Tugas 1 Mata Kuliah Business Intelligence

Dosen Pengampu Bapak Dr. Hilman F. Pardede, MEICT

Nama : Ahmad Maulid Ridwan
Kelas : 14.3B.01
NIM : 14210252

FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS NUSA MANDIRI
JAKARTA

2023

Car evaluation classification using machine learning

1. Abstrak

Mengevaluasi kondisi mobil sebelum membeli memainkan peran penting dalam pengambilan keputusan. Secara manual, mengklasifikasikan mobil berkondisi baik atau dapat diterima dari mobil berkondisi tidak dapat diterima cukup memakan waktu. Kami dapat memanfaatkan teknik Pembelajaran Mesin untuk mengembangkan sistem otomatis untuk evaluasi mobil karena machine learning telah menunjukkan hasil yang menjanjikan dalam menyelesaikan masalah terkait klasifikasi. Dalam tulisan ini, kami akan menganalisis kualifikasi fisik mobil, membantu/merekomendasikan pengguna dalam proses pengambilan keputusan berdasarkan atribut mobil. Pada penelitian ini model menghasilkan skor presisi, recall, f1 masing-masing adalah 96%, 89%, dan 92%, di mana jumlah sampel mobil berkondisi yang dapat diterima (acc) sebanyak 83. Di sisi lain, skor presisi, recall, f1 untuk kelas yang tidak dapat diterima masing-masing adalah 99%, 100%, dan 99%, di mana jumlah sampel mobil yang tidak dapat diterima (unacc) sebanyak 235. Dari hasil ini, kita dapat melihat bahwa performa model lebih baik untuk sampel mobil yang tidak dapat diterima daripada mobil yang dapat diterima. Akurasi keseluruhan adalah 96% pada model ini.

Keyword : car evaluation, machine learning, classification, random forest

2. Introduction

Di bidang pemrosesan data dan pemrosesan informasi, penggunaan teknologi pembelajaran mesin semakin populer. Kategorisasi mobil menggunakan dataset car evaluation adalah salah satu contoh bagaimana teknologi ini digunakan. Untuk membuat model klasifikasi yang dapat memprediksi kelas penilaian mobil berdasarkan beberapa kriteria termasuk buying, maint, doors, persons, lug_boot dan safety, dataset penilaian mobil akan digunakan dalam tulisan ini.

Informasi tentang 1.728 mobil termasuk dalam kumpulan data Car Evaluation. Mobil-mobil ini akan dinilai berdasarkan enam kriteria: buying, maint, doors, persons, lug_boot dan safety. Ada empat kategori penilaian: unacc (unacceptable), acc (acceptable), baik (good), dan vgood (very good). Kumpulan data ini dapat dianalisis dan diperiksa menggunakan teknologi pembelajaran mesin untuk membuat model klasifikasi yang akurat untuk memprediksi kelas penilaian mobil berdasarkan karakteristik yang disebutkan di atas.

Model klasifikasi terbaik akan dibuat dalam tulisan ini menggunakan pendekatan mesin learning, yaitu Random Forest. Hasil dari penelitian ini diharapkan dapat membantu membuat proses evaluasi mobil menjadi lebih efektif dan efisien sekaligus memberikan informasi yang lebih akurat kepada pengguna untuk membantu mereka memilih mobil yang sesuai dengan kebutuhan mereka.

3. Methode

Pendekatan random forest diterapkan dalam penelitian ini. Metode pembelajaran mesin yang disebut random forest digunakan untuk menentukan kelas objek berdasarkan karakteristiknya. Metode ini menggabungkan beberapa pohon keputusan untuk mendapatkan prediksi yang lebih akurat.

Dataset Car Evaluation akan diperiksa dan diproses menggunakan pendekatan random forest dalam penelitian ini. Dataset pertama-tama akan dipecah menjadi set data pelatihan dan pengujian. Beberapa pohon keputusan dibuat dengan menggunakan pendekatan random forest dan masing-masing pohon memilih subhimpunan sifat untuk menarik kesimpulan. Setiap pohon keputusan akan memilih subset atribut secara acak dan hanya mempertimbangkan subset tersebut saat membuat keputusan.

Dengan menggunakan temuan mayoritas dari setiap pohon keputusan setelah pohon keputusan dibuat, prediksi kelas penilaian mobil dapat dibentuk. Pada metode ini dibandingkan dengan metodologi pohon keputusan yang hanya menggunakan satu pohon keputusan, prediksi kelas evaluasi mobil yang dihasilkan oleh model random forest akan lebih akurat.

