





عنوان پایان نامه :

آنالیز رضایت مشتریان از کیفیت هواپیما با هوش مصنوعی

استاد راهنما:

جناب آقای دکتر افشاری

نام دانشجو :

محمد جواد صادقی

شماره دانشجویی :

۴۰۰۶۲۱۳۰۹۶

سال تحصیلی :

۱۴۰۲ بهمن

تقدیر و شکر

شکر شایان نثار ایندومان که توفیق را رفیق را هم ساخت تا این پایان نامه را به پایان برسانم. از استاد فاضل و اندیشمند جناب
آقای دکتر افشاری به عنوان استاد راهنما که همواره بنده را مورد لطف و محبت خود قرار داده اند، کمال شکر را دارم.

تقدیم به:

این پایان نامه را ضمن تشکر و سپاس بیکران و در کمال افتخار و امتنان تقدیم می نمایم به مادر عزیزم به خاطر همه ی تلاشهای محبت آمیزی که در دوران مختلف زندگی ام انجام داده است و بامهربانی چگونه زیستن را به من آموخته اند.

فهرست مطالب

عنوان	صفحه
فصل اول: بیان مساله.....	۵
۱-۱- تاریخچه هوش مصنوعی: از رویا تا واقعیت	۶
۱-۱-۱- دوران آغازین: رؤیاپردزی و ایده‌های اولیه (قبل از دهه ۱۹۵۰)	۶
۱-۱-۲- تولد هوش مصنوعی: کنفرانس دارتموث و هیجان اولیه (دهه ۱۹۵۰)	۶
۱-۱-۳- زمستان هوش مصنوعی: ناامیدی و کاهش بودجه (دهه ۱۹۷۰ و ۱۹۸۰)	۷
۱-۱-۴- رنسانس هوش مصنوعی: ظهور یادگیری ماشین و شبکه‌های عصبی	۷
۱-۲- چالش‌ها و آینده هوش مصنوعی	۸
۱-۳- داده کاوی در هوش مصنوعی : کشف گنجینه های پنهان در داده ها	۸
۱-۳-۱- تاریخچه مختصر داده‌کاوی	۸
۱-۳-۲- چرا داده کاوی مهم است؟	۸
۱-۳-۳- مراحل اصلی داده‌کاوی	۹
۱-۳-۴- تکنیک‌های اصلی داده کاوی	۹
۱-۳-۵- کاربردهای داده‌کاوی	۹
۱-۳-۶- چالش‌های داده‌کاوی	۱۰
۱-۳-۷- آینده داده‌کاوی	۱۰
۱-۴- تاریخچه یادگیری ماشین: از ایده تا واقعیت	۱۰
۱-۴-۱- ریشه‌های اولیه	۱۱
۱-۴-۲- دستاوردهای مهم	۱۲
۱-۴-۲- آینده یادگیری ماشین	۱۲
۱-۵- داده های جدولی	۱۳
۱-۵-۱- ساختار داده‌های جدولی	۱۳
۱-۵-۲- چرا داده‌های جدولی مهم هستند؟	۱۳
۱-۵-۳- انواع داده‌ها در جدول	۱۳

۱-۵-۴- کاربردهای داده‌های جدولی	۱۴
۱-۵-۵- فرمت‌های ذخیره‌سازی داده‌های جدولی	۱۴
۱-۵-۶- چالش‌های کار با داده‌های جدولی	۱۴
فصل دوم: نرم افزارهای مورد استفاده.....	۱۵
مقدمه	۱۶
۱-۲- حداقل سیستم مورد نیاز برای نصب pycharm	۱۶
۲-۲- آموزش اولیه	۱۶
۱-۲-۲- ایجاد پروژه پایتون	۱۶
۲-۳- آشنایی با انواع محیط های پایتون	۲۷
فصل سوم: تحلیل و طراحی نرم افزار.....	۲۹
مقدمه	۳۰
۳-۱ افزودن کتابخانه های لازم (import)	۳۰
۳-۲- دانلود دیتاست (dataset)	۳۰
فصل چهارم: نتیجه گیری و پیشنهادات.....	۶۱
۴-۱- نتیجه گیری	۶۲
۴-۲- پیشنهادات	۶۲
منابع	۷۵

فهرست شکل‌ها

عنوان	صفحه
شکل ۱-۲- قابلیت های کلی نرم افزار	۱۷
شکل ۲-۲- انتخاب دستور file	۱۷
شکل ۳-۲- ایجاد فایل	۱۸
شکل ۴-۲- بارگذاری فایل از دکستاپ	۱۸
شکل ۵-۲- نصب کتابخانه بر روی پایتون	۱۹
شکل ۶-۲- کتابخانه numpy	۲۰
شکل ۷-۲- کتابخانه پانداس	۲۲
شکل ۸-۲- کتابخانه matplotlib	۲۳
شکل ۹-۲- کتابخانه keras	۲۶
شکل ۱-۳- import library	۳۰
شکل ۲-۳- بررسی تمیز بودن داده	۳۴
شکل ۳-۳- مصور سازی ستون کشور	۳۶
شکل ۴-۳- مصور سازی نمودار کشور	۳۶
شکل ۵-۳- مصور سازی نمودار مسیر	۳۷
شکل ۶-۳- مصور سازی نمودار مسیر	۳۷
شکل ۷-۳- مصور سازی نمودار نوع صندلی	۳۷
شکل ۸-۳- مصور سازی نمودار نوع صندلی	۳۸
شکل ۹-۳- مصور سازی نمودار ستاره های هواپیما	۳۸
شکل ۱۰-۳- مصور سازی نمودار ستاره های هواپیما	۳۸
شکل ۱۱-۳- مصور سازی نمودار نوع سفر	۳۹
شکل ۱۲-۳- مصور سازی نمودار نوع سفر	۳۹
شکل ۱۳-۳- جدا سازی زمان	۴۰
شکل ۱۴-۳- جدا سازی پسوند	۴۰
شکل ۱۵-۳- تبدیل ماه	۴۰
شکل ۱۶-۳- حذف زمان	۴۱
شکل ۱۷-۳- برچسب گذاری	۴۲
شکل ۱۸-۳- جای گذاری به جای مقادیر رشته	۴۳

شکل ۳-۱۹-dtype	۴۳
شکل ۳-۲۰- تغییر دادن dtype	۴۳
شکل ۳-۲۱- حذف سطر نا کارآمد	۴۴
شکل ۳-۲۲- نمایش ضرایب همبستگی	۴۵
شکل ۳-۲۳- ایجاد متغیر هدف	۴۵
شکل ۳-۲۴- استفاده از تابع minmax	۴۶
شکل ۳-۲۵- آماده کردن دیتا برای آموزش و بررسی درستی	۴۶
شکل ۳-۲۶- آموزش الگوریتم logistic regression	۴۹
شکل ۳-۲۷- آموزش الگوریتم درخت تصمیم	۵۱
شکل ۳-۲۸- پیاده سازی الگوریتم تقویت گرادیان	۵۳
شکل ۳-۲۹- پیاده سازی الگوریتم جنگل تصادفی	۵۳
شکل ۳-۳۰- پیاده سازی svm	۵۵
شکل ۳-۳۱- پیاده سازی گزارش طبقه بندی	۵۷
شکل ۳-۳۲- پیاده سازی شبکه عصبی مصنوعی	۵۹
شکل ۳-۳۳- ۵ آموزش برتر الگوریتم	۶۰
شکل ۴-۱- مقایسه نهایی الگوریتم های یادگیری ماشین	۶۲

فصل اول

بیان مساله

مقدمه

هوش مصنوعی امروزه به بخشی جدایی‌ناپذیر از زندگی روزمره ما تبدیل شده است. از گوشی‌های هوشمند گرفته تا خودروها، سیستم‌های بهداشتی و حتی خانه‌های هوشمند، همه تحت تأثیر این فناوری قرار دارند. کاربردهای هوش مصنوعی بسیار گسترده است و هر روز بر دامنه آن‌ها افزوده می‌شود. از جمله این کاربردها می‌توان به دستیارهای صوتی هوشمند، سیستم‌های تشخیص چهره، خودروهای خودران، تشخیص بیماری‌ها، تحلیل داده‌های بزرگ، و شخصی‌سازی خدمات مشتری اشاره کرد. هوش مصنوعی با یادگیری از داده‌های عظیم، الگوها را شناسایی کرده و تصمیمات هوشمندانه‌ای اتخاذ می‌کند که زندگی ما را آسان‌تر و کارآمدتر می‌سازد.

در ادامه ابتدا به تاریخچه‌های چند مورد از مباحث اصلی اشاره می‌کنم و در فصول بعد با پروژه بیشتر آشنا می‌شویم.

۱-۱- تاریخچه هوش مصنوعی: از رویا تا واقعیت

هوش مصنوعی Artificial Intelligence یا AI، رشته‌ای است که به ساخت ماشین‌هایی هوشمند می‌پردازد که قادر به انجام کارهایی هستند که معمولاً نیاز به هوش انسانی دارند. از بازی شطرنج گرفته تا تشخیص بیماری‌ها، هوش مصنوعی در بسیاری از جنبه‌های زندگی ما نفوذ کرده است. اما این فناوری از کجا آمده و چه مسیری را طی کرده است؟ در ادامه به بررسی تاریخچه جذاب هوش مصنوعی می‌پردازیم.

۱-۱-۱- دوران آغازین: رؤیاپردازی و ایده‌های اولیه (قبل از دهه ۱۹۵۰)

افسانه‌ها و ادبیات: ایده ایجاد موجودات مصنوعی هوشمند از دیرباز در افسانه‌ها و ادبیات بشر وجود داشته است. از غول‌های آهنین در اساطیر یونان تا ربات‌های داستان‌های علمی تخیلی، این ایده همیشه ذهن بشر را به خود مشغول کرده است.

فیلسوفان و ریاضیدانان: فیلسوفان و ریاضیدانان نیز به بررسی ماهیت هوش و امکان شبیه‌سازی آن پرداخته‌اند. آلن تورینگ، ریاضیدان انگلیسی، با ارائه تست تورینگ در سال ۱۹۵۰، معیاری برای سنجش هوش ماشینی ارائه داد.

۱-۱-۲- تولد هوش مصنوعی: کنفرانس دارتموث و هیجان اولیه (دهه ۱۹۵۰)

کنفرانس دارتموث: در سال ۱۹۵۶، گروهی از دانشمندان در کنفرانس دارتموث گرد هم آمدند و اصطلاح "هوش مصنوعی" را برای اولین بار به کار بردند. این کنفرانس نقطه عطفی در تاریخ هوش مصنوعی بود و منجر به ایجاد هیجان و امیدواری فراوان به آینده این رشته شد.

پیشرفت‌های اولیه: در دهه ۱۹۵۰ و ۱۹۶۰، پژوهشگران به موفقیت‌هایی در زمینه حل مسائل ریاضی، ترجمه ماشینی و بازی‌های ساده دست یافتند. این موفقیت‌ها باعث شد تا بسیاری به این باور برسند که ساخت ماشین‌های هوشمند در آینده‌ای نزدیک ممکن خواهد بود.

۱-۱-۳- زمستان هوش مصنوعی: ناامیدی و کاهش بودجه (دهه ۱۹۷۰ و ۱۹۸۰)

محدودیت‌های اولیه: با وجود پیشرفت‌های اولیه، هوش مصنوعی با چالش‌های جدی روبرو شد. بسیاری از مشکلات پیچیده‌تر از آنچه تصور میشد بودند و الگوریتم‌های موجود قادر به حل آن‌ها نبودند.

کاهش بودجه: به دلیل عدم تحقق وعده‌های اولیه، دولت‌ها و مؤسسات سرمایه‌گذاری خود را از پروژه‌های هوش مصنوعی کاهش دادند. این دوره که "زمستان هوش مصنوعی" نامیده می‌شود، منجر به کاهش فعالیت‌ها و تحقیقات در این حوزه شد.

۱-۱-۴- رنسانس هوش مصنوعی: ظهور یادگیری ماشین و شبکه‌های عصبی (دهه ۱۹۹۰ تاکنون)

یادگیری ماشین: با افزایش قدرت محاسباتی رایانه‌ها و دسترسی به داده‌های عظیم، یادگیری ماشین به عنوان روشی قدرتمند برای ساخت سیستم‌های هوشمند مطرح شد.

شبکه‌های عصبی مصنوعی: الهام گرفته از مغز انسان، شبکه‌های عصبی مصنوعی توانایی یادگیری الگوها و تصمیم‌گیری‌های پیچیده را به ماشین‌ها بخشیدند.

کاربردهای گسترده: هوش مصنوعی در دهه‌های اخیر در بسیاری از زمینه‌ها از جمله پزشکی، مالی، خودروسازی، سرگرمی و ... کاربرد پیدا کرده است. دستیارهای صوتی، خودروهای خودران، سیستم‌های تشخیص چهره و بسیاری از فناوری‌های مدرن دیگر، محصول پیشرفت‌های هوش مصنوعی هستند.

۱-۲- چالش‌ها و آینده هوش مصنوعی

اخلاقیات: با پیشرفت هوش مصنوعی، سوالات اخلاقی مهمی مطرح می‌شود. برای مثال، استفاده از هوش مصنوعی در سلاح‌ها، حریم خصوصی داده‌ها و تأثیر آن بر بازار کار از جمله مسائلی هستند که باید به دقت بررسی شوند.

توسعه پایدار: توسعه هوش مصنوعی باید به گونه‌ای باشد که منافع همه انسان‌ها را تأمین کند و به کاهش نابرابری‌ها کمک کند.

آینده: آینده هوش مصنوعی بسیار روشن به نظر می‌رسد. با پیشرفت فناوری و افزایش حجم داده‌ها، می‌توان انتظار داشت که هوش مصنوعی در آینده نقش پررنگ‌تری در زندگی ما ایفا کند.

در نتیجه تاریخچه هوش مصنوعی، پر فراز و نشیب بوده است. از رؤیاهای اولیه تا زمستان هوش مصنوعی و رنسانس کنونی، این فناوری مسیری طولانی را طی کرده است. با وجود چالش‌ها، هوش مصنوعی به یکی از مهم‌ترین فناوری‌های قرن حاضر تبدیل شده است و تأثیر آن بر زندگی بشر روز به روز بیشتر می‌شود.

۱-۳- داده کاوی در هوش مصنوعی: کشف گنجینه‌های پنهان در داده‌ها

داده کاوی Data Mining یکی از شاخه‌های مهم هوش مصنوعی است که به استخراج اطلاعات مفید، الگوها و روابط پنهان در حجم عظیمی از داده‌ها می‌پردازد. این فرآیند، به ما کمک می‌کند تا از داده‌های خام و بی‌ساختار، بینش‌های ارزشمندی کسب کرده و تصمیم‌گیری‌های بهتری بگیریم.

۱-۳-۱- تاریخچه مختصر داده کاوی

ریشه‌های داده کاوی را می‌توان در تلاش‌های اولیه برای تجزیه و تحلیل داده‌ها، از جمله آمار و یادگیری ماشین، جستجو کرد. با این حال، داده کاوی به عنوان یک حوزه مستقل در دهه ۱۹۹۰ شکل گرفت و با رشد روزافزون حجم داده‌ها، اهمیت آن نیز افزایش یافت.

۱-۳-۲- چرا داده کاوی مهم است؟

کشف الگوهای پنهان: داده کاوی به ما کمک می‌کند تا الگوها و روابط پیچیده‌ای را در داده‌ها شناسایی کنیم که با روش‌های سنتی قابل مشاهده نیستند.

پیش‌بینی رویدادهای آینده: با استفاده از داده‌های گذشته، می‌توانیم رویدادهای آینده را با دقت بیشتری پیش‌بینی کنیم.

بهبود تصمیم‌گیری: بینش‌هایی که از طریق داده‌کاوی به دست می‌آید، به ما کمک می‌کند تا تصمیمات آگاهانه‌تری بگیریم.

کشف دانش جدید: داده‌کاوی می‌تواند به کشف دانش جدید و نوآوری در حوزه‌های مختلف منجر شود.

۱-۳-۳- مراحل اصلی داده‌کاوی

- جمع‌آوری داده‌ها: گردآوری داده‌ها از منابع مختلف و یکپارچه‌سازی آن‌ها.
- پیش‌پردازش داده‌ها: تمیز کردن، تبدیل و آماده‌سازی داده‌ها برای تحلیل.
- کاوش داده‌ها: استفاده از الگوریتم‌ها و تکنیک‌های مختلف برای کشف الگوها و روابط.
- ارزیابی نتایج: ارزیابی مدل‌ها و نتایج حاصل از تحلیل داده‌ها.
- تفسیر و ارائه نتایج: ارائه نتایج به صورت قابل فهم برای تصمیم‌گیران.

