product-details-prediction

June 28, 2024

[ ]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
```

[1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[ ]:
```
products= pd.read_csv('/content/drive/MyDrive/ml proj/Message Group - Product.
↪csv')
```

[ ]:
```
x=products[['MRP','SellPrice']]
y=products['Discount']
```

[ ]:
```
# Assuming 'products' is your DataFrame
products = products.replace('#REF!', np.nan).dropna()  # Replace '#REF!' with␣
↪NaN and drop rows with NaN values

x = products[['MRP', 'SellPrice']]

# Extract numerical discount values and handle non-numerical values
products['Discount'] = products['Discount'].str.extract('(\d+)').astype(float)
y = products['Discount']

k = 3
knn = KNeighborsClassifier(n_neighbors=k)
knn.fit(x, y)
```

[ ]:
```
KNeighborsClassifier(n_neighbors=3)
```

[ ]:
```
new_data = np.array([[3900,3120]])
prediction = knn.predict(new_data)
```

```
if prediction<50:
    print("price is high")
elif prediction>50 and prediction<100:
    print("price is medium")
else:
    print("price is low")
```

```
price is high
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but KNeighborsClassifier was fitted with feature
names
  warnings.warn(
```

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Assuming 'x' and 'y' are already defined from your previous code
# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
 ↪random_state=42)

# Create and fit the LinearRegression model
model = LinearRegression()
model.fit(x_train, y_train)

# Example prediction for new data
new_data = np.array([[3900, 3120]])
prediction = model.predict(new_data)
print("Prediction for new data:", prediction)
```

```
Prediction for new data: [31.59923834]
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

[3]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
import numpy as np
```

```
# Load the dataset and assign it to the variable 'products'
products = pd.read_csv("/content/drive/MyDrive/ml proj/Message Group - Product.
 ↪csv")

# Preprocess the data
x=products[['MRP','SellPrice']]
y=products['Discount']

# Check for non-numeric values and handle them
# For demonstration, we'll replace '#REF!' with NaN and then drop those rows
x = x.replace('#REF!', pd.NA).dropna()
y = y[x.index]  # Update y to match the indices of the cleaned x

# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
 ↪random_state=42)

# Train Decision Tree model
model = DecisionTreeClassifier()
model.fit(x_train, y_train)  # Now this should work without error
y_pred = model.predict(x_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Classification Report:\n", report)

# Function to get user input and predict output
def get_user_input():
    mrp = float(input("Enter MRP: "))
    sell_price = float(input("Enter Sell Price: "))
    return np.array([[mrp, sell_price]])

# Get user input and predict
user_input = get_user_input()
user_prediction = model.predict(user_input)

# Adjust output based on the problem you are solving
print("Prediction:", user_prediction[0])
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
```

```
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0
in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0
in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0
in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))

Accuracy: 0.9736263736263736
Classification Report:
              precision    recall  f1-score   support

     10% off       0.97      1.00      0.99       260
     12% off       1.00      1.00      1.00         2
     15% off       0.00      0.00      0.00         3
     20% off       1.00      0.97      0.99       104
     25% off       1.00      0.33      0.50         3
     27% off       0.00      0.00      0.00         1
     30% off       0.95      0.99      0.97       168
     33% off       0.50      1.00      0.67         1
     35% off       1.00      0.43      0.60         7
     40% off       0.99      0.96      0.97        81
     45% off       1.00      1.00      1.00         3
     47% off       1.00      1.00      1.00         1
      5% off       1.00      0.75      0.86        12
     50% off       1.00      1.00      1.00       204
     51% off       0.67      1.00      0.80         2
     53% off       0.50      1.00      0.67         1
     54% off       0.00      0.00      0.00         1
```

```
     55% off       0.00      0.00      0.00         1
     60% off       1.00      0.97      0.99        39
     70% off       0.94      1.00      0.97        15
     71% off       0.00      0.00      0.00         0
     74% off       0.00      0.00      0.00         1

    accuracy                           0.97       910
   macro avg       0.66      0.66      0.63       910
weighted avg       0.97      0.97      0.97       910

Enter MRP: 30
Enter Sell Price: 30
Prediction: 10% off
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but DecisionTreeClassifier was fitted with feature
names
  warnings.warn(
```

[4]:
```
from sklearn.ensemble import RandomForestClassifier
```

[5]:
```
RFC = RandomForestClassifier(random_state=0)
RFC.fit(x_train,y_train)
```

[5]:
```
RandomForestClassifier(random_state=0)
```

[10]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
import numpy as np

# Load the dataset and assign it to the variable 'products'
products = pd.read_csv("/content/drive/MyDrive/ml proj/Message Group - Product.
 ↪csv")

# Preprocess the data
x = products[['MRP', 'SellPrice']]
y = products['Discount']

# Check for non-numeric values and handle them
# For demonstration, we'll replace '#REF!' with NaN and then drop those rows
x = x.replace('#REF!', pd.NA).dropna()
y = y[x.index]  # Update y to match the indices of the cleaned x

# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
 ↪random_state=42)
```

```
# Train Random Forest model
model = RandomForestClassifier()
model.fit(x_train, y_train)  # Now this should work without error
y_pred = model.predict(x_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)

print("Accuracy:", accuracy)
print("Classification Report:\n", report)

# Function to get user input and predict output
def get_user_input():
    mrp = float(input("Enter MRP: "))
    sell_price = float(input("Enter Sell Price: "))
    return np.array([[mrp, sell_price]])

# Get user input and predict
user_input = get_user_input()
user_prediction = model.predict(user_input)

# Adjust output based on the problem you are solving
print("Prediction:", user_prediction[0])
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0
in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0
in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
```

```
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Recall and F-score are ill-defined and being set to 0.0
in labels with no true samples. Use `zero_division` parameter to control this
behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
Accuracy: 0.9725274725274725
Classification Report:
              precision    recall  f1-score   support

     10% off       0.97      0.99      0.98       260
     12% off       1.00      1.00      1.00         2
     15% off       0.00      0.00      0.00         3
     20% off       0.99      0.97      0.98       104
     25% off       1.00      0.33      0.50         3
     27% off       0.00      0.00      0.00         1
     30% off       0.94      1.00      0.97       168
     33% off       1.00      1.00      1.00         1
     35% off       1.00      0.43      0.60         7
     40% off       0.97      0.96      0.97        81
     45% off       1.00      1.00      1.00         3
     47% off       1.00      1.00      1.00         1
      5% off       0.80      0.67      0.73        12
     50% off       1.00      1.00      1.00       204
     51% off       0.67      1.00      0.80         2
     53% off       1.00      1.00      1.00         1
     54% off       0.00      0.00      0.00         1
     55% off       0.00      0.00      0.00         1
     60% off       1.00      1.00      1.00        39
     70% off       1.00      1.00      1.00        15
     71% off       0.00      0.00      0.00         0
     74% off       0.00      0.00      0.00         1

    accuracy                           0.97       910
   macro avg       0.70      0.65      0.66       910
weighted avg       0.97      0.97      0.97       910
```

```
Enter MRP: 30
Enter Sell Price: 30
Prediction: 10% off
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but RandomForestClassifier was fitted with feature
names
  warnings.warn(
```