

DỰ ĐOÁN ĐỘT QUY BẰNG CÁC KỸ THUẬT HỌC MÁY

Khổng Đức Quang - MSSV: 20225072

Ngày 10 tháng 12 năm 2024

1 Tóm tắt

Đột quy là một **tình trạng y tế khẩn cấp** xảy ra khi dòng máu đến một phần não bị gián đoạn do xuất huyết hoặc tắc nghẽn bởi cục máu đông. Đây là nguyên nhân tử vong đứng thứ hai trên toàn cầu, với khoảng **5,5 triệu ca tử vong** mỗi năm. Theo thống kê, mỗi năm có hơn **15 triệu người** trên thế giới bị ảnh hưởng bởi đột quy, và *trung bình cứ mỗi 4 phút lại có một trường hợp tử vong* do tình trạng này.

Phần lớn các trường hợp đột quy có liên quan chặt chẽ đến lối sống không lành mạnh, dẫn đến ước tính khoảng 80% các trường hợp có thể được phòng ngừa. Do đó, việc xây dựng các mô hình dự đoán nguy cơ đột quy có ý nghĩa quan trọng trong việc ngăn chặn các tổn thương nghiêm trọng và giảm thiểu tỷ lệ tử vong liên quan. Công tác dự đoán không chỉ giúp cảnh báo sớm mà còn hỗ trợ trong việc đưa ra các biện pháp can thiệp kịp thời nhằm bảo vệ sức khỏe cộng đồng.

2 Mục tiêu dự án

Mục tiêu của dự án này là dự đoán khả năng xảy ra đột quy não bằng cách ứng dụng các kỹ thuật học máy. Bằng cách phân tích dữ liệu y tế, em sẽ huấn luyện một số mô hình học máy nhằm nhận diện các mẫu và yếu tố rủi ro liên quan đến đột quy. Điều này sẽ hỗ trợ phát hiện sớm, cung cấp những thông tin quan trọng giúp đưa ra các biện pháp phòng ngừa và can thiệp kịp thời.

Mục lục

1	Tóm tắt	1
2	Mục tiêu dự án	1
3	Mô tả bộ dữ liệu	4
4	Tiền xử lý dữ liệu	5
5	Phân tích dữ liệu	6
6	Xem xét sự mất cân bằng dữ liệu	12
7	Huấn luyện mô hình	14
7.1	LogisticRegression	14
7.2	Decision Tree Classifier	16
7.3	K Nearest Neighbor	18
7.4	Random Forest Classifier	21
7.5	Support Vector Machine	23
8	Đánh giá mô hình	25
9	Kết luận	27

Danh sách hình vẽ

1	Kiểm tra giá trị NULL	5
2	Correlation Heatmap	6
3	Biểu đồ gender	7
4	Biểu đồ residencetype_type	7
5	Biểu đồ ever_married	8
6	Biểu đồ work_type	8
7	Biểu đồ smoking_status	9
8	Biểu đồ hypertension	9
9	Biểu đồ heart_disease	10
10	Biểu đồ stroke liên hệ age và stroke	10
11	Biểu đồ liên hệ average_glucose_level	11
12	Biểu đồ liên hệ bmi và stroke	11
13	Biểu đồ so sánh số lượng người đột quỵ và không đột quỵ	12
14	Phân bố tỷ lệ người có và không đột quỵ	13
15	Xử lý dữ liệu mất cân bằng	14
16	14
17	15
18	15
19	Đường cong ROC - Logistic Regression	16
20	16
21	17
22	Đường cong ROC - Decision Tree Classifier	18
23	18
24	19
25	Đường cong ROC - K Nearest Neighbor	20
26	21
27	22
28	Đường cong ROC - Random Forest Classifier	23
29	23
30	24
31	Đường cong ROC - Support Vector Machine	25
32	Tổng hợp AUC	25
33	So sánh các đường cong ROC	26

3 Mô tả bộ dữ liệu

Bộ dữ liệu được em thu thập từ trang Web Kaggle để ước tính xem bệnh nhân có khả năng đột quỵ hay không. Đây là bộ dữ liệu về thông tin của 5110 người bao gồm 11 thuộc tính và 1 cột stroke(output) là có khả năng đột quỵ hay không. (FEDESORIANO 2021).

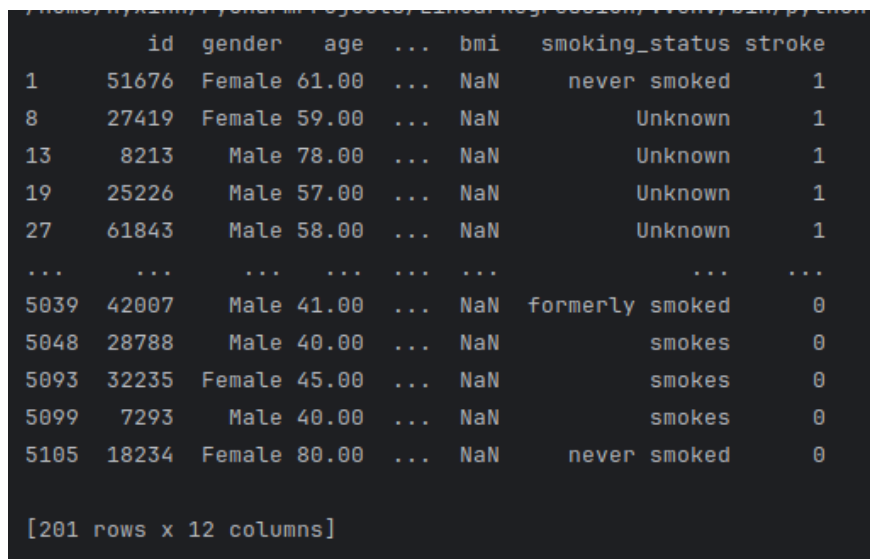
Dưới đây là danh sách các thuộc tính:

1. **Id (Integer Feature)**: Đây là dữ liệu kiểu số nhằm. Tuy nhiên thuộc tính này không làm ảnh hưởng tới Output nên em sẽ không phân tích thêm.
2. **Gender (Nominal Feature)**: Đây là thuộc tính kiểu chữ bao gồm các giá trị: Male, Female, Other. Thuộc tính "Gender"(giới tính) có thể ảnh hưởng tới khả năng đột quỵ do sự khác biệt sinh học, nội tiết tố, và yếu tố hành vi giữa nam và nữ.
3. **Age (Integer Feature)**: Dữ liệu kiểu số. Thuộc tính này là một yếu tố quan trọng ảnh hưởng đến khả năng đột quỵ.
4. **Hypertension (Integer Feature)**: Với hai giá trị khác nhau là: 0, 1. Đây là yếu tố nguy cơ lớn nhất đối với đột quỵ. Nó có ảnh hưởng sâu sắc đến khả năng đột quỵ.
5. **Heart Disease (Integer Feature)**: Với hai giá trị khác nhau là: 0, 1. Là một yếu tố có nguy cơ đáng kể gây đột quỵ.
6. **Ever married (Boolean Feature)**: Với hai giá trị khác nhau là: True, False. Có thể liên quan đến nguy cơ đột quỵ qua các yếu tố gián tiếp, chẳng hạn như lối sống, sức khỏe tâm lý, và sự hỗ trợ xã hội.
7. **Work type (Nominal Feature)**: Có 5 loại giá trị khác nhau: Private, Self-employed, children, Govt-job, Never-worked. Và ảnh hưởng đáng kể đến nguy cơ đột quỵ thông qua các yếu tố như mức độ căng thẳng, hoạt động thể chất, và tiếp xúc với các yếu tố nguy cơ môi trường.
8. **Residence type (Nominal Feature)**: Có hai giá trị khác nhau là: Urban, Rural. Ảnh hưởng đến nguy cơ đột quỵ thông qua các yếu tố môi trường, điều kiện sống, và khả năng tiếp cận dịch vụ chăm sóc sức khỏe.
9. **Avg glucose level (Float Feature)**: Là một yếu tố quan trọng liên quan đến nguy cơ đột quỵ, đặc biệt thông qua mối liên hệ với bệnh tiểu đường và rối loạn chuyển hóa. Mức đường huyết bất thường, cả cao lẫn thấp, đều có thể làm tăng nguy cơ đột quỵ.
10. **BMI (Float Feature)**: Là một chỉ số quan trọng để đánh giá mức độ béo phì hoặc thừa cân của một người, và nó có mối liên hệ mạnh mẽ với nguy cơ đột quỵ.
11. **Smoking status(Nominal Feature)**: Với 4 giá trị khác nhau: Never smoked, Unknown, formerly smoked, smokes. Là một yếu tố nguy cơ quan trọng đối với nhiều bệnh lý, bao gồm đột quỵ. Hút thuốc lá ảnh hưởng đến sức khỏe của tim và mạch máu, gây ra những tác động tiêu cực trực tiếp làm tăng nguy cơ đột quỵ.

