```
# -*- coding: utf-8 -*-
"""
Created on Tue Aug 25 15:50:05 2018

@author: Walter Stevens

Simple example of a deep neural net built using Keras using a public data set of how US congressmen
voted on 17 different issues in the year 1984.
Based on how they voted, I try to determine their political party.

"""



from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout



from sklearn.model_selection import cross_val_score
from tensorflow.keras.wrappers.scikit_learn import KerasClassifier

import pandas as pd

feature_names =  ['party','handicapped-infants', 'water-project-cost-sharing',
                    'adoption-of-the-budget-resolution', 'physician-fee-freeze',
                    'el-salvador-aid', 'religious-groups-in-schools',
                    'anti-satellite-test-ban', 'aid-to-nicaraguan-contras',
                    'mx-missle', 'immigration', 'synfuels-corporation-cutback',
                    'education-spending', 'superfund-right-to-sue', 'crime',
                    'duty-free-exports', 'export-administration-act-south-africa']
```

```
voting_data = pd.read_csv('house-votes-84.data.txt', na_values=['?'],
                          names = feature_names)
#voting_data.head()

voting_data.dropna(inplace=True)
#voting_data.describe()

voting_data.replace(('y', 'n'), (1, 0), inplace=True)
voting_data.replace(('democrat', 'republican'), (1, 0), inplace=True)

all_features = voting_data[feature_names].drop('party', axis=1).values
all_classes = voting_data['party'].values


def create_model():
    model = Sequential()
    #16 the 16 votes serves as inputs going into an 32-unit layer
    model.add(Dense(32, input_dim=16, kernel_initializer='normal', activation='relu'))
    # Another hidden layer of 16 units
    model.add(Dense(16, kernel_initializer='normal', activation='relu'))
    # Output layer with a binary classification (Democrat or Republican political party)
    model.add(Dense(1, kernel_initializer='normal', activation='sigmoid'))
    # Compile model
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model

# Wrap the Keras model in an estimator compatible with scikit_learn
estimator = KerasClassifier(build_fn=create_model, epochs=100, verbose=0)
```

```
# Now use scikit_learn's cross_val_score to evaluate this model
cv_scores = cross_val_score(estimator, all_features, all_classes, cv=10)
cv_scores.mean()

"""
cv_scores.mean()
Out[34]: 0.9394927561283112

"""
```