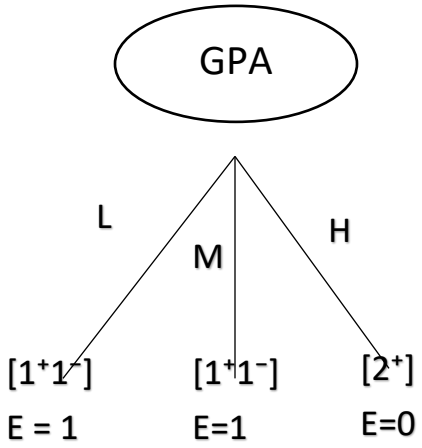
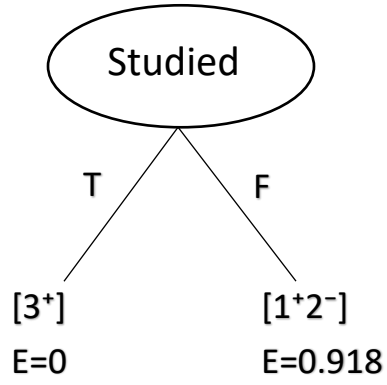
นายฟาริซ หะยีมะสาและ 61050258

1. Entropy Passed = $-\frac{4}{2}\log_2\frac{4}{2}-\frac{2}{6}\log_2\frac{2}{6}$ = 0.918

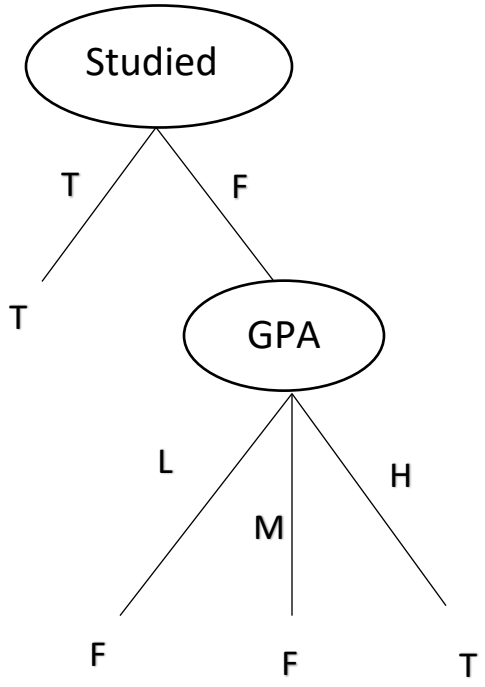


Gain = $0.918-\left(\left(\frac{2}{6}*1\right)+\left(\frac{2}{6}*1\right)+\left(\frac{2}{6}*0\right)\right)$ = 0.251



Gain = $0.918-\left(\left(\frac{3}{6}*0\right)+\left(\frac{3}{6}*0.918\right)\right)$ = 0.459

เพราะฉะนั้นเลือก Studied ไปใช้ เพราะมีค่า Gain มากที่สุด และจึงใช้ GPA เป็นตัวต่อไป(ไม่ต้องคิดค่า Gain ใหม่ เพราะเหลือ GPA เป็นตัวสุดท้าย)



2. (**โค้ดอยู่ใน** Colab Exam2 **นะครับ**)

2.1)เกิดปัญหา Overfitting ขึ้น เพราะ มีจำนวนข้อมูลน้อยเกินไปและพอแบ่งข้อมูล Train และ Test ก็จะเกิดการเอนเอียงของข้อมูลที่นำไป train อาจมีข้อมูลที่เป็น class = Don't ที่เข้าไป train มากกว่า class = Invite และทำให้ model ทำนายค่าได้ไม่ดี วิธีแก้ปัญหา คือ เพิ่มจำนวนของข้อมูลให้ Class = Invite และ Class = Don't มีจำนวนข้อมูลพอๆกัน

