Đại Học Quốc Gia Thành Phố Hồ Chí Minh

Đại Học Khoa Học Tự Nhiên

Khoa Công Nghệ Thông Tin

Project 2 – Triển khai mô hình phân loại bệnh ung thư vú

1. Thực hiện:

a) Môi trường:

- Ngôn ngữ: Python
- Công cụ: Jupyter Notebook cho xử lý dữ liệu và tkinter xây dựng UI cho application.
- File dữ liệu: data.csv và modified_data.csv là phiên bản rút gọn của data.csv.
- Mô hình sử dụng: Logistic Regression.

b) Thành viên và phân công:

MSSV	Tên
19127592	Lê Minh Trí
19127587	Trương Chí Toàn
19127630	Trần Quốc Việt

Công việc	Người thực hiện	Chi tiết	Hoàn thành
Chuẩn bị dữ liệu	Lê Minh Trí	Làm sạch dữ liệu, tiền xử lý, giảm thiểu trường dữ liệu không cần thiết	100%
Thống kê chi tiết + UI	Trần Quốc Việt	Thống kê cơ bản các trường dữ liệu + xây dựng UI cơ bản	100%
Trực quan dữ liệu + UI	Trương Chí Toàn	Trực quan các trường dữ liệu qua biểu đồ + xây dựng UI cơ bản	100%

2. Chuẩn bị dữ liệu:

2.1. Thu nhập dữ liệu

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	te
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	...	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	...	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	...	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	...	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	...	
...	...	...	...	...	...	...	...	...	...	...	...	
564	926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	...	
565	926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	...	
566	926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	...	
567	927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	...	
568	92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	...	

Có bao nhiêu dòng cột? Có dòng nào bị lặp không ?

```
num_rows = len(brcc_df)
num_cols = len(brcc_df.columns)
print(num_rows , num_cols)
```

✓ 0.2s

```
569 33
```

```
have_duplicated_rows = brcc_df.duplicated().any()
have_duplicated_rows
```

✓ 0.1s

```
False
```

2.2. Khám phá dữ liệu

```
#   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
0   id                      569 non-null    int64
1   diagnosis               569 non-null    object
2   radius_mean             569 non-null    float64
3   texture_mean            569 non-null    float64
4   perimeter_mean          569 non-null    float64
5   area_mean               569 non-null    float64
6   smoothness_mean         569 non-null    float64
7   compactness_mean        569 non-null    float64
8   concavity_mean          569 non-null    float64
9   concave points_mean     569 non-null    float64
10  symmetry_mean           569 non-null    float64
11  fractal_dimension_mean  569 non-null    float64
12  radius_se               569 non-null    float64
13  texture_se              569 non-null    float64
14  perimeter_se            569 non-null    float64
15  area_se                 569 non-null    float64
16  smoothness_se           569 non-null    float64
17  compactness_se          569 non-null    float64
18  concavity_se            569 non-null    float64
19  concave points_se       569 non-null    float64
...
31  fractal_dimension_worst 569 non-null    float64
32  Unnamed: 32             0 non-null      float64
```

Bởi vì có 569 dòng và 33 cột, các trường đều thể hiện đầy đủ dữ liệu trừ cột cuối cùng "Unamed: 32". Ta sẽ drop cột này ở khâu tiền xử lý dữ liệu.

Ngoài ra, các trường dữ liệu dạng số đều là float64 nên không cần chuyển đổi dạng dữ liệu gì trong tập dữ liệu này.

2.3. Làm sạch dữ liệu

Để chắc chắn hơn ở các giá trị null, ta cần phải check xem các giá trị 0 ở các trường dữ liệu bởi vì đối với việc chẩn đoán bệnh và liên quan đến cơ thể con người,các giá trị cần được minh bạch.

