین پیش‌بینی‌ها انتخاب شد.

فصل چهارم

پیاده‌سازی کارگزار اصلی

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 نوشته از آنها استفاده خواهیم کرد، مشخص می‌کنیم. در قسمت بعد مراحل پیش‌پردازش[1] را که روی این داده‌ها انجام می‌شود به ترتیب توضیح می‌دهیم و سپس ویژگی‌هایی که نیاز داریم از این داده‌های خام دربیاوریم 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 ۵-۱، قالب بسته‌ی هوش مصنوعی توسعه داده‌شده برای کارگزار اصلی را مشاهده می‌کنید که در این فصل به توضیح بخش‌های مختلف آن می‌پردازیم.

```
analytics
├── __init__.py
├── clustering.py
├── preprocesser.py
├── ransac.py
├── rul.py
└── transformer.py
```

شکل ۵-۱: ساختار کلی بسته‌ی هوش مصنوعی

۵-۱ 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 بدست‌آمده برای حسگرهای متفاوت بازه‌های زمانی مختلف را در بر می‌گیرند و همگن نیستند. ثانیاً این داده‌های دارای انحرافاتی در اندازه‌گیری بدلیل 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[۴]. با توجه به نکاتی که مطرح کردیم، پیاده‌کردن یک سیستم پیش‌پردازش و استخراج‌کننده‌ی ویژگی‌های مناسب، الزامی است.

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

[1] 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^l_{nm}	بردار kبعدی مربوط به لرزش در محور $l \in \{x, y, z\}$

جدول ۵-۲: برچسب‌های استفاده‌شده برای تعیین وضعیت دستگاه‌ها

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

تعیین کردن وضعیت گره‌های موجود استفاده می‌کنیم.

۵-۲ پیش‌پردازش

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

۵-۲-۱ از بین بردن انحرافات

حسگرهای کم‌هزینه MEMS ،که داده‌های جمع‌آوری‌شده برای این پروژه توسط این نوع از حسگرها تأمین شده است، غالباً با گذشت زمان دچار انحرافاتی در اندازه‌گیری خواهند شد که منجر به اضافه یا کم شدن یک مقدار شتاب 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[۴].

[2] Outlier Data

برای از بین بردن این مشکل همانطور که در برابری (۵-۱) آورده شده است[۲۶]، از هنجارکردن[۳] داده با کم‌کردن میانگین مقادیر شتاب اندازه‌گیری شده در هر کدام از سه محور از مقادیر اندازه‌گیری‌شده استفاده کرده‌ایم. لازم به ذکر است همانطور که مشخص است، $\hat{a}^l_{nm}$ نماد ماتریس هنجارشده است.

