به نام خدا

پروژه کارشناسی

موضوع:

پیشبینی بیماری سرطان ریه با استفاده از یادگیری ماشین

نام و نام خانوادگی:

هاتف حسین پور

شماره دانشجویی:

۹۷۱۲۲۹۱۰۰۰۹

استاد راهنما:

دکتر احسان عطایی

نیم سال دوم ۱۴۰۰-۱۴۰۱

فهرست

مقدمه

بیماری‌های سرطان اگر به موقع تشخیص داده شوند قابل درمان هستند. در حال حاضر بسیاری از سرطان‌ها چون زود تشخیص داده می‌شوند می‌توان بیمار را تا حد زیادی زنده نگهداشت و بیمار می‌تواند به زندگی خود ادامه دهد این کار با انجام خود آزمایی و غربالگری امکان‌پذیر است. سرطان ریه بیماری است که سلول‌های سرطانی در بافت ریه شروع به تقسیم بی‌رویه می‌کند و توانایی تنفس فرد را کاهش می‌کند. این بیماری اصلی‌ترین دلیل مرگ و میر در زنان و مردان مبتلا به سرطان است و درمان قطعی ندارد. برخی از روش‌های درمانی مورد استفاده در درمان سرطان ریه شامل این موارد است؛ جراحی، شیمی درمانی، پرتودرمانی، ایمونوتراپی، درمان‌های گیاهی. نقش هوش مصنوعی و یادگیری ماشین در تحقیقات سرطان چندین مزیت را ارائه می‌دهد که در درجه اول افزایش پردازش اطلاعات و افزایش دقت تصمیم‌گیری بالینی است. ابزارهای کلیدی فعال در حال حاضر در پزشکی دقیق، که در اینجا به عنوان سیستم‌های هوشمند نامیده می‌شوند، از حجم داده‌های بی‌سابقه استفاده می‌کنند و با هدف مدل‌سازی تأثیرات و متغیرهای ناهمگن زمینه‌ای آن‌ها مرتبط با پیامدهای بیماران هستند.

نرم افزار های مشابه

۱. بیماری قلب

تیمی از مهندسین از Caltech، موسسه تحقیقات پزشکی هانتینگتون و دانشگاه کالیفرنیای جنوبی (USC) اپلیکیشنی را برای نظارت بر سلامت قلب و عروق توسعه داده اند.

این برنامه با نظارت بر کسر جهشی بطن چپ (LVEF) یا به عبارت عامیانه تر، ضربان قلب کار می‌کند. بیماران گوشی هوشمند را روی گردن خود می‌گیرند، جایی که می‌تواند جریان خون را با اندازه‌گیری انبساط و انقباض دیواره‌های شریان کنترل کند.

۲. سرطان پانکراس

نرخ بقای پنج ساله سرطان پانکراس فقط ۹ درصد است، تا حدی به این دلیل که علائم واضحی وجود ندارد. در حال حاضر، تنها ابزار غربالگری پیشگیرانه، آزمایش خون طولانی و اغلب پرهزینه است.

محققان در دانشگاه واشنگتن اپلیکیشنی به نام BiliScreen توسعه داده اند که می‌تواند تشخیص زودهنگام را بی نهایت آسان کند. یکی از علائم اولیه سرطان لوزالمعده زردی است که به دلیل تجمع بیلی روبین در خون ایجاد می‌شود. با این حال، زمانی که تغییر رنگ زرد مشخصه آن در پوست و صلبیه با چشم غیر مسلح قابل مشاهده باشد، سرطان معمولاً کاملاً پیشرفته است.

این برنامه از الگوریتم‌های بینایی کامپیوتری و ابزارهای یادگیری ماشینی برای تشخیص سطوح بیلی‌روبین حداقلی استفاده می‌کند که تشخیص زودهنگام را آسان‌تر از همیشه می‌کند. مطالعه بالینی اولیه نشان داد که این اپلیکیشن در ۸۹/۷ درصد موارد به درستی موارد نگران کننده را شناسایی کرده است.

۳. آسیب مغزی

گروه دیگری از محققان دانشگاه واشنگتن (UW) اپلیکیشنی به نام PupilScreen ساخته اند که می‌تواند به سرعت علائم ضربه مغزی را شناسایی کند.

این برنامه تغییر در پاسخ مردمک به نور را اندازه‌گیری می‌کند و به ویژه هیجان‌انگیز است زیرا می‌تواند مؤثرتر از روش‌های فعلی برای شناسایی ضربه مغزی باشد.

مهندسان این پروژه گزارش داده اند: «جایی که آزمایش نور قلمی آسیب‌های عمده مغزی را بررسی می‌کند، PupilScreen می‌تواند تغییرات ظریف‌تر مرتبط با ضربه مغزی را شناسایی کند، تغییراتی که برای چشم انسان نامحسوس است.»

راحتی و دقت برنامه به این معنی است که این برنامه پتانسیل آن را دارد که در درمان آسیب‌های ورزشی مؤثر باشد، جایی که تشخیص در حاشیه می‌تواند نادرست باشد و تحت تأثیر خارجی قرار گیرد.

توصیف پروژه

در این پروژه سعی شده‌است با استفاده از یادگیری ماشین احتمال ابتلای فرد را به سرطان ریه بررسی شود. در این وب اپلیکیشن سوالاتی از کاربر درباره علائم این بیماری پرسیده می‌شود و با بررسی پاسخ های کاربر به احتمال بیماری او پیشبینی می‌شود لازم به ذکر است که احتمالی که توسط مدل یادگیری ماشین محاسبه می‌شود صرفا ابتلای صد در صدی فرد به بیماری نیست و در این زمینه باید پزشک متخصص نظر نهایی را بدهد.

حوزه داده کاوی و یادگیری ماشین، یکی از حوزه‌های هوش مصنوعی است که به سه دسته یادگیری نظارت شده(که ورودی و خروجی آن مشخص است) ، یادگیری نظارت نشده (که داده های مشخصی از قبل وجود ندارد و هدف ارتباط ورودی و خروجی نیست، بلکه تنها دسته‌بندی آن‌ها مهم است) و یادگیری نیمه نظارت شده (که هم از داده های طبقه بندی شده (برچسب خورده) و هم از داده های غیر طبقه بندی شده (برچسب نخورده) به صورت همزمان استفاده می‌شود) تقسیم می‌شود.

مسائل یادگیری نظارت شده را می‌توان بیشتر به مسائل رگرسیون(Regression) و طبقه بندی(Classification) دسته بندی کرد.

مسائل Regression زمانی است که متغیر خروجی یک مقدار واقعی یا پیوسته است، مانند "حقوق" یا "وزن".

مسائل Classification زمانی است که متغیر خروجی یک دسته است مانند فیلتر کردن ایمیل‌های "اسپم" یا "غیر اسپم".

برای پیشبینی توسط مدل یادگیری ماشین به یک دیتاست نیاز است که یادگیری توسط نمونه‌های آن انجام شود. دیتاست مورد استفاده در این پروژه از این لینک قابل دسترسی است.

