4xa2erveu

August 2, 2023

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

[2]:
```
from google.colab import drive
drive.mount('/content/drive')
```

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

[3]:
```
df=pd.read_csv("/content/drive/MyDrive/mydatasets/c7_used_cars.csv")
df
```

[3]:
```
       Unnamed: 0   model  year  price transmission  mileage fuelType  tax  \
0               0   T-Roc  2019  25000    Automatic    13904   Diesel  145
1               1   T-Roc  2019  26883    Automatic     4562   Diesel  145
2               2   T-Roc  2019  20000       Manual     7414   Diesel  145
3               3   T-Roc  2019  33492    Automatic     4825   Petrol  145
4               4   T-Roc  2019  22900    Semi-Auto     6500   Petrol  150
…             …     …     …      …            …        …      …    …
99182       10663      A3  2020  16999       Manual     4018   Petrol  145
99183       10664      A3  2020  16999       Manual     1978   Petrol  150
99184       10665      A3  2020  17199       Manual      609   Petrol  150
99185       10666      Q3  2017  19499    Automatic     8646   Petrol  150
99186       10667      Q3  2016  15999       Manual    11855   Petrol  150

        mpg  engineSize  Make
0      49.6         2.0    VW
1      49.6         2.0    VW
2      50.4         2.0    VW
3      32.5         2.0    VW
4      39.8         1.5    VW
…       …           …     …
99182  49.6         1.0  Audi
99183  49.6         1.0  Audi
99184  49.6         1.0  Audi
```

```
99185  47.9         1.4  Audi
99186  47.9         1.4  Audi

[99187 rows x 11 columns]
```

[4]:
```
df.head()
```

[4]:
```
   Unnamed: 0   model  year  price transmission  mileage fuelType  tax   mpg  \
0           0   T-Roc  2019  25000    Automatic    13904   Diesel  145  49.6
1           1   T-Roc  2019  26883    Automatic     4562   Diesel  145  49.6
2           2   T-Roc  2019  20000       Manual     7414   Diesel  145  50.4
3           3   T-Roc  2019  33492    Automatic     4825   Petrol  145  32.5
4           4   T-Roc  2019  22900    Semi-Auto     6500   Petrol  150  39.8

   engineSize Make
0         2.0   VW
1         2.0   VW
2         2.0   VW
3         2.0   VW
4         1.5   VW
```

1 Data Cleaning and Data Preprocessing

[5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99187 entries, 0 to 99186
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Unnamed: 0    99187 non-null  int64
 1   model         99187 non-null  object
 2   year          99187 non-null  int64
 3   price         99187 non-null  int64
 4   transmission  99187 non-null  object
 5   mileage       99187 non-null  int64
 6   fuelType      99187 non-null  object
 7   tax           99187 non-null  int64
 8   mpg           99187 non-null  float64
 9   engineSize    99187 non-null  float64
 10  Make          99187 non-null  object
dtypes: float64(2), int64(5), object(4)
memory usage: 8.3+ MB
```

[6]:
```
df.describe()
```

```
[6]:           Unnamed: 0          year          price        mileage           tax  \
     count  99187.000000  99187.000000   99187.000000   99187.000000  99187.000000
     mean    6294.413532   2017.087723   16805.347656   23058.914213    120.299838
     std     4265.588536      2.123934    9866.773417   21148.523721     63.150926
     min        0.000000   1970.000000     450.000000       1.000000      0.000000
     25%     2755.000000   2016.000000    9999.000000    7425.000000    125.000000
     50%     5591.000000   2017.000000   14495.000000   17460.000000    145.000000
     75%     9420.000000   2019.000000   20870.000000   32339.000000    145.000000
     max    17964.000000   2060.000000  159999.000000  323000.000000    580.000000

                     mpg    engineSize
     count  99187.000000  99187.000000
     mean      55.166825      1.663280
     std       16.138522      0.557646
     min        0.300000      0.000000
     25%       47.100000      1.200000
     50%       54.300000      1.600000
     75%       62.800000      2.000000
     max      470.800000      6.600000
```

```
[7]: df.columns
```

```
[7]: Index(['Unnamed: 0', 'model', 'year', 'price', 'transmission', 'mileage',
            'fuelType', 'tax', 'mpg', 'engineSize', 'Make'],
           dtype='object')
```

```
[8]: feature_matrix = df[['Unnamed: 0', 'year', 'price', 'mileage',
             'tax', 'mpg', 'engineSize']]
     target_vector = df[['Make']]
```

