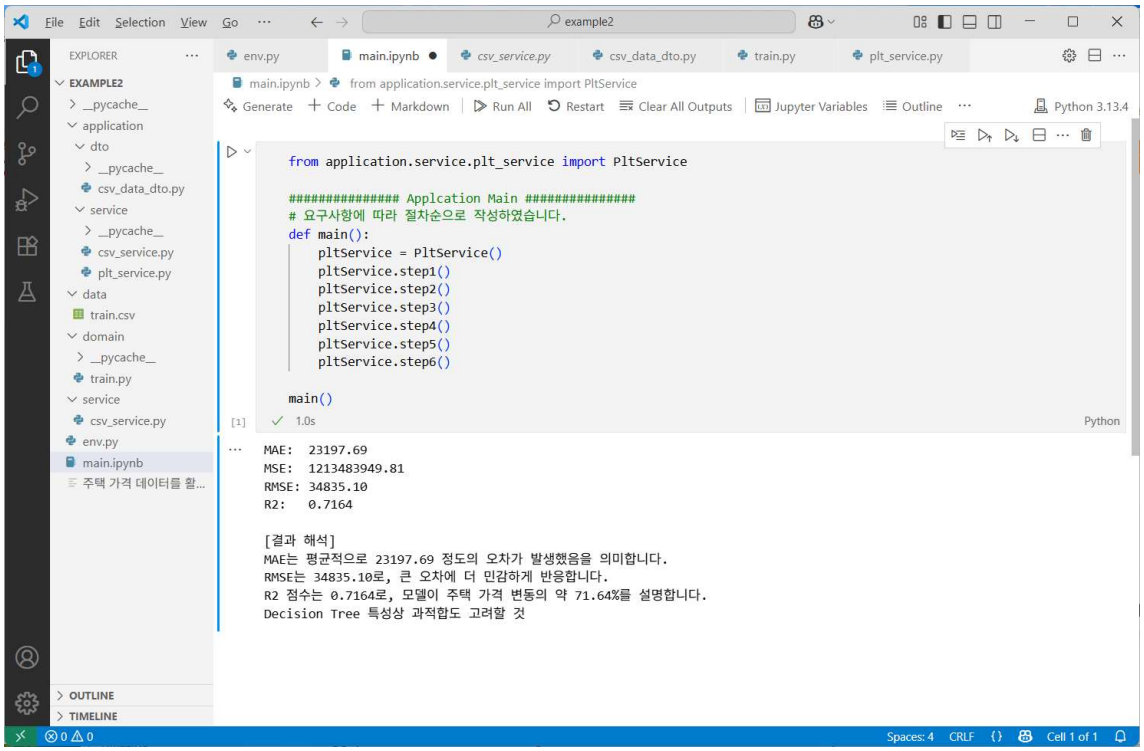
# 주택 가격 데이터를 활용한 회귀 모델 학습 및 예측

26기_장상훈

## 프로그램 실행 결과




## 요구사항 1. 결측치 처리

```python
# 1. 결측치 처리
def step1(self):
    missing_ratio = self.entity.df.isnull().mean()
    # 30% 이상 결측치 열 삭제
    cols_to_drop = missing_ratio[missing_ratio > 0.3].index
    self.entity.df = self.entity.df.drop(columns=cols_to_drop)
    # LotFrontage 평균으로 대체
    self.entity.df['LotFrontage'] = self.entity.df['LotFrontage'].fillna(self.entity.df['LotFrontage'].mean())
    # 나머지 결측치는 삭제
    self.entity.df = self.entity.df.dropna()
```

## 요구사항 2. 범주형 데이터 인코딩

```python
# 2. 범주형 데이터 인코딩
def step2(self):
    self.entity.df = pd.get_dummies(self.entity.df)
```

요구사항 3. 불필요한 열 제거

```
# 3. 불필요한 열 제거
def step3(self):
    self.entity.df = self.entity.df.drop(columns=['Id'])
```

요구사항 4. 학습/테스트 데이터 분리

```
# 4. 학습/테스트 데이터 분리
def step4(self):
    X = self.entity.df.drop(columns=['SalePrice'])
    y = self.entity.df['SalePrice']
    self.X_train, self.X_test, self.y_train, self.y_test = train_test_split(
        X,
        y,
        test_size=0.2, # << 여기 테스트 데이터 20%
        random_state=42
        )
    self.model = DecisionTreeRegressor(random_state=42)
    self.model.fit(self.X_train, self.y_train)
```

모델 평가

```
# 모델 평가
def step5(self):
    y_pred = self.model.predict(self.X_test)

    self.mae = metrics.mean_absolute_error(self.y_test, y_pred)
    self.mse = metrics.mean_squared_error(self.y_test, y_pred)
    self.rmse = np.sqrt(self.mse)
    self.r2 = metrics.r2_score(self.y_test, y_pred)

    print(f"MAE:  {self.mae:.2f}")
    print(f"MSE:  {self.mse:.2f}")
    print(f"RMSE: {self.rmse:.2f}")
    print(f"R2:   {self.r2:.4f}")
```

요구사항 6. 평가 결과 해석

```python
# 평가 결과 해석
def step6(self):
    print("\n[결과 해석]")
    print(f"MAE는 평균적으로 {self.mae:.2f} 정도의 오차가 발생했음을 의미합니다.")
    print(f"RMSE는 {self.rmse:.2f}로, 큰 오차에 더 민감하게 반응합니다.")
    print(f"R2 점수는 {self.r2:.4f}로, 모델이 주택 가격 변동의 약 {self.r2*100:.2f}%를 설명합니다.")
    print("Decision Tree 특성상 과적합도 고려할 것")
```