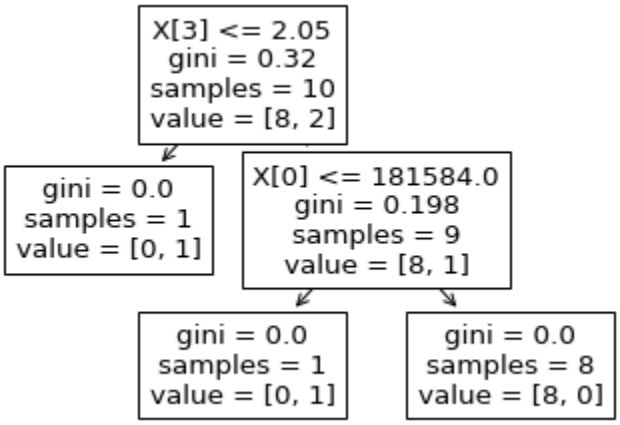
2.2) Random Forest ได้ผลลัพธ์ที่ดีกว่า Decision Tree เพราะ random forest สร้าง Tree มาหลายต้น และทำการคำนวนหาต้นไม้ที่ดีที่สุดมาใช้งาน แต่ Decision Tree จะสร้าง Tree มาแค่ต้นเดียวแล้วนำมาใช้งาน

2.3) ผมได้ค่า Accuracy ของ Random Forest และ XGBoost เท่ากันครับ ซึ่งทั้งสองตัวทำงานคล้ายๆกันโดย นำ Decision Tree หลายๆต้นมาทำต่อโดย random forest การ prediction ในแต่ละต้นไม้จะทำการ predict ของต้นไม้นั้นๆและเอาค่าที่ดีที่สุดมาโชว์ ส่วน XGBoost จะทำต้นไม้ต่อๆกันโดยต้นไม้ด้านล่างจะเรียนรู้ error จากต้นไม้ด้านบน ทำให้ได้ผลลัพธ์ที่ดีขึ้น (ที่ผมได้ค่า Accuracyเท่ากันอาจเป็นเพราะการเลือกใช้คอลัมของผมด้วยครับ)

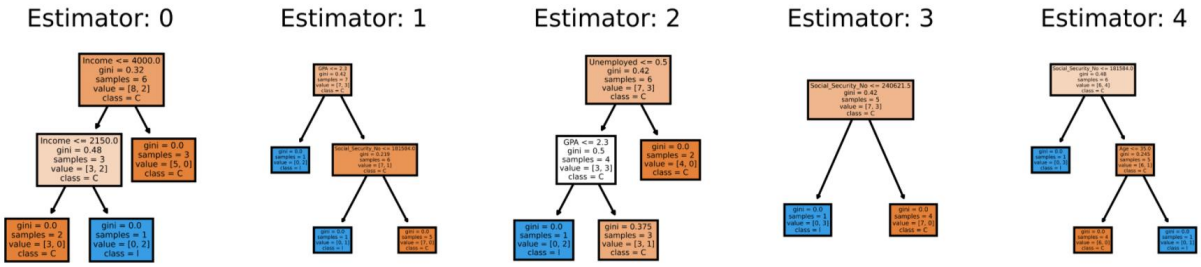
2.4)ได้ค่า Accuracy พอๆกับ ตอนที่ไม่ได้ใช้ครับ เลยคิดว่าไม่เกี่ยวข้องกับประสิธิภาพครับ อาจเป็นดพราะมีข้อมูลน้อยเกินไปเลยทำให้ไม่เห็นความต่างกันครับ

2.5)

