3rd-proj-major

June 26, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.ensemble import RandomForestRegressor
from sklearn.pipeline import Pipeline
from sklearn.metrics import mean_squared_error
```

[2]:
```
# Correct file paths using raw strings
train_path = r"D:\cbnst_lab\Capstone Data Analytics\Dmart sales analysis\train␣
 ↪(1).csv"
test_path = r"D:\cbnst_lab\Capstone Data Analytics\Dmart sales analysis\test␣
 ↪(1).csv"
```

[3]:
```
# Load the data
train_df = pd.read_csv(train_path)
test_df = pd.read_csv(test_path)
```

[4]:
```
# Data exploration
print("Training Data Overview:\n", train_df.head())
print("Test Data Overview:\n", test_df.head())
print("Training Data Info:\n", train_df.info())
```

```
Training Data Overview:
   Item_Identifier  Item_Weight Item_Fat_Content  Item_Visibility  \
0           FDA15         9.30          Low Fat         0.016047
1           DRC01         5.92          Regular         0.019278
2           FDN15        17.50          Low Fat         0.016760
3           FDX07        19.20          Regular         0.000000
4           NCD19         8.93          Low Fat         0.000000

              Item_Type  Item_MRP Outlet_Identifier  \
0                 Dairy  249.8092            OUT049
1           Soft Drinks   48.2692            OUT018
2                  Meat  141.6180            OUT049
```

```
3  Fruits and Vegetables  182.0950            OUT010
4              Household   53.8614            OUT013

   Outlet_Establishment_Year Outlet_Size Outlet_Location_Type  \
0                       1999      Medium               Tier 1
1                       2009      Medium               Tier 3
2                       1999      Medium               Tier 1
3                       1998         NaN               Tier 3
4                       1987        High               Tier 3

         Outlet_Type  Item_Outlet_Sales
0  Supermarket Type1          3735.1380
1  Supermarket Type2           443.4228
2  Supermarket Type1          2097.2700
3      Grocery Store           732.3800
4  Supermarket Type1           994.7052
Test Data Overview:
   Item_Identifier  Item_Weight Item_Fat_Content  Item_Visibility    Item_Type
\
0           FDW58       20.750          Low Fat         0.007565  Snack Foods
1           FDW14        8.300              reg         0.038428        Dairy
2           NCN55       14.600          Low Fat         0.099575       Others
3           FDQ58        7.315          Low Fat         0.015388  Snack Foods
4           FDY38          NaN          Regular         0.118599        Dairy

   Item_MRP Outlet_Identifier  Outlet_Establishment_Year Outlet_Size  \
0  107.8622            OUT049                       1999      Medium
1   87.3198            OUT017                       2007         NaN
2  241.7538            OUT010                       1998         NaN
3  155.0340            OUT017                       2007         NaN
4  234.2300            OUT027                       1985      Medium

  Outlet_Location_Type        Outlet_Type
0               Tier 1  Supermarket Type1
1               Tier 2  Supermarket Type1
2               Tier 3      Grocery Store
3               Tier 2  Supermarket Type1
4               Tier 3  Supermarket Type3
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 12 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Item_Identifier            8523 non-null   object
 1   Item_Weight                7060 non-null   float64
 2   Item_Fat_Content           8523 non-null   object
 3   Item_Visibility            8523 non-null   float64
 4   Item_Type                  8523 non-null   object
```

```
 5   Item_MRP                   8523 non-null   float64
 6   Outlet_Identifier          8523 non-null   object
 7   Outlet_Establishment_Year  8523 non-null   int64
 8   Outlet_Size                6113 non-null   object
 9   Outlet_Location_Type       8523 non-null   object
 10  Outlet_Type                8523 non-null   object
 11  Item_Outlet_Sales          8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
Training Data Info:
 None
```

[5]:
```
# Combine both dataframes for preprocessing
data = pd.concat([train_df, test_df], ignore_index=True)
```

[6]:
```
# Handle missing values
data['Item_Weight'].fillna(data['Item_Weight'].mean(), inplace=True)
data['Outlet_Size'].fillna(data['Outlet_Size'].mode()[0], inplace=True)
```

[7]:
```
# Feature engineering
data['Item_Visibility'] = data['Item_Visibility'].replace(0, np.nan)
data['Item_Visibility'].fillna(data['Item_Visibility'].mean(), inplace=True)
```

[8]:
```
data['Years_Operational'] = 2024 - data['Outlet_Establishment_Year']
data.drop(columns=['Outlet_Establishment_Year'], inplace=True)
```

[9]:
```
# Define categorical and numeric columns
categorical_cols = ['Item_Fat_Content', 'Outlet_Location_Type', 'Outlet_Type',␣
 ↪'Outlet_Size', 'Outlet_Identifier', 'Item_Type']
numeric_cols = ['Item_Weight', 'Item_Visibility', 'Item_MRP',␣
 ↪'Years_Operational']
```

[10]:
```
# Split the data back into train and test sets
train = data[~data['Item_Outlet_Sales'].isna()]
test = data[data['Item_Outlet_Sales'].isna()].
 ↪drop(columns=['Item_Outlet_Sales'])
```

[11]:
```
# Prepare features and target
X = train.drop(columns=['Item_Outlet_Sales'])
y = train['Item_Outlet_Sales']
```

[12]:
```
# Create preprocessing pipeline
preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(), numeric_cols),
        ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_cols)
    ])
```

[13]:
```
# Model training pipeline using RandomForestRegressor
model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', RandomForestRegressor(n_estimators=100, random_state=42))
])
```

[14]:
```
# Train-test split
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[15]:
```
# Fit the model
model.fit(X_train, y_train)
```

[15]:
```
Pipeline(steps=[('preprocessor',
                 ColumnTransformer(transformers=[('num', StandardScaler(),
                                                  ['Item_Weight',
                                                   'Item_Visibility',
                                                   'Item_MRP',
                                                   'Years_Operational']),
                                                 ('cat',
OneHotEncoder(handle_unknown='ignore'),
                                                  ['Item_Fat_Content',
                                                   'Outlet_Location_Type',
                                                   'Outlet_Type', 'Outlet_Size',
                                                   'Outlet_Identifier',
                                                   'Item_Type'])])),
                ('regressor', RandomForestRegressor(random_state=42))])
```

[16]:
```
# Prediction
y_pred = model.predict(X_val)
mse = mean_squared_error(y_val, y_pred)
print("Validation Mean Squared Error:", mse)
```

```
Validation Mean Squared Error: 1156478.0556633975
```

[17]:
```
# Final training on the full training data
model.fit(X, y)
test_predictions = model.predict(test)
```

[18]:
```
# Prepare submission
submission = pd.DataFrame({
    'Item_Identifier': test_df['Item_Identifier'],
    'Outlet_Identifier': test_df['Outlet_Identifier'],
    'Item_Outlet_Sales': test_predictions
})
```

[19]:
```
# Save submission file
submission.to_csv('submission.csv', index=False)
print("Submission file saved as 'submission.csv'")
```

```
Submission file saved as 'submission.csv'
```

[20]:
```
sub_path = r"C:\Users\jhasa\submission.csv"
```

[21]:
```
sub_df = pd.read_csv(sub_path)
```

[22]:
```
print("Submission Data Overview:\n", sub_df.head())
```

```
Submission Data Overview:
   Item_Identifier Outlet_Identifier  Item_Outlet_Sales
0           FDW58            OUT049        1704.621108
1           FDW14            OUT017        1175.496532
2           NCN55            OUT010         616.284454
3           FDQ58            OUT017        2227.254134
4           FDY38            OUT027        6357.138296
```

[ ]:
```

```