Overview

• Objective: 'Car-Dataset is used to classify the car acceptability into classes: unacceptable, acceptable, good and very good.

• Methodology: Get the different classification model of car data. With the help of the build model, we will be able to identify and classify test dataset into different classes.

Dataset and EDA analysis

• There are **6 features (independent) and 1 class variable (dependent variable)** in the car dataset. •
Size of the dataset: **(1728, 7)**
• Total number of records in dataset = **1728**
• Total number of attributes in dataset = **7**
• What kind of data – **Categorical (binary/nominal Variable): ['buying', 'maint', 'doors', 'persons', 'lug_boot', 'safety']**
• Class

- **class 0: acc (384)**
- **class 1: good (69)**
- **class 2: unacc (1210)**
- **class 3: vgood (65)**
-

• Check for missing data and preprocessing challenges:

KNNImputer with neighbour: 5 - For Numeric variable
KNN imputation (n_neighbour = 5 means that the missing values will be replaced by the mean value of 5 nearest neighbors)
Mode imputation: for Categorical/Nominal variable

#Scaling and normalization of features

Numerical variable: Standardize using z-score (StandardScaler) normalization
Categorical variable: categorical variables are not to normalize (to avoid losing the nature of categorical variable), hence **creating dummy variables/ one-hot encoding for categorical features**

#Encoding target class using label encoding

• Distribution of Training set, testing set:

- Training Dataset (1382, 15) (1382,)
- Testing Dataset (346, 15) (346,)

Descriptive and correlation analysis

Descriptive analytics

```
Shape of dataset: (1728, 7)
Total number of records in dataset = 1728
Total number of attributes in dataset = 7

There is no missing values in the dataset.


Data Top Head
  buying  maint doors persons lug_boot safety classNames
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


{'buying': ['vhigh', ' high', ' med', ' low.'], 'maint': ['vhigh', ' high', ' med', ' low.'], 'doors': ['2', ' 3', ' 4', ' 5mor
e.'], 'persons': ['2', ' 4', ' more.'], 'lug_boot': ['small', ' med', ' big.'], 'safety': ['low', ' med', ' high.']} ['unacc',
'acc', 'good', 'vgood']


Data Description
        buying maint doors persons lug_boot safety classNames
count     1728  1728  1728    1728     1728   1728       1728
unique       4     4     4       3        3      3          4
top        low   low     2       2      big    low      unacc
freq       432   432   432     576      576    576       1210
------------------------------------------------------------------------
```

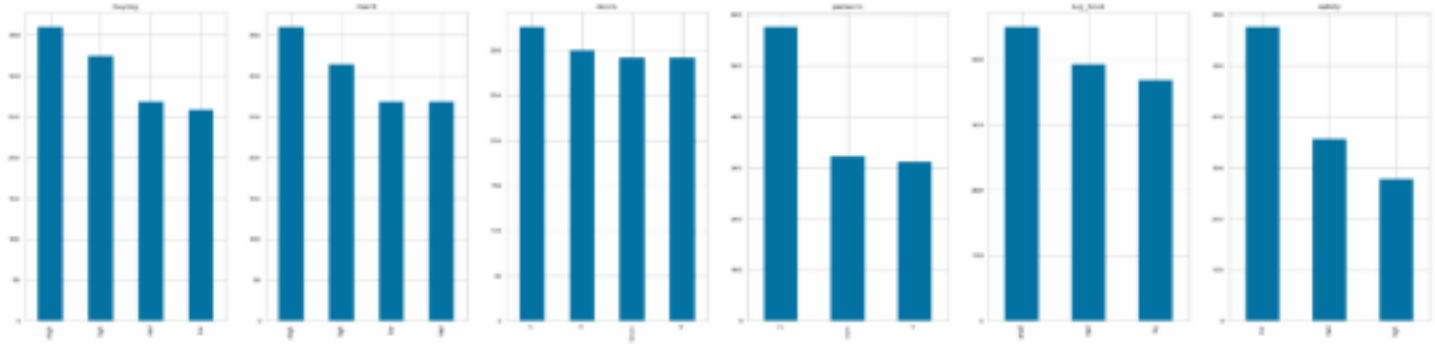
Null value check

```
Are there missing values in Target Class? False

Are there missing values in the features?
buying          False
maint           False
doors           False
persons         False
lug_boot        False
safety          False
classNames      False
dtype: bool

Now, Are there any missing values in Features? buying_low        False
buying_med         False
buying_vhigh       False
maint_low          False
maint_med          False
maint_vhigh        False
doors_3            False
doors_4            False
doors_5more        False
persons_4          False
persons_more       False
lug_boot_med       False
lug_boot_small     False
safety_low         False
safety_med         False
dtype: bool
```