۱-۳-۴- تکنیک‌های اصلی داده‌کاوی

- طبقه‌بندی: تقسیم داده‌ها به گروه‌های مختلف بر اساس ویژگی‌های مشترک.
- خوشه‌بندی: گروه‌بندی داده‌ها به صورت خودکار بر اساس شباهت‌ها.
- رگرسیون: پیش‌بینی یک مقدار پیوسته بر اساس سایر ویژگی‌ها.
- تحلیل انجمنی: کشف روابط بین آیتم‌ها در یک مجموعه داده.
- کشف قوانین انجمنی: کشف قوانین "اگر-آنگاه" بین متغیرها.

۱-۳-۵- کاربردهای داده‌کاوی

داده‌کاوی در حوزه‌های مختلفی مانند:

- تجارت: پیش‌بینی رفتار مشتریان، شخصی‌سازی محصولات، تشخیص تقلب

- پزشکی: کشف داروهای جدید، تشخیص زودهنگام بیماری‌ها، تحلیل داده‌های ژنومی
- مالی: تشخیص تقلب، ارزیابی ریسک، توصیه‌های سرمایه‌گذاری
- بازاریابی: تحلیل رفتار مشتریان، هدف‌گذاری تبلیغات
- علم: کشف الگوهای پنهان در داده‌های علمی

۱-۳-۶- چالش‌های داده‌کاوی

حجم عظیم داده‌ها: مدیریت و پردازش داده‌های حجیم نیازمند زیرساخت‌های قدرتمند است. کیفیت داده‌ها: داده‌های ناقص، ناسازگار یا دارای نویز می‌توانند نتایج تحلیل را تحت تأثیر قرار دهند.

پیچیدگی الگوریتم‌ها: برخی از الگوریتم‌های داده‌کاوی بسیار پیچیده هستند و نیاز به تخصص بالایی دارند.

حریم خصوصی داده‌ها: استفاده از داده‌های شخصی با چالش‌های اخلاقی و قانونی همراه است.

۱-۳-۷- آینده داده‌کاوی

با پیشرفت فناوری‌های هوش مصنوعی و افزایش حجم داده‌ها، اهمیت داده‌کاوی روزافزون خواهد شد. یادگیری ماشین، یادگیری عمیق و هوش مصنوعی کوانتومی، به عنوان برخی از فناوری‌های نوظهور، به تحول داده‌کاوی کمک خواهند کرد.

داده‌کاوی، ابزاری قدرتمند برای کشف دانش پنهان در داده‌هاست. با استفاده صحیح از این ابزار، می‌توانیم به تصمیم‌گیری‌های بهتر، نوآوری و پیشرفت در حوزه‌های مختلف کمک کنیم.

۱-۴- تاریخچه یادگیری ماشین: از ایده تا واقعیت

یادگیری ماشین، زیرمجموعه‌ای از هوش مصنوعی است که به کامپیوترها اجازه می‌دهد بدون برنامه‌نویسی صریح، از روی داده‌ها یاد بگیرند و بهبود یابند. این فناوری در دهه‌های اخیر پیشرفت شگرفی کرده و در بسیاری از جنبه‌های زندگی ما نفوذ کرده است. اما تاریخچه یادگیری ماشین به کجا برمی‌گردد؟

۱-۴-۱- ریشه‌های اولیه

دهه ۱۹۴۰ و ۱۹۵۰:

مدل‌های اولیه نورون: اولین قدم‌ها به سوی یادگیری ماشین با توسعه مدل‌های ریاضی ساده‌ای از نورون‌های زیستی برداشته شد. این مدل‌ها به عنوان پایه و اساس شبکه‌های عصبی مصنوعی شناخته می‌شوند.

تست تورینگ: آلن تورینگ، ریاضیدان انگلیسی، با ارائه تست تورینگ در سال ۱۹۵۰، معیاری برای سنجش هوش ماشینی ارائه داد.

دهه ۱۹۵۰ و ۱۹۶۰:

کنفرانس دارتموث: در سال ۱۹۵۶، کنفرانس دارتموث به عنوان نقطه عطفی در تاریخ هوش مصنوعی و یادگیری ماشین شناخته می‌شود. در این کنفرانس، اصطلاح "هوش مصنوعی" برای اولین بار به کار رفت.

پرسپترون: فرانک روزنبلت، پرسپترون را به عنوان اولین شبکه عصبی مصنوعی معرفی کرد.

زمستان یادگیری ماشین

دهه ۱۹۷۰ و ۱۹۸۰: با وجود پیشرفت‌های اولیه، یادگیری ماشین با چالش‌های جدی روبرو شد. محدودیت‌های محاسباتی و عدم دستیابی به نتایج قابل انتظار، منجر به کاهش بودجه و علاقه به این حوزه شد. این دوره، "زمستان یادگیری ماشین" نامیده می‌شود.

رنسانس یادگیری ماشین

دهه ۱۹۹۰ تاکنون: با افزایش قدرت محاسباتی رایانه‌ها، دسترسی به داده‌های عظیم و توسعه الگوریتم‌های جدید، یادگیری ماشین وارد دوران رنسانس خود شد.

یادگیری عمیق: شبکه‌های عصبی مصنوعی با لایه‌های متعدد (شبکه‌های عصبی عمیق) توانایی یادگیری الگوهای پیچیده‌تری را پیدا کردند و در بسیاری از حوزه‌ها مانند پردازش تصویر و پردازش زبان طبیعی به موفقیت‌های چشمگیری دست یافتند.

کاربردهای گسترده: یادگیری ماشین امروزه در بسیاری از صنایع و کاربردها از جمله تشخیص تصویر، پردازش زبان طبیعی، خودروهای خودران، توصیه‌گرهای شخصی و بسیاری دیگر مورد استفاده قرار می‌گیرد.

۱-۴-۲- دستاوردهای مهم

الگوریتم‌های یادگیری ماشین: الگوریتم‌های متنوعی مانند رگرسیون خطی، رگرسیون لجستیک، درخت تصمیم‌گیری، جنگل تصادفی، ماشین بردار پشتیبان و شبکه‌های عصبی مصنوعی توسعه یافته‌اند.

یادگیری تقویتی: این روش به ماشین‌ها اجازه می‌دهد تا با تعامل با محیط یاد بگیرند و تصمیمات بهتری بگیرند.

یادگیری بدون نظارت: این روش به ماشین‌ها اجازه می‌دهد بدون برچسب‌گذاری داده‌ها، الگوها و ساختارهای نهفته در داده‌ها را کشف کنند.

۱-۴-۲- آینده یادگیری ماشین

آینده یادگیری ماشین بسیار روشن به نظر می‌رسد. با پیشرفت فناوری‌های سخت‌افزاری و نرم‌افزاری، می‌توان انتظار داشت که یادگیری ماشین در آینده به طور فزاینده‌ای در زندگی ما نفوذ کند. برخی از حوزه‌های مهم در آینده یادگیری ماشین عبارتند از:

هوش مصنوعی عمومی: توسعه سیستم‌های هوشمندی که قادر به انجام هر کاری که یک انسان می‌تواند انجام دهد.

یادگیری ماشین قابل تفسیر: ایجاد مدل‌های یادگیری ماشینی که تصمیمات خود را قابل توضیح کنند.

یادگیری ماشینی همگانی: گسترش دسترسی به ابزارها و منابع یادگیری ماشین برای همه.

در کل، یادگیری ماشین مسیری طولانی را طی کرده است و همچنان در حال تحول و پیشرفت است. این فناوری به عنوان یکی از مهم‌ترین فناوری‌های قرن بیست و یکم شناخته می‌شود و تأثیر آن بر زندگی بشر روز به روز بیشتر می‌شود.

۱-۵- داده های جدولی

داده های جدولی یکی از رایج ترین و شناخته شده ترین فرمت های ذخیره و نمایش داده ها هستند. این نوع داده ها به صورت یک جدول سازماندهی می شوند که در آن هر سطر نشان دهنده یک مشاهده یا نمونه و هر ستون نشان دهنده یک ویژگی یا متغیر است.

۱-۵-۱- ساختار داده های جدولی

سطرها (ردیف ها): هر سطر در جدول یک مشاهده یا نمونه منفرد را نشان می دهد. مثلاً در یک جدول حاوی اطلاعات مشتریان، هر سطر مربوط به یک مشتری خاص است.

ستون ها: هر ستون در جدول یک ویژگی یا متغیر را نشان می دهد. مثلاً در همان جدول مشتریان، ستون هایی مانند نام، سن، جنسیت و میزان خرید وجود دارد.

مثال ساده یک داده جدولی

۱-۵-۲- چرا داده های جدولی مهم هستند؟

سادگی و درک آسان: ساختار جدولی برای انسان ها بسیار قابل فهم است و به راحتی می توان داده ها را بررسی و تحلیل کرد.

کاربرد گسترده: داده های جدولی در بسیاری از حوزه ها مانند علوم اجتماعی، اقتصاد، پزشکی، مهندسی و ... مورد استفاده قرار می گیرند.

سازگاری با ابزارهای تحلیل داده: اکثر ابزارها و نرم افزارهای تحلیل داده (مانند اکسل، SPSS، R، پایتون و ...) از داده های جدولی پشتیبانی می کنند.

۱-۵-۳- انواع داده ها در جدول

داده های کمی: داده هایی که مقدار عددی دارند و می توان عملیات ریاضی روی آن ها انجام داد (مثلاً سن، وزن، درآمد).

داده های کیفی: داده هایی که به صورت دسته ها یا گروه ها طبقه بندی می شوند (مثلاً جنسیت، رنگ مو، شهر).

۱-۵-۴- کاربردهای داده‌های جدولی

تحلیل آماری: محاسبه میانگین، انحراف استاندارد، همبستگی و ...

مدل‌سازی پیش‌بینی: ساخت مدل‌هایی برای پیش‌بینی آینده (مثلاً پیش‌بینی فروش، قیمت سهام)

طبقه‌بندی: تقسیم داده‌ها به گروه‌های مختلف (مثلاً طبقه‌بندی مشتریان به گروه‌های مختلف بر اساس رفتار خرید)

خوشه‌بندی: گروه‌بندی داده‌ها به صورت خودکار بر اساس شباهت‌ها (مثلاً خوشه‌بندی مشتریان بر اساس ویژگی‌های مشترک)

۱-۵-۵- فرمت‌های ذخیره‌سازی داده‌های جدولی

CSV (Comma Separated Values): فرمتی ساده و متنی که از کاما برای جدا کردن مقادیر استفاده می‌کند.

Excel: نرم‌افزار اکسل یکی از محبوب‌ترین ابزارها برای ایجاد و ویرایش داده‌های جدولی است.

SQL: زبان ساختار یافته پرس‌وجو برای مدیریت پایگاه‌های داده رابطه‌ای.

JSON: فرمت متنی سبک‌وزن برای تبادل داده‌ها.

۱-۵-۶- چالش‌های کار با داده‌های جدولی

داده‌های ناقص: وجود داده‌های گم‌شده یا نادرست می‌تواند بر نتایج تحلیل تأثیر بگذارد.

داده‌های ناسازگار: داده‌هایی که با هم تناقض دارند، می‌توانند باعث ایجاد خطا شوند.

حجم بالای داده‌ها: پردازش حجم عظیمی از داده‌ها نیازمند منابع محاسباتی قوی است.

فصل دوم

نرم افزارهای مورد استفاده

مقدمه

باتوجه به اینکه داده کاوی با زبان برنامه نویسی پایتون و R آمیخته شده است . اکثر کمپانی های بزرگ تمایل به استفاده از زبان پایتون به دلیل ساده بودن و پشتیبانی از شی گرایی و همینطور اینکه این زبان از محیط Jupyter notebook پشتیبانی میکند علاوه بر قدرت پایتون در زمینه هوش مصنوعی این زبان برنامه نویسی چند منظوره نیز میباشد و میتوان از این زبان برای طراحی وب ساخت اپلیکیشن و.... استفاده نمود.

Pycharm

کد نویسی پایتون در محیط های برنامه نویسی زیادی ساپورت میشود که معروف ترین این محیط visual studio,pycharm میباشد ما در اینجا به وسیله محیط برنامه نویسی pycharm پیاده سازی نمودیم.

Pycharm بر اساس IntelliJ IDEA برای توسعه پایتون توزیع شده است. این ابزار در می ۲۰۰۹ برای منتشر شده بود، در ۲۰۲۳ نسخه نهایی این برنامه معرفی گردید.

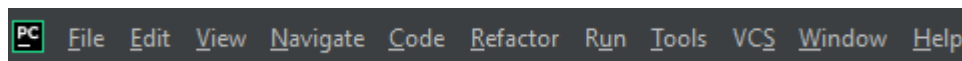
۲-۱- حداقل سیستم مورد نیاز برای نصب pycharm

- ویندوز 7-8-10-11 نسخه ۳۲ یا ۶۴ بیتی
- حداقل ۴ گیگابایت رم ، ۸ گیگابایت مقدار توصیه شده
- 2 گیگابایت فضای خالی در هارد دیسک، مقدار توصیه شده ۴ گیگابایت
- صفحه نمایش با حداقل رزولوشن 700×1280
- برای شبیه سازی سریع تر، بهتر است از CPU های اینتل ۶۴ بیتی استفاده شود
- ترجیحا پردازنده چند هسته ای باشد
- به دلیل سنگین بودن نرم افزار در صورت دسترسی به حافظه ssd بهتر اجرا میگردد.

۲-۲- آموزش اولیه

۲-۲-۱- ایجاد پروژه پایتون

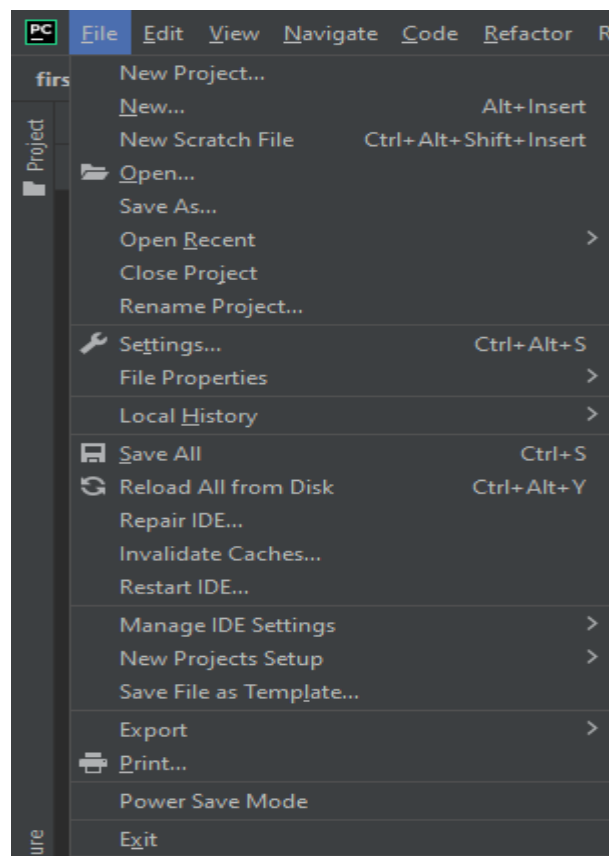
ابتدا با باز کردن محیط pycharm خود، صبر میکنیم تا نرم افزار اجرا گردد سپس در منو بالایی نرم افزار قابلیت های کلی نرم افزار از قبیل، file , edit , view,navigate,code,refactor,run,tool,vcs, مشاهده میکنید.



