دانشگاه علم و صنعت ایران

دانشکده مهندسی کامپیوتر

عنوان: پروژه اول درس داده‌کاوی
پیش‌بینی بیماری با مدل رگرسیون

نام و نام خانوادگی: آیلین نائب‌زاده

شماره دانشجویی: ۹۹۵۲۲۱۸۵

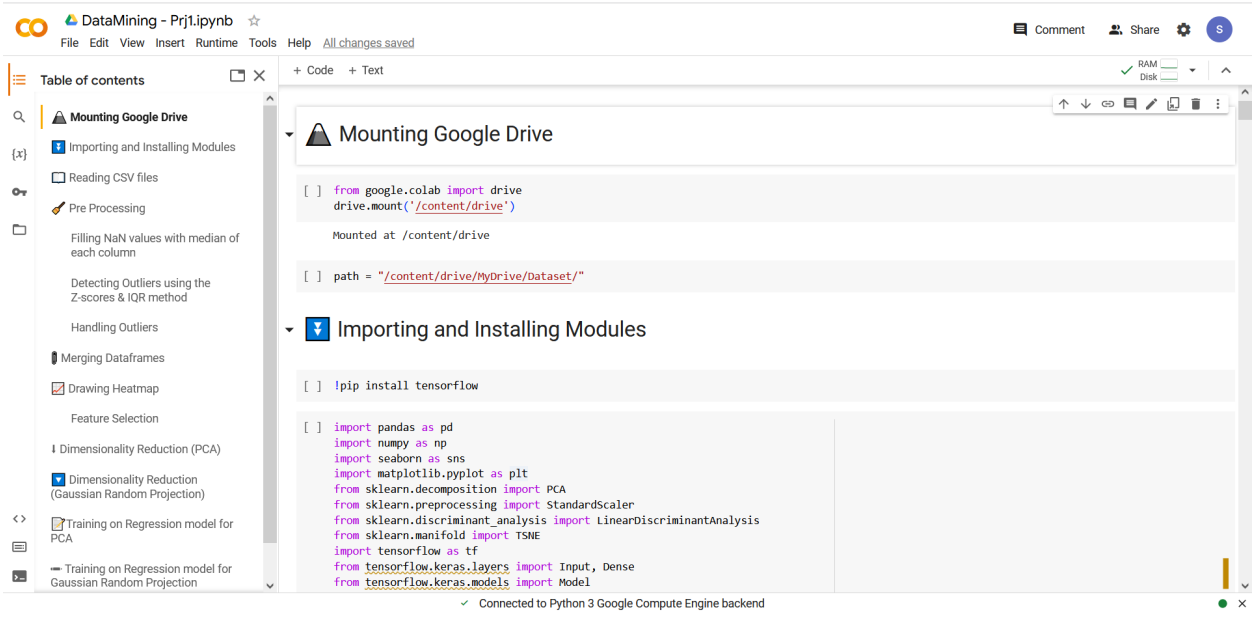
نیم‌سال تحصیلی: پائیز ۱۴۰۲

مدرّس: دکتر حسین رحمانی

فهرست مطالب

۱ گام اول

در طی این پروژه، باتوجه به اینکه داده‌های ما در پنج فایل مجزا قرار گرفته‌اند، در ابتدا نیاز است که جداگانه سعی کنیم محتوای هر یک از فایل‌ها را بشناسیم. در اولین قدم با مراجعه به سایت Google Colab و ساخت یک پروژه جدید و فراخوانی بعضی از کتابخانه‌های معروف زبان Python شروع به مشاهده داده‌های موجود در هر فایل و انجام تحلیل‌های ابتدایی می‌کنیم. لیست برخی کتابخانه‌های استفاده‌شده به‌همراه محیط کلی پروژه را در تصویر زیر می‌توانید مشاهده کنید.



شکل ۱: نمای کلی از پروژه notebook در فضای Google Colab

برای راحتی کار و سرعت بیشتر، فایل‌های موجود در پوشه Dataset را در Google Drive می‌توانیم آپلود کنیم و سپس بااستفاده از دستور google.mount و تعریف مسیر درست، به‌راحتی به فایل‌ها دسترسی داشته باشیم. در ادامه برای خواندن فایل‌های csv و excel و تبدیل آن‌ها به شیءهایی از نوع dataframe از توابع آماده و از پیش تعریف شده read_csv و read_excel که در کتابخانه pandas موجود هستند، استفاده می‌کنیم.

