

**Софийски Университет „Св. Климент Охридски“**

**Факултет по математика и информатика**

**Проект по**

**„Изкуствен Интелект“**

Предсказване изхода от футболни срещи

Изготвил:

Борис Иванов Иванов, Ф.Н. 81326, Компютърни науки, курс 4, поток 1, група 3

Дата: 03 февруари 2019 година

# Съдържание

[Съдържание 2](#_Toc46693)

[1. Декларация за липса на плагиатство 3](#_Toc46694)

[2. Мотивация 3](#_Toc46695)

[3. Кратък обзор 3](#_Toc46696)

[4. Решение 4](#_Toc46697)

[5. Програмна реализация 5](#_Toc46698)

[5.1. Обучаващо множество (Training data set) 5](#_Toc46699)

[5.1.1. Структура 5](#_Toc46700)

[5.1.2. Събиране на данните 6](#_Toc46701)

[5.2. Представяне на данните в приложението 8](#_Toc46702)

[5.3. Класификатор 9](#_Toc46703)

[5.4. Предварителна обработка на данните 11](#_Toc46704)

[6. Постигнати резултати 12](#_Toc46705)

[7. Заключение 12](#_Toc46706)

# Декларация за липса на плагиатство

* + Тази курсова работа е моя работа, като всички изречения, илюстрации и програми от други хора са изрично цитирани.
  + Тази курсова работа или нейна версия не са представени в друг университет или друга учебна институция.
  + Разбирам, че ако се установи плагиатство в работата ми ще получа оценка “Слаб”.
  + Трите имена и подпис на студента:

# Мотивация

Футболът е една от най-известните игри в света. Всеки ден се играят десетки мачове. Футболът има и най-голяма популярност в спортните залагания. Почти всеки човек някога е залагал на изхода на даден мач.

Целта на този проект е създаването на софтуер, който предсказва как ще завърши даден футболен мач. Това от една страна е огромно предизвикателство и от друга, начин да се упражнят и практикуват знанията придобити по време на курса по Изкуствен интелект.

# Кратък обзор

Всички букмейкъри правят предвиждания за изхода на всеки мач и на основата на това изчисляват коефициентите си на печалба. Основният метод за определяне на изхода на даден мач е чрез статистически методи.

Все пак, до сега никой не е успял да измисли софтуер, който да предсказва всички или достатъчно мачове, за да може човек винаги да е на печалба от спортните залагания.

# Решение

Проектът е реализиран на програмния език Java.

Основният метод за разпознаване на изхода от мачове е методът за машинно самообучение **K най-близки съседа (K-Nearest-Neighbors**).

С цел оптимизация, обучаващото множество се **обработва** преди да обучи алгоритъма. Също така, при предсказване на даден мач първо се изчислява статистически какъв ще е изхода - смята се вероятността за всеки възможен изход (Победа, Равен, Загуба) и се взима най-голямата вероятност. Ако тази най-голяма вероятност е над 65 %, това е директно резултатът, който връща програмата. В противен случай се извиква класификаторът KNN.