شکل ۱-۲- قابلیت های کلی نرم افزار

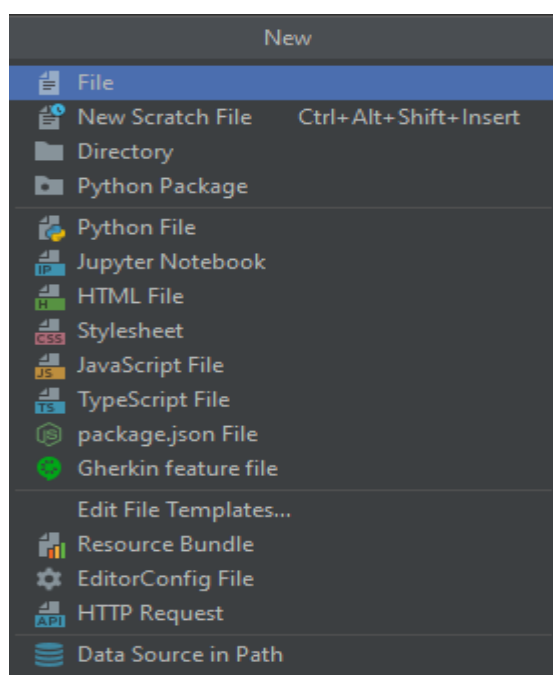
سپس برنامه را با استفاده از tap فایل ایجاد میکنیم توسط مراحل زیر نمایش میدهیم:

۱. انتخاب file



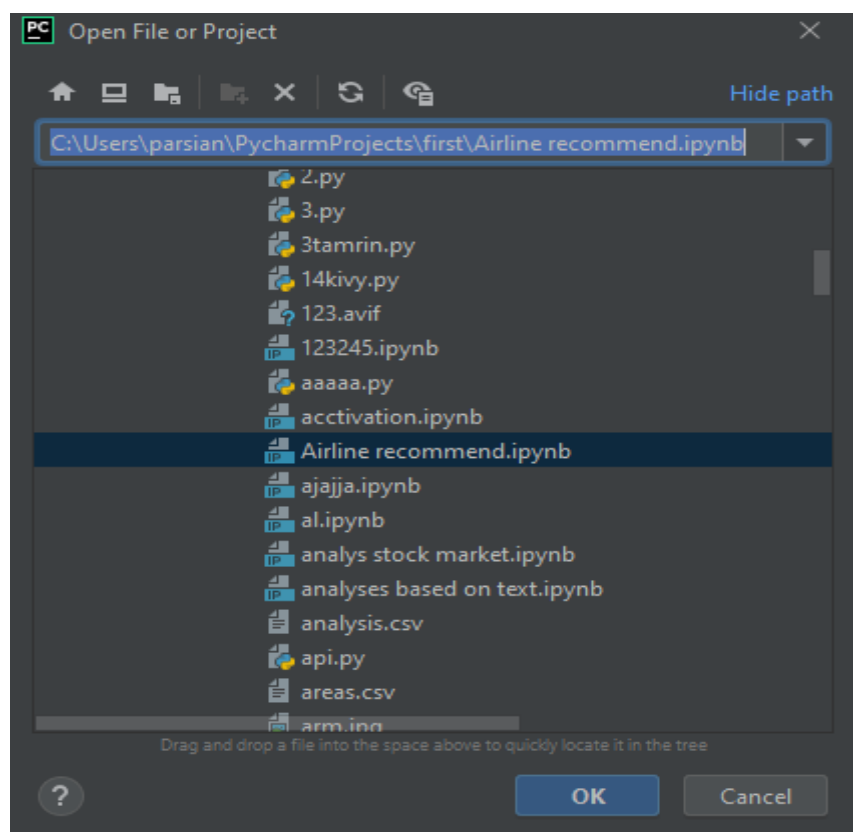
شکل ۲-۲- انتخاب دستور file

۲. در این مرحله ما میتوانیم با دستور new فایلی را ایجاد کنیم که نوع آن به شرح زیر است یا میتوانیم فایلی را از درون فایل های دستگاه بار گذاری کنیم.



شکل ۲-۳- ایجاد فایل

ما در اینجا دو حالت فایل میتوانیم ایجاد کنیم فایل ساده پایتون با پسوند py. یا فایل jupyter notebook با پسوند ipynb.



شکل ۲-۴- بارگذاری فایل از دکستاپ

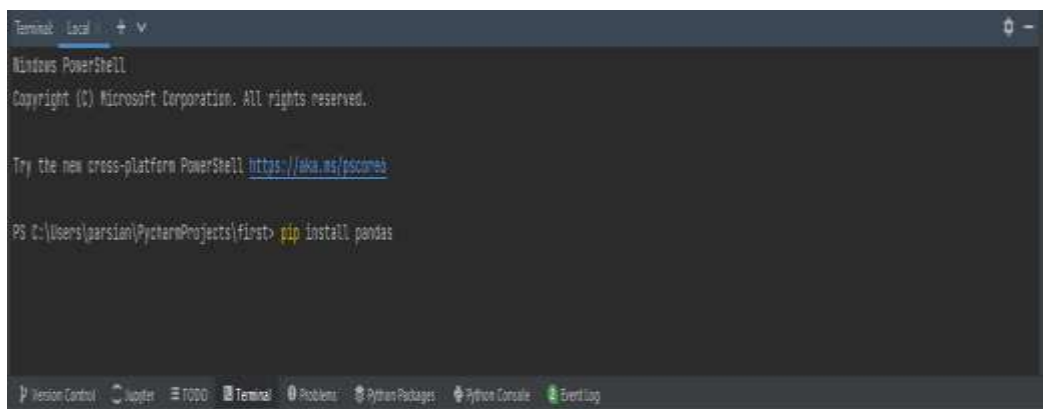
طریقه نصب کتابخانه در پایتون

برای اضافه کردن کتابخانه در pycharm چندین راه وجود دارد که ساده ترین این راه استفاده از دستور pip میباشد مراحل کار به شرح زیر می باشد.

ابتدا در منو بار پایین صفحه دستور ترمینال را انتخاب میکنید سپس وارد ترمینال که شده اید دستور زیر را انتخاب می کنید.

Pip Install -----

به جای بخش خط چین کتابخانه مورد نظر را مینویسد تا نصب شود توجه داشته باشد که برای نصب دسترسی به اینترنت الزامیست



شکل ۲-۵- نصب کتابخانه بر روی پایتون

کتابخانه NumPy ابزاری قدرتمند برای محاسبات عددی در پایتون:

NumPy یا Numerical Python یکی از قدرتمندترین و پرکاربردترین کتابخانه های پایتون برای انجام محاسبات عددی است. این کتابخانه به شما امکان می دهد آرایه های چندبعدی (arrays) بسیار بزرگی را با سرعت و کارایی بالا ایجاد، دستکاری و محاسبات پیچیده روی آنها انجام دهید).

چرا NumPy مهم است؟

- سرعت بالا: عملیات روی آرایه های NumPy به مراتب سریع تر از لیست های معمولی پایتون است.
- کارایی بالا: NumPy امکان انجام عملیات جبری خطی، تبدیل فوریه و بسیاری از محاسبات پیچیده دیگر را به صورت بسیار کارآمد فراهم می کند.
- سادگی استفاده: سینتکس NumPy بسیار ساده و شبیه به ماتریس های ریاضی است.

- پشتیبانی گسترده: NumPy پایه و اساس بسیاری از کتابخانه‌های علمی و یادگیری ماشین در پایتون مانند SciPy, Pandas, Matplotlib و TensorFlow است.

کاربردهای NumPy

- علم داده: تحلیل داده‌ها، آماده‌سازی داده‌ها برای مدل‌سازی، محاسبات آماری و ...
- یادگیری ماشین: ایجاد و دستکاری ماتریس‌ها و بردارها، پیاده‌سازی الگوریتم‌های یادگیری ماشین
- پردازش تصویر: نمایش تصاویر به صورت آرایه‌های عددی و انجام عملیات پردازش تصویر
- شبیه‌سازی: ایجاد مدل‌های ریاضی و شبیه‌سازی سیستم‌های مختلف

مفاهیم کلیدی در NumPy

- آرایه: ndarray ساختار داده اصلی NumPy که یک جدول چندبعدی از عناصر همگن (معمولاً اعداد) است.
- شکل (shape): ابعاد یک آرایه را نشان می‌دهد.
- نوع داده: dtype نوع داده عناصر یک آرایه را مشخص می‌کند مثلاً int, float, complex.
- اندیس‌گذاری (indexing): دسترسی به عناصر خاص یک آرایه.
- برش زدن (slicing): استخراج بخش‌هایی از یک آرایه.
- عملیات جبری: جمع، تفریق، ضرب، تقسیم و سایر عملیات ریاضی روی آرایه‌ها.
- توابع جهانی (universal functions): توابعی که عملیات ریاضی را روی هر عنصر یک آرایه اعمال می‌کنند.



شکل ۲-۶- کتابخانه numpy

پانداس: ابزار قدرتمند برای تحلیل داده در پایتون

پانداس یکی از محبوب‌ترین و پرکاربردترین کتابخانه‌های پایتون برای تحلیل داده است. این کتابخانه بر پایه NumPy ساخته شده و ساختارهای داده‌ای کارآمدی را برای کار با داده‌های ساخت‌یافته فراهم می‌کند. با استفاده از پانداس، می‌توانید به سادگی داده‌ها را وارد، تمیز، دستکاری، تجزیه و تحلیل و به شکل‌های مختلف نمایش دهید.

چرا پانداس؟

- سرعت و کارایی بالا: پانداس به دلیل استفاده از NumPy، عملیات روی داده‌ها را با سرعت بسیار بالایی انجام می‌دهد.
- سادگی استفاده: سینتکس پانداس بسیار شبیه به جداول و داده‌های ساخت‌یافته در Excel است و به همین دلیل یادگیری و استفاده از آن آسان است.
- انعطاف‌پذیری بالا: پانداس قابلیت کار با انواع مختلف داده‌ها، از جمله داده‌های عددی، رشته‌ای و تاریخ را دارد.
- ابزارهای قدرتمند: پانداس مجموعه‌ای غنی از ابزارها برای انجام عملیات مختلف روی داده‌ها، از جمله دسته‌بندی، فیلتر کردن، ادغام، گروه‌بندی و تجمیع داده‌ها را فراهم می‌کند.
- توسعه‌پذیری بالا: پانداس به راحتی با سایر کتابخانه‌های پایتون مانند NumPy، Matplotlib و Scikit-learn ادغام می‌شود.

ساختارهای داده‌ای اصلی در پانداس

- سری: Series یک آرایه یک بعدی برچسب‌گذاری شده است که می‌تواند انواع مختلف داده‌ها را در خود نگه دارد.
- دیتافریم: DataFrame یک جدول دو بعدی برچسب‌گذاری شده است که هر ستون آن می‌تواند نوع داده متفاوتی داشته باشد. دیتافریم‌ها شبیه به جداول در Excel یا DataFrame‌ها در R هستند.

کاربردهای پانداس

- وارد کردن داده‌ها: خواندن داده‌ها از فایل‌های CSV، Excel، پایگاه داده‌ها و سایر منابع داده‌ای.

- تمیز کردن داده‌ها: حذف مقادیر گم شده، تشخیص و حذف داده‌های پرت، و استانداردسازی داده‌ها.



شکل ۲-۷- کتابخانه پانداس

Matplotlib ابزاری قدرتمند برای تجسم داده در پایتون:

Matplotlib یکی از محبوب‌ترین و پرکاربردترین کتابخانه‌های پایتون برای ایجاد انواع مختلف نمودارها و گراف‌ها است. این کتابخانه به شما امکان می‌دهد داده‌های خود را به صورت بصری نمایش دهید و بینش‌های جدیدی از آن به دست آورید. با استفاده از Matplotlib می‌توانید نمودارهای ساده مانند خطی، میله‌ای و پراکندگی تا نمودارهای پیچیده‌تر مانند سه‌بعدی، قطبی و نقشه‌ها را ایجاد کنید.

چرا Matplotlib؟

- انعطاف‌پذیری بالا: Matplotlib به شما اجازه می‌دهد تا کنترل کاملی بر ظاهر و محتوای نمودارهای خود داشته باشید.
- سادگی استفاده: این کتابخانه دارای یک رابط کاربری ساده و شهودی است که به شما امکان می‌دهد به سرعت نمودارهای دلخواه خود را ایجاد کنید.
- پشتیبانی گسترده: Matplotlib به طور گسترده مورد استفاده قرار می‌گیرد و جامعه کاربری بزرگی دارد. این بدان معناست که شما می‌توانید به راحتی به منابع و آموزش‌های زیادی دسترسی پیدا کنید.
- سازگاری با سایر کتابخانه‌ها: Matplotlib به خوبی با سایر کتابخانه‌های پایتون مانند NumPy و Pandas کار می‌کند.

کاربردهای Matplotlib

- علم داده: نمایش توزیع داده‌ها، مقایسه گروه‌ها، و بررسی روابط بین متغیرها.
- یادگیری ماشین: تجسم داده‌های آموزشی، ارزیابی مدل‌ها و تحلیل نتایج.

- مهندسی: نمایش داده‌های مهندسی مانند سیگنال‌ها، تصاویر و داده‌های حسگر.
- تحلیل مالی: نمایش روندهای بازار، مقایسه عملکرد دارایی‌ها و تحلیل ریسک.

مفاهیم کلیدی در Matplotlib

- Figure: یک ظرف است که در آن تمام عناصر نمودار (مانند محورها، خطوط، و برچسب‌ها) قرار می‌گیرند.
- Axes: یک سیستم مختصات است که در داخل Figure قرار دارد و برای نمایش داده‌ها استفاده می‌شود.
- Plot: یک خط، نقطه یا ناحیه‌ای است که داده‌ها را در Axes نمایش می‌دهد.
- انواع نمودارهای قابل ایجاد با Matplotlib
- خطی: برای نمایش تغییرات یک متغیر در طول زمان
- میله‌ای: برای مقایسه مقادیر بین گروه‌های مختلف
- پراکندگی: برای نمایش رابطه بین دو متغیر عددی
- هیستوگرام: برای نمایش توزیع یک متغیر عددی
- دایره‌ای: برای نمایش نسبت‌های مختلف یک کل
- بسیاری دیگر...



شکل ۲-۸ - کتابخانه matplotlib

Scikit-learn: ابزار قدرتمند برای یادگیری ماشین در پایتون

Scikit-learn یکی از پرکاربردترین و محبوب‌ترین کتابخانه‌های پایتون برای یادگیری ماشین است. این کتابخانه بر پایه کتابخانه‌های NumPy، SciPy و Matplotlib ساخته شده و ابزارهای قدرتمندی را برای انجام طیف وسیعی از الگوریتم‌های یادگیری ماشین در اختیار شما قرار می‌دهد.

چرا Scikit-learn؟

- سادگی استفاده: Scikit-learn دارای یک رابط کاربری ساده و یکپارچه است که به شما امکان می‌دهد به سرعت مدل‌های یادگیری ماشین خود را ایجاد و ارزیابی کنید.
- انعطاف‌پذیری بالا: این کتابخانه از طیف وسیعی از الگوریتم‌های یادگیری ماشین، از جمله طبقه‌بندی، رگرسیون، خوشه‌بندی و کاهش ابعاد پشتیبانی می‌کند.
- جامعه کاربری بزرگ: Scikit-learn دارای یک جامعه کاربری فعال و بزرگ است که به شما امکان می‌دهد به راحتی به منابع و آموزش‌های زیادی دسترسی پیدا کنید.
- توسعه‌پذیری بالا: Scikit-learn به خوبی با سایر کتابخانه‌های پایتون مانند Pandas و Matplotlib کار می‌کند.

کاربردهای Scikit-learn

- طبقه‌بندی: پیش‌بینی برچسب کلاس برای داده‌های جدید (مثلاً تشخیص اسپم، تشخیص بیماری‌ها)
- رگرسیون: پیش‌بینی مقدار عددی یک متغیر هدف (مثلاً پیش‌بینی قیمت خانه، پیش‌بینی میزان فروش)
- خوشه‌بندی: گروه‌بندی داده‌ها بر اساس شباهت (مثلاً بخش‌بندی مشتریان، تشخیص الگوها)
- کاهش ابعاد: کاهش تعداد ویژگی‌های داده‌ها بدون از دست دادن اطلاعات مهم (مثلاً تجسم داده‌ها، انتخاب ویژگی)

مفاهیم کلیدی در Scikit-learn

- مدل: یک نمایش ریاضی از رابطه بین ویژگی‌ها و برچسب‌های هدف.
- آموزش: فرایند تنظیم پارامترهای مدل بر اساس داده‌های آموزشی.
- پیش‌بینی: استفاده از مدل آموزش دیده برای پیش‌بینی برچسب‌های داده‌های جدید.

- ارزیابی: ارزیابی عملکرد مدل بر روی داده‌های تست.

انواع الگوریتم‌های موجود در Scikit-learn

- الگوریتم‌های طبقه‌بندی: SVM, KNN, درخت تصمیم، جنگل تصادفی، شبکه‌های عصبی و ...
- الگوریتم‌های رگرسیون: رگرسیون خطی، رگرسیون لجستیک، رگرسیون درخت تصمیم، رگرسیون ساپورت وکتور و ...
- الگوریتم‌های خوشه‌بندی: K-means, DBSCAN, Hierarchical Clustering و ...
- الگوریتم‌های کاهش ابعاد: PCA, ...



شکل ۱۰-۲ کتابخانه sklearn

Keras: ابزاری قدرتمند برای یادگیری عمیق در پایتون

Keras یک کتابخانه یادگیری عمیق سطح بالا و متن‌باز است که برای ساخت و آموزش مدل‌های شبکه‌های عصبی مصنوعی به کار می‌رود. Keras به دلیل سادگی استفاده، انعطاف‌پذیری و سرعت بالا، یکی از محبوب‌ترین کتابخانه‌های یادگیری عمیق در میان محققان و مهندسان داده است.

چرا Keras؟

- سادگی استفاده: Keras با یک API ساده و شهودی طراحی شده است که به شما اجازه می‌دهد به سرعت و به راحتی مدل‌های شبکه عصبی خود را ایجاد و آموزش دهید.
- انعطاف‌پذیری بالا: Keras از انواع مختلف شبکه‌های عصبی، از جمله شبکه‌های عصبی کانولوشنی (CNN)، شبکه‌های عصبی بازگشتی (RNN) و شبکه‌های عصبی مولد (GAN) پشتیبانی می‌کند.
- سرعت بالا: Keras از کتابخانه‌های قدرتمند مانند TensorFlow یا Theano به عنوان بک‌اند استفاده می‌کند که به آن امکان می‌دهد محاسبات پیچیده شبکه‌های عصبی را با سرعت بالا انجام دهد.