```
[9]: fs = StandardScaler().fit_transform(feature_matrix)
     logr = LogisticRegression()
     logr.fit(fs,target_vector)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/utils/validation.py:1143:
DataConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples, ), for example using
ravel().
  y = column_or_1d(y, warn=True)
```

```
[9]: LogisticRegression()
```

```
[10]: observation=[[1,2,3,4,5,6,7]]
      prediction = logr.predict(observation)
      print(prediction)
```

```
['BMW']
```

```
[11]: logr.classes_
```

```
[11]: array(['Audi', 'BMW', 'VW', 'ford', 'hyundi', 'merc', 'skoda', 'toyota',
             'vauxhall'], dtype=object)
```

```
[12]: logr.predict_proba(observation)
```

```
[12]: array([[2.74122931e-05, 9.36836737e-01, 2.51395992e-08, 5.85008303e-09,
              3.09237182e-12, 6.31357545e-02, 6.44018883e-09, 5.85474765e-08,
              7.49581427e-16]])
```

Random Forest

```
[21]: df['Make'].value_counts()
```

```
[21]: 4    17965
      3    15157
      9    13632
      6    13119
      2    10781
      1    10668
      8     6738
      7     6267
      5     4860
      Name: Make, dtype: int64
```

```
[22]: x=df[['Unnamed: 0','year', 'price',  'mileage',
              'tax', 'mpg', 'engineSize']]
      y=df['Make']
```

```
[23]: g1={ 'Make':{'Audi':1, 'BMW':2, 'VW':3, 'ford':4, 'hyundi':5, 'merc':6, 'skoda':
        ↪7, 'toyota':8,
              'vauxhall':9}}
      df=df.replace(g1)
      df
```

```
[23]:        Unnamed: 0   model  year  price transmission  mileage fuelType  tax  \
      0               0   T-Roc  2019  25000    Automatic    13904   Diesel  145
      1               1   T-Roc  2019  26883    Automatic     4562   Diesel  145
      2               2   T-Roc  2019  20000       Manual     7414   Diesel  145
      3               3   T-Roc  2019  33492    Automatic     4825   Petrol  145
      4               4   T-Roc  2019  22900    Semi-Auto     6500   Petrol  150
      …             …    …   …    …          …        …    …   …
      99182       10663      A3  2020  16999       Manual     4018   Petrol  145
      99183       10664      A3  2020  16999       Manual     1978   Petrol  150
      99184       10665      A3  2020  17199       Manual      609   Petrol  150
      99185       10666      Q3  2017  19499    Automatic     8646   Petrol  150
```

```
99186       10667      Q3  2016  15999       Manual     11855    Petrol  150

        mpg  engineSize  Make
0      49.6         2.0     3
1      49.6         2.0     3
2      50.4         2.0     3
3      32.5         2.0     3
4      39.8         1.5     3
…      …          …    …
99182  49.6         1.0     1
99183  49.6         1.0     1
99184  49.6         1.0     1
99185  47.9         1.4     1
99186  47.9         1.4     1

[99187 rows x 11 columns]
```

[24]:
```
from sklearn.model_selection import train_test_split
```

[25]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.70)
```

[26]:
```
from sklearn.ensemble import RandomForestClassifier
```

[27]:
```
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

[27]:
```
RandomForestClassifier()
```

[28]:
```
parameters={'max_depth':[1,2,3,4,5],
            'min_samples_leaf':[5,10,15,20,25],
            'n_estimators':[10,20,30,40,50]
}
```

[29]:
```
from sklearn.model_selection import GridSearchCV
grid_search␣
 ↪=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
grid_search.fit(x_train,y_train)
```

[29]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5],
                         'min_samples_leaf': [5, 10, 15, 20, 25],
                         'n_estimators': [10, 20, 30, 40, 50]},
             scoring='accuracy')
```

[30]:
```
grid_search.best_score_
```

[30]:
```
0.5165346392049547
```

[31]:
```
rfc_best=grid_search.best_estimator_
```

[32]:
```
from sklearn.tree import plot_tree

plt.figure(figsize=(80,40))
plot_tree(rfc_best.estimators_[5],feature_names=x.
 ↪columns,class_names=['a','b','c','d','e','f','g','h','i'],filled=True)
```