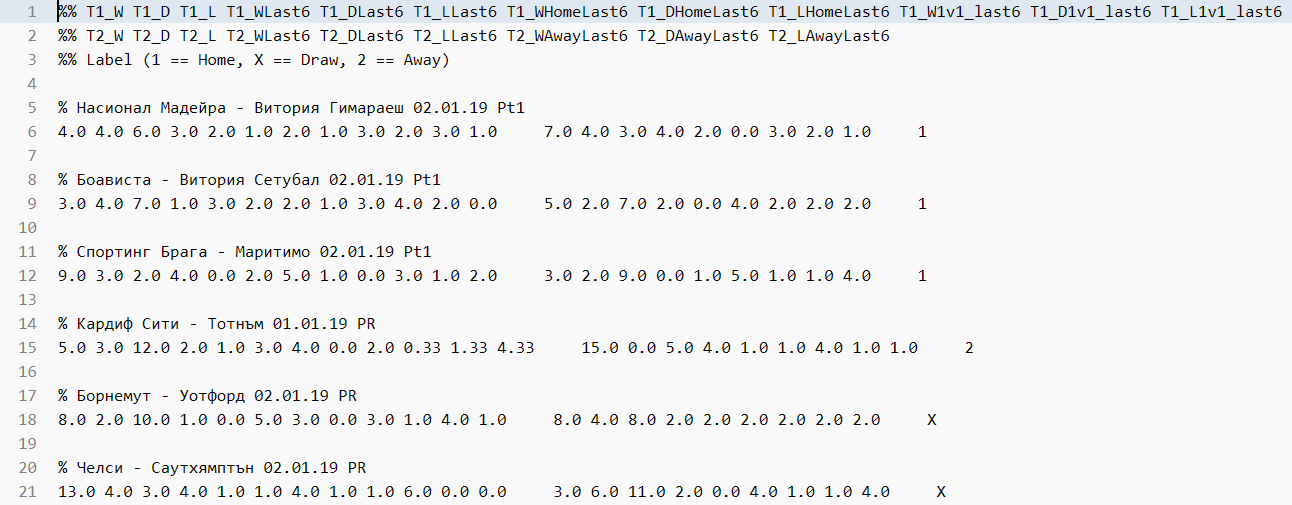
# Програмна реализация

## Обучаващо множество (Training data set)

### Структура

Всеки един ред в множеството (*Фиг. 1*) е текущото състояние на 2 отбора за един конкретен мач:

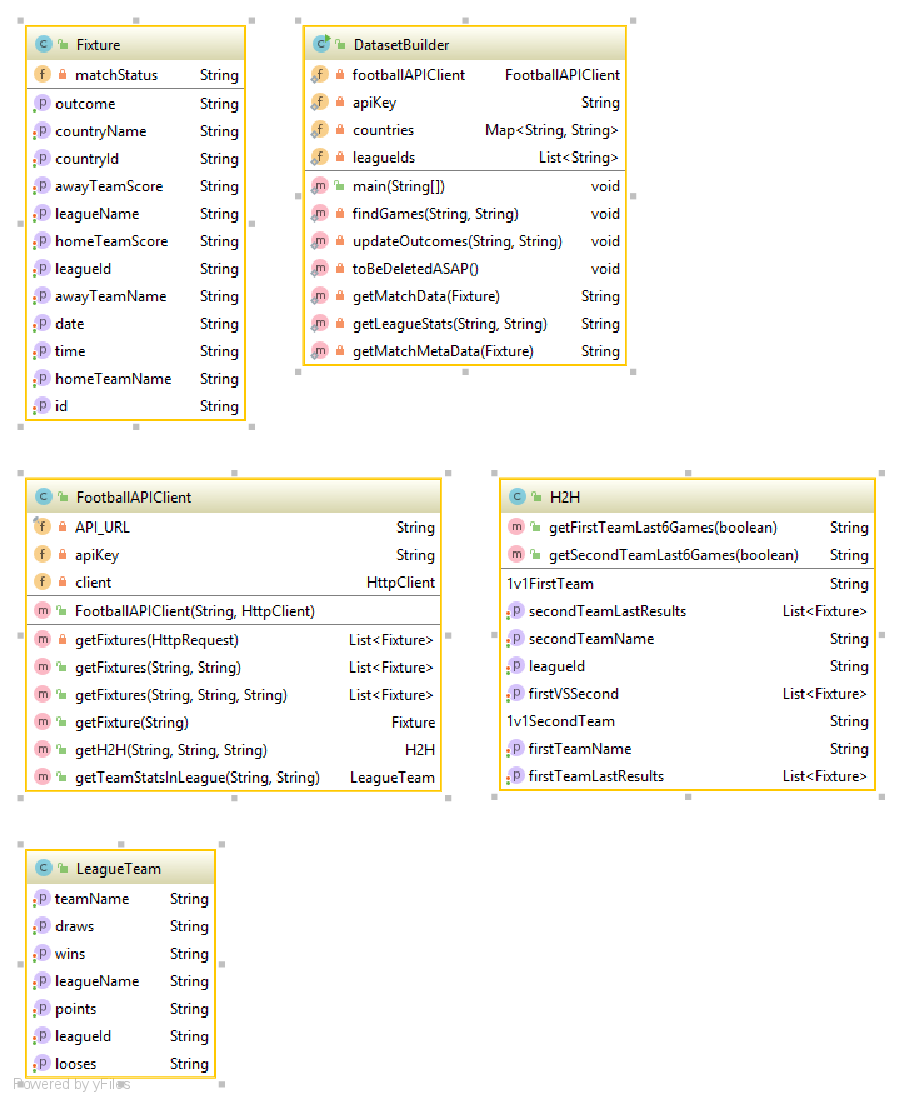
1. Победи за домакина в първенството
2. Равенства за домакина в първенството
3. Загуби за домакина в първенството
4. Победи за домакина в последните 6 игри
5. Равенства за домакина в последните 6 игри
6. Загуби за домакина в последните 6 игри
7. Победи за домакина в последните 6 игри на собствен терен
8. Равенства за домакина в последните 6 игри на собствен терен
9. Загуби за домакина в последните 6 игри на собствен терен
10. Победи за домакина срещу госта в последните 6 игри
11. Равенства за домакина срещу госта в последните 6 игри
12. Загуби за домакина срещу госта в последните 6 игри
13. Победи за госта в първенството
14. Равенства за госта в първенството
15. Загуби за госта в първенството
16. Победи за госта в последните 6 игри
17. Равенства за госта в последните 6 игри
18. Загуби за госта в последните 6 игри
19. Победи за госта в последните 6 игри на чужд терен
20. Равенства за госта в последните 6 игри на чужд терен
21. Загуби за госта в последните 6 игри на чужд терен
22. Резултат от мача: **1** или **2** или **X**