Model random forest kemudian harus diuji kebenarannya menggunakan data uji setelah dibangun. Untuk menentukan seberapa baik model ini memprediksi kelas penilaian mobil, hasil akurasi model akan dipelajari dan dinilai.

Untuk memprediksi kelas penilaian mobil berdasarkan propertinya dalam kumpulan data penilaian mobil, metode random forest adalah metodologi pembelajaran mesin yang berguna. Metode ini dapat meningkatkan akurasi perkiraan, memberikan informasi yang lebih akurat kepada pengguna untuk membantu mereka memilih mobil yang sesuai dengan kebutuhan mereka.

4. Experimental setup

A. Dataset

Repositori machine learning UCI menyediakan kumpulan data untuk tugas pengklasifikasian evaluasi mobil. Kumpulan data ini, yang dibuat pada tahun 1997 oleh Marko Bohanec dan Blaz Zupan, digunakan untuk menilai seberapa baik kinerja metode klasifikasi saat diterapkan pada penilaian mobil.

Data dari berbagai dealer mobil dikumpulkan sebagai bagian dari proses kompilasi kumpulan data Car Evaluation. Kumpulan data yang diperlukan untuk penelitian ini dibuat setelah data dikumpulkan dan dianalisis. Ada 1728 data dalam koleksi ini, dan ada 6 atribut. Dataset Car Evaluation memiliki enam fitur: buying, maint, doors, persons, lug_boot, dan safety. Atribut buying, maint, doors, persons, lug_boot, dan safety masing-masing mewakili aspek fungsi mobil yang berbeda. buying mewakili harga kendaraan, maint mewakili biaya pemeliharaan, door mewakili jumlah pintu, persons mewakili jumlah orang, lug_boot mewakili bagasi dan safety mewakili tingkat keamanan. Distribusi kelas dataset Car Evaluation seimbang, dengan jumlah kasus yang kira-kira sama di setiap kelas. Dalam kumpulan data ini, 4 kategori peringkat mobil paling sering diamati: unacc (70%), acc (22.2%), good (4%), dan vgood (3.8%).

Dataset Car Evaluation akan digunakan dalam eksperimen ini untuk membangun model klasifikasi dengan menggunakan strategi machine learning yaitu pendekatan random forest. Dataset ini akan dibagi menjadi set pelatihan dan pengujian dengan rasio 80:20. Model random forest akan dilatih menggunakan data pelatihan, dan data uji akan digunakan untuk mengevaluasi seberapa baik model memprediksi masa depan. Paket scikit-learn, pandas dan bahasa pemrograman Python akan digunakan dalam proyek ini untuk membuat teknik random forest.

Dataset Car Evaluation akan diproses terlebih dahulu dengan OrdinalEncoder. Karena Kami menemukan bahwa semua variabel diatur secara kategoris. Meskipun paket scikit-learn yang akan kita gunakan untuk pelajaran ini tidak dapat secara langsung menangani variabel kategorikal, Akibatnya, langkah konversi diperlukan. Namun, pertama-tama kita harus memeriksa jenis variabel kategori yang termasuk dalam kumpulan data. Kita dapat menentukan bahwa variabel adalah variabel kategori ordinal dari bagian pra-lab karena semua kategori untuk setiap variabel terkait dengan peringkat atau urutan. Untuk menjaga urutan variabel, maka kami melakukan pengkodean ordinal pada variabel. Misalnya, ada tiga ukuran untuk bagasi boot: small, medium, dan big yang masing-masing memiliki pengaturan tertentu. Untuk memulai, kami menentukan kategori untuk masing-masing variabel. Untuk masalah ini, kami akan menggunakan paket sklearn bernama OrdinalEncoder dan sekumpulan kategori. Untuk doors variabel, kategorinya adalah ['2', '3', '4', '5more'] menjadi [0, 1, 2, 3] setelah konversi. Misalnya, nilai buying akan dikonversi menjadi [0, 1, 2, 3] dari ['low','med', 'high', 'vhigh']. Ini dilakukan untuk mengubah kualitas nominal menjadi atribut numerik sehingga algoritma machine learning dapat memprosesnya. Dan proses selanjutnya pada penelitian ini menggunakan fungsi train_test_split dari paket scikit-learn, dataset kemudian akan dipecah menjadi data latih dan data uji. Model akan dilatih menggunakan data pelatihan, dan performanya akan dievaluasi menggunakan data pengujian.