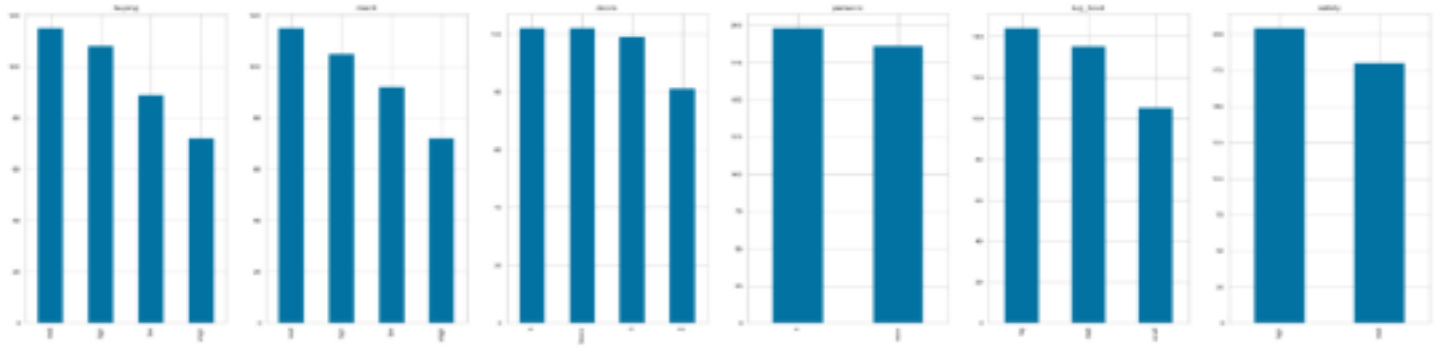
Correlation analysis and plot

Distribution of class with respect to independent/ feature variables

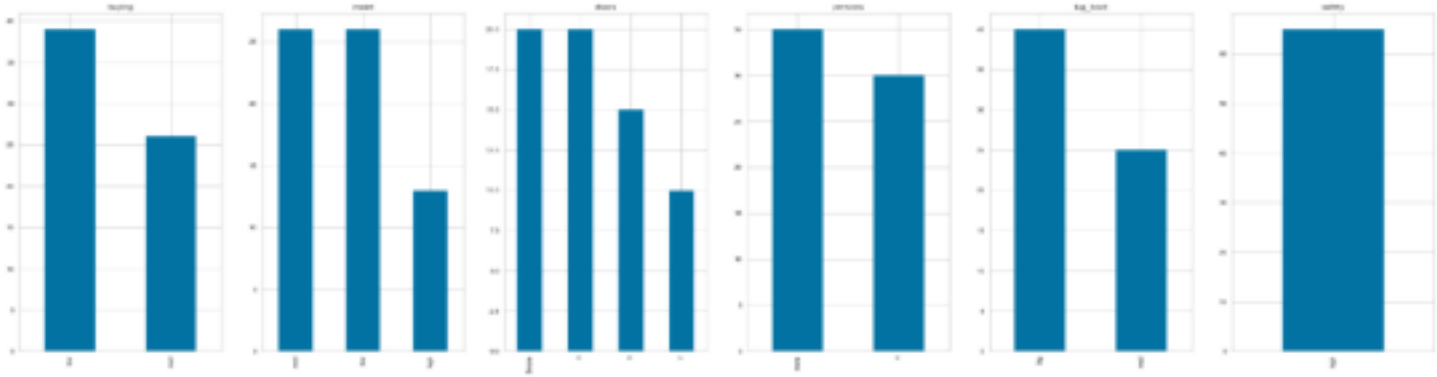
unacc



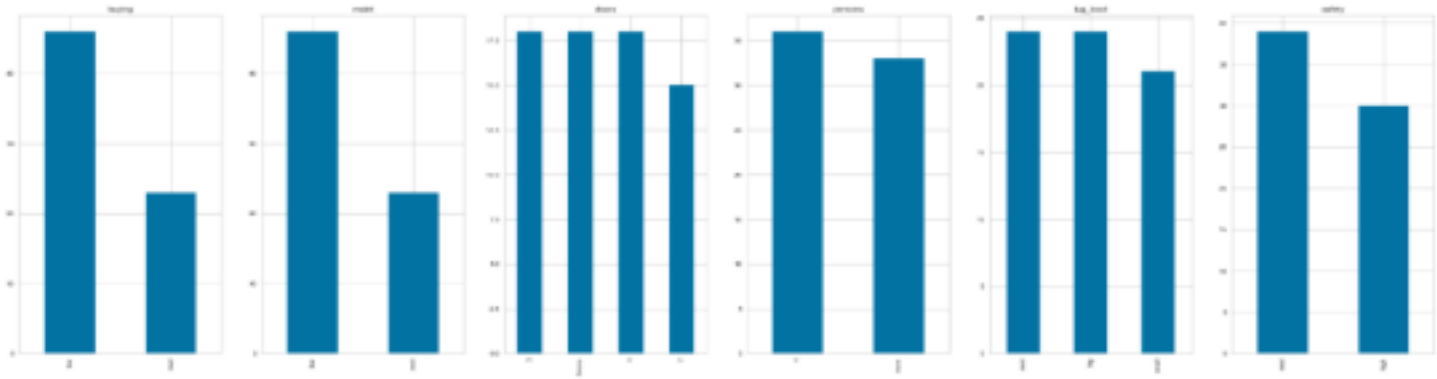
acc



vgood



good



Machine Learning Methodology

• We applied four machine learning technique

1. Used Logistic regression machine learning technique: Logistic regression is a statistical analysis method used to predict a data value based on prior observations of a data set. Logistic regression has become an important tool in the discipline of machine learning. The approach allows an algorithm being used in a machine learning application to classify incoming data based on historical data. As more relevant data comes in, the algorithm should get better at predicting classifications within data sets.

In the **multiclass case,** the training algorithm uses the **one-vs-rest (OvR) scheme** if the 'multi_class' option is set to '**ovr'** and uses the cross-entropy loss if the 'multi_class' option is set to 'multinomial'.

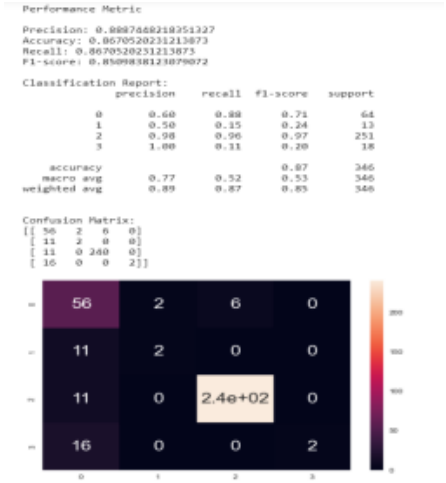
2. A **random forest** is a machine learning technique that's used to solve regression and classification problems. It utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems. A random forest algorithm consists of many decision trees.