- قابلیت توسعه‌پذیری: Keras به شما اجازه می‌دهد تا مدل‌های خود را به صورت سفارشی سازی کنید و لایه‌های جدید و توابع فعال‌سازی را به آن اضافه کنید.

کاربردهای Keras

- بینایی ماشین: تشخیص اشیاء، تشخیص چهره، تولید تصاویر
- پردازش زبان طبیعی: ترجمه ماشینی، تولید متن، تحلیل احساسات
- تولید صدا: تولید موسیقی، تبدیل متن به گفتار
- تقویت یادگیری: آموزش عامل‌ها برای انجام وظایف در محیط‌های شبیه‌سازی شده

مفاهیم کلیدی در Keras

- مدل: یک نمایش ریاضی از یک شبکه عصبی است.
- لایه: یک واحد ساختاری در یک شبکه عصبی است که عملیات خاصی را روی داده‌ها انجام می‌دهد.
- کامپایل کردن: فرایند آماده‌سازی یک مدل برای آموزش است که در آن تابع هزینه و بهینه‌ساز مشخص می‌شود.
- آموزش: فرایند تنظیم وزن‌های یک مدل به گونه‌ای که بتواند داده‌های ورودی را به خوبی پیش‌بینی کند.
- ارزیابی: فرایند ارزیابی عملکرد یک مدل بر روی داده‌های تست.



شکل ۲-۹ - کتابخانه keras

۲-۳- آشنایی با انواع محیط های پایتون

پایتون ساده و ژوپیتِر نوت‌بوک هر دو ابزار قدرتمندی برای برنامه‌نویسی به زبان پایتون هستند، اما برای اهداف و روش‌های مختلفی به کار می‌روند. در این توضیح، تفاوت‌ها و شباهت‌های این دو محیط را بررسی می‌کنیم تا بتوانید بهترین گزینه را برای پروژه خود انتخاب کنید.

پایتون ساده (Python Shell)

تعریف: یک محیط تعاملی است که در آن می‌توانید دستورات پایتون را به صورت خط به خط وارد کرده و خروجی آن‌ها را به صورت فوری مشاهده کنید.

کاربردها:

- آزمایش کدهای کوتاه
- یادگیری مفاهیم اولیه پایتون
- انجام محاسبات سریع

مزایا:

- سادگی استفاده
- مناسب برای یادگیری سریع

معایب:

- برای برنامه‌های بزرگ و پیچیده مناسب نیست
- قابلیت سازماندهی و ویرایش کد محدود است

Jupyter note book

تعریف: یک محیط وب‌بیس تعاملی است که برای ایجاد و اشتراک‌گذاری اسناد قابل اجرا حاوی کد، معادلات، ویژوالایزیشن‌ها و متن فرمت‌شده استفاده می‌شود.

کاربردها:

- تحلیل داده‌ها
- یادگیری ماشین
- آموزش برنامه‌نویسی

- مستندسازی پروژه‌ها

مزایا:

- قابلیت ایجاد اسناد تعاملی و جذاب
- پشتیبانی از انواع مختلف زبان‌های برنامه‌نویسی (نه فقط پایتون)
- قابلیت اشتراک‌گذاری و همکاری

معایب:

- پیچیدگی بیشتر نسبت به پایتون ساده
- نیاز به نصب و پیکربندی

چه زمانی از کدام یک استفاده کنیم؟

پایتون ساده:

- زمانی که می‌خواهید یک محاسبه سریع انجام دهید یا یک مفهوم پایتونی را آزمایش کنید.
- برای برنامه‌نویسان مبتدی که به دنبال یک محیط ساده برای شروع هستند.

Jupyter notebook

- برای پروژه‌های تحلیل داده، یادگیری ماشین و سایر پروژه‌های علمی.
- زمانی که می‌خواهید نتایج خود را به صورت تعاملی و جذاب ارائه دهید.
- برای ایجاد آموزش‌های برنامه‌نویسی و مستندسازی پروژه‌ها.

در نهایت، انتخاب بین پایتون ساده و ژوپیتِر نوت‌بوک به نیازها و ترجیحات شما بستگی دارد. اگر به دنبال یک محیط ساده و سریع برای آزمایش کد هستید، پایتون ساده گزینه مناسبی است. اما اگر به دنبال یک ابزار قدرتمند برای تحلیل داده و ایجاد اسناد تعاملی هستید، jupyter notebook انتخاب بهتری خواهد بود.

فصل سوم

تحليل و طراحی نرم افزار

مقدمه

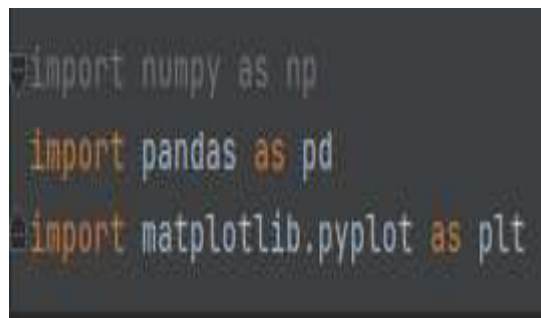
در اینجا شرح کدهای اصلی برنامه می پردازیم که سعی کردیم تا حد امکان مختصر و جامع باشد.

۳-۱ افزودن کتابخانه های لازم (import)

اعمالی که در این قسمت انجام می شود عبارت است از:

- بررسی موجود بودن کتابخانه
- اضافه کردن کتابخانه های لازم

نکته ای که در اینجا ذکر باید شود دو کتابخانه keras,sklearn به دلیل سنگین بودن در ادامه تابع های آن اضافه می گردد نه کل کتابخانه



```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

شکل ۳-۱- import library

۳-۲- دانلود دیتاست (dataset)

به طور کلی خوانش دیتا برای هوش مصنوعی چند راه حل دارد که در پایین نام میبریم:

۱. دانلود دیتاست: برای اینکار میتوانیم به وبسایت هایی از قبیل kaggle, hugging face,.....

مراجعه کنیم و در صورت موجود بودن آن excel یا csv آن را دانلود کنیم.

۲. API: بعضی از سازمان ها و نهاد ها دیتا خود را به شکل زنده قابل دسترسی قرار میدهند

و میتوانیم با اضافه کردن کتابخانه آن ها از این دیتا استفاده کنیم. از قبیل yfinance

۳. پایگاه داده: ما میتوانیم از دیتا درون سایت که به پایگاه داده آن دسترسی داریم با

دستور select دیتا که می خواهیم را دانلود کنیم.

۴. Web scrapping: یکی از راهکار های دیگر که میتوانیم دیتاست خود را بسازیم این هست

که وارد سایت مدنظر خود بشویم و اطلاعاتی را که نیاز داریم را با استفاده از پکیج های

BeautifulSoup, Selenium دیتا خود را به شکل جدولی بسازیم.

دانلود دیتاست از وبسایت kaggle

چرا Kaggle؟

- داده‌های متنوع: Kaggle میزبان طیف گسترده‌ای از داده‌ها در حوزه‌های مختلف مانند پزشکی، اقتصاد، تصویر، متن و ... است.
- مسابقات جذاب: شرکت در مسابقات Kaggle فرصتی عالی برای یادگیری، بهبود مهارت‌ها و رقابت با دیگر دانشمندان داده است.
- جامعه فعال: Kaggle یک جامعه بزرگ از دانشمندان داده، مهندسين یادگیری ماشین و علاقه‌مندان به این حوزه است که می‌توانید با آن‌ها به تبادل نظر و همکاری بپردازید.
- ابزارهای قدرتمند: Kaggle ابزارها و محیط‌های کاری مختلفی را برای آماده‌سازی، تحلیل و مدل‌سازی داده‌ها در اختیار کاربران قرار می‌دهد.

چگونه از Kaggle برای دانلود دیتاست استفاده کنیم؟

ثبت‌نام: برای استفاده از Kaggle باید یک حساب کاربری ایجاد کنید. این فرآیند بسیار ساده و رایگان است.

در ابتدا پس از کمی جست و جو دیتاست را پیدا کردیم که دارای ۲۵۰۰ سطر و ۸ ستون می‌باشد ستون‌های دیتا به نام‌های زیر است:

- ستون review (نظرات مردم)
- ستون date (تاریخ پرواز)
- ستون country (کشور هواپیما)
- ستون seat type (نوع صندلی)
- ستون recommended (توصیه شده یا نه و به طور کلی نتیجه ما از سفر)
- ستون stars (ستاره هر هواپیما)
- ستون route (مبدأ و مقصد سفر)
- ستون type of traveller (نوع سفر)

خواندن csv فایل

```
df=pd.read_csv('British_Airways_Review.csv')
```

df

	date	country	seat_type	recommended	stars	route	type_of_traveller
I had the most fant...	1st August 2023	Hong Kong	Business Class	yes	5	Heathrow to Las Vegas	Family Leisure
Couldn't book in on...	31st July 2023	United Kingdom	Economy Class	no	3	Rome to Heathrow	Solo Leisure
ondon Heathrow to H...	31st July 2023	Iceland	Business Class	yes	3	Gatwick to Venice	Solo Leisure
eflavik, Iceland to...	31st July 2023	Iceland	Business Class	yes	5	London to Luanda	Couple Leisure
errible Experience...	29th July 2023	Canada	Economy Class	no	5	Denver to Heathrow	Family Leisure
n airline that live...	30th July 2023	Qatar	Business Class	no	3	BKK to LHR	Business
Chania-Sa Racha route	30th July 2023	Malta	Economy Class	no	5	London to Tampa	Couple Leisure

1500 rows x 8 columns Open in new tab

تمیزکاری داده: کلید موفقیت در تحلیل داده

تمیزکاری داده یا پاکسازی داده فرایندی است که در آن داده‌های خام و ناقص را به داده‌های تمیز، دقیق و قابل استفاده برای تحلیل تبدیل می‌کنیم. این مرحله یکی از مهم‌ترین مراحل در هر پروژه تحلیل داده است، زیرا کیفیت نتایج تحلیل به طور مستقیم به کیفیت داده‌های ورودی بستگی دارد.

چرا تمیزکاری داده مهم است؟

- دقت در نتایج: داده‌های نادرست یا ناقص می‌توانند به نتایج تحلیل اشتباه منجر شوند و تصمیم‌گیری‌های نادرستی را در پی داشته باشند.
- بهبود کارایی مدل‌ها: مدل‌های یادگیری ماشین برای عملکرد بهتر به داده‌های تمیز و با کیفیت نیاز دارند.
- صرفه‌جویی در زمان و هزینه: تمیزکاری داده در مراحل اولیه پروژه می‌تواند از صرف هزینه و زمان اضافی در مراحل بعدی جلوگیری کند.

مراحل معمول تمیزکاری داده:

- شناسایی داده‌های نادرست:
- داده‌های تکراری
- داده‌های گم‌شده
- داده‌های نامعتبر (مانند تاریخ‌های اشتباه، مقادیر عددی غیرممکن)
- داده‌های ناسازگار (مانند نوع داده‌ای متفاوت در یک ستون)
- اصلاح داده‌های نادرست:
- حذف داده‌های تکراری

- پر کردن داده‌های گم‌شده (با میانگین، مد، یا روش‌های پیشرفته‌تر)
- تصحیح داده‌های نامعتبر
- تبدیل داده‌ها به فرمت یکسان

استانداردسازی داده‌ها:

- تبدیل داده‌ها به مقیاس یکسان (مانند نرمال‌سازی)
- کدگذاری داده‌های کیفی (مانند تبدیل جنسیت به مقادیر عددی)

ابزارهای مفید برای تمیزکاری داده:

- پایتون: با کتابخانه‌هایی مانند NumPy، Pandas و Scikit-learn
- SQL: برای تمیزکاری داده‌های موجود در پایگاه داده‌ها

ابزارهای بصری‌سازی داده: برای شناسایی آسان‌تر ناهنجاری‌ها

مثال‌هایی از تمیزکاری داده:

- حذف سطرهایی که شامل مقدار گم‌شده در ستون سن هستند.
- تبدیل ستون تاریخ تولد به فرمت استاندارد.
- جایگزینی مقادیر گم‌شده در ستون درآمد با میانگین درآمد افراد هم‌سن.
- حذف داده‌های پرت (Outliers) که به شدت از سایر داده‌ها فاصله دارند.
- توجه: تمیزکاری داده یک فرآیند تکراری است و ممکن است نیاز به چندین بار تکرار داشته باشد تا داده‌ها به کیفیت مطلوب برسند.
- در نهایت، تمیزکاری داده یک مهارت ضروری برای هر دانشمند داده است و به شما کمک می‌کند تا از داده‌های خود بیشترین بهره را ببرید.

ما ابتدا به بررسی تمیز بودن داده می‌پردازیم به طور عام اکثر داده‌هایی که از kaggle دانلود می‌وشند تمیز هستند.

```
df.isnull().sum()
```

	data
reviews	0
date	0
country	0
seat_type	0
recommended	0
stars	0
route	0

شکل ۳-۲- بررسی تمیز بودن داده

پس از بررسی خالی بودن داده به این نتیجه رسیدیم که داده ها تمیز هستند و نیاز به تمیزکاری بیشتر ندارند.

مصورسازی داده: تبدیل اعداد به داستان‌های بصری

مصورسازی داده یا تجسم داده (Data Visualization) فرایندی است که در آن داده‌های خام و پیچیده به شکل‌های گرافیکی ساده و قابل فهم تبدیل می‌شوند. این کار به ما کمک می‌کند تا الگوها، روندها و ارتباطات پنهان در داده‌ها را به سرعت و به صورت بصری شناسایی کنیم. در واقع، مصورسازی داده، یک پل ارتباطی بین داده‌ها و ذهن انسان است.

چرا مصورسازی داده مهم است؟

- درک بهتر: مغز انسان تصاویر را بسیار سریع‌تر از اعداد و ارقام پردازش می‌کند. به همین دلیل، مصورسازی داده به ما کمک می‌کند تا اطلاعات پیچیده را به سرعت درک کنیم.
- کشف الگوها: با مشاهده نمودارها و گراف‌ها، می‌توانیم به راحتی الگوها، روندها و انحرافات از حالت نرمال را شناسایی کنیم.
- ارتباط مؤثر: مصورسازی داده به ما این امکان را می‌دهد تا یافته‌های خود را به صورت واضح و مختصر با دیگران به اشتراک بگذاریم.
- داستان‌سرایی با داده‌ها: با استفاده از مصورسازی، می‌توانیم داستان‌هایی جذاب از داده‌ها خلق کنیم و مخاطبان را درگیر کنیم.

انواع نمودارها و گرافها

برای مصورسازی داده‌ها از انواع مختلف نمودارها و گرافها استفاده می‌شود که هر کدام برای نمایش نوع خاصی از داده‌ها مناسب هستند. برخی از مهم‌ترین انواع آن‌ها عبارتند از:

- نمودار خطی: برای نمایش روند تغییرات یک متغیر در طول زمان
- نمودار میله‌ای: برای مقایسه مقادیر مختلف یک متغیر
- نمودار دایره‌ای: برای نمایش نسبت بخش‌های مختلف یک کل
- نمودار پراکندگی: برای نشان دادن رابطه بین دو متغیر کمی
- نقشه‌های حرارتی: برای نمایش شدت یک متغیر در یک ناحیه خاص
- شبکه‌ها: برای نمایش روابط بین موجودیت‌ها

ابزارهای مصورسازی داده

برای ایجاد نمودارها و گراف‌های زیبا و حرفه‌ای، ابزارهای مختلفی وجود دارد. برخی از محبوب‌ترین آن‌ها عبارتند از:

- پایتون: با کتابخانه‌هایی مانند Matplotlib, Seaborn و Plotly
- Tableau: یک ابزار قدرتمند برای ایجاد داشبوردهای تعاملی
- Power BI: یک ابزار مایکروسافت برای تجزیه و تحلیل داده‌ها و مصورسازی
- Google Data Studio: یک ابزار رایگان برای ایجاد داشبوردهای ساده

کاربردهای مصورسازی داده

مصورسازی داده در حوزه‌های مختلفی مانند تجارت، علم، پزشکی، و ... کاربرد دارد. برخی از مهم‌ترین کاربردهای آن عبارتند از:

- تحلیل کسب‌وکار: شناسایی روندهای بازار، رفتار مشتریان و عملکرد کسب‌وکار
- تحلیل داده‌های علمی: کشف الگوها و روابط بین متغیرهای مختلف
- ایجاد داشبوردهای مدیریتی: نمایش اطلاعات کلیدی به مدیران برای تصمیم‌گیری بهتر
- آموزش و پرورش: انتقال مفاهیم پیچیده به صورت بصری و جذاب

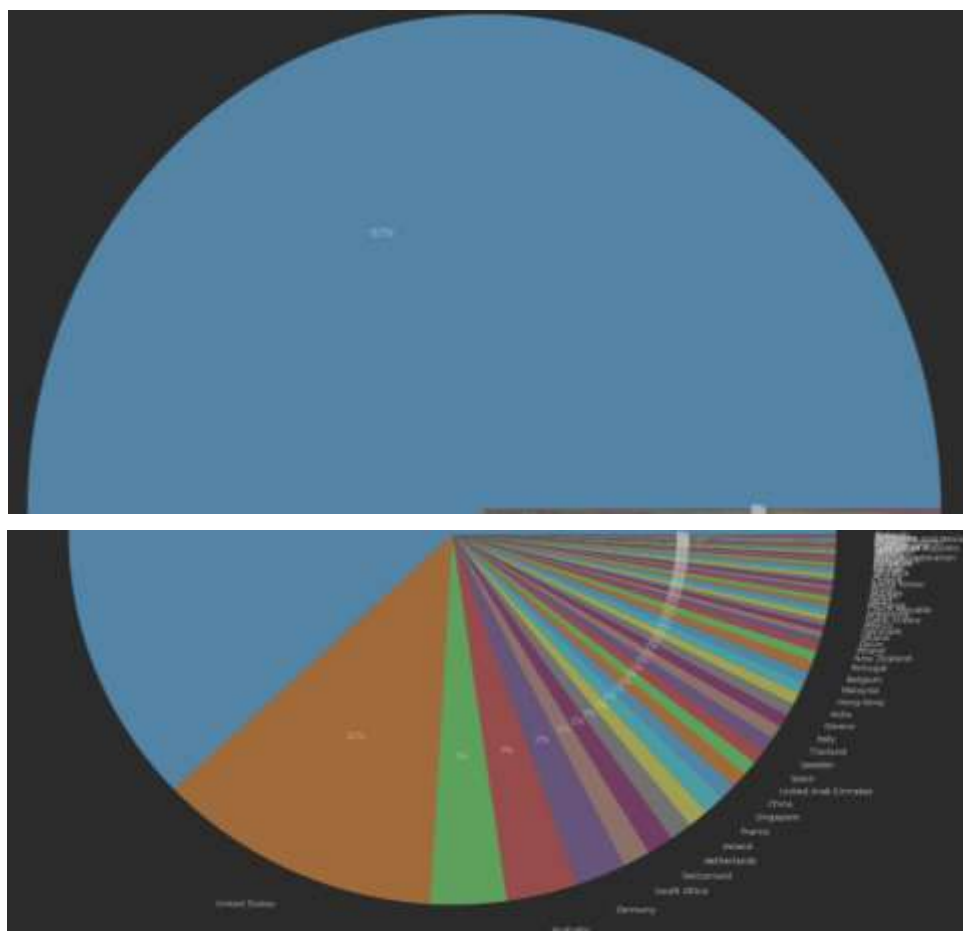
در نهایت، مصورسازی داده یک مهارت ضروری برای هر کسی است که با داده‌ها سروکار دارد. با استفاده از این مهارت، می‌توانید اطلاعات پیچیده را به صورت ساده و قابل فهم به دیگران منتقل کنید و تصمیم‌گیری‌های بهتری بگیرید.

پس از توضیحات بالا و تمیز کاری داده نیاز است تا هر ستون را مصور سازی کنیم تا درک بهتری از داده‌ها داشته باشیم. مصور سازی ها با نمودار دایره ای صورت گرفته است که برای هر ستون به شرح زیر است:

