20250608_01

June 8, 2025

[1]:

```python
# Load dataset
import pandas as pd

data = pd.read_csv('melb_data.csv')
data.head()
```

[1]:

```
       Suburb           Address  Rooms Type      Price Method SellerG  \
0  Abbotsford      85 Turner St      2    h  1480000.0      S  Biggin
1  Abbotsford   25 Bloomburg St      2    h  1035000.0      S  Biggin
2  Abbotsford      5 Charles St      3    h  1465000.0     SP  Biggin
3  Abbotsford  40 Federation La      3    h   850000.0     PI  Biggin
4  Abbotsford       55a Park St      4    h  1600000.0     VB  Nelson

        Date  Distance  Postcode  …  Bathroom  Car  Landsize  BuildingArea  \
0  3/12/2016       2.5    3067.0  …       1.0  1.0     202.0           NaN
1  4/02/2016       2.5    3067.0  …       1.0  0.0     156.0          79.0
2  4/03/2017       2.5    3067.0  …       2.0  0.0     134.0         150.0
3  4/03/2017       2.5    3067.0  …       2.0  1.0      94.0           NaN
4  4/06/2016       2.5    3067.0  …       1.0  2.0     120.0         142.0

   YearBuilt  CouncilArea Lattitude  Longtitude             Regionname  \
0        NaN        Yarra  -37.7996    144.9984  Northern Metropolitan
1     1900.0        Yarra  -37.8079    144.9934  Northern Metropolitan
2     1900.0        Yarra  -37.8093    144.9944  Northern Metropolitan
3        NaN        Yarra  -37.7969    144.9969  Northern Metropolitan
4     2014.0        Yarra  -37.8072    144.9941  Northern Metropolitan

  Propertycount
0        4019.0
1        4019.0
2        4019.0
3        4019.0
4        4019.0

[5 rows x 21 columns]
```

[2]:

```
# Set features and target
X = data.drop('Price', axis = 1)
y = data['Price']
```

[3]:

```
# Split features into numeric and categorical
# Add .columns to only return the col names, otherwise it will return the whole␣
  ↪dataset
num_cols = X.select_dtypes(include = ['int64', 'float64']).columns
cat_cols = X.select_dtypes(include = ['object']).columns
```

[4]:

```
# Handle missing values
from sklearn.impute import SimpleImputer

# Initialize the imputer with mean
imputer_num = SimpleImputer(strategy = 'mean')

# Fit and transform the numeric data
X_num = pd.DataFrame(imputer_num.fit_transform(X[num_cols]), columns = num_cols)

# Create dummy variables for categorical columns, including NaNs
X_cat = pd.get_dummies(X[cat_cols], dummy_na = True)
```

[5]:

```
# Put them back together
X_processed = pd.concat([X_num, X_cat], axis = 1)
```

[6]:

```
# Now we try the clean dataset
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression

# Split data
X_train, X_test, y_train, y_test = train_test_split(X_processed, y,␣
  ↪random_state = 42)

# Train model
model = LinearRegression()
model.fit(X_train, y_train)

# Evaluate, which is default to using R2
print("Model R² Score:", model.score(X_test, y_test))
```

```
Model R² Score: -2.0371647495946337
```

[8]:

```
# Oof, now we  drop some cols
high_card_cols = [col for col in cat_cols if X[col].nunique() > 50]
X = X.drop(columns = high_card_cols)
cat_cols = [col for col in cat_cols if col not in high_card_cols]

X_cat = pd.get_dummies(X[cat_cols], dummy_na = True)
```

```python
X_processed = pd.concat([X_num, X_cat], axis = 1)

X_train, X_test, y_train, y_test = train_test_split(X_processed, y, random_state = 42)

model_retry = LinearRegression()
model_retry.fit(X_train, y_train)

# Evaluate, which is default to using R2
print("Model R² Score:", model_retry.score(X_test, y_test))
```

```
Model R² Score: 0.6508737168099696
```