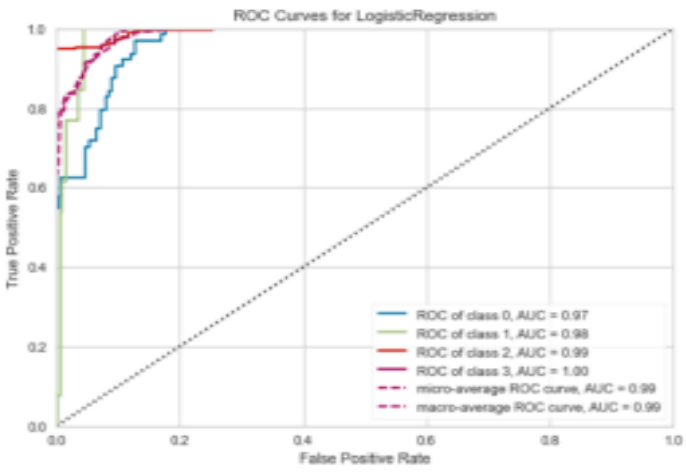
3. Used **k-nearest neighbors (KNN) algorithm, KNN** is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

4. SVM classifier SVM or Support Vector Machine is a linear model for classification and regression problems. It can solve linear and non-linear problems and work well for many practical problems. The idea of SVM is simple: The algorithm creates a line or a hyperplane which separates the data into classes.

Result comparison

• Logistic Regression Model (**multi_class='ovr'**)

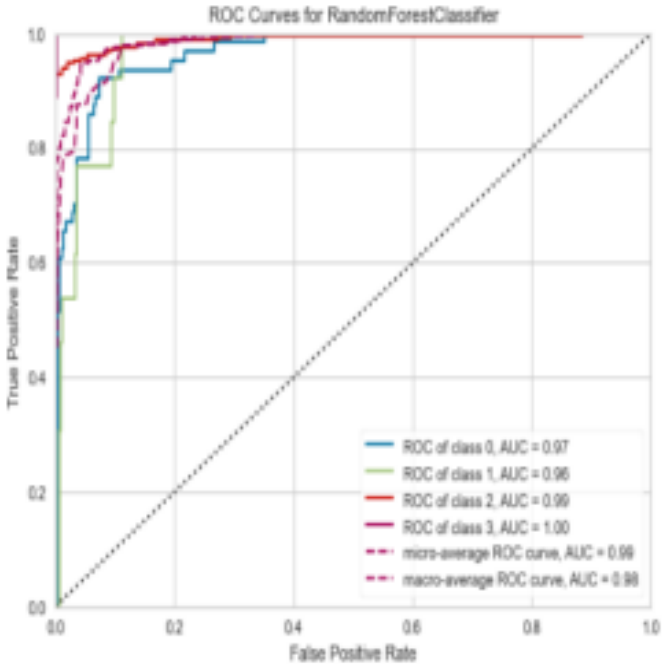




Result comparison

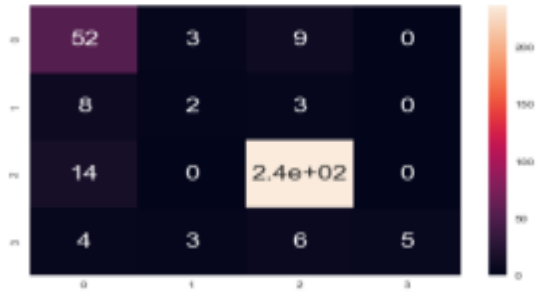
• Random Forest Classifier (**max_depth=10**)