$$\hat{a}^l_{nm} = a^l_{nm} - \sum_{k=1}^{K} \frac{a^l_{nmk}}{K} \tag{۵-۱}$$

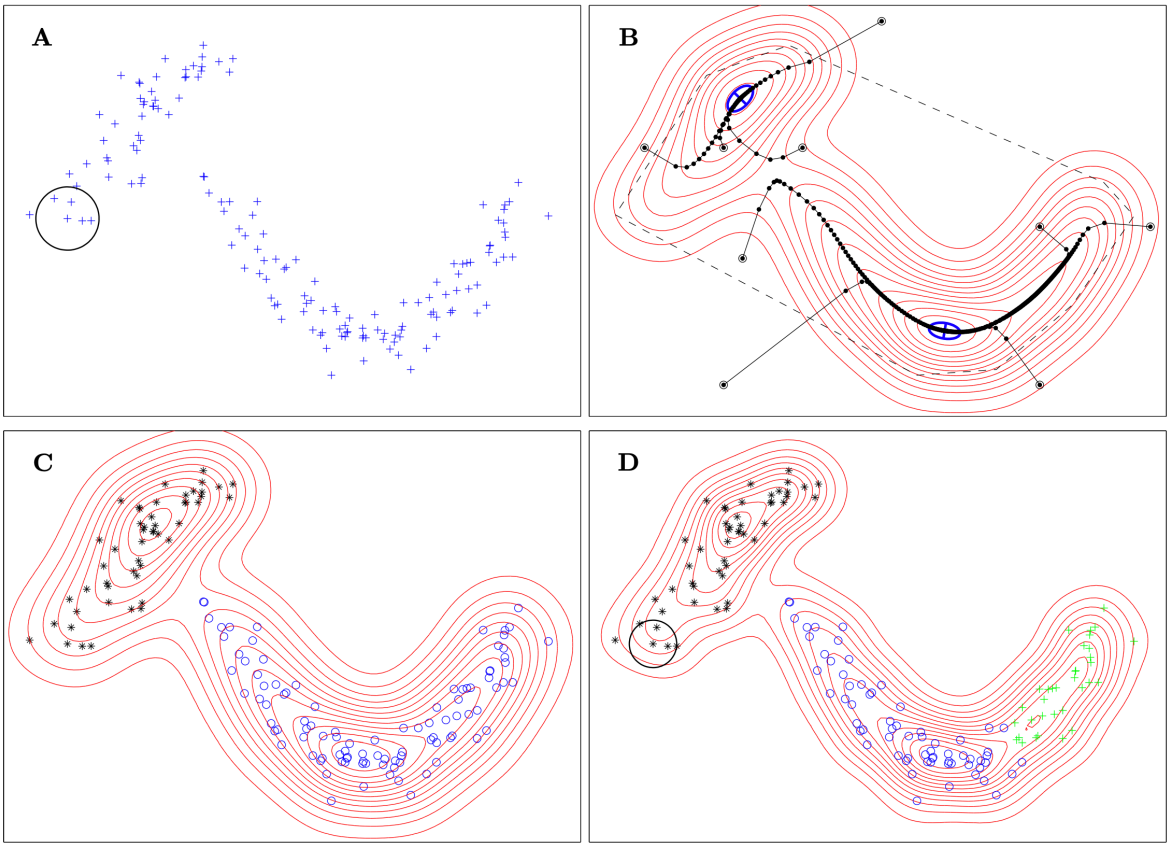
۵-۲-۲ از بین بردن داده‌های پرت

در سیستم طراحی‌شده برای جمع‌آوری اطلاعات، ممکن است که تعدادی از حسگرها دچار مشکل شده باشند و داده‌ای که تحویل دروازه می‌دهند دقیق و در راستای داده‌های از قبل جمع‌آوری شده نباشد. به طور کلی، پایدار بودن میانگین شتاب دریافت‌شده از هر اندازه‌گیری، معیار خوبی برای تشخیص صحت و درستی اطلاعات جمع‌آوری شده است[۴]. به عبارتی دیگر، میانگین لرزش‌های اندازه گرفته‌شده نباید به طور ناگهانی بالا یا پایین روند و باید در همان حدود اندازه‌گیری‌های قبل باشند. برای جدا کردن این داده‌ها که اصطلاحا به آنها داده‌های پرت می‌گوییم، پیش از شروع یادگیری مدل، ابتدا میانگین شتاب جمع‌آوری شده برای هر اندازه‌گیری موجود را در هر سه بعد حساب کرده و سپس با کمک یک الگوریتم خوشه‌بندی[۴]، این داده‌ها را جدا می‌کنیم.

