# DATA SCIENCE TECHNIQUES AND APPLICATIONS

## Coursework II: Multi-dimensional analysis and dimensionality reduction on a Kaggle dataset

Eirini Zygoura
Student ID: 13177951



### Phase 1: Selection of dimensions

In part 1 of the coursework I selected to work with the “Breast Cancer Wisconsin - benign or malignant” dataset [1]. The dataset contains 683 observations of 9 cytological characteristics of breast fine-needle aspirates as predictor variables and 1 target variable to indicate whether a cancer cell is benign or malignant.

Feature Importance obtained by an Extra Trees Classifier indicated the 3 most important predictors in descending relevance order:

- Bare nuclei
- Uniformity of cell shape and
- Uniformity of cell size

These predictors are highly correlated with the target class, a fact that consolidates the selection obtained by the Classifier.

In Figure 1 the scatterplots of the 3 dominant predictors are presented, with respect to the target class. Also, density plots for the predictors are shown.

Classes are not separated perfectly, but in a satisfactory level. Generally, low values (<4) for each feature indicate benign cancer cells (target class value 2) and higher values malignant cells (target class value 4). However there are feature values where benign and malignant cells coexist and there is no clear class separation. So, predictions considering only one of the features will lead to lower accuracy, rather than taking into account all the dominant dimensions. The level of separation of the classes can also be seen in Figure 2, where there is a 3D representation of the data space of the dominant dimensions.

The density plots report highest densities for the benign class, corresponding to the fact that the dataset is unbalanced in favour of that class (reference to part 1 of the coursework, phase 3: Data quality).