Fungsi Random Forest Classifier dari paket scikit-learn akan digunakan untuk melatih model random forest setelah dataset dipartisi. Setelah model dilatih, ukuran akurasi akan digunakan untuk menilai kinerja model. Proporsi perkiraan akurat terhadap prediksi total dikenal sebagai akurasi. Model berkinerja lebih baik pada klasifikasi dengan nilai akurasi yang lebih besar. Ukuran penilaian lainnya, seperti presisi, recal, dan skor F1, akan digunakan selain akurasi untuk menilai kinerja model.

Tujuan utama dari seluruh percobaan adalah untuk membuat model klasifikasi yang dapat secara akurat memprediksi penilaian mobil dan digunakan dalam keadaan evaluasi mobil dunia nyata.

B. Setup hyper parameters

Beberapa hyperparameter harus dimodifikasi dalam pendekatan random forest untuk kategorisasi penilaian mobil agar model dapat beroperasi dengan sebaik-baiknya. Berikut adalah ilustrasi cara mengonfigurasi hyperparameter model random forest untuk penelitian ini:

Jumlah pohon (n_estimator): Untuk kasus ini, 10 dan 100 pohon akan digunakan. Hal ini dilakukan untuk mendapatkan kinerja model terbaik sekaligus menjaga agar durasi pelatihan dapat diatur.

Model random forest diharapkan dapat melakukan yang terbaik ketika mengkategorikan penilaian mobil pada dataset Evaluasi mobil dengan konfigurasi hyperparameter yang tepat. Namun, ketahuilah bahwa konfigurasi hyperparameter terbaik dapat bervariasi berdasarkan kumpulan data. Oleh karena itu, pengaturan hyperparameter dapat dilihat sebagai komponen penting dari eksperimen pembelajaran mesin dan harus dipelajari dengan cermat.

5. Results and Discussions

Kumpulan data Evaluasi mobil digunakan untuk menguji pendekatan random forest dan temuannya menunjukkan bahwa pendekatan tersebut berkinerja cukup baik dalam mengkategorikan peringkat mobil. Berikut ini adalah temuan dari penilaian model yang dilatih:

Presisi: 96%

Akurasi: 96%

Recall : 89%

F1-score : 92%

Menurut temuan ini, model random forest dapat mengklasifikasikan evaluasi mobil dengan akurasi yang cukup tinggi. Nilai akurasi yang tinggi menunjukkan bahwa model dapat melakukan prediksi dengan benar pada sebagian besar data, sedangkan nilai presisi dan recall menunjukkan bahwa model dapat menghindari terjadinya false positive dan false negative pada kasus klasifikasi.

Dalam penelitian ini, pendekatan random forest menghasilkan kinerja klasifikasi yang baik untuk dataset Car Evaluation. Ini menunjukkan bagaimana pendekatan machine learning yang menggabungkan banyak pohon keputusan dapat meningkatkan kinerja model secara keseluruhan. Temuan ini juga menunjukkan bagaimana kategorisasi penilaian mobil dapat sangat dipengaruhi oleh fitur dalam kumpulan data Car Evaluation. Saat mengkategorikan, kualitas seperti safety, persons dan buying menunjukkan relevansi yang tinggi, namun atribut seperti maint, lug_boot dan doors tidak.

Pengaturan hyperparameter yang tepat juga dapat sangat meningkatkan kinerja model dalam eksperimen ini. Performa model mungkin sangat dipengaruhi oleh pengaturan hyperparameter untuk pohon (n_estimator). Oleh karena itu, dalam setiap percobaan pembelajaran mesin, pilihan hyperparameter harus diperhitungkan dengan baik.

Secara keseluruhan, temuan percobaan menunjukkan bahwa metode random forest dapat secara efektif melakukan klasifikasi evaluasi mobil pada dataset Car Evaluation. Namun, perlu diingat bahwa hasil eksperimen ini hanya berlaku untuk set data dan keadaan tertentu, dan set data dari kasus lain dapat memberikan hasil yang berbeda. Oleh karena itu, penelitian lebih lanjut dapat dilakukan untuk mengkonfirmasi temuan ini di berbagai kumpulan data dan kasus yang berbeda.

6. Conclusions

Pada hasil analisa di atas, skor presisi, recall, f1 masing-masing adalah 96%, 89%, dan 92%, di mana jumlah sampel mobil berkondisi yang dapat diterima (acc) adalah 83. Di sisi lain, skor presisi, recall, f1 untuk kelas yang tidak dapat diterima masing-masing adalah 99%, 100%, dan

99%, di mana jumlah sampel mobil yang tidak dapat diterima (unacc) adalah 235. Dari hasil ini, kita dapat melihat bahwa performa model lebih baik untuk sampel mobil yang tidak dapat diterima (unacc) daripada mobil yang dapat diterima (acc). Akurasi keseluruhan adalah 96%.