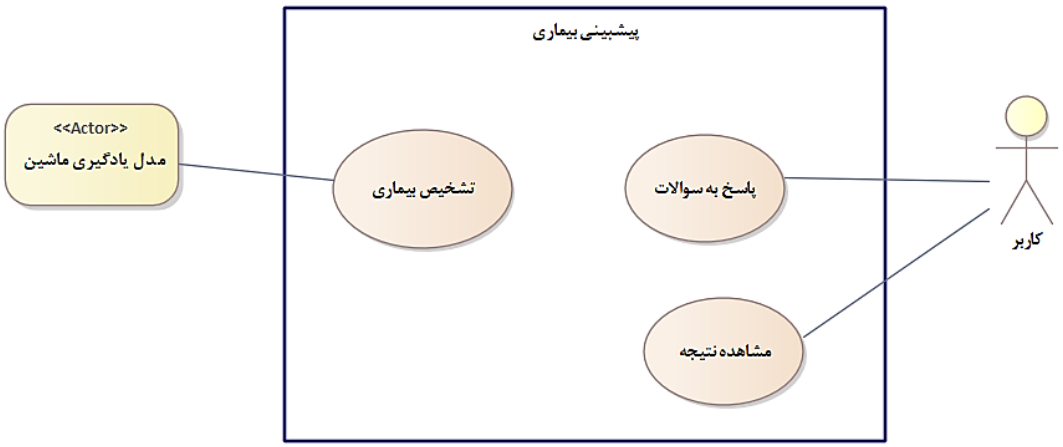
در دیتاست ما متغیر نتیجه یا متغیر وابسته داریم که فقط دو مجموعه مقدار دارد، ۲ (بدخیم) یا ۱ (خوش‌خیم). بنابراین از الگوریتم classification یادگیری نظارت شده استفاده خواهیم کرد.

جدول اطلاعات آماری دیتاست:

اطلاعات / ویژگی	میانگین	انحراف معیار	چارک‌اول	میانه	چارک‌سوم	مینیمم	ماکزیمم
Gender	0.502004	0.50000	0	0	1	0	1
Age	44.137614	15.13309217	31	44	57	18	87
Smoking	1.499532	0.500004	1	1	2	1	2
Yellow Finger	1.496299	0.499991	1	1	2	1	2
Anxiety	1.500614	1.500004	1	1	2	1	2
Peer Pressure	1.496769	0.499994	1	1	2	1	2
Chronic Disease	1.501047	0.500003	1	1	2	1	2
Fatigue	1.497924	0.500000	1	1	2	1	2
Allergy	1.501914	0.500001	1	1	2	1	2
Wheezing	1.501065	0.500003	1	1	2	1	2
Alcohol Consuming	1.498447	0.500002	1	1	2	1	2
Coughing	1.505055	0.499979	1	1	2	1	2
Shortness of Breath	1.500758	0.500004	1	1	2	1	2
Swallowing Difficulty	1.499964	0.500005	1	1	2	1	2
Chest Pain	1.501643	0.500002	1	1	2	1	2

تحلیل و طراحی پروژه

نمودار مورد کاربرد



توصیف موارد کاربرد

مورد کاربرد: پاسخ به سوالات
شناسه: ۱
توصیف مختصر: کاربر به سوالات مرتبط با علائم بیماری پاسخ می‌دهد
کنشگر اصلی: کاربر
کنشگر فرعی: مدل یادگیری ماشین
پیش شرط: کاربر بر دکمه پیشبینی کلیک کند
جریان اصلی: ابتدا از کاربر سوالاتی پرسیده می‌شود که این کاربر به سوالات باید پاسخ دهد و دکمه تایید کلیک کند
پس شرط: اطلاعات به مدل برای پیشبینی فرستاده می‌شود
جریان فرعی: ندارد

مورد کاربرد: مشاهده نتیجه
شناسه: ۲
توصیف مختصر: کاربر وضعیت احتمال ابتلای خود به بیماری را مشاهده می‌کند.
کنشگر اصلی: کاربر
کنشگر فرعی: مدل یادگیری ماشین
پیش شرط: دریافت پاسخ کاربر
جریان اصلی: براساس نوع مدل یادگیری ماشین ابتلا یا عدم ابتلا به بیماری به کاربر نمایش داده خواهد شد.
پس شرط: ندارد
جریان فرعی: ندارد

مورد کاربرد: تشخیص بیماری
شناسه: ۳
توصیف مختصر: ماشین براساس پاسخ سولات کاربر و با توجه به یادگیری که انجام شده است ابتلا یا عدم ابتلا به بیماری را بررسی می‌کند
کنشگر اصلی: مدل یادگیری ماشین
کنشگر فرعی: کاربر
پیش شرط: ندارد
جریان اصلی: ۱. کاربر به سوالات پاسخ می‌دهد ۲. پاسخ این سوالات به مدل ارسال می‌شود **۳.** مدل ابتلای به بیماری را بررسی می‌کند.
پس شرط: فرستادن نتیجه برای کاربر
جریان فرعی: ندارد

سند مشخصات تکمیلی

طراحی:

- تحلیل و طراحی Uml این سایت توسط نرم افزار Sparx Enterprise Architect Ultimate 15.2.1554 نوشته شده است.
- محیط برنامه نویسی برای این سایت Visul Studio Code است.
- طراحی front end این سایت با استفاده از HTML5 ، CSS3 ، Bootstrap v5.1.3 ، JavaScript و JQuery 3.6.0 انجام شده است.
- طراحی back end سایت با استفاده از زبان برنامه نویسیPython و فریمورک Flask انجام شده است.

قابلیت سازگاری:

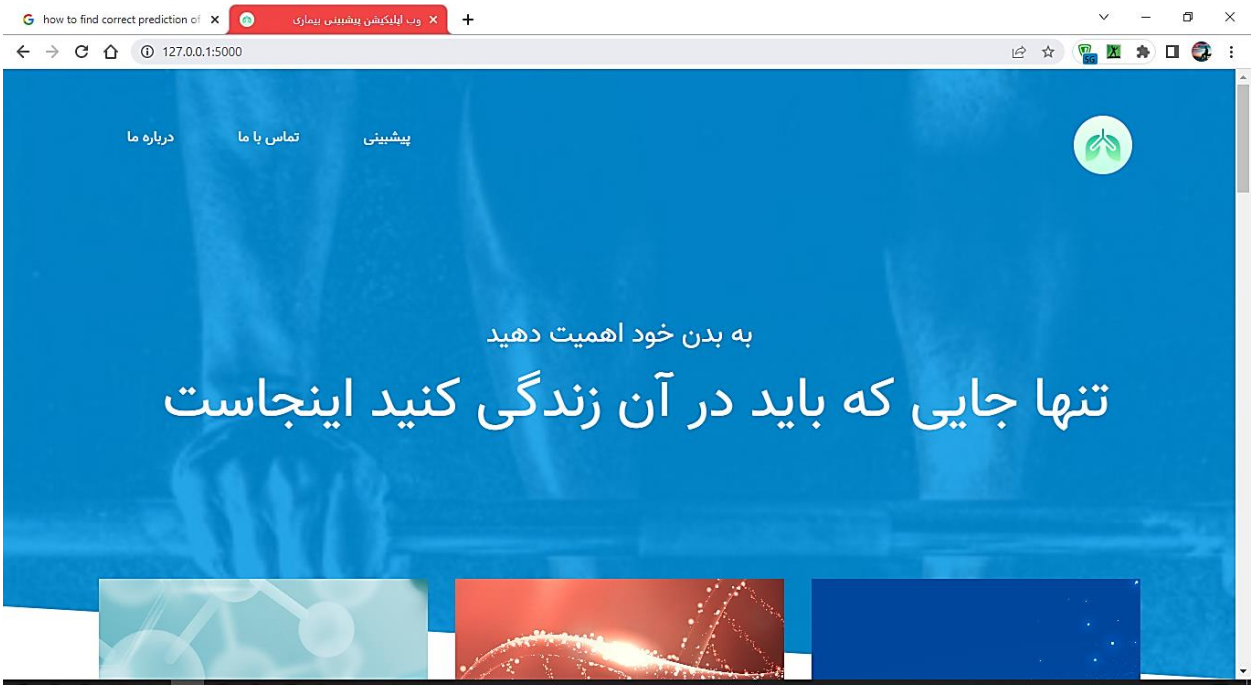
این اپلیکیشن باید قابلیت اجرا بر روی بیشتر مرورگر های وب مانند Chrome، Firefox، Microsoft Edgeو ... را داشته باشد.