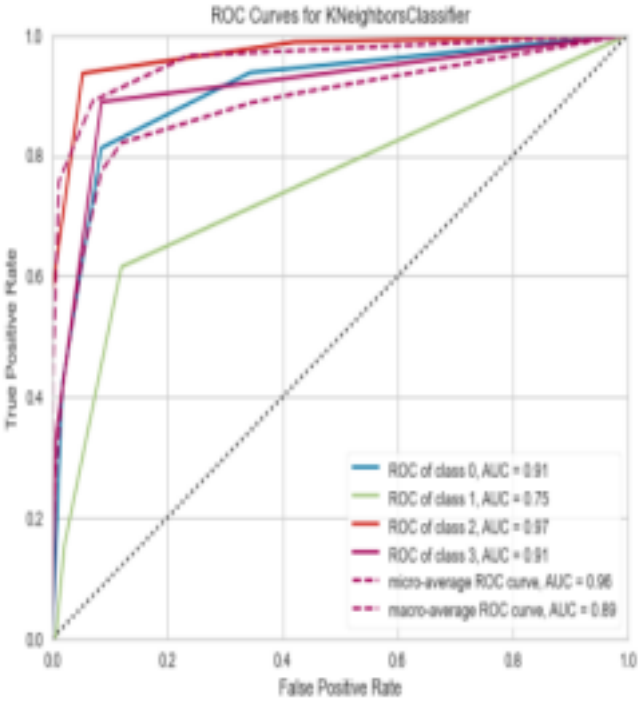
```
Performance Metric

Precision: 0.9129873922217621
Accuracy: 0.9075144508670521
Recall: 0.9075144508670521
F1-score: 0.9063635292180499

Classification Report:
              precision    recall  f1-score   support

           0       0.74      0.88      0.80        64
           1       0.60      0.46      0.52        13
           2       0.97      0.96      0.96       251
           3       1.00      0.61      0.76        18

    accuracy                           0.91       346
   macro avg       0.83      0.73      0.76       346
weighted avg       0.91      0.91      0.91       346

Confusion Matrix:
[[ 56   2   6   0]
 [  5   6   2   0]
 [ 10   0 241   0]
 [  5   2   0  11]]
```



• KNN Classifier (**target_class['classNames'].max()+1**)

```
Performance Metric

Precision: 0.8589567040689108
Accuracy: 0.8554913294797688
Recall: 0.8554913294797688
F1-score: 0.8448034655571407

Classification Report:
              precision    recall  f1-score   support

           0       0.67      0.81      0.73        64
           1       0.25      0.15      0.19        13
           2       0.93      0.94      0.94       251
           3       1.00      0.28      0.43        18

    accuracy                           0.86       346
   macro avg       0.71      0.55      0.57       346
weighted avg       0.86      0.86      0.84       346

Confusion Matrix:
[[ 52   3   9   0]
 [  8   2   3   0]
 [ 14   0 237   0]
 [  4   3   6   5]]
```

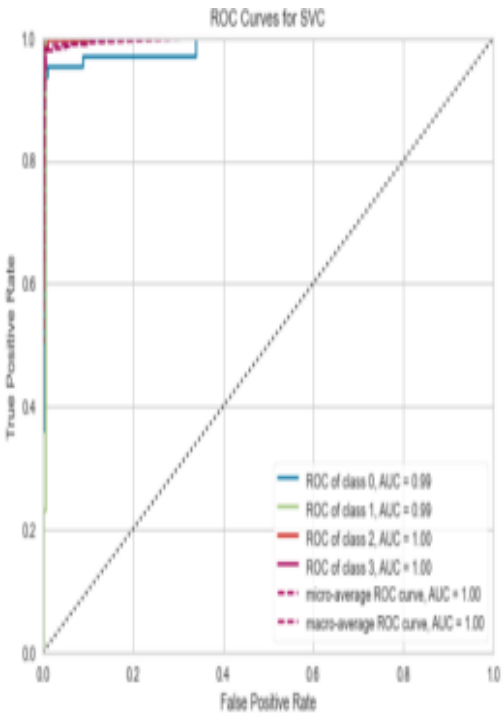




Result comparison

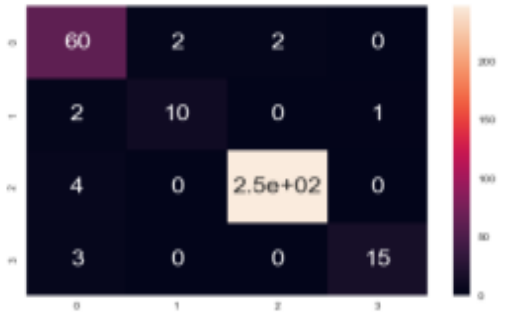
- SVM Classifier (**kernel='poly'**)

