

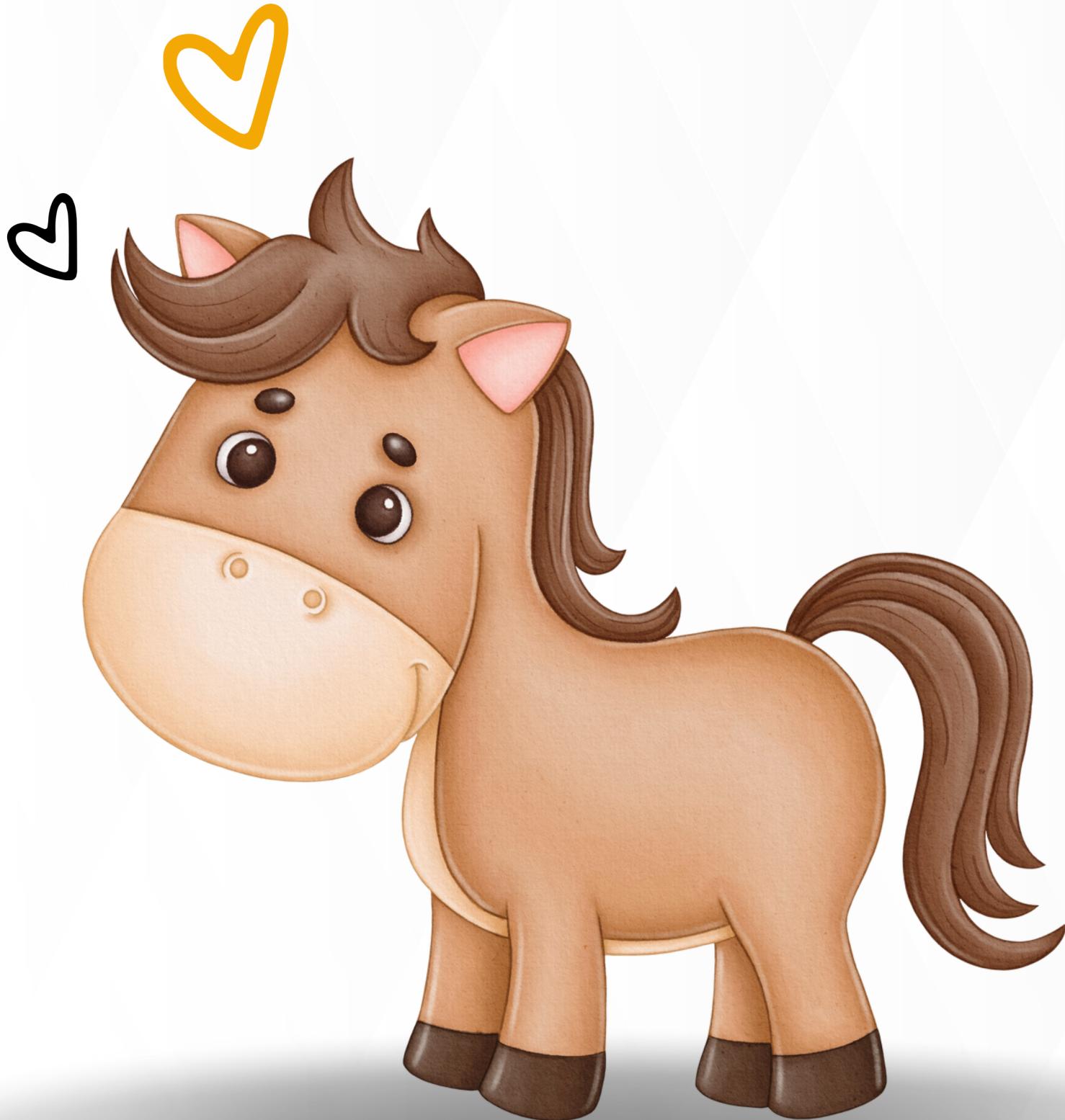


Machine learning

040613702 : 2566

PREDICT HEALTH OUTCOME OF HORSE

คำนายสุขภาพของม้า



จัดทำโดย

นาย ปิติกร วงศ์ทองอ่อน 6404062610120

นางสาว นันทินี แสวงโชคพาหะ 6404062610090



ที่มาและความสำคัญของโครงงาน

ในปัจจุบันการตรวจสอบสุขภาพของม้านั้นจำเป็นต้องใช้สัตวแพทย์ที่มีความชำนาญของผู้เชี่ยวชาญในการตรวจสุขภาพทำให้มีราคาที่สูงในการดำเนินการ และในกรณีที่ม้าเป็นจันวนมากนั้นอาจต้องใช้เวลาในการตรวจและวินิจฉัยเป็นเวลานาน ดังนั้นเพื่อช่วยในการตรวจและวินิจฉัยเป็นเวลานาน จึงได้มีการนำข้อมูลและวิธีการทางด้านวิทยาศาสตร์ข้อมูล และเทคโนโลยีสารสนเทศเพื่อช่วยในการถูกและรักษาม้าให้มีสุขภาพดีและป้องกันโรคในม้า



ประโยชน์ของโครงการและภาระนำไปใช้

- ◆ สามารถกำหนดรายได้จากการขายสุขภาพของม้าแต่ละช่วงอายุได้
- ◆ เพื่อช่วยในการตรวจและวินิจฉัยสุขภาพของม้า และช่วยในการดูแลรักษาม้าให้มีสุขภาพดีและป้องกันโรค ในม้า



ข้อมูลที่นำมาใช้ทำโครงงาน

1

ที่มาของข้อมูลจาก Kaggle

<https://www.kaggle.com/competitions/playground-series-s3e22>



ข้อมูลที่นำมาใช้ทำโครงงาน

2

Attribute และ Feature

: id, surgery (ผ่าตัด), age, rectal_temp (อุณหภูมิทางทวาร), pulse (ชีพจร), respiratory_rate (อัตราการหายใจ), temp_of_extremities (อุณหภูมิแขนขา)

# id	surgery	age	rectal_temp	pulse	respiratory...	temp_of_e...
0	yes	adult	38.1	132.0	24.0	cool
1	yes	adult	37.5	88.0	12.0	cool