[32]:
```
[Text(0.5, 0.9166666666666666, 'tax <= 15.0\ngini = 0.873\nsamples =
43992\nvalue = [7438, 7514, 10654, 12525, 3409, 9225, 4287, 4665, 9713]\nclass =
d'),
 Text(0.25, 0.75, 'mileage <= 35384.0\ngini = 0.778\nsamples = 2779\nvalue =
[375, 220, 404, 1531, 113, 122, 163, 1228, 262]\nclass = d'),
 Text(0.125, 0.5833333333333334, 'engineSize <= 1.55\ngini = 0.714\nsamples =
1573\nvalue = [160, 92, 182, 979, 50, 42, 65, 865, 65]\nclass = d'),
 Text(0.0625, 0.4166666666666667, 'price <= 6999.5\ngini = 0.623\nsamples =
1175\nvalue = [73, 65, 68, 971, 7, 18, 45, 589, 25]\nclass = d'),
 Text(0.03125, 0.25, 'Unnamed: 0 <= 5605.0\ngini = 0.426\nsamples = 196\nvalue =
[0, 0, 1, 41, 0, 0, 31, 226, 7]\nclass = h'),
 Text(0.015625, 0.08333333333333333, 'gini = 0.31\nsamples = 174\nvalue = [0, 0,
0, 13, 0, 0, 30, 226, 6]\nclass = h'),
 Text(0.046875, 0.08333333333333333, 'gini = 0.181\nsamples = 22\nvalue = [0, 0,
1, 28, 0, 0, 1, 0, 1]\nclass = d'),
 Text(0.09375, 0.25, 'mileage <= 21901.5\ngini = 0.582\nsamples = 979\nvalue =
[73, 65, 67, 930, 7, 18, 14, 363, 18]\nclass = d'),
 Text(0.078125, 0.08333333333333333, 'gini = 0.577\nsamples = 505\nvalue = [24,
24, 27, 462, 1, 10, 10, 225, 10]\nclass = d'),
 Text(0.109375, 0.08333333333333333, 'gini = 0.58\nsamples = 474\nvalue = [49,
41, 40, 468, 6, 8, 4, 138, 8]\nclass = d'),
 Text(0.1875, 0.4166666666666667, 'engineSize <= 1.7\ngini = 0.75\nsamples =
398\nvalue = [87, 27, 114, 8, 43, 24, 20, 276, 40]\nclass = h'),
 Text(0.15625, 0.25, 'Unnamed: 0 <= 6859.0\ngini = 0.749\nsamples = 178\nvalue =
[65, 3, 114, 6, 43, 0, 20, 0, 38]\nclass = c'),
 Text(0.140625, 0.08333333333333333, 'gini = 0.736\nsamples = 140\nvalue = [29,
0, 97, 4, 43, 0, 20, 0, 35]\nclass = c'),
 Text(0.171875, 0.08333333333333333, 'gini = 0.568\nsamples = 38\nvalue = [36,
3, 17, 2, 0, 0, 0, 0, 3]\nclass = a'),
 Text(0.21875, 0.25, 'Unnamed: 0 <= 6720.0\ngini = 0.365\nsamples = 220\nvalue =
[22, 24, 0, 2, 0, 24, 0, 276, 2]\nclass = h'),
 Text(0.203125, 0.08333333333333333, 'gini = 0.231\nsamples = 198\nvalue = [18,
11, 0, 0, 0, 11, 0, 276, 0]\nclass = h'),
 Text(0.234375, 0.08333333333333333, 'gini = 0.687\nsamples = 22\nvalue = [4,
13, 0, 2, 0, 13, 0, 0, 2]\nclass = b'),
 Text(0.375, 0.5833333333333334, 'year <= 2014.5\ngini = 0.835\nsamples =
1206\nvalue = [215, 128, 222, 552, 63, 80, 98, 363, 197]\nclass = d'),
 Text(0.3125, 0.4166666666666667, 'mpg <= 66.5\ngini = 0.761\nsamples =
281\nvalue = [45, 11, 41, 147, 17, 7, 18, 140, 12]\nclass = d'),
 Text(0.28125, 0.25, 'Unnamed: 0 <= 6324.5\ngini = 0.307\nsamples = 80\nvalue =
[0, 0, 1, 104, 0, 0, 0, 23, 0]\nclass = d'),
 Text(0.265625, 0.08333333333333333, 'gini = 0.467\nsamples = 37\nvalue = [0, 0,
0, 39, 0, 0, 0, 23, 0]\nclass = d'),
 Text(0.296875, 0.08333333333333333, 'gini = 0.03\nsamples = 43\nvalue = [0, 0,
