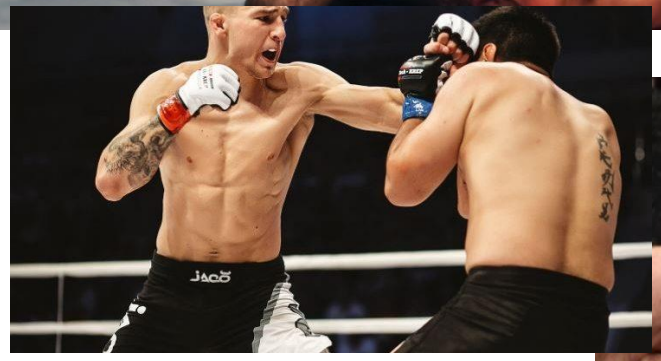
MMA classifier

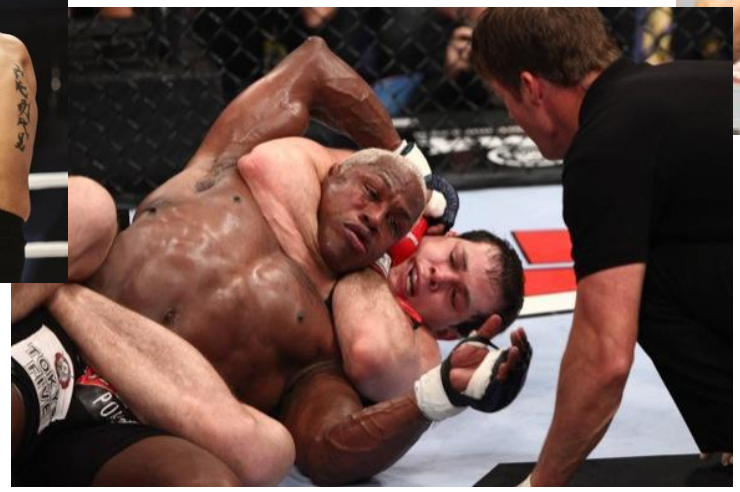
If it looks like a duck, swims like a duck, and quacks like a duck, then it probably *is* a duck.

Duck test

From Wikipedia, the free encyclopedia

Mixed Martial Arts = Judo + boxing + karate + kickboxing + greco-roman wrestling + jiu-jitsu + sambo…





CONOR MCGREGOR
"NOTORIOUS"

BORN: 1988-07-14
AGE: 29

DUBLIN
IRELAND

HEIGHT	WEIGHT	ASSOCIATION:
5'8"	**155 LBS**	**SBG IRELAND**
172.72 CM	70.31 KG	CLASS: **LIGHTWEIGHT**

WINS **21**	18 KO/TKO (86%)	
	1 SUBMISSIONS (5%)	
	2 DECISIONS (10%)	
LOSSES **3**	0 KO/TKO (0%)	
	3 SUBMISSIONS (100%)	
	0 DECISIONS (0%)	

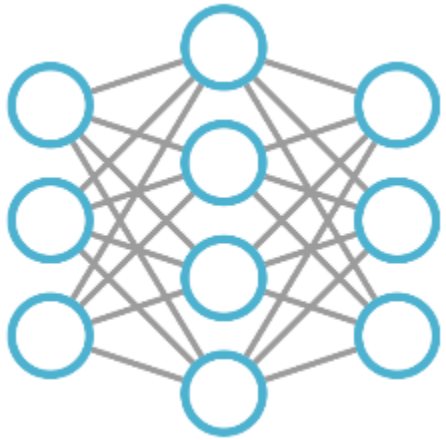
FIGHT HISTORY - PRO

RESULT	FIGHTER	EVENT	METHOD/REFEREE	R	TIME
WIN	Eddie Alvarez	UFC 205 - Alvarez vs. McGregor Nov / 12 / 2016	TKO (Punches) John McCarthy	2	3:04
WIN	Nate Diaz	UFC 202 - Diaz vs. McGregor 2 Aug / 20 / 2016	Decision (Majority) John McCarthy	5	5:00
LOSS	Nate Diaz	UFC 196 - McGregor vs. Diaz Mar / 05 / 2016	Submission (Rear-Naked Choke) Herb Dean	2	4:12
WIN	Jose Aldo	UFC 194 - Aldo vs. McGregor Dec / 12 / 2015	KO (Punch) John McCarthy	1	0:13
WIN	Chad Mendes	UFC 189 - Mendes vs. McGregor Jul / 11 / 2015	TKO (Punches) Herb Dean	2	4:57
		UFC Fight Night 59 -			

Het idee:

UNIBET

betfair



Wat weet ik?