4 Tiền xử lý dữ liệu

Trong quá trình tiền xử lý số liệu, em nhận thấy có các vấn đề sau cần xử lý:

1. Đầu tiên em **loại bỏ đi cột ID** do cột này không ảnh hưởng tới khả năng đột quy.
2. **Xử lý các giá trị NULL**



```

7 name/ny/nam/ny/ gender/male/female/age/age/ bmi/ bmi/ smoking_status/ stroke/ stroke/
1      51676  Female  61.00  ...  NaN      never smoked      1
8      27419  Female  59.00  ...  NaN      Unknown      1
13     8213   Male   78.00  ...  NaN      Unknown      1
19    25226   Male   57.00  ...  NaN      Unknown      1
27    61843   Male   58.00  ...  NaN      Unknown      1
...     ...     ...     ...  ...  ...     ...     ...
5039  42007   Male   41.00  ...  NaN  formerly smoked      0
5048  28788   Male   40.00  ...  NaN      smokes      0
5093  32235  Female   45.00  ...  NaN      smokes      0
5099   7293   Male   40.00  ...  NaN      smokes      0
5105  18234  Female   80.00  ...  NaN      never smoked      0

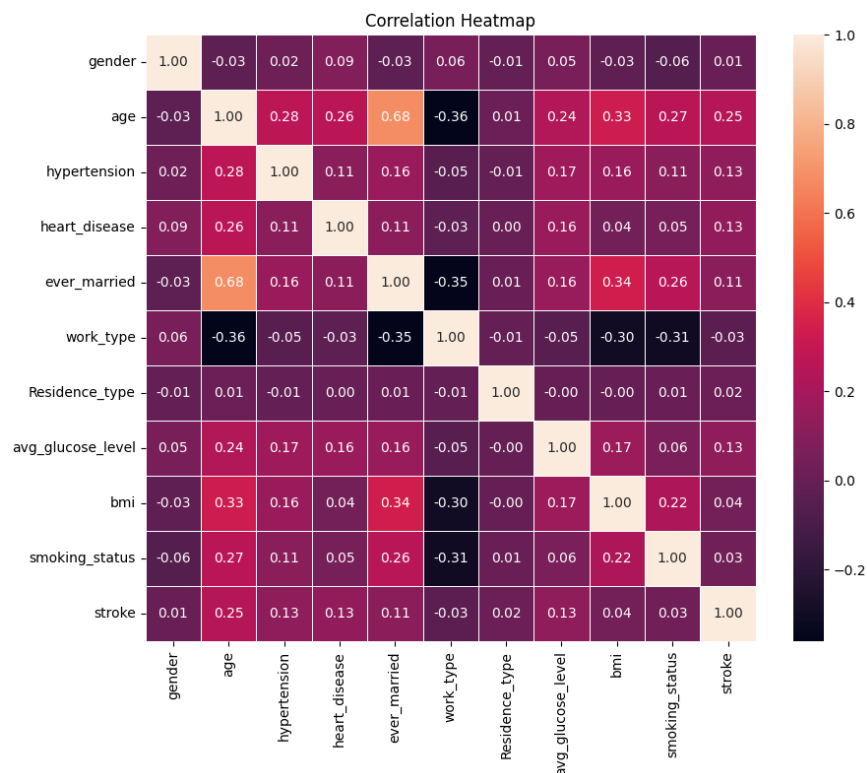
[201 rows x 12 columns]
```

Hình 1: Kiểm tra giá trị NULL

Sau khi kiểm tra em nhận thấy cột thuộc tính "bmi" có 201 giá trị NULL. Để xử lý vấn đề này có ba phương pháp phổ biến: Xóa các dòng chứa giá trị NULL, Điền giá trị trung bình hoặc trung vị, Điền giá trị dựa trên mô hình (Model-based Imputation). Sau khi tìm hiểu ưu và nhược điểm của từng phương pháp em quyết định lựa chọn cách điền giá trị trung bình là lựa chọn vừa đơn giản vừa hiệu quả, giúp giữ được lại toàn bộ dữ liệu mà không làm giảm kích thước bộ dữ liệu.

3. Bước tiếp theo là **chuyển đổi dữ liệu kiểu Categorical thành dữ liệu kiểu Numerical**. Phương pháp Hash e Encoding và One-hot Encoding được sử dụng để thực hiện các bước chuyển đổi này.
4. **Phân chia dữ liệu**: Em chia bộ dữ liệu thành hai tập là: tập huấn luyện(80%), tập kiểm tra(20%) bằng cách sử dụng hàm `train_test_split` được cung cấp trong thư viện `scikit-learn`
5. **Feature scaling (chuẩn hóa đặc trưng)** là bước cuối cùng

5 Phân tích dữ liệu



Hình 2: Correlation Heatmap

Nhận xét:

- Một số tương quan dương mạnh:

+) Age và ever_married (0.68) có tương quan dương mạnh nhất

+) Age cho thấy mối tương quan tích cực vừa phải với BMI (0.33), Hypertension (0.28) và smoking_status (0.27)

- Một số tương quan yếu đáng chú ý:

+) Work_type có một số tương quan yếu với: age (-0.36), ever_married (-0.35), bmi (-0.30), smoking_status (-0.31)

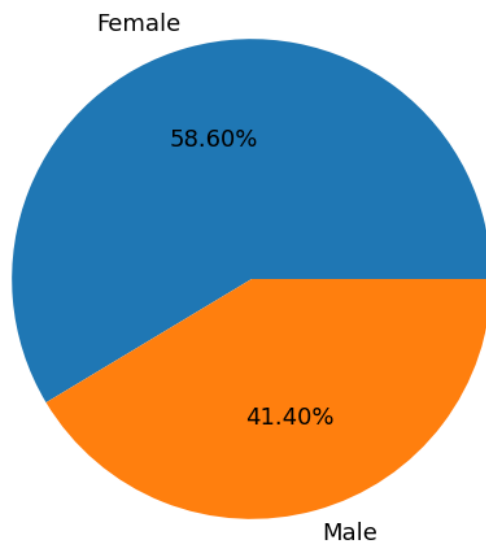
- Tương quan đột quỵ:

+) Stroke có tương quan yếu hoặc trung bình yếu với hầu hết các thuộc tính trong bảng.