Menurut hasil, keselamatan (safety) adalah atribut utama bagi pembeli mobil. Jika pelanggan menganggap mobil tidak aman, dia tidak akan membelinya. Lalu soal kapasitas orang yang bisa diangkut (persons), kalau mobil punya tempat duduk lebih dari 4 orang, pelanggan tidak jadi membelinya. harga beli (buying) dan biaya pemeliharaan (maint) menjadi pertimbangan dan masuk ke dalam evaluasi. Kapasitas bagasi (lug_boot) dan Jumlah pintu (doors) adalah pertimbangan terakhir dalam mengevaluasi mobil.

7. Refferesi

1. Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science

2. https://towardsdatascience.com/car-evaluation-analysis-using-decision-tree-classifier-61a8ff12bf6f

2.Explorasi dataset

1.Tampilkan 10 data pertama

In [1]:
```
# Nenampilkan data
from pandas import read_csv
filename = "car.data"
names = ['buying' , 'maint' , 'doors' , 'persons', 'lug_boot', 'safety', 'class']
data = read_csv(filename, names=names)
peek = data.head(10)
print(peek)
```

```
  buying  maint doors persons lug_boot safety  class
0  vhigh  vhigh     2       2    small    low  unacc
1  vhigh  vhigh     2       2    small    med  unacc
2  vhigh  vhigh     2       2    small   high  unacc
3  vhigh  vhigh     2       2      med    low  unacc
4  vhigh  vhigh     2       2      med    med  unacc
5  vhigh  vhigh     2       2      med   high  unacc
6  vhigh  vhigh     2       2      big    low  unacc
7  vhigh  vhigh     2       2      big    med  unacc
8  vhigh  vhigh     2       2      big   high  unacc
9  vhigh  vhigh     2       4    small    low  unacc
```

Cek missing value pada dataset

In [2]:
```
print(data.isnull().sum())
```

```
buying      0
maint       0
doors       0
persons     0
lug_boot    0
safety      0
class       0
dtype: int64
```

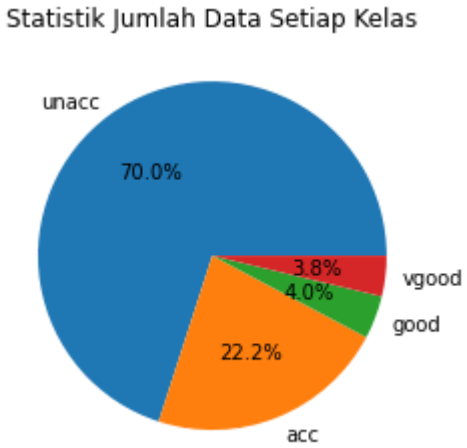
Cek baris duplicate dan distribusi kelas

In [3]:
```
print(data.duplicated().sum())

# check class distribution
print(data["class"].value_counts())
```

```
0
unacc    1210
acc       384
good       69
vgood      65
Name: class, dtype: int64
```

1. Menampilkan statistik Jumlah data dari setiap kelas pada dataset tersebut dalam Pie Chart

In [4]:
```
import matplotlib.pyplot as plt

class_counts = data['class'].value_counts()
labels = ['unacc', 'acc', 'good', 'vgood']
sizes = [class_counts[0], class_counts[1], class_counts[2], class_counts[3]]

plt.pie(sizes, labels=labels, autopct='%1.1f%%')
plt.title('Statistik Jumlah Data Setiap Kelas')
plt.show()
```