- Basale kennis Machine Learning: Masterclass van Arno Knobbe in 2012, Leiden Institute of Advanced Computer Science, UL
- Geen Python ervaring, wel 20 jaar programmeerervaring in o.a. Java, C, C#...

Doel

Rijk worden 😊

maar ook leren…

Python

API, JSON

Toepassing ML

Python libraries

Web scraper

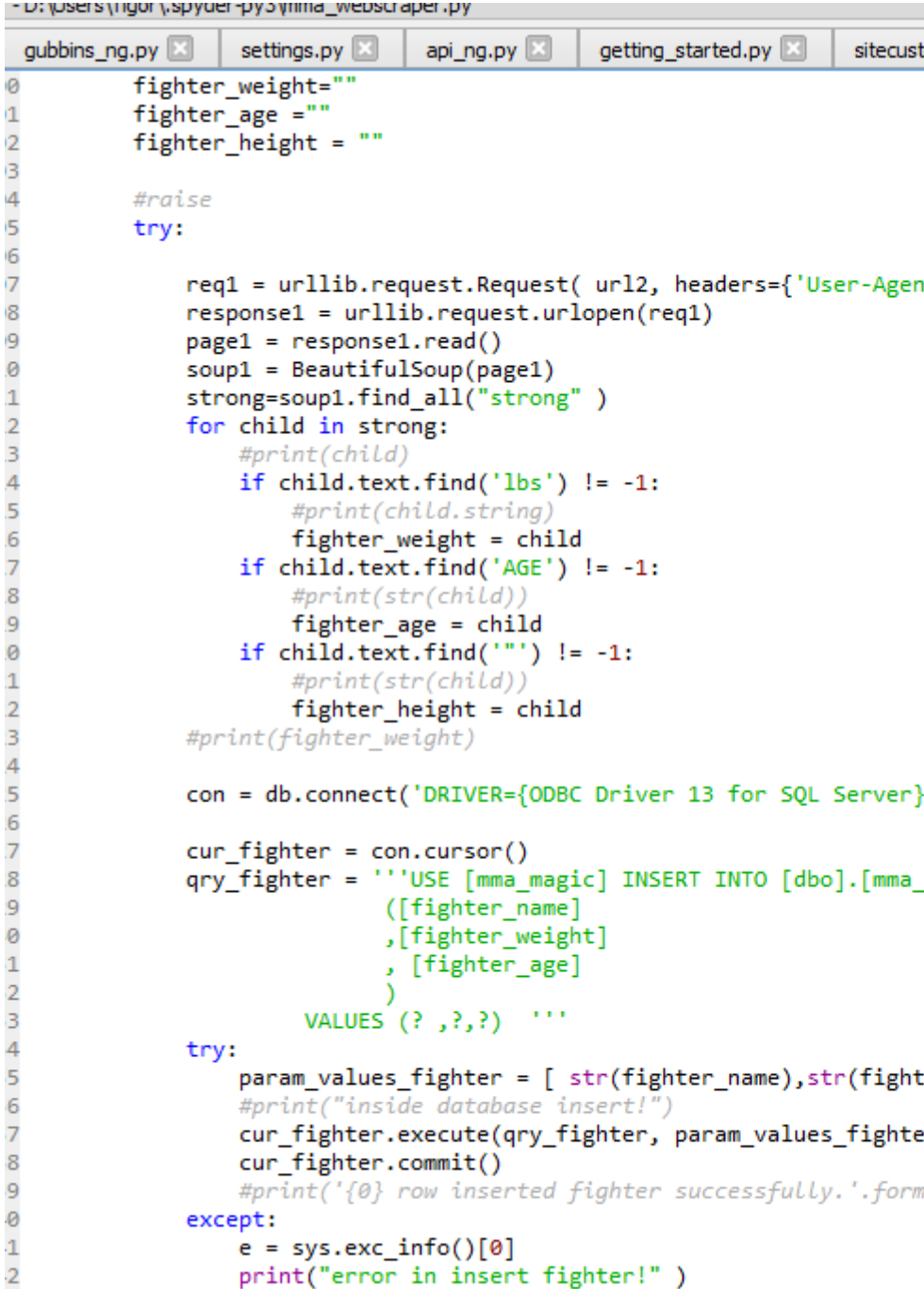
BeautifulSoup

ML classifier

Scikit-learn

Automatic betting

Urllib (Betfair Eschange API via Jason-RPC)