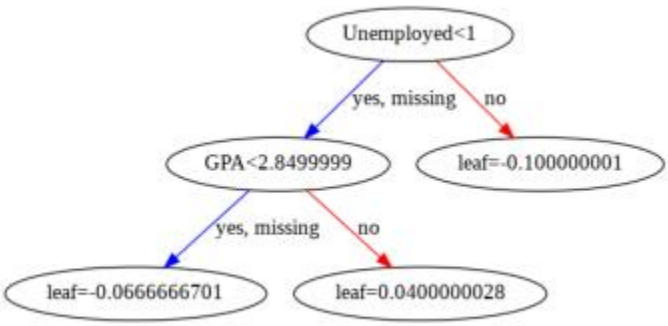
Decision Tree



Random Forest (โชว์ออกมาแค่ 5 ต้นนะครับ)



XGBoost



Referents :

https://medium.com/@witchapongdaroontham/%E0%B8%A3%E0%B8%B9%E0%B9%89%E0%B8%88%E0%B8%B1%E0%B8%81-decision-tree-random-forrest-%E0%B9%81%E0%B8%A5%E0%B8%B0-xgboost-part-1-cb49c4ac1315 , https://towardsdatascience.com/visualizing-decision-trees-with-python-scikit-learn-graphviz-matplotlib-1c50b4aa68dc , https://machinelearningmastery.com/visualize-gradient-boosting-decision-trees-xgboost-python/ , https://scikit-learn.org/stable/modules/cross_validation.html

3. **(ดูโค้ดใน Colab Exam3 นะครับ)**

หลังจากทำ SFFS แล้วได้เลือกมาทั้งหมด 25 คอลัม เหตุผลเพราะ เป็นคะแนนที่ดีที่สุดและมีจำนวนคอลัมที่ใส่มากที่สุด เพราะ หลังจากใส่คอลัมไปมากกว่า 25 คอลัม จะทำให้คะแนนลดลง จึงเลือกใช้แค่ 25 คอลัม หลังจากนั้นได้นำมาทำ Cross validation แบบ 5 และ 10 และ ได้นำไปทำ Normalization แล้วนำไปทำ Cross validation อีกรอบ ได้ผลลัพธ์ ค่า Accuracy ของ model ก่อนทำ Normalization ดีกว่าค่า Normalization อาจเป็นเพราะข้อมูลได้ถูกปรับปรุงมาก่อนแล้วจึงไม่เกิดความซ้ำซ้อนของข้อมูลจึงทำให้หลังจากทำ Normalization อีกรอบ จึงทำให้ข้อมูลเกิดความผิดเพี้ยน

ค่า Accuracy ก่อนทำ Normalization

Cross Validation แบบ 5

```
Accuracy Decision Tree: 0.88 (+/- 0.13)
Accuracy RandomForest: 0.84 (+/- 0.21)
Accuracy XGBoost: 0.81 (+/- 0.23)
```

Cross Validation แบบ 10

```
Accuracy Decision Tree: 0.88 (+/- 0.31)
Accuracy RandomForest: 0.84 (+/- 0.44)
Accuracy XGBoost: 0.81 (+/- 0.53)
```

ค่า Accuracy หลังทำ Normalization

Cross Validation แบบ 5

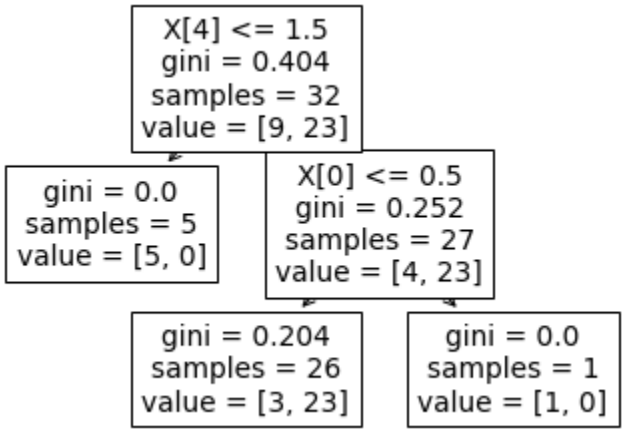
```
Accuracy Decision Tree: 0.81 (+/- 0.14)
Accuracy RandomForest: 0.84 (+/- 0.21)
Accuracy XGBoost: 0.81 (+/- 0.23)
```