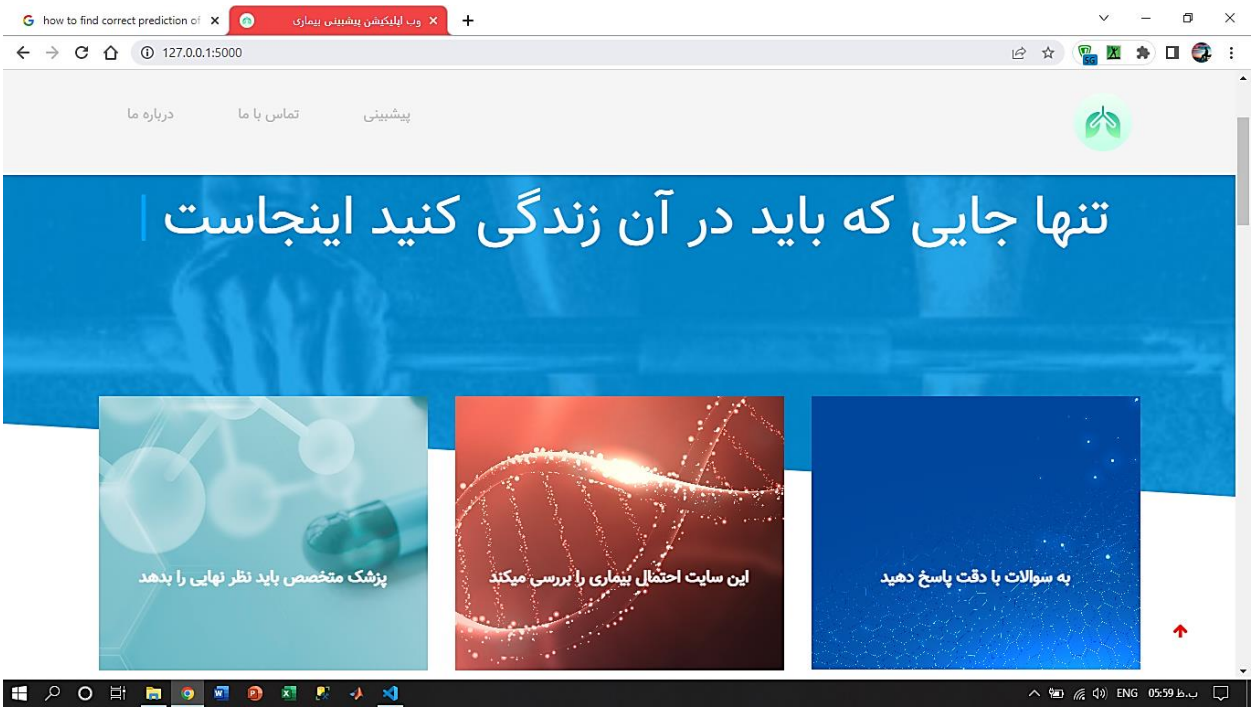
اطمینان:

اطلاعات شخصی کاربران این سیستم به صورت محرمانه ثبت خواهد شد.

