# EESTech Challenge 2017
# Patras Local Round
# Exercise 1

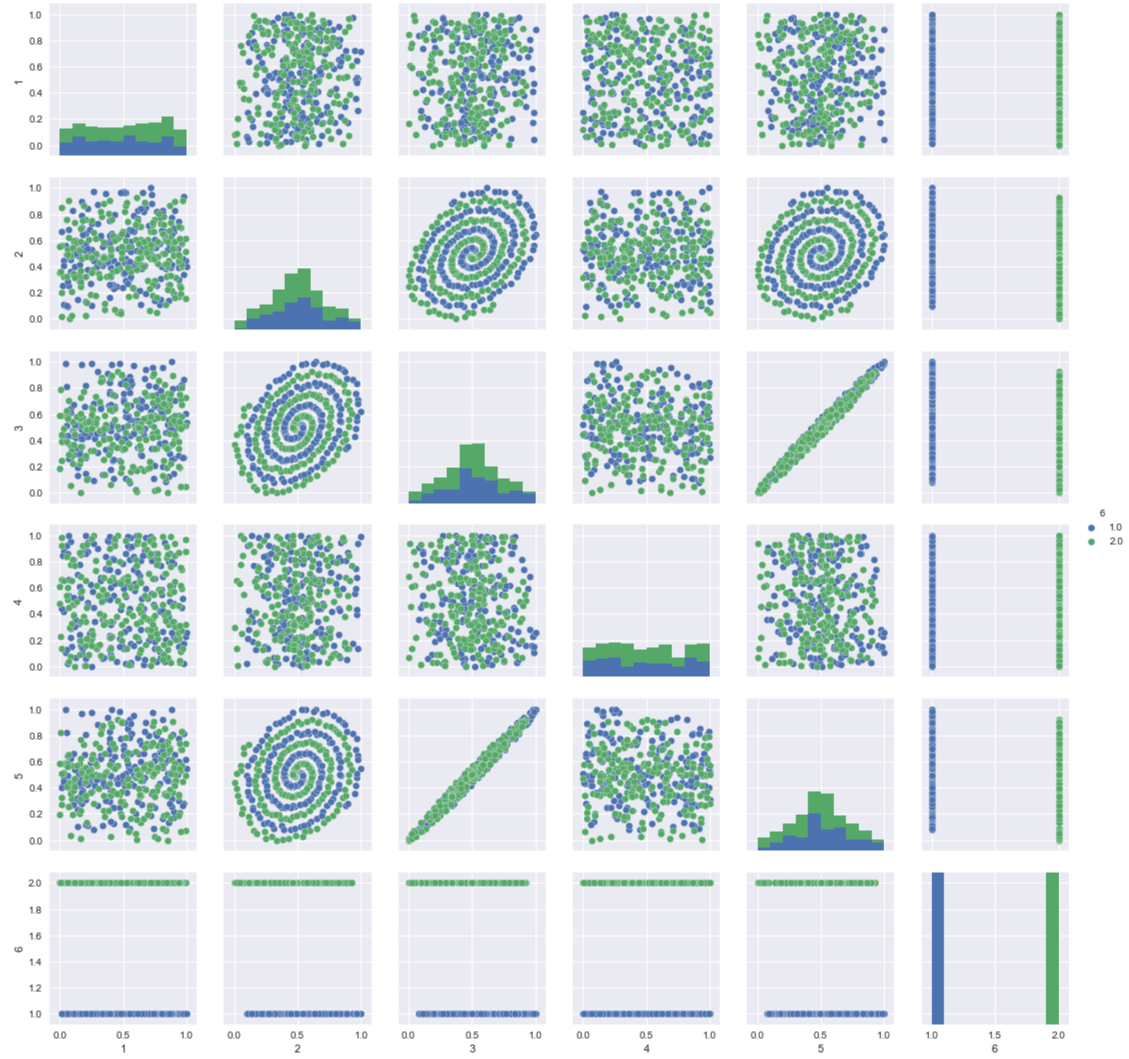
March 31, 2017

```
In [ ]: # -*- coding: utf-8 -*-
        """
        Created on Fri Mar 31 15:27:58 2017
        for eestec competition by team Bicameral Minds
        Apostolis Kemos, Spiros Kaftanis, Tilemahos Doganis
        """
```