Sau khi thực hiện kiểm tra các dòng dữ liệu có kết quả bằng 0, ta được kết quả sau…

```
for col in numerical_cols:
    s = brcc_df[col].value_counts()
    try:
        print(col,s[0])
    except:
        continue
```

✓ 0.9s

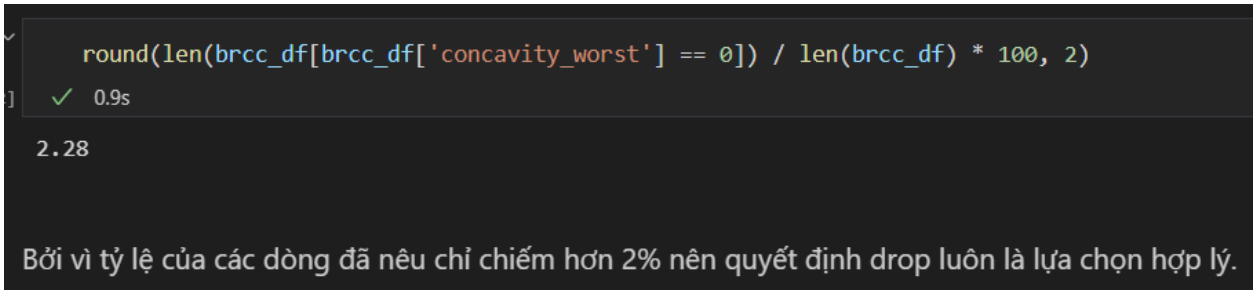
```
concavity_mean 13
concave points_mean 13
concavity_se 13
concave points_se 13
concavity_worst 13
concave points_worst 13
```

Hình ảnh chứng tỏ các giá trị 0 xuất hiện ở các cột liên quan đến concavity và có tất cả 13 dòng.

Với mỗi cột bất kì trong các trường dưới đây có giá trị bằng 0 thì các cột còn lại đều bằng 0.

	concavity_mean	concave points_mean	concavity_se	concave points_se	concavity_worst	concave points_worst
101	0.0	0.0	0.0	0.0	0.0	0.0
140	0.0	0.0	0.0	0.0	0.0	0.0
174	0.0	0.0	0.0	0.0	0.0	0.0
175	0.0	0.0	0.0	0.0	0.0	0.0
192	0.0	0.0	0.0	0.0	0.0	0.0
314	0.0	0.0	0.0	0.0	0.0	0.0
391	0.0	0.0	0.0	0.0	0.0	0.0
473	0.0	0.0	0.0	0.0	0.0	0.0
538	0.0	0.0	0.0	0.0	0.0	0.0
550	0.0	0.0	0.0	0.0	0.0	0.0
557	0.0	0.0	0.0	0.0	0.0	0.0
561	0.0	0.0	0.0	0.0	0.0	0.0
568	0.0	0.0	0.0	0.0	0.0	0.0

Vậy, ta sẽ thực hiện làm sạch dữ liệu bằng cách bỏ các dòng này đi, cụ thể là 13 dòng như trên. Việc bỏ này là hợp lí vì số lượng các dòng này chỉ chiếm 2.28% trên tập dữ liệu tổng.