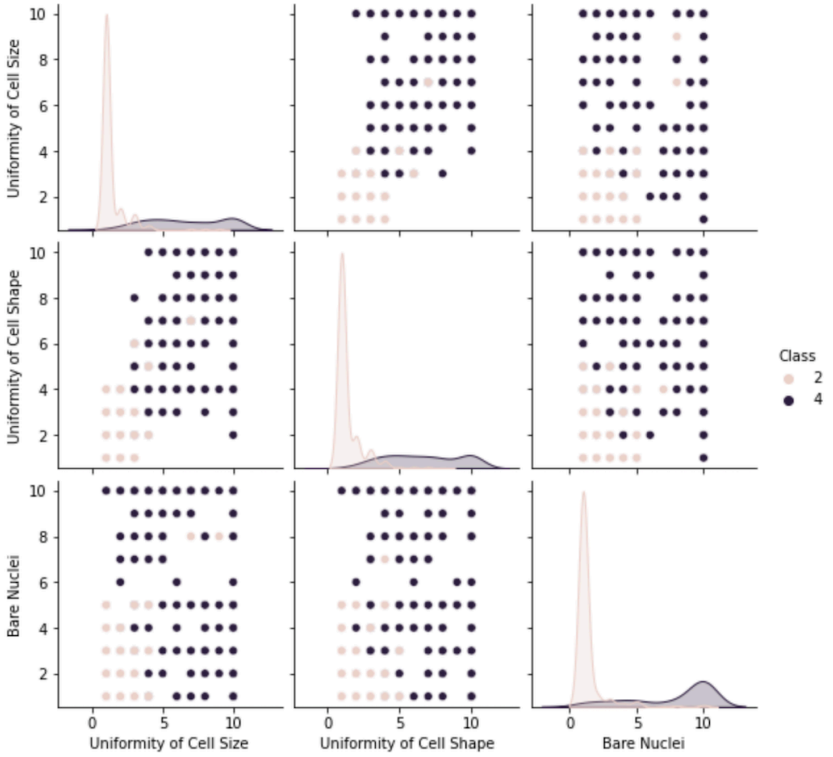


Figure 1

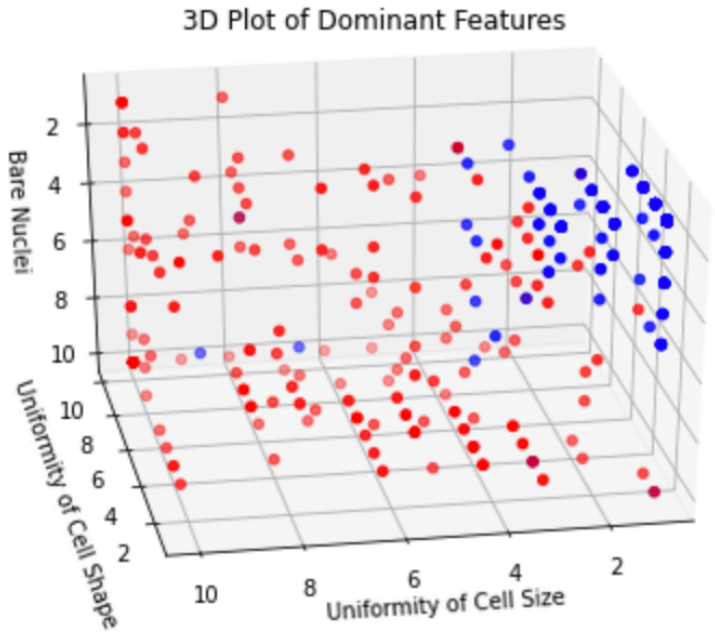


Figure 2

## Phase 2: Principal Component Analysis

Before applying Principal Component Analysis to the selected data, feature scaling is required. As mentioned in Scikit-learn documentation [2], PCA is “a prime example of when normalisation is important”. Although there is not great difference in standard deviations of

the dominant features (coursework part 1, Fig.3), I choose to proceed with scaling the data for better results. I used the StandardScaler class in order to achieve mean of 0 and standard deviation of 1 for the features.

In order to verify that the selection to work with the 3 dominant features is correct I performed PCA:

- using all the dimensions and
- on the 3 chosen features.

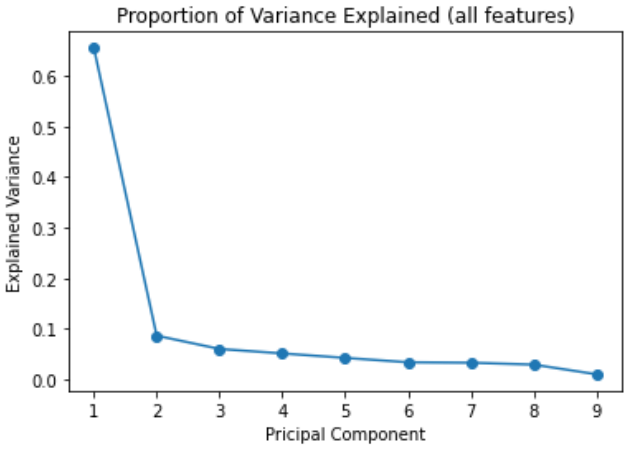


Figure 3

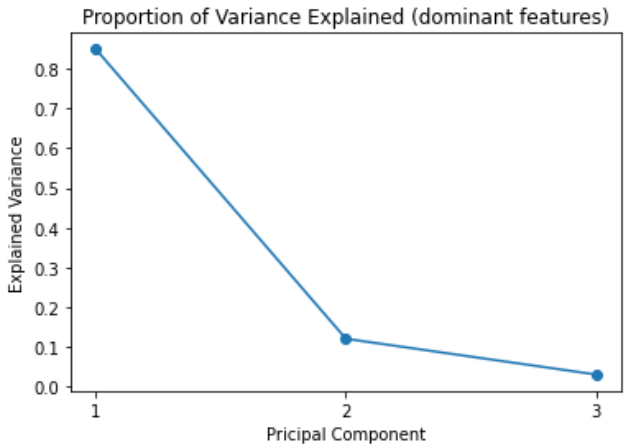


Figure 4

Results presented in Figures 3 and 4 show the proportion of variance that is explained by each principal component. For the first case the variance explained by the first principal component is only 65.5%, whereas in the second case is 84.9%. Taking into account the first 2 principal components while considering all the predictors 74.1% of the variance is explained and it reaches up to 96.9% for the dominant features. While considering the three most dominant predictors of the dataset almost all the necessary information that can be obtained is included in the first two principal components and we can achieve dimensionality reduction to 2-D for the feature data space. Which means less processing time and cost of resources and easier interpretation.

## **Phase 3: PCA results**

Figures 5 and 6 are the 3-D representations of the 3 dominant eigenvectors obtained from all the predictors and the selected set of the predictors respectively. Comparing with Figure 2 there is a better separation of the classes performing PCA. However, there is still a mixture of the two classes, slightly improved in case of using only the selected features.

