# 20250417_01

April 17, 2025

[69]:

```
# Try to predict final score
import pandas as pd
import numpy as np

from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, root_mean_squared_error

# Making dataset
df =pd.DataFrame({'student_id':[1, 2, 3, 4, 5],
                  'math_score':[75, 88, 95, 65, 50],
                  'english_score':[82, 79, 91, 70, 60],
                  'gender':['F', 'M', 'M', 'F', 'F'],
                  'school_type':['public', 'private', 'private', 'public',␣
 ↪'public'],
                  'final_score':[80, 85, 90, 70, 60]})

df.head()
```

[69]:

```
   student_id  math_score  english_score gender school_type  final_score
0           1          75             82      F      public           80
1           2          88             79      M     private           85
2           3          95             91      M     private           90
3           4          65             70      F      public           70
4           5          50             60      F      public           60
```

[71]:

```
# Feature
X = df.drop(columns = ['student_id', 'final_score'])
X.head()
```

[71]:

```
   math_score  english_score gender school_type
0          75             82      F      public
1          88             79      M     private
2          95             91      M     private
3          65             70      F      public
4          50             60      F      public
```

[73]:

```
# Target, in this case, final score
y = df['final_score']
y.head()
```

[73]:

```
0    80
1    85
2    90
3    70
4    60
Name: final_score, dtype: int64
```

[75]:

```
# 60% data for training, 40% data for testing, and 42 just for fun.
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.4,␣
 ↪random_state = 42)
```

[77]:

```
# Classify
num_cols = ['math_score', 'english_score']
cat_cols = ['gender', 'school_type']

# Setting what for each class to do
preprocessor = ColumnTransformer([('num', StandardScaler(), num_cols),
                                  ('cat', OneHotEncoder(drop = 'first',␣
 ↪sparse_output = False), cat_cols)])
```

[79]:

```
# Fit and transform
X_train_processed = preprocessor.fit_transform(X_train)
```

[81]:

```
# Creating model
model = LinearRegression() # Initailizing

# Use X_train_processed and y_train as training data
model.fit(X_train_processed, y_train)
```

[81]:

```
LinearRegression()
```

[83]:

```
# Testing, but we need to process X first
X_test_processed = preprocessor.transform(X_test)

# Predicting
y_predict = model.predict(X_test_processed)

# Comparing
print('Predict outcome :', y_predict)
print('Real score :', y_test.values)
```

```
Predict outcome : [80.70523752 60.09947219]
Real score : [85 60]
```

[87]:

```
# Evaluating model

mae = mean_absolute_error(y_test, y_predict) # Average error
rmse = root_mean_squared_error(y_test, y_predict) # Enhance the effect of␣
 ↪outliers to error

print('MAE :', mae)
print('RMSE :', rmse)
```

```
MAE : 2.197117336581403
RMSE : 3.0376701200830367
```