برای انجام عملیات تشخیص داده‌های پرت، از الگوریتم خوشه‌بندی میانگین تغییر[۵] استفاده کرده‌ایم که الگوریتمی بر اساس تخمین تراکم هسته[۶] می‌باشد. در شکل ۵-۲[۲۷] نمونه‌ای از یک الگوریتم خوشه‌بندی تراکم‌محور آورده شده است. روند کار این الگوریتم بدین صورت است که به ازای ورودی به صورت نقاط و پارامتر ورودی پهنای باند[۷]، الگوریتم به طور مکرر هر نقطه داده را به نزدیکترین مرکز خوشه اختصاص می‌دهد و جهت نزدیکترین مرکز خوشه بر اساس جایی که اکثر نقاط نزدیک در آن قرار دارند تعیین می‌شود. در هر بار تکرار، هر نقطه داده به جایی که بیشترین نقاط در آن قرار دارد، نزدیکتر می‌شود، که در نهایت به مرکز خوشه منجر خواهد شد. هنگامی که الگوریتم متوقف می‌شود، هر نقطه به یک خوشه اختصاص داده می‌شود. همانطور که گفته‌شد، این الگوریتم بغیر از پهنای باند به پارامتر دیگری نیاز ندارد. این امر سبب می‌شود که مواردی همانند تعداد و مراکز هر خوشه، توسط خود الگوریتم

[3] Normalizing
[4] Clustering
[5] Mean Shift
[6] Kernel Density Estimate
[7] Bandwidth

مشخص شوند[۲۷].



شکل ۵-۲: روند خوشه‌بندی یک الگوریتم تراکم‌محور[۲۷]

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، خوشه‌بندی میانگین تغییر سه‌بعدی استفاده می‌کنیم و در نهایت داده‌هایی که در خوشه‌ی 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 کتابخانه‌ی Scikit-Learn ارث‌بری می‌کند.

۵-۲-۳ 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[۸] 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 دامنه‌ی فرکانسی با کمک تبدیل فوریه[۹]، شروع خوبی است.

ویژگی Root Mean Square (RMS)

ویژگی مربع میانگین ریشه یا به اختصار RMS تنها بزرگی اندازه‌ی لرزش‌های اندازه گرفته‌شده را نمایان می‌کند و برای بردارهای لرزش هر اندازه‌گیری منسوب به هر دستگاه اینترنت اشیاء موجود، به صورت برابری (۵-۲) محاسبه می‌شود. در شکل ۵-۳ مقادیر محاسبه شده‌ی این ویژگی را برای همه‌ی اندازه‌گیری‌های یک گره موجود حساب کرده‌ایم. محور افقی شناسه‌ی هر یک از اندازه‌گیری‌ها می‌باشد.

$$
\begin{aligned}
r_{nm}^{l} &= \frac{1}{\sqrt{K}} \|a_{nm}^{l}\| \\
r_{nm}^{2} &= \sum_{l \in \{x,y,z\}} (r_{nm}^{l})^2
\end{aligned}
\qquad (۵-۲)
$$

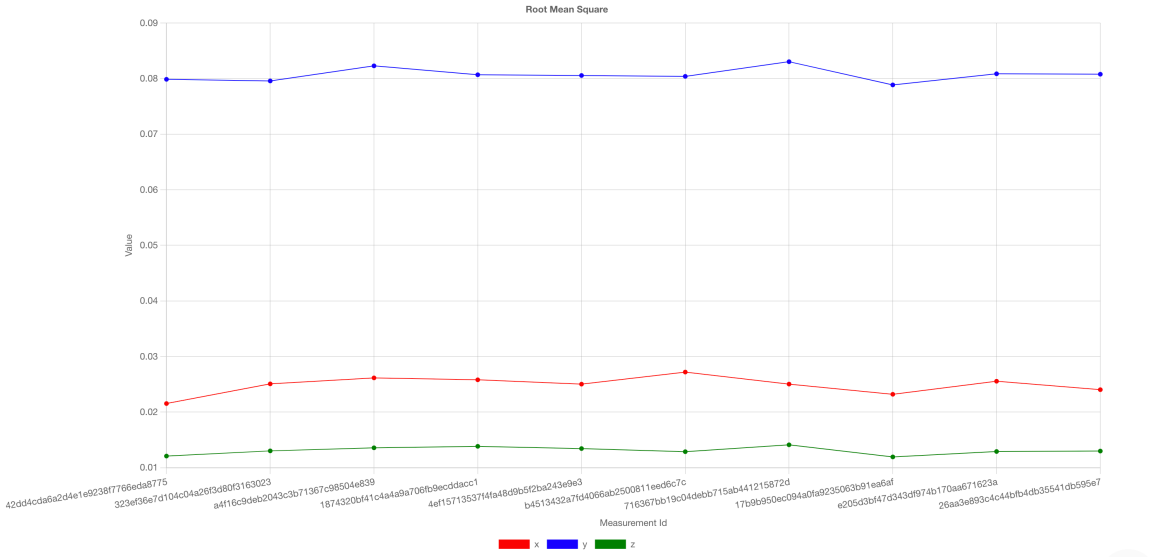
همانطور که گفته‌شد این ویژگی اطلاعات زیادی را در اختیار ما قرار نمی‌دهد. نکته‌ی قابل توجه دیگر در مورد این ویژگی این است که مقادیر بدست آمده در این ویژگی منسوب به دامنه‌ی زمانی می‌باشد و با توجه به مسئله‌ی کنونی، این ویژگی کاربردی در تحلیل اطلاعات لرزش برای ما نخواهد داشت.