```
plt.figure(figsize=[30,20])
df['country'].value_counts().plot(kind='pie',autopct='%0.0f%%')
```

شکل ۳-۳- مصور سازی ستون کشور

ما ابتدا به دلیل زیاد بودن تعداد کشور های از `figsize[30,20]` استفاده کردیم سپس با فراخوانی تابع شمارش اعضا و مصور کردن از نوع دایره ای و با ۰ رقم اعشار نمایش دادیم که به شرح زیر می باشد.



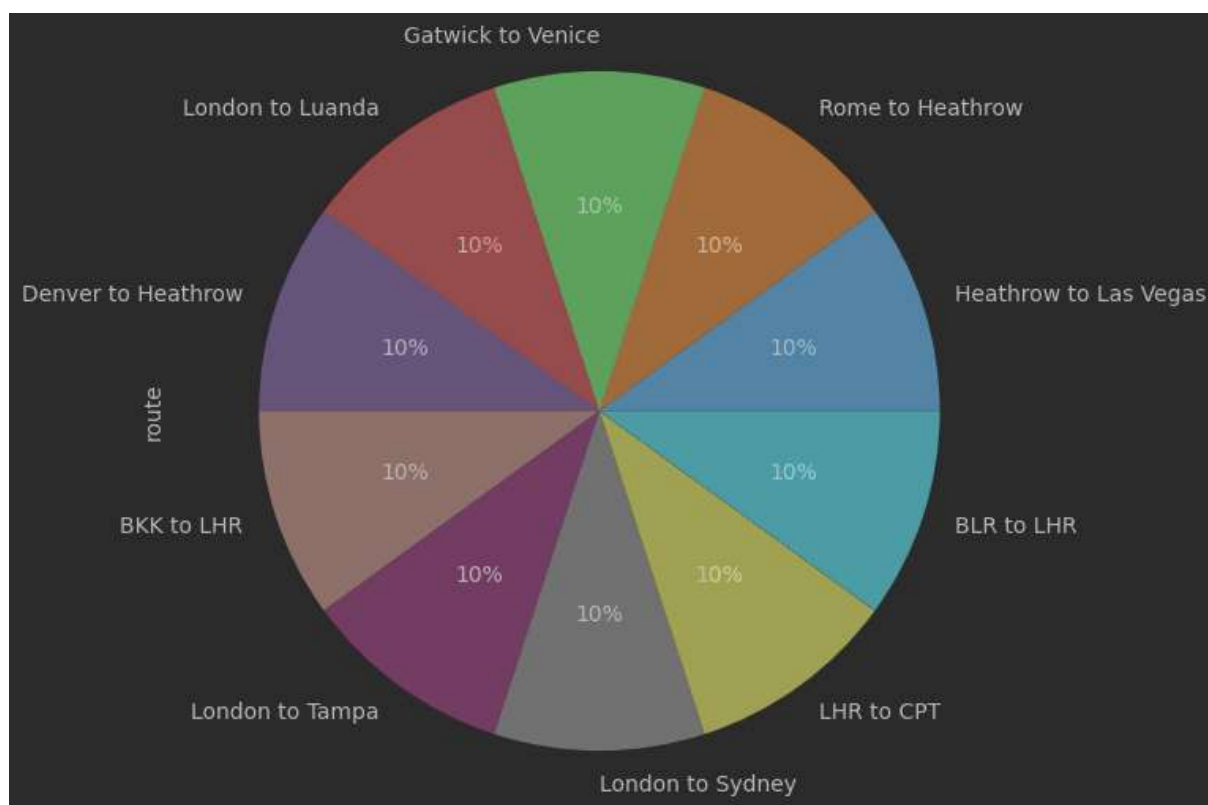
شکل ۴-۳- مصور سازی نمودار کشور

سپس نمودار بعدی را مصور سازی میکنیم:

در این مرحله به دلیل متوسط بودن اعضای جامعه ما اندازه نمودار را کاهش دادیم

```
plt.figure(figsize=[7,7])
df.route.value_counts().plot(kind='pie',autopct='%0f%%')
```

شکل ۳-۵- مصور سازی نمودار مسیر

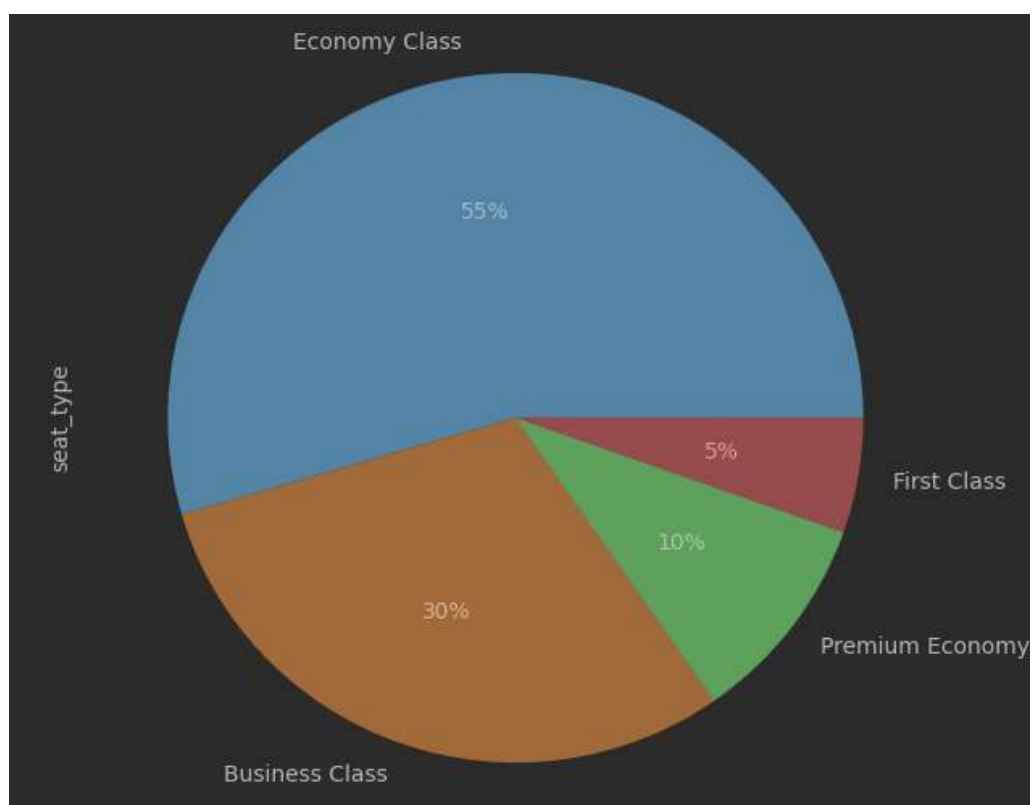


شکل ۳-۶- مصور سازی نمودار مسیر

باقی ستون ها را نیز به همین ترتیب مصور سازی میکنیم:

```
plt.figure(figsize=[7,7])
df.seat_type.value_counts().plot(kind='pie',autopct='%0f%%')
```

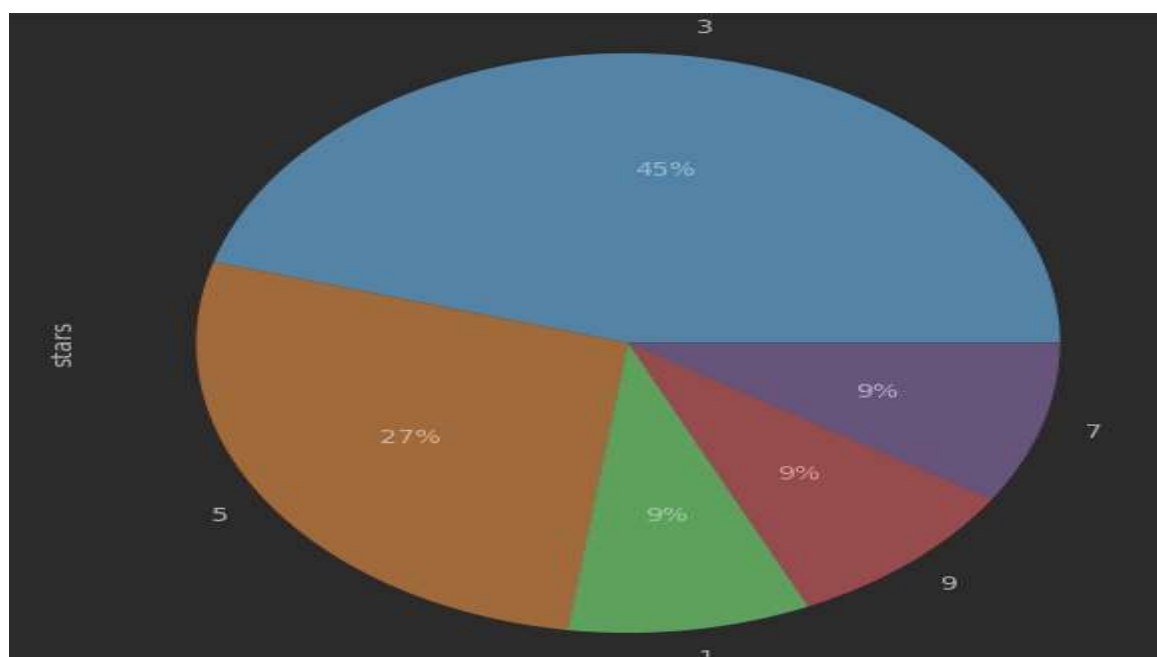
شکل ۳-۷- مصور سازی نمودار نوع صندلی



شکل ۳-۸- مصور سازی نمودار نوع صندلی

```
plt.figure(figsize=[7,7])
df.stars.value_counts().plot(kind='pie', autopct='%0f%%')
```

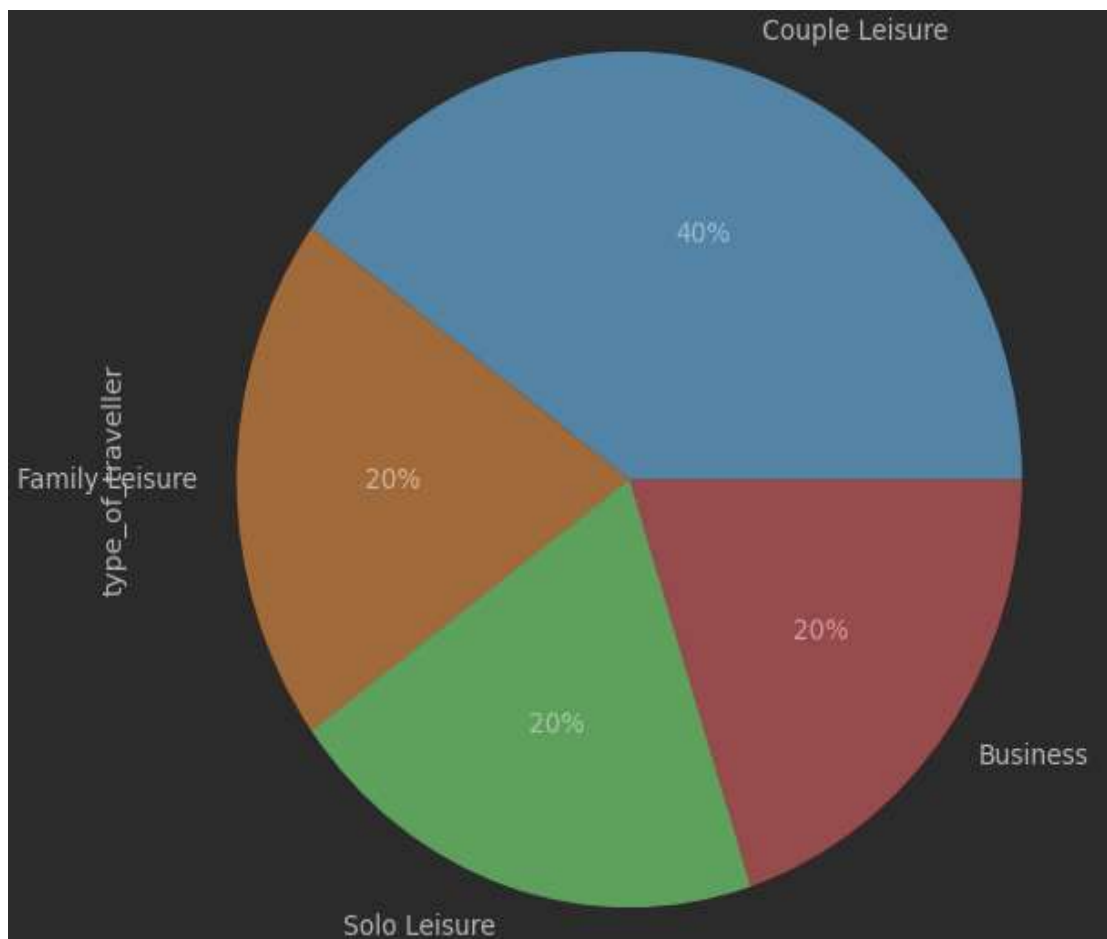
شکل ۳-۹- مصور سازی نمودار ستاره های هواپیما



شکل ۳-۱۰- مصور سازی نمودار ستاره های هواپیما

```
plt.figure(figsize=[7,7])
df.type_of_traveller.value_counts().plot(kind='pie',autopct='%0f%%')
```

شکل ۳-۱۱- مصور سازی نمودار نوع سفر



شکل ۳-۱۲- مصور سازی نمودار نوع سفر

جدا سازی زمان ها:

در این مرحله به دلیل اینکه برای پیش بینی باید دیتا عددی یا شی درون دیتا فریم باشد تا بتوانیم پیش بینی عددی اعداد را انجام دهیم باید تمیزکاری روی زمان انجام دهیم در مرحله اول ستون زمان را از دیتا فریم به سه ستون سال ماه روز تبدیل میکنیم و سپس کتابخانه خطا را استفاده میکنیم تا از اخطار ها صرف نظر شود. کد برنامه به شرح زیر میباشد.