همچنین در حین خوانش هر یک از فایل‌ها، می‌توانیم مواردی مانند نام ستون‌های موجود در هر فایل، تعداد مقادیر صفر در هر ستون، تعداد مقادیر تعریف نشده(NaN) و درصد این‌گونه موارد به کل را حساب کنیم. بطور مثال گونه‌ای از این تحلیل‌ها را برای فایل demographic.csv در تکه کد زیر می‌توانید مشاهده کنید.

```
df_demographic = pd.read_csv(path + 'demographic.csv')

for column in df_demographic.columns:
  total_count = len(df_demographic[column])
  zero_count = (df_demographic[column] == 0).sum()
  nan_count = (df_demographic[column].isna()).sum()
  nan_percentage = (nan_count / total_count) * 100
  print(f"Column '{column}': {nan_count} cells with a value of NaN")
  print(f"Column '{column}': {zero_count} cells with a value of 0")
  print(f"Column '{column}': {nan_percentage:.2f}% of cells with a value of
      NaN")
  print("---------------------------------------")

df_demographic.head(n=10)
```

۲ گام دوم

در مرحله پیش پردازش داده از روش‌های مختلفی می‌توانیم بهره ببریم.

در ابتدا برای هر یک از فایل‌ها و ستون‌های موجود در آن‌ها می‌توانیم باتعریف یک حد آستانه مشخص کنیم که ستون‌هایی که تعداد مقادیر NaN در آن‌ها از آن حد بیشتر باشد، حذف شوند. سپس طبق توضیحات موجود در فایل پروژه به‌دنبال مقادیر ترکیب از ۷ و ۹ می‌گردیم و آن‌ها را با NaN جایگزین می‌کنیم و مجددا مرحله قبل را تکرار می‌کنیم. در ادامه باتوجه به اینکه مقادیر SEQN نقشی همانند Primary Key در فایل‌های ما دارند، چک می‌کنیم که در یک فایل، دو یا بیش از دو SEQN یکسان وجود نداشته باشد و درصورت وجود نیز تنها اولین نمونه آن را نگه می‌داریم.

در مرحله بعدی مقادیر NaN باقی مانده را، با median همان ستون خاص جایگزین می‌کنیم، چراکه این مقدار نسبت به Outlierها نیز حساس نمی‌باشد.

در نهایت برای شناسایی مقادیر Outlier در هر فایل و در هر ستون آن‌ها، از روش IQR استفاده می‌کنیم. البته روش‌های دیگری نیز همانند Z-Score وجود دارند ولی همانطور که در فایل پروژه می‌توانید مشاهده کنید، روش IQR تعداد Outlierهای بیشتری را شناسایی کرده‌است.