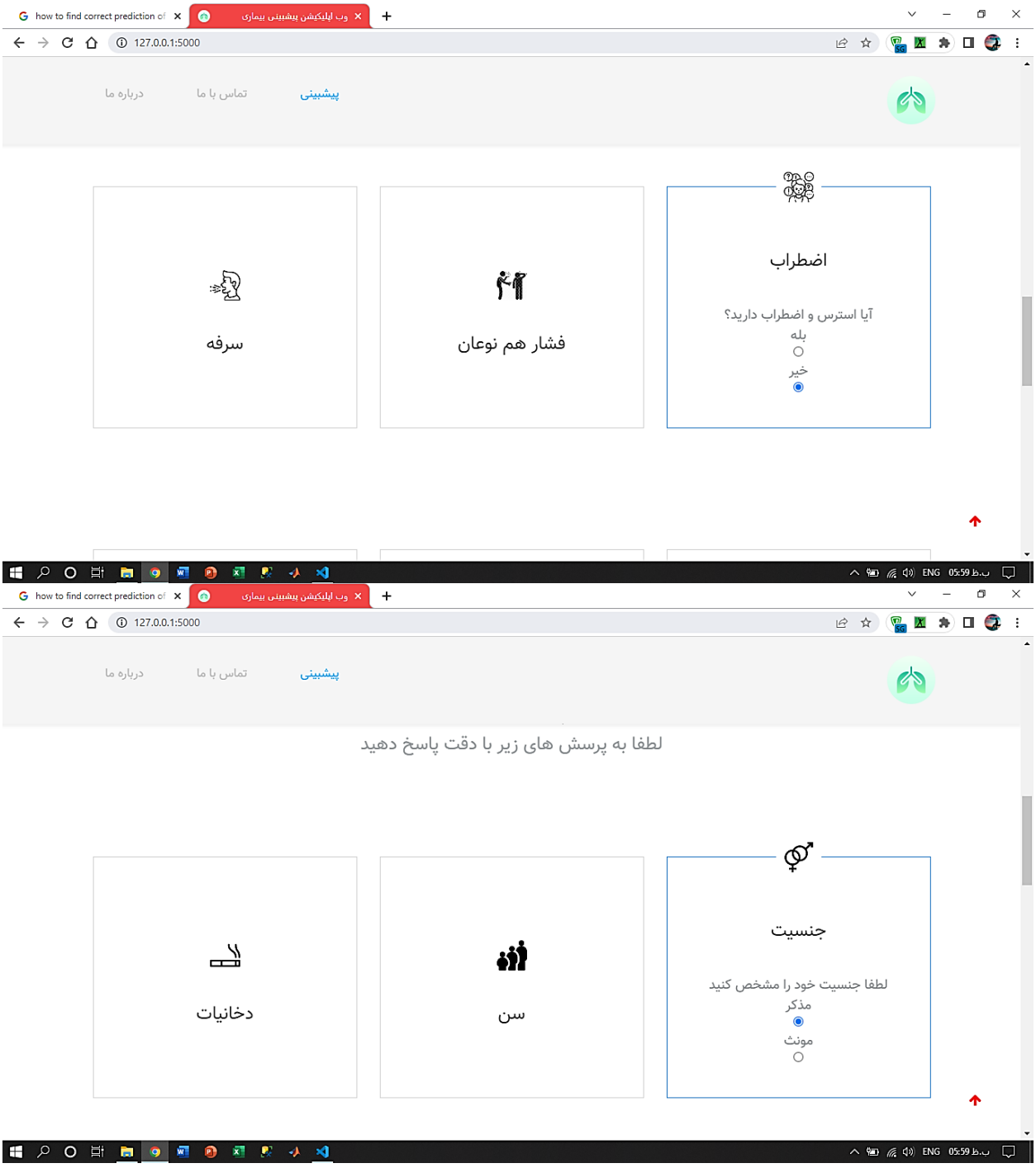
سند واسط کاربری

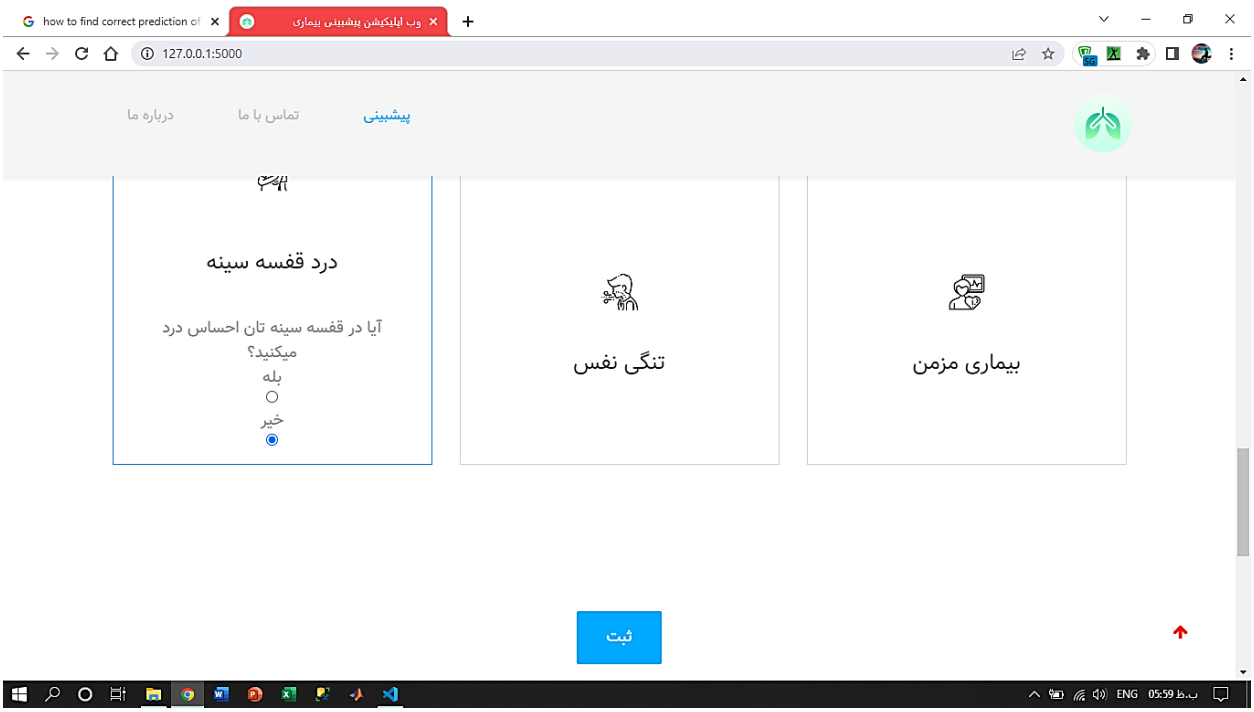
هدر سایت



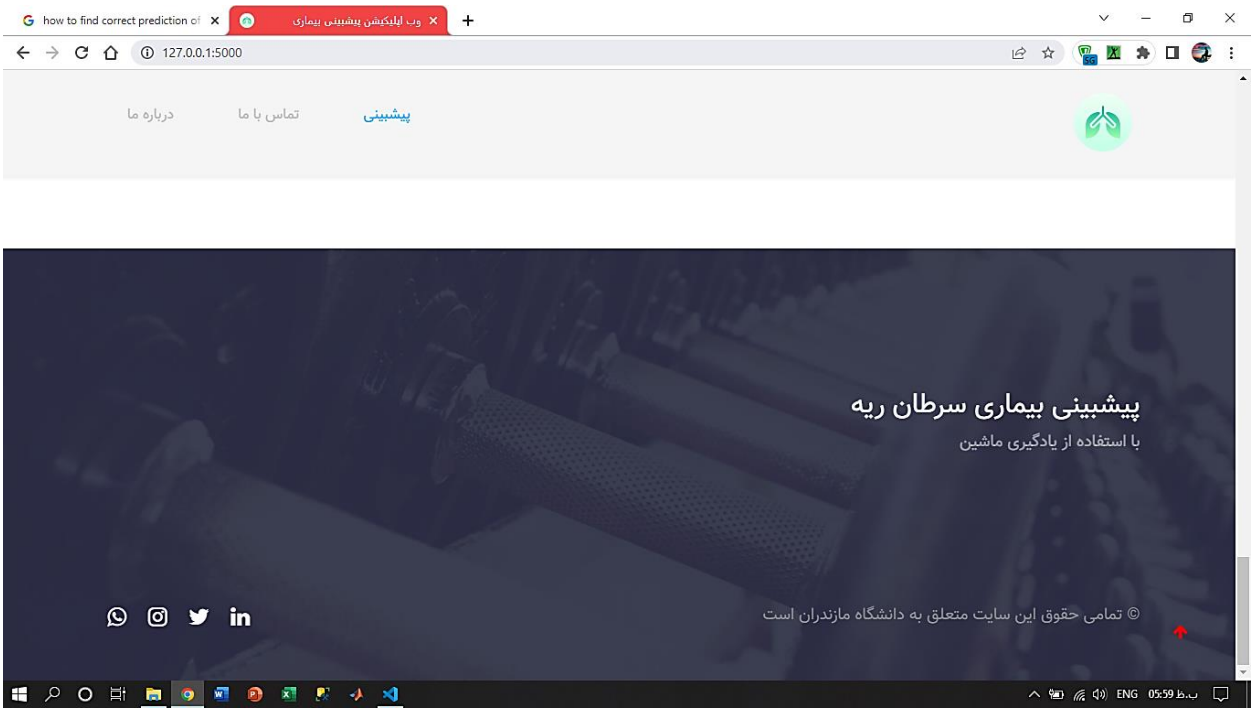


قسمتی که سوالات از کاربر پرسیده می‌شود

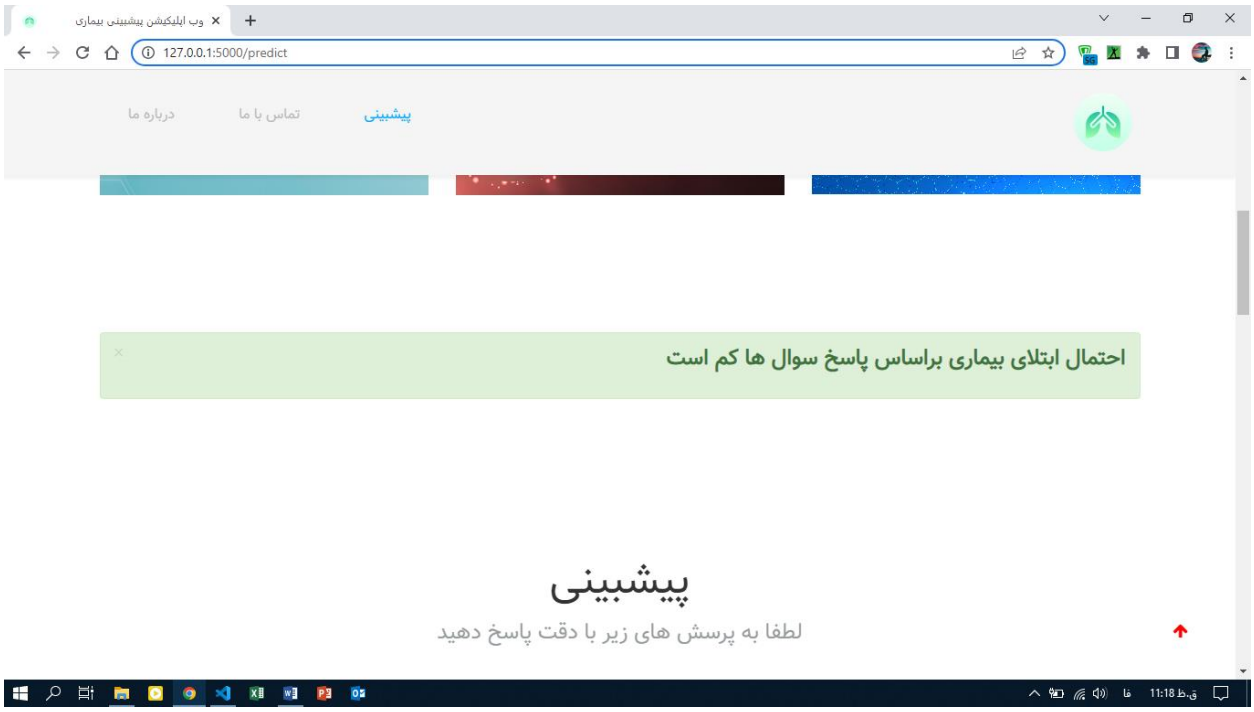



how to find correct prediction of
وب اپلیکیشن پیشبینی بیماری
127.0.0.1:5000
پیشبینی
تماس با ما
درباره ما