```

import warnings

warnings.simplefilter('ignore')

df['day']=df['date'].str.split(' ',1).str[0]
df['month']=df['date'].str.split(' ',1).str[1].str[:2]
df['year']=df['date'].str.split(' ',1).str[1].str[-2:]
df

```

te	country	seat_type	recommended	stars	route	type_of_traveller	day	month	year
1st August 2023	Hong Kong	Business Class	yes	5	Heathrow to Las Vegas	Family Leisure	1st	Aug	23
31st July 2023	United Kingdom	Economy Class	no	3	Rome to Heathrow	Solo Leisure	31st	Jul	23
31st July 2023	Iceland	Business Class	yes	3	Gatwick to Venice	Solo Leisure	31st	Jul	23
31st July 2023	Iceland	Business Class	yes	5	London to Luanda	Couple Leisure	31st	Jul	23
29th July 2023	Canada	Economy Class	no	5	Denver to Heathrow	Family Leisure	29th	Jul	23
26th July 2023	Qatar	Business Class	no	3	DHX to LHR	Business	26th	Jul	23
26th July 2023	United Kingdom	Economy Class	no	3	London to Tampa	Family Leisure	26th	Jul	23

شکل ۳-۱۳- جدا سازی زمان

در ادامه این مرحله پسوند های روز را از دیتا فریم جدا میکنیم که کد مراحل آن به شرح زیر میباشد.

```

df['day']=df['day'].str.extract('(^\d+)')
df

```

te	country	seat_type	recommended	stars	route	type_of_traveller	day	month	year
1st August 2023	Hong Kong	Business Class	yes	5	Heathrow to Las Vegas	Family Leisure	1	Aug	23
31st July 2023	United Kingdom	Economy Class	no	3	Rome to Heathrow	Solo Leisure	31	Jul	23
31st July 2023	Iceland	Business Class	yes	3	Gatwick to Venice	Solo Leisure	31	Jul	23
31st July 2023	Iceland	Business Class	yes	5	London to Luanda	Couple Leisure	31	Jul	23
29th July 2023	Canada	Economy Class	no	5	Denver to Heathrow	Family Leisure	29	Jul	23
26th July 2023	Qatar	Business Class	no	3	DHX to LHR	Business	26	Jul	23
26th July 2023	United Kingdom	Economy Class	no	3	London to Tampa	Family Leisure	26	Jul	23

شکل ۳-۱۴- جدا سازی پسوند

در مرحله بعد باید ماه ها را به عدد اصلی آن در زمان تبدیل کنیم برای این کار باید از کتابخانه pandas تابع replace فراخوانی کرده و کار مورد نظر را انجام دهیم. سپس از آرگومان استفاده میکنیم تا تغییرات روی کل دیتا فریم ذخیره گردد.

```

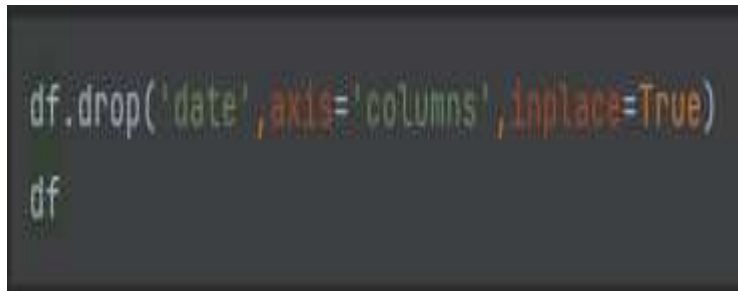
df['month'].replace({'Jan':1,'Feb':2,'Mar':3,'Apr':4,'May':5,'Jun':6,'Jul':7,'Aug':8,'Sep':9,'Oct':10,'Nov':11,'Dec':12}, inplace=True)
df

```

	reviews	date	country	seat_type	recommended	stars	route
0	✓ Trip Verified I had the best fant...	1st August 2023	Hong Kong	Business Class	yes	5	Heathrow to La...
1	✓ Trip Verified Couldn't book in on...	31st July 2023	United Kingdom	Economy Class	no	3	Rome to H...
2	✓ Trip Verified London Heathrow to H...	31st July 2023	Iceland	Business Class	yes	3	Gatwick to...
3	✓ Trip Verified Keflavik, Iceland to...	31st July 2023	Iceland	Business Class	yes	5	London to...
4	✓ Trip Verified Terrible Experience ...	29th July 2023	Canada	Economy Class	no	5	Denver to H...
5	✓ Trip Verified An airline that live...	26th July 2023	Qatar	Business Class	no	3	DHX
6	✓ Trip Verified Check in back into...	26th July 2023	United Kingdom	Economy Class	no	3	London to T...

شکل ۳-۱۵- تبدیل ماه

در مرحله بعد زمان را از دیتا فریم حذف میکنیم چون نیازی به آن نداریم.



```
df.drop('date', axis='columns', inplace=True)
df
```

شکل ۳-۱۶- حذف زمان

برچسب‌گذار Label Encoder در sklearn: تبدیل داده‌های کیفی به کمی

برچسب‌گذار Label Encoder یک ابزار مفید در کتابخانه Scikit-learn است که برای تبدیل داده‌های کیفی (Categorical) به داده‌های کمی (Numerical) استفاده می‌شود. بسیاری از الگوریتم‌های یادگیری ماشین تنها با داده‌های عددی کار می‌کنند، بنابراین تبدیل داده‌های کیفی به عددی یک مرحله ضروری در پیش‌پردازش داده‌ها است.

چرا از برچسب‌گذار استفاده می‌کنیم؟

- سازگاری با الگوریتم‌ها: اکثر الگوریتم‌های یادگیری ماشین، به ویژه الگوریتم‌های مبتنی بر فاصله (مانند KNN) یا الگوریتم‌هایی که از عملیات ریاضی استفاده می‌کنند، به داده‌های عددی نیاز دارند.
- ساده‌سازی محاسبات: تبدیل داده‌های کیفی به عددی، محاسبات را ساده‌تر کرده و سرعت آموزش مدل را افزایش می‌دهد.

محدودیت‌های برچسب‌گذار

- ترتیب اهمیت: برچسب‌گذار به طور خودکار به هر کلاس یک عدد اختصاص می‌دهد. این ممکن است منجر به ایجاد یک ترتیب کاذب بین کلاس‌ها شود که ممکن است بر عملکرد مدل تأثیر بگذارد.
- اطلاعات اضافی: برچسب‌گذاری ساده، اطلاعات اضافی موجود در داده‌های کیفی را از بین می‌برد. برای مثال، اگر سه کلاس "کم"، "متوسط" و "زیاد" داشته باشیم، برچسب‌گذار ممکن است به آن‌ها اعداد ۰، ۱ و ۲ را اختصاص دهد که ترتیب آن‌ها لزوماً نشان‌دهنده ترتیب اهمیت نیست.

چه زمانی از برچسب‌گذار استفاده نکنیم؟

- داده‌های ترتیبی: اگر داده‌های کیفی شما ترتیبی هستند (مثلاً "کم"، "متوسط"، "زیاد")، بهتر است از روش‌های دیگری مانند One-Hot Encoding استفاده کنید تا ترتیب بین کلاس‌ها حفظ شود.
- داده‌های نامتناهی: اگر تعداد کلاس‌های شما بسیار زیاد یا نامتناهی است، برچسب‌گذاری ممکن است مناسب نباشد.

چه زمانی از برچسب‌گذار استفاده کنیم؟

- داده‌های اسمی: برای داده‌های اسمی (مانند رنگ، کشور) که هیچ ترتیبی بین آن‌ها وجود ندارد، برچسب‌گذاری یک روش مناسب است.
- سادگی: اگر هدف شما صرفاً تبدیل داده‌های کیفی به عددی برای استفاده در یک الگوریتم یادگیری ماشین است، برچسب‌گذاری یک روش ساده و سریع است.

جمع‌بندی برچسب‌گذار یک ابزار مفید در Scikit-learn است که برای تبدیل داده‌های کیفی به عددی استفاده می‌شود. با درک محدودیت‌ها و کاربردهای آن، می‌توانید تصمیم بگیرید که آیا برچسب‌گذاری برای داده‌های شما مناسب است یا خیر. برای عددی کردن دیتافریم از برچسب‌گذار استفاده کردیم و چهار ستون کشور نوع صندلی مسیر نوع سفر را برای پیش‌بینی قابل درک نمودیم در ادامه کد این کار را مشاهده می‌کنید.

```
from sklearn.preprocessing import LabelEncoder
label=LabelEncoder()
df['country']=label.fit_transform(df['country'])
df['seat_type']=label.fit_transform(df['seat_type'])
df['route']=label.fit_transform(df['route'])
df['type_of_traveller']=label.fit_transform(df['type_of_traveller'])
df
```

rows	country	seat_type	recommended	stars	route	type_of_traveller	day	month	year
rip Verified I had the most fant...	26	0	yes	5	4	2	1	8	23
rip Verified Couldn't book in an...	64	1	no	3	9	3	31	8	23
rip Verified London Heathrow to M...	26	0	yes	3	3	3	31	8	23
rip Verified Nerfaria, Iceland to...	26	0	yes	5	6	1	31	8	23
rip Verified Terrible Experience ...	9	1	no	5	2	2	29	8	23
rip Verified An airline that live...	47	0	no	3	0	0	26	8	23
rip Verified Phone in back mode	46	1	no	5	8	1	28	8	23

شکل ۳-۱۷- برچسب‌گذاری

در ادامه قرار است پیش‌بینی ما روی ستون توصیه شده باشد پس با جای‌گذاری بله را برابر ۱ و نه را برابر ۰ قرار دادیم.

```
df['recommended'].replace('yes', 'No', inplace=True)
```

df:

ews	country	seat_type	recommended	stars	route	type_of_traveller	day	month	year
rip Verified I had the most feet...	24	0	3	5	4	3	1	8	23
rip Verified Couldn't book in dn...	64	1	0	3	9	3	31	0	23
rip Verified London Heathrow to M...	20	0	3	3	3	3	31	0	23
rip Verified Keflavik, Iceland to...	26	0	3	5	0	1	31	0	23
rip Verified Terrible Experience ...	9	1	0	5	2	2	29	0	23
rip Verified An airline that live...	47	0	0	3	8	0	26	0	23
rip Verified Pharo in Rock route	66	1	0	1	0	1	26	0	23

2500 rows x 10 columns Open in new tab

شکل ۳-۱۸- جای گذاری به جای مقادیر رشته

مرحله بعد برای مشخص کردن نوع هر ستون میباشد.

```
df.dtypes
```

reviews	object
country	int32
seat_type	int32
recommended	int64
stars	int64
route	int32
type_of_traveller	int32
day	object
month	int64
year	object

شکل ۳-۱۹- dtype

در این مرحله ستون های که شی هستند را به int 32 تبدیل میکنیم و نتیجه به صورت زیر میباشد.

```
df['year']=df['year'].astype('int32')
df['day']=df['day'].astype('int32')
df['month']=df['month'].astype('int32')

df['recommended']=df['recommended'].astype('int32')
df['stars']=df['stars'].astype('int32')
```

```
df.dtypes
```

	data
country	int64
seat_type	int64
recommended	int32
stars	int32
route	int64
type_of_traveller	int64
day	int32

شکل ۳-۲۰- تغییر دادن dtype

در مرحله بعد ستون review را که از همان اول جزو اهداف پروژه نبود را حذف میکنیم.

```
df.drop('reviews',axis='columns',inplace=True)
df
```

شکل ۳-۲۱- حذف سطر نا کارآمد

همبستگی رابطه بین متغیرها

همبستگی (Correlation) در آمار و یادگیری ماشین، به اندازه گیری میزان ارتباط بین دو یا چند متغیر گفته می شود. در واقع، همبستگی نشان می دهد که تا چه اندازه تغییرات در یک متغیر با تغییرات در متغیر دیگر همراه است.

چرا همبستگی مهم است؟

- شناسایی متغیرهای مهم: با شناسایی متغیرهایی که بیشترین همبستگی را با متغیر هدف دارند، می توانیم متغیرهای مهم تر را برای مدل سازی انتخاب کنیم.
- کاهش ابعاد: متغیرهایی که همبستگی بالایی با هم دارند، ممکن است اطلاعات تکراری ارائه دهند. با حذف یکی از این متغیرها، می توانیم ابعاد داده ها را کاهش دهیم و پیچیدگی مدل را کم کنیم.
- تشخیص چندخطی: در رگرسیون، چندخطی (Multicollinearity) زمانی رخ می دهد که متغیرهای مستقل با هم همبستگی بالایی داشته باشند. این مسئله می تواند بر دقت مدل تأثیر منفی بگذارد.
- ماتریس همبستگی: یک ماتریس مربعی است که در آن هر سلول نشان دهنده همبستگی بین دو متغیر است.
- ضریب همبستگی: عددی بین -۱ تا ۱ است که قدرت و جهت رابطه بین دو متغیر را نشان می دهد.

کاربردهای همبستگی

- انتخاب ویژگی: با شناسایی ویژگی هایی که همبستگی بالایی با متغیر هدف دارند، می توانیم ویژگی های مهم تر را برای مدل سازی انتخاب کنیم.

- کاهش ابعاد: با حذف ویژگی‌هایی که همبستگی بالایی با هم دارند، می‌توانیم ابعاد داده‌ها را کاهش دهیم و پیچیدگی مدل را کم کنیم.
- تشخیص چندخطی: با بررسی ماتریس همبستگی، می‌توانیم متغیرهایی که همبستگی بالایی با هم دارند را شناسایی کرده و مشکل چندخطی را برطرف کنیم.
- همبستگی در دیتافریم را با دستور زیر به کمک پانداس نمایش دادیم.

```
corr_matrix=df.corr()
corr_matrix
```

	country	seat_type	recommended	stars	route	type_of_traveller	day	month	year
recommended	-0.020134	-0.032596	1.000000e+00	0.015943	-0.029159	-4.358355e-17	0.014127	0.030933	-0.078662
stars	0.010548	-0.005959	1.594289e-02	1.000000	-0.003261	1.102317e-03	0.000082	0.000108	-0.000630
route	-0.034615	-0.019699	-2.915910e-02	-0.003261	1.000000	3.755330e-01	-0.003908	-0.005238	0.001380
type_of_traveller	0.017590	-0.002627	-4.358355e-17	0.001102	0.375534	1.000000e+00	-0.000794	-0.006332	0.003801
day	-0.003724	0.008989	1.412727e-02	0.000082	-0.003908	-7.939000e-04	1.000000	0.028156	0.032967
month	-0.010566	0.037127	3.093330e-02	0.000108	-0.005238	-6.331509e-03	0.028156	1.000000	-0.167861
year	0.005779	-0.008896	-7.866184e-02	-0.000630	0.001380	3.806671e-03	0.032967	-0.167861	1.000000

شکل ۳-۲۲- نمایش ضرایب همبستگی

در ادامه متغیر هدف را جدا می‌کنیم تا بتوانیم پیش بینی را انجام دهیم. کد زیر اینکار را برای ما انجام می‌دهد.

```
x=df.drop('recommended',axis='columns')
y=df.recommended
```

شکل ۳-۲۳- ایجاد متغیر هدف

تابع minmax

تابع minmax این تابع که در کتابخانه sklearn استفاده میشود عدد بزرگ هر ستون را یک در نظر میگیرد و باقی اعداد را به نسبت بزرگی آن عدد بین ۰ تا ۱ مقدار دهی میکند این کار باعث بهبود پیش بینی میشود و در شبکه های عصبی حتما باید اعداد بین ۰ تا ۱ داده شود در تحقیق ما نیز به دلیل اینکه از کلاس بندی و شبکه عصبی استفاده شده انجام این کار ضروری میباشد.

در بخش پایین این تابع را برای کل دیتافریم نوشتیم.

