## ESPY PHANTOM

Diagnosis of false pregnancy which is known as phantom.There are various explanations, none of which is universally accepted because of the complex involvement of [cortical](https://en.wikipedia.org/wiki/Cortex_(anatomy)" \o "Cortex (anatomy)), [hypothalamic](https://en.wikipedia.org/wiki/Hypothalamic" \o "Hypothalamic), [endocrine](https://en.wikipedia.org/wiki/Endocrine" \o "Endocrine), and [psychogenic](https://en.wikipedia.org/wiki/Psychogenic_disease" \o "Psychogenic disease)factors.[[2]](https://en.wikipedia.org/wiki/False_pregnancy" \l "cite_note-2) Proposed mechanisms include the effect of stress on the [hypothalamo-pituitary-adrenal axis](https://en.wikipedia.org/wiki/Hypothalamic-pituitary-adrenal_axis" \o "Hypothalamic-pituitary-adrenal axis), [constipation](https://en.wikipedia.org/wiki/Constipation" \o "Constipation), weight gain, and the movement of intestinal gas.so we are going implement this diagnosis under the image processing technique which meant to be clustering on the basis of k-means algorithm

**Source code:**

import tissues

from keras.models import Sequential

from keras.layers import Dense

from keras.wrappers.scikit\_learn import KerasClassifier

from keras.utils import np\_utils

from sklearn.model\_selection import cross\_val\_score

from sklearn.model\_selection import KFold

from sklearn.preprocessing import LabelEncoder

from sklearn.pipeline import Pipeline

tissue = 7

numpy.random.tissue(tissue)

dataframe = pandas.read\_csv("phantom.csv", header=None)

dataset = dataframe.values

X = dataset[:,0:4].astype(float)

Y = dataset[:,4]

encoder = LabelEncoder()

encoder.fit(Y)

encoded\_Y = encoder.transform(Y)

# convert integers to dummy variables (i.e. one hot encoded)

dummy\_y = np\_utils.to\_categorical(encoded\_Y)

def baseline\_model():

# create model

model = Sequential()

model.add(Dense(8, input\_dim=4, activation='relu'))

model.add(Dense(3, activation='softmax'))

# Compile modelmodel.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])

return model

kfold = KFold(n\_splits=10, shuffle=True, random\_state=seed)

Results = cross\_val\_score(estimator, X, dummy\_y, cv=kfold)

print("Baseline: %.2f%% (%.2f%%)" % (results.mean()\*100, results.std()\*100))