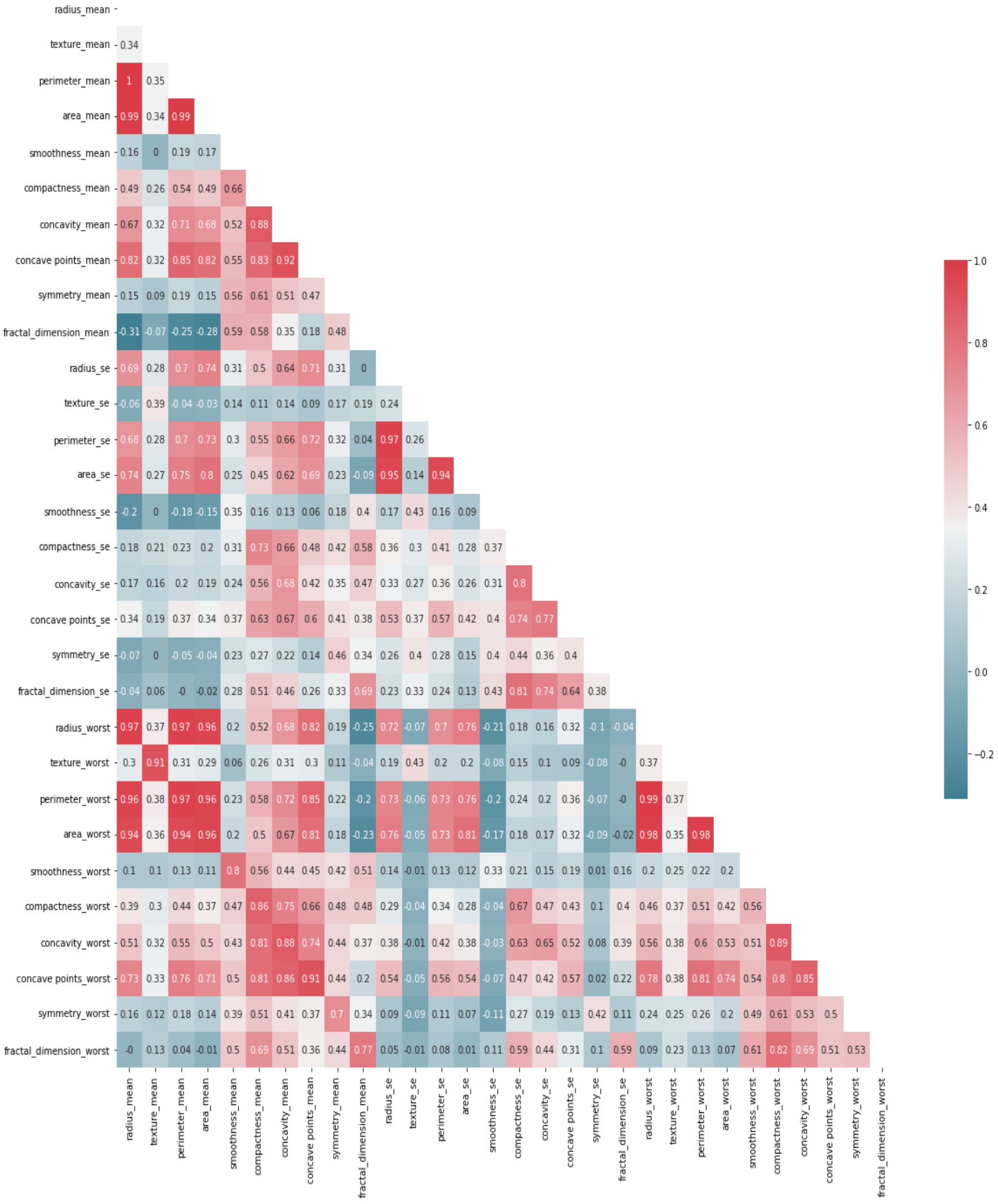


2.4. Tiền xử lý dữ liệu

Các dữ liệu đều là float và nằm trong khoảng cho phép nên ta không cần chuẩn hóa. Khi thực hiện mô hình hóa, các trường liên quan đến nhận dạng của từng bệnh nhân là không cần thiết vì ta chỉ cần quan tâm đến kết quả đạt được, ta có thể bỏ cột ID cũng như cột Unamed: 32 như đã nhắc ở phần khám phá dữ liệu.

Vẫn còn một vấn đề nữa, bởi vì đề yêu cầu ta nhập vào các giá trị đầu của các thuộc tính để bắt đầu chẩn đoán kết quả. Nhưng ta lại có tận 31 cột với các giá trị khác nhau để nhập, ta cần tìm sự tương quan giữa các trường với nhau để loại bỏ các trường không cần thiết và thu gọn tập dữ liệu.

Bằng cách áp dụng kỹ thuật heatmap, ta có thể trực quan hệ số tương quan giữa các biến với nhau:


radius_mean
texture_mean 0.34
perimeter_mean 1 0.35
area_mean 0.99 0.34 0.99
smoothness_mean 0.16 0 0.19 0.17
compactness_mean 0.49 0.26 0.54 0.49 0.66
concavity_mean 0.67 0.32 0.71 0.68 0.52 0.88
concave points_mean 0.82 0.32 0.85 0.82 0.55 0.83 0.92
symmetry_mean 0.15 0.09 0.19 0.15 0.56 0.61 0.51 0.47
fractal_dimension_mean -0.31 -0.07 -0.25 -0.28 0.59 0.58 0.35 0.18 0.48
radius_se 0.69 0.28 0.7 0.74 0.31 0.5 0.64 0.71 0.31 0
texture_se -0.06 0.39 -0.04 -0.03 0.14 0.11 0.14 0.09 0.17 0.19 0.24
perimeter_se 0.68 0.28 0.7 0.73 0.3 0.55 0.66 0.72 0.32 0.04 0.97 0.26
area_se 0.74 0.27 0.75 0.8 0.25 0.45 0.62 0.69 0.23 -0.09 0.95 0.14 0.94
smoothness_se -0.2 0 -0.18 -0.15 0.35 0.16 0.13 0.06 0.18 0.4 0.17 0.43 0.16 0.09
compactness_se 0.18 0.21 0.23 0.2 0.31 0.73 0.66 0.48 0.42 0.58 0.36 0.3 0.41 0.28 0.37
concavity_se 0.17 0.16 0.2 0.19 0.24 0.56 0.68 0.42 0.35 0.47 0.33 0.27 0.36 0.26 0.31 0.8
concave points_se 0.34 0.19 0.37 0.34 0.37 0.63 0.67 0.6 0.41 0.38 0.53 0.37 0.57 0.42 0.4 0.74 0.77
symmetry_se -0.07 0 -0.05 -0.04 0.23 0.27 0.22 0.14 0.46 0.34 0.26 0.4 0.28 0.15 0.4 0.44 0.36 0.4
fractal_dimension_se -0.04 0.06 -0 -0.02 0.28 0.51 0.46 0.26 0.33 0.69 0.23 0.33 0.24 0.13 0.43 0.81 0.74 0.64 0.38
radius_worst 0.97 0.37 0.97 0.96 0.2 0.52 0.68 0.82 0.19 -0.25 0.72 -0.07 0.7 0.76 -0.21 0.18 0.16 0.32 -0.1 -0.04
texture_worst 0.3 0.91 0.31 0.29 0.06 0.26 0.31 0.3 0.11 -0.04 0.19 0.43 0.2 0.2 -0.08 0.15 0.1 0.09 -0.08 -0 0.37
perimeter_worst 0.96 0.38 0.97 0.96 0.23 0.58 0.72 0.85 0.22 -0.2 0.73 -0.06 0.73 0.76 -0.2 0.24 0.2 0.36 -0.07 -0 0.99 0.37
area_worst 0.94 0.36 0.94 0.96 0.2 0.5 0.67 0.81 0.18 -0.23 0.76 -0.05 0.73 0.81 -0.17 0.18 0.17 0.32 -0.09 -0.02 0.98 0.35 0.98
smoothness_worst 0.1 0.1 0.13 0.11 0.8 0.56 0.44 0.45 0.42 0.51 0.14 -0.01 0.13 0.12 0.33 0.21 0.15 0.19 0.01 0.16 0.2 0.25 0.22 0.2
compactness_worst 0.39 0.3 0.44 0.37 0.47 0.86 0.75 0.66 0.48 0.48 0.29 -0.04 0.34 0.28 -0.04 0.67 0.47 0.43 0.1 0.4 0.46 0.37 0.51 0.42 0.56
concavity_worst 0.51 0.32 0.55 0.5 0.43 0.81 0.88 0.74 0.44 0.37 0.38 -0.01 0.42 0.38 -0.03 0.63 0.65 0.52 0.08 0.39 0.56 0.38 0.6 0.53 0.51 0.89
concave points_worst 0.73 0.33 0.76 0.71 0.5 0.81 0.86 0.91 0.44 0.2 0.54 -0.05 0.56 0.54 -0.07 0.47 0.42 0.57 0.02 0.22 0.78 0.38 0.81 0.74 0.54 0.8 0.85
symmetry_worst 0.16 0.12 0.18 0.14 0.39 0.51 0.41 0.37 0.7 0.34 0.09 -0.09 0.11 0.07 -0.11 0.27 0.19 0.13 0.42 0.11 0.24 0.25 0.26 0.2 0.49 0.61 0.53 0.5
fractal_dimension_worst -0 0.13 0.04 -0.01 0.5 0.69 0.51 0.36 0.44 0.77 0.05 -0.01 0.08 0.01 0.11 0.59 0.44 0.31 0.1 0.59 0.09 0.23 0.13 0.07 0.61 0.82 0.69 0.51 0.53
radius_mean
texture_mean
perimeter_mean
area_mean
smoothness_mean
compactness_mean
concavity_mean
concave points_mean
symmetry_mean
fractal_dimension_mean
radius_se
texture_se
perimeter_se
area_se
smoothness_se
compactness_se
concavity_se
concave points_se
symmetry_se
fractal_dimension_se
radius_worst
texture_worst
perimeter_worst
area_worst
smoothness_worst
compactness_worst
concavity_worst
concave points_worst
symmetry_worst
fractal_dimension_worst
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