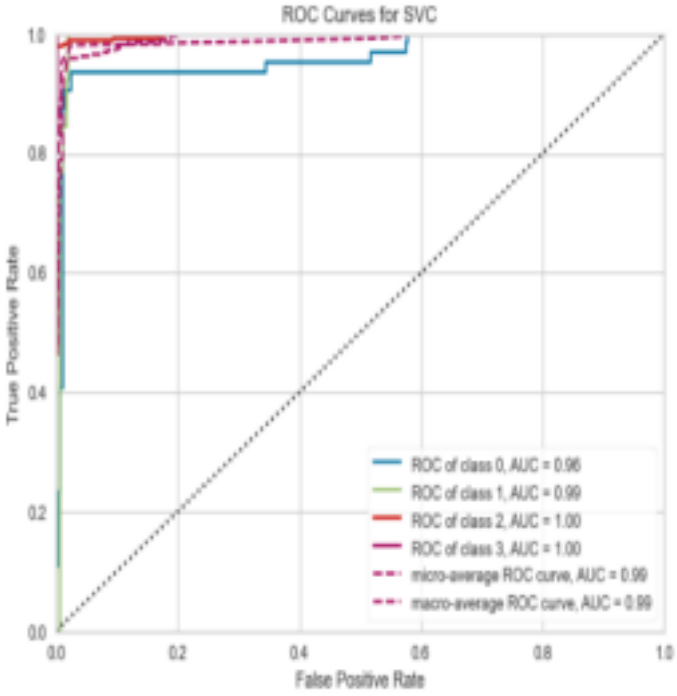
```
Performance Metric

Precision: 0.9807173558610645
Accuracy: 0.9797687861271677
Recall: 0.9797687861271677
F1-score: 0.9796595280867561

Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.95      0.95        64
           1       0.87      1.00      0.93        13
           2       1.00      1.00      1.00       251
           3       1.00      0.83      0.91        18

    accuracy                           0.98       346
   macro avg       0.95      0.95      0.94       346
weighted avg       0.98      0.98      0.98       346

Confusion Matrix:
[[ 61   2   1   0]
 [  0  13   0   0]
 [  1   0 250   0]
 [  3   0   0  15]]
```



- KNN Classifier (**kernel='rbf'**)

```
Precision: 0.9605330722235511
Accuracy: 0.9595375722543352
Recall: 0.9595375722543352
F1-score: 0.9595000978133876

Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.94      0.90        64
           1       0.83      0.77      0.80        13
           2       0.99      0.98      0.99       251
           3       0.94      0.83      0.88        18

    accuracy                           0.96       346
   macro avg       0.91      0.88      0.89       346
weighted avg       0.96      0.96      0.96       346

Confusion Matrix:
[[ 60   2   2   0]
 [  2  10   0   1]
 [  4   0 247   0]
 [  3   0   0  15]]
```





Conclusion and Recommendation

Result Analysis

From the four-model used for analysis, we have found: SVM (RBF and Polynomial) kernel-based model perform good compared to Logistic, KNN, and Random Forest model-- based on the computed:

1. **weighted precision value of:**

- 98%: Polynomial kernel svm
- 96%: RBF kernel based svm

2. **weighted accuracy value of:**

- 98%: Polynomial kernel svm
- 96%: RBF kernel based svm

We have also found Random Forest performance is comparable with SVM model output.

- For the production environment, we could deploy SVM based kernel method for car dataset classification