

**فهرست مطالب**

[**مشخصات کلی پروژه** 5](#_Toc174028742)

[**بیان مسأله** 7](#_Toc174028743)

[**معرفی داده‌ها** 8](#_Toc174028744)

[**پرسش‌ها** 8](#_Toc174028745)

[**پاکسازی داده‌ها و تجزیه و تحلیل اکتشافی** 9](#_Toc174028746)

[**پاکسازی داده‌ها** 9](#_Toc174028747)

[**تحلیل اکتشافی داده‌ها** 9](#_Toc174028748)

[**کتابخانه‌ها** 10](#_Toc174028749)

[**شروع پروژه** 14](#_Toc174028750)

[**1.** **تعریف مسیر فایل** 14](#_Toc174028751)

[**2.** **ساخت مسیر کامل فایل** 14](#_Toc174028752)

[**3.** **باز کردن و خواندن فایل CSV** 14](#_Toc174028753)

[**4.** **بررسی ابعاد داده‌ها** 14](#_Toc174028754)

[**5.** **تحلیل و تفسیر کلی** 14](#_Toc174028755)

[**اهمیت مراحل بالا** 15](#_Toc174028756)

[**معرفی داده‌های پروژه:** 16](#_Toc174028757)

[**خواندن داده‌ها** 17](#_Toc174028758)

[**بررسی بانک داده‌ها** 18](#_Toc174028759)

# **مشخصات کلی پروژه**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **دانشگاه آزاد اسلامی واحد پردیس- گروه کامپیوتر و فناوری‌اطلاعات** | | | | |
| **ردیف** | **نام و نام خانوادگی** | **موقعیت شغلی** | **شماره دانشجویی** | **لینک تلگرام** |
| 1 | صالح عابدی‌نژاد | برنامه نویس و  تحلیلگر داده | 40150140893010 |  |
| **موضوع پروژه**: پیش‌بینی خطر ابتلا به دیابت | | | | |
| **تعریف پروژه:** تجزیه و تحلیل داده‌های مرتبط با بیماری دیابت با استفاده از الگوریتم‌های یادگیری ماشین  **هدف:** ایجاد یک بستر برای درک شرایط موجود و تولید راهکار موثر در پیشگیری از ابتلا به دیابت  **خروجی:**   * بررسی صحت داده‌ها * پاکسازی داده‌ها EDA * Correlation یا بررسی همبستگی داده‌ها با یکدیگر * پیاده سازی الگوریتم‌های یادگیری مختلف مناسب با داده‌های اکتشافی    **نتایج و فواید پروژه:** | | | | |
| **زبان برنامه نویسی**: پایتون | | | | |
| **سایر ابزارهای مورد استفاده**: ژوپیتر نوتبوک، matplotlib, scipy, numpy, pandas و ...  **طراحی صفحه:** مایکروسافت ورد، اکسل، کورل دراو، فتوشاپ و ... | | | | |

**توجه!**

فایل‌های مرتبط با پروژه شامل داکیومنت، کد پروژه و سورس‌های مرتبط در گیت‌هاب پروژه بارگزاری شده است.

[لینک گیت‌هاب پروژه](https://github.com/imsalione/Diabetes)

# **بیان مسأله**

دیابت یکی از شایع‌ترین بیماری‌ها در دنیاست که هر ساله بر زندگی میلیون‌ها نفر تأثیر می‌گذارد و بار مالی قابل توجهی را بر اقتصاد کشورها به دنبال دارد. دیابت یک بیماری مزمن و پیش‌رونده است که در آن افراد قابلیت تنظیم سطح قند خون را به خوبی از دست می‌دهند. این وضعیت در نهایت منجر به کاهش کیفیت زندگی و کاهش امید به زندگی می‌شود.

پس از تجزیه غذاهای مختلف به قندها در طول فرآیند هضم، قند وارد خون می‌شود. این عمل باعث می‌شود که غده پانکراس انسولین آزاد کند تا سلول‌ها بتوانند از قند موجود در خون برای تولید انرژی استفاده ‌کنند. دیابت به عدم تولید کافی انسولین یا ناتوانی بدن در استفاده از انسولین برای تبدیل قند خون به انرژی اتلاق می‌شود.

عوارضی همچون بیماری‌های قلبی، از دست دادن دید، قطع اندام‌های پایین در نتیجه‌ی بریدگی، بیماری‌های کلیوی و موارد دیگر همگی ناشی از سطوح بالای قند خون در افراد دیابتی است. برای دیابت هیچ درمان مشخصی وجود ندارد. با این حال راهکارهایی مانند از دست‌دادن وزن، تغذیه سالم، فعالیت بدنی و دریافت درمان‌های پزشکی در کاهش آسیب‌های بیماری و کنترل آن نقش مهمی ایفا می‌کنند. تشخیص زودهنگام می‌تواند منجر به تغییرات سبک زندگی مبتلایان و درمان‌های موثر‌تر شود. بنابراین مدل‌های پیش‌بینی خطر دیابت، ابزارهای بسیار مهمی برای ارتقای آگاهی جامعه و مسئولان بهداشت و سلامت عمومی در کشورها است.

بر اساس اعلام مرکز کنترل و پیشگیری بیماری‌های آمریکا (CDC) تا سال ۲۰۱۸، ۳۴/۲ میلیون آمریکایی دیابت دارند و 88 میلیون نفر نیز پیش‌دیابت دارند. علاوه بر این، CDC برآورد می‌کند که ۱ نفر از هر ۵ نفر دیابتی، و تقریبا ۸ نفر از هر ۱۰ نفر پیش‌دیابتی از وضعیت خود اطلاعی ندارند. در حالی که انواع مختلفی از دیابت وجود دارد، دیابت نوع دوم شایع‌ترین نوع آن است و شیوع آن بستگی به سن، تحصیلات، درآمد، محل زندگی، نژاد و سایر عوامل تعیین‌کننده‌ی اجتماعی سلامت دارد. بار بیماری بیشتر بر دوش افراد با وضعیت اقتصادی پایین است. دیابت همچنین بار سنگینی را بر اقتصاد به دنبال دارد. مجموع هزینه‌‌ی افراد تشخیص‌داده شده به دیابت تقریبا ۳۲۷ میلیارد دلار، و هزینه‌های کل به همراه افراد دیابتی تشخیص داده نشده و افراد پیش‌دیابتی در حدود سالانه ۴۰۰ میلیارد دلار برآورد می‌شود.