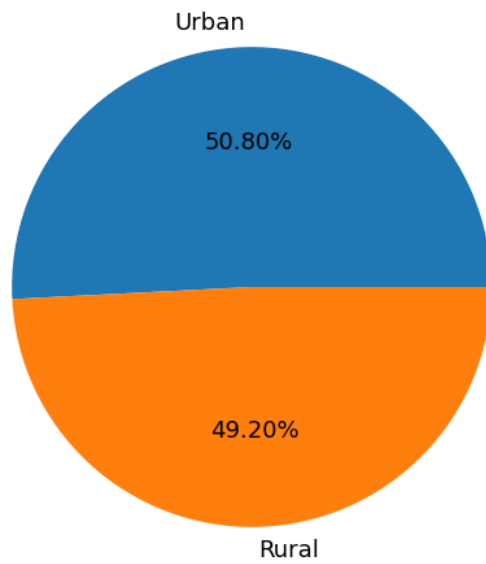
+) Age có tương quan cao nhất nhưng vẫn là tương quan yếu với Stroke (0.25), tương tự là heart_disease và hypertension, cả hai đều là 0.13

- Hầu hết các tương quan đều là yếu và rất yếu, gần giá trị 0

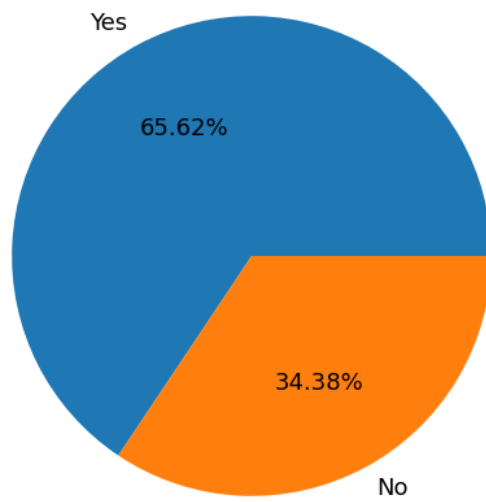
Tóm lại, age là yếu tố có ảnh hưởng lớn nhất đến khả năng đột quỵ.