Webscraping

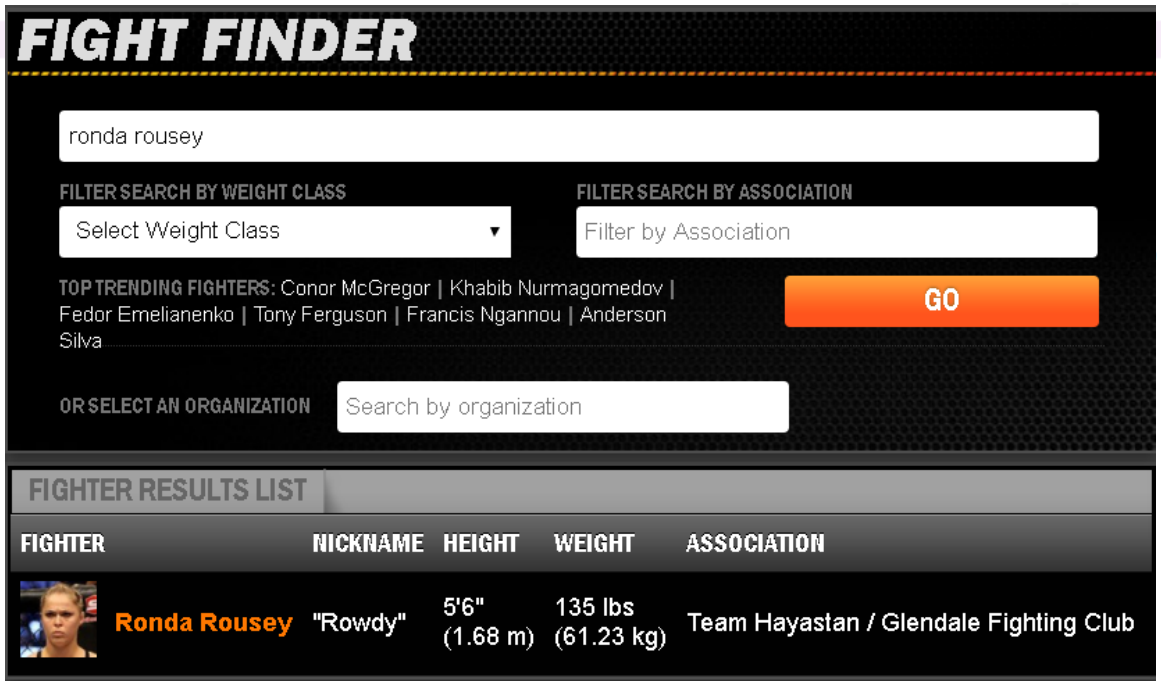
http://www.sherdog.com/robots.txt

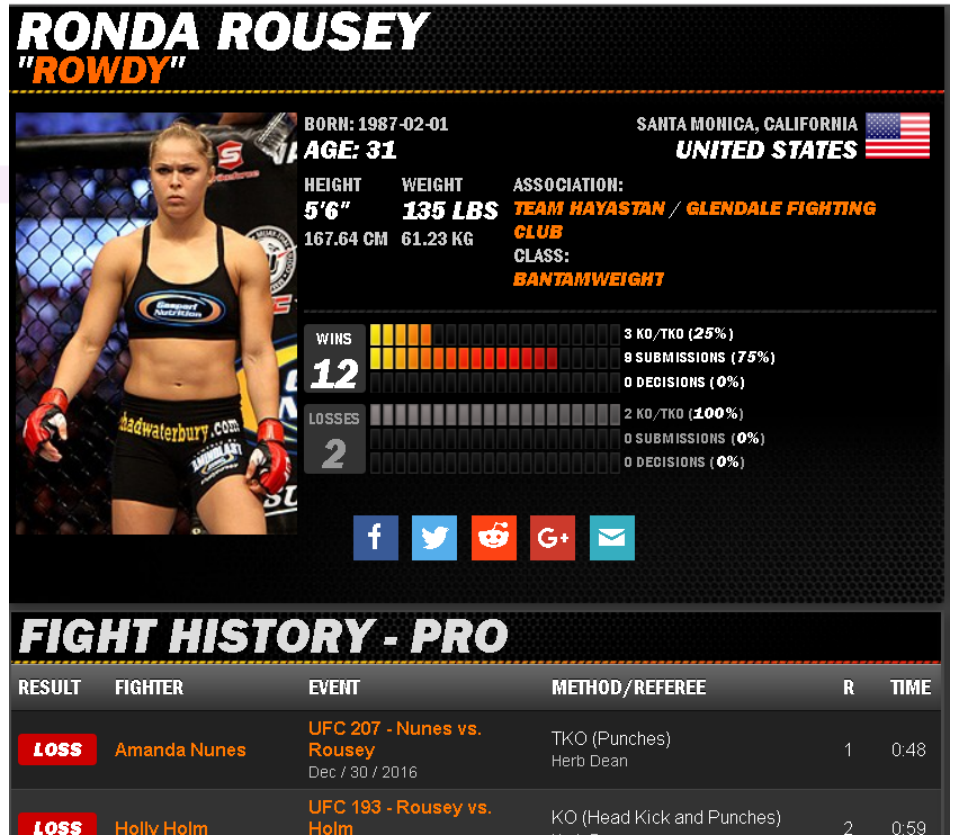
```
def get_mma_stats(url, fighter_name):
    print(url)
    user_agent = 'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.0.7) Gecko/2009021910 Firefox/3.0.7'
    #url = 'http://www.sherdog.com/stats/fightfinder?SearchTxt=&weight=6&association='

    try:

        req1 = urllib.request.Request(url, headers={'User-Agent':user_agent,})
        response1 = urllib.request.urlopen(req1)
        page1 = response1.read()
        soup1 = BeautifulSoup(page1)

        all_links = soup1.find_all('a')
```





