Task 3 — Linear Regression (All-in-one)

Contents

---

1) Overview & objective

Implement and understand simple & multiple linear regression with scikit-learn. Preprocess data, split into train/test, train a LinearRegression model, evaluate it (MAE, MSE, R^2), and visualize results.

---

2) Dataset

Dataset path to be used:

```
/Users/pratham/Downloads/Housing 2.csv
```

---

3) Step-by-step instructions

1. Create a new GitHub repo for Task-3.
2. Add the dataset at `/Users/pratham/Downloads/Housing 2.csv`.
3. Install requirements: `pip install -r requirements.txt`.
4. Run `python linear_regression_task3.py`.

---

4) Full Python code (script: `linear_regression_task3.py`)

```python
# linear_regression_task3.py
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
import joblib

# ------------- Configuration -------------
DATA_PATH = '/Users/pratham/Downloads/Housing 2.csv'  # dataset path
TARGET = 'price'
RANDOM_STATE = 42
TEST_SIZE = 0.2

# ------------- Load data -------------
df = pd.read_csv(DATA_PATH)
print('Dataset shape:', df.shape)
print(df.head())

# Detect numeric and categorical columns
numeric_cols = df.select_dtypes(include=['int64',
'float64']).columns.tolist()
if TARGET in numeric_cols:
    numeric_cols.remove(TARGET)
cat_cols = df.select_dtypes(include=['object', 'category']).columns.tolist()

# Handle missing values
for col in numeric_cols:
    if df[col].isnull().sum() > 0:
        df[col].fillna(df[col].median(), inplace=True)
for col in cat_cols:
    if df[col].isnull().sum() > 0:
        df[col].fillna(df[col].mode()[0], inplace=True)

# Features / target
X = df.drop(columns=[TARGET])
y = df[TARGET]

# Column transformer
preprocessor = ColumnTransformer([
    ('num', StandardScaler(), numeric_cols),
    ('cat', OneHotEncoder(handle_unknown='ignore', sparse=False), cat_cols)
])

# Pipeline
model = Pipeline([
    ('preproc', preprocessor),
    ('lr', LinearRegression())
])

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=TEST_SIZE, random_state=RANDOM_STATE)
```

```
# Fit
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Metrics
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print('\nEvaluation on test set:')
print(f'MAE: {mae:.3f}')
print(f'MSE: {mse:.3f}')
print(f'RMSE: {np.sqrt(mse):.3f}')
print(f'R^2: {r2:.3f}')

# Save model
joblib.dump(model, 'linear_regression_pipeline.joblib')
print('\nSaved pipeline to linear_regression_pipeline.joblib')

# Coefficients
try:
    num_features = numeric_cols
    cat_encoder = model.named_steps['preproc'].named_transformers_['cat']
    cat_feature_names = list(cat_encoder.get_feature_names_out(cat_cols)) if
cat_cols else []
    feature_names = num_features + cat_feature_names
    coefs = model.named_steps['lr'].coef_
    coef_df = pd.DataFrame({'feature': feature_names, 'coefficient': coefs})
    coef_df = coef_df.sort_values(by='coefficient', key=abs, ascending=False)
    coef_df.to_csv('coefficients.csv', index=False)
    print('\nSaved coefficients.csv')
except Exception as e:
    print('\nCould not extract coefficients cleanly:', e)

print('\nScript finished.')
```

---

5) Model evaluation & saving

- `linear_regression_pipeline.joblib` — trained model
- `coefficients.csv` — feature coefficients

---

6) README

```
# Task 3 — Linear Regression

## Overview
Implementation of linear regression on housing dataset.

## Dataset
Path: `/Users/pratham/Downloads/Housing 2.csv`

## Files
- `linear_regression_task3.py`
- `linear_regression_pipeline.joblib`
- `coefficients.csv`

## Requirements
```

scikit-learn pandas numpy matplotlib joblib

```
## Run
```bash
python linear_regression_task3.py
```
```

---

7) Interview Questions & Answers

1. What assumptions does linear regression make?
Linearity, independence, homoscedasticity, normality of errors, low multicollinearity.

2. How do you interpret the coefficients?
Each coefficient = expected change in target when predictor increases by one unit, keeping others constant.

3. What is R^2 score and its significance?
Proportion of variance explained by model. Closer to 1 → better fit.

4. When would you prefer MSE over MAE?
MSE penalizes large errors more. Useful when large errors are costly. MAE is robust to outliers.

5. How do you detect multicollinearity?
Correlation matrix, VIF (Variance Inflation Factor). VIF > 10 indicates strong multicollinearity.

6. What is the difference between simple and multiple regression?
Simple regression: one predictor. Multiple regression: more than one predictor.

7. Can linear regression be used for classification?
Not directly. Logistic regression is used for classification.

8. What happens if you violate regression assumptions?
Biased coefficients, wrong inference, poor predictions, unreliable intervals.