2	yes	adult	38.3	120.0	28.0	cool



ข้อมูลที่นำมาใช้ทำโครงงาน

3

Attribute และ Feature (Cont.)

: peripheral_pulse (ชีพจรล่วนปลาย), mucous_membran (สีของเยื่อเมือก), capillary_refill_time (การวัดการไหลเวียนของเส้นเลือดผอย) , pain (ความเจ็บปวด), peristalsis (การบีบตัวของลำไส้), abdominal_distension (อาการท้องอืด), nasogastric_tube(ก้าลออกมานอกห้องชมูก)

▲ peripheral...	▲ mucous_m...	▲ capillary_r...	▲ pain	▲ peristalsis	▲ abdominal...	▲ nasogasti...
reduced	dark_cyanotic	more_3_sec	depressed	absent	slight	slight
normal	pale_cyanotic	more_3_sec	mild_pain	absent	moderate	none
reduced	pale_pink	less_3_sec	extreme_pain	hypomotile	moderate	slight

ข้อมูลที่นำมาใช้ทำโครงงาน

4

Attribute และ Feature (Cont.)

: nasogastric_reflux(กรดไหลย้อนทางจมูก), nasogastric_reflux_ph(ค่า ph กรดไหลย้อนทางจมูก)
, rectal_exam_feces(การตรวจอุจจาระ), abdomen(ช่องท้อง),
packed_cell_volume(เปอร์เซนต์ของเม็ดเลือด), total_protein(โปรตีนรวม), abdomo_appearance
(ของเหลวจากท้อง)

# nasogastric_reflux	# nasogastric_reflux_ph	# rectal_exam_feces	# abdomen	# packed_cell_volume	# total_protein	# abdomo_appearance
less_1_liter	6.5	decreased	distend_small	57.0	8.5	serosanguious
more_1_liter	2.0	absent	distend_small	33.0	64.0	serosanguious
none	3.5	None	distend_large	37.0	6.4	serosanguious

ข้อมูลที่นำมาใช้ทำโครงงาน

5

Attribute และ Feature (Cont.)