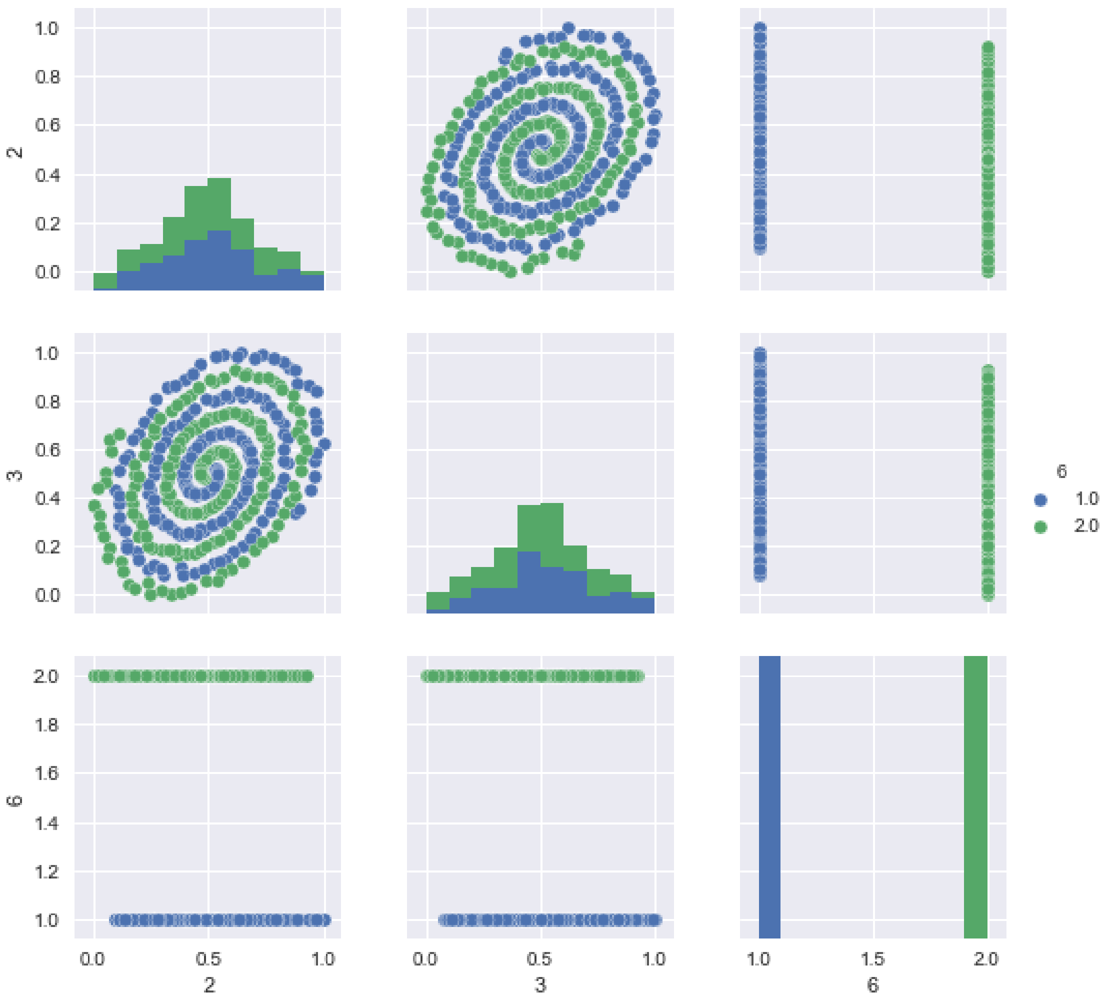
```
In [1]: #%% Read data file into pandas DataFrame
        import pandas as pd
        pd.options.display.max_columns = 100


        data = pd.read_csv('artificial.data',  sep=' ',header=None, na_filter=False)
        data = data.sample(frac=1, random_state = 5) # Shuffle data
        plot_data = data # Keep a copy of data for plotting
        labels = data[6] # Store labels
```

```
In [2]: #%% Scale Data to [0,1]
        from sklearn.preprocessing import MinMaxScaler
        mms = MinMaxScaler()
        plot_data.ix[:,1:5] = mms.fit_transform(plot_data.ix[:,1:5])

        # Original Data Visualization
        import seaborn as sns
        from matplotlib import pyplot as plt # Necessary for displaying in Jupyter
        print("Total data visualization")
        sns.set() # Set Seaborn visualization parameters (default)
        sns.pairplot(plot_data, hue=6) # Visualize pairplot of all parameters
        sns.plt.show() # For Jupyter visualization

        print("Chosen features' data visualization")
        scaled_data = plot_data.ix[:,[2,3]]
        plot_data = plot_data.ix[:,[2,3,6]]
        sns.pairplot(plot_data, hue=6)
        sns.plt.show() # For Jupyter visualization
```

Total data visualization



```
Chosen features' data visualization
```



```
In [3]: #%% Initialize SVM classifier object
        from sklearn.svm import SVC
        clf = SVC(C=2, cache_size=200, class_weight=None, coef0=0.0,
        decision_function_shape=None, degree=3, gamma=600, kernel='rbf',
        max_iter=-1, probability=False, random_state=1, shrinking=True,
        tol=0.001, verbose=False)
```

```
In [4]: #%% Split original DataFrame (400x2) into a list of ten DataFrames (40x2)
        import numpy as np
        num_folds = 10
        new_train_Xfolds = np.array_split(scaled_data, num_folds)
        new_train_Yfolds = np.array_split(labels, num_folds)
```

```
In [5]: cv_scores = []
        for j in range(num_folds):
            # Use j-th DataFrame as test set and the rest as training set.
            # Convert from list of DataFrames to 360x2 Numpy-array via 'np.vstack':
            X_train_cv = np.vstack(new_train_Xfolds[0:j]+new_train_Xfolds[j+1:])

            # Convert from DataFrame to Numpy-array via 'as_matrix()':
            X_test_cv = new_train_Xfolds[j].as_matrix()

            # Similarly for the labels' pandas Series:
            y_train_cv = np.hstack(new_train_Yfolds[0:j]+new_train_Yfolds[j+1:])
            y_test_cv = new_train_Yfolds[j].as_matrix()

            # Fit the SVM model to the training data:
            clf.fit(X_train_cv, y_train_cv)

            # Return the mean accuracy for given data and labels:
            scores_training = clf.score(X_train_cv, y_train_cv)
            score = clf.score( X_test_cv, y_test_cv)

            # Append current step's score to the list
            cv_scores.append(score) In
```

```
[6]: #%% Result output

        print("10-Fold Cross Validation Mean Score:", np.mean(cv_scores)*100,"%")
```

```
10-Fold Cross Validation Mean Score: 97.5 %
```