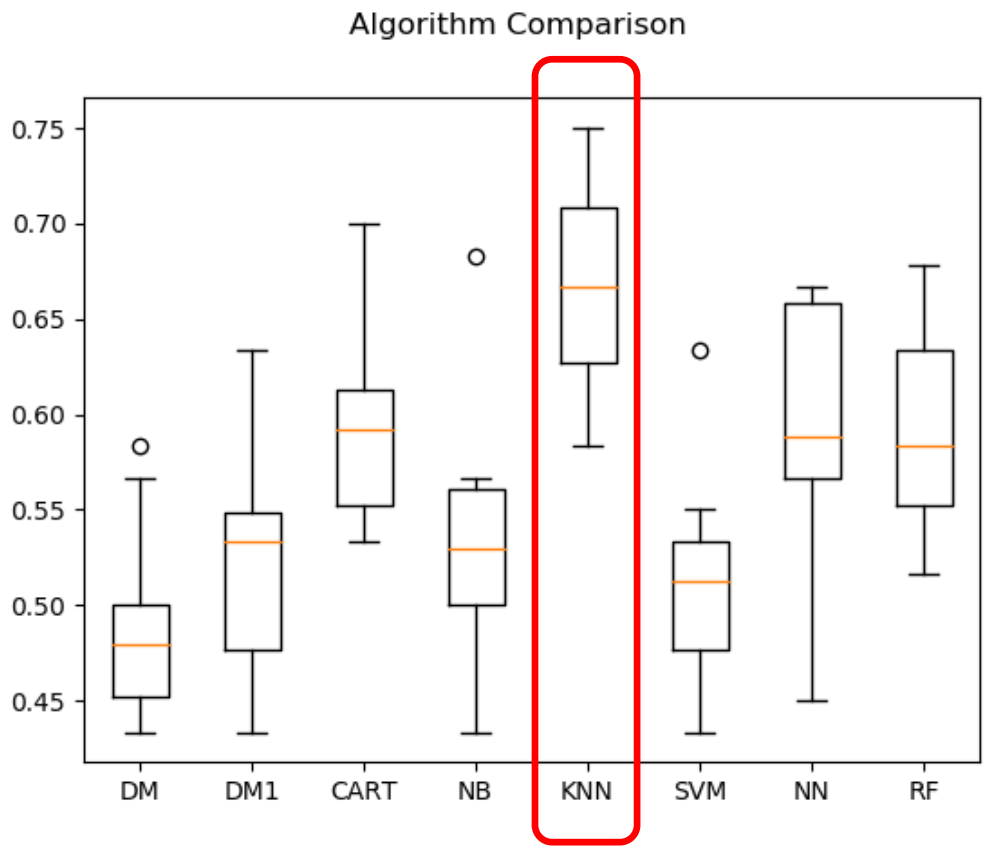
ML: **Data, data, data**

- Feature selection
- Imputation
- Normalization

```
In [6]: print(rescaledX[0:5,:])
[[0.         0.         0.22222222 0.1        0.13333333 0.22222222
  0.         0.3        0.22222222 0.55       0.36666667 0.16666667]
 [0.41666667 0.2        0.         0.4        0.36666667 0.
  0.66666667 0.2        0.66666667 0.4        1.         0.11111111]
 [0.25       0.3        0.22222222 0.2        0.1        0.44444444
  0.16666667 0.3        0.22222222 0.4        0.2        0.27777778]
 [0.16666667 0.3        0.22222222 0.4        0.2        0.27777778
  0.25       0.3        0.22222222 0.2        0.1        0.44444444]
 [0.66666667 0.2        0.22222222 0.5        0.13333333 0.77777778
  0.41666667 0.1        0.22222222 0.55       0.16666667 0.5       ]]
```

LOSSES_DECISIONS	LOSSES_KOTKO	LOSSES_SUBMISSIONS	WINS_DECISIONS	WINS_KOTKO	WINS_SUBMISSIONS	LOSSES_DECISIONS_O	LOSSES_KOTKO_O	LOSSES_SUBMISSIONS_O	WINS_DECISIONS_O	WINS_KOTKO_O	WINS_SUBMISSIONS_O	win_loss
0	0	2	2	4	4	0	3	2	11	11	3	0
5	2	0	8	11	0	8	2	6	8	30	2	0
3	3	2	4	3	8	2	3	2	8	6	5	1
2	3	2	8	6	5	3	3	2	4	3	8	1
8	2	2	10	4	14	5	1	2	11	5	9	0
0	0	1	6	1	5	4	2	2	7	5	12	1
4	2	2	7	5	12	0	0	1	6	1	5	0
2	2	0	2	5	3	0	3	1	2	4	3	1
2	1	3	7	25	12	0	3	0	4	6	4	1
2	1	0	7	8	4	1	0	0	1	4	6	1

Accuracy per classifier, 10 splits
Cross validation



Dummy (uniform)

Dummy (most frequent)