Thông qua heatmap ở trên, ta có thể rút ra các nhận xét sau:

- radius_mean đối với area-mean và perimeter_mean có sự tương quan cao lần lượt là 1 và 0.99 tức hai chỉ số này có liên quan mật thiết đến radius_mean. Bên cạnh đó, area-mean và perimeter_mean cũng có chỉ số tương quan là 0.99. Vậy ta có thể bỏ hai cột area-mean và perimeter_mean.
- Các chỉ số tương quan giữa compactness, concativity và concavepoints như đã nói Data Cleaning cũng có sự liên quan cao, cho nên ta cũng có thể áp dụng cách này để xóa các cột liên quan với nhau.
- Tiếp theo, các chỉ số liên quan se cũng có sự tương quan cao khi radius_es với area-mean và perimeter_mean là 0.95 và 0.97. Tương tự, ta bỏ luôn hai cột này, giữ lại radius_es.
- Cuối cùng, ta có thể dễ dàng nhận ra các ô màu đỏ tập trung các chỉ số worst đối với chỉ số mean. Tức là các trường này cũng có chỉ số tương quan khá cao. Bởi vì các cột như area và perimeter đều theo radius nên điều tương tự sẽ xảy ra với các cột worst. Bên cạnh đó radius_mean và radius_worst cũng tương quan cao nên ta chỉ cần giữ lại radius_mean là được.

➔ Chung quy lại, ta sẽ bỏ các cột liên quan đến perimeter và area(mean và es), các cột worst và cuối cùng là các cột compactness, concavity, và concave points.

```
removed_cols = ['radius_worst',
        'texture_worst',
        'perimeter_worst',
        'area_worst',
        'smoothness_worst',
        'compactness_worst',
        'concavity_worst',
        'concave points_worst',
        'symmetry_worst',
        'fractal_dimension_worst',
        'perimeter_mean',
        'perimeter_se',
        'area_mean',
        'area_se',
        'concavity_mean',
        'concavity_se',
        'concave points_mean',
        'concave points_se'
        ]
brcc_df = brcc_df.drop(removed_cols, axis=1)
```

Kết quả cuối cùng đạt được, ta chỉ còn lại 1 trường dữ liệu thật sự cần thiết cho việc xây dựng và huấn luyện mô hình.

	diagnosis	radius_mean	texture_mean	smoothness_mean	compactness_mean	symmetry_mean	fractal_dimension_mean	radius_se	texture_se	smoothness_s
0	M	17.99	10.38	0.11840	0.27760	0.2419	0.07871	1.0950	0.9053	0.00639
1	M	20.57	17.77	0.08474	0.07864	0.1812	0.05667	0.5435	0.7339	0.00522
2	M	19.69	21.25	0.10960	0.15990	0.2069	0.05999	0.7456	0.7869	0.00615
3	M	11.42	20.38	0.14250	0.28390	0.2597	0.09744	0.4956	1.1560	0.00911
4	M	20.29	14.34	0.10030	0.13280	0.1809	0.05883	0.7572	0.7813	0.01149
...	...	...	...	...	...	...	...	...	...	.
563	M	20.92	25.09	0.10990	0.22360	0.2149	0.06879	0.9622	1.0260	0.00639
564	M	21.56	22.39	0.11100	0.11590	0.1726	0.05623	1.1760	1.2560	0.01030
565	M	20.13	28.25	0.09780	0.10340	0.1752	0.05533	0.7655	2.4630	0.00576
566	M	16.60	28.08	0.08455	0.10230	0.1590	0.05648	0.4564	1.0750	0.00590
567	M	20.60	29.33	0.11780	0.27700	0.2397	0.07016	0.7260	1.5950	0.00652

