```
# -*- coding: utf-8 -*-
"""
@date: 9/19/23
@author: Jayson Rhea

"""

import numpy as np
import tensorflow as tf
from tensorflow import keras
from sklearn.tree import *
from sklearn.model_selection import *
from sklearn.metrics import *
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from imblearn.over_sampling import RandomOverSampler
from sklearn.utils.class_weight import compute_class_weight

data = pd.read_csv('Norm_QB_Data_ML.csv')

X = data.iloc[:, :-1]
y = data.iloc[:,-1]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.6, random_state=42)

#Setting Class Weights
class_weights = compute_class_weight('balanced', classes=np.unique(y), y=y)


#Oversampling/Augmentation
oversampler = RandomOverSampler(sampling_strategy='all')
X_resampled, y_resampled = oversampler.fit_resample(X_train, y_train)

#Training with class weights
clf = SVC(class_weight={'R': class_weights[0], 'HR': class_weights[1], 'DNR': class_weights[2]
a = clf.fit(X_resampled, y_resampled)

#KNN
k = 3  # Number of neighbors
knn_classifier = KNeighborsClassifier(n_neighbors=k)
b = knn_classifier.fit(X_train, y_train)
y_pred = knn_classifier.predict(X_resampled)
report = classification_report(y_resampled, y_pred)

"""
### KNN
              precision    recall  f1-score   support

         DNR       0.62      0.93      0.74        14
          HR       1.00      0.64      0.78        14
           R       0.50      0.43      0.46        14

    accuracy                           0.67        42
   macro avg       0.71      0.67      0.66        42
weighted avg       0.71      0.67      0.66        42
```

```
This report indicates that KNN has an issue with finding the true value for
"recommended QBs." It also displays some strength in identifying
"Highly Recommended QBs."
"""



#DECISION TREE CLASSIFIER
classifier = DecisionTreeClassifier(random_state=42)
classifier.fit(X_train, y_train)
y_pred_DTclassifier = classifier.predict(X_resampled)
report_DTclassifier = classification_report(y_resampled, y_pred_DTclassifier)


"""
### Decision Tree Classifier
              precision    recall  f1-score   support

         DNR       0.62      0.93      0.74        14
          HR       1.00      0.64      0.78        14
           R       0.50      0.43      0.46        14

    accuracy                           0.67        42
   macro avg       0.71      0.67      0.66        42
weighted avg       0.71      0.67      0.66        42

This report indicates that my sample space is too small and
my data displays "overfitting."
"""



#LOGISTICS REGRESSION
model_LR = LogisticRegression(random_state=42, solver='lbfgs', max_iter=500)
model_LR.fit(X_train, y_train)
y_pred_LR = model_LR.predict(X_resampled)
report_LR = classification_report(y_resampled, y_pred_LR)

"""
### Logistics Regression
              precision    recall  f1-score   support

         DNR       0.72      0.93      0.81        14
          HR       1.00      1.00      1.00        14
           R       0.90      0.64      0.75        14

    accuracy                           0.86        42
   macro avg       0.87      0.86      0.85        42
weighted avg       0.87      0.86      0.85        42

This report indicates that Logistics Regression can predict "HR" perfectly while
confidently predicting R is still a struggle.
"""



#SUPPORT VECTOR MODEL
model_SVM = SVC(kernel='rbf', random_state=42)
model_SVM.fit(X_train, y_train)
```

```
y_pred_SVM = model_SVM.predict(X_resampled)
report_SVM = classification_report(y_resampled, y_pred_SVM)

"""
### Suppport Vector Model
              precision    recall  f1-score   support

         DNR       0.67      1.00      0.80        14
          HR       1.00      1.00      1.00        14
           R       1.00      0.50      0.67        14

    accuracy                           0.83        42
   macro avg       0.89      0.83      0.82        42
weighted avg       0.89      0.83      0.82        42
"""

#ARTIFICIAL NEURAL NETWORKS

#Number of neurons

c = 3 # Number of classes
nf = 15 #Number of features

N = (c + nf) / 3 #number of neurons

mlp = MLPClassifier(hidden_layer_sizes=(1,int(N)),
activation='tanh', max_iter=10000, random_state=10)
mlp.fit(X_train, y_train)
y_pred_ANN = mlp.predict(X_resampled)
report_ANN = classification_report(y_resampled, y_pred_ANN)
print(report_ANN)


"""
### Artificial Neural Networks
              precision    recall  f1-score   support

         DNR       0.82      1.00      0.90        14
          HR       0.58      1.00      0.74        14
           R       1.00      0.07      0.13        14

    accuracy                           0.69        42
   macro avg       0.80      0.69      0.59        42
weighted avg       0.80      0.69      0.59        42
"""
```