# **معرفی داده‌ها**

سیستم نظارت بر عوامل مخاطرات رفتاری (BRFSS) یک نظرسنجی تلفنی مربوط به سلامت است که سالانه توسط مرکز کنترل و پیشگیری از بیماری‌ها در آمریکا جمع‌آوری می‌شود. هر سال، این نظرسنجی پاسخ‌های بیش از ۴۰۰،۰۰۰ آمریکایی را در مورد رفتارهای خطرناک سلامت، بیماری‌های مزمن و استفاده از خدمات پیشگیری جمع‌آوری می‌کند. این نظرسنجی هر ساله از سال ۱۹۸۴ برگزار می‌شود. نتایج این نظرسنجی در سال 2015 در فایلی با فرمت csv در دسترس است. این مجموعه داده اصلی شامل پاسخ‌های ۴۴۱،۴۵۵ پرسش‌شونده و ۳۳۰ ویژگی است. این ویژگی‌ها یا به‌صورت مستقیم از شرکت‌کنندگان پرسیده شده‌اند و یا متغیرهای محاسبه‌شده بر اساس پاسخ‌های شرکت‌کنندگان فردی هستند.

جدول زیر داده‌ی تمیز شده شامل 21 ویژگی و ۲۵۳،۸۶۰ ردیف از پاسخ‌های افراد به نظرسنجی BRFSS در سال 2015 است. متغیر هدف (Diabetes\_012)  در این داده از سه کلاس تشکیل شده است:

1. عدم ابتلا به دیابت و یا دیابت در دوران بارداری (0)
2. پیش دیابت (1)
3. دیابت (2)

## **پرسش‌ها**

1. آیا نتایج نظرسنجی BRFSS شانسی برای ارائه یک پیش‌بینی قابل قبول از وضعیت فرد دارد؟
2. آیا توازن و تعداد کلاس‌های متغیر هدف (ابتلا و عدم ابتلا به دیابت) به یکدیگر نزدیک است؟ عدم توازن کلاس‌های متغیر هدف چه تاثیری بر مدلسازی خواهد داشت؟
3. با توجه به پاسخ به پرسش  قبل آیا تلفیق کلاس پیش‌دیابت و دیابت می‌تواند تصمیم معقولی باشد؟
4. حتی پس از تلفیق کلاس‌های دیابت و پیش‌دیابت، باز هم ممکن است توازن دو کلاس ایجاد شده (عدم دیابت / دیابت) یکسان نباشد. برای حل مسئله عدم توازن کلاس‌های متغیر هدف چه راهکاری باید در پیش گرفت؟
5. ایجاد گزارش خطر ابتلا به دیابت یک مدل دسته‌بندی (classification) و عملکرد آن را با استفاده از داده‌های train و validation و test
6. با استفاده از مجموعه validation بهترین پارامتر(ها)ی هر مدلی که ایجاد می‌کنید را تخمین بزنید.
7. مهم‌ترین متغیرها در پیش‌بینی دیابت کدام متغیرها هستند؟
8. کدام متغیرها در پیش‌بینی دیابت نقشی بازی نمی‌کنند و بهتر از است آن‌ها را از مدل نهایی حذف کنیم تا اطلاعات غیرضروری از کاربر دریافت نکنیم؟
9. آیا مدل ما می‌تواند در مدت زمان معقولی (مثلا در کمتر یک ثانیه) پاسخ خود را ارائه کند؟

## **پاکسازی داده‌ها و تجزیه و تحلیل اکتشافی**

پاسخ‌های این پروژه در سه قسمت مجزا که هر یک شامل یک فایل است بارگزاری شده است.

این سه قسمت عبارتند از:

1. وارد کردن داده‌ها، پاکسازی داده‌ها و تحلیل اکتشافی
2. مدل‌سازی و بررسی کیفیت مدل
3. فایل ارائه نتایج در فرمت pdf

### **پاکسازی داده‌ها**

تقریبا در تمام پروژه‌های تحلیل داده پس از وارد کردن داده‌ها به محیط کار متوجه می‌شوید داده‌ها آن‌طور که شما می‌خواهید نیستند. تغییر و تبدیل داده‌ها جز اولین کارهایی هستند که لازم است شما انجام دهید.

### **تحلیل اکتشافی داده‌ها**

سوالاتی که با استفاده از تحلیل اکتشافی داده‌ها قابل پاسخ است دراین بخش آورده شده است. همچنین بررسی‌هایی در خصوص توزیع داده‌ها و یا فهم بیشتر از داده‌ها ذکر شده است.

## **کتابخانه‌ها**

کتابخانه در پایتون مجموعه‌ای از ماژول‌ها و توابع از پیش نوشته شده است که به توسعه‌دهندگان امکان می‌دهد تا به راحتی وظایف مختلف برنامه‌نویسی را انجام دهند بدون اینکه نیاز باشد همه چیز را از ابتدا بنویسند. این کتابخانه‌ها برای انجام وظایف متنوعی مانند محاسبات ریاضی، پردازش داده‌ها، ایجاد نمودارها، یادگیری ماشین، و تعامل با سیستم عامل طراحی شده‌اند. با استفاده از کتابخانه‌ها، توسعه‌دهندگان می‌توانند کارهای پیچیده را با استفاده از کدهای ساده‌تر و قابل فهم‌تر انجام دهند، که این به افزایش کارایی و کاهش خطاهای برنامه‌نویسی کمک می‌کند. کتابخانه‌های مورد استفاده در این پروژه به شرح زیر می‌باشد:

1. **os:**

کاربرد: کار با سیستم عامل، مدیریت فایل‌ها، دایرکتوری‌ها و وظایف مربوط به سیستم.

مثال: `os.path.join` برای ترکیب مسیرها به صورت مناسب برای سیستم عامل.

1. **collections:**

کاربرد: ارائه ساختارهای داده‌ای خاص مثل شمارنده‌ها (Counters)، دیکشنری‌های پیش‌فرض (defaultdicts) و نام‌گذاری شده‌ها (named tuples).

مثال: `Counter` برای شمارش تعداد تکرار عناصر در یک لیست.

1. **matplotlib.pyplot:**

کاربرد: ایجاد نمودارها و پلات‌های گرافیکی.

مثال: `plt.plot` برای رسم نمودار خطی.

1. **numpy:**

کاربرد: انجام محاسبات عددی و کار با آرایه‌ها و ماتریس‌ها به صورت کارآمد.

مثال: `np.array` برای ایجاد یک آرایه.

1. **pandas:**

کاربرد: دستکاری و تحلیل داده‌ها، ارائه ابزارهایی برای کار با داده‌های ساختار یافته مانند DataFrameها و Seriesها.

مثال: `pd.read\_csv` برای خواندن فایل‌های CSV و تبدیل آن‌ها به DataFrame.