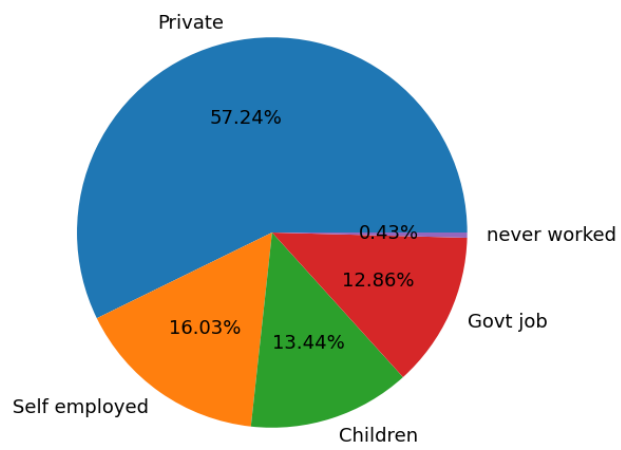
Hình 3: Biểu đồ gender



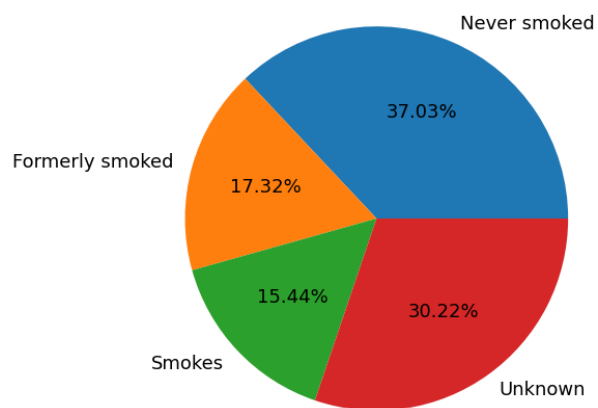
Hình 4: Biểu đồ residencetype_type



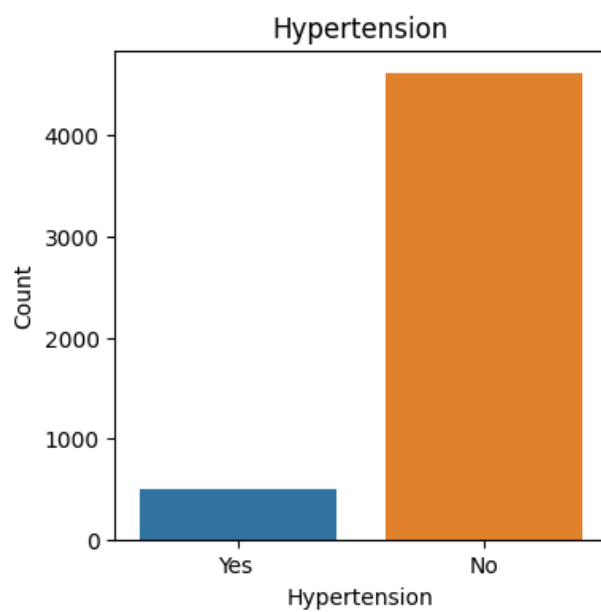
Hình 5: Biểu đồ ever_married



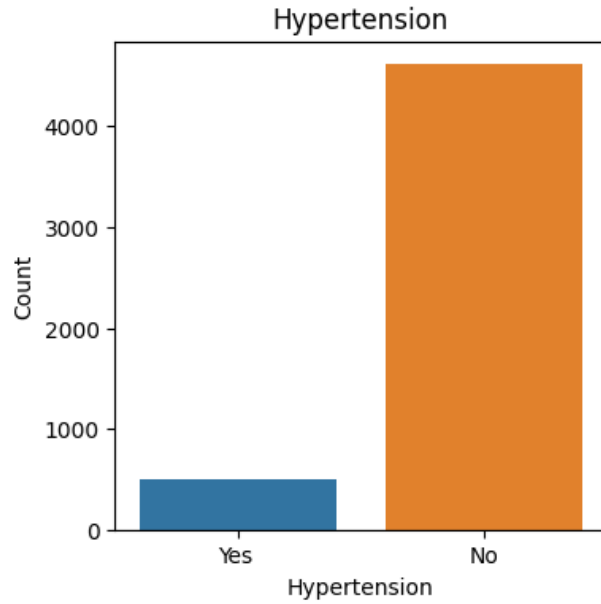
Hình 6: Biểu đồ work_type



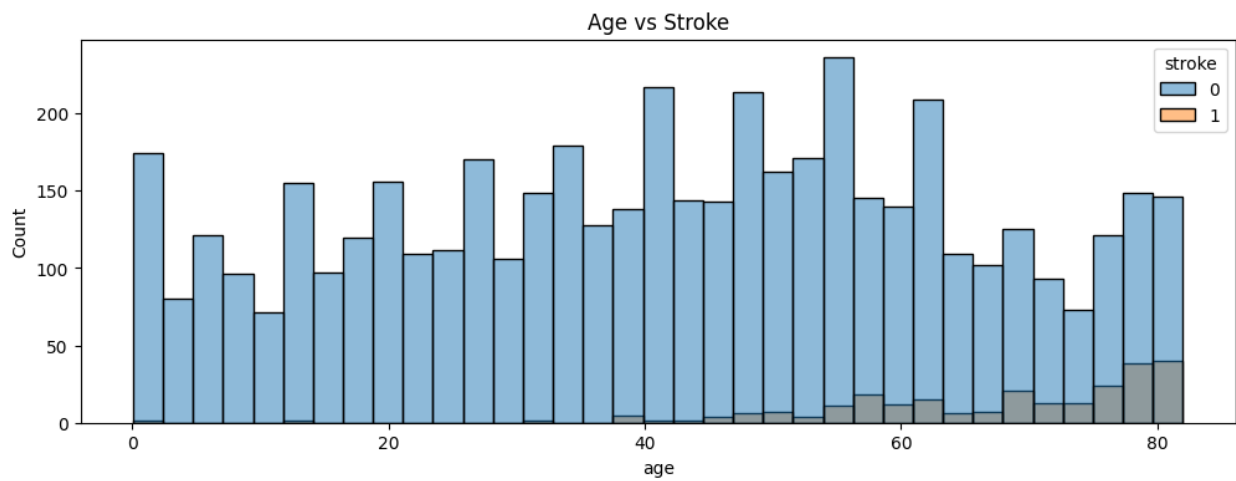
Hình 7: Biểu đồ smoking_status



Hình 8: Biểu đồ hypertension



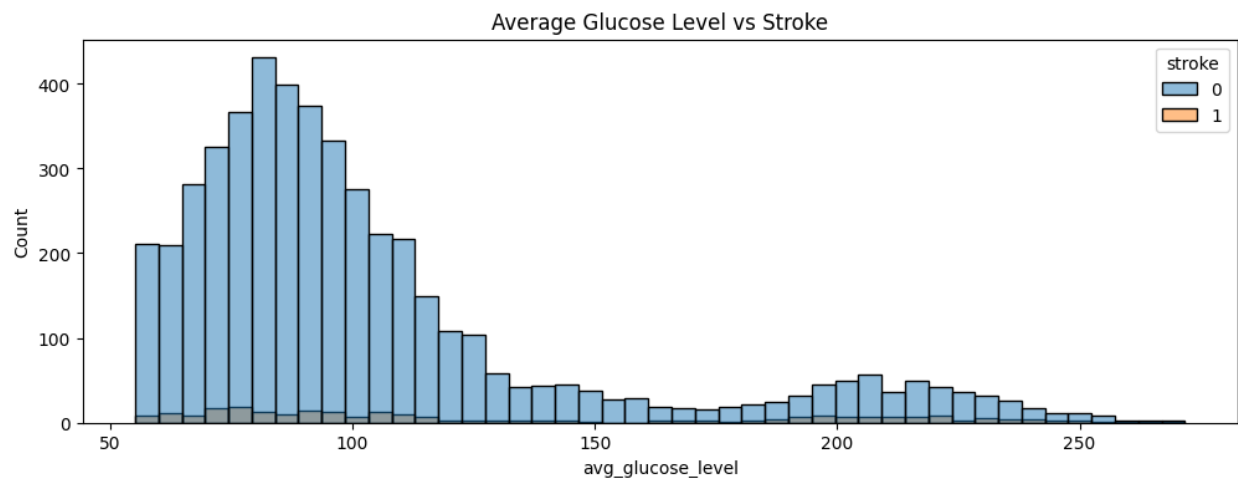
Hình 9: Biểu đồ heart_disease



Hình 10: Biểu đồstroke liên hệ age và stroke

Age và Stroke:

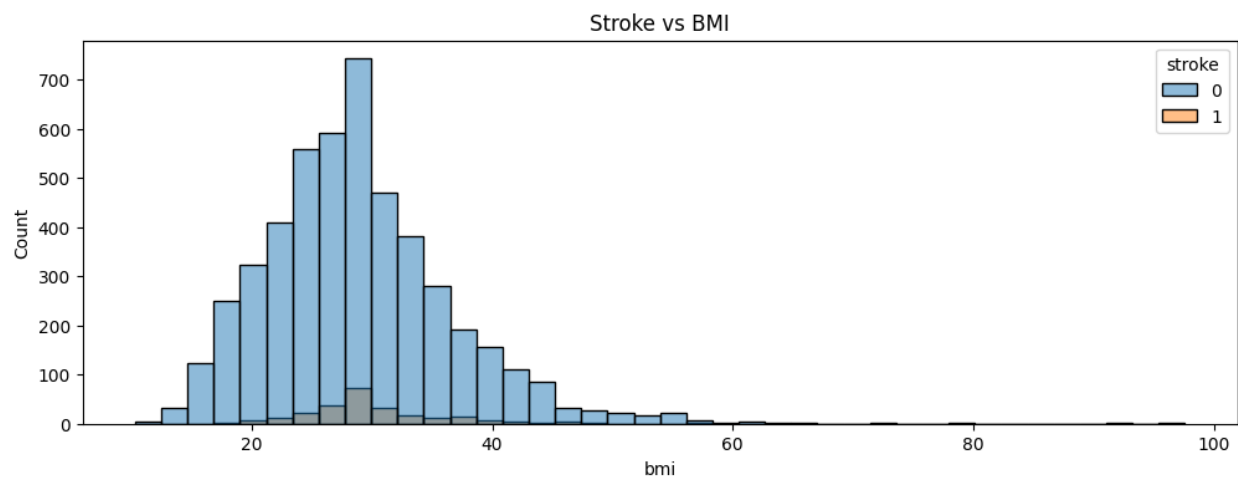
1. Các trường hợp đột quỵ tăng dần sau 40 tuổi.
2. Tỷ lệ đột quỵ cao nhất nằm trong độ tuổi từ 60-80 tuổi.
3. Có rất ít trường hợp đột quỵ dưới 40 tuổi.
4. Phân bố cho thấy tuổi tác là yếu tố có ảnh hưởng đáng kể đối với đột quỵ.



Hình 11: Biểu đồ liên hệ average_glucose_level

Average_glucose_level và StrokeS:

1. Các trường hợp đột quỵ có xu hướng xuất hiện nhiều hơn ở các mức đường huyết trung bình cao hơn, đặc biệt từ 120 mg/dL trở lên.
2. Trong phạm vi đường huyết từ 180 mg/dL đến 250 mg/dL, tỷ lệ người bị đột quỵ tăng lên đáng kể, dù số lượng tổng thể thấp hơn.
3. Ở mức đường huyết dưới 100 mg/dL, tỷ lệ đột quỵ thấp hơn rõ rệt.



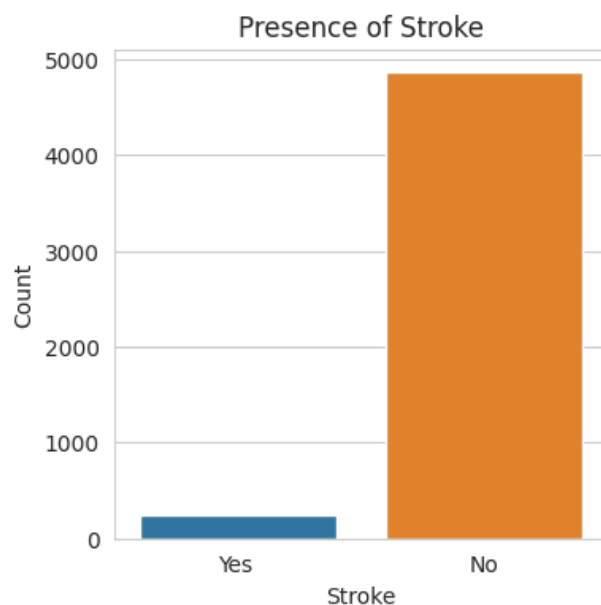
Hình 12: Biểu đồ liên hệ bmi và stroke

BMI và Stroke:

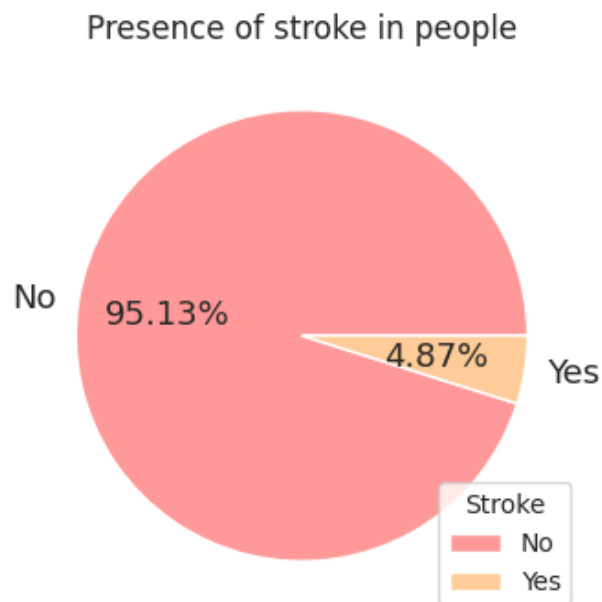
1. Phần lớn mọi người có chỉ số BMI từ 20-40.
2. Phân bố đỉnh nằm trong khoảng 25-35.