DecisionTree

Naive Bayes

K-Nearest Neighbor

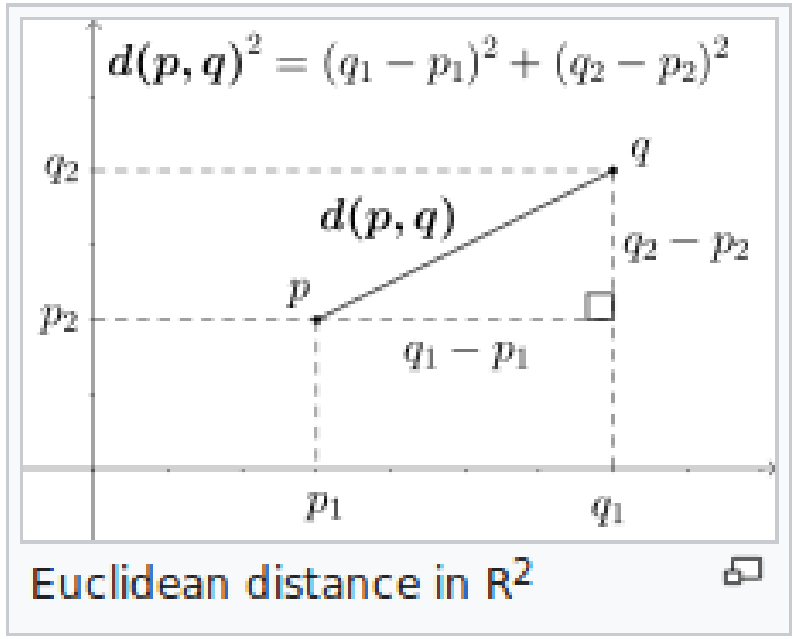
Neural Network

Support Vector Machine

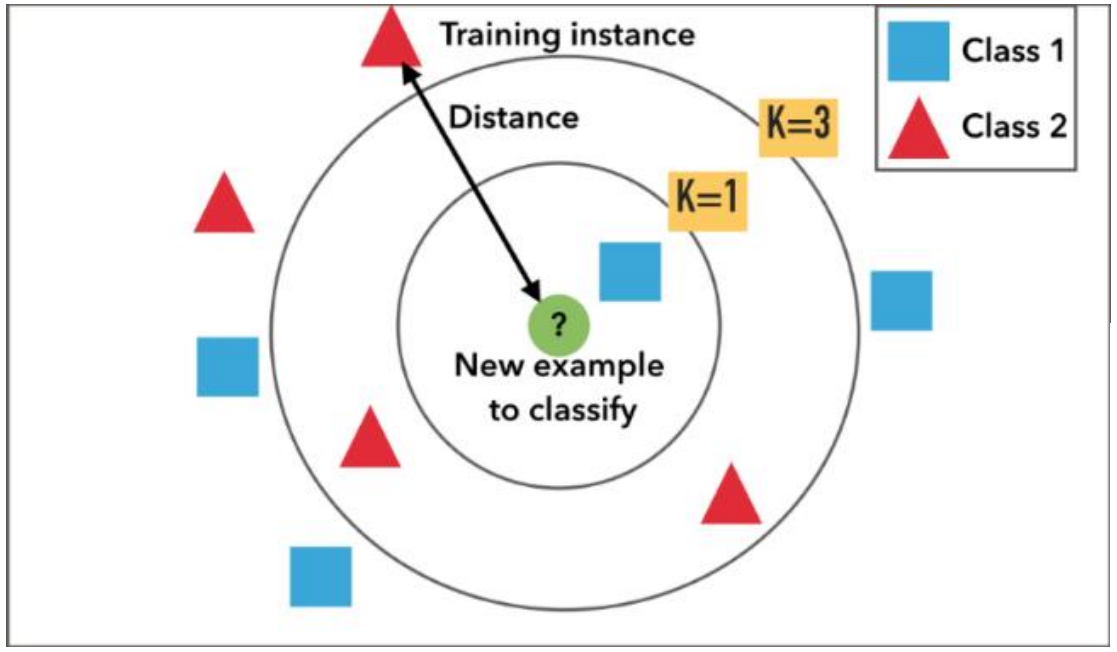
Random Forest

K Nearest Neighbors

“Simple and lazy learner”
generaliseert (leert) niet, geen echte trainingsfase
heeft de hele training set nodig tijdens testing