ویژگی Power Spectral Density (PSD)

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

[8] Time Domain

[9] Fourier Transform



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

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

$$
\begin{aligned}
s_{nm}^l &= \frac{1}{2K}(a_{nm}^l \times W_K)^2 \\
s_{nm} &= \sum_{l \in \{x,y,z\}} s_{nm}^l
\end{aligned}
\qquad (۵-۳)
$$

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 معمولا در هنگام محاسبات ($s^T s$) منجر به ایجاد ماتریس منفرد[۱۲] 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 به دلیل انحراف اندازه گیری ذاتی در سنسور MEMS غیرقابل اتکا است.

[10] Discrete Cosine Transform (DCT)
[11] Parseval's Theorem
[12] Singular Matrix

ویژگی Harmonic Peaks Feature

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[۱۳] با پنجره‌ی هان[۱۴] که در برابری (۵-۴) آورده شده است(در این برابری، n_h اندازه‌ی پنجره است که در مسئله‌ی ما ۱۶ انتخاب شده است). پس از انجام این عملیات، سیگنال ویژگی‌ها هموار[۱۵] می‌شود و از این پس می‌توان عملیات جست‌وجو برای بیشترین مقادیر را شروع کرد.

$$w_h(n) = 0.5(1 - \cos\frac{2\pi n}{n_h - 1}) \qquad (۵-۴)$$

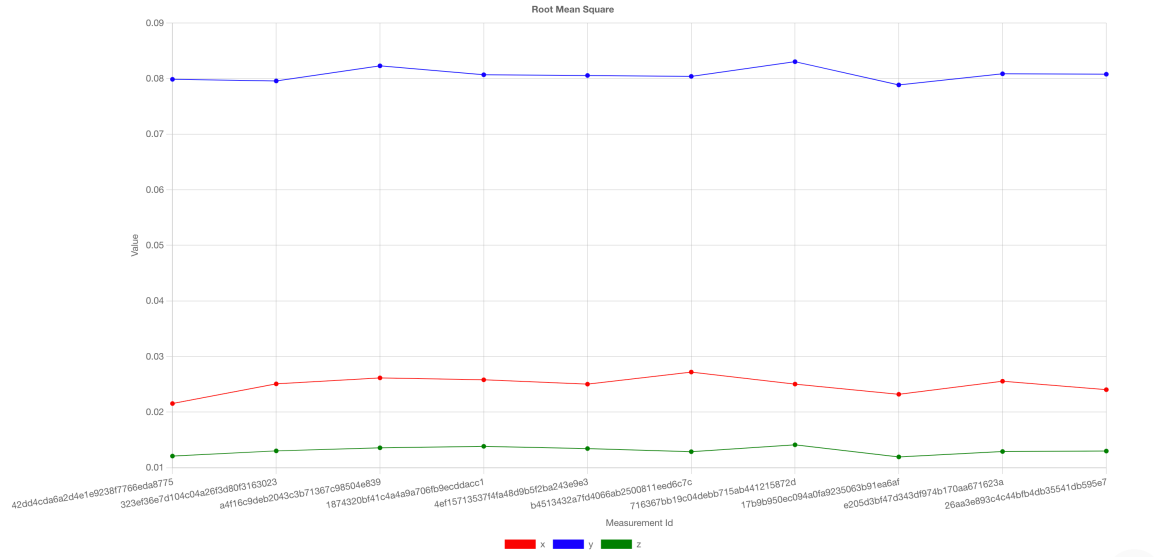
- پیدا کردن ۲۰ تا از بیشترین مقادیر ویژگی‌های هموارشده از طریق شناسایی نقاطی که مشتق اول سیگنال در آنها از مثبت به منفی، تغییر علامت می‌دهد