1. **seaborn:**

کاربرد: ایجاد نمودارهای آماری که زیبا و اطلاعاتی هستند، ساخته شده بر روی Matplotlib.

مثال: `sns.heatmap` برای رسم نقشه حرارتی.

1. **sklearn (scikit-learn):**

کاربرد: انجام وظایف یادگیری ماشین، ارائه ابزارهایی برای طبقه‌بندی، رگرسیون، خوشه‌بندی و بیشتر، همراه با امکاناتی برای پیش‌پردازش داده‌ها و ارزیابی مدل‌ها.

مثال: `RandomForestClassifier` برای ایجاد مدل‌های جنگل تصادفی.

1. **imblearn (imbalanced-learn):**

کاربرد: برای نمونه‌برداری بیش از حد، نمونه‌برداری کمتر و تولید داده‌های مصنوعی برای بهبود عملکرد مدل‌ها در داده‌های نامتوازن.

مثال: `SMOTE` برای نمونه‌برداری بیش از حد.

1. **scipy.stats:**

کاربرد: محاسبات علمی و فنی، ارائه دامنه وسیعی از توابع و ابزارها برای بهینه‌سازی، یکپارچه‌سازی، درون‌یابی، جبر خطی، آمار و بیشتر.

مثال: `randint` برای تولید اعداد تصادفی صحیح در یک محدوده.

1. **xgboost:**

کاربرد: ارائه الگوریتم‌های تقویت گرادیان (Gradient Boosting) که بهبود عملکرد و مقیاس‌پذیری دارند و به خصوص برای داده‌های ساختاریافته/جدولی مناسب هستند.

مثال: `XGBClassifier` برای ایجاد مدل‌های تقویت گرادیان.

1. **sklearn.metrics:**

کاربرد: ارزیابی عملکرد مدل‌های یادگیری ماشین، ارائه توابعی برای محاسبه معیارهایی مانند دقت، بازخوانی، F1-اسکور و بیشتر.

مثال: `accuracy\_score` برای محاسبه دقت مدل.

1. **sklearn.model\_selection:**

کاربرد: انتخاب مدل و تنظیم هایپرمتر، ارائه ابزارهایی برای تقسیم‌بندی داده‌ها به مجموعه‌های آموزش و تست، انجام اعتبارسنجی متقابل و جستجوی بهترین هایپرمترها.

مثال: `train\_test\_split` برای تقسیم داده‌ها به مجموعه‌های آموزش و تست.

1. **sklearn.neural\_network:**

کاربرد: ساخت و آموزش مدل‌های شبکه عصبی، ارائه رابطی ساده برای ایجاد پرسپترون‌های چند لایه (MLPs) برای وظایف رگرسیون و طبقه‌بندی.

مثال: `MLPClassifier` برای ایجاد مدل شبکه عصبی چند لایه.

1. **sklearn.preprocessing:**

کاربرد: وظایف پیش‌پردازش داده‌ها، ارائه توابع و کلاس‌هایی برای تبدیل و پیش‌پردازش داده‌های ورودی قبل از استفاده برای آموزش یا تست مدل‌های یادگیری ماشین.

مثال: `StandardScaler` برای استانداردسازی داده‌ها.

1. **sklearn.svm:**

کاربرد: برای الگوریتم‌های ماشین بردار پشتیبان (SVM) که برای وظایف طبقه‌بندی و رگرسیون استفاده می‌شوند.

مثال: `SVC` برای ایجاد مدل SVM.

1. **sklearn.tree:**

کاربرد: برای الگوریتم‌های مبتنی بر درخت تصمیم‌گیری، ارائه ابزارهایی برای ساخت، نمایش و استفاده از مدل‌های درخت تصمیم‌گیری برای وظایف طبقه‌بندی و رگرسیون.

مثال: `DecisionTreeClassifier` برای ایجاد مدل درخت تصمیم‌گیری.

1. **sklearn.ensemble:**

کاربرد: برای روش‌های یادگیری جمعی که مدل‌های فردی متعدد را ترکیب می‌کنند تا مدل پیش‌بینی قوی‌تری ایجاد شود.

مثال: `GradientBoostingClassifier` برای ایجاد مدل‌های تقویت گرادیان.

1. **sklearn.naive\_bayes:**

کاربرد: برای پیاده‌سازی الگوریتم‌های بیز ساده، یک خانواده از روش‌های طبقه‌بندی احتمالی بر اساس قضیه بیز.

مثال: `GaussianNB` برای ایجاد مدل بیز ساده.

1. **sklearn.neighbors:**

کاربرد: برای الگوریتم‌های مبتنی بر نزدیک‌ترین همسایه، ارائه ابزارهایی برای ایجاد مدل‌های مانند K-نزدیک‌ترین همسایه (KNN) برای وظایف طبقه‌بندی و رگرسیون.

مثال: `KNeighborsClassifier` برای ایجاد مدل KNN.

# **شروع پروژه**

در شروع کار، ما مراحل اولیه یک پروژه یادگیری ماشین را انجام می‌دهیم که شامل بارگذاری داده‌ها و بررسی ابعاد آن‌ها است. به طور کلی، کارهای زیر انجام می‌شود:

## **تعریف مسیر فایل**

ابتدا، نام پوشه و نام فایل که داده‌های ما در آن ذخیره شده‌اند، تعریف می‌شوند. این مقادیر به ترتیب "src" و "diabetes.csv" هستند.

## **ساخت مسیر کامل فایل**

با استفاده از تابع `os.path.join`، مسیر کامل فایل ایجاد می‌شود. این تابع پوشه و نام فایل را به صورت مناسب برای سیستم عامل ترکیب می‌کند.

## **باز کردن و خواندن فایل CSV**

فایل CSV با استفاده از دستور `with open` به صورت خواندنی باز می‌شود. سپس، محتویات فایل با استفاده از تابع `pd.read\_csv` از کتابخانه‌ی `pandas` خوانده می‌شود و به یک DataFrame تبدیل می‌گردد.

## **بررسی ابعاد داده‌ها**

در پایان، ابعاد داده‌های موجود در DataFrame با استفاده از `df.shape` چاپ می‌شود. این تابع تعداد ردیف‌ها و ستون‌های DataFrame را به صورت یک tuple برمی‌گرداند.