*Фиг. 1: Извадка от обучаващото множество*

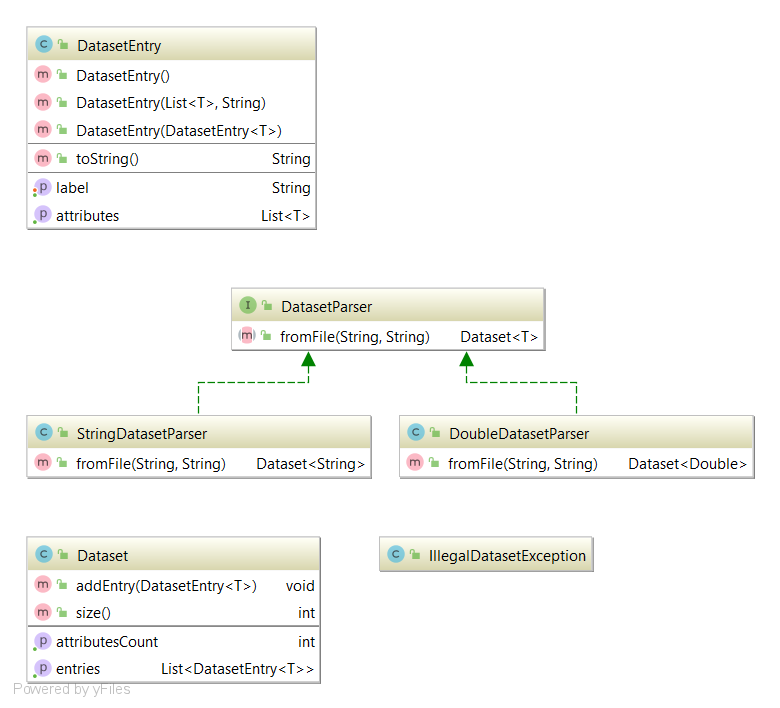
### Събиране на данните

Първоначално данните са взимани на ръка ден за ден. В последствие беше разработен специален модул за взимане на нови мачове чрез **RESTful API** (*Фиг. 2*).



*Фиг. 2: Автоматизирано построяване на обучаващото множество чрез RESTful API*

## Представяне на данните в приложението



*Фиг. 3: Модул за работа с Dataset*

Множеството е представено чрез абстракцията **Dataset**, която съдържа списък с всички записи от множеството. Един запис се представя чрез абстракцията **DatasetEntry**, която съдържа списък с *атрибути* от произволен тип и *Клас* на записа.

Интерфейсът **DatasetParser** служи за парсване на файла с данните. Предоставени са 2 имплементации на този интерфейс, като за целите на проекта се използва само **DoubleDatasetParser**, тъй като K-Nearest-Neighbors работи с числа.

## 5.3. Класификатор

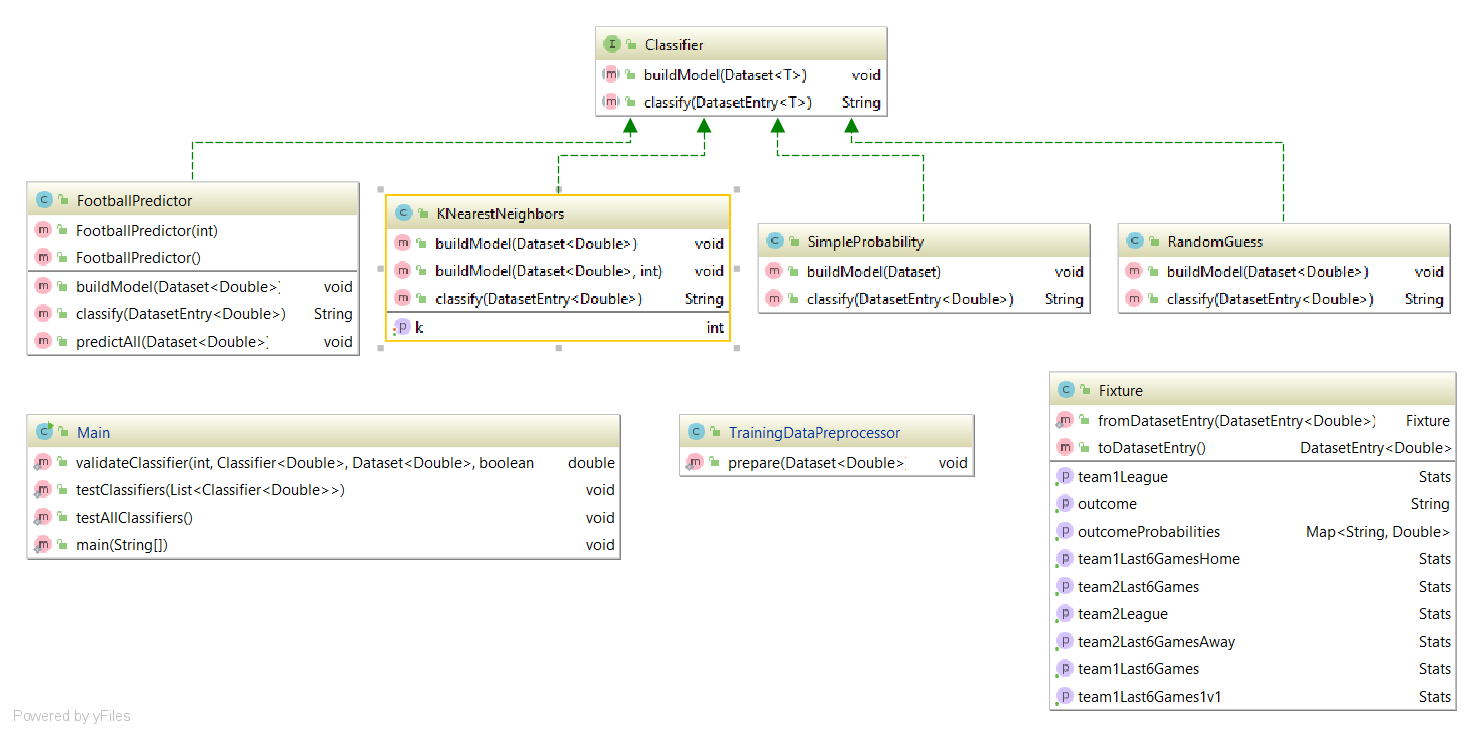
В основата на класифициращия алгоритъм лежи **K-Nearest-Neighbors** (K най-близки съседа).

**Подход**: Записите (от обучаващото и тестовото множество) се представят като *точки в n-мерното пространство*. Чрез Евклидово разстояние намираме първите K най-близки точки до тази, която класифицираме и връщаме *доминиращия* клас в тези К най-близки точки.

При класифицирането се използва и **допълнителна техника** с цел оптимизация: при предсказване на даден мач първо се изчислява статистически какъв ще е изхода - смята се вероятността за всеки възможен изход (Победа, Равен, Загуба) и се взима **най-голямата вероятност**. Ако тази най-голяма вероятност е над 65 %, това е директно резултатът, който връща програмата. В противен случай се извиква класификаторът KNN.

Тази вероятност се смята на база на атрибутите: последни 6 мача домакин/гост и последни 6 срещи един срещу друг.

Също така, данните от обучаващото множество се **обработват предварително**, преди да се подадат на алгоритъма.



*Фиг. 4: Класификатор*

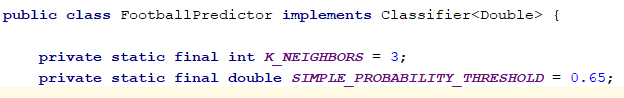
Имаме интерфейс **Classifier** с 2 публични метода: ***buildModel(Dataset<T>)*** и ***classify(DatasetEntry<T>)***.

Имплементират го следните 4 класа:

* **KNearestNeighbors** - имплементация на алгоритъма за K най-близки съседа
* **RandomGuess** - връща случаен отговор. Използва се за анализ на резултатите
* **SimpleProbability** - изчислява статистически изхода от събитие по горепосочения начин. Използва се за анализ на резултатите
* **FootballPredictor** - реалният класификатор, който използва предоставеното решение.

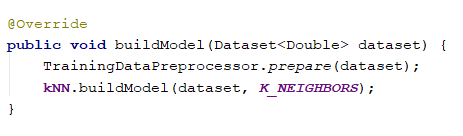
Процес на класифициране чрез FootballPredictor:

1. Софтуерна конфигурация на константи, ако е необходимо.



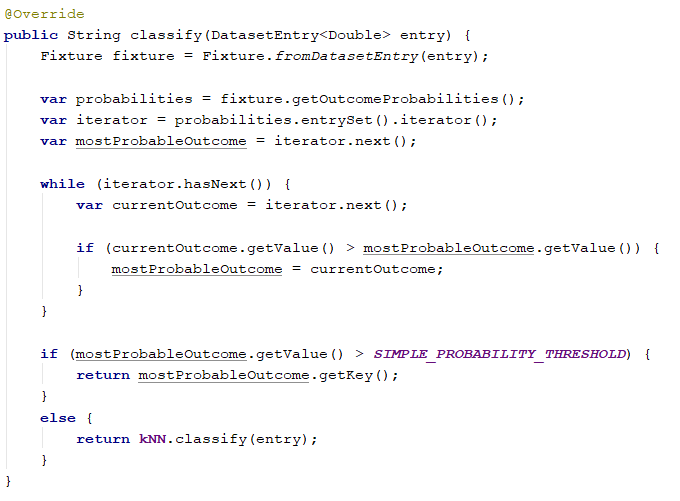
*Фиг. 5: FootballPredictor конфигурация*

1. Създайте обект от класа FootballPredictor
2. Извикайте buildModel() метод, който:
   * Подава обучаващото множество на **TrainingDataPreprocessor.prepare()** метода
   * Създава инстанция на **KNearestNeighbors** със съответното множество и настройки.



*Фиг. 6: FootballPredictor.buildModel(Dataset<Double> dataset)*

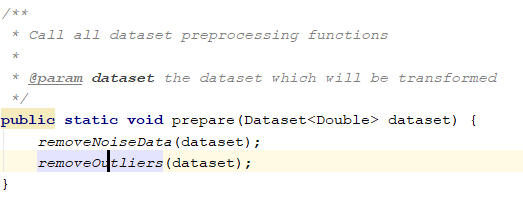
1. Извикайте метода classify(DatasetEntry<Double> entry), който класифицира конкретния запис (entry) по следния начин:
   * Първо се изчислява **статистически какъв ще е изхода** - смята се вероятността за всеки възможен изход (Победа, Равен, Загуба) и се взима най-голямата вероятност. Ако тази най-голяма вероятност е над SIMPLE\_PROBABILITY\_THRESHOLD , това е директно резултатът, който връща програмата.
   * В противен случай се извиква класификаторът **KNN**.



*Фиг. 7: FootballPredictor.* *classify(DatasetEntry<Double> entry)*

## 5.4. Предварителна обработка на данните

Текущата имплементация прави 2 предварителни обработки на множеството: премахва гранични случаи (outliers) и маха данни, които внасят шум в множеството. (*Фиг. 8*)



*Фиг. 8: TrainingDataPreprocessor.* *prepare(Dataset<Double> dataset)*

# Постигнати резултати

Решението беше тествано като обучаващото множество се разделя на 2 части - обучаващо и тестващо множество. За всеки запис от тестващото множество се извиква класификаторът FootballPredictor и се смята успеваемост на алгоритъма - колко от мачовете за предсказани коректно.

Резултат: 48% успеваемост

Извод: Всеки мач може да завърши по 3 възможни начина, което означава, че има вероятност да познаем отговора в 33 % от случаите. Имплементираният алгоритъм увеличава шансът за вярно предсказване почти наполовина.

Сравнение между успеваемостта на FootballPredictor, отгатване и смятане на статистическа вероятност за изход на мач:

* RandomGuess => Accuracy: 32.24137931034483 %
* SimpleProbability => Accuracy: 38.93103448275861 %
* FootballPredictor => Accuracy: 47.86206896551724 %

# Заключение

Чрез алгоритми за машинно самообучение можем да постигнем по-добра успеваемост за отгатване на изхода от футболни събития в сравнение с налучкване или изчисляване на вероятност кой ще победи, базирайки се на статистика от текущото състояние на отбора.