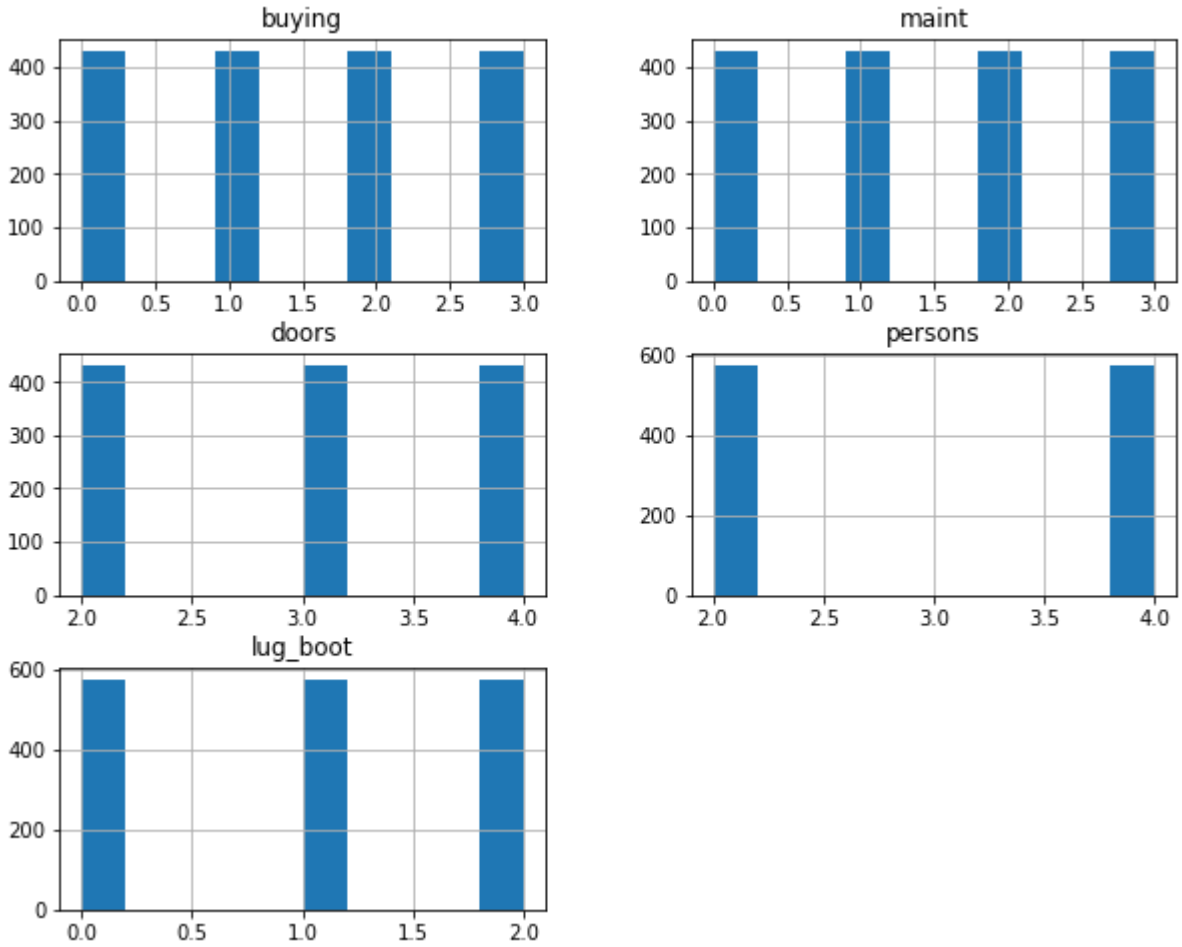
3.Menampilkan statistik (Mean, Median, Mode, Standard Deviation, dan Kuartil) dari setiap fitur data . karena data berupa string maka convert data string menjadi numeric

In [5]:
```
import pandas as pd

# convert non-numeric features to numeric
data["buying"] = pd.Categorical(data["buying"], categories=["low", "med", "high", "vhi
data["maint"] = pd.Categorical(data["maint"], categories=["low", "med", "high", "vhigh
data["doors"] = pd.to_numeric(data["doors"], errors="coerce")
data["persons"] = pd.to_numeric(data["persons"], errors="coerce")
data["lug_boot"] = pd.Categorical(data["lug_boot"], categories=["small", "med", "big"]
data["safety"] = pd.Categorical(data["safety"], categories=["low", "med", "high"], orc
data["class"] = pd.Categorical(data["class"], categories=["unacc", "acc", "good", "vgo

# display statistics
print("Mean:")
print(data.mean())
print("\nMedian:")
print(data.median())
print("\nMode:")
print(data.mode())
print("\nStandard Deviation:")
print(data.std())
print("\nQuartiles:")
print(data.quantile([0.25, 0.5, 0.75]))
```

```
Mean:
buying      1.500000
maint       1.500000
doors       3.000000
persons     3.000000
lug_boot    1.000000
safety      1.000000
class       0.414931
dtype: float64

Median:
buying      1.5
maint       1.5
doors       3.0
persons     3.0
lug_boot    1.0
safety      1.0
class       0.0
dtype: float64

Mode:
   buying  maint  doors  persons  lug_boot  safety  class
0       0      0    2.0      2.0       0.0     0.0    0.0
1       1      1    3.0      4.0       1.0     1.0    NaN
2       2      2    4.0      NaN       2.0     2.0    NaN
3       3      3    NaN      NaN       NaN     NaN    NaN

Standard Deviation:
buying      1.118358
maint       1.118358
doors       0.816812
persons     1.000434
lug_boot    0.816733
safety      0.816733
class       0.740700
dtype: float64

Quartiles:
      buying  maint  doors  persons  lug_boot  safety  class
0.25    0.75   0.75    2.0      2.0       0.0     0.0    0.0
0.50    1.50   1.50    3.0      3.0       1.0     1.0    0.0
0.75    2.25   2.25    4.0      4.0       2.0     2.0    1.0
```