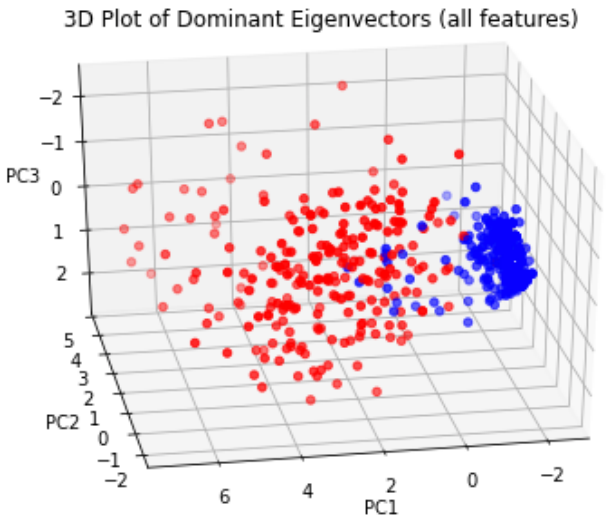


Figure 5

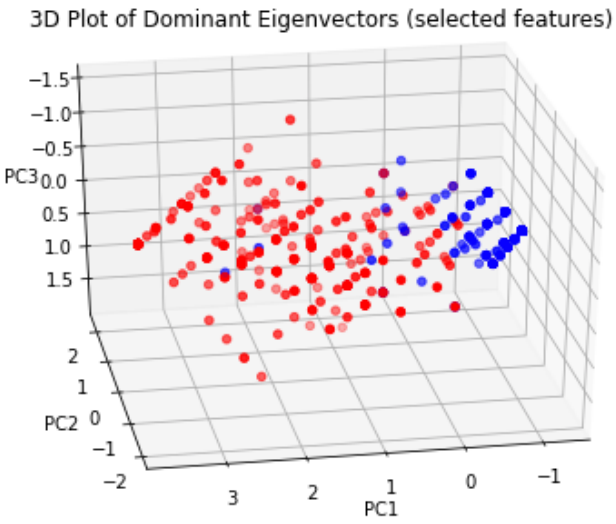


Figure 6

In order to evaluate the effect of the dimensionality reduction on predicting the type of a cancer cell due to its characteristics we need to apply a classification model. I chose to apply a SVM classifier using Radial Basis function as kernel parameter to a randomly selected subset of the constructed data space and evaluate the model in the rest data (split in 70-30 %). As evaluation metrics I chose Accuracy and Error Type 2 (False Negative) which is of major importance in cancer prediction. The model was applied to the 3-D constructed data spaces from all the features and from the selected features, after applying PCA. Results are presented in Table 1.

| SVM trained on data from: | Accuracy | Error Type 2 |
|---|---|---|
| **All features** | 0.980 | 1 |
| **PCA to all features** | 0.976 | 1 |
| **PCA to selected features** | 0.956 | 4 |

Table 1

PCA slightly reduced the performance of the classification model while performing to all the features. This could be expected as we considered only the 3 out of 9 principal components while performing the classification and information was lost. It is interesting and almost unexpected that after applying PCA to the 3 selected features and keeping again 3 principal components accuracy was decreased (by small amount) and False Negative predictions increased. It should be noted that with 3 selected features and 3 principal components maximum level of accuracy can be achieved as we have full

representation of the data space. So, decreasing the value of principal components accuracy will be decreased and Type 2 Errors will be increased.

Since PCA's goal the dimensionality reduction and that is achieved in a good level whether we perform it on all the 9 features or the 3 dominant ones, and then chose a smaller subset of the 2 or 3 most dominant eigenvectors.

In the "Breast Cancer Wisconsin - benign or malignant" dataset I suggest performing PCA to all the predictors, and then reproducing the data space on 3 or 4 dimensions. Furthermore, it seems that the 3 selected predictors are not enough for correct classification, as:

- evaluation metrics are worst (Table 1),
- in the density plots (Figure 1) there are areas with no clear class separation for the 3 selected features
- feature importance for Marginal Adhesion is smaller but still comparable to the importance of Uniformity of Cell Size and selecting it may improve the performance of the model. The same applies to the next 4 features in the order which have similar importance values.

**REFERENCES**


1. Kaggle Breast Cancer Wisconsin dataset, available online at: *https://www.kaggle.com/ninjacoding/breast-cancer-wisconsin-benign-or-malignant*

2. Importance of Feature Scaling, Scikit-learn documentation, available online at: *https://scikit-learn.org/stable/auto_examples/preprocessing/plot_scaling_importance.html*