Cross Validation แบบ 10

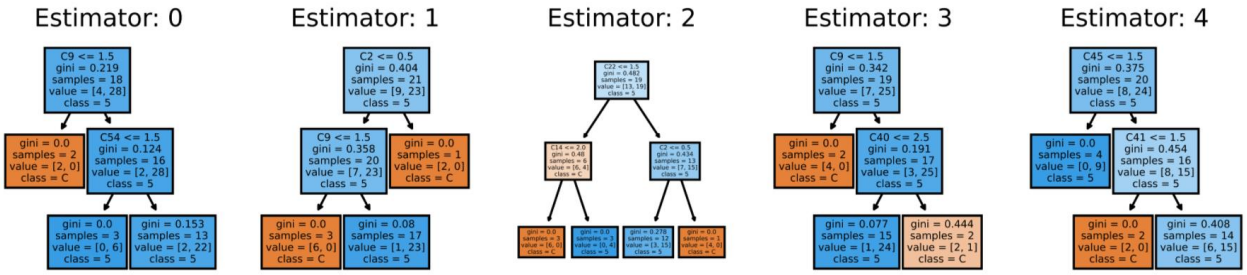
```
Accuracy Decision Tree: 0.84 (+/- 0.44)
Accuracy RandomForest: 0.81 (+/- 0.43)
Accuracy XGBoost: 0.81 (+/- 0.53)
```