## **تحلیل و تفسیر کلی**

این قطعه کد یک مقدمه برای شروع پروژه یادگیری ماشین است. ما با بارگذاری داده‌ها از یک فایل CSV و تبدیل آن به یک ساختار داده‌ای مناسب (DataFrame) که توسط کتابخانه‌ی `pandas` پشتیبانی می‌شود، آغاز می‌کنیم. سپس، با بررسی ابعاد داده‌ها (تعداد ردیف‌ها و ستون‌ها)، اطلاعات اولیه‌ای درباره حجم داده‌هایی که با آن‌ها کار خواهیم کرد، به دست می‌آوریم. این مراحل ابتدایی، بخش مهمی از فرآیند پیش‌پردازش داده‌ها هستند که در پروژه‌های یادگیری ماشین ضروری‌اند.

## **اهمیت مراحل بالا**

* 1. بارگذاری داده‌ها: اولین گام در هر پروژه یادگیری ماشین، دسترسی به داده‌ها و بارگذاری آن‌ها به یک ساختار مناسب برای تحلیل و مدل‌سازی است.
  2. بررسی ابعاد داده‌ها: شناخت ابعاد داده‌ها به ما کمک می‌کند تا درک بهتری از حجم و ساختار داده‌ها داشته باشیم و برنامه‌ریزی مناسبی برای مراحل بعدی پیش‌پردازش و مدل‌سازی انجام دهیم.

**این مراحل اولیه به ما امکان می‌دهد که پروژه را با یک پایه محکم شروع کنیم و آماده مراحل پیشرفته‌تر مانند تحلیل داده‌ها، پیش‌پردازش، انتخاب مدل‌ها و ارزیابی آن‌ها شویم.**

## **معرفی داده‌های پروژه:**

|  |  |
| --- | --- |
| نام ویژگی | عنوان فارسی |
| Diabetes\_012 | دیابت (0: بدون دیابت، 1: دیابت نوع 1، 2: دیابت نوع 2) |
| HighBP | فشار خون بالا |
| HighChol | کلسترول بالا |
| CholCheck | بررسی کلسترول |
| BMI | شاخص توده بدنی (BMI) |
| Smoker | سیگار کشیدن |
| Stroke | سکته مغزی |
| HeartDiseaseorAttack | بیماری قلبی یا حمله قلبی |
| PhysActivity | فعالیت بدنی |
| Fruits | مصرف میوه |
| Veggies | مصرف سبزیجات |
| HvyAlcoholConsump | مصرف زیاد الکل |
| AnyHealthcare | دریافت هر نوع مراقبت‌های بهداشتی |
| NoDocbcCost | عدم مراجعه به پزشک به دلیل هزینه |
| GenHlth | سلامت عمومی |
| MentHlth | سلامت روان |
| PhysHlth | سلامت جسمانی |
| DiffWalk | مشکل در راه رفتن |
| Sex | جنسیت |
| Age | سن |
| Education | سطح تحصیلات |
| Income | سطح درآمد |

# **خواندن داده‌ها**

|  |
| --- |
| **شرح کد:** |
|  |
| این قطعه کد وظیفه خواندن یک فایل CSV از یک پوشه مشخص و سپس چاپ ابعاد داده‌های موجود در آن را دارد. ابتدا، نام پوشه و نام فایل به ترتیب به صورت `folder\_name` و `file\_name` تعریف می‌شوند که مقادیر آنها "src" و "diabetes.csv" هستند. سپس، با استفاده از تابع `os.path.join` از کتابخانه‌ی `os`، مسیر کامل فایل ساخته می‌شود و در متغیر `absolute\_path` ذخیره می‌گردد. در ادامه، با استفاده از دستور `with open`، فایل به حالت خواندنی باز می‌شود و با استفاده از کتابخانه‌ی `pandas` و تابع `pd.read\_csv`، محتویات فایل به یک DataFrame تبدیل می‌شود که در متغیر `df` ذخیره می‌گردد. خط کامنت شده‌ای که وجود دارد، در صورت کامنت نبودن، مسیر کامل فایل را چاپ می‌کرد. در پایان، ابعاد داده‌های موجود در DataFrame با استفاده از `df.shape` که تعداد ردیف‌ها و ستون‌های DataFrame را برمی‌گرداند، چاپ می‌شود. |
| **خروجی کد:** |
| Dataset shape:  (253680, 22) |