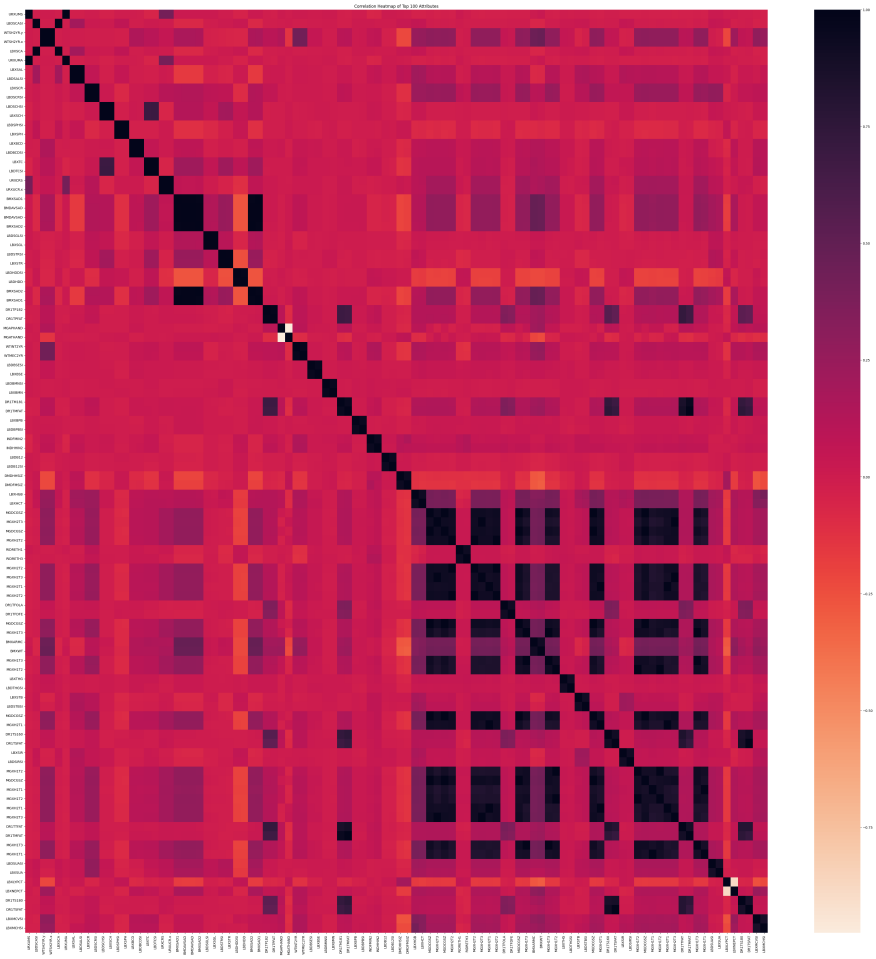
درنهایت پس از شناسایی Outlierها، آن‌ها را نیز با میانه موجود در هر ستون جایگزین می‌کنیم. یکی از چالش‌ها و نکاتی که در این مرحله متوجه شدم، وجود مقادیری همچون P، S و ... بود که به ما اجازه تشخیص مقادیر Outlier را نمی‌دادند. و در نتیجه نیاز بود تا در ابتدا مقادیر از نوع str را نیز به حالت عددی در بیاوریم و به هر یک از آن مقادیر یک کد منحصر به‌فرد اختصاص دهیم.

۳ گام سوم

در گام سوم و برای تشخیص ویژگی‌های مهم‌تر در ابتدا تمامی فایل‌ها بجز فایل questionnaire.csv را با یکدیگر مرج میزنیم. همچنین باید دقت کنیم که تمامی این عملیات براساس ویژگی SEQN صورت می‌گیرد. سپس بااستفاده از تابع ()corr. موجود در کتابخانه pandas همبستگی بین دوبه‌دوی ویژگی‌ها را بدست می‌آوریم. سپس بااستفاده از توابعی مانند abs. و sort. سعی می‌کنیم صد جفت ویژگی که با هم بیشترین همبستگی را براساس اندازه مقدار Correlation Coefficient دارند را برگردانیم.

البته به این نکته هم دقت می‌کنیم که جفت ویژگی‌هایی که بین یک ویژگی و خودش هست را در نظر نگیریم. همچنین برای نمایش میزان همبستگی بین ویژگی‌ها و درک روابط بین صد ویژگی استخراج شده، از توابع موجود در کتابخانه matplotlib و seaborn استفاده می‌کنیم.

تصویر مربوط به نمودار نقشه حرارتی صد ویژگی استخراج شده را در تصویر زیر می‌توانید مشاهده کنید.



شکل ۲: نمودار نقشه حرارتی صد ویژگی با بیشترین همبستگی

در نهایت برای انتخاب تنها یک ویژگی از میان ویژگی‌هایی که بیشترین همبستگی را با یکدیگر دارند، با تعریف یک حد آستانه(در این پروژه ۰.۸) ویژگی‌هایی که بیشترین ارتباط را با یکدیگر دارند در ابتدا در یک مجموعه ذخیره می‌کنیم و سپس اولین عضو آن‌ها را انتخاب می‌کنیم.

۴ گام چهارم

در این مرحله باتوجه به آرایه selected_features که تعریف کرده بودیم و توابع آماده موجود در کتابخانه sklearn همانند PCA سعی می‌کنیم تا ابعاد داده‌های خود را کاهش دهیم. همچنین پیش از استفاده از ماژول PCA در ابتدا سعی می‌کنیم با استفاده از ماژول StandardScalar داده‌های خود را نرملایز کنیم. همچنین می‌توانیم با تغییر مقدار n_components تعداد کامپوننت‌هایی که می‌خواهیم داده‌های خود را به آن‌ها تقسیم کنیم، مشخص کنیم.

