Introduction

งานชิ้นนี้เป็นการเปรียบเทียบประสิทธิภาพการวิเคราะห์ระหว่าง Traditional Machine Learning (ML) กับ Mutilayer Perceptron (MLP) ในข้อมูลประเภท Tabular Data ซึ่ง Traditional ML ที่นำมาใช้ในงานชิ้นนี้ คือ KNN, XGBoost, Decision Tree, SVM และ Naive Bayes โดยในการเปรียบเทียบครั้งนี้ข้อมูลที่นำมาใช้จะเป็นการจำแนกข้อมูลแบบ Binary Classification

Data

ชุดข้อมูลนี้เป็นการวิเคราะห์รายได้ของประชากรจากปัจจัยต่าง ๆ โดยที่ผู้มีรายได้มากกว่า 50,000$ จะแสดงผลเป็น 1 หากมีรายได้น้อยกว่าหรือเท่ากับ 50,000$ จะแสดงผลเป็น 0 ซึ่งปัจจัยที่ใช้วิเคราะห์มีดังภาพที่ 1

ภาพที่ 1 แสดงรายละเอียดของชุดข้อมูลรายได้ของประชากรจากปัจจัยต่าง ๆ

```
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   age              43957 non-null  int64
 1   workclass        41459 non-null  object
 2   fnlwgt           43957 non-null  int64
 3   education        43957 non-null  object
 4   educational-num  43957 non-null  int64
 5   marital-status   43957 non-null  object
 6   occupation       41451 non-null  object
 7   relationship     43957 non-null  object
 8   race             43957 non-null  object
 9   gender           43957 non-null  object
 10  capital-gain     43957 non-null  int64
 11  capital-loss     43957 non-null  int64
 12  hours-per-week   43957 non-null  int64
 13  native-country   43194 non-null  object
 14  income_>50K      43957 non-null  int64
dtypes: int64(7), object(8)
```

โดยแต่ละปัจจัยมีความหมายดังต่อไปนี้

age: อายุ

workclass: หน่วยงานที่สังกัด

fnlwgt: จำนวนคนที่มีข้อมูลตรงกับปัจจัยเหล่านี้

education: ระดับการศึกษา (Categolical)

education-num: ระดับการศึกษา (Numerical)

marital-status: สถานภาพสมรส

occupation: อาชีพ

relationship: สถานภาพครอบครัว

race: เชื้อชาติ

gender: เพศสภาพ

capital-gain: กำไรจากหลักทรัพย์

capital-loss: ขาดทุนจากหลักทรัพย์

hours-per-week: จำนวนชั่วโมงที่ทำงานใน 1 สัปดาห์

native-country: ชาติกำเนิด

income_>50K: รายได้มากกว่า $50,000

ซึ่งจากที่ได้ทำการประเมิน พบว่าปัจจัย education กับ education-num นั้นซ้ำกันในเชิงความหมาย และพบว่าในบางปัจจัยของบางแถวนั้นไม่มีข้อมูล จึงได้มีการปรับปัจจัยดังต่อไปนี้

1. ตัดตัวแปร education-num ออก
2. สร้างปัจจัย capital เพิ่มเติม โดยกำหนดให้ capital = capital-gain + capital-loss จากนั้นตัดตัวแปร capital-gain และ capital-loss ออก
3. ตัดแถวที่พบปัจจัยที่ว่าง
4. ทำ LabelEncoder เพื่อให้ columns ที่เป็น object type เป็น int type
5. ทำ Normalization
6. ทำ Data split เพื่อแยกข้อมูลระหว่าง Train data และ Test data

จะได้ว่า ข้อมูลนั้นมีทั้งหมด 40,727 แถว 13 คอลัมน์ ดังภาพที่ 2

ภาพที่ 2 แสดงรายละเอียดของชุดข้อมูลหลังผ่านการ Cleansing

```
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   age             40727 non-null  int64
 1   workclass       40727 non-null  object
 2   fnlwgt          40727 non-null  int64
 3   education       40727 non-null  object
 4   marital-status  40727 non-null  object
 5   occupation      40727 non-null  object
 6   relationship    40727 non-null  object
 7   race            40727 non-null  object
 8   gender          40727 non-null  object
 9   hours-per-week  40727 non-null  int64
 10  native-country  40727 non-null  object
 11  income_>50K     40727 non-null  int64
 12  capital         40727 non-null  int64
dtypes: int64(5), object(8)
```

Network architecture and Training

สถาปัตยกรรมของเครือข่ายโครงสร้างประสาทเทียมที่ใช้ในการศึกษา binary classification ครั้งนี้เป็นแบบ sequential ประกอบด้วย input layer และ output layer ที่มี 12 และ 1 โหนดตามลำดับ โดยปรับค่าจำนวนของ hidden nodes เป็น 4, 9, 14 และ hidden layers ตั้งแต่ 2 ถึง 4 ดังตาราง ซึ่ง activation function ที่ใช้ใน hidden node และ output node คือ relu และ sigmoid ตามลำดับ วาง batch normalization (BN) layer หลังการใช้ activation function ของแต่ละ hidden node นอกจากนี้สุ่มเอา node ใน hidden layer สุดท้ายออกด้วย dropout rate เท่ากับ 0.1 และ 0.3 ดังตาราง เพื่อลดปัญหา overfitting เลือก ADAM เป็น optimizer ที่มี learning rate เท่ากับ 0.001 และ binary cross entropy (BCE) เป็น loss function ใช้ batch size เท่ากับ 64 และ training dataset จำนวน 20% ถูกแบ่งออกมาเป็น validating set โดยเลือกเครือข่ายที่ให้ validating accuracy สูงสุดจากทั้งหมด 50 epochs

ตารางที่ 1 แสดงจำนวน node และ layer ของ MLP

MLP model	1	2	3	4	5	6	7	8	9
# Hidden nodes	4	4	4	4	4	4	9	9	9
# Hidden layers	2	2	3	3	4	4	2	2	3
Dropout rate	0.1	0.3	0.1	0.3	0.1	0.3	0.1	0.3	0.1

MLP model	10	11	12	13	14	15	16	17	18
# Hidden nodes	9	9	9	14	14	14	14	14	14
# Hidden layers	3	4	4	2	2	3	3	4	4
Dropout rate	0.3	0.1	0.3	0.1	0.3	0.1	0.3	0.1	0.3

Results

ตัวจำแนก MLP ทั้งหมด 18 แบบถูกเทรนผ่านโมดูล tf.keras ของ TensorFlow บน Tesla T4 GPU ด้วยภาษา Python โดยเมื่อเทียบกับตัวจำแนก MLP แบบอื่น พบว่าตัวจำแนก MLP ที่มี hidden nodes และ hidden layers เป็นจำนวนเท่ากับ 4 และใช้ dropout rate เท่ากับ 0.1 ให้ความแม่นยำของการทำนายคลาสของข้อมูลทดสอบ (testing data) สูงสุดเท่ากับ 80.37% และใช้เวลาสอนตัวแบบเป็นระยะเวลา 83.1788 วินาที ในขณะที่ตัวจำแนก MLP ที่มี hidden nodes และ hidden layers เป็นจำนวนเท่ากับ 14 และ 2 ตามลำดับ และใช้ dropout rate เท่ากับ 0.3 ใช้เวลาสอนตัวแบบน้อยสุดเป็นระยะเวลา 57.2679 วินาที และให้ความแม่นยำเท่ากับ 80.06% โดยตัวจำแนก MLP แต่ละแบบที่ปรากฏอยู่ในตารางถูกเขียนแทนด้วยสัญลักษณ์ MLP (จำนวนของ hidden nodes, จำนวนของ hidden layers, dropout rate)

เมื่อเปรียบตัวจำแนก MLP ทั้ง 18 แบบกับตัวจำแนกแบบอื่น พบว่า SVM ให้ความแม่นยำสูงสุดเท่ากับ 84.13% และใช้สอนตัวแบบเป็นระยะเวลา 11.9070 วินาที ในขณะที่ Decision tree ใช้เวลาสอนตัวแบบน้อยสุดเป็นระยะเวลา 0.0029 วินาที และให้ความแม่นยำเท่ากับ 82.75% ส่วนตัวจำแนก Bayes ที่มี prior probability ของรายได้มากกว่า $50,000 เท่ากับ 0.52 ให้ความแม่นยำเท่ากับ 83.56% ซึ่งสูงสุดเมื่อพิจารณาค่าของ prior probability จาก 0.02 ถึง 0.98 โดยใช้ step size เท่ากับ 0.02 และใช้เวลาสอนตัวแบบเป็นระยะเวลา 0.0295 วินาที

ตารางที่ 2 แสดงค่า Accuracy และ Runtime ของแต่ละ Model

Model	**Accuracy (%)**	**Runtime (s)**
Decision tree	82.75	0.0029
XGboost	86.41	2.272
KNN	72.41	0.105
SVM	84.13	11.9070
Bayes (prior prob = 0.52)	83.56	0.0295
MLP (4, 2, 0.1)	78.68	72.5352
MLP (4, 2, 0.3)	78.76	57.3147

MLP (4, 3, 0.1)	32.73	67.1089
MLP (4, 3, 0.3)	80.24	66.4144
MLP (4, 4, 0.1)	80.37	83.1788
MLP (4, 4, 0.3)	78.94	83.4410
MLP (9, 2, 0.1)	78.04	82.7276
MLP (9, 2, 0.3)	78.91	57.4319
MLP (9, 3, 0.1)	79.90	66.9790
MLP (9, 3, 0.3)	79.34	67.5228
MLP (9, 4, 0.1)	79.14	79.1706
MLP (9, 4, 0.3)	74.06	83.5223
MLP (14, 2, 0.1)	37.89	82.7364
MLP (14, 2, 0.3)	80.06	57.2679
MLP (14, 3, 0.1)	78.80	66.8978
MLP (14, 3, 0.3)	79.44	67.4754
MLP (14, 4, 0.1)	79.44	77.6001
MLP (14, 4, 0.3)	79.79	83.1449

Discussion and Conclusion

ตารางที่ 3 เปรียบเทียบค่าเฉลี่ยของ Accuracy และ Runtime ในการสอนตัวแบบของแต่ละ Model

Model	Average Accuracy (%)	Average Runtime (s)
Decision Tree	82.75	0.0029
XGboost	86.41	2.272
KNN	72.41	0.105
SVM	84.13	11.907
Bayes (49 models)	81.65	0.0335
MLP (18 networks)	73.87	72.3594

จากการทดลองเปรียบเทียบประสิทธิภาพการวิเคราะห์ระหว่าง Traditional ML กับ MLP ในข้อมูลประเภท Tabular Data พบว่า MLP มีค่าเฉลี่ยของ accuracy อยู่ที่ 73.87% และใช้เวลาในการสอนตัวแบบ 72.3594 วินาที ซึ่งเมื่อเทียบกับ ML ที่มี accuracy ที่ใกล้เคียงกัน ได้แก่ KNN ที่มีค่า accuracy อยู่ที่ 72.41% แต่ใช้เวลาในการสอนตัวแบบ 0.105 วินาที หรือ ML ที่ใช้ระยะเวลามากที่สุดคือ SVM ที่มี accuracy อยู่ที่ 84.13% แต่ใช้เวลาในการสอนตัวแบบ 11.907 วินาที จะได้ว่า ML นั้นให้ความแม่นยำที่ใกล้เคียง หรือสูงกว่า MLP โดยที่ใช้ระยะเวลาในการสอนตัวแบบน้อยกว่า และใช้ระยะเวลาในการปรับค่า parameter ต่าง ๆ น้อยกว่า ทั้งนี้อาจเป็นเพราะชุดข้อมูลที่เลือกใช้ในครั้งนี้มีความซับซ้อนไม่เพียงพอต่อความจำเป็นในการเลือกใช้ MLP

Reference

Decision Trees: https://scikit-learn.org/stable/modules/tree.html#classification

K-Nearest Neighbors: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

Naive Bayes: https://scikit-learn.org/stable/modules/naive_bayes.html

Naive Bayes for Mixed Typed Data: https://medium.com/analytics-vidhya/naive-bayes-for-mixed-typed-data-in-scikit-learn-fb6843e241f0

Module tf.keras: https://www.tensorflow.org/api_docs/python/tf/keras/

Support Vector Machines: https://scikit-learn.org/stable/modules/svm.html

Tuning the hyper-parameters of an estimator: https://scikit-learn.org/stable/modules/grid_search.html

Weighted Regression: https://jermwatt.github.io/machine_learning_refined/notes/5_Linear_regression/5_5_Weighted.html

XGboost: https://xgboost.readthedocs.io/en/stable/

End credit

งานชิ้นนี้เป็นส่วนหนึ่งของรายวิชา วขวข 7202 การเรียนรู้เชิงลึก (DADS 7202 Deep Learning) หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาการวิเคราะห์ข้อมูลและวิทยาการข้อมูล (Master of Science Program in Data Analytics and Data Science) คณะสถิติประยุกต์ สถาบันบัณฑิตพัฒนบริหารศาสตร์ (Graduate School of Applied Statistics, National Institute of Development Administration)

จัดทำโดย 6310432002 ทรงคมกฤช ไชยกาล
6410412005 เดโช ศรีสวัสดิ์
6410412015 คุณานนต์ กลิ่นจันทร์หอม