(ECE 418) Machine Learning Project

Team Members:

Yeshwanth sai Gokarakonda (AP19110030008, Mechanical)

Raja Venkata Pramod (AP19110030017, Mechanical)

Prajwal Katakam (AP19110010286, CSE-E)

Devendra Sai (AP19110010279, CSE-E)

Sai Krishnam Raju (AP19110010267, CSE-E)

Importing Libraries

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

from warnings import filterwarnings
filterwarnings(action='ignore')
```

Loading Dataset

```
wine = pd.read_csv("/content/winequality-red.csv")
print("Successfully Imported Data!")
wine.head()
```

```
Successfully Imported Data!
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	s
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	

Description and Finding Null Values

```
print(wine.shape)
wine.describe(include='all')
print(wine.isna().sum())
```

```
(1599, 12)
fixed acidity          0
```

```
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

```
wine.groupby('quality').mean()
```

quality	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	dens
3	8.360000	0.884500	0.171000	2.635000	0.122500	11.000000	24.900000	0.997
4	7.779245	0.693962	0.174151	2.694340	0.090679	12.264151	36.245283	0.996
5	8.167254	0.577041	0.243686	2.528855	0.092736	16.983847	56.513950	0.997
6	8.347179	0.497484	0.273824	2.477194	0.084956	15.711599	40.869906	0.996
7	8.872362	0.403920	0.375176	2.720603	0.076588	14.045226	35.020101	0.996

Feature Selection

```
# Create Classification version of target variable
wine['goodquality'] = [1 if x >= 7 else 0 for x in wine['quality']]# Separate feature vari
X = wine.drop(['quality','goodquality'], axis = 1)
Y = wine['goodquality']
```

```
# See proportion of good vs bad wines
wine['goodquality'].value_counts()
```

```
0    1382
1     217
Name: goodquality, dtype: int64
```

```
X # attributes which are responsible for classification
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH
0	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51
1	7.8	0.880	0.00	2.6	0.098	25.0	67.0	0.99680	3.20
2	7.8	0.760	0.04	2.3	0.092	15.0	54.0	0.99700	3.26
3	11.2	0.280	0.56	1.9	0.075	17.0	60.0	0.99800	3.16
4	7.4	0.700	0.00	1.9	0.076	11.0	34.0	0.99780	3.51
...	...	...	...	...	...	...	...	...	...
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45

```
print(Y)
```

```
0       0
1       0
2       0
3       0
4       0
       ..
1594    0
1595    0
1596    0
1597    0
1598    0
Name: goodquality, Length: 1599, dtype: int64
```

Feature Importance

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()

from sklearn.ensemble import ExtraTreesClassifier
classifiern = ExtraTreesClassifier()
classifiern.fit(X,Y)
score = classifiern.feature_importances_
print(score)
```

```
[0.07556531 0.09627568 0.0986353  0.07197105 0.06738145 0.06913588
 0.0876835  0.08411249 0.06778224 0.10949398 0.17196312]
```

Splitting Dataset

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size=0.3,random_state=7)
```

LogisticRegression:

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_train,Y_train)
Y_pred = model.predict(X_test)

from sklearn.metrics import accuracy_score,confusion_matrix
print("Accuracy Score:",accuracy_score(Y_test,Y_pred))
```

```
Accuracy Score: 0.8729166666666667
```

```
confusion_mat = confusion_matrix(Y_test,Y_pred)
print(confusion_mat)
```

```
[[399  18]
 [ 43  20]]
```

Using KNN:

```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=3)
model.fit(X_train,Y_train)
y_pred = model.predict(X_test)

from sklearn.metrics import accuracy_score
print("Accuracy Score:",accuracy_score(Y_test,y_pred))
```

```
Accuracy Score: 0.8729166666666667
```

Using SVM:

```
from sklearn.svm import SVC
model = SVC()
model.fit(X_train,Y_train)
pred_y = model.predict(X_test)

from sklearn.metrics import accuracy_score
print("Accuracy Score:",accuracy_score(Y_test,pred_y))
```

```
Accuracy Score: 0.86875
```

Using Decision Tree:

```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier(criterion='entropy',random_state=7)
model.fit(X_train,Y_train)
y_pred = model.predict(X_test)

from sklearn.metrics import accuracy_score
print("Accuracy Score:",accuracy_score(Y_test,y_pred))
```

```
Accuracy Score: 0.8645833333333334
```

Using Random Forest:

```
from sklearn.ensemble import RandomForestClassifier
model2 = RandomForestClassifier(random_state=1)
model2.fit(X_train, Y_train)
y_pred2 = model2.predict(X_test)

from sklearn.metrics import accuracy_score
print("Accuracy Score:",accuracy_score(Y_test,y_pred2))
```

```
Accuracy Score: 0.89375
```

✓ 0s completed at 1:19 AM