: abemo_protein(ค่าโปรตีนบริเวณหน้าท้อง), surgical lesion(แผลผ่าตัด), lesion_1(รอยโรค1), lesion_2(รอยโรค2), lesion_3(รอยโรค3)

# abemo_p...	✓ surgical_le...	# lesion_1	# lesion_2	# lesion_3	✓ cp_data
3.4	yes	2209	0	0	no
2.0	yes	2208	0	0	no
3.4	yes	5124	0	0	no

ข้อมูลที่นำมาใช้ทำโครงงาน

6

Output หรือ Label ที่ต้องการ

- Id 0,1,2,3....
- Outcome (died,euthanized,lived)



id	outcome	
0	lived	46%
1	died	33%
2	Other (251)	20%
3	died	
4	euthanized	
5	lived	

Platform ที่ใช้ในการทำโครงงาน

Google colab ชื่อเต็มคือ Google Colaboratory เป็นบริการ Software as a Service (SaaS) โอลต์ไปรแกรม Jupyter Notebook บน Cloud จาก Google.



Kaggle Kernel ช่วยให้คุณสามารถประมวลผลการดำเนินการของ ML บนคอมพิวเตอร์ระบบคลาวด์แทนที่จะดำเนินการบนคอมพิวเตอร์ของคุณเอง



ผลลัพธ์ที่คาดหวัง

ด้วยตัวชี้วัดทางการแพทย์ต่างๆ ให้ทำนาย
ผลลัพธ์ด้านสุขภาพของม้า
(died, euthanized, lived)



เทคนิค/อัลกอริทึม ML ที่เลือกใช้

อัลกอริทึมที่เลือกใช้

- Logistic Regression
- RandomForest Classifier
- XGBoost Classifier
- CatBoost Classifier

เทคนิคการแบ่งข้อมูล

- KFold

เทคนิคการประเมินประสิทธิภาพ

- F1 score

```
models = {
    'logistic_regressor': LogisticRegression(random_state=42, max_iter = 1000),
    'randomforest_classifier': RandomForestClassifier(random_state=42, verbose = 0),
    'xgb_classifier': xgb.XGBClassifier(random_state=42, verbosity = 0),
    'catboost_classifier': cb.CatBoostClassifier(random_state = 42, logging_level="Silent")
}

for model_name, model in models.items():
    f1_scores = cross_val_score(model, X, y, cv=kf, scoring='f1_micro')
    avg_f1_score = f1_scores.mean()
    print(f'{model_name}\'s average F1 score across {N_SPLITS}-Fold CV is {avg_f1_score * 100:.3f}%')

logistic_regressor's average F1 score across 5-Fold CV is 63.806%
randomforest_classifier's average F1 score across 5-Fold CV is 69.474%
xgb_classifier's average F1 score across 5-Fold CV is 70.364%
catboost_classifier's average F1 score across 5-Fold CV is 69.879%
```

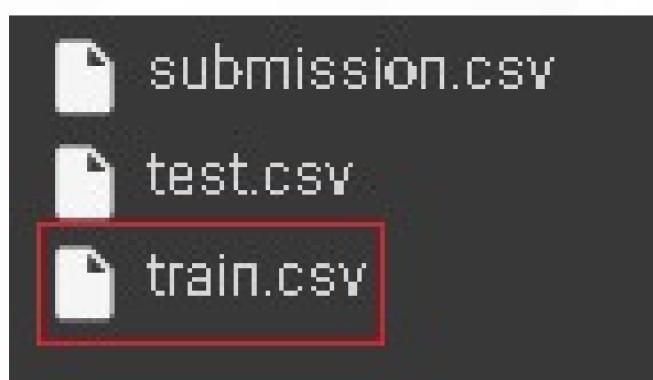
1. ศึกษา Data ทั้ง input และ output
2. ปรับแต่ง input
3. เลือก model ที่ต้องการใช้
4. Train model
5. ประเมิน model และ ปรับปรุง parameter



ขั้นตอนการ ดำเนินงาน

ศึกษา Data ห้อง input และ output

ข้อมูลที่ใช้ในการ Train model เป็นไฟล์ชื่อ train.csv
ซึ่งในไฟล์ข้อมูลจะมีค่าที่เป็น None เราจึงควรปรับค่า
ของค่านี้ๆๆ



A peripheral_pulse	A mucous_membra...	A capillary_refill_t...			
reduced	59%	pale_pink	23%	less_3_sec	68%
normal	35%	normal_pink	21%	more_3_sec	32%
Other (82)	7%	Other (691)	56%	Other (8)	1%
None	None	None	None	None	None
None	pale_pink	less_3_sec			
None	bright_red	None			
None	pale_pink	less_3_sec			
None	bright_red	more_3_sec			
None	normal_pink	less_3_sec			
None	dark_cyanotic	more_3_sec			
None	None	less_3_sec			