روش‌های بسیار دیگری برای کاهش ابعاد داده‌ها وجود دارند. باتوجه به نوع داده‌های مسئله، از ماژول GaussianRandomProjection به‌عنوان روش دوم برای کاهش ابعاد استفاده کردم.

۵ گام پنجم

در مرحله پنجم باتوجه به خواسته مسئله که استفاده از خروجی‌های مرحله پیشین و آموزش مدل‌های Regression بااستفاده از آن‌ها می‌باشد، می‌توانیم از مدل‌های Regression موجود در کتابخانه sklearn استفاده کنیم. در این بخش سه مدل LinearRegression، KNeighborsRegressor و DecisionTreeRegressor را برای داده‌های کاهش یافته از هر دو حالت PCA و Gaussian Random Projection در نظر می‌گیریم.

همچنین پیش از استفاده از این دو مدل، نیاز است که دیتاست مقادیر ورودی(X) و لیبل(y) خود را تعریف کنیم. دیتاست مقادیر ورودی درواقع داده‌های کاهش یافته می‌باشند و دیتاست لیبل‌ها نیز یک بار آرایه‌ای از مقادیر ابتلا/عدم ابتلا به سرطان و بار دیگر آرایه‌ای از ابتلا/عدم ابتلا به مشکلات کبدی می‌باشد.

همچنین باتوجه به اینکه تعداد ردیف‌های دو دیتاست داده‌های ورودی و مقادیر خروجی(لیبل‌ها) نیز باید یکسان باشد، باید تنها ردیف‌هایی از دیتاست لیبل‌ها را انتخاب کنیم که SEQN آن‌ها در dataframe تجمیع شده‌ای که بااستفاده از آن داده‌های کاهش یافته را ساختیم، موجود باشد.

سپس بااستفاده از تابع train_test_split داده‌های ورودی خود را به یک نسبت دلخواه به دیتاست تست و آموزشی تجزیه می‌کنیم.

در نهایت بااستفاده از دو تابع fit. و predict. مدل‌های خود را آموزش می‌دهیم و از آن‌ها برای پیش‌بینی مقادیر دیتاست تست استفاده می‌کنیم.

بطور مثال در صورتی که از داده‌های خروجی الگوریتم PCA و لیبل‌های سرطانی بخواهیم استفاده کنیم کد پیاده‌سازی شده بصورت زیر می‌شود.

```
1  X_train, X_test, y_train_MCQ220, y_test_MCQ220 = train_test_split(X_list,
       MCQ220_elements, test_size=0.3, random_state=42)
2
3  # Initialize regression models
4  linear_reg_pca = LinearRegression()
5  knn_reg_pca = KNeighborsRegressor()
6  decision_tree_reg_pca = DecisionTreeRegressor()
7
8  # Train the models
9  linear_reg_pca.fit(X_train, y_train_MCQ220)
10 knn_reg_pca.fit(X_train, y_train_MCQ220)
11 decision_tree_reg_pca.fit(X_train, y_train_MCQ220)
12
13 # Make predictions on the testing data
14 linear_reg_pred_pca_cancer = linear_reg_pca.predict(X_test)
15 knn_reg_pred_pca_cancer = knn_reg_pca.predict(X_test)
16 decision_tree_reg_pred_pca_cancer = decision_tree_reg_pca.predict(X_test)
```

۶ گام ششم

در گام آخر می‌توانیم با معیارهایی همانند Mean Squared Error و Mean Absolute Error مدل خود را ارزیابی کنیم. نتایج مربوط به این ارزیابی را در چهار تصویر زیر می‌توانید مشاهده کنید.

```
---------------------- Using PCA Reduced Data ----------------------
Linear Regression MSE for Cancer Detection:  0.2345489610866928
-----------------------------
KNN Regression MSE for Cancer Detection:  0.2520567568934667
-----------------------------
Decision Tree Regression MSE for Cancer Detection:  0.32502090233786773
==============================================
Linear Regression MAE for Cancer Detection:  0.10362948193953896
-----------------------------
KNN Regression MAE for Cancer Detection:  0.10203804347826086
-----------------------------
Decision Tree Regression MAE for Cancer Detection:  0.10563858695652174
```

