ARTIFICIAL INTELLIGENCE

Theme 3 - Supervised Learning

Practical Assignment 2 - Checkpoint

Stylianos Tsagkarakis // up201911231
Vasileios Konstantaras // up201911213

Presentation contents

01 Dataset

02 Specification of the work

03 Research

04 Tools and Algorithms

05 Organizing

06 Work already implemented

01 Dataset

European Soccer Database by Hugo Mathien

A soccer database that contains:

1. +25,000 matches
2. +10,000 players
3. 11 European Countries including their lead championship
4. Seasons 2008 to 2016
5. Players and Teams' attributes
6. Team line up with squad formation (X, Y coordinates)
7. Betting odds from up to 10 providers
8. Detailed match events (goal types, possession, corner, cross, fouls, cards etc...) for +10,000 matches

02 | Specification of work

- Exploratory analysis of the dataset
- Examination of the data
  - pre-processing
  - transformation
- Parameterization of supervised learning algorithms

- Algorithms comparison
- Analysis of the confusion matrix
- Demonstration of comparison through appropriate graphs
- Obtain best precision, recall, accuracy, F-measure

03 Research

kaggle

Match Outcome Prediction in Football (link)

kaggle

EUROPEAN FOOTBALL DATA ANALYSIS (link)

ResearchGate

Exploring and modelling team performances of the Kaggle European Soccer database (link)

04 | Tools and Algorithms

- Python
- Google Colab
- Scikit Learn
- Numpy
- MatplotLib
- Pandas
- Pipelines

- **Naive Bayes Classifier**
- **k Nearest Neighbors Classifier (kNN)**
- XGBoosts classifier (?)
- Gaussian Model
- **kNN with GridsearchCV**
- **MPL Classifier**

05 Organizing

Python notebook will be organized in 4 Sections:

Section A: Our Team

Section B: Introduction to the dataset

- Dataset description
- Data retrieval
- Remove empty values
- Label frequencies
- Over / under sample
- Split to test / train set

Section C: Baseline classification

- **kNN Classifier**
- Dummy Classifier
- **Naive Bayes Classifier**
- XGBoosts classifier (?)
- **MPL Classifier**
- Gaussian Model

Section D: Optimizing classifiers

- Pre-processing
- Balance dataset
- Standardization
- Variance Threshold
- Scaling
- GridSearchCV

06 | Work already implemented

NULL values in the dataset:

- removed samples (lines) from dataset
- other option: fill values with mean / most frequent values of feature

```
imr = Imputer(missing_values='NaN', strategy='mean', axis=0)
imr = imr.fit(df.values)
imputed_data = imr.transform (df.values)
```

LINUX:

```
cat data.csv | grep "?" |  wc -l
cat data.csv | grep -v "?" > nomissing.data.csv
```

Map important non-numeric values to numeric

```
Mapping = {'of':1, 'def':2, 'gk':3}
df['size'] = df['size'].map(mapping)
```

Keep only samples and features with numeric values

```
df.dropna(axis=1)   int : ['home_team_goal', 'away_team_goal']
                    Categorical: ['month', 'home_team_name', 'away_team_name',
                    'country_name', 'league_name']
```

06 | Work already implemented

Reduce dimensions

```
selector = VarianceThreshold (threshold=0.5)
train_reduced = selector.fit_transform (C_trainData)
```

Normalize values

```
min_max_x = (x - np.min(x) )/ (np.max(x) - np.min(x))
```

Possible usage:

`imbalanced-learn` to over/under-sample dataset

`principal components analysis - PCA`

```
req_cols = ['overall_rating', 'crossing', 'finishing','heading_accuracy',
          'short_passing', 'volleys', 'dribbling', 'curve',
          'free_kick_accuracy','long_passing', 'ball_control', 'acceleration',
          'sprint_speed','agility', 'reactions', 'balance', 'shot_power', 'jumping',
          'stamina','strength', 'long_shots', 'aggression', 'interceptions',
          'positioning','vision', 'penalties', 'marking', 'standing_tackle',
          'sliding_tackle','gk_diving', 'gk_handling', 'gk_kicking',
          'gk_positioning','gk_reflexes']

data = player_data[req_cols]


data = player_data.drop(labels = ['id', 'player_fifa_api_id', 'player_api_id',
'date',
                                  'potential', 'preferred_foot',
                                  'attacking_work_rate',
                                  'defensive_work_rate'], axis = 1)
data.fillna(0, inplace=True)
#data.isnull().values.any()
data.corr()a
```

```
from sklearn.cross_validation import train_test_split
feature_cols = ['crossing', 'finishing', 'heading_accuracy', 'short_passing',
       'dribbling', 'curve', 'free_kick_accuracy',
       'long_passing', 'ball_control', 'acceleration', 'sprint_speed',
       'agility', 'reactions', 'balance', 'shot_power', 'jumping', 'stamina',
       'strength', 'long_shots', 'aggression', 'interceptions', 'positioning',
       'vision', 'penalties', 'marking', 'standing_tackle', 'sliding_tackle',
       'gk_diving', 'gk_handling', 'gk_kicking', 'gk_positioning',
'gk_reflexes']

x = data[feature_cols]
y = data.overall_rating

x_train, x_test,
y_train, y_test = train_test_split(x, y, test_size = 0.25, random_state = 42)
```

Thank you !