3. Các trường hợp đột quỵ (màu xám) xuất hiện thường xuyên hơn ở nhóm thừa cân và béo phì ($BMI > 25$).
4. Dữ liệu cho thấy tình trạng thừa cân hoặc béo phì có thể làm tăng nguy cơ đột quỵ, mặc dù mối quan hệ này có vẻ ít nghiêm trọng hơn so với tuổi tác hoặc lượng đường trong máu.

6 Xem xét sự mất cân bằng dữ liệu



Hình 13: Biểu đồ so sánh số lượng người đột quỵ và không đột quỵ



Hình 14: Phân bố tỷ lệ người có và không đột quỵ

Nhận xét:

1. Mất cân bằng dữ liệu nghiêm trọng. Tỷ lệ người không bị đột quỵ (No) *chiếm 95.13%* tổng số dữ liệu. Trong khi đó, tỷ lệ người bị đột quỵ (Yes) *chỉ chiếm 4.87%*. Điều này cho thấy dữ liệu mất cân bằng nghiêm trọng giữa hai nhóm.
2. Việc mất cân bằng dữ liệu này có thể dẫn đến việc mô hình học máy thiên vị, tập trung vào việc dự đoán chính xác lớp chiếm đa số (No) mà bỏ qua lớp thiểu số (Yes). **Từ đó dẫn đến kết quả** mô hình có thể đạt độ chính xác cao nhưng không phát hiện chính xác các trường hợp bị đột quỵ, làm tăng tỷ lệ False Negative (bỏ sót những người thực sự bị đột quỵ).

Giải pháp:

Để giải quyết vấn đề trên cũng như cải thiện khả năng dự đoán của mô hình, em sẽ áp dụng kỹ thuật xử lý dữ liệu mất cân bằng đó là Oversampling: Tăng số lượng mẫu cho lớp thiểu số bằng cách sử dụng SMOTE.

```
data1 = data.copy()
X = data.drop('stroke', axis = 1)
Y = data['stroke']
Y = pd.DataFrame(Y)
[133] ✓ 0.0s Python

smote = SMOTE(random_state = 10)
X1, Y1 = smote.fit_resample(X, Y)
[148] ✓ 0.0s Python

Y1 = pd.DataFrame(Y1)
X1 = pd.DataFrame(X1)
Y1.value_counts()
[149] ✓ 0.0s Python

... stroke
0      4860
1      4860
Name: count, dtype: int64
```

Hình 15: Xử lý dữ liệu mất cân bằng

Sau khi sử dụng kỹ thuật SMOTE để xử lý sự mất cân bằng của dữ liệu thì số trường hợp đột quỵ và không đột quỵ đã bằng nhau như kết quả trên hình.

7 Huấn luyện mô hình

7.1 LogisticRegression

```
LR = LogisticRegression()
LR.fit(X_train, Y_train)
Y_pre = LR.predict(X_test)
LR_Accuracy = accuracy_score(Y_test, Y_pre)
print(LR_Accuracy*100, '%')
✓ 0.1s Python
81.6872427983539 %
```

Hình 16:

```
CM = confusion_matrix(Y_test, Y_pre)
print(CM)
```

✓ 0.0s Python

```
[[783 190]
 [166 805]]
```

print(classification_report(Y_test, Y_pre))
print("specificity =", CM[0][0]/(CM[0][0]+CM[0][1]))

✓ 0.0s Python

	precision	recall	f1-score	support
0	0.83	0.80	0.81	973
1	0.81	0.83	0.82	971
accuracy			0.82	1944
macro avg	0.82	0.82	0.82	1944
weighted avg	0.82	0.82	0.82	1944

specificity = 0.8047276464542652

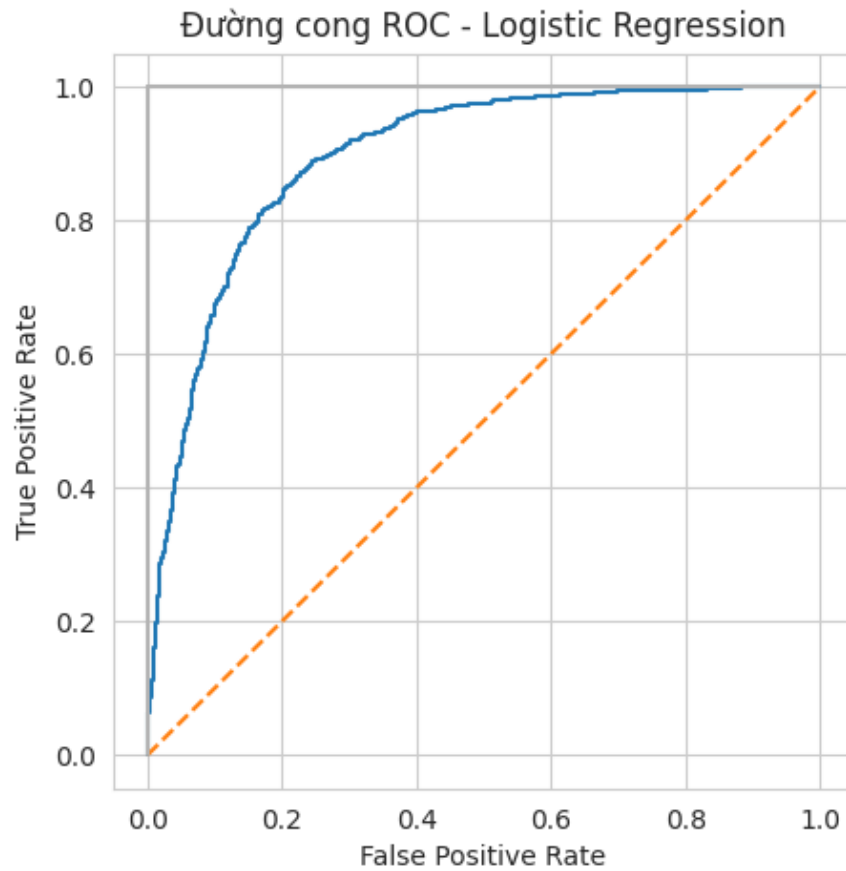
Hình 17:

```
print('roc_auc_score for Logistic Regression: ', roc_auc_score(Y_test, y))
```

✓ 0.0s Python

roc_auc_score for Logistic Regression: 0.8949896431244

Hình 18:



Hình 19: Đường cong ROC - Logistic Regression

7.2 Decision Tree Classifier

```
DTC = DecisionTreeClassifier()
DTC.fit(X_train,Y_train)
```

Python

```
Y_pred_DTC = DTC.predict(X_test)
DTC_Accuracy = accuracy_score(Y_test, Y_pred_DTC)
print(DTC_Accuracy*100,'%')
```

✓ 0.0s Python

90.74074074074075 %

```
CM = confusion_matrix(Y_test, Y_pred_DTC)
print(CM)
```

✓ 0.0s Python

```
[[870 103]
 [ 77 894]]
```

Hình 20:


```
print(classification_report(Y_test, Y_pred_DTC))
```

✓ 0.0s Python

	precision	recall	f1-score	support
0	0.92	0.89	0.91	973
1	0.90	0.92	0.91	971
accuracy			0.91	1944
macro avg	0.91	0.91	0.91	1944
weighted avg	0.91	0.91	0.91	1944

