20250419_01

April 19, 2025

[39]:
```python
import pandas as pd
import numpy as np

# Modeling and preprocessing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# Evaluation
from sklearn.metrics import mean_absolute_error, root_mean_squared_error,␣
↪r2_score

# 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]})
```

[41]:
```python
# Feature engineering
num_cols = ['math_score', 'english_score']
cat_cols = ['gender', 'school_type']

# Checking if any potential problems
{col: df[col].unique() for col in cat_cols}
```

[41]:
```
{'gender': array(['F', 'M'], dtype=object),
 'school_type': array(['public', 'private'], dtype=object)}
```

[43]:
```python
# preprocessing
preprocessor = ColumnTransformer([('num', StandardScaler(), num_cols),
                                  ('cat', OneHotEncoder(drop = 'first'),␣
↪cat_cols)])
```

1 First model : Linear Regression

[46]:
```
# Creating pipeline
pipe_lr = Pipeline([('Preprocessing', preprocessor), ('model',␣
 ↪LinearRegression())])

# Making training and test sets
X = df.drop(columns = ['student_id', 'final_score'])
y = df['final_score']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.4,␣
 ↪random_state = 42)

# Training model
pipe_lr.fit(X_train, y_train)

# Prediction
y_pred_lr = pipe_lr.predict(X_test)
```

[48]:
```
# Evaluating model
mae_lr = mean_absolute_error(y_test, y_pred_lr)
rmse_lr = root_mean_squared_error(y_test, y_pred_lr)
r2_lr = r2_score(y_test, y_pred_lr)

print('Linear Regression')
print('MAE: ', mae_lr)
print('RMSE: ', rmse_lr)
print('R²: ', r2_lr)
```

```
Linear Regression
MAE:  2.197117336581403
RMSE:  3.0376701200830367
R²:  0.9409443855459502
```

2 Second model : Decision Tree

[51]:
```
# Creating another pipeline
pipe_tree = Pipeline([('Preprocessing', preprocessor),
                      ('model', DecisionTreeRegressor(random_state = 42))])

# Training model
pipe_tree.fit(X_train, y_train)

# Prediction
y_pred_tree = pipe_tree.predict(X_test)
```

[53]:
```
# Evaluating model
mae_tree = mean_absolute_error(y_test, y_pred_tree)
```

```python
rmse_tree = root_mean_squared_error(y_test, y_pred_tree)
r2_tree = r2_score(y_test, y_pred_tree)

print('Decision Tree')
print('MAE: ', mae_tree)
print('RMSE: ', rmse_tree)
print('R²: ', r2_tree)
```

```
Decision Tree
MAE:  7.5
RMSE:  7.905694150420948
R²:  0.6
```

In this dataset, the linear regression model achieved significantly better performance than the decision tree regressor (MAE = 2.20 vs 7.50, R^2 = 0.94 vs 0.60). The tree-based model likely overfit due to the small dataset and limited feature diversity. For this problem, a linear model appears to be more appropriate.