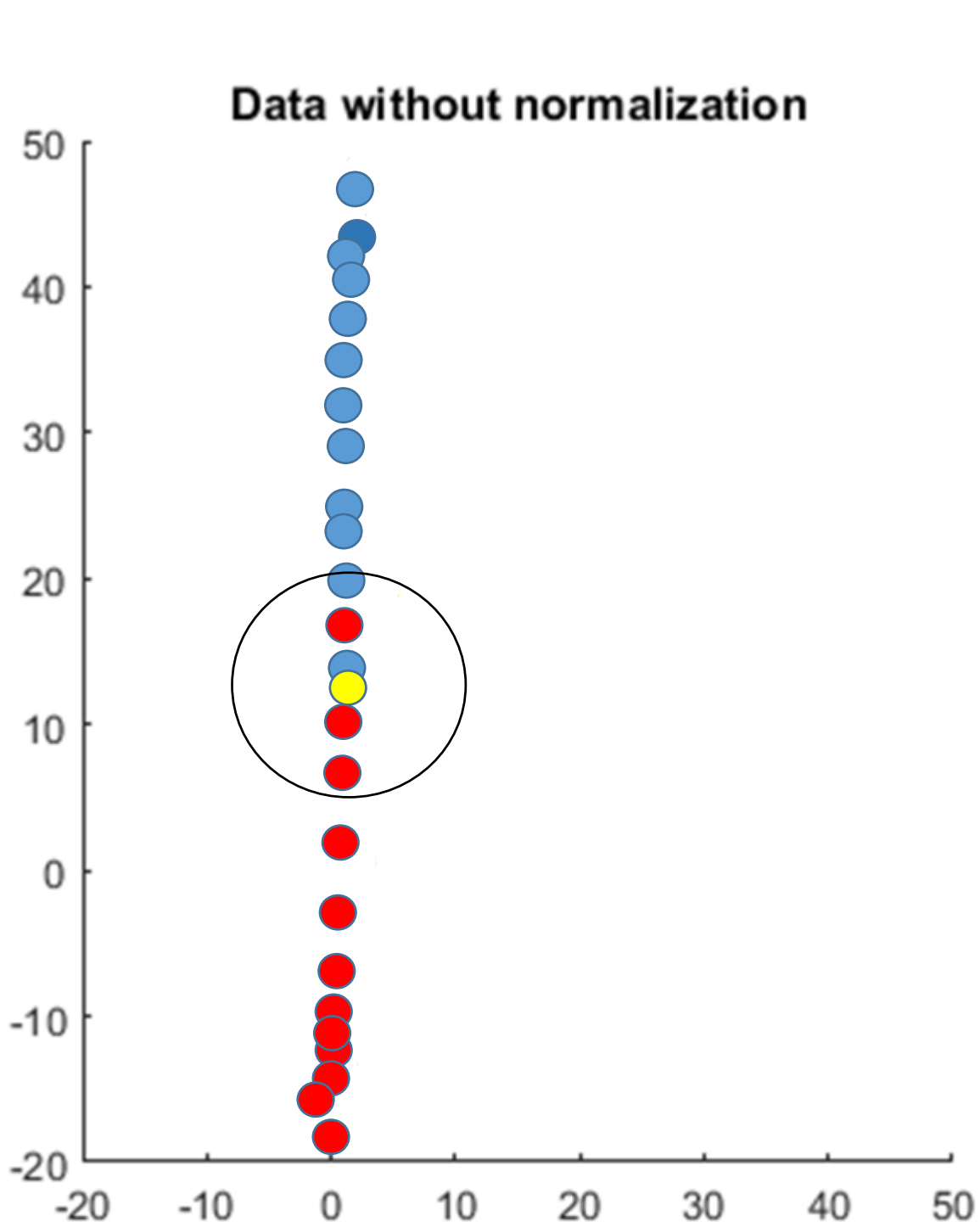


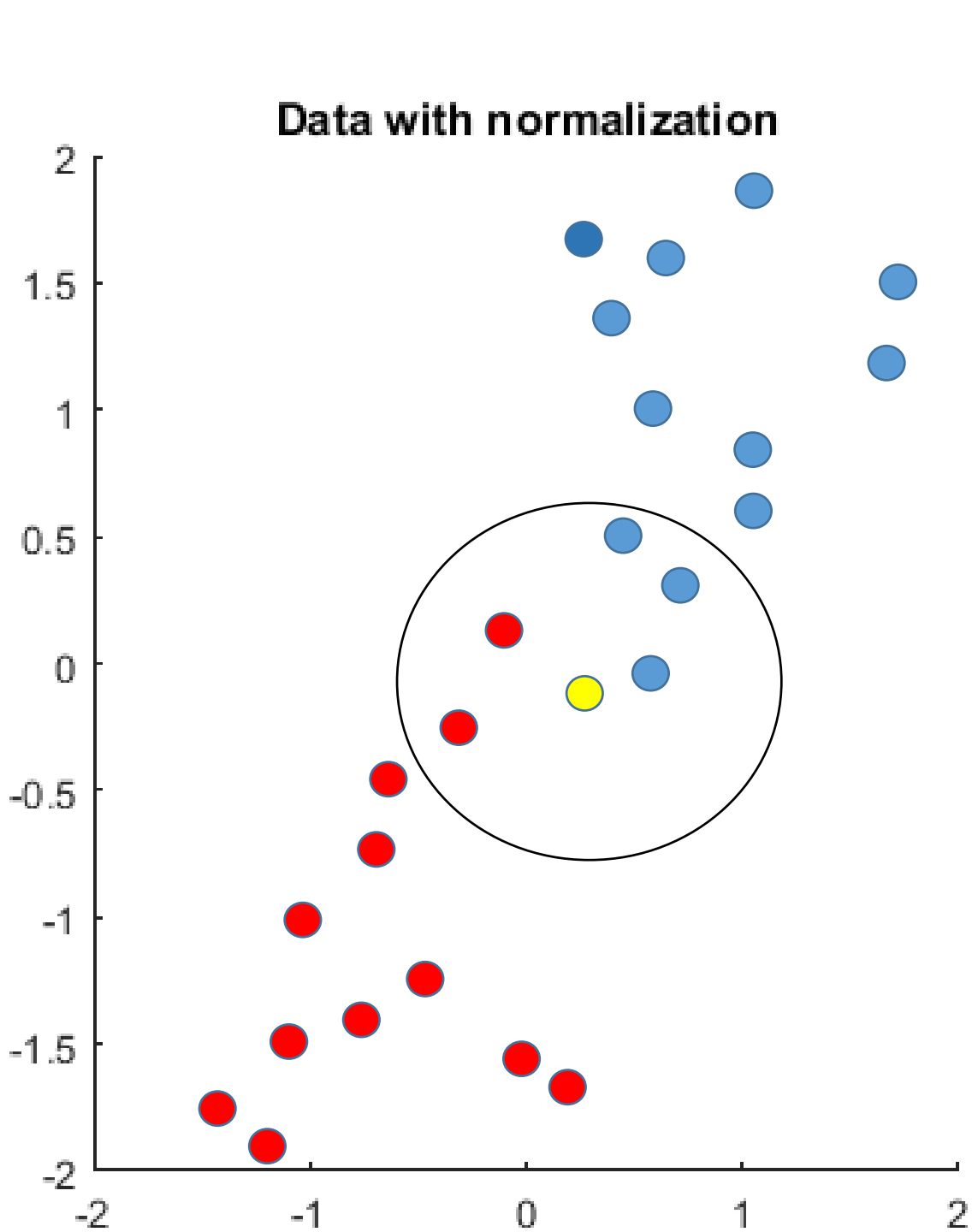
Euclidean distance in R^2