درد قفسه سینه
آیا در قفسه سینه تان احساس درد میکنید؟
بله
خیر
تنگی نفس
بیماری مزمن
ثبت

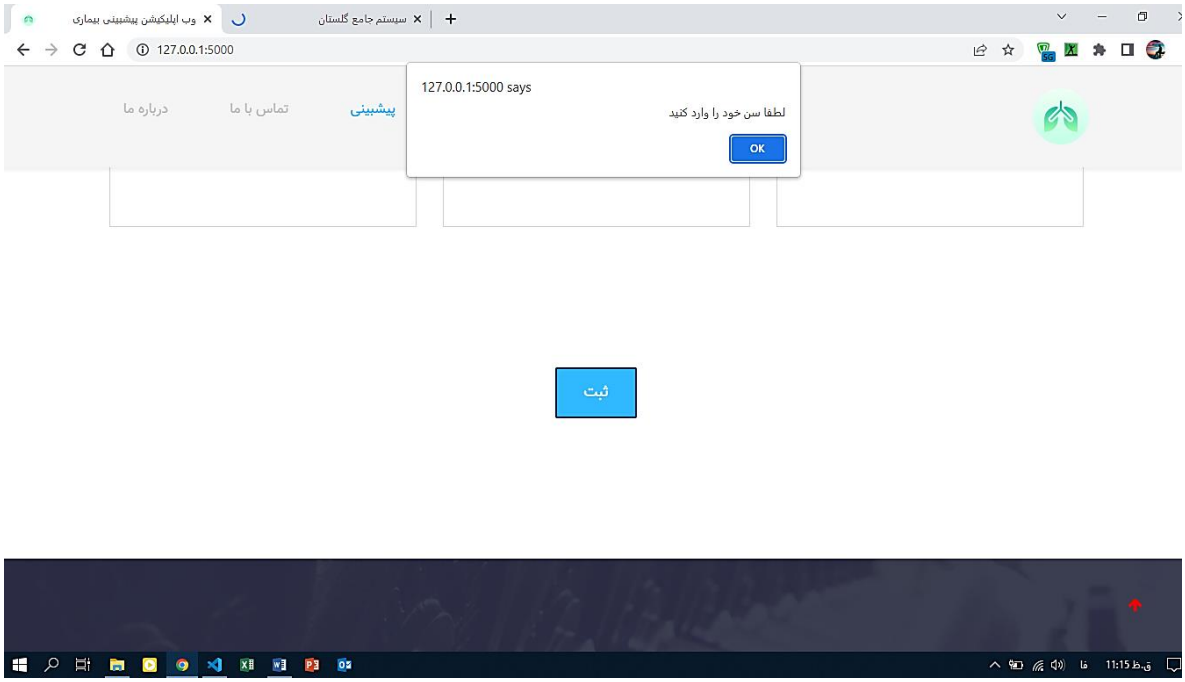
فوتر سایت


how to find correct prediction of
وب اپلیکیشن پیشبینی بیماری
127.0.0.1:5000
پیشبینی
تماس با ما
درباره ما
پیشبینی بیماری سرطان ریه
با استفاده از یادگیری ماشین
© تمامی حقوق این سایت متعلق به دانشگاه مازندران است