شکل ۳: خروجی میزان ارور(خطا) در حالت استفاده از الگوریتم PCA و برای داده‌های سرطانی

```
---------------------- Using PCA Reduced Data ----------------------
Linear Regression MSE for Liver Detection: 0.15937871445367988
-----------------------------
KNN Regression MSE for Liver Detection: 0.16963836215032105
-----------------------------
Decision Tree Regression MSE for Liver Detection: 0.20020370061297676
==============================================
Linear Regression MAE for Liver Detection:  0.04635734643033035
-----------------------------
KNN Regression MAE for Liver Detection:  0.0452445652173913
-----------------------------
Decision Tree Regression MAE for Liver Detection:  0.04008152173913043
```

شکل ۴: خروجی میزان ارور(خطا) در حالت استفاده از الگوریتم PCA و برای داده‌های کبدی

```
---------- Using Gaussian Random Projection Reduced Data ----------
Linear Regression MSE for Cancer Detection:  0.2337204615909028
-----------------------------
KNN Regression MSE for Cancer Detection:  0.25297147115315455
-----------------------------
Decision Tree Regression MSE for Cancer Detection:  0.3202835021338602
==========================================
Linear Regression MAE for Cancer Detection:  0.10213023048001295
-----------------------------
KNN Regression MAE for Cancer Detection:  0.10285326086956523
-----------------------------
Decision Tree Regression MAE for Cancer Detection:  0.10258152173913043
```

شکل ۵: خروجی میزان ارور(خطا) در حالت استفاده از الگوریتم GRP و برای داده‌های سرطانی

```
---------- Using Gaussian Random Projection Reduced Data ----------
Linear Regression MSE for Liver Detection:  0.2337204615909028
-----------------------------
KNN Regression MSE for Liver Detection:  0.25297147115315455
-----------------------------
Decision Tree Regression MSE for Liver Detection:  0.32134229345857573
==========================================
Linear Regression MAE for Liver Detection:  0.10213023048001295
-----------------------------
KNN Regression MAE for Liver Detection:  0.10285326086956523
-----------------------------
Decision Tree Regression MAE for Liver Detection:  0.10326086956521739
```

شکل ۶: خروجی میزان ارور(خطا) در حالت استفاده از الگوریتم GRP و برای داده‌های کبدی

۷ نتیجه‌گیری

۱.۷ نتیجه ارزیابی‌ها با روش **PCA** و روش انتخابی خود

در حالتی که لیبل‌های سرطانی را بررسی کرده باشیم، میزان ارور برگردانده‌شده در دو حالت استفاده از الگوریتم PCA و GRP بسیار شبیه به هم خواهد شد. ولی در صورتی که بخواهیم لیبل‌های مشکلات کبدی را بررسی کنیم، میزان ارور یا خطای برگردانده‌شده بااستفاده از الگوریتم PCA به نسبت کمتر از حالتی خواهد شد که از الگوریتم GRP استفاده کرده باشیم.

۲.۷ تاثیرات انتخاب ویژگی‌ها - مزایا و معایب

بله قطعا باتوجه به حجم بالای داده و تعداد بسیار زیاد ویژگی‌ها که به بیش از ۵۰۰ می‌رسید، فیلتر ویژگی‌ها و انتخاب بعضی از آن‌ها سرعت اجرای بعضی از الگوریتم‌ها را افزایش می‌دهد. ولی باید به این نکته نیز توجه داشته باشیم که همواره حذف بعضی از ویژگی‌ها می‌تواند به معنی از دست رفتن اطلاعات نیز باشد. همچنین در صورت وجود تعداد بالای ویژگی‌ها تحلیل مدل‌ها و تفسیرپذیری آن‌ها نیز کاهش می‌یابد.

۳.۷ مقایسه ویژگی‌های سرعت و دقت در دو مدل مختلف

بطور کلی می‌توانیم بگوئیم PCA معمولا صحت بیشتری دارد ولی به مدت زمان بیشتری برای اجرا احتیاج دارد. درصورتی که الگوریتم GRP صحت کمتری دارد ولی سریع‌تر عمل می‌کند. ولی همچنان نمی‌توانیم نظر قطعی بدهیم و تمامی این موارد و معیارهای این چنینی وابسته به نوع داده‌های ورودی و مسئله هستند. چرا که بطور مثال سرعت و دقت PCA وابسته به تعداد ویژگی‌ها و تعداد componentهای مشخص شده دارد.