+ Code + Markdown

```
print("specificity =", CM[0][0]/(CM[0][0]+CM[0][1]))
```

✓ 0.0s Python

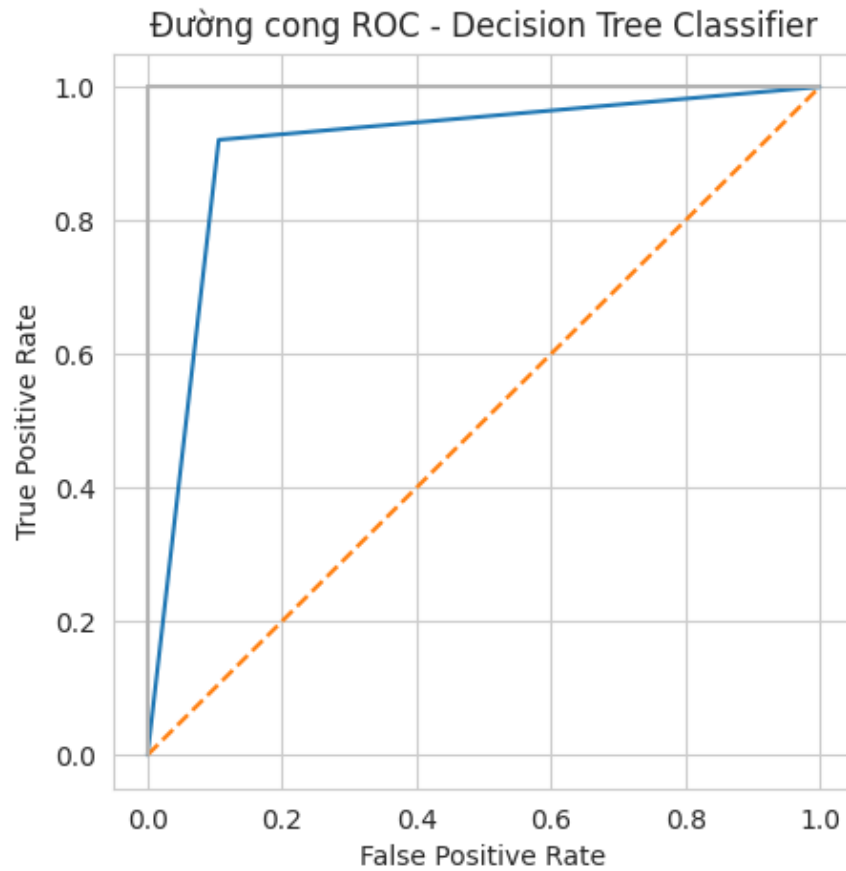
specificity = 0.894141829393628

```
y_score_DTC = DTC.predict_proba(X_test)[:,-1]
false_positive_rate3, true_positive_rate3, threshold3 = roc_curve(Y_test, y_score_DTC)
print('roc_auc_score for Decision Tree Classifier: ', roc_auc_score(Y_test, y_score_DTC))
```

✓ 0.0s Python

roc_auc_score for Decision Tree Classifier: 0.9074210691767315

Hình 21:



Hình 22: Đường cong ROC - Decision Tree Classifier

7.3 K Nearest Neighbor

```
KNN_clf = KNeighborsClassifier()
KNN_clf.fit(X_train,Y_train)
Y_Res_KNN=KNN_clf.predict(X_test)
KNN_Accuracy = accuracy_score(Y_test, Y_Res_KNN)
print(KNN_Accuracy*100,'%')
```

✓ 0.1s Python

89.0432098765432 %
[/home/hyxin/venv/lib/python3.10/site-packages/sklearn/neighbors/_classific](#)
 return self._fit(X, y)

```
CM = confusion_matrix(Y_test, Y_Res_KNN)
print(CM)
```

✓ 0.0s Python

```
[[783 190]
 [ 23 948]]
```

Hình 23:

```

print("True Positives : " , CM[1][1])
print("True Negatives : " , CM[0][0])
print("False Positives : " , CM[0][1])
print("False Negatives : " , CM[1][0])
print(classification_report(Y_test, Y_Res_KNN))

```

✓ 0.0s Python

True Positives : 948
True Negatives : 783
False Positives : 190
False Negatives : 23

	precision	recall	f1-score	support
0	0.97	0.80	0.88	973
1	0.83	0.98	0.90	971
accuracy			0.89	1944
macro avg	0.90	0.89	0.89	1944
weighted avg	0.90	0.89	0.89	1944

+ Code + Markdown

Add Markdown Cell

```

print("specificity =", CM[0][0]/(CM[0][0]+CM[0][1]))

```

✓ 0.0s Python

specificity = 0.8047276464542652

```

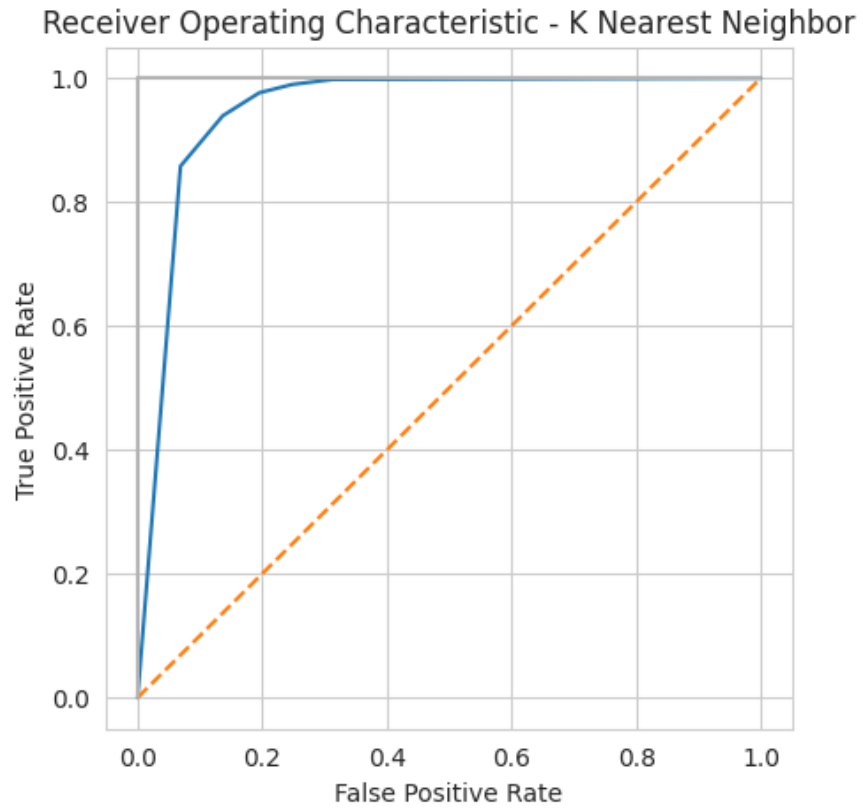
clf.predict_proba(X_test)[:,1]
e4, true_positive_rate4, threshold4 = roc_curve(Y_test, y_score_KNN)
re for K Nearest Neighbour is: ', roc_auc_score(Y_test, y_score_KNN))

```

✓ 0.0s Python

roc_auc_score for K Nearest Neighbour is: 0.9493185207608519

Hình 24:



Hình 25: Đường cong ROC - K Nearest Neighbor

7.4 Random Forest Classifier

```
RANDOM FOREST CLASSIFIER

RF = RandomForestClassifier(n_estimators=1000, random_state=47, n_jobs
RF.fit(X_train, Y_train)
Y_Res_RF=RF.predict(X_test)
RF_Accuracy = accuracy_score(Y_test, Y_Res_RF)
print(RF_Accuracy*100, '%')

✓ 6.8s Python

/home/hyxin/venv/lib/python3.10/site-packages/sklearn/base.py:1473: DataC
return fit_method(estimator, *args, **kwargs)
93.77572016460906 %

CM = confusion_matrix(Y_test, Y_Res_RF)
print(CM)
print("True Positives : ", CM[1][1])
print("True Negatives : ", CM[0][0])
print("False Positives : ", CM[0][1])
print("False Negatives : ", CM[1][0])

✓ 0.0s Python

[[901  72]
 [ 49 922]]
True Positives : 922
True Negatives : 901
False Positives : 72
False Negatives : 49
```