1, 65, 0, 0, 0, 0, 0]\nclass = d'),
 Text(0.34375, 0.25, 'year <= 2012.5\ngini = 0.791\nsamples = 201\nvalue = [45,
11, 40, 43, 17, 7, 18, 117, 12]\nclass = h'),
 Text(0.328125, 0.08333333333333333, 'gini = 0.531\nsamples = 20\nvalue = [9, 0,
0, 3, 0, 0, 0, 19, 0]\nclass = h'),
 Text(0.359375, 0.08333333333333333, 'gini = 0.807\nsamples = 181\nvalue = [36,
11, 40, 40, 17, 7, 18, 98, 12]\nclass = h'),
 Text(0.4375, 0.4166666666666667, 'mileage <= 56869.5\ngini = 0.846\nsamples =
925\nvalue = [170, 117, 181, 405, 46, 73, 80, 223, 185]\nclass = d'),
 Text(0.40625, 0.25, 'mpg <= 65.85\ngini = 0.836\nsamples = 639\nvalue = [118,
63, 140, 289, 34, 51, 48, 203, 95]\nclass = d'),
 Text(0.390625, 0.08333333333333333, 'gini = 0.234\nsamples = 74\nvalue = [0, 0,
10, 108, 0, 0, 0, 2, 4]\nclass = d'),
 Text(0.421875, 0.08333333333333333, 'gini = 0.855\nsamples = 565\nvalue = [118,
63, 130, 181, 34, 51, 48, 201, 91]\nclass = h'),
 Text(0.46875, 0.25, 'mileage <= 71256.5\ngini = 0.84\nsamples = 286\nvalue =
[52, 54, 41, 116, 12, 22, 32, 20, 90]\nclass = d'),
 Text(0.453125, 0.08333333333333333, 'gini = 0.802\nsamples = 152\nvalue = [22,
25, 22, 85, 9, 3, 16, 13, 39]\nclass = d'),
 Text(0.484375, 0.08333333333333333, 'gini = 0.849\nsamples = 134\nvalue = [30,
29, 19, 31, 3, 19, 16, 7, 51]\nclass = i'),
 Text(0.75, 0.75, 'mpg <= 56.0\ngini = 0.872\nsamples = 41213\nvalue = [7063,
7294, 10250, 10994, 3296, 9103, 4124, 3437, 9451]\nclass = d'),
 Text(0.625, 0.5833333333333334, 'Unnamed: 0 <= 10754.5\ngini = 0.862\nsamples =
24039\nvalue = [5171, 4447, 6216, 5034, 1715, 4534, 1865, 1224, 7681]\nclass =
i'),
 Text(0.5625, 0.4166666666666667, 'tax <= 130.0\ngini = 0.866\nsamples =
19785\nvalue = [5171, 4439, 3986, 2788, 1715, 3826, 1865, 1224, 6083]\nclass =
i'),
 Text(0.53125, 0.25, 'price <= 10947.5\ngini = 0.809\nsamples = 2345\nvalue =
[313, 172, 457, 589, 115, 189, 270, 273, 1316]\nclass = i'),
 Text(0.515625, 0.08333333333333333, 'gini = 0.67\nsamples = 1533\nvalue = [55,
34, 129, 521, 115, 3, 108, 200, 1254]\nclass = i'),
 Text(0.546875, 0.08333333333333333, 'gini = 0.835\nsamples = 812\nvalue = [258,
138, 328, 68, 0, 186, 162, 73, 62]\nclass = c'),
 Text(0.59375, 0.25, 'Unnamed: 0 <= 6806.5\ngini = 0.865\nsamples = 17440\nvalue
= [4858, 4267, 3529, 2199, 1600, 3637, 1595, 951, 4767]\nclass = a'),
 Text(0.578125, 0.08333333333333333, 'gini = 0.875\nsamples = 12101\nvalue =
[3282, 3099, 2579, 1333, 1600, 2268, 1595, 951, 2367]\nclass = a'),
 Text(0.609375, 0.08333333333333333, 'gini = 0.811\nsamples = 5339\nvalue =
[1576, 1168, 950, 866, 0, 1369, 0, 0, 2400]\nclass = i'),
 Text(0.6875, 0.4166666666666667, 'engineSize <= 1.95\ngini = 0.716\nsamples =
4254\nvalue = [0, 8, 2230, 2246, 0, 708, 0, 0, 1598]\nclass = d'),