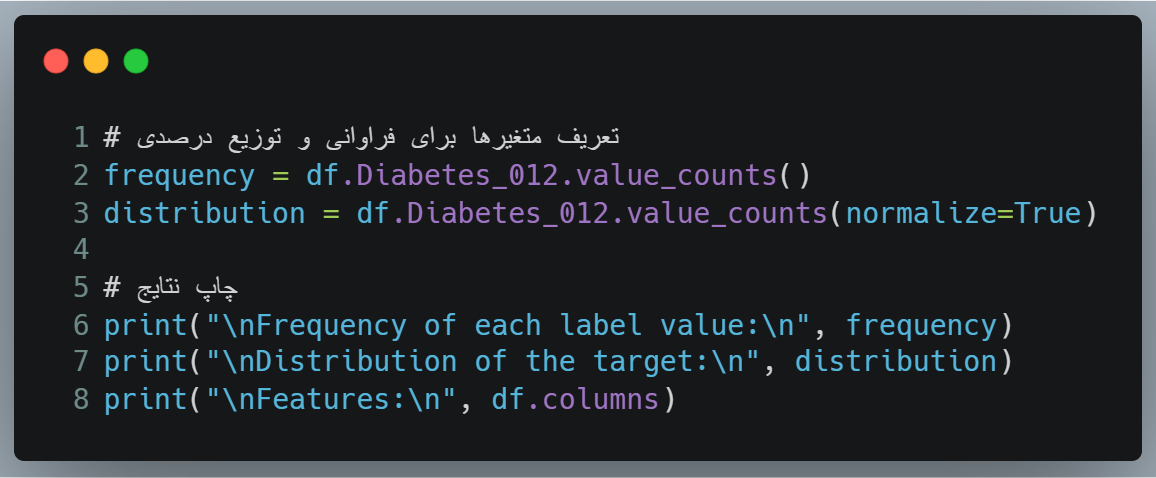
## **بررسی بانک داده‌ها**

|  |
| --- |
| **شرح کد:** |
|  |
| 1. خط 1: تعداد مقادیر گمشده (NaN) در هر ستون دیتاست محاسبه و در یک سری ذخیره می‌شود. 2. خط 2: درصد مقادیر گمشده برای هر ستون محاسبه می‌شود. این درصد با تقسیم تعداد مقادیر گمشده بر تعداد کل رکوردها و ضرب در 100 به دست می‌آید. 3. خط 6-9: یک DataFrame جدید به نام `missing\_summary` ایجاد می‌شود که شامل دو ستون است: یکی برای تعداد مقادیر گمشده و دیگری برای درصد مقادیر گمشده در هر ستون. 4. خط 10: DataFrame `missing\_summary` فیلتر می‌شود تا فقط ستون‌هایی که دارای مقادیر گمشده هستند نمایش داده شوند. 5. خط 11: DataFrame `missing\_summary` بر اساس درصد داده‌های گمشده به صورت نزولی مرتب می‌شود تا ستون‌هایی با بیشترین درصد مقادیر گمشده در بالا قرار گیرند. 6. خط 16: DataFrame `missing\_summary` که شامل خلاصه‌ای از داده‌های گمشده است، چاپ می‌شود. 7. خط 18: سری `missing\_data` که تعداد مقادیر گمشده در هر ستون را نشان می‌دهد، چاپ می‌شود. |
| **خروجی کد:** |
| Empty DataFrame  Columns: [Missing Values, Percentage]  --------------------------  Diabetes\_012 0  HighBP 0  HighChol 0  CholCheck 0  BMI 0  Smoker 0  Stroke 0  HeartDiseaseorAttack 0  PhysActivity 0  Fruits 0  Veggies 0  HvyAlcoholConsump 0  AnyHealthcare 0  NoDocbcCost 0  GenHlth 0  MentHlth 0  PhysHlth 0  DiffWalk 0  Sex 0  Age 0  Education 0  Income 0  dtype: int64 |
| **تحلیل خروجی کد:** |
| خروجی کد نشان می‌دهد که هیچ داده گمشده‌ای در دیتاست ما وجود ندارد. به عبارت دیگر، تمام مقادیر در ستون‌های دیتاست ما کامل هستند و هیچ سلولی خالی نیست.   1. DataFrame خالی:    1. بخش اول خروجی (`Empty DataFrame`) نشان می‌دهد که DataFrame `missing\_summary` خالی است. این اتفاق به این دلیل می‌افتد که ما فیلتر مقادیر بزرگتر از صفر را در خط ۱۲ کد خود اعمال کرده‌ایم و چون هیچ ستونی داده‌های گمشده ندارد، هیچ سطری در این DataFrame وجود نخواهد داشت. 2. تعداد داده‌های گمشده:    1. بخش دوم خروجی (لیست ستون‌ها با تعداد داده‌های گمشده برابر با 0 تأیید می‌کند که در هر ستون دیتاست، تعداد داده‌های گمشده صفر است. این به این معناست که تمام ستون‌ها به صورت کامل پر شده‌اند.   به طور خلاصه، دیتاست ما به خوبی تمیز و کامل است و هیچ نیازی به مدیریت داده‌های گمشده نیست. |

مرحله چهارم

نحوه توزیع متغیر هدف

در این قسمت توزیع متغیر هدف خود را بررسی می کنیم. یعنی توزیع افراد سالم، افراد پیش دیابت و افراد دیابتی چگونه است. سپس این افراد را در محدوده 0 تا 1 نرمالایز می کنم.



1. خط 2-3:
   1. مقدار فراوانی (تعداد) هر برچسب (0، 1، 2) در ستون ` Diabetes\_012` محاسبه و در متغیری به نام `frequency` ذخیره می‌شود.
   2. توزیع درصدی هر برچسب (نسبت هر مقدار به کل داده‌ها) نیز محاسبه و در متغیری به نام `distribution` ذخیره می‌شود.
2. خط 6:
   1. چاپ یک متن توضیحی که به کاربر اطلاع می‌دهد که فراوانی هر برچسب در ستون ` Diabetes\_012` قرار است نمایش داده شود.
   2. سپس، متغیر `frequency` که حاوی تعداد تکرار هر برچسب است، چاپ می‌شود.
3. خط 7:
   1. چاپ یک متن توضیحی که به کاربر اطلاع می‌دهد که توزیع درصدی مقادیر مختلف در ستون `Diabetes\_012` قرار است نمایش داده شود.
   2. سپس، متغیر `distribution` که حاوی درصد هر برچسب نسبت به کل رکوردها است، چاپ می‌شود.
4. خط 8:
   1. چاپ یک متن توضیحی که به کاربر اطلاع می‌دهد که لیست کامل ستون‌های دیتاست نمایش داده خواهد شد.
   2. سپس، نام تمام ستون‌های دیتاست با استفاده از `df.columns` چاپ می‌شود.

به طور خلاصه: این کد ابتدا فراوانی و توزیع درصدی مقادیر ستون `Diabetes\_012` را محاسبه کرده و سپس آن‌ها را همراه با نام تمام ستون‌های دیتاست چاپ می‌کند. این نسخه کد، به دلیل ذخیره نتایج محاسبات در متغیرهای موقت (`frequency` و `distribution`) از تکرار محاسبات جلوگیری می‌کند و کد را خواناتر می‌سازد.

**شرح خروجی کد:**

این خروجی اطلاعات کلیدی در مورد توزیع مقادیر ستون `Diabetes\_012` و ویژگی‌های دیتاست ما ارائه می‌دهد:

1. **فراوانی مقادیر (`Frequency of each label value`):**

تعداد رکوردهای مربوط به هر مقدار در ستون `Diabetes\_012` نشان داده شده است:

**0.0:** 213,703 رکورد (بدون دیابت)

**2.0:** 35,346 رکورد (دیابت نوع 2)

**1.0:** 4,631 رکورد (دیابت نوع 1)

این آمار نشان می‌دهد که بیشتر داده‌ها مربوط به افرادی است که دیابت ندارند (0.0).

1. **توزیع درصدی مقادیر (`Distribution of the target`):**

نسبت هر مقدار نسبت به کل رکوردها نمایش داده شده است:

**0.0:** 84.24% از کل داده‌ها (بدون دیابت)

**2.0:** 13.93% از کل داده‌ها (دیابت نوع 2)

**1.0:** 1.83% از کل داده‌ها (دیابت نوع 1)

این آمار نشان می‌دهد که درصد بسیار بالایی از داده‌ها به افراد بدون دیابت تعلق دارد و درصد کمی از افراد به دیابت نوع 1 مبتلا هستند.