Tree Non-Normalization

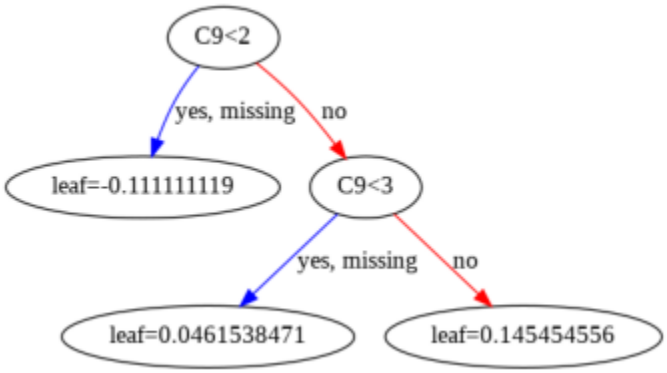
Decision Tree



Random Forest

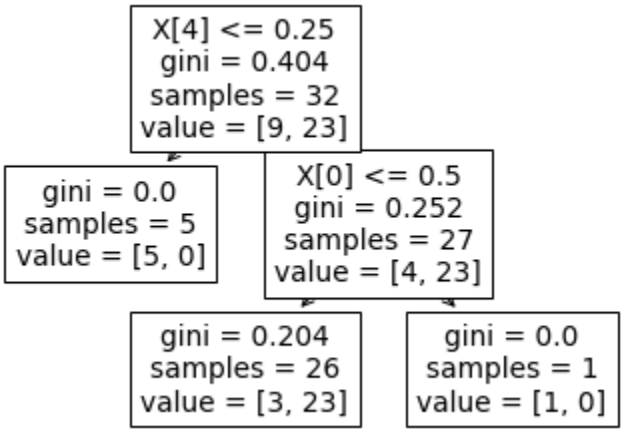


XGBoost

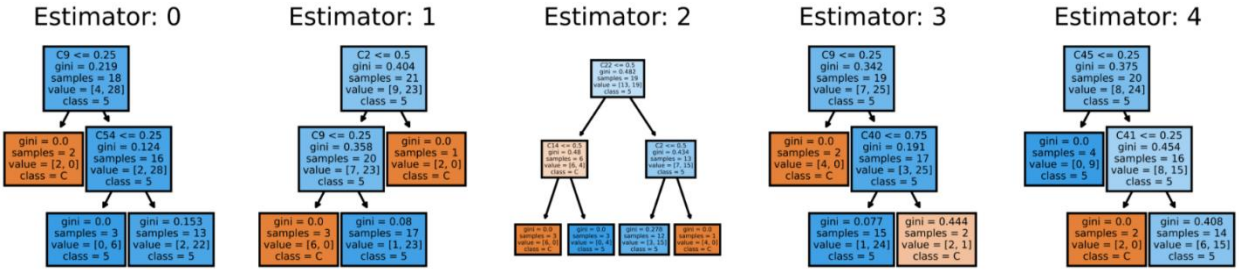


Tree Non-Normalization

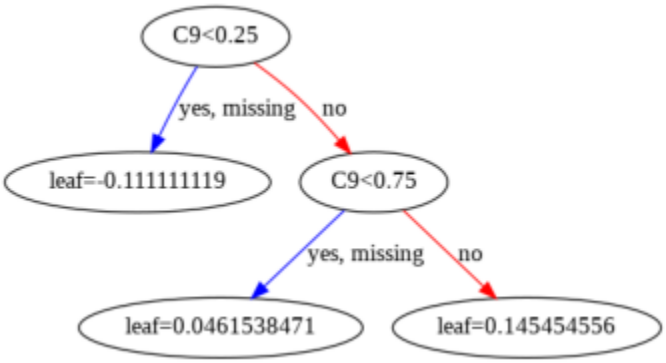
Decision Tree



Random Forest



XGBoost



4. (ดูโค้ดจาก Colab Exam4 นะครับ)

4.1) Multiple regression

Intercept

```
intercept 54.967291987653475
```

Coefficient

```
coefficient [ 1.41319717e-02 -3.26348910e+01  3.37722297e-01  1.16899164e-01
  1.60229534e-01  2.60292355e-02 -1.21189329e-01  5.55575229e-02]
```

Adjusted r square

```
adjusted r square 0.989769950831482
```

```
test_nor = pd.DataFrame()
test_nor['ACTUAL'] = y_test
test_nor['Predict'] = pred_nor
test_nor['MAE%'] = (test_nor['ACTUAL']-test_nor['Predict']).abs()/test_nor['ACTUAL']*100
test_nor
```

	ACTUAL	Predict	MAE%
553	16.275246	15.582258	4.257930
331	18.496499	17.914998	3.143843
241	25.781250	25.871918	0.351682
1957	41.527273	40.458647	2.573312
1691	38.140162	38.968016	2.170558
...	...	...	...
1012	28.488629	28.407204	0.285816
340	22.222222	21.655078	2.552150
2005	41.497955	40.437778	2.554769
1671	36.241264	35.609993	1.741859
1618	37.814523	38.180327	0.967364

4.2) แต่ละตัวแปรไม่มีคว่มสัมพันเกิน 0.80

	Age	Height	Weight	FCVC	NCP	CH2O	FAF	TUE
Age	1.000000	0.025958	0.202560	0.016291	0.043944	0.045304	0.144938	0.296931
Height	0.025958	1.000000	0.463136	0.038121	0.243672	0.213376	0.294709	0.051912
Weight	0.202560	0.463136	1.000000	0.216125	0.107469	0.200575	0.051436	0.071561
FCVC	0.016291	0.038121	0.216125	1.000000	0.042216	0.068461	0.019939	0.101135
NCP	0.043944	0.243672	0.107469	0.042216	1.000000	0.057088	0.129504	0.036326
CH2O	0.045304	0.213376	0.200575	0.068461	0.057088	1.000000	0.167236	0.011965
FAF	0.144938	0.294709	0.051436	0.019939	0.129504	0.167236	1.000000	0.058562
TUE	0.296931	0.051912	0.071561	0.101135	0.036326	0.011965	0.058562	1.000000

4.3) **ไม่สามารถ ค่า MAE เพราะไม่มีข้อมูลผลลัพเพื่อมาเทียบความถูกต้อง**