در شکل ۵-۴ مقادیر ویژگی چگالی طیفی توان و ۲۰ تا از بیشترین قله‌های موزون را برای یکی از اندازه‌گیری‌های منسوب به یک گره، نمایش داده‌ایم.

۵-۳ نحوه‌ی 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 کمّی بین ویژگی‌های مختلف پیدا کرد.

[13] Convolution
[14] Hann Window
[15] Smooth



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

۵-۳-۱ محاسبه‌ی مشابهت بین وضعیت دستگاه‌ها

در این مرحله، برای مشخص کردن میزان تفاوت بین دو ویژگی قله‌های موزون p_i و p_j، 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].

۵-۳-۲ 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 دقیقا همان چیزی است که به ما در شناسایی دستگاه‌های با کارکرد غیر عادی 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 به کار دورتر می‌شود. برای این منظور، پارامتر D_a را برای تحلیل دستگاه‌ها معرفی می‌کنیم که عبارت‌است از میزان متفاوت بودن دستگاه مورد نظر با یک دستگاه نو (کلاس کاری A در جدول ۵-۲). تا زمانیکه هیچ نگهداری‌ای برای دستگاه صورت نگیرد، D_a به مرور زمان به صورت

الگوریتم ۱ تشخیص کمّی 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 $[f_{j1}...f_{j20}]$
    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
  $D_{ij} \leftarrow (sum + \sum_{k=1}^{20} p_{jk})/(cnt + len(p_j))$

یکنواخت افزایش می‌یابد.

الگوریتم RANSAC

برای یادگیری این مدل خطی و پیش‌بینی زمان سرویس دستگاه‌ها از رویکرد اجماع نمونه‌ی تصادفی[۱۶] یا به اختصار RANSAC استفاده می‌کنیم. این الگوریتم، به صورت بازگشتی، مدل خطی‌ای بر اساس داده‌های آموزشی درست می‌کند و نکته‌ی قابل توجه این الگوریتم این است که توانایی تشخیص و در نظر نگرفتن داده‌های پرت را دارد و از این رو بسیار مناسب استفاده در این مسئله است. به طور کلی روند ایجاد مدل در این الگوریتم به صورت زیر است[۲۸]:

۱. به صورت تصادفی حداقل تعداد نقاط لازم، برای پیداکردن پارامترهای مدل مشخص می‌شود.

۲. پارامترهای مدل بدست آورده می‌شوند.

