car-evaluation-model

February 1, 2024

Day99-100 Car Evaluation Model By: Loga Aswin

[17]:
```
#import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[18]:
```
#load datasets
df = pd.read_csv('/content/car_evaluation.csv')
```

Exploratory Data Analysis(EDA):

[19]:
```
df.head()
```

[19]:

	vhigh	vhigh.1	2	2.1	small	low	unacc
0	vhigh	vhigh	2	2	small	med	unacc
1	vhigh	vhigh	2	2	small	high	unacc
2	vhigh	vhigh	2	2	med	low	unacc
3	vhigh	vhigh	2	2	med	med	unacc
4	vhigh	vhigh	2	2	med	high	unacc

[20]:
```
df.shape
```

[20]:
```
(1727, 7)
```

Rename column name as it labelled 0,1,2...

[21]:
```
col_names = ['paint', 'break', 'alloy', 'wheel', 'headlight', 'gear', 'engine']
df.columns = col_names
col_names
```

[21]:
```
['paint', 'break', 'alloy', 'wheel', 'headlight', 'gear', 'engine']
```

[22]:
```
df.head()
```

[22]:

	paint	break	alloy	wheel	headlight	gear	engine
0	vhigh	vhigh	2	2	small	med	unacc
1	vhigh	vhigh	2	2	small	high	unacc

```
2  vhigh  vhigh     2     2       med   low  unacc
3  vhigh  vhigh     2     2       med   med  unacc
4  vhigh  vhigh     2     2       med  high  unacc
```

[23]:
```
# Summary
df.info
```

[23]:
```
<bound method DataFrame.info of       paint  break  alloy wheel headlight  gear
engine
0     vhigh  vhigh      2     2     small   med  unacc
1     vhigh  vhigh      2     2     small  high  unacc
2     vhigh  vhigh      2     2       med   low  unacc
3     vhigh  vhigh      2     2       med   med  unacc
4     vhigh  vhigh      2     2       med  high  unacc
…       …      …      …     …         …   …     …
1722    low    low  5more  more       med   med   good
1723    low    low  5more  more       med  high  vgood
1724    low    low  5more  more       big   low  unacc
1725    low    low  5more  more       big   med   good
1726    low    low  5more  more       big  high  vgood

[1727 rows x 7 columns]>
```

[24]:
```
col_names = ['paint', 'break', 'alloy', 'wheel', 'headlight', 'gear', 'engine']

for col in col_names:
  print(df[col].value_counts())
```

```
high     432
med      432
low      432
vhigh    431
Name: paint, dtype: int64
high     432
med      432
low      432
vhigh    431
Name: break, dtype: int64
3        432
4        432
5more    432
2        431
Name: alloy, dtype: int64
4       576
more    576
2       575
Name: wheel, dtype: int64
```

```
med      576
big      576
small    575
Name: headlight, dtype: int64
med     576
high    576
low     575
Name: gear, dtype: int64
unacc    1209
acc       384
good       69
vgood      65
Name: engine, dtype: int64
```

[25]:
```
df['engine'].value_counts()
```

[25]:
```
unacc    1209
acc       384
good       69
vgood      65
Name: engine, dtype: int64
```

[26]:
```
# checking missing values
df.isnull().sum()
```

[26]:
```
paint        0
break        0
alloy        0
wheel        0
headlight    0
gear         0
engine       0
dtype: int64
```

[27]:
```
# target variable
X = df.drop(['engine'],axis=1)
y = df['engine']
```

[28]:
```
# Split into train and test data
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,␣
 ↪random_state=42)
```

[29]:
```
X_train.shape, X_test.shape
```

[29]:
```
((1381, 6), (346, 6))
```

Feature Engineering:

[30]:
```
!pip install category_encoders
```

```
Requirement already satisfied: category_encoders in
/usr/local/lib/python3.10/dist-packages (2.6.3)
Requirement already satisfied: numpy>=1.14.0 in /usr/local/lib/python3.10/dist-
packages (from category_encoders) (1.23.5)
Requirement already satisfied: scikit-learn>=0.20.0 in
/usr/local/lib/python3.10/dist-packages (from category_encoders) (1.2.2)
Requirement already satisfied: scipy>=1.0.0 in /usr/local/lib/python3.10/dist-
packages (from category_encoders) (1.11.4)
Requirement already satisfied: statsmodels>=0.9.0 in
/usr/local/lib/python3.10/dist-packages (from category_encoders) (0.14.1)
Requirement already satisfied: pandas>=1.0.5 in /usr/local/lib/python3.10/dist-
packages (from category_encoders) (1.5.3)
Requirement already satisfied: patsy>=0.5.1 in /usr/local/lib/python3.10/dist-
packages (from category_encoders) (0.5.6)
Requirement already satisfied: python-dateutil>=2.8.1 in
/usr/local/lib/python3.10/dist-packages (from pandas>=1.0.5->category_encoders)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas>=1.0.5->category_encoders) (2023.3.post1)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages
(from patsy>=0.5.1->category_encoders) (1.16.0)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn>=0.20.0->category_encoders) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-
learn>=0.20.0->category_encoders) (3.2.0)
Requirement already satisfied: packaging>=21.3 in
/usr/local/lib/python3.10/dist-packages (from
statsmodels>=0.9.0->category_encoders) (23.2)
```

[31]:
```
import category_encoders as ce

encoder = ce.OrdinalEncoder(cols=['paint', 'break', 'alloy', 'wheel',␣
 ↪'headlight', 'gear'])

X_train = encoder.fit_transform(X_train)
X_test = encoder.transform(X_test)
```

[32]:
```
X_train.head()
```

[32]:

	paint	break	alloy	wheel	headlight	gear
107	1	1	1	1	1	1
900	2	2	2	2	1	2
1708	3	3	3	1	2	3

```
705       4      4      4      1          3     2
678       4      4      2      1          3     2
```

RandomForest Classifier Model:

[33]:
```
from sklearn.ensemble import RandomForestClassifier

# classifier
model = RandomForestClassifier(random_state=0)

model.fit(X_train, y_train)

# Predict test results

y_pred = model.predict(X_test)
```

Model Evaluation Metrics:

[34]:
```
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix

accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
```

```
0.9479768786127167
```

[35]:
```
matrix = confusion_matrix(y_test, y_pred)
print(matrix)
```

```
[[ 73   3   0   1]
 [  3   8   0   4]
 [  3   0 234   0]
 [  4   0   0  13]]
```

[36]:
```
report = classification_report(y_test, y_pred)
print(report)
```

```
              precision    recall  f1-score   support

         acc       0.88      0.95      0.91        77
        good       0.73      0.53      0.62        15
       unacc       1.00      0.99      0.99       237
       vgood       0.72      0.76      0.74        17

    accuracy                           0.95       346
   macro avg       0.83      0.81      0.82       346
weighted avg       0.95      0.95      0.95       346
```

[37]:
```
#Confusion Matrix
import seaborn as sns

plt.figure(figsize=(8, 6))
sns.heatmap(matrix, annot=True, fmt="d", cmap="Blues", xticklabels=model.
 ↪classes_, yticklabels=model.classes_)
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.title('Confusion Matrix')
plt.show()
```