```
from sklearn.preprocessing import MinMaxScaler
mi=MinMaxScaler()
X=mi.fit_transform(X)
```

	0	1	2	3	4	5	6	7
0	0.363636	0.000000	0.50	0.444444	0.666667	0.000000	0.636364	1.0
1	0.969697	0.333333	0.25	1.000000	1.000000	1.000000	0.454545	1.0
2	0.393939	0.000000	0.25	0.333333	1.000000	1.000000	0.454545	1.0
3	0.393939	0.000000	0.50	0.666667	0.333333	1.000000	0.454545	1.0
4	0.136364	0.333333	0.50	0.222222	0.666667	0.933333	0.454545	1.0
5	0.712121	0.000000	0.25	0.000000	0.000000	0.833333	0.454545	1.0
6	0.969697	0.333333	0.25	0.888889	0.333333	0.833333	0.454545	1.0

شکل ۳-۲۴ - استفاده از تابع minmax

۱. تقسیم داده‌ها به مجموعه‌های آموزش و تست (Train-Test Split)

هدف: قبل از آموزش یک مدل یادگیری ماشین، داده‌ها به دو قسمت تقسیم می‌شوند:

مجموعه آموزش: برای آموزش مدل استفاده می‌شود.

مجموعه تست: برای ارزیابی عملکرد مدل روی داده‌های دیده نشده استفاده می‌شود.

نحوه انجام: از تابع train_test_split در کتابخانه sklearn استفاده می‌شود. این تابع داده‌های

ورودی (X) و برچسب‌ها (y) را به صورت تصادفی به دو قسمت تقسیم می‌کند.

در ادامه ما ۷۵٪ را برای آموزش مدل و ۲۵٪ را برای بررسی درستی جدا کردیم و بعد shape آن

ها را نمایش دادیم .

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2)

print(X_train.shape)

(2000, 8)

print(X_test.shape)

(500, 8)
```

شکل ۳-۲۵ - آماده کردن دیتا برای آموزش و بررسی درستی

مرحله train

در این مرحله به طور کلی مهم ترین مرحله برای یک پروژه آنالیز داده است هدف کلی این مرحله پیدا کردن راه حل هایی است که بتوانیم به دقت بالاتری پیش بینی این مراحل را انجام دهیم به طور کلی در پروژه دیتا ساینس دو نوع هدف مطرح هست کلاس بندی یا رگرسیون در اینجا ما به تحلیل کلاس بندی برای پیش بینی نیاز مند هستیم هر چند تحلیل های دیگری از قبیل خوشه بندی نیز مورد استفاده قرار میگیرد

در ادامه ابتدا به توضیح چند الگوریتم میپردازیم و نتیجه های آن روی سوال ما را بررسی میکنیم. امید است با توجه به تلاشات در مرحله قبل بتوانیم به دقت مورد نظر دست یابیم. در ادامه الگوریتم هایی که توانایی انجام کار کلاس بندی را دارند مورد بررسی قرار میدهیم.

رگرسیون لجستیک: ابزاری قدرتمند برای طبقه بندی

رگرسیون لجستیک یکی از پرکاربردترین الگوریتم های یادگیری ماشین برای مسائل طبقه بندی است. این الگوریتم به ما کمک می کند تا احتمال وقوع یک رویداد را بر اساس مجموعه ای از متغیرهای مستقل پیش بینی کنیم. به عنوان مثال، می توان از رگرسیون لجستیک برای پیش بینی اینکه آیا یک ایمیل اسپم است یا خیر، یا اینکه آیا یک بیمار به بیماری خاصی مبتلا می شود یا خیر، استفاده کرد.

چگونه رگرسیون لجستیک کار می کند؟

در رگرسیون لجستیک، به جای پیش بینی یک مقدار پیوسته (مانند رگرسیون خطی)، هدف پیش بینی احتمال وقوع یک رویداد است که به صورت یک مقدار بین ۰ تا ۱ بیان می شود. برای این کار، از یک تابع سیگموئید استفاده می شود که خروجی یک مدل خطی را به یک احتمال تبدیل می کند.

مراحل کلی رگرسیون لجستیک:

- ایجاد مدل خطی: مشابه رگرسیون خطی، یک مدل خطی ایجاد می شود که ترکیب خطی از متغیرهای مستقل و وزن های آن ها را نشان می دهد.
- تابع سیگموئید: خروجی مدل خطی به عنوان ورودی تابع سیگموئید قرار می گیرد. تابع سیگموئید خروجی را به یک مقدار بین ۰ تا ۱ نگاشت می کند که به عنوان احتمال تفسیر می شود.

- تعیین آستانه: یک آستانه مشخص می‌شود. اگر احتمال پیش‌بینی شده بیشتر از آستانه باشد، نمونه به یک کلاس و در غیر این صورت به کلاس دیگر اختصاص داده می‌شود.
- آموزش مدل: با استفاده از روش‌هایی مانند حداکثر درست‌نمایی، پارامترهای مدل (وزن‌ها) به گونه‌ای تنظیم می‌شوند که خطای پیش‌بینی به حداقل برسد.

مزایای رگرسیون لجستیک

- سادگی و تفسیرپذیری: مدل رگرسیون لجستیک نسبتاً ساده است و تفسیر ضرایب آن آسان می‌باشد.
- کاربرد گسترده: در بسیاری از حوزه‌ها از جمله پزشکی، بازاریابی، و امور مالی کاربرد دارد.
- قابلیت تعمیم به مسائل چند کلاسه: با استفاده از تکنیک‌هایی مانند one-vs-rest یا one-vs-one می‌توان رگرسیون لجستیک را برای مسائل چند کلاسه نیز بکار برد.

محدودیت‌های رگرسیون لجستیک

- خطی بودن رابطه: رگرسیون لجستیک فرض می‌کند که رابطه بین متغیرهای مستقل و متغیر وابسته خطی است.
- داده‌های جداپذیر خطی: برای داده‌هایی که به صورت خطی قابل جداسازی نیستند، عملکرد رگرسیون لجستیک ممکن است ضعیف باشد.
- توزیع داده‌ها: رگرسیون لجستیک فرض می‌کند که داده‌ها به صورت مستقل و یکسان توزیع شده‌اند.

کاربردهای رگرسیون لجستیک

- طبقه‌بندی اسپم: تشخیص ایمیل‌های اسپم از ایمیل‌های غیر اسپم
- تشخیص بیماری: پیش‌بینی احتمال ابتلا به یک بیماری بر اساس علائم و نتایج آزمایشات
- پیش‌بینی رفتار مشتری: پیش‌بینی اینکه آیا یک مشتری محصول خاصی را خریداری می‌کند یا خیر
- ارزیابی اعتبار: ارزیابی اعتبار اعتباری مشتریان

در کل، رگرسیون لجستیک یک ابزار قدرتمند برای طبقه‌بندی داده‌ها است که در بسیاری از حوزه‌ها کاربرد دارد. با درک اصول کارکرد این الگوریتم، می‌توانید از آن برای حل مسائل مختلف طبقه‌بندی استفاده کنید.

در ادامه ما پیش‌بینی را برای logesestic regression انجام دادیم و نتیجه دقت ۶۸٪ بود.

```
from sklearn.linear_model import LogisticRegression
lo=LogisticRegression()
lo.fit(X_train,y_train)

LogisticRegression()

lo.score(X_test,y_test)

0.68
```

شکل ۳-۲۶- آموزش الگوریتم logistic regression

الگوریتم درخت تصمیم: یک راه ساده برای تصمیم‌گیری ماشین

درخت تصمیم یکی از محبوب‌ترین و قابل فهم‌ترین الگوریتم‌های یادگیری ماشین است که برای حل مسائل طبقه‌بندی و رگرسیون استفاده می‌شود. تصور کنید می‌خواهید تصمیم بگیرید که امروز به پیاده‌روی بروید یا نه. شما عواملی مانند آب و هوا، دمای هوا و میزان رطوبت را در نظر می‌گیرید و بر اساس این عوامل تصمیم می‌گیرید. درخت تصمیم هم به همین شکل عمل می‌کند.

ساختار درخت تصمیم

درخت تصمیم شبیه به یک نمودار درختی است که در آن هر گره داخلی نشان‌دهنده یک ویژگی (feature) و هر شاخه نشان‌دهنده یک مقدار ممکن برای آن ویژگی است. برگ‌های درخت نیز کلاس‌ها یا مقادیر هدف را نشان می‌دهند.

گره ریشه: نقطه شروع درخت است و معمولاً مهم‌ترین ویژگی برای تقسیم داده‌ها انتخاب می‌شود.

گره‌های داخلی: این گره‌ها بر اساس یک ویژگی داده‌ها را به دو یا چند زیرمجموعه تقسیم می‌کنند.

برگ‌ها: این گره‌ها نشان‌دهنده کلاس یا مقدار پیش‌بینی شده برای نمونه‌های داده‌ای هستند که به آن برگ می‌رسند.

چگونه درخت تصمیم کار می‌کند؟

انتخاب ویژگی ریشه: اولین گره درخت، ویژگی مهم‌ترین برای تقسیم داده‌ها را انتخاب می‌کند. برای این کار از معیارهایی مانند اطلاعات گین ((Information Gain، گین ریشه‌ای (Gain Ratio) یا ضریب جینی (Gini Index) استفاده می‌شود.

تقسیم داده‌ها: داده‌ها بر اساس مقدار ویژگی انتخاب شده در گره ریشه به چندین زیرمجموعه تقسیم می‌شوند.

تکرار فرآیند: برای هر زیرمجموعه، مراحل ۱ و ۲ تکرار می‌شود تا زمانی که همه نمونه‌ها به یک کلاس خالص تعلق بگیرند یا یک شرط توقف دیگر برقرار شود.

مزایای درخت تصمیم

- سادگی و تفسیرپذیری: درخت تصمیم به صورت یک نمودار درختی نمایش داده می‌شود و به راحتی قابل درک است.
- قابلیت مدیریت داده‌های مختلف: درخت تصمیم می‌تواند با انواع مختلف داده‌ها (عددی، اسمی، ترتیبی) کار کند.
- نیاز به پیش‌پردازش کم: درخت تصمیم به پیش‌پردازش زیادی نیاز ندارد.
- قابلیت استفاده برای مسائل طبقه‌بندی و رگرسیون: درخت تصمیم هم برای طبقه‌بندی و هم برای پیش‌بینی مقادیر عددی قابل استفاده است.

معایب درخت تصمیم

- حساسیت به نویز: درخت تصمیم ممکن است به نویز داده‌ها حساس باشد و به راحتی بیش‌برازش (Overfitting) شود.
- ناپایداری: تغییرات کوچک در داده‌ها ممکن است منجر به تغییرات بزرگ در ساختار درخت شود.
- مشکل در نمایش روابط غیرخطی پیچیده: درخت تصمیم ممکن است برای نمایش روابط پیچیده بین متغیرها مناسب نباشد.

کاربردهای درخت تصمیم

- طبقه‌بندی: تشخیص اسپم، تشخیص بیماری، پیش‌بینی مشتریان
- رگرسیون: پیش‌بینی قیمت خانه، پیش‌بینی فروش
- انتخاب ویژگی: شناسایی مهم‌ترین ویژگی‌های داده‌ها

در ادامه ما به بررسی الگوریتم درخت تصمیم پرداختیم و تعداد حداکثر ریشه را ۱۵۰ برای آن در نظر گرفتیم و توانستیم به دقت ۶۳/۸٪ برسیم.

```
from sklearn.tree import DecisionTreeClassifier
Do=DecisionTreeClassifier(max_leaf_nodes=150)
Do.fit(X_train,y_train)

DecisionTreeClassifier
DecisionTreeClassifier(max_leaf_nodes=150)

Do.score(X_test,y_test)

0.636
```

شکل ۳-۲۷- آموزش الگوریتم درخت تصمیم

تقویت گرادیان (Gradient Boosting): یک روش قدرتمند برای یادگیری ماشین

تقویت گرادیان (Gradient Boosting) یک روش یادگیری ماشین است که در آن چندین مدل ضعیف (مثل درخت‌های تصمیم) به صورت متوالی آموزش داده می‌شوند تا یک مدل قوی ایجاد کنند. این روش به دلیل دقت بالا و توانایی در مدیریت انواع مختلف داده‌ها، یکی از محبوب‌ترین روش‌ها در یادگیری ماشین است.

چگونه تقویت گرادیان کار می‌کند؟

- مدل پایه: ابتدا یک مدل پایه (مثل یک درخت تصمیم ساده) آموزش داده می‌شود.
- محاسبه باقیمانده‌ها: پس از آموزش مدل پایه، باقیمانده‌های پیش‌بینی (تفاوت بین مقادیر واقعی و مقادیر پیش‌بینی شده) محاسبه می‌شوند.

- آموزش مدل بعدی: یک مدل جدید آموزش داده می‌شود تا این باقیمانده‌ها را پیش‌بینی کند.
- ترکیب مدل‌ها: مدل‌های قبلی و جدید با هم ترکیب می‌شوند تا یک مدل قوی‌تر ایجاد شود.
- تکرار مراحل: مراحل ۳ و ۴ تا رسیدن به یک تعداد مشخص از مدل‌ها یا تا زمانی که بهبود قابل توجهی در دقت حاصل نشود، تکرار می‌شوند.

چرا تقویت گرادیان موثر است؟

- یادگیری تدریجی: با افزودن تدریجی مدل‌های جدید، الگوریتم می‌تواند به تدریج پیچیدگی مدل را افزایش دهد و به الگوهای پیچیده‌تر داده‌ها دست یابد.
- کاهش خطا: با تمرکز بر باقیمانده‌ها در هر مرحله، الگوریتم می‌تواند به تدریج خطای مدل را کاهش دهد.
- انعطاف‌پذیری: تقویت گرادیان می‌تواند برای مسائل مختلفی از جمله طبقه‌بندی، رگرسیون و رتبه‌بندی استفاده شود.
- دقت بالا: معمولاً مدل‌های تقویت گرادیان دقت بسیار بالایی دارند و در بسیاری از مسابقات یادگیری ماشین برنده می‌شوند.

کاربردهای تقویت گرادیان

- طبقه‌بندی: تشخیص تقلب، تشخیص بیماری، طبقه‌بندی متن
- رگرسیون: پیش‌بینی قیمت خانه، پیش‌بینی فروش
- رتبه‌بندی: توصیه‌گرهای محصولات، موتورهای جستجو

مزایا و معایب تقویت گرادیان

- مزایا: دقت بالا، انعطاف‌پذیری، قابلیت مدیریت داده‌های پیچیده
- معایب: پیچیدگی مدل، زمان آموزش طولانی‌تر نسبت به برخی الگوریتم‌های دیگر، احتمال بیش‌برازش

در کل، تقویت گرادیان یک روش قدرتمند و همه‌کاره برای یادگیری ماشین است که در بسیاری از کاربردهای دنیای واقعی مورد استفاده قرار می‌گیرد.

در اینجا ما gradient boost را با استفاده از کتابخانه sklearn برای داده‌ها پیاده‌سازی کردیم و آرگومان‌های نرخ یادگیری را با مقدار ۰/۵ و ۱۰۰ درخت تصمیم‌گیری را برای الگوریتم لحاظ کردیم و حداکثر عمق را نیز ۳ در نظر گرفتیم و در نهایت به دقت ۶۶٪ رسیدیم. مراحل کد نویسی الگوریتم به شرح زیر است.

```
from sklearn.ensemble import GradientBoostingClassifier
G=GradientBoostingClassifier(learning_rate=0.5,n_estimators=100,max_depth=3)
G.fit(X_train,y_train)
```

▼ GradientBoostingClassifier

GradientBoostingClassifier(learning_rate=0.5)

```
G.score(X_test,y_test)
```

0.66

شکل ۳-۲۸- پیاده‌سازی الگوریتم تقویت گرادیان

در ادامه یکی دیگر از الگوریتم‌های ensemble را مورد بررسی قرار دادیم این الگوریتم جنگل تصادفی نام دارد که هم برای کلاس بندی استفاده میشود و هم برای رگرسیون و شباهت زیادی به الگوریتم تقویت گرادیان دارد. جنگل تصادفی را با sklearn پیاده‌سازی نمودیم و آرگومان‌های ما هم ۱۰۰ درخت تصمیم‌گیری و حداکثر عمق هم ۱۰۰ میباشد. پیاده‌سازی الگوریتم در زیر قابل مشاهده است.

```
from sklearn.ensemble import RandomForestClassifier
ro=RandomForestClassifier(n_estimators=100,max_depth=100)
ro.fit(X_train,y_train)
```

▼ RandomForestClassifier

RandomForestClassifier(max_depth=100)

```
ro.score(X_test,y_test)
```

0.638

شکل ۳-۲۹- پیاده‌سازی الگوریتم جنگل تصادفی

ماشین بردار پشتیبان (SVM)

ماشین بردار پشتیبان (Support Vector Machine) یا SVM یکی از قدرتمندترین و پرکاربردترین الگوریتم‌های یادگیری ماشین است که برای طبقه‌بندی و رگرسیون استفاده می‌شود. SVM با پیدا کردن بهترین مرز تصمیم‌گیری (Hyperplane) بین داده‌های مختلف، به طبقه‌بندی آن‌ها می‌پردازد.

چگونه SVM کار می‌کند؟

یافتن بهترین ابرصفحه: SVM به دنبال یافتن ابرصفحه‌ای است که با حداکثر فاصله از نزدیک‌ترین نقاط داده‌های هر کلاس قرار داشته باشد. این نقاط را بردارهای پشتیبان Support Vectors می‌نامند. حداکثر کردن حاشیه: فاصله بین ابرصفحه و نزدیک‌ترین بردارهای پشتیبان را حاشیه می‌نامند. SVM سعی می‌کند حاشیه را حداکثر کند تا مدل قوی‌تر و تعمیم‌پذیرتر شود. تبدیل داده‌ها به فضای با ابعاد بالاتر: در مواردی که داده‌ها به صورت خطی قابل جداسازی نباشند، SVM از تکنیکی به نام "تبدیل هسته" (Kernel Trick) استفاده می‌کند تا داده‌ها را به یک فضای با ابعاد بالاتر نگاشت کند و در آن فضا، داده‌ها را به صورت خطی جداسازی نماید.

