**INTERNSHIP**

**PROJECT REPORT**

**Car Model Acceptance Using AI-ML Model**

****

**Hexnbit Online Internship**

**www.hexnbit.com**

*Intern Detail*

Car Model Acceptance Using AI-ML Model

**Name of Project:** ………………………………………………………………………………………………………………….

Sudhish Subramaniam

**Name of Intern:**  ………………………………………………………………………………………………………………….

9594322793

**Intern’s Ph. no.:**  ………………………………………………………………………………………………………………….

1204, Shree Ambika Complex, Ghansoli, sector -, plot number 18, Navi Mumbai: 400701

**Intern’s Add.:**  ………………………………………………………………………………………………………………….

Vellore Institute of Technology, VIT Vellore (632014)

**Name of College:** ………………………………………………………………………………………………………………….

VIT, Vellore Campus, Tiruvalam Rd, Katpadi, Vellore, Tamil Nadu 632014

**College’s Add.: ……..**………………………………………………………………………………………………………….

Electronics and Communication Engineering

**Branch:**  ………………………………………………………………………………………………………………….

23/08/2021

**Project Submission Date:** ………………………..

*Table of content*

|  |  |  |
| --- | --- | --- |
| S.No. | Headings | Page No. |
| 1 | *Project Algorithm* | 1 |
| 2 | *List of Software used and libraries imported* | 2 |
| 3 | *Flow chart* | 3 |
| 4 | *Results, inferences and future scope* | 4 |
| 5 | *Code* | 5 |
| 6 | *List of References* | 17 |

*Project Algorithm*

**Algorithm:**

1. Read Dataset
2. Check for Missing Data
3. Handle Categorical Values
4. Split Data for Training and Testing
5. Apply different Classification Algorithms and tune them
6. Get performance metrics for all the applied classifiers
7. Visually compare the performance of all classifiers

*List of Software used and libraries imported*

**List of Software:**

* Jupyter Notebook

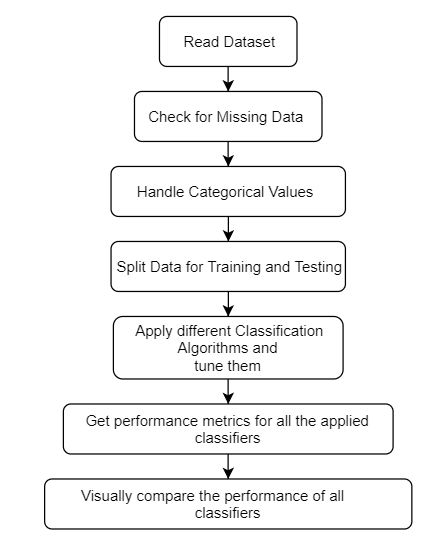
**Dataset:**

<https://archive.ics.uci.edu/ml/datasets/Car+Evaluation>

**Libraries Imported:**

1. Pandas
2. train\_test\_split from sklearn.model\_selection
3. KNeighborsClassifier from sklearn.neighbors
4. LogisticRegression from sklearn.linear\_model
5. confusion\_matrix,accuracy\_score from sklearn.metrics
6. matplotlib.pyplot

*Flow chart*

****

*Results, inferences and future scope*

**Results:**

**No missing data in the dataset.**

**Accuracy scores:**

1. KNeighbors Classifier = 88.4%
2. Logistic Regression = 69.5%

Sensitivity and specificity for the dataset when random forest classifier is used is 1, 0.8235 respectively. We can see that random forest classifier gives best results for this dataset.

**Inferences**:

1. "unacc" here is the output column which decides whether the car is unacc, acc, good or v-good.
2. KNeighbors Classifier gives the best results in terms of classification.

**Future Scope:**

1. Identification of Cars will be easier.
2. Cars can be easily segregated or classified using KNeighbors Classifier.
3. KNeighbors Classifier can be used in more applications.
4. More categorical values can be added.
5. A dataset with more values may give more accurate results, more the data more accurate the results.

*Code*

#!/usr/bin/env python

# coding: utf-8

# <a href='https://www.hexnbit.com/'> <img src='https://www.hexnbit.com/wp-content/uploads/2019/09/hexnbit\_final\_66px.png'/> </a>

# All cells must be suitably commented / documented.

# ### Read Dataset

# In[101]:

import pandas as pd

# In[102]:

# Importing all the libraries

# In[103]:

df1 = pd.read\_csv("car.data")

df1.head()

# In[104]:

# Reading the dataset

# ### Check for Missing Data

# In[105]:

df1.isnull() # Check for Missing Data

# In[106]:

df1.isnull().sum() # Counting the number of Missing Data

# ### Handle Categorical Values

# In[107]:

df1['vhigh'].unique()