Hình 26:

```
print(classification_report(Y_test, Y_Res_RF))
print("specificity =", CM[0][0]/(CM[0][0]+CM[0][1]))
```

✓ 0.0s Python

	precision	recall	f1-score	support
0	0.95	0.93	0.94	973
1	0.93	0.95	0.94	971
accuracy			0.94	1944
macro avg	0.94	0.94	0.94	1944
weighted avg	0.94	0.94	0.94	1944

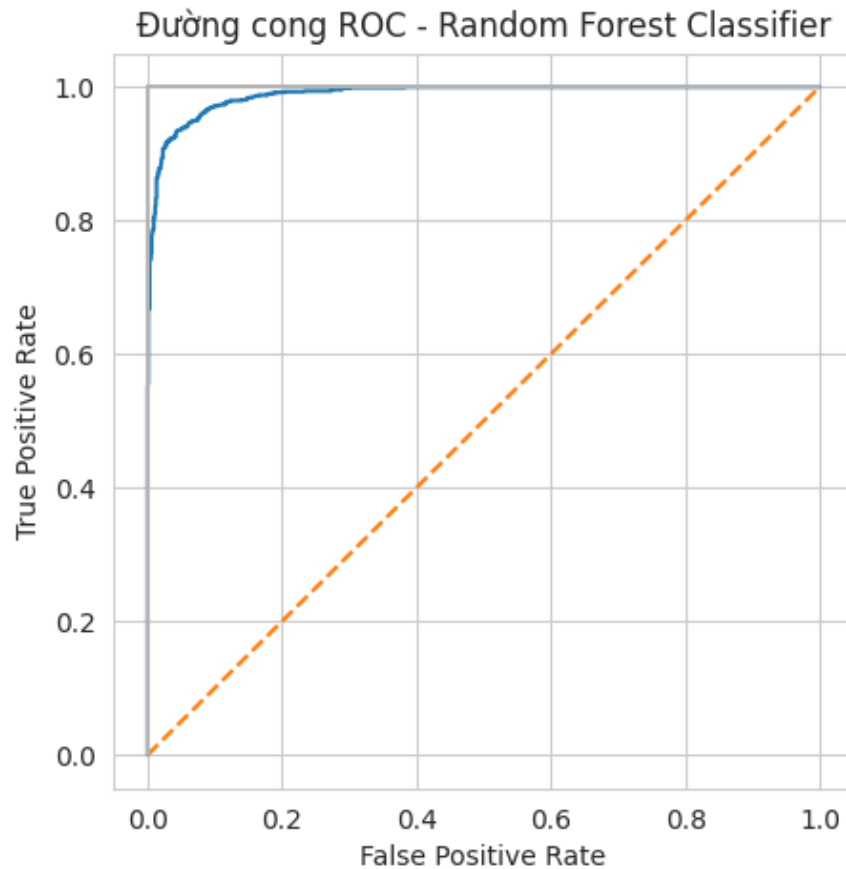
specificity = 0.9260020554984584

```
y_score_RF = RF.predict_proba(X_test)[:,-1]
false_positive_rate7, true_positive_rate7, threshold7 = roc_curve(Y_test, y_score_RF)
print('roc_auc_score for Random Forest Classifier is: ', roc_auc_score(Y_test, y_score_RF))
```

✓ 0.5s Python

roc_auc_score for Random Forest Classifier is: 0.9887228072478018

Hình 27:



Hình 28: Đường cong ROC - Random Forest Classifier

7.5 Support Vector Machine

```
SVC_clf = SVC(kernel='rbf', probability= True)
SVC_clf.fit(X_train, Y_train)
Y_res_SVC = SVC_clf.predict(X_test)
SVC_Accuracy = accuracy_score(Y_test, Y_res_SVC)
print(SVC_Accuracy*100, '%')
```

✓ 11.9s Python

[/home/hyxdnh/venv/lib/python3.10/site-packages/sklearn/utils/validation.py:](/home/hyxdnh/venv/lib/python3.10/site-packages/sklearn/utils/validation.py)
y = column_or_1d(y, warn=True)
78.5493827160494 %

```
CM = confusion_matrix(Y_test, Y_res_SVC)
print(CM)
```

✓ 0.0s Python

```
[[715 258]
 [159 812]]
```

Hình 29:

```
print("True Positives : " , CM[1][1])
print("True Negatives : " , CM[0][0])
print("False Positives : " , CM[0][1])
print("False Negatives : " , CM[1][0])
print(classification_report(Y_test, Y_res_SVC))
```

✓ 0.0s Python

True Positives : 812
True Negatives : 715
False Positives : 258
False Negatives : 159

	precision	recall	f1-score	support
0	0.82	0.73	0.77	973
1	0.76	0.84	0.80	971
accuracy			0.79	1944
macro avg	0.79	0.79	0.78	1944
weighted avg	0.79	0.79	0.78	1944

```
print("specificity =", CM[0][0]/(CM[0][0]+CM[0][1]))
```

✓ 0.0s Python

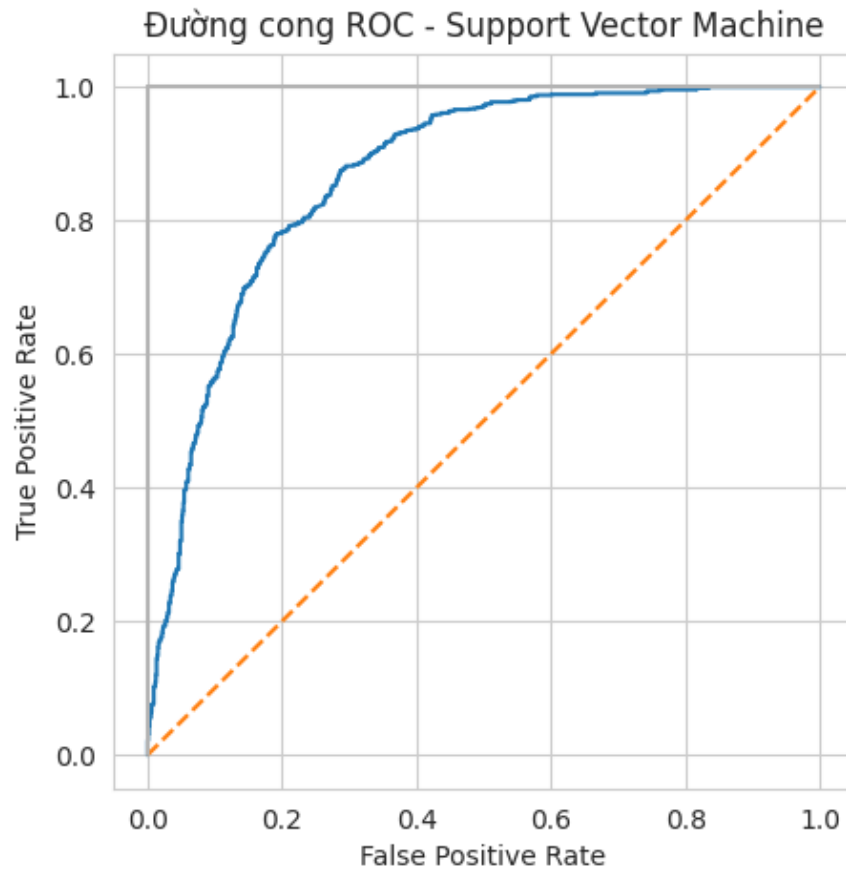
specificity = 0.7348406988694759

```
y_score_SVC = SVC_clf.predict_proba(X_test)[:,:1]
false_positive_rate4, true_positive_rate4, threshold4 = roc_curve(Y_test, y_score_SVC)
print('roc_auc_score for SVC is: ', roc_auc_score(Y_test, y_score_SVC))
```

✓ 0.4s Python

roc_auc_score for SVC is: 0.8678056230901698

Hình 30:



Hình 31: Đường cong ROC - Support Vector Machine

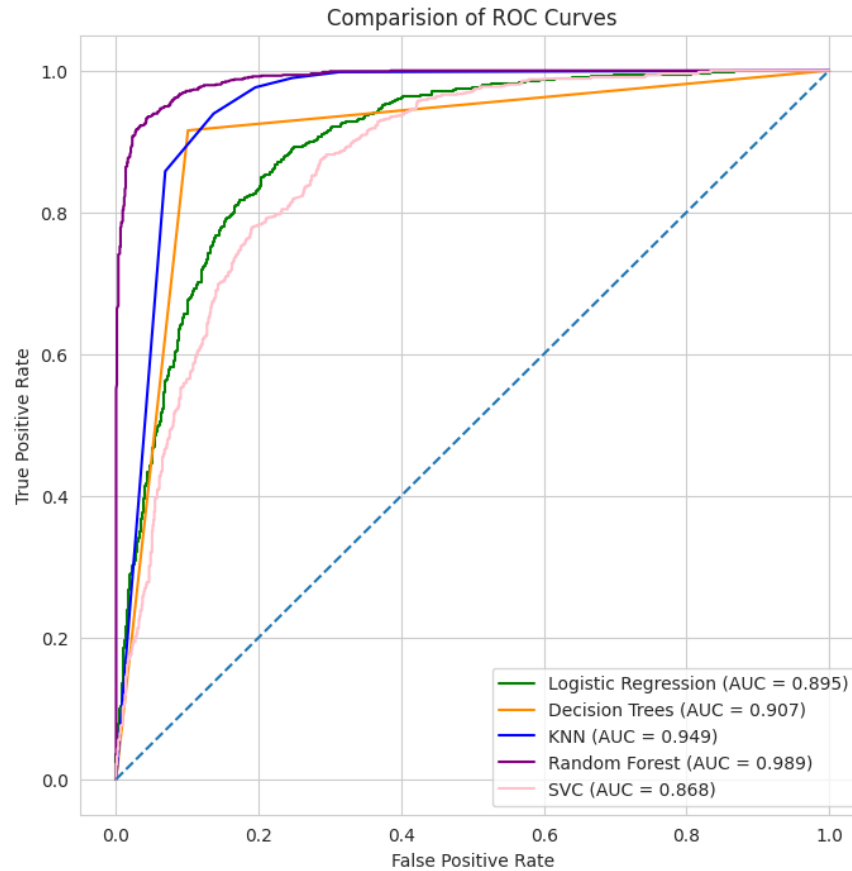
8 Đánh giá mô hình

```
A = {'Models' : ['Logistic Regression', 'Decision Tree Classifier', 'K Nearest Neighbors', 'Random Forest Classifier', 'Support Vector Machine']}
A = pd.DataFrame(A)
A
```

✓ 0.0s Python

	Models	AUC Scores
0	Logistic Regression	0.894990
1	Decision Tree Classifier	0.907421
2	K Nearest Neighbors	0.949319
3	Random Forest Classifier	0.988723
4	Support Vector Machine	0.867806

Hình 32: Tổng hợp AUC



Hình 33: So sánh các đường cong ROC

Nhận xét:

So sánh từng mô hình:

1. Random Forest (AUC = 0.989):
Đường cong gần như bao phủ toàn bộ phần góc trên bên trái.
Đây là mô hình có hiệu suất tốt nhất trong số các mô hình.
Tỷ lệ **True Positive** rất cao trong khi tỷ lệ **False Positive** thấp.
2. KNN(AUC = 0.949):
Hiệu suất rất tốt, với đường cong ROC nằm ngay sau Random Forest.
Mô hình này cũng thể hiện khả năng phân loại rất tốt.
3. Decision Tree (AUC = 0.907):
Hiệu suất tốt, nhưng thấp hơn Random Forest và KNN
Đường cong ROC cho thấy nó đáng tin cậy nhưng không phải lựa chọn tối ưu nhất
4. Logistic Regression (AUC = 0.895):
Hiệu suất khá tốt nhưng thấp hơn ba mô hình trên.

Đây là mô hình đơn giản và dễ triển khai, có thể phù hợp nếu cần cân bằng giữa độ phức tạp và hiệu quả.

5. Support Vector Machine ($AUC = 0.868$):

Hiệu suất thấp nhất trong số các mô hình được so sánh.

Đường cong ROC nằm gần đường đoán ngẫu nhiên hơn các mô hình khác, cho thấy SVC không phải là lựa chọn tối ưu.

Như vậy, Random Forest là mô hình tốt nhất với AUC gần như đạt tối đa (0.989), cho thấy nó có khả năng phân loại mạnh mẽ. KNN cũng rất ấn tượng với AUC cao (0.949), có thể là lựa chọn thay thế khi cần sự đơn giản hơn so với Random Forest. SVC là mô hình kém hiệu quả nhất trong bài toán này, cần được xem xét kỹ nếu sử dụng.

9 Kết luận

Trong bài toán so sánh hiệu suất của các mô hình phân loại qua biểu đồ ROC và chỉ số AUC, chúng ta nhận thấy sự khác biệt rõ rệt về hiệu quả giữa các thuật toán. Random Forest là mô hình vượt trội nhất với AUC đạt 0.989, gần như tối đa, cho thấy khả năng phân loại rất chính xác và độ tin cậy cao. Đường cong ROC của mô hình này nằm sát góc trên bên trái, thể hiện tỷ lệ True Positive (TPR) cao ngay cả khi tỷ lệ False Positive (FPR) thấp. Điều này làm cho Random Forest trở thành lựa chọn lý tưởng khi cần một mô hình mạnh mẽ, đặc biệt trong các bài toán có yêu cầu cao về độ chính xác.

KNN (K-Nearest Neighbors) cũng cho thấy hiệu suất xuất sắc với AUC đạt 0.949. Mặc dù kém hơn Random Forest, KNN vẫn là một lựa chọn tốt, nhất là khi cần một mô hình dễ triển khai và không quá phức tạp. Tuy nhiên, nhược điểm của KNN là có thể gặp khó khăn trong xử lý dữ liệu lớn vì yêu cầu tính toán khoảng cách cho từng điểm dữ liệu.

Các mô hình khác như Decision Tree ($AUC = 0.907$) và Logistic Regression ($AUC = 0.895$) cũng cho thấy hiệu suất khá tốt, nhưng thấp hơn so với Random Forest và KNN. Decision Tree có thể là lựa chọn phù hợp trong các bài toán đơn giản hoặc khi cần giải thích mô hình rõ ràng, nhưng nó dễ gặp vấn đề overfitting khi dữ liệu phức tạp. Trong khi đó, Logistic Regression, với hiệu suất ổn định và dễ hiểu, là một lựa chọn phổ biến khi cần cân bằng giữa độ phức tạp và hiệu quả.

Ngược lại, SVC (Support Vector Machine) có AUC chỉ đạt 0.868, thấp nhất trong tất cả các mô hình. Điều này cho thấy mô hình SVC không phải là lựa chọn tốt trong bài toán này, có thể do đặc thù dữ liệu không phù hợp hoặc yêu cầu cao về tài nguyên tính toán.

Tóm lại, Random Forest là mô hình phù hợp nhất để sử dụng nếu mục tiêu là đạt được hiệu suất cao nhất. Nếu cần một mô hình đơn giản hơn mà vẫn đảm bảo hiệu quả, KNN là lựa chọn thay thế tốt. Các mô hình khác như Decision Tree và Logistic Regression có thể được sử dụng trong những tình huống đơn giản hơn hoặc khi cần giải thích rõ ràng. Việc chọn mô hình phù hợp cuối cùng phụ thuộc vào yêu cầu cụ thể của bài toán, tài nguyên tính toán, và khả năng mở rộng trong ứng dụng thực tế.