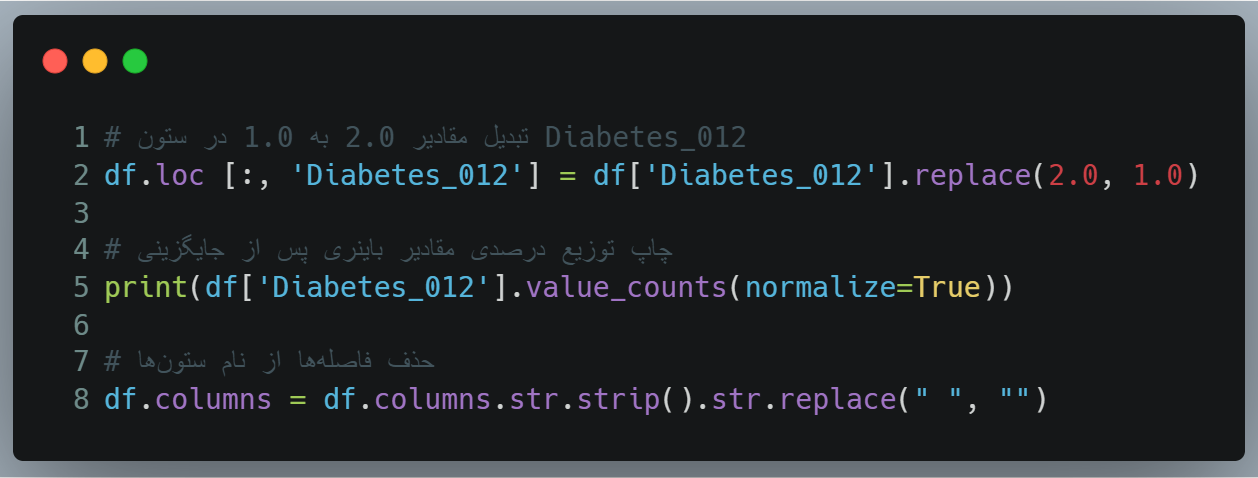
1. **ویژگی‌ها (`Features`):**

لیستی از نام تمام ستون‌های دیتاست را نمایش می‌دهد. این ویژگی‌ها شامل متغیرهای مختلفی مانند فشار خون بالا (`HighBP`)، شاخص توده بدنی (`BMI`)، سیگار کشیدن (`Smoker`)، فعالیت بدنی (`PhysActivity`) و غیره است.

**نتیجه‌گیری کلی:**

این خروجی نشان می‌دهد که در دیتاست ما، بیشتر رکوردها به افراد بدون دیابت تعلق دارند. همچنین، این خروجی اطلاعاتی در مورد توزیع مقادیر دیابت و نام ویژگی‌های مختلف در دیتاست ارائه می‌دهد که برای تحلیل‌های بیشتر مفید است.

مرحله پنجم

تبدیل داده‌ها به نوع باینری

1. خط 2:

ستون `Diabetes\_012` را انتخاب می‌کند و تمام مقادیر `2.0` را با `1.0` جایگزین می‌کند. این عملیات ستون را به یک متغیر باینری تبدیل می‌کند که شامل دو مقدار `0.0` (عدم وجود دیابت) و `1.0` (وجود دیابت، شامل هر دو نوع 1 و 2) می‌شود.

1. خط 5:

توزیع درصدی مقادیر جدید (باینری) در ستون `Diabetes\_012` را محاسبه و چاپ می‌کند. با استفاده از `normalize=True`، نسبت هر مقدار به کل داده‌ها نمایش داده می‌شود. این خروجی به شما نشان می‌دهد که چند درصد از داده‌ها به هر یک از مقادیر `0.0` و `1.0` تعلق دارند.

1. خط 8:

نام تمام ستون‌های دیتاست را بررسی می‌کند و هرگونه فاصله‌ی اضافی در ابتدا و انتهای نام ستون‌ها را حذف می‌کند (با استفاده از `str.strip()`).

سپس، تمام فاصله‌های داخلی در نام ستون‌ها را با رشته‌ی خالی جایگزین می‌کند (با استفاده از `str.replace(" ", "")`). این کار نام ستون‌ها را تمیز و منظم می‌کند و ممکن است در زمان استفاده از کدهای دیگر مفید باشد.

**شرح خروجی کد:**

این خروجی نشان می‌دهد که پس از تبدیل مقادیر `2.0` به `1.0` در ستون `Diabetes\_012`، داده‌های ما اکنون به دو کلاس باینری تقسیم شده‌اند:

|  |  |
| --- | --- |
| **0.0** | 84.24% از داده‌ها به افرادی اختصاص دارد که دیابت ندارند. |
| **1.0** | 15.76% از داده‌ها به افرادی اختصاص دارد که دیابت دارند (شامل هر دو نوع 1 و 2). |

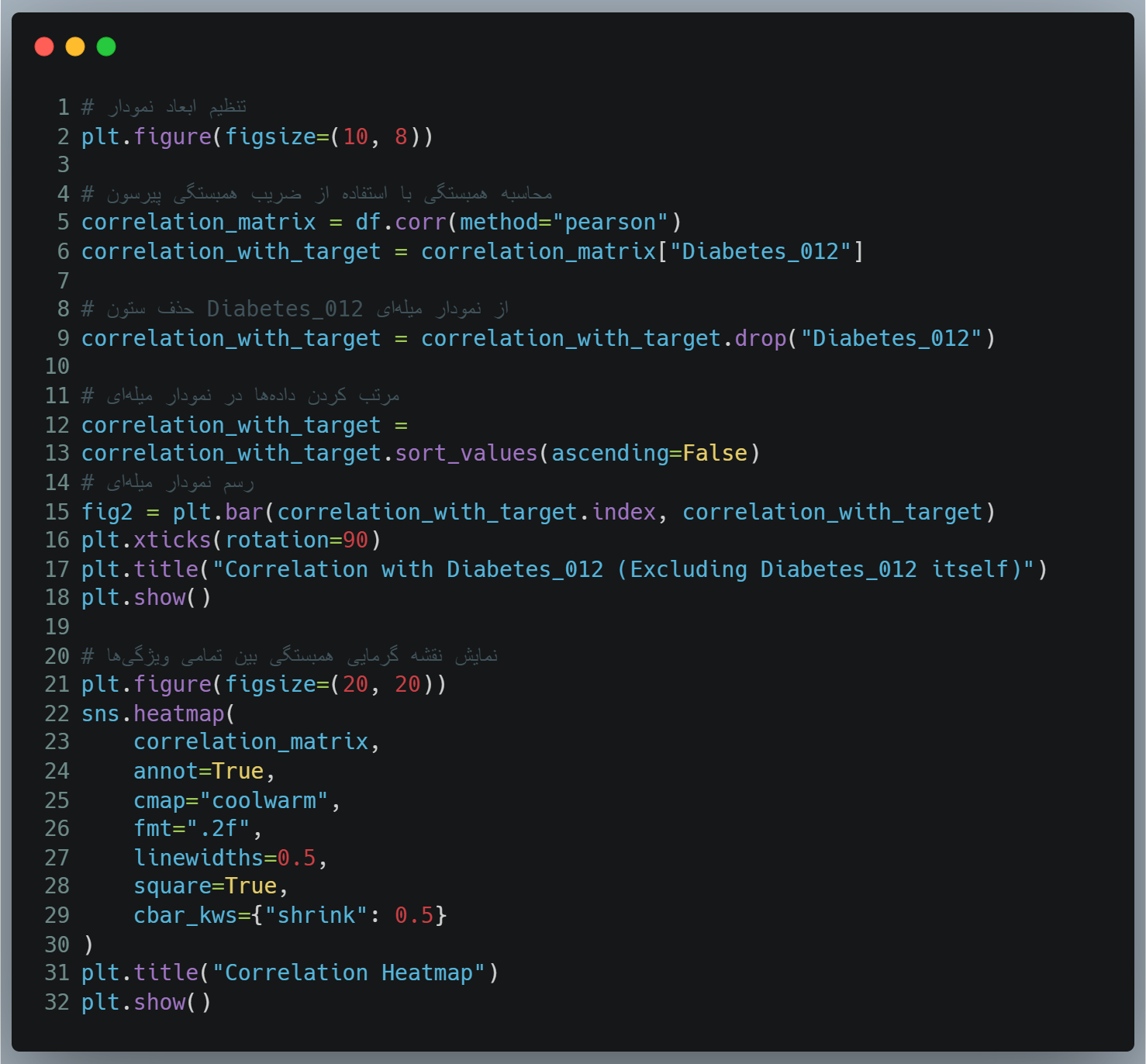
تحلیل مختصر:

به طور کلی اکثر داده‌ها (بیش از 84%) به افرادی مربوط می‌شود که دیابت ندارند. تنها حدود 16% از داده‌ها به افرادی تعلق دارد که دیابت دارند، که این نسبت پس از ادغام داده‌های مربوط به دیابت نوع 1 و 2 به دست آمده است.

این توزیع می‌تواند نشان‌دهنده‌ی عدم توازن (class imbalance) در داده‌ها باشد، که در مراحل بعدی تحلیل یا مدلسازی می‌تواند تاثیرگذار باشد. به عنوان مثال، اگر قصد داشته باشیم یک مدل یادگیری ماشین بسازیم، باید به این عدم توازن توجه داشته باشیم و شاید نیاز به روش‌های خاصی مانند **oversampling** یا **undersampling** برای مدیریت این مسئله داشته باشیم.

**مرحله ششم**

**بررسی همبستگی داده‌ها**



1. **خط 2:**

ابعاد نمودار را به 10 در 8 اینچ تنظیم می‌کند تا فضای کافی برای نمایش داده‌ها فراهم شود.

1. **خط 5:**

ماتریس همبستگی بین تمام ویژگی‌ها را با استفاده از ضریب همبستگی پیرسون محاسبه می‌کند و در متغیر `correlation\_matrix` ذخیره می‌کند.

1. **خط 6:**

همبستگی ستون `Diabetes\_012` با سایر ویژگی‌ها را از`correlation\_matrix` استخراج می‌کند و در متغیر `correlation\_with\_target` ذخیره می‌کند.

1. **خط 8:**

ستون `Diabetes\_012` را از داده‌های `correlation\_with\_target` حذف می‌کند تا فقط همبستگی سایر ویژگی‌ها با `Diabetes\_012` نمایش داده شود.

1. **خط 11:**

یک نمودار میله‌ای (bar plot) برای نمایش همبستگی‌های باقی‌مانده رسم می‌کند. محور افقی نشان‌دهنده نام ویژگی‌ها و محور عمودی مقدار همبستگی است.

1. **خط 12:**

برچسب‌های محور افقی (نام ویژگی‌ها) را با چرخاندن آن‌ها به 90 درجه تنظیم می‌کند تا بهتر قابل مشاهده باشند.

1. **خط 13:**

عنوان نمودار میله‌ای را به "Correlation with Diabetes\_012 (Excluding Diabetes\_012 itself)" تغییر می‌دهد تا مشخص کند که این نمودار همبستگی سایر ویژگی‌ها با `Diabetes\_012` را بدون خود `Diabetes\_012` نمایش می‌دهد.

1. **خط 14:**

نمودار میله‌ای را به نمایش می‌گذارد تا کاربر بتواند آن را مشاهده کند.

1. **خط 16:**

ابعاد دومین نمودار را به 20 در 20 اینچ تنظیم می‌کند تا فضای کافی برای نمایش نقشه گرمایی (heatmap) بزرگ فراهم شود.

1. **خط 17:**

نقشه گرمایی (heatmap) همبستگی بین تمامی ویژگی‌ها را با استفاده از `sns.heatmap` رسم می‌کند. این نقشه گرمایی شامل ویژگی‌های زیر است:

|  |  |
| --- | --- |
| مقادیر همبستگی در هر سلول نقشه گرمایی نمایش داده می‌شود. | annot=True |
| نقشه رنگی برای نقشه گرمایی که طیف رنگی از سرد به گرم را نمایش می‌دهد. | cmap="coolwarm |
| مقادیر همبستگی با دو رقم اعشار نمایش داده می‌شود. | fmt=".2f |
| عرض خطوط شبکه در نقشه گرمایی برای وضوح بیشتر تنظیم شده است. | linewidths=0.5 |
| سلول‌های نقشه گرمایی به‌صورت مربع نمایش داده می‌شوند. | square=True |
| نوار رنگی (color bar) کوچکتر نمایش داده می‌شود تا نقشه گرمایی جمع‌وجورتر باشد. | cbar\_kws={"shrink": 0.5} |

1. **خط 18:**

عنوان نقشه گرمایی را به "Correlation Heatmap" تغییر می‌دهد تا نشان دهد که نقشه همبستگی بین تمام ویژگی‌ها را نمایش می‌دهد.