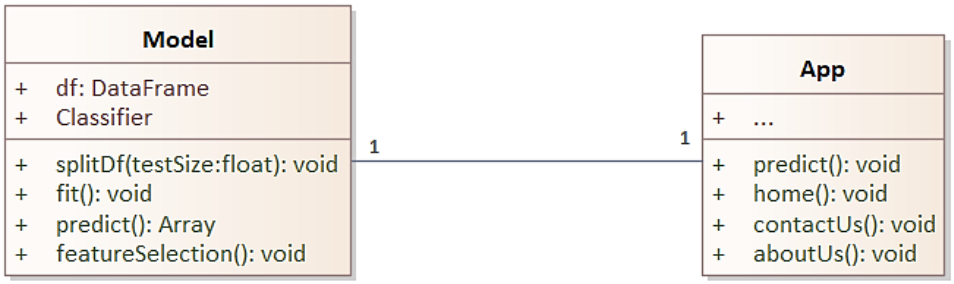
خروجی بعد از تایید اطلاعات توسط کاربر



اعتبارسنجی فرم

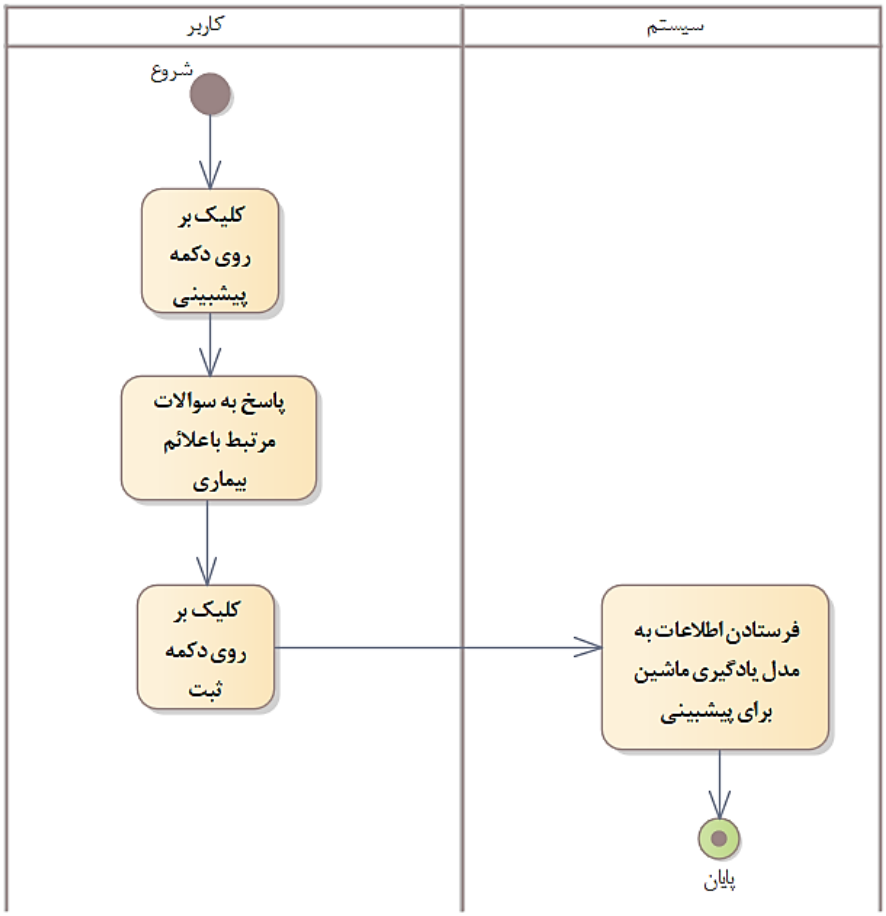


نمودار رده

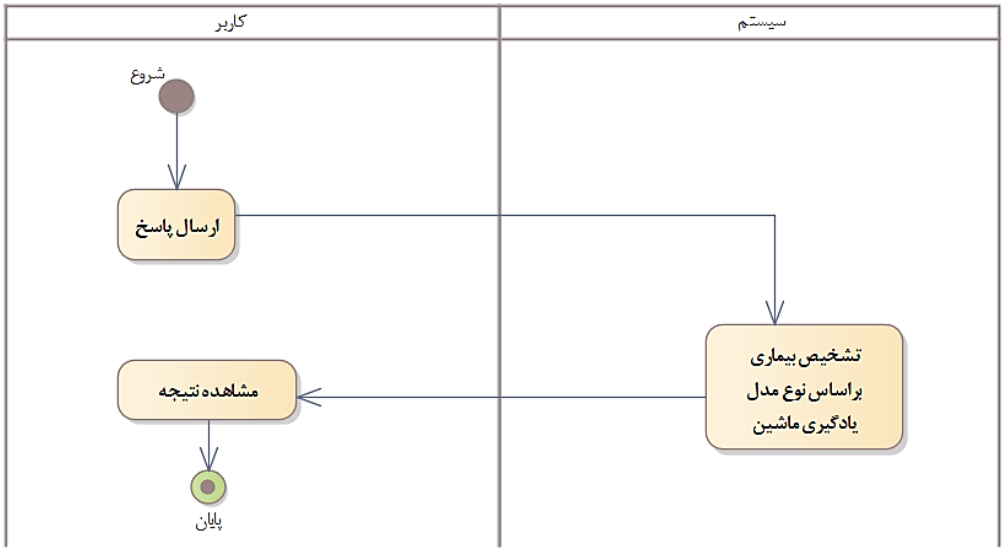

Model
+ df: DataFrame
+ Classifier
+ splitDf(testSize:float): void
+ fit(): void
+ predict(): Array
+ featureSelection(): void
1
1
App
+ ...
+ predict(): void
+ home(): void
+ contactUs(): void
+ aboutUs(): void

نمودار فعالیت

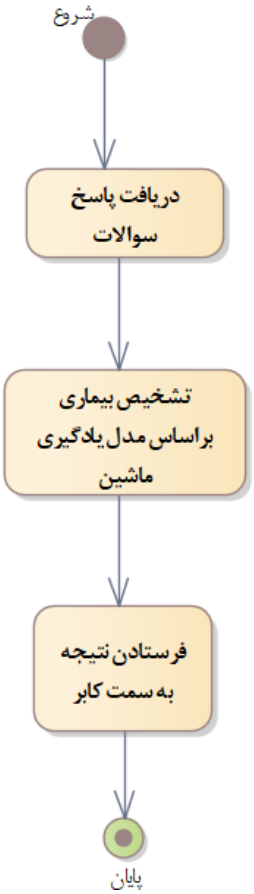
مورد کاربرد: پاسخ به سوالات



مورد کاربرد: مشاهده نتیجه

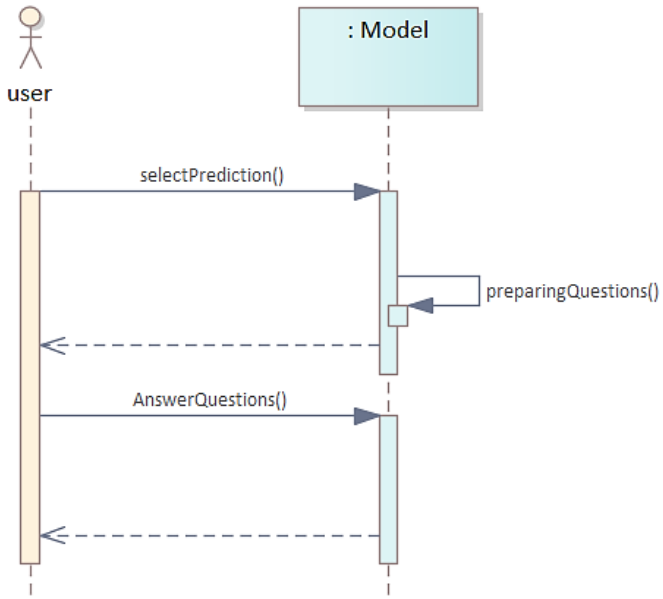


مورد کاربرد: تشخیص بیماری

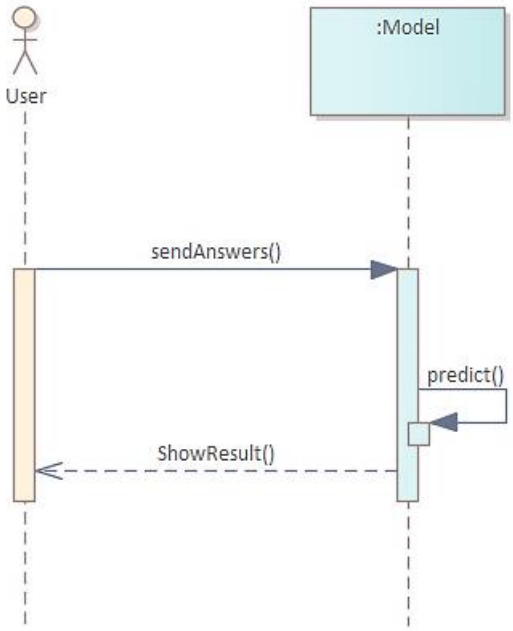


نودار توالی

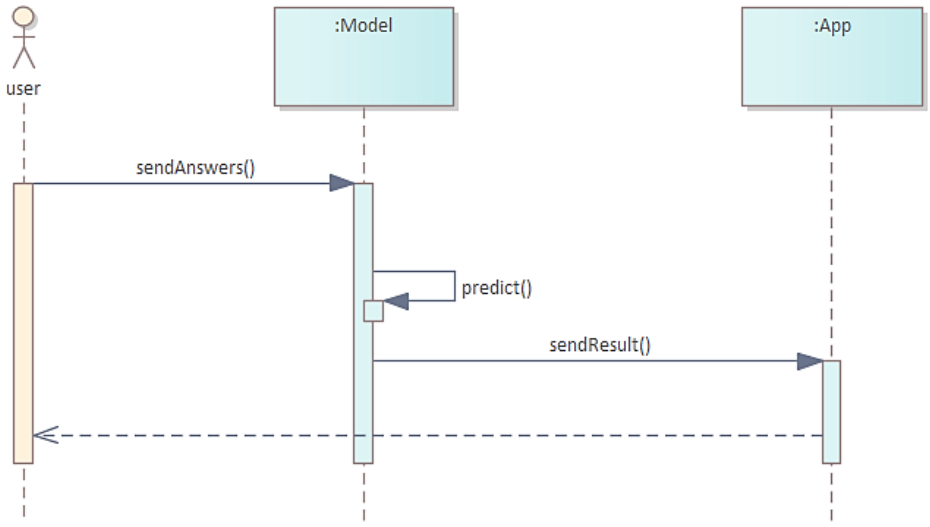
مورد کاربرد: پاسخ به سوالات



مورد کاربرد: مشاهده نتیجه



مورد کاربرد: تشخیص بیماری



توضیحات مرتبط با کد پروژه

فاز اول - کاوش داده:

ابتدا با استفاده از کتابخانه Pandas دیتاست مورد نظر را در یک دیتافریم ذخیره می‌کنیم. سپس دیتافریم را به دو قسمت x و y تقسیم می‌کنیم. توجه کنید که قسمت x تمام ویژگی های مستقل دیتاست است و y ویژگی وابسته یا class lable است.

داده‌هایی که استفاده می‌کنیم معمولاً به داده‌های آموزشی و داده‌های آزمایشی تقسیم می‌شوند. مجموعه آموزشی حاوی یک خروجی شناخته شده است و مدل بر روی این داده ها می‌آموزد تا بعداً به داده های دیگر تعمیم یابد. ما مجموعه داده آزمایشی (یا زیر مجموعه) را داریم تا پیش‌بینی مدل خود را روی این زیر مجموعه آزمایش کنیم.

ما این کار را با استفاده از کتابخانه SciKit-Learn در پایتون با استفاده از روش train_test_split انجام خواهیم داد. به طور معمول می‌توان ۲۰ تا ۳۰ درصد از داده را به عنوان داده آموزشی در نظر گرفت.

```
49  ################## Split df into to train set and test set ################
50
51  def splitDf(self, testSize):
52      # print(len(self.df))
53
54      # Remove all duplicate rows
55      self.df.drop_duplicates(inplace=True)
56
57      # print(len(self.df))
58
59      # after feature selection this feature will be removed
60      self.df.drop(
61          self.df.columns[[3, 7, 8, 9, 10, 13]], axis=1, inplace=True)
62
63      self.x = self.df.drop(columns=["LUNG_CANCER"])  # independent value
64      self.y = self.df["LUNG_CANCER"]  # dependent value Class Lable
65
66      self.x_train, self.x_test, self.y_train, self.y_test = train_test_split(
67          self.x, self.y, test_size=testSize, random_state=10, shuffle=True)
68
69      # print(self.df.isnull().sum())
```

انتخاب ویژگی فرآیندی است که در آن به طور خودکار یا دستی ویژگی‌هایی را در مجموعه داده انتخاب می‌کنید که بیشترین کمک را به متغیر پیش‌بینی یا خروجی مورد علاقه شما دارند. در این پروژه با استفاده از متد SelectKBest از کتابخانه SciKit-Learn که K ویژگی با امتیاز بالا را انتخاب می‌کند.

```
92
93      def featureSelection(self):
94
95          bestfeatures = SelectKBest(score_func=chi2, k=9)
96          fit = bestfeatures.fit(self.x, self.y)
97
98          dfscores = pd.DataFrame(fit.scores_)
99          dfcolumns = pd.DataFrame(self.x.columns)
100
101         # concat two dataframes for better visualization
102         featureScores = pd.concat([dfcolumns, dfscores], axis=1)
103         # naming the dataframe columns
104         featureScores.columns = ["Specs", "Score"]
105         selectedFeatures = featureScores.nlargest(9, "Score")
106
107         return selectedFeatures
108
```

	Specs	Score
1	AGE	3.239057
0	GENDER	1.523405
4	PEER_PRESSURE	0.625218
6	COUGHING	0.443232
5	CHRONIC DISEASE	0.256805
3	ANXIETY	0.196494
8	CHEST PAIN	0.132435
7	SHORTNESS OF BREATH	0.119491
2	SMOKING	0.049624

فاز دوم - مقیاس‌بندی ویژگی‌ها:

در بیشتر مواقع، مجموعه داده شما حاوی ویژگی هایی است که از نظر بزرگی، واحد و محدوده بسیار متفاوت هستند. اما از آنجایی که اکثر الگوریتم های یادگیری ماشینی از فاصله اقلیدسی بین دو نقطه داده در محاسبات خود استفاده می‌کنند. ما باید همه ویژگی ها را به یک سطح از بزرگی برسانیم. این را می‌توان با مقیاس بندی به دست آورد. این به این معنی است که شما داده‌های خود را طوری تغییر می‌دهید که در یک مقیاس خاص، مانند ۱۰۰-۰ یا ۱-۰ قرار بگیرند.

```
71      ################################### Feature Scaling ######
72
73      def fit(self):
74
75          self.sc = StandardScaler()
76          self.x_train = self.sc.fit_transform(self.x_train)
77          self.x_test = self.sc.transform(self.x_test)
78
79          model = self.voting.fit(self.x_train, self.y_train)
80
81          return model
```

ما از روش StandardScaler از کتابخانه SciKit-Learn استفاده خواهیم کرد.

فاز سوم - انتخاب مدل:

ما انواع مختلفی از الگوریتم های طبقه بندی(Classification) را در یادگیری ماشین داریم:

۱. Logistic Regression: نمونه ای از یادگیری نظارت‌شده است. برای محاسبه یا پیش بینی احتمال وقوع یک رویداد باینری (بله/خیر) استفاده می‌شود.

۲. Nearest Neighbor: یک الگوریتم یادگیری ماشین ساده و نظارت‌شده که می‌تواند برای حل مسائل Classification و Regression استفاده شود.

۳. Support Vector Machines: در یادگیری ماشینی،SVM مدل‌های یادگیری نظارت‌شده با الگوریتم‌های یادگیری مرتبط هستند که داده‌ها را برای Classification و Regression تجزیه و تحلیل می‌کنند.

۴. Kernel SVM: Kernel تابعی است که در SVM برای کمک به حل مسائل استفاده می شود.

۵. Naïve Bayes: یک الگوریتم طبقه بندی برای مسائل Classification باینری (دو کلاسه) و چند کلاسه است. درک این تکنیک زمانی که با استفاده از مقادیر ورودی باینری یا طبقه‌بندی توصیف می‌شود، راحت‌تر است.

۶. Decision Tree Algorithm: رویکردی که در یادگیری ماشین نظارت‌شده استفاده می‌شود، تکنیکی که از مجموعه داده های ورودی و خروجی برچسب‌دار برای آموزش مدل‌ها استفاده می‌کند. این رویکرد عمدتاً برای حل مسائل Classification استفاده می شود.

۷. Random Forest Classification: یک الگوریتم یادگیری ماشینی نظارت شده است که به طور گسترده در مسائل Classification و Regression استفاده می‌شود. درخت های تصمیم را بر روی نمونه های مختلف می-سازد و اکثریت آن ها را برای طبقه بندی و در صورت رگرسیون میانگین می‌گیرد.

اما روش‌های دیگری هم برای Classification موجود است یکی از آنها ensemble هستند.