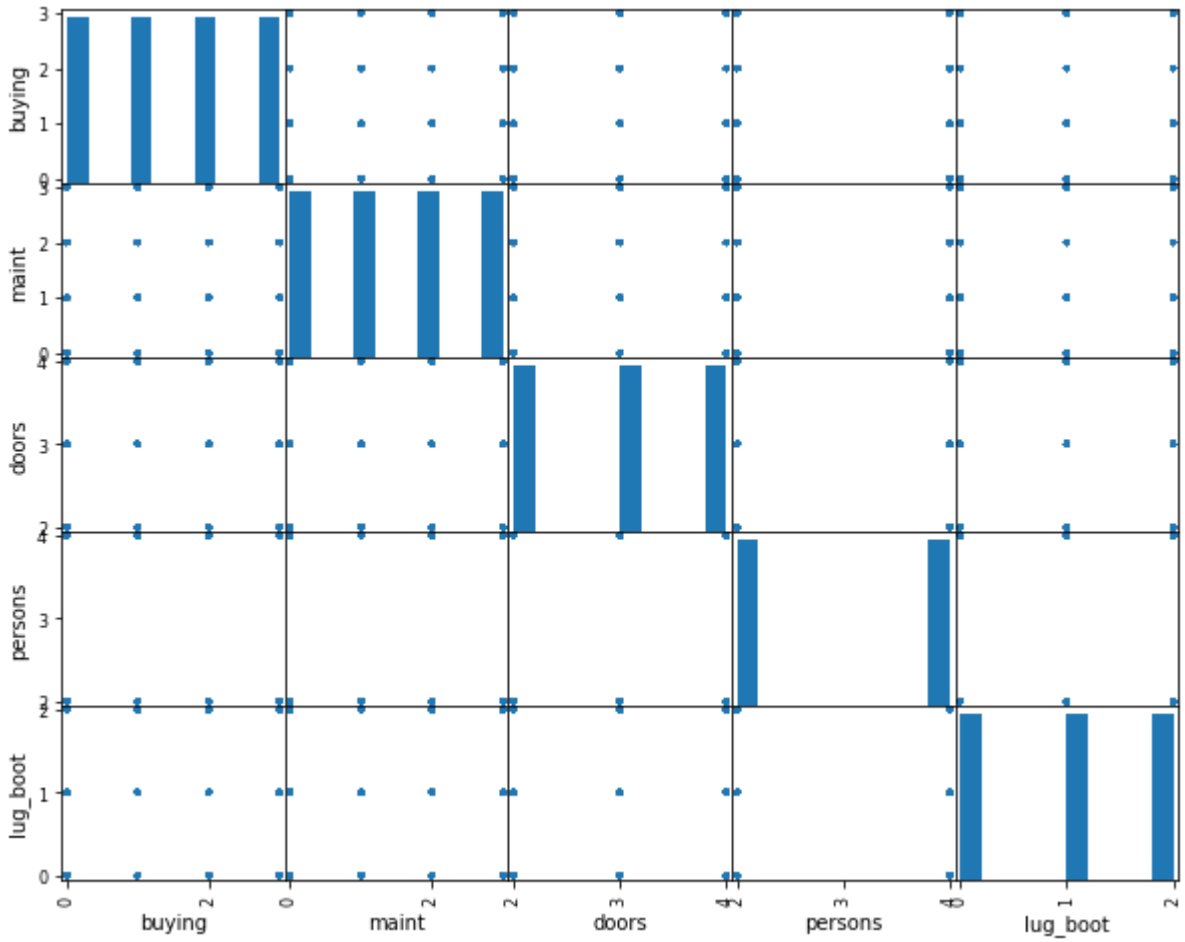
1. Menampilkan distribusi dari fitur menggunakan Plot histogram. Pada histogram ini hanya menampilkan 5 fitur pertama saja