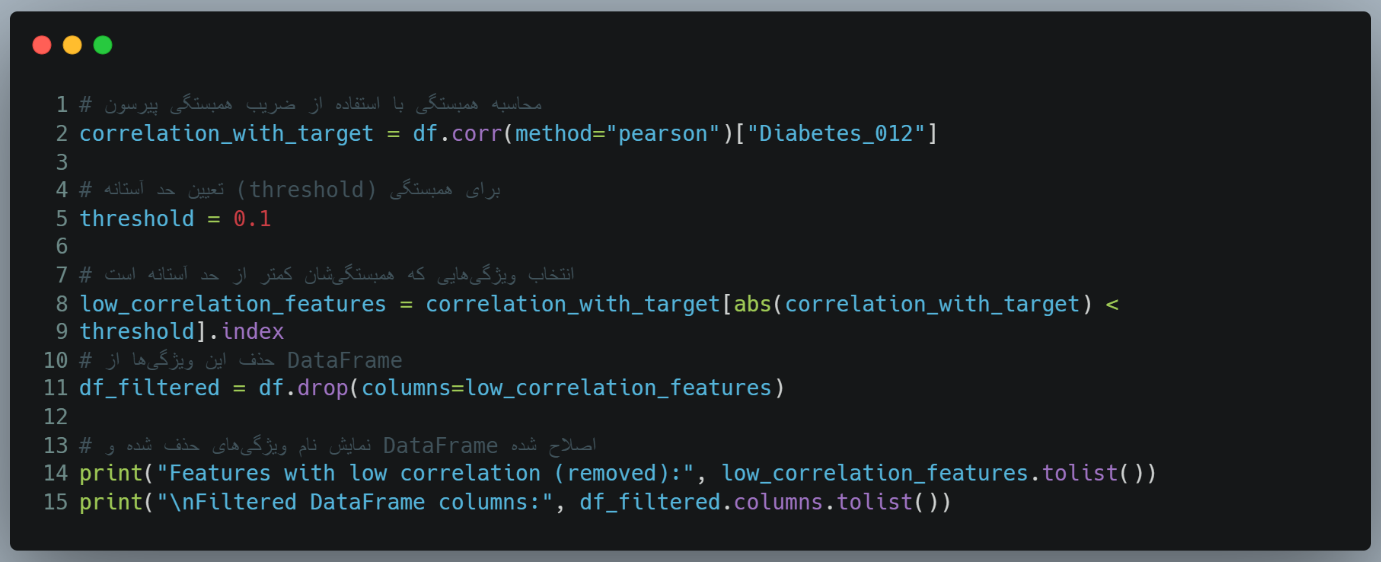
1. **خط 19:**

نقشه گرمایی را به نمایش می‌گذارد تا کاربر بتواند آن را مشاهده کند.

**خلاصه عملکرد:**

کد ما ابتدا یک نمودار میله‌ای برای نمایش همبستگی هر ویژگی با ستون `Diabetes\_012`ایجاد می‌کند، بدون نمایش ستون `Diabetes\_012`در نمودار. سپس، نقشه گرمایی از همبستگی بین تمام ویژگی‌ها را رسم می‌کند. این مراحل به ما کمک می‌کند تا همبستگی ویژگی‌های مختلف را به‌طور بصری و واضح مشاهده کنیم و تحلیل بهتری از داده‌ها ارائه دهیم.

مرحله هفتم

حذف داده‌های با ضریب همبستگی کمتر

استاندارد سازی داده‌ها

استانداردسازی داده‌ها (Data Standardization) یکی از تکنیک‌های پیش‌پردازش داده‌ها است که در آن مقیاس داده‌ها تغییر می‌کند تا میانگین داده‌ها صفر و انحراف معیار آنها یک شود. این روش معمولاً در تحلیل داده‌ها و یادگیری ماشین استفاده می‌شود تا داده‌ها به یک مقیاس مشترک تبدیل شوند.

### چرا استانداردسازی داده‌ها اهمیت دارد؟

1. همگرایی سریع‌تر الگوریتم‌ها:

- بسیاری از الگوریتم‌های یادگیری ماشین، مانند رگرسیون لجستیک، شبکه‌های عصبی و ماشین‌های بردار پشتیبان (SVM)، به این حساس هستند که ویژگی‌ها در مقیاس‌های مختلفی قرار دارند. اگر داده‌ها در مقیاس‌های متفاوت باشند، این الگوریتم‌ها ممکن است به درستی عمل نکنند یا زمان بیشتری برای همگرایی نیاز داشته باشند. استانداردسازی باعث می‌شود این الگوریتم‌ها بهینه‌تر و سریع‌تر به جواب برسند.

2. بهبود دقت مدل‌ها:

- برخی الگوریتم‌ها مانند K-نزدیک‌ترین همسایه (KNN) یا الگوریتم‌های مبتنی بر فاصله، تحت تأثیر مقیاس ویژگی‌ها قرار دارند. در این الگوریتم‌ها، ویژگی‌هایی که مقیاس بزرگتری دارند، ممکن است بر نتایج تأثیر بیشتری داشته باشند. استانداردسازی باعث می‌شود تمامی ویژگی‌ها به یک اندازه در مدل تأثیرگذار باشند.

3. تسهیل مقایسه ویژگی‌ها:

- استانداردسازی داده‌ها به شما این امکان را می‌دهد که ویژگی‌های مختلف را با یکدیگر مقایسه کنید، زیرا تمام ویژگی‌ها به یک مقیاس یکسان تبدیل شده‌اند.

### چگونه استانداردسازی انجام می‌شود؟

استانداردسازی معمولاً با استفاده از فرمول زیر انجام می‌شود:

\[ z = \frac{x - \mu}{\sigma} \]

- \( z \): مقدار استانداردشده

- \( x \): مقدار اصلی داده

- \( \mu \): میانگین مقادیر ویژگی

- \( \sigma \): انحراف معیار ویژگی

### مثال:

فرض کنید یک ویژگی به نام `"BMI"` داریم که دارای میانگین 25 و انحراف معیار 5 است. اگر مقداری در این ستون 30 باشد، پس از استانداردسازی، مقدار آن به صورت زیر محاسبه می‌شود:

\[ z = \frac{30 - 25}{5} = 1 \]

بنابراین مقدار استانداردشده این ویژگی برابر با 1 خواهد بود.

### ابزارهای استانداردسازی در پایتون:

در پایتون، می‌توانید از کتابخانه‌های مختلفی مانند `scikit-learn` برای انجام استانداردسازی استفاده کنید. یک ابزار معمول در این کتابخانه کلاس `StandardScaler` است که به شما اجازه می‌دهد داده‌ها را به سادگی استانداردسازی کنید.

### نکته مهم:

استانداردسازی زمانی مفید است که ویژگی‌ها در مقیاس‌های متفاوت قرار دارند و این تفاوت می‌تواند بر دقت مدل‌های یادگیری ماشین تأثیر بگذارد. با این حال، در برخی موارد، استانداردسازی ممکن است ضروری نباشد، مثلاً در الگوریتم‌های درخت تصمیم که به مقیاس داده‌ها حساس نیستند.

### خلاصه:

استانداردسازی داده‌ها یک تکنیک پیش‌پردازش است که داده‌ها را به یک مقیاس مشترک تبدیل می‌کند. این کار باعث بهبود عملکرد و دقت مدل‌های یادگیری ماشین می‌شود و به مقایسه منصفانه ویژگی‌ها کمک می‌کند.