556 rows × 13 columns

3. Huấn luyện mô hình:

Nhóm quyết định sử dụng Logistic Regression bởi vì mục đích của mô hình là dự đoán Yes hoặc No. Yes hoặc No ở đây chính là M(ác tính) hoặc B(lành tính) ở cột diagnosis, mô hình Logistic Regression đưa output trả ra về khoảng từ [0,1] từ hàm logit rất phù hợp cho mục đícht trên. Nếu kết quả output nào nhỏ hơn 0.5(điểm giữa của mô hình) thì được xem là M(ác tính) và ngược lại là B(lành tính)

3.1. Chia tập dữ liệu huấn luyện, kiểm thử

Ta có 4 tập dữ liệu chính:

- X_train: các trường tham số cho mục đích huấn luyện
- X_test: các trường tham số cho mục đích kiểm thử
- y_train: cột diagnosis với các dòng cho mục đích huấn luyện
- y_test: cột diagnosis với các dòng cho mục đích kiểm thử

Nhóm sử dụng thư viện statsmodels để thực hiện huấn luyện mô hình với hàm glm() (mô hình tuyến tính tổng quát, một loại mô hình bao gồm hồi quy logistic). Lý do sử dụng bởi vì nếu sử dụng trực tiếp LogisticRegression từ thư viện sklearn, các vấn đề overfitting có thể xảy ra. Glm thường trích xuất sự tuyến tính giữa các tham số đầu vào, tránh overfitting cho mô hình. Hơn nữa, Glm được xem là một mô hình tổng quát, việc áp dụng các mô hình khác ngoài Logistic cũng có thể xảy ra với sự tùy biến cao hơn.

Về vấn đề phân chia dữ liệu, ta sử dụng hàm train_test_split với test_size = 0.3 và random_state = 40.

```
X = brcc_df
y = brcc_df['diagnosis']
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=40)
```
0.1s

Ta sẽ chia tập dữ liệu theo tỷ lệ 7:3, 7 cho huấn luyện và 3 cho kiểm thử với số lượng khởi tạo là 40.

3.2. Thiết lập tinh chỉnh tham số

Ở phần chia tập huấn luyện, kiểm thử, random_state ở một vài docs hay trên mạng thường để 42. Tuy nhiên khi đặt random_state=40, nhóm lại phát hiện ra tỷ lệ chính xác lại cao hơn nên quyết định đặt là 40.
Khi thiết lập tham số cho mô hình bằng statsmodels, ta không cần làm gì nhiều vì các tinh chỉnh sẽ được thực hiện sẵn. Cuối cùng, ta sẽ có bảng report về mô hình sau đây:

Generalized Linear Model Regression Results

Dep. Variable:	['diagnosis[B]', 'diagnosis[M]']	No. Observations:	389
Model:	GLM	Df Residuals:	376
Model Family:	Binomial	Df Model:	12
Link Function:	Logit	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-53.666
Date:	Sat, 19 Mar 2022	Deviance:	107.33
Time:	19:24:19	Pearson chi2:	180.
No. Iterations:	9	Pseudo R-squ. (CS):	0.6483
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	45.3082	11.175	4.055	0.000	23.406	67.210
radius_mean	-1.1816	0.295	-4.001	0.000	-1.760	-0.603
texture_mean	-0.4975	0.092	-5.400	0.000	-0.678	-0.317
smoothness_mean	-73.5190	40.460	-1.817	0.069	-152.820	5.782
compactness_mean	-6.8423	21.318	-0.321	0.748	-48.625	34.941
symmetry_mean	-34.4258	15.219	-2.262	0.024	-64.255	-4.596
fractal_dimension_mean	-107.3911	118.839	-0.904	0.366	-340.311	125.528
radius_se	-7.0481	2.956	-2.384	0.017	-12.842	-1.254
texture_se	1.0402	0.769	1.353	0.176	-0.467	2.548
smoothness_se	19.1891	116.669	0.164	0.869	-209.478	247.857
compactness_se	-65.0364	49.172	-1.323	0.186	-161.412	31.339
symmetry_se	168.3310	60.257	2.794	0.005	50.229	286.433
fractal_dimension_se	699.0183	387.732	1.803	0.071	-60.922	1458.959