## APPENDIX

PYTHON CODE USED:

```
# -*- coding: utf-8 -*-
"""DSTA 2 .ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1P2u8OPmd5MRBuusfFabY9eAF5ErH6pm3
"""

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from pandas import read_csv
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix

# load dataset
data= pd.read_csv("tumor.csv")
data = data.drop(columns='Sample code number')

# Get the name of each column after erasing the code number column
dic_names = {index: value for index, value in enumerate(data.columns)}
dic_names

# separate predictors and target
X = data.drop(columns='Class')
Y = data[['Class']]

# dataset with selected predictors and target
# separate selected predictors and target
x = data[['Uniformity of Cell Size','Uniformity of Cell Shape', 'Bare Nuclei']]
y = Y

# pair plot of selected features coloured by class
sns.pairplot(data[['Uniformity of Cell Size','Uniformity of Cell Shape', 'Bare Nuclei',
'Class']],hue= 'Class')

# Create 3D Scatter Plot of dominant features
plot = Axes3D(plt.figure(1),elev=-150, azim=100)
plot.scatter(x['Uniformity of Cell Size'],x['Uniformity of Cell Shape'],x['Bare Nuclei'], c = y,
cmap = plt.cm.bwr)
```

```
plot.set_title("3D Plot of Dominant Features")
plot.set_xlabel("Uniformity of Cell Size")
plot.set_ylabel("Uniformity of Cell Shape")
plot.set_zlabel("Bare Nuclei")
plt.show()

# Standarise data

# all features
X_scaler = StandardScaler().fit(X)
X_rescaled = X_scaler.transform(X)

# selected features
x_scaler = StandardScaler().fit(x)
x_rescaled = x_scaler.transform(x)

# PCA analysis on all features
fit = PCA(n_components=9).fit(X_rescaled)
explained_variance = fit.explained_variance_ratio_
print("Explained variance: %s" % explained_variance)
print(pd.DataFrame(fit.components_))

# Variance explained Plot
plt.plot(explained_variance, marker = 'o')
plt.xlabel('Pricipal Component')
plt.xticks(np.arange(0,9,step=1), ['1','2','3','4','5','6','7','8','9'])
plt.ylabel('Explained Variance')
plt.title("Proportion of Variance Explained (all features)")
plt.show()

# Create the new dimensional space
fit = PCA(n_components=3).fit(X_rescaled)
X_PCA = fit.fit_transform(X_rescaled)

# PCA analysis on selected features
fit = PCA(n_components=3).fit(x_rescaled)
explained_variance = fit.explained_variance_ratio_
print("Explained variance: %s" % fit.explained_variance_ratio_)
print(pd.DataFrame(fit.components_))

# Variance explained Plot
plt.plot(explained_variance, marker = "o")
plt.xlabel('Pricipal Component')
plt.xticks(np.arange(0,3,step=1), ['1','2','3'])
plt.ylabel('Explained Variance')
plt.title("Proportion of Variance Explained (dominant features)")
plt.show()

# Create the new dimensional space
x_PCA = fit.fit_transform(x_rescaled)

# Create 3D Scatter Plot of eigenvectors - ALL FEATURES
```

```
plot = Axes3D(plt.figure(1),elev=-150, azim=100)
plot.scatter(X_PCA[:,0],X_PCA[:,1],X_PCA[:,2], c = y, cmap = plt.cm.bwr)
plot.set_title("3D Plot of Dominant Eigenvectors (all features)")
plot.set_xlabel("PC1")
plot.set_ylabel("PC2")
plot.set_zlabel("PC3")
plt.show()

# Create 3D Scatter Plot of eigenvectors - DOMINANT FEATURES
plot = Axes3D(plt.figure(1),elev=-150, azim=100)
plot.scatter(x_PCA[:,0],x_PCA[:,1],x_PCA[:,2], c = y, cmap = plt.cm.bwr)
plot.set_title("3D Plot of Dominant Eigenvectors (selected features)")
plot.set_xlabel("PC1")
plot.set_ylabel("PC2")
plot.set_zlabel("PC3")
plt.show()

# Split into train and test sets

# normalised data (All features before PCA)
resc_X_train, resc_X_test, resc_Y_train, resc_Y_test = train_test_split(X_rescaled, Y,
test_size=0.3, random_state=6)

# with PCA on all features
X_train, X_test, Y_train, Y_test = train_test_split(X_PCA, Y, test_size=0.3,
random_state=6)
# with PCA on dominant features
x_train, x_test, y_train, y_test = train_test_split(x_PCA, y, test_size=0.3, random_state=6)

# SVM Classification
model = SVC(kernel='rbf')

# to all data without PCA
model.fit(resc_X_train, resc_Y_train)
resc_Y_predicted = model.predict(resc_X_test)
acc = accuracy_score(resc_Y_test, resc_Y_predicted)
error_2 = confusion_matrix(resc_Y_test, resc_Y_predicted)[1,0]
print("Accuracy:", acc, " and Error Type 2:", error_2)

# to all features
model.fit(X_train, Y_train)
Y_predicted = model.predict(X_test)
acc = accuracy_score(Y_test, Y_predicted)
error_2 = confusion_matrix(Y_test, Y_predicted)[1,0]
print("Accuracy:", acc, " and Error Type 2:", error_2)

# to dominant features
model.fit(x_train, y_train)
y_predicted = model.predict(x_test)
acc = accuracy_score(y_test, y_predicted)
error_2 = confusion_matrix(y_test, y_predicted)[1,0]
print("Accuracy:", acc, " and Error Type 2:", error_2)
```