# In[108]:

df1['vhigh.1'].unique()

# In[109]:

df1['2'].unique()

# In[110]:

df1['2.1'].unique()

# In[111]:

df1['small'].unique()

# In[112]:

df1['low'].unique()

# In[113]:

df1['unacc'].unique()

# In[114]:

# Checking for missing data

# In[115]:

df1.info()

# In[116]:

df1['vhigh']=df1['vhigh'].astype("category")

df1['vhigh.1']=df1['vhigh.1'].astype("category")

df1['2']=df1['2'].astype("category")

df1['2.1']=df1['2.1'].astype("category")

df1['small']=df1['small'].astype("category")

df1['low']=df1['low'].astype("category")

df1['unacc']=df1['unacc'].astype("category")

df1.info()

# In[117]:

# Converting the onject to category data type

# In[118]:

dict(enumerate(df1['vhigh'].cat.categories))

# In[119]:

dict(enumerate(df1['vhigh.1'].cat.categories))

# In[120]:

dict(enumerate(df1['2'].cat.categories))

# In[121]:

dict(enumerate(df1['2.1'].cat.categories))

# In[122]:

dict(enumerate(df1['small'].cat.categories))

# In[123]:

dict(enumerate(df1['low'].cat.categories))

# In[124]:

dict(enumerate(df1['unacc'].cat.categories))

# In[125]:

# Displaying the categoreis in which individual columns are converted into

# In[126]:

df1['vhigh']=df1['vhigh'].cat.codes

df1['vhigh.1']=df1['vhigh.1'].cat.codes

df1['2']=df1['2'].cat.codes

df1['2.1']=df1['2.1'].cat.codes

df1['small']=df1['small'].cat.codes

df1['low']=df1['low'].cat.codes

df1['unacc']=df1['unacc'].cat.codes

df1.info()

# In[127]:

# Converting the each column values into categories

# ### Split Data for Training and Testing

# In[128]:

output = df1.pop('unacc')

# In[129]:

# Segregating the output column

# In[130]:

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(df1,output,test\_size=0.3,random\_state=0)

# In[131]:

# Code for splitting the dataset into train and test

# ### Apply different Classification Algorithms and tune them

# In[132]:

from sklearn.neighbors import KNeighborsClassifier

# In[133]:

# Importing KNeighborsClassifier

# In[134]:

knn=KNeighborsClassifier()

knn.fit(x\_train,y\_train)

y\_pred1 = knn.predict(x\_test)

# In[135]:

# Applying KNeighborsClassifier and predicting using KNeighborsClassifier

# In[136]:

from sklearn.linear\_model import LogisticRegression

regressor = LogisticRegression()

regressor.fit(x\_train ,y\_train)

# In[137]:

# Applying Logistic Regression algorithm

# In[138]:

y\_pred2 = regressor.predict(x\_test)

# In[139]:

# Predicting using Logistic Regression

# ### Get performance metrics for all the applied classifiers

# In[140]:

from sklearn.metrics import confusion\_matrix,accuracy\_score # Formula for confusion matrix

# In[141]:

cm1=confusion\_matrix(y\_test,y\_pred1)

cm1

# In[142]:

# Calculating the confusion matrix of KNeighborsClassifier

# In[143]:

cm2 = confusion\_matrix(y\_test,y\_pred2)

print(cm2)

# In[144]:

# Calculating the confusion matrix for Logistic Regression

# In[145]:

accuracy\_score(y\_test,y\_pred1)

# In[146]:

# Calculating the accuracy score of KNeighborsClassifier

# In[147]:

accuracy\_score(y\_test,y\_pred2)

# In[148]:

# Calculating the accuracy score for Logistic Regression

# ### Visually compare the performance of all classifiers

# In[149]:

import matplotlib.pyplot as plt

# In[150]:

courses = ['KNeighbors Classifier','Logistic Regression']

values = [0.884393063583815,0.6955684007707129]

fig = plt.figure(figsize = (10, 5))

plt.bar(courses, values, color ='maroon',width = 0.4)

plt.xlabel("Algorithms")

plt.ylabel("Accuracy Score")

plt.title("Visually Comparing the performance of all classifiers")

plt.show()

# In[151]:

# Creating the bar plot to Visually Comparing the performance of all classifiers

*List of References*

* <https://dzone.com/articles/introduction-to-classification-algorithms>
* <https://www.hexnbit.com/>
* [https://data-flair.training/blogs/machine-learning-classification-algorithms](https://data-flair.training/blogs/machine-learning-classification-algorithms/)/
* https://www.analyticsvidhya.com/blog/2021/05/5-classification-algorithms-you-should-know-introductory-guide/