 Text(0.65625, 0.25, 'Unnamed: 0 <= 13632.0\ngini = 0.672\nsamples = 2414\nvalue
= [0, 1, 706, 1597, 0, 219, 0, 0, 1328]\nclass = d'),
 Text(0.640625, 0.08333333333333333, 'gini = 0.647\nsamples = 1634\nvalue = [0,
1, 413, 643, 0, 219, 0, 0, 1328]\nclass = i'),
 Text(0.671875, 0.08333333333333333, 'gini = 0.36\nsamples = 780\nvalue = [0, 0,
293, 954, 0, 0, 0, 0, 0]\nclass = d'),
 Text(0.71875, 0.25, 'price <= 21336.5\ngini = 0.646\nsamples = 1840\nvalue =
[0, 7, 1524, 649, 0, 489, 0, 0, 270]\nclass = c'),
 Text(0.703125, 0.08333333333333333, 'gini = 0.721\nsamples = 934\nvalue = [0,
7, 496, 495, 0, 209, 0, 0, 268]\nclass = c'),
 Text(0.734375, 0.08333333333333333, 'gini = 0.459\nsamples = 906\nvalue = [0,
0, 1028, 154, 0, 280, 0, 0, 2]\nclass = c'),
 Text(0.875, 0.5833333333333334, 'tax <= 25.0\ngini = 0.864\nsamples =
17174\nvalue = [1892, 2847, 4034, 5960, 1581, 4569, 2259, 2213, 1770]\nclass =
d'),
 Text(0.8125, 0.4166666666666667, 'Unnamed: 0 <= 13215.5\ngini = 0.818\nsamples
= 3637\nvalue = [418, 327, 1904, 827, 192, 895, 548, 100, 557]\nclass = c'),
 Text(0.78125, 0.25, 'mileage <= 23780.0\ngini = 0.813\nsamples = 3448\nvalue =
[418, 327, 1875, 546, 192, 895, 548, 100, 554]\nclass = c'),
 Text(0.765625, 0.08333333333333333, 'gini = 0.784\nsamples = 1079\nvalue = [58,
75, 681, 241, 72, 183, 202, 34, 182]\nclass = c'),
 Text(0.796875, 0.08333333333333333, 'gini = 0.82\nsamples = 2369\nvalue = [360,
252, 1194, 305, 120, 712, 346, 66, 372]\nclass = c'),
 Text(0.84375, 0.25, 'mileage <= 17027.0\ngini = 0.185\nsamples = 189\nvalue =
[0, 0, 29, 281, 0, 0, 0, 0, 3]\nclass = d'),
 Text(0.828125, 0.08333333333333333, 'gini = 0.368\nsamples = 21\nvalue = [0, 0,
9, 28, 0, 0, 0, 0, 0]\nclass = d'),
 Text(0.859375, 0.08333333333333333, 'gini = 0.154\nsamples = 168\nvalue = [0,
0, 20, 253, 0, 0, 0, 0, 3]\nclass = d'),
 Text(0.9375, 0.4166666666666667, 'price <= 18033.5\ngini = 0.86\nsamples =
13537\nvalue = [1474, 2520, 2130, 5133, 1389, 3674, 1711, 2113, 1213]\nclass =
d'),
 Text(0.90625, 0.25, 'price <= 9998.5\ngini = 0.845\nsamples = 10165\nvalue =
[1025, 1482, 1784, 4843, 1320, 1001, 1579, 1772, 1176]\nclass = d'),
 Text(0.890625, 0.08333333333333333, 'gini = 0.831\nsamples = 3019\nvalue = [79,
189, 419, 1155, 650, 52, 542, 1095, 543]\nclass = d'),
 Text(0.921875, 0.08333333333333333, 'gini = 0.832\nsamples = 7146\nvalue =
[946, 1293, 1365, 3688, 670, 949, 1037, 677, 633]\nclass = d'),
 Text(0.96875, 0.25, 'engineSize <= 2.05\ngini = 0.697\nsamples = 3372\nvalue =
[449, 1038, 346, 290, 69, 2673, 132, 341, 37]\nclass = f'),
 Text(0.953125, 0.08333333333333333, 'gini = 0.776\nsamples = 2590\nvalue =
[418, 968, 346, 286, 69, 1532, 132, 330, 37]\nclass = f'),
 Text(0.984375, 0.08333333333333333, 'gini = 0.172\nsamples = 782\nvalue = [31,
70, 0, 4, 0, 1141, 0, 11, 0]\nclass = f')]
```