4. Báo cáo kết quả, so chính xác

Khi nhìn vào bảng report mô hình trên, ta có một vài vấn đề sau:
Ta cần tập trung vào giá trị P>|z|, theo lý thuyết thống kê, khi giá trị P càng lớn(cụ thể là vượt ngưỡng 0.05), độ tương quan giữa các trường với mô hình càng thấp hay gọi là giả thuyết vô hiệu(null-hypothesis). Việc bỏ các trường này sẽ không ảnh hưởng đến kết quả mô hình cũng như tỷ lệ chính xác. Ở đây ta cho ngưỡng là 0.1 để quan sát trước mô hình sau khi biến đổi sẽ xảy ra như thế nào.

Generalized Linear Model Regression Results

Dep. Variable:	['diagnosis[B]', 'diagnosis[M]']	No. Observations:	389
Model:	GLM	Df Residuals:	379
Model Family:	Binomial	Df Model:	9
Link Function:	Logit	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-55.262
Date:	Sat, 19 Mar 2022	Deviance:	110.52
Time:	19:53:16	Pearson chi2:	197.
No. Iterations:	9	Pseudo R-squ. (CS):	0.6454
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	40.4312	6.242	6.477	0.000	28.197	52.666
radius_mean	-1.0518	0.207	-5.077	0.000	-1.458	-0.646
texture_mean	-0.5059	0.088	-5.767	0.000	-0.678	-0.334
smoothness_mean	-109.1463	28.007	-3.897	0.000	-164.039	-54.254
symmetry_mean	-37.7359	14.713	-2.565	0.010	-66.572	-8.900
radius_se	-6.9743	2.939	-2.373	0.018	-12.735	-1.213
texture_se	1.4898	0.703	2.119	0.034	0.112	2.868
compactness_se	-83.8935	34.526	-2.430	0.015	-151.564	-16.223
symmetry_se	181.5882	56.701	3.203	0.001	70.455	292.721
fractal_dimension_se	597.1621	320.190	1.865	0.062	-30.400	1224.724

Vậy các giá trị P sau khi thay đổi tập dữ liệu đã trở nên tốt hơn khi đều dưới 0.1 và rất nhiều biến nhỏ hơn 0.05 chứng tỏ mô hình đã tốt hơn trước rất nhiều.

Tiếp theo, ta trực quan mô hình cũng như tỷ lệ chính xác sau khi dự đoán kết quả thông quá tập test cho trước bằng classification_report và confusion_matrix.

```
              precision    recall  f1-score   support

           B      0.925     0.990     0.957       100
           M      0.983     0.881     0.929        67

    accuracy                          0.946       167
   macro avg      0.954     0.935     0.943       167
weighted avg      0.949     0.946     0.946       167

Confusion Matrix:
 [[99  1]
 [ 8 59]]

Accuracy Score:  94.6 %
```

Nhận xét:

Thông qua confusion_matrix, ta có thể hiểu thêm về các thông số của dữ liệu, ta có:

- True positives(dương đúng) : 99
- True negatives(dương giả) : 59
- False positives(âm đúng) : 8
- False negatives(âm giả): 1

Ta có 100 bệnh nhân B(lành tính) và 67 bênh nhân M(ác tính)(sum=167). Với từng loại bệnh nhân ta có từng chỉ số mô hình khác nhau. Nhìn chung cả hai loại đều có tỷ lệ cao, hơn 90%(tuy tỷ lệ recall của M chỉ có 88% nhưng cũng được xem là cao).
Tóm gọn lại ở accuracy score, ta áp dụng công thức:

$$\text{Accuracy score} = \frac{TN + TP}{Sum} = \frac{99 + 59}{167} = 94.6\%$$