In [6]:
```
# plot histogram of the first five features
data.iloc[:,:5].hist(bins=10, figsize=(10, 8))
plt.show()
```



5.Menampilkan scatter plot relasi antar fitur dari data. Hanya menampilkan 5 fitur pertama saja

In [7]:
```
# scatter plot of the first five features
pd.plotting.scatter_matrix(data.iloc[:,:5], alpha=0.5, figsize=(10, 8))
plt.show()
```



Car Evaluation Classification using Random Forest

Load library

In [8]:
```
import matplotlib.pyplot as plt # data visualization
import seaborn as sns # statistical data visualization
%matplotlib inline
import warnings

warnings.filterwarnings('ignore')
```

Load dataset

In [9]:
```
data = 'car.data'
df = pd.read_csv(data, header=None)
df.shape
```

Out[9]:
```
(1728, 7)
```

In [10]:
```
df.head()
```

Out[10]:

	0	1	2	3	4	5	6
0	vhigh	vhigh	2	2	small	low	unacc
1	vhigh	vhigh	2	2	small	med	unacc
2	vhigh	vhigh	2	2	small	high	unacc
3	vhigh	vhigh	2	2	med	low	unacc
4	vhigh	vhigh	2	2	med	med	unacc

In [11]:
```
col_names = ['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety', 'class']
df.columns = col_names
col_names
```

Out[11]:
```
['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety', 'class']
```

In [12]:
```
df.head()
```

Out[12]:

	buying	maint	doors	persons	lug_boot	safety	class
0	vhigh	vhigh	2	2	small	low	unacc
1	vhigh	vhigh	2	2	small	med	unacc
2	vhigh	vhigh	2	2	small	high	unacc
3	vhigh	vhigh	2	2	med	low	unacc
4	vhigh	vhigh	2	2	med	med	unacc

In [13]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1728 entries, 0 to 1727
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   buying    1728 non-null   object
 1   maint     1728 non-null   object
 2   doors     1728 non-null   object
 3   persons   1728 non-null   object
 4   lug_boot  1728 non-null   object
 5   safety    1728 non-null   object
 6   class     1728 non-null   object
dtypes: object(7)
memory usage: 94.6+ KB
```

In [14]:
```
col_names = ['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety', 'class']
for col in col_names:
    print(df[col].value_counts())
```

```
vhigh    432
high     432
med      432
low      432
Name: buying, dtype: int64
vhigh    432
high     432
med      432
low      432
Name: maint, dtype: int64
2        432
3        432
4        432
5more    432
Name: doors, dtype: int64
2       576
4       576
more    576
Name: persons, dtype: int64
small    576
med      576
big      576
Name: lug_boot, dtype: int64
low     576
med     576
high    576
Name: safety, dtype: int64
unacc    1210
acc       384
good       69
vgood      65
Name: class, dtype: int64
```

In [15]:
```
df['class'].value_counts()
```

Out[15]:
```
unacc    1210
acc       384
good       69
vgood      65
Name: class, dtype: int64
```

In [16]:
```
df.isnull().sum()
```

Out[16]:
```
buying      0
maint       0
doors       0
persons     0
lug_boot    0
safety      0
class       0
dtype: int64
```

In [17]:
```
X = df.drop(['class'], axis=1)
y = df['class']
```

Splitt dataset menjadi train dan test

In [18]:
```
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_stat
X_train.shape, X_test.shape
```

Out[18]:
```
((1382, 6), (346, 6))
```

In [19]:
```
X_train.dtypes
```

Out[19]:
```
buying      object
maint       object
doors       object
persons     object
lug_boot    object
safety      object
dtype: object
```

In [20]:
```
X_train.head()
```

Out[20]:

	buying	maint	doors	persons	lug_boot	safety
107	vhigh	vhigh	5more	more	big	high
901	med	vhigh	3	4	small	med
1709	low	low	5more	2	big	high
706	high	med	4	2	med	med
678	high	med	3	2	med	low

Pra-processing menggunakan Ordinal Encoding

kami menemukan bahwa semua variabel dalam format kategori. Meskipun pengklasifikasi pohon keputusan dapat menangani variabel format kategori dan numerik, paket scikit-learn yang akan kita gunakan tidak dapat secara langsung menangani variabel kategori. Jadi, kita harus melakukan proses konversi terlebih dahulu.

In [21]:
```
import category_encoders as ce