ศึกษา Data ห้อง input และ output

ข้อมูลที่ใช้ในการ Submission จะใช้เป็น table
id และ outcome ในการทดสอบ

id	outcome	
0	lived	46%
1	died	33%
2	Other (251)	20%
3	died	
4	euthanized	
5	lived	
6	lived	
7	euthanized	

ปรับแต่ง input

เนื่องจากมีข้อมูลบางส่วนเก็บค่าเป็น object ดังนั้นเราจึงต้องเปลี่ยนค่าให้เป็นตัวเลขก่อน โดยค่า None จะเป็น -1

ตัวอย่างโค้ดที่ใช้

```
train_rows = train.shape[0]
merged_df = pd.concat([train, test])

tmp_ext = {"None": -1, "warm": 1, "normal": 0, "cool": 2, "cold": 3}
per_purse = {"None": -1, "absent": 1, "reduced": 2, "normal": 0, "increased": 3}

for col in ["surgery", "age", "mucous_membrane", "surgical_lesion", "cp_data", "outcome"]:
    merged_df[col] = label.fit_transform(merged_df[col])

merged_df["temp_of_extremities"] = merged_df["temp_of_extremities"].map(tmp_ext)
merged_df["peripheral_pulse"] = merged_df["peripheral_pulse"].map(per_purse)
```

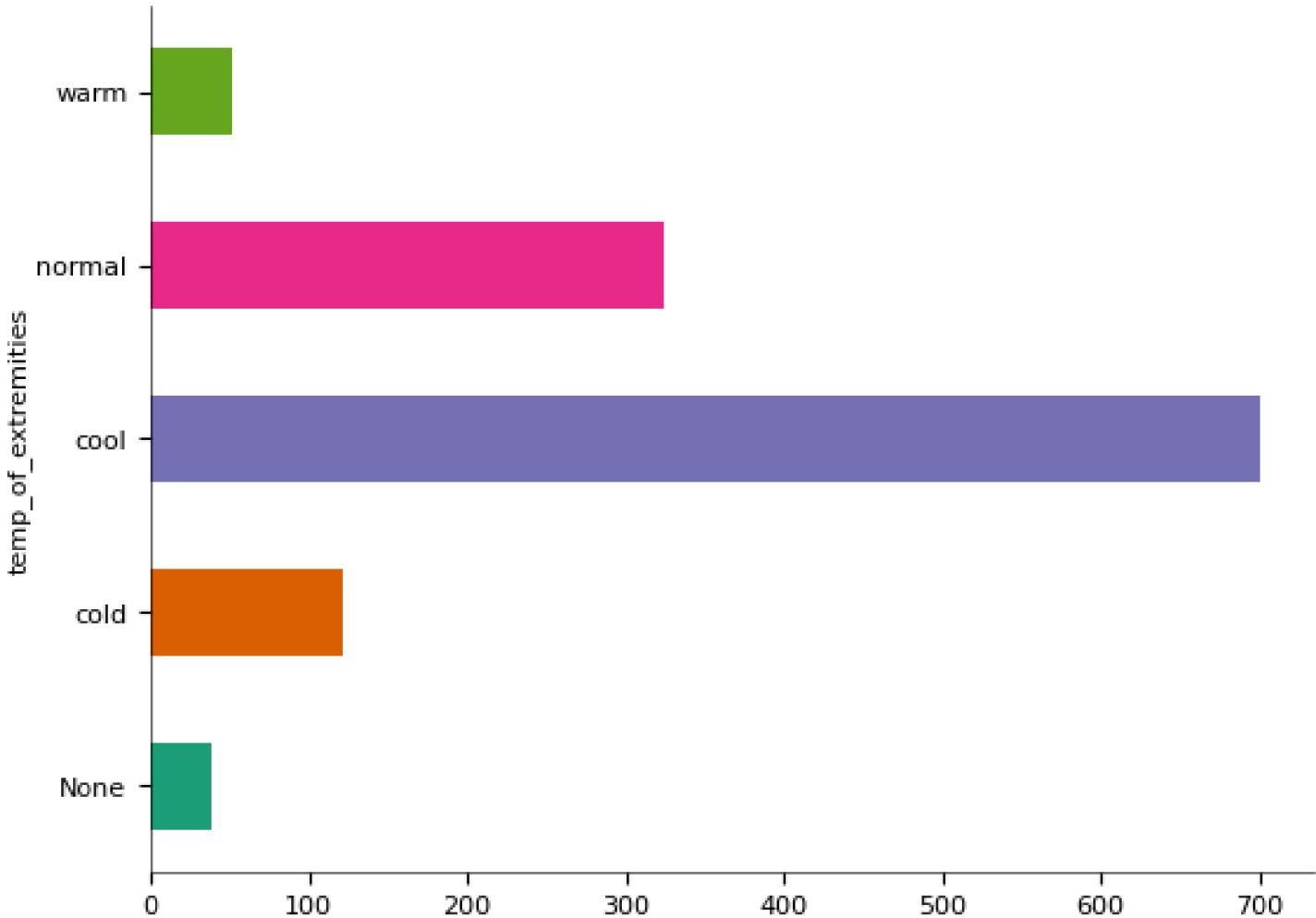
train.dtypes	
surgery	object
age	object
hospital_number	int64
rectal_temp	float64
pulse	float64
respiratory_rate	float64
temp_of_extremities	object
peripheral_pulse	object
mucous_membrane	object
capillary_refill_time	object
pain	object
peristalsis	object
abdominal_distention	object
nasogastric_tube	object
nasogastric_reflux	object
nasogastric_reflux_ph	float64

ปรับแต่ง input

เนื่องจากมีข้อมูลบางค่ามีค่าเป็น None ดังกราฟที่แสดง
เรายังต้องทำการปรับค่า None ให้เป็นค่าก่อนหน้าของ
ข้อมูลที่ขาดหายในคอลัมน์นั้นๆ โดยใช้ ‘fillna’ และ
drop คอลัมน์ที่ไม่จำเป็นทั้ง

ตัวอย่างโค้ดที่ใช้

```
for df in [train, test]:  
    df.drop(columns = ["hospital_number"], inplace=True)  
    df.fillna(method = "ffill", inplace = True)
```



ปรับแต่ง input

ทำการแบ่งข้อมูลที่จะใช้ในการเลือกและ Train model แบ่งเป็น X และ y โดย X จะเลือกทุก columน์ยกเว้น columน์สุดท้าย ส่วน y จะเอา columน์สุดท้ายมาเก็บ หลังจากเก็บค่า X และ y เสร็จแล้วจะทำการสร้าง kf ของคลาส KFold โดยระบุจำนวนแบ่งข้อมูลเป็น 5 และให้ shuffle เป็น true เพื่อให้การแบ่งข้อมูลทดสอบมีการสุ่มและความเป็นอิสระมากขึ้น

ตัวอย่างโค้ดที่ใช้

```
X = train.iloc[:, :-1]
y = train.iloc[:, -1]

N_SPLITS = 5
kf = KFold(n_splits=N_SPLITS, shuffle=True, random_state=42)
```

เลือก model ที่ต้องการใช้

เนื่องจากมีไม่เดลจำนวนนวนมากให้เลือกใช้งานแต่เราไม่รู้ว่า
ไมเดลไหนมีประสิทธิภาพมากที่สุดเราจึงทำการเลือกมา
4 ไมเดลเพื่อหาว่าควรใช้ ไมเดลไหน ดังนี้

Logistic Regression

RandomForest Classifier

XGBoost Classifier

CatBoost Classifier

เลือก model ที่ต้องการใช้

โดยทำการใช้ F1-score ในการประเมินว่าโมเดลไหนมี

ประสิทธิภาพมากที่สุด

ตัวอย่างได้

```
models = {
    'logistic_regressor': LogisticRegression(random_state=42, max_iter = 1000),
    'randomforest_classifier': RandomForestClassifier(random_state=42, verbose = 0),
    'xgb_classifier': xgb.XGBClassifier(random_state=42, verbosity = 0),
    'catboost_classifier': cb.CatBoostClassifier(random_state = 42, logging_level="Silent")
}

for model_name, model in models.items():
    f1_scores = cross_val_score(model, X, y, cv=kf, scoring='f1_micro')
    avg_f1_score = f1_scores.mean()
    print(f'{model_name}\'s average F1 score across {N_SPLITS}-Fold CV is {avg_f1_score * 100:.3f}%')
```

เลือก model ที่ต้องการใช้

ซึ่งผลลัพธ์ที่ได้มีดังนี้

```
logistic_regressor's average F1 score across 5-Fold CV is 63.887%
randomforest_classifier's average F1 score across 5-Fold CV is 69.474%
xgb_classifier's average F1 score across 5-Fold CV is 70.364%
catboost_classifier's average F1 score across 5-Fold CV is 69.879%
```

จากผลลัพธ์ที่ได้ ได้ผลสรุปว่า model ที่เลือกใช้คือ XGBoost Classifier ที่มีค่า F1 score ที่ 70.364 %

ซึ่งมีประสิทธิภาพมากที่สุด จากทั้ง 4 model

Train model

สร้าง model โดยใช้ XGBoost กำหนดพารามิเตอร์ `n_estimators = 100`, `max_depth = 3` และ `learning_rate = 0.5` และ Train model โดยใช้ข้อมูล `X` และ `y` ที่แบ่งไว้

ตัวอย่างได้ดังนี้

```
model = xgb.XGBClassifier(n_estimators = 100,max_depth = 3, learning_rate = 0.5)
model.fit(X, y)
```

ประเมิน model และปรับปรุง parameter

เมื่อทำการ fit model และทดสอบความแม่นยำได้ 69.069%

ตัวอย่างโค้ดที่ใช้

```
scores = cross_val_score(model, X, y, cv=5, scoring='accuracy')
mean_accuracy = scores.mean()
print(f"accuracy: {mean_accuracy * 100:.3f}%")

accuracy: 69.069%
```

ประเมิน model และปรับปรุง parameter

จะเห็นได้ว่ายังไงใช่ค่าที่ดีที่สุด จึงทำการปรับปรุงค่าพารามิเตอร์ให้ได้ค่าที่ดีกว่าเดิม

ตัวอย่างโดยทั่วไป

```
xgb_model = xgb.XGBClassifier(random_state=42, verbosity=0)

param = {
    'n_estimators': [100, 200, 300],
    'max_depth': [3, 4, 5],
    'learning_rate': [0.01, 0.1, 0.2],
    'min_child_weight': [1, 2, 3]
}

grid_search = GridSearchCV(xgb_model, param, cv=kf, scoring='f1_micro')

grid_search.fit(X, y)
```

ประเมิน model และปรับปรุง parameter

จะเห็นได้ว่าเมื่อทำการ fit model หลังจากปรับพารามิเตอร์แล้ว ผลลัพท์ที่ได้ออกมา มีความถูกต้องมากถึง 71.903 %

ตัวอย่างโดยทั่วไป

```
best_params = grid_search.best_params_
best_score = grid_search.best_score_
print(f'Best Hyperparameters: {best_params}')
print(f'Best F1 Score: {best_score * 100:.3f}%')
```

```
Best Hyperparameters: {'learning_rate': 0.1, 'max_depth': 3, 'min_child_weight': 1, 'n_estimators': 200}
Best F1 Score: 71.903%
```

Submission

A banner for a Kaggle competition titled "Playground Prediction Competition" featuring "Predict Health Outcomes of Horses". It shows a whimsical illustration of white swans swimming in a pond with a bridge and trees in the background.

Playground Prediction Competition

Predict Health Outcomes of Horses

Playground Series - Season 3, Episode 22

Kaggle · 1,541 teams · a month ago

Overview Data Code Models Discussion Leaderboard Rules Team Submissions Late Submission ...

Submissions

0/2

You selected 0 of 2 submissions to be evaluated for your final leaderboard score. Since you selected less than 2 submission, Kaggle auto-selected up to 2 submissions from among your public best-scoring unselected submissions for evaluation. The evaluated submission with the best Private Score is used for your final score.

Submissions evaluated for final score

All Successful Selected Errors

Recent ▾

Submission and Description

Private Score ⓘ

Public Score ⓘ

Selected

submission.csv
Complete (after deadline) · 7h ago

0.7303

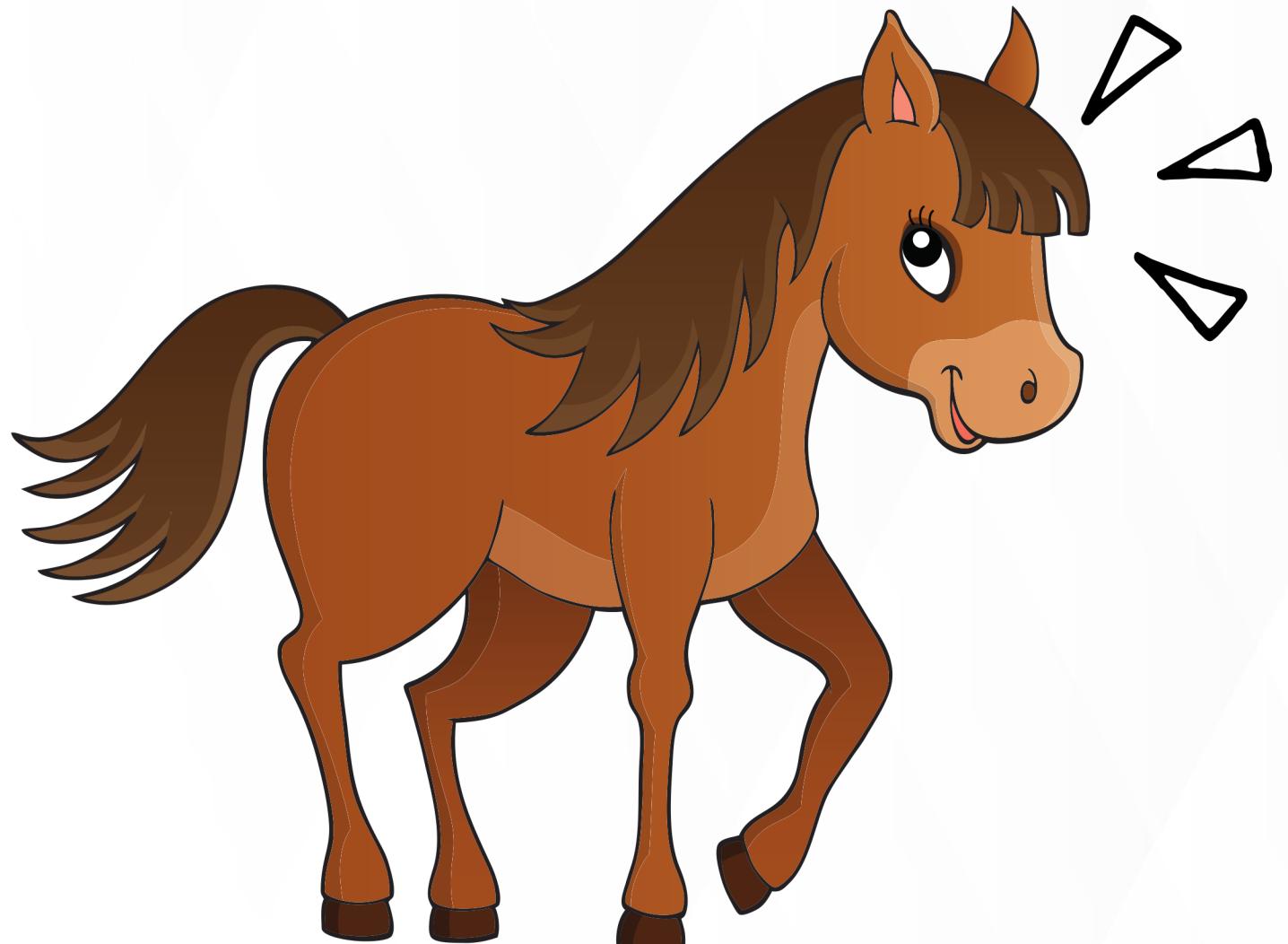
0.76829





Machine learning

040613702 : 2566



SUMMARY

เริ่มแรกจะเห็นได้ว่าการเลือกใช้โมเดลสามารถเลือกใช้ได้หลาย
หลาย เราจึงได้ทำการหาโมเดลที่เหมาะสมที่สุดในการใช้งาน ซึ่งก็คือ¹
การใช้ XGBoost ที่มีการกำหนดค่า `n_estimators`,
`max_depth` และ `learning_rate` ผลลัพธ์ที่ได้ออกมาอยู่ใน²
เกณฑ์ที่น่าพอใจ แต่เราติดว่าสามารถทำให้โมเดลมีความถูกต้องมาก
ยิ่งขึ้นโดยการปรับค่าพารามิเตอร์ของโมเดล ซึ่งหลังการปรับเปลี่ยน
ค่าพารามิเตอร์ เราพบว่าโมเดลสามารถทำงานด้วยค่าได้ถูกต้องมากยิ่งขึ้น



Thank You