مزایای SVM

- دقت بالا: SVM معمولاً دقت بالایی در مسائل طبقه‌بندی و رگرسیون دارد.
- تعریف مدل: SVM یک مدل با تعداد پارامترهای کم است که منجر به تعمیم‌پذیری بهتر می‌شود.
- قابلیت مدیریت داده‌های با ابعاد بالا: SVM می‌تواند با داده‌های با ابعاد بالا به خوبی کار کند.
- قابلیت استفاده برای مسائل طبقه‌بندی و رگرسیون: SVM هم برای مسائل طبقه‌بندی دو کلاسه و چند کلاسه و هم برای مسائل رگرسیون قابل استفاده است.

معایب SVM

- زمان آموزش بالا: برای مجموعه داده‌های بزرگ، آموزش SVM می‌تواند زمان‌بر باشد.
- انتخاب پارامترهای مناسب: انتخاب پارامترهای مناسب برای هسته و تنظیم پارامترهای پهنایی می‌تواند پیچیده باشد.

کاربردهای SVM

- طبقه‌بندی تصاویر: تشخیص اشیاء در تصاویر، تشخیص چهره
- پردازش زبان طبیعی: طبقه‌بندی متن، تجزیه احساسات
- بیوانفورماتیک: پیش‌بینی ساختار پروتئین، طبقه‌بندی ژن‌ها
- تشخیص تقلب: تشخیص تراکنش‌های مالی تقلبی

انواع هسته در SVM

- هسته خطی: برای داده‌هایی که به صورت خطی قابل جداسازی هستند.
- هسته چندجمله‌ای: برای روابط غیرخطی پیچیده‌تر.
- هسته (Radial Basis Function) RBF: یکی از پرکاربردترین هسته‌ها که برای داده‌های غیرخطی بسیار موثر است.
- هسته سیگموئید: مشابه تابع سیگموئید در شبکه‌های عصبی.

جمع‌بندی

SVM یک الگوریتم قدرتمند و همه‌کاره است که در بسیاری از حوزه‌های یادگیری ماشین کاربرد دارد. با درک اصول کارکرد SVM و انتخاب مناسب پارامترهای آن، می‌توانید مدل‌های طبقه‌بندی و رگرسیون بسیار دقیقی ایجاد کنید. در ادامه الگوریتم صفحه بردار پشتیبان را پیاده‌سازی کردیم آرگومان‌های ما به این شرح است:

Kernel=poly: توانایی قدرت بالاتر در موارد غیر خطی

C=200: تعامل بین حداکثر سازی حاشیه و حداقل سازی خطا و افزایش دقت در موارد نویز

Degree=3: استفاده از خط‌های درجه ۳

پس از توزیحات فوق و پیاده‌سازی به دقت ۶۹/۴٪ رسیدیم که دقت نسبتاً خوبی می‌باشد.

```
from sklearn.svm import SVC
svc=SVC(kernel='poly',C=200,degree=3)
svc.fit(X_train,y_train)
```

SVC

SVC(C=200, kernel='poly')

```
svc.score(X_test,y_test)
```

0.694

شکل ۳-۳- پیاده‌سازی svm

در ادامه به دلیل اینکه قوی ترین الگوریتم یادگیری ماشین به غیر از شبکه عصبی شد classification report را برای آن پیاده سازی نمودیم.

در حوزه یادگیری ماشین و به خصوص در مسائل طبقه بندی، ارزیابی عملکرد مدل از اهمیت بالایی برخوردار است. برای این منظور، از معیارهای مختلفی استفاده می شود که هر کدام جنبه خاصی از عملکرد مدل را نشان می دهند. سه معیار مهم در این زمینه، دقت (Precision)، فراخوانی (Recall) و (F1-Score) هستند.

دقت (Precision)

دقت نشان می دهد از بین نمونه هایی که مدل مثبت پیش بینی کرده است، چه تعداد واقعاً مثبت بوده اند. به عبارت دیگر، دقت بیان می کند که مدل تا چه اندازه در تشخیص نمونه های مثبت موفق بوده است.

فرمول دقت:

دقت = تعداد نمونه های مثبت صحیح / تعداد کل نمونه هایی که مثبت پیش بینی شده اند

فراخوانی (Recall)

فراخوانی نشان می دهد از بین تمام نمونه های مثبت واقعی، چه تعداد توسط مدل به درستی مثبت پیش بینی شده اند. به عبارت دیگر، فراخوانی بیان می کند که مدل تا چه اندازه توانسته است همه نمونه های مثبت را شناسایی کند.

فرمول فراخوانی:

فراخوانی = تعداد نمونه های مثبت صحیح / تعداد کل نمونه های مثبت واقعی

F1-Score

F1-Score میانگین هارمونیک دقت و فراخوانی است و به عنوان یک معیار جامع برای ارزیابی مدل استفاده می شود. F1-Score زمانی که دقت و فراخوانی هر دو مهم باشند، بسیار مفید است.

F1-Score میانگین هارمونیک دقت و فراخوانی، تعادل بین دقت و فراخوانی

در ادامه موارد فوق گفته شده را پیاده سازی کردیم.

```

y_pred=svc.predict(X_test)
from sklearn.metrics import confusion_matrix,classification_report
print(classification_report(y_test,y_pred))

```

	precision	recall	f1-score	support
0	0.70	0.98	0.81	340
1	0.65	0.09	0.16	160
accuracy			0.69	500
macro avg	0.67	0.54	0.49	500
weighted avg	0.68	0.69	0.61	500

شکل ۳-۳۱- پیاده سازی گزارش طبقه بندی

شبکه‌های عصبی مصنوعی (ANN)

شبکه‌های عصبی مصنوعی (Artificial Neural Networks) الگوریتم‌هایی هستند که از ساختار مغز انسان الهام گرفته‌اند و برای یادگیری الگوها و روابط پیچیده در داده‌ها استفاده می‌شوند. این شبکه‌ها از مجموعه‌ای از واحدهای پردازشی به نام نورون تشکیل شده‌اند که به هم متصل هستند و به صورت موازی کار می‌کنند.

چگونه یک شبکه عصبی کار می‌کند؟

لایه ورودی: داده‌های ورودی به شبکه وارد می‌شوند. هر داده ورودی به یک نورون در لایه ورودی متصل است.

- لایه‌های پنهان: داده‌ها از لایه ورودی به لایه‌های پنهان منتقل می‌شوند. در این لایه‌ها، داده‌ها پردازش شده و ویژگی‌های پیچیده‌تری از داده‌ها استخراج می‌شود.
- لایه خروجی: در نهایت، داده‌ها به لایه خروجی می‌رسند و شبکه تصمیم می‌گیرد که کدام کلاس یا مقدار به داده ورودی اختصاص داده شود.
- آموزش شبکه: شبکه عصبی با استفاده از یک مجموعه داده آموزشی آموزش داده می‌شود. در طی آموزش، وزن‌های اتصالات بین نورون‌ها به گونه‌ای تنظیم می‌شوند که خطای بین خروجی پیش‌بینی شده و خروجی واقعی به حداقل برسد.

اجزای اصلی یک شبکه عصبی

- نورون: واحد اصلی پردازش در شبکه عصبی است که داده‌های ورودی را دریافت می‌کند، آن‌ها را پردازش می‌کند و یک خروجی تولید می‌کند.
- وزن‌ها: اعدادی هستند که قدرت اتصال بین نورون‌ها را نشان می‌دهند. وزن‌ها در طی آموزش شبکه تنظیم می‌شوند.
- تابع فعال‌سازی: تابعی است که خروجی یک نورون را تعیین می‌کند. توابع فعال‌سازی مختلفی مانند سیگموئید، تانج و ReLU وجود دارد.
- لایه: مجموعه‌ای از نورون‌ها است که به صورت موازی کار می‌کنند.

انواع شبکه‌های عصبی

- شبکه‌های پرسپترون چند لایه (MLP): ساده‌ترین نوع شبکه‌های عصبی هستند و از چندین لایه تشکیل شده‌اند.
- شبکه‌های کانولوشنی (CNN): برای پردازش داده‌های تصویری بسیار مناسب هستند و از فیلترهایی برای استخراج ویژگی‌های محلی تصاویر استفاده می‌کنند.
- شبکه‌های بازگشتی (RNN): برای پردازش داده‌های دنباله‌ای مانند متن و صوت مناسب هستند و از حافظه داخلی برای ذخیره اطلاعات گذشته استفاده می‌کنند.

کاربردهای شبکه‌های عصبی

- تشخیص تصویر: تشخیص اشیاء در تصاویر، تشخیص چهره
- پردازش زبان طبیعی: ترجمه ماشینی، تحلیل احساسات
- پیش‌بینی سری زمانی: پیش‌بینی قیمت سهام، پیش‌بینی تقاضا
- توصیه‌گرهای سیستم: سیستم‌های توصیه‌گر محصولات

مزایای شبکه‌های عصبی

- توانایی یادگیری الگوهای پیچیده: شبکه‌های عصبی می‌توانند الگوهای بسیار پیچیده‌ای را در داده‌ها یاد بگیرند.
- انعطاف‌پذیری: شبکه‌های عصبی می‌توانند برای حل مسائل مختلفی استفاده شوند.
- دقت بالا: در بسیاری از مسائل، شبکه‌های عصبی دقت بسیار بالایی دارند.

معایب شبکه‌های عصبی

- نیاز به داده‌های زیاد: برای آموزش یک شبکه عصبی به داده‌های زیادی نیاز است.
- زمان آموزش طولانی: آموزش شبکه‌های عصبی می‌تواند زمان‌بر باشد.
- پیچیدگی: درک و پیاده‌سازی شبکه‌های عصبی پیچیده می‌تواند دشوار باشد.

در کل، شبکه‌های عصبی یکی از قدرتمندترین ابزارها در حوزه یادگیری ماشین هستند که در بسیاری از کاربردهای دنیای واقعی مورد استفاده قرار می‌گیرند.

در الگوریتم آخر که مورد بررسی قرار می‌دهیم به شبکه عصبی میپردازیم این الگوریتم را با استفاده از کتابخانه keras که درون کتابخانه tensorflow ساخته شده است پیاده سازی کردیم. ابتدا لایه‌ها را ایجاد کردیم که ورودی آن را اندازه ستون و پارامتر بعدی را خالی گذاشتیم تا کل سطرها را دریافت کند و از ۸ لایه شروع کردیم تابع تمام لایه‌ها غیر از لایه آخر را relu لحاظ کردیم تا مقادیر وزن دار مثبت را به لایه‌های بعدی بفرستند لایه بعدی ۴ لایه بعد آن ۲ لایه و در آخر لایه پایانی را با تابع فعال ساز sigmoid به دلیل اینکه سوال ما طبقه بندی دو گانه بود پیاده سازی کردیم.

در مرحله بعد کامپایل مدل را با استفاده از تابع هزینه binary_crossentropy پیاده سازی کرده و تابع بهینه ساز adam به دلیل سرعت و دقت بالا پیاده سازی کردیم و ماتریس accuracy را به آن افزودیم تا بتوانیم دقت مدل را در مرحله train ببینیم

در آخر مدل را با ۷۵۰ epoch آموزش دادیم کد مراحل را با keras به شکل زیر پیاده سازی کردیم

```
from tensorflow import keras

model=keras.Sequential([
    keras.layers.Dense(8,input_shape=(8,),activation='relu'),
    keras.layers.Dense(4,activation='relu'),
    keras.layers.Dense(2,activation='relu'),
    keras.layers.Dense(1,activation='sigmoid'),
])

model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])
model.fit(X_train,y_train,epochs=750)
```

شکل ۳-۳- پیاده سازی شبکه عصبی مصنوعی

در بخش پایانی epoch ۵ آخر را نمایش می دهیم.

```
Epoch 746/750
63/63 [=====] - 0s 4ms/step - loss: 0.6319 - accuracy: 0.6635
Epoch 747/750
63/63 [=====] - 0s 4ms/step - loss: 0.6323 - accuracy: 0.6645
Epoch 748/750
63/63 [=====] - 0s 4ms/step - loss: 0.6320 - accuracy: 0.6625
Epoch 749/750
63/63 [=====] - 0s 4ms/step - loss: 0.6316 - accuracy: 0.6620
Epoch 750/750
63/63 [=====] - 0s 4ms/step - loss: 0.6321 - accuracy: 0.6650
<keras.callbacks.History at 0x2958f813670>
```

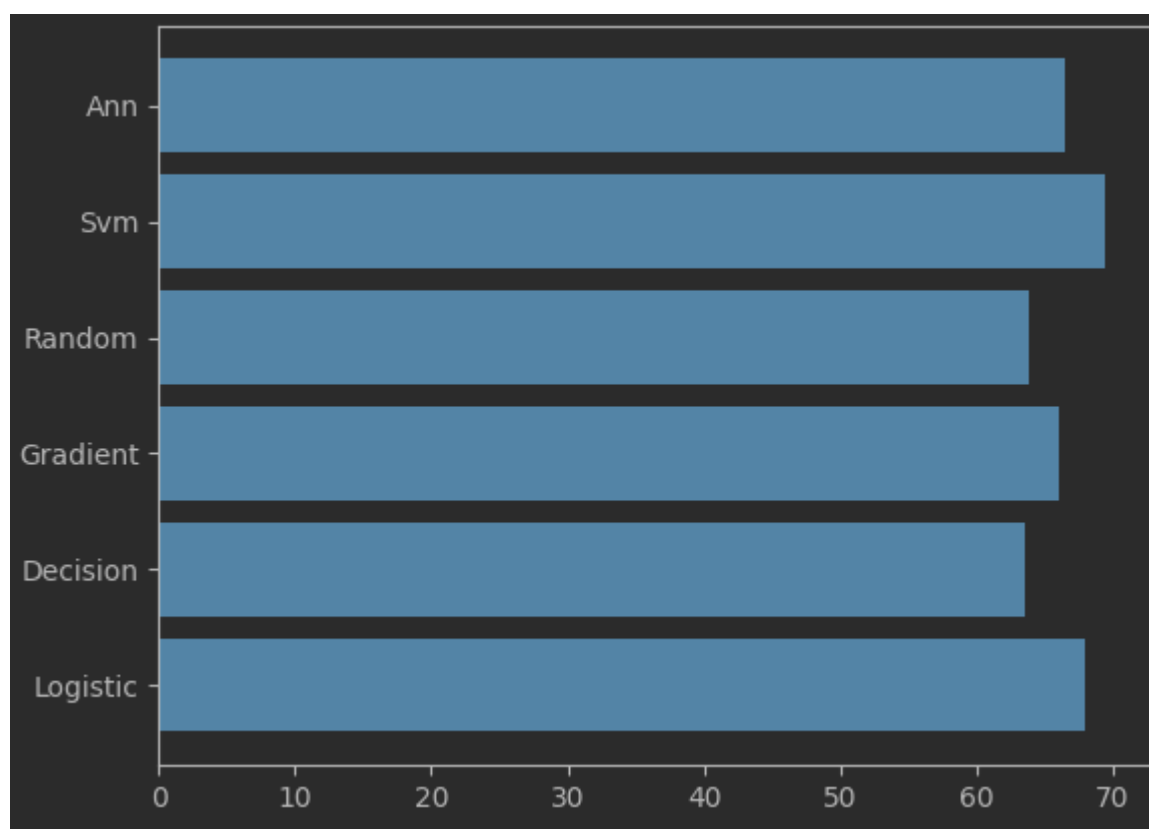
شکل ۳-۳۳-۵ آموزش برتر الگوریتم

فصل چهارم

نتیجه گیری و پیشنهادات

۴-۱- نتیجه گیری

در بخش نتیجه گیری به این نتیجه رسیدیم که الگوریتم svm قوی ترین الگوریتم برای پیش بینی ما بود به طور کلی در الگوریتم های یادگیری ماشین در بخش طبقه بندی الگوریتم svm و شبکه های عصبی قوی تر هستند و در بخش رگرسیون الگوریتم های ensemble و شبکه های عصبی قوی تر هستند و در پروژه های واقعی در اکثر مواقع این الگوریتم ها جواب بهتری میدهند. در زیر مقایسه ای از نتیجه کلی تحقیقمان را با نمودار شرح دادیم .



شکل ۴-۱- مقایسه نهایی الگوریتم های یادگیری ماشین

۴-۲- پیشنهادات

در پایان بحث چند پیشنهاد هم در بحث داده کاوی داشتیم در عصر هوش مصنوعی اگر کشور بتواند که در مباحث دارو سازی از الگوریتم های یادگیری ماشین استفاده شود شاید هزینه ها به مراتب کمتر شود و بتوانیم به دارو های با بازده به مراتب بهتری دست پیدا کنیم .

منابع

استفاده از مطالب موجود در سایت‌های

- <https://scikit-learn.org/stable/>
 - <https://numpy.org/>
 - <https://keras.io/>
 - <https://gemini.google.com/>
 - www.wikipedia.org
 - <https://www.kaggle.com/datasets/minnikeswarrao/british-airlines>
-