encoder = ce.OrdinalEncoder(cols=['buying', 'maint', 'doors', 'persons', 'lug_boot', '
X_train = encoder.fit_transform(X_train)
X_test = encoder.transform(X_test)
```

In [22]:
```
X_train.head()
```

Out[22]:

	buying	maint	doors	persons	lug_boot	safety
107	1	1	1	1	1	1
901	2	1	2	2	2	2
1709	3	2	1	3	1	1
706	4	3	3	3	3	2
678	4	3	2	3	3	3

In [23]:
```
X_test.head()
```

Out[23]:

	buying	maint	doors	persons	lug_boot	safety
599	4	4	3	3	3	1
1201	2	2	4	2	3	2
628	4	4	1	3	1	2
1498	3	4	1	2	3	2
1263	2	2	3	1	3	3

Melakukan pengklasifikasian menggunakan random forest

In [24]:
```
from sklearn.ensemble import RandomForestClassifier
# instantiate the classifier
rfc = RandomForestClassifier(random_state=0)

# fit the model
rfc.fit(X_train, y_train)

# Predict the Test set results
y_pred = rfc.predict(X_test)

# Check accuracy score
from sklearn.metrics import accuracy_score
print('Model accuracy score with 10 decision-trees : {0:0.4f}'. format(accuracy_score(
```

```
Model accuracy score with 10 decision-trees : 0.9624
```

In [25]:
```
rfc_100 = RandomForestClassifier(n_estimators=100, random_state=0)

# fit the model to the training set
rfc_100.fit(X_train, y_train)

# Predict on the test set results
y_pred_100 = rfc_100.predict(X_test)

# Check accuracy score
print('Model accuracy score with 100 decision-trees : {0:0.4f}'. format(accuracy_score
```

```
Model accuracy score with 100 decision-trees : 0.9624
```

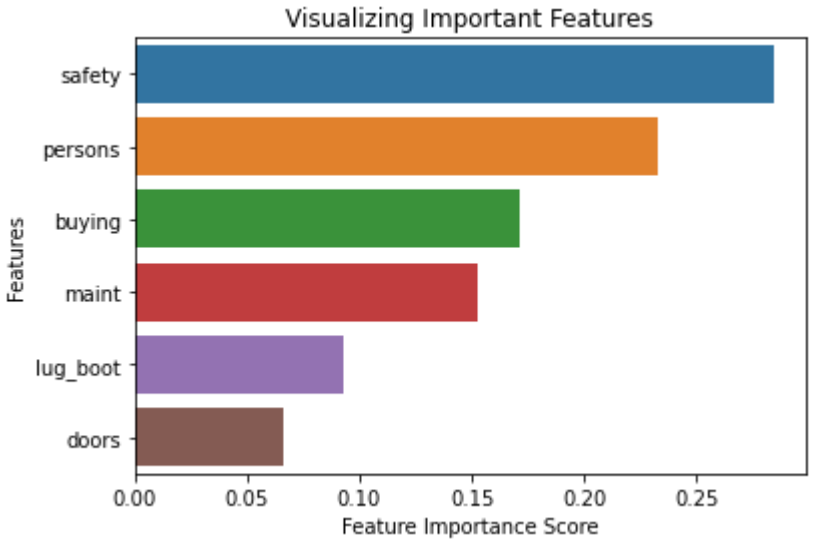
In [26]:
```
clf = RandomForestClassifier(n_estimators=100, random_state=0)
clf.fit(X_train, y_train)
```

Out[26]:
```
RandomForestClassifier(random_state=0)
```

In [27]:
```
feature_scores = pd.Series(clf.feature_importances_, index=X_train.columns).sort_value
feature_scores
```

Out[27]:
```
safety      0.284409
persons     0.232452
buying      0.171266
maint       0.152731
lug_boot    0.093004
doors       0.066137
dtype: float64
```

In [28]:
```
sns.barplot(x=feature_scores, y=feature_scores.index)

# Add labels to the graph
plt.xlabel('Feature Importance Score')
plt.ylabel('Features')

# Add title to the graph
plt.title("Visualizing Important Features")

# Visualize the graph
plt.show()
```



confusion matrix untuk menemukan kinerja model

In [29]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print('Confusion matrix\n\n', cm)
```

```
Confusion matrix

 [[ 74   6   3   0]
 [  0  10   0   1]
 [  1   0 234   0]
 [  2   0   0  15]]
```

Print classification report

In [30]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
acc	0.96	0.89	0.92	83
good	0.62	0.91	0.74	11
unacc	0.99	1.00	0.99	235
vgood	0.94	0.88	0.91	17
accuracy			0.96	346
macro avg	0.88	0.92	0.89	346
weighted avg	0.97	0.96	0.96	346

In [ ]: