แนะนำโปรเจกต์

ในยุคที่ข้อมูลมีบทบาทสำคัญในทุกมิติของชีวิต การพยากรณ์หรือการทำนายสามารถช่วยให้เราเข้าใจและตัดสินใจได้อย่างแม่นยำมากขึ้น ในโปรเจกต์นี้ เราได้พัฒนาแอปพลิเคชันเพื่อ **“พยากรณ์คะแนนคณิตศาสตร์ของนักเรียน”** โดยใช้เทคโนโลยี **Deep Learning** ร่วมกับ **Streamlit** เพื่อสร้างอินเทอร์เฟซที่ใช้งานง่าย

โปรเจกต์นี้ออกแบบมาสำหรับครู ผู้ปกครอง หรือผู้ที่สนใจอยากวิเคราะห์ข้อมูลนักเรียนเพื่อประเมินผลลัพธ์ทางการศึกษาล่วงหน้า

วัตถุประสงค์ของโปรเจกต์

- พัฒนาระบบพยากรณ์คะแนนคณิตศาสตร์โดยใช้โมเดลปัญญาประดิษฐ์ (AI)
- สร้างอินเทอร์เฟซที่ใช้งานง่ายด้วย **Streamlit**
- เพิ่มความเข้าใจเกี่ยวกับความสัมพันธ์ระหว่างปัจจัยต่าง ๆ เช่น เพศ การศึกษา ฯลฯ กับคะแนนคณิตศาสตร์

Import Libraries

นำเข้าไลบรารีที่จำเป็นต่อการพัฒนาโครงการ เช่น **Pandas** สำหรับการจัดการข้อมูล,

TensorFlow สำหรับสร้างโมเดล และ **Streamlit** สำหรับสร้างแอปพลิเคชัน

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
import streamlit as st
```

Load Dataset

โหลดชุดข้อมูล **StudentsPerformance.csv** เพื่อใช้ในกระบวนการสร้างโมเดล

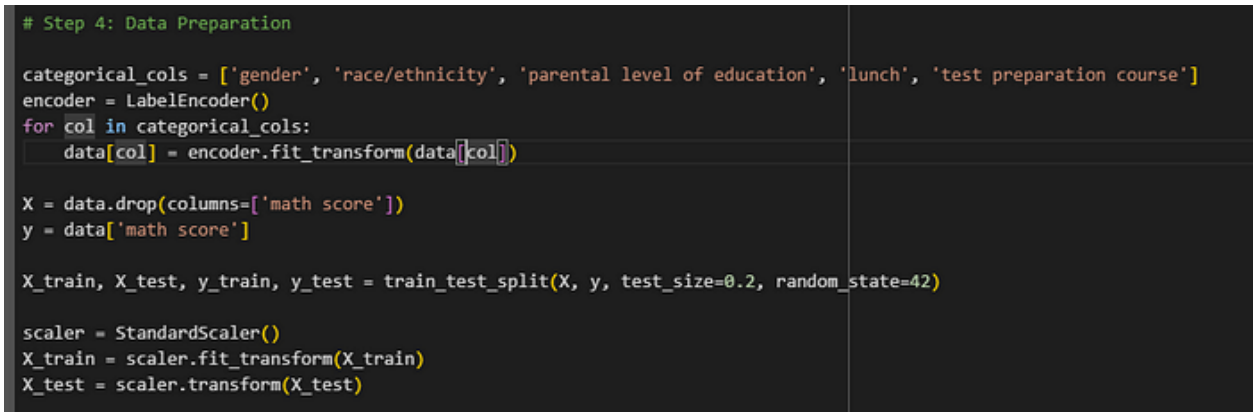
```
# Step 2: Load Dataset
file_path = "/content/StudentsPerformance.csv"
data = pd.read_csv(file_path)
```

Data Preparation

แปลงข้อมูลที่เป็นหมวดหมู่ (เช่น เพศ หรือ เชื้อชาติ) เป็นตัวเลขด้วย

LabelEncoder

- แยกข้อมูลเป็น **Features (X)** และ **Target (y)**
- แบ่งชุดข้อมูลเป็นชุดฝึก (Train) และชุดทดสอบ (Test)
- ปรับข้อมูลให้อยู่ในรูปแบบมาตรฐานด้วย **StandardScaler**



```
# Step 4: Data Preparation

categorical_cols = ['gender', 'race/ethnicity', 'parental level of education', 'lunch', 'test preparation course']
encoder = LabelEncoder()
for col in categorical_cols:
    data[col] = encoder.fit_transform(data[col])

X = data.drop(columns=['math score'])
y = data['math score']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Build Neural Network Model

สร้างโมเดล Neural Network ด้วย Keras สำหรับปัญหา Regression

```
# Step 5: Build Neural Network Model
model = Sequential([
    Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
    Dense(32, activation='relu'),
    Dense(1, activation='linear')  # Linear activation for regression
])

model.compile(optimizer='adam', loss='mean_squared_error', metrics=['mae'])
```

Train the Model

ฝึกโมเดลด้วยข้อมูลชุดฝึก (Training Set) และประเมินผลด้วยข้อมูลชุดทดสอบ

(Validation Set)

```
# Step 6: Train the Model
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=50, batch_size=16, verbose=1)
```

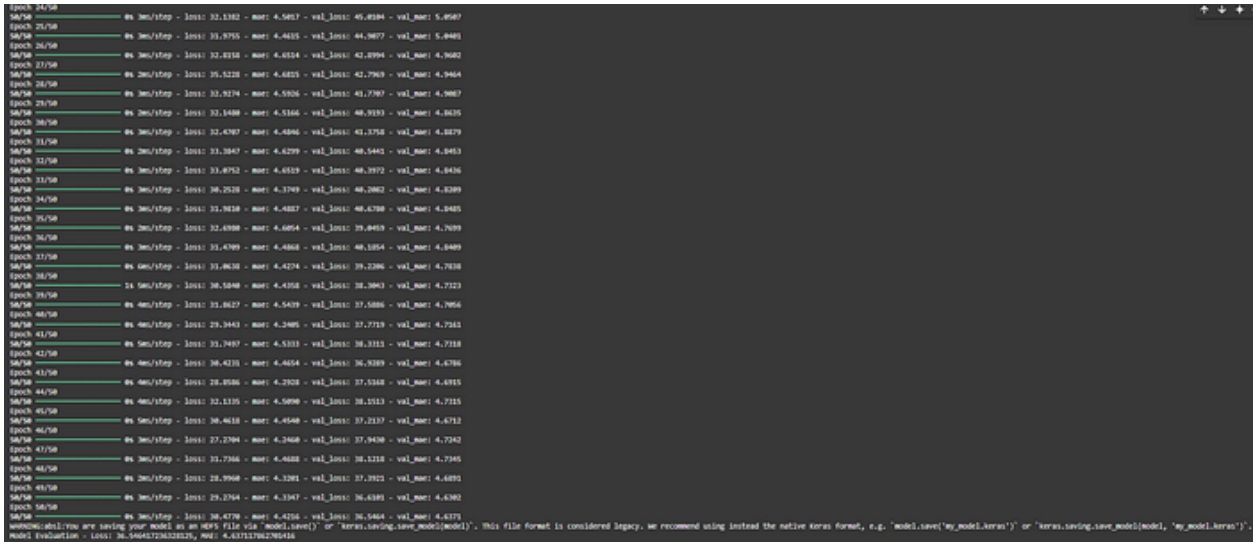
Evaluate the Model

ประเมินผลการทำงานของโมเดลด้วยค่า Loss และ Mean Absolute Error (MAE)

```
# Save the trained model
model.save('math_score_predictor.h5')
```

การตั้งค่าและการฝึกโมเดล

- **Optimizer**:
- ใช้ **Adam Optimizer** ซึ่งมีประสิทธิภาพในการปรับค่าการเรียนรู้อัตโนมัติ
- **Loss Function**:
- ใช้ `Mean Squared Error (MSE)` ซึ่งเป็นค่าความผิดพลาดเฉลี่ยกำลังสอง เหมาะสำหรับปัญหา Regression
- **Metrics**:
- ใช้ `Mean Absolute Error (MAE)` ในการประเมินค่าความผิดพลาด
- **Training**:
- ฝึกโมเดลเป็นเวลา 50 Epochs โดยใช้ Batch Size = 16
- ใช้ชุด Validation เพื่อติดตามค่าความผิดพลาดในแต่ละรอบการฝึก



ได้ผลการประเมินโมเดลบนชุด Test Set:

- `Loss` (MSE) = 36.55
- `MAE` = 4.64
- แสดงว่าโมเดลสามารถทำนายคะแนนคณิตศาสตร์ได้ โดยเฉลี่ยค่าผิดพลาดอยู่ที่ประมาณ ±4.64 คะแนน

Create Streamlit Application

พัฒนาแอปพลิเคชันสำหรับให้ผู้ใช้กรอกข้อมูล และแสดงผลลัพธ์ที่โมเดลทำนาย

```python
import streamlit as st
import numpy as np
import tensorflow as tf
from sklearn.preprocessing import LabelEncoder, StandardScaler
import pandas as pd

# โหลดโมเดลที่ฝึกมาแล้ว
model = tf.keras.models.load_model('math_score_predictor.h5')

# โหลดและเตรียมข้อมูลดิบแบบสำหรับ LabelEncoder และ Scaler
data = pd.read_csv("StudentsPerformance.csv")
categorical_cols = ['gender', 'race/ethnicity', 'parental level of education', 'lunch', 'test preparation course']

encoders = {}
for col in categorical_cols:
    le = LabelEncoder()
    data[col] = le.fit_transform(data[col])
    encoders[col] = le

scaler = StandardScaler()
X = data.drop(columns=['math score'])
scaler.fit(X)

# ฟังก์ชันตรวจสอบและเพิ่มค่าที่ไม่เคยเห็นมาก่อนใน LabelEncoder
def safe_transform(encoder, value):
    if value not in encoder.classes_:
        encoder.classes_ = np.append(encoder.classes_, value)
    return encoder.transform([value])[0]

# แอป Streamlit
st.title("ตัวทำนายคะแนนคณิตศาสตร์")

# ฟอร์มกรอกข้อมูลจากผู้ใช้
gender = st.selectbox("เพศ", list(encoders['gender'].classes_))
race = st.selectbox("เชื้อชาติ/กลุ่มชาติพันธุ์", list(encoders['race/ethnicity'].classes_))
parental_education = st.selectbox("ระดับการศึกษาของผู้ปกครอง", list(encoders['parental level of education'].classes_))
lunch = st.selectbox("ประเภทมื้ออาหาร", list(encoders['lunch'].classes_))
test_preparation = st.selectbox("หลักสูตรเตรียมสอบ", list(encoders['test preparation course'].classes_))
reading_score = st.slider("คะแนนการอ่าน", 0, 100, 50)
writing_score = st.slider("คะแนนการเขียน", 0, 100, 50)
```

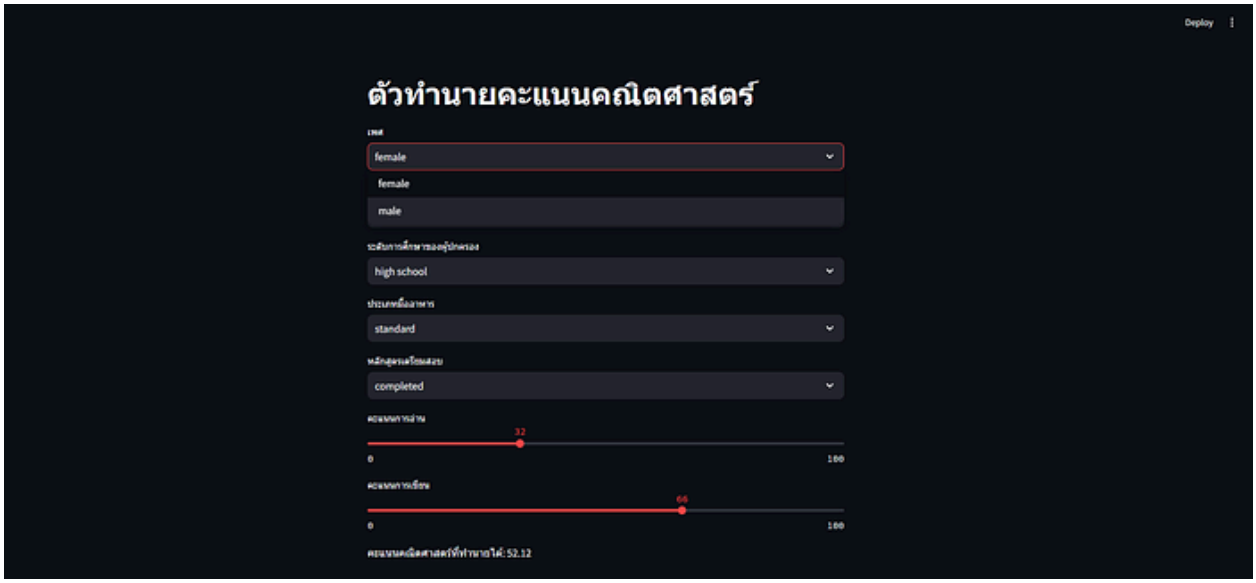
```python
writing_score = st.slider("คะแนนการเขียน", 0, 100, 50)

# การประมวลผลข้อมูลจากผู้ใช้
user_data = np.array([
    safe_transform(encoders['gender'], gender),
    safe_transform(encoders['race/ethnicity'], race),
    safe_transform(encoders['parental level of education'], parental_education),
    safe_transform(encoders['lunch'], lunch),
    safe_transform(encoders['test preparation course'], test_preparation),
    reading_score,
    writing_score
]).reshape(1, -1)

user_data = scaler.transform(user_data)

# ทำนายคะแนน
prediction = model.predict(user_data)
st.write(f"คะแนนคณิตศาสตร์ที่ทำนายได้: {prediction[0][0]:.2f}")
```

แสดงหน้าเว็บอินเทอร์เฟซของแอปพลิเคชัน



สรุปและข้อเสนอแนะ

โมเดลนี้เป็นตัวอย่างที่ดีของการใช้ **Neural Networks** สำหรับการพยากรณ์คะแนนทางคณิตศาสตร์จากข้อมูลเกี่ยวกับนักเรียน โดยการพัฒนาระบบมีการออกแบบที่เหมาะสมกับปัญหา Regression และสามารถนำไปปรับปรุงเพิ่มเติมได้

จุดเด่นของโมเดล

1. **ความแม่นยำที่เหมาะสม**: โมเดลแสดงค่าความผิดพลาด Mean Absolute Error (MAE) ที่ต่ำ (ประมาณ 4.63) ซึ่งบ่งบอกว่าการพยากรณ์มีความแม่นยำในระดับดี
2. **การออกแบบโมเดลที่ยืดหยุ่น**: โครงสร้างโมเดลที่ใช้ Dense Layers ทำให้สามารถนำไปปรับปรุงต่อยอดได้ง่าย
3. **กระบวนการเรียนรู้**: การเลือก Optimizer (Adam) และ Loss Function (MSE) เหมาะสมกับปัญหา และช่วยให้การเรียนรู้รวดเร็วและเสถียร

ข้อเสนอแนะสำหรับการปรับปรุง

การเพิ่มประสิทธิภาพโมเดล:

- ใช้เทคนิค **Feature Engineering** เพิ่มเติม เพื่อเน้นความสำคัญของข้อมูลบางส่วน เช่น ความสัมพันธ์ระหว่างคะแนนวิชาอื่น
- เพิ่มหรือปรับจำนวน Neurons และ Layers ให้เหมาะสมกับความซับซ้อนของข้อมูล

1. **การป้องกัน Overfitting**:

- ใช้ **Dropout Layers** หรือ **Regularization** เพื่อลดปัญหา Overfitting
- เพิ่มจำนวนข้อมูลด้วย **Data Augmentation** เพื่อให้โมเดลมีความยืดหยุ่นมากขึ้น