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

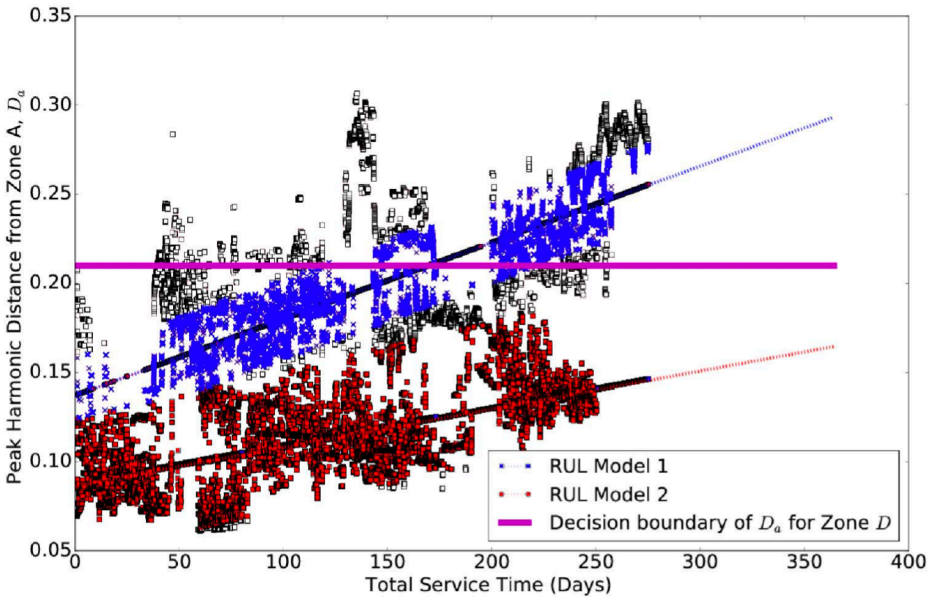
[16] RANdom SAmple Consensus (RANSAC)

۴. اگر نسبت نقاط مشمول در مدل به تعداد کل نقاط از یک مقدار از پیش تعیین‌شده‌ی τ 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 حداکثر 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 کمینه کردن 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 داده‌شده است. محور افقی زمان اندازه‌گیری‌ها بر حسب روز و محور عمودی فاصله‌ی ویژگی‌های دستگاه‌ها با ویژگی دستگاه‌های سالم و نو است. همچنین خط بنفش‌رنگ، آستانه‌ی ورود دستگاه از کلاس دستگاه‌های سالم به کلاس دستگاه‌های معیوب می‌باشد (این مقدار آستانه در شکل برابر با ۰/۲۱ 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]

۵-۴ 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 هموار کردن و سپس با انتخاب کردن 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

چالش‌ها و محدودیت‌ها

۶-۱ چالش‌ها

۶-۲ محدودیت‌ها

فصل هفتم

جمع‌بندی، نتیجه‌گیری و پیشنهادات

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, Dishant. 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] García, Salvador, Luengo, Julián, and Herrera, Francisco. *Data preprocessing in data mining*, vol. 72. Springer, 2015.

[27] Carreira-Perpinán, Miguel A. A review of mean-shift algorithms for clustering. *arXiv preprint arXiv:1503.00687*, 2015.

[28] Derpanis, Konstantinos G. Overview of the ransac algorithm. *Image Rochester NY*, 4(1):2–3, 2010.

پیوست

موضوعات مرتبط با متن گزارش پایان نامه که در یکی از گروه‌های زیر قرار می‌گیرد، در بخش پیوست‌ها آورده شوند:

۱. اثبات های ریاضی یا عملیات ریاضی طولانی.

۲. داده و اطلاعات نمونه (های) مورد مطالعه (Case Study) چنانچه طولانی باشد.

۳. نتایج کارهای دیگران چنانچه نیاز به تفصیل باشد.

۴. مجموعه تعاریف متغیرها و پارامترها، چنانچه طولانی بوده و در متن به انجام نرسیده باشد.

کد میپل

```
with(DifferentialGeometry):
with(Tensor):
DGsetup([x, y, z], M)
frame name: M
a := evalDG(D_x)
D_x
b := evalDG(-2 y z D_x+2 x D_y/z^3-D_z/z^2)
```

Abstract

This page is accurate translation from Persian abstract into English.

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

Arian Boukani

Supervisor

Dr. Hamidreza Zarandi

July 2023