Ensembling یک تکنیک قدرتمند برای بهبود عملکرد مدل با ترکیب مدل های پایه مختلف به منظور تولید یک مدل بهینه و قوی است. روش استفاده شده در این پروژه Voting Classification، تخمینزننده(Estimator) یادگیری ماشینی است که مدل‌های پایه یا برآوردگرهای مختلفی را آموزش می‌دهد و بر اساس جمع‌آوری یافته‌های هر Estimator پایه، پیش‌بینی می‌کند. معیارهای تجمیع را می‌توان ترکیبی از تصمیم رای گیری برای خروجی هر برآوردگر باشد. معیارهای Voting می‌تواند دو نوع باشد:

Hard Voting: Voting بر اساس کلاس خروجی پیش بینی شده محاسبه می‌شود.

Soft Voting: Voting بر اساس احتمال پیش بینی شده کلاس خروجی محاسبه می‌شود.

اکنون نتایج مجموعه تست را پیش‌بینی می‌کنیم و دقت را با هر یک از مدل‌های خود بررسی می‌کنیم

```python
class Model:
    def __init__(self, dataFile="survey lung cancer.csv"):

        self.df = pd.read_csv(dataFile)

        pd.set_option("display.max.rows", None)
        pd.set_option("display.max.columns", None)

        self.dt = DecisionTreeClassifier(criterion='entropy', random_state=0)
        self.rnd = RandomForestClassifier(n_estimators=100, criterion="entropy", random_state=0)
        self.knn = KNeighborsClassifier(n_neighbors=5, metric='minkowski', p=2)

        self.voting = VotingClassifier(
            estimators=[
                ("decision_tree", self.dt),
                ("random_forest", self.rnd),
                ("k_neighbors", self.knn),
            ],
            voting="hard"
        )
```

برای بررسی دقت باید متد Confusion_matrix را وارد کنیم. ماتریس Confusion راهی برای جدول‌بندی تعداد Classification های اشتباه است. ما از روش Classification Accuracy برای یافتن دقت مدل های خود استفاده خواهیم کرد. وقتی از اصطلاح دقت استفاده می‌کنیم، منظور ما معمولاً دقت طبقه بندی است. این نسبت تعداد پیش بینی های صحیح به تعداد کل نمونه های ورودی است.

```python
    def predict(self):

        y_pred = self.voting.predict(self.x_test)

        Accuracy = accuracy_score(self.y_test, y_pred)
        Conf_Matrix = confusion_matrix(self.y_test, y_pred)

        # tn, fp, fn, tp = confusion_matrix(self.y_test, y_pred).ravel()
        # Accuracy = ((tp+tn)/(tn+fp+fn+tp))

        return [Accuracy, Conf_Matrix]

```

مقدار Accuracy این مدل ۵۰% و ماتریس Confusion مانند شکل زیر است:

		پیشبینی‌شده	
		۱(خوشخیم)	۲(بدخیم)
واقعی	۱(خوشخیم)	TN = 3007	FP = 2466
	۲(بدخیم)	FN = 2977	TP = 2451

مثبت واقعی (TP): اینها مواردی هستند که آنها "2" پیش بینی کردیم (آنها این بیماری را دارند) و آنها این بیماری را دارند.

منفی واقعی (TN): ما "1" پیش بینی کردیم ، و آنها این بیماری را ندارند.

موارد مثبت کاذب (FP): ما "2" پیش بینی کردیم، اما آنها در واقع این بیماری را ندارند. (همچنین به عنوان "خطای نوع I" شناخته می‌شود.)

منفی کاذب (FN): ما "1" پیش بینی کردیم ، اما آنها در واقع این بیماری را دارند. (همچنین به عنوان "خطای نوع II" شناخته می‌شود.)

فاز چهارم - استقرار مدل:

برای استفاده از مدل یادگیری ماشین در قسمت BackEnd این پروژه که با استفاده از فریمورک Flask پایتون طراحی شده است اژ ماژول pickle پایتون استفاده ابتدا مدل یادگیری ماشین را در یک فایل باینری model.pkl ذخیره می‌کنیم.

```
118
119     pickle.dump(model, open('model.pkl', 'wb'))
120
```

سپس در مسیر BackEnd آنرا بارگذاری کرده و از آن برای پیشبینی استفاده می‌کنیم. با استفاده از متد request که از کتابخانه Flask فراخوانی می‌شود از form مورد نظر در سند Html مقادیر input ها که از نوع Radio button هستند دریافت می‌شوند و به مدل یادگیری ماشین برای پیشبینی داده می‌شوند.

ml-finalProjects > templates > main.html > ...

```
159     <!-- start services section -->
160     <form action="{{ url_for('predict') }}" method="post">
161         <section id="services" class="section section-padded">
162             <div class="container">
163                 <div class="row text-center title">
164                     <h2>پیشبینی</h2>
165                     <h4 class="light muted"> لطفا به پرسش های زیر با دقت پاسخ دهید </h4>
166                 </div>
167                 <div class="row services">
168                     <!-- start services section box1 Gender -->
169 
170                     <div class="col-md-4">
171                         <div class="service">
172                             <div class="icon-holder">
173                                 <img src="{{ url_for('static', filename='pics/gender.png') }}" alt="" class="icon">
174                             </div>
175                             <h4 class="heading" id="gender-heading">جنسیت</h4>
176                             <p class="description"> لطفا جنسیت خود را مشخص کنید</p>
177                             <p class="description"> مذکر </p>
178                             <input type="radio" name="gender" value="1" id="gender-1"
179                                 class="description form-check-input" required>
180                             <p class="description form-check-label"> مونث </p>
181                             <input type="radio" name="gender" value="0" id="gende-0"
182                                 class="description form-check-input" required>
183 
184                         </div>
185 
186                     </div>
```

ml-finalProjects > app.py > Flask

```python
from flask import Flask, render_template, request, flash, session
import pickle

app = Flask(__name__)


model = pickle.load(open("model.pkl", "rb"))


@app.route("/")
def home():
    return render_template("main.html")


@app.route("/ContactUs")
def contactUs():
    return render_template("ContactUs.html")


@app.route("/AboutUs")
def aboutUs():
    return render_template("AboutUs.html")


@app.route("/predict", methods=["POST"])
def predict():

```

واژه نامه

واژه	توضیح
اضطراب **معادل: استرس، Anxiety**	تشخیص سرطان باعث احساس ترس، استرس یا اضطراب می‌شود. در بعضی موارد هم می‌تواند علائم بیماری باشد.
خستگی **معادل: Fatigue**	در برخی افراد، خستگی یکی از علائم اولیه وجود سرطان ریه است.
مشکل در بلع **معادل: Swallowing Difficulty**	مشکل در بلع، یکی از علائم بیماری سرطان ریه است
فشار همسالان **معادل: Peer Pressure**	رفتاری که انسان تحت تاثیر فشاری که از طرف هم‌نوعان و همسالان انجام می‌دهد.
دقت **معادل: Accuracy**	دقت الگوریتم Classification یادگیری ماشین یکی از راه‌های اندازه‌گیری تعداد دفعات طبقه‌بندی صحیح یک داده توسط الگوریتم است.
طبقه بندی کننده **معادل: Classifier**	طبقه‌بندی‌کننده در یادگیری ماشین الگوریتمی است که به‌طور خودکار داده‌ها را به یک یا چند مجموعه از «کلاس‌ها» دسته‌بندی می‌کند.
Dataframe	ساختار داده ای که داده ها را در یک جدول دو بعدی از ردیف ها و ستون ها سازماندهی می‌کند.