logistic-candy

March 21, 2024

[1]:
```python
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

[2]:
```python
df = pd.read_csv("logistic-candy-data.csv")
df
```

[2]:
```
                competitorname  chocolate  fruity  caramel  peanutyalmondy  \
0                    100 Grand          1       0        1               0
1                 3 Musketeers          1       0        0               0
2                     One dime          0       0        0               0
3                  One quarter          0       0        0               0
4                    Air Heads          0       1        0               0
..                          …       …     …      …              …
80                   Twizzlers          0       1        0               0
81                    Warheads          0       1        0               0
82        WelchÕs Fruit Snacks          0       1        0               0
83  WertherÕs Original Caramel          0       0        1               0
84                    Whoppers          1       0        0               0

    nougat  crispedricewafer  hard  bar  pluribus  sugarpercent  pricepercent  \
0        0                 1     0    1         0         0.732         0.860
1        1                 0     0    1         0         0.604         0.511
2        0                 0     0    0         0         0.011         0.116
3        0                 0     0    0         0         0.011         0.511
4        0                 0     0    0         0         0.906         0.511
..     …               …   …  …       …           …           …
80       0                 0     0    0         0         0.220         0.116
81       0                 0     1    0         0         0.093         0.116
82       0                 0     0    0         1         0.313         0.313
83       0                 0     1    0         0         0.186         0.267
84       0                 1     0    0         1         0.872         0.848

    winpercent
0    66.971725
1    67.602936
2    32.261086
3    46.116505
```

```
4    52.341465
..         …
80   45.466282
81   39.011898
82   44.375519
83   41.904308
84   49.524113

[85 rows x 13 columns]
```

[3]:
```
df.shape
```

[3]:
```
(85, 13)
```

[4]:
```
df.isnull().sum()
```

[4]:
```
competitorname      0
chocolate           0
fruity              0
caramel             0
peanutyalmondy      0
nougat              0
crispedricewafer    0
hard                0
bar                 0
pluribus            0
sugarpercent        0
pricepercent        0
winpercent          0
dtype: int64
```

[5]:
```
df['chocolate'].value_counts()
```

[5]:
```
chocolate
0    48
1    37
Name: count, dtype: int64
```

[6]:
```
x = df.drop(columns=["competitorname" ,"chocolate", "fruity", "caramel",␣
 ↪"peanutyalmondy", "nougat", "crispedricewafer", "hard","bar", "pluribus"],␣
 ↪axis=1)
x
```

[6]:
```
    sugarpercent  pricepercent  winpercent
0          0.732         0.860   66.971725
1          0.604         0.511   67.602936
2          0.011         0.116   32.261086
3          0.011         0.511   46.116505
```

```
4          0.906         0.511   52.341465
..           …             …         …
80         0.220         0.116   45.466282
81         0.093         0.116   39.011898
82         0.313         0.313   44.375519
83         0.186         0.267   41.904308
84         0.872         0.848   49.524113

[85 rows x 3 columns]
```

[7]:
```python
y = df["chocolate"]
y
```

[7]:
```
0     1
1     1
2     0
3     0
4     0
     ..
80    0
81    0
82    0
83    0
84    1
Name: chocolate, Length: 85, dtype: int64
```

[8]:
```python
from sklearn.model_selection import train_test_split
```

[9]:
```python
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
 ↪random_state=42, stratify=y)
```

[10]:
```python
from sklearn.linear_model import LogisticRegression
```

[11]:
```python
model = LogisticRegression(max_iter=1000).fit(x_train, y_train)
model
```

[11]:
```
LogisticRegression(max_iter=1000)
```

[12]:
```python
y_pred = model.predict(x_test)
y_pred
```

[12]:
```
array([0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1], dtype=int64)
```

[13]:
```python
from sklearn.metrics import accuracy_score
```

[14]:
```python
accuracy = accuracy_score(y_test, y_pred)
```

[15]:
```python
print(f"Accuracy = ", accuracy)
```

```
Accuracy =  0.8823529411764706
```

MODEL OPTIMIZATION

[16]:
```
from sklearn.model_selection import GridSearchCV
```

[17]:
```
model = LogisticRegression()
model
```

[17]:
```
LogisticRegression()
```

[18]:
```
param_grid = {
    'penalty' :['l2', None],
    'solver':['liblinear','newton-cg', 'newton-cholesky', 'sag', 'saga'],
    'C':[1.0, 1.5]
}
```

[19]:
```
grid_search = GridSearchCV(model, param_grid, cv=5, n_jobs=-1)
grid_search.fit(x_train, y_train)
```

[19]:
```
GridSearchCV(cv=5, estimator=LogisticRegression(), n_jobs=-1,
             param_grid={'C': [1.0, 1.5], 'penalty': ['l2', None],
                         'solver': ['liblinear', 'newton-cg', 'newton-cholesky',
                                    'sag', 'saga']})
```

[20]:
```
best_params = grid_search.best_params_
print("Best Parameters :", best_params)
```

```
Best Parameters : {'C': 1.0, 'penalty': 'l2', 'solver': 'liblinear'}
```

[21]:
```
best_model = LogisticRegression(**best_params)
best_model.fit(x_train, y_train)
best_model
```

[21]:
```
LogisticRegression(solver='liblinear')
```

[22]:
```
y_pred = best_model.predict(x_test)
y_pred
```

[22]:
```
array([0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1], dtype=int64)
```

[23]:
```
accuracy = accuracy_score(y_test, y_pred)
```

[24]:
```
print("Best Parameters :", best_params)
print(f"Accuracy = ", accuracy)
```

```
Best Parameters : {'C': 1.0, 'penalty': 'l2', 'solver': 'liblinear'}
Accuracy =  0.8823529411764706
```

[ ]:
```

```