$$d(\mathbf{p},\mathbf{q}) = d(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

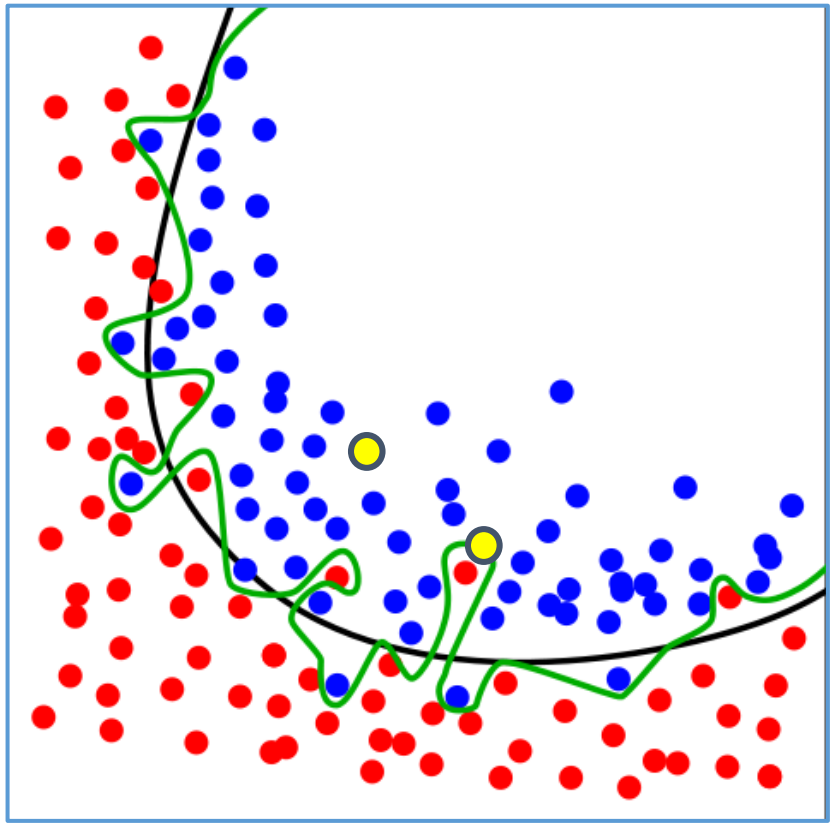
$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$


Data without normalization
50
40
30
20
10
0
-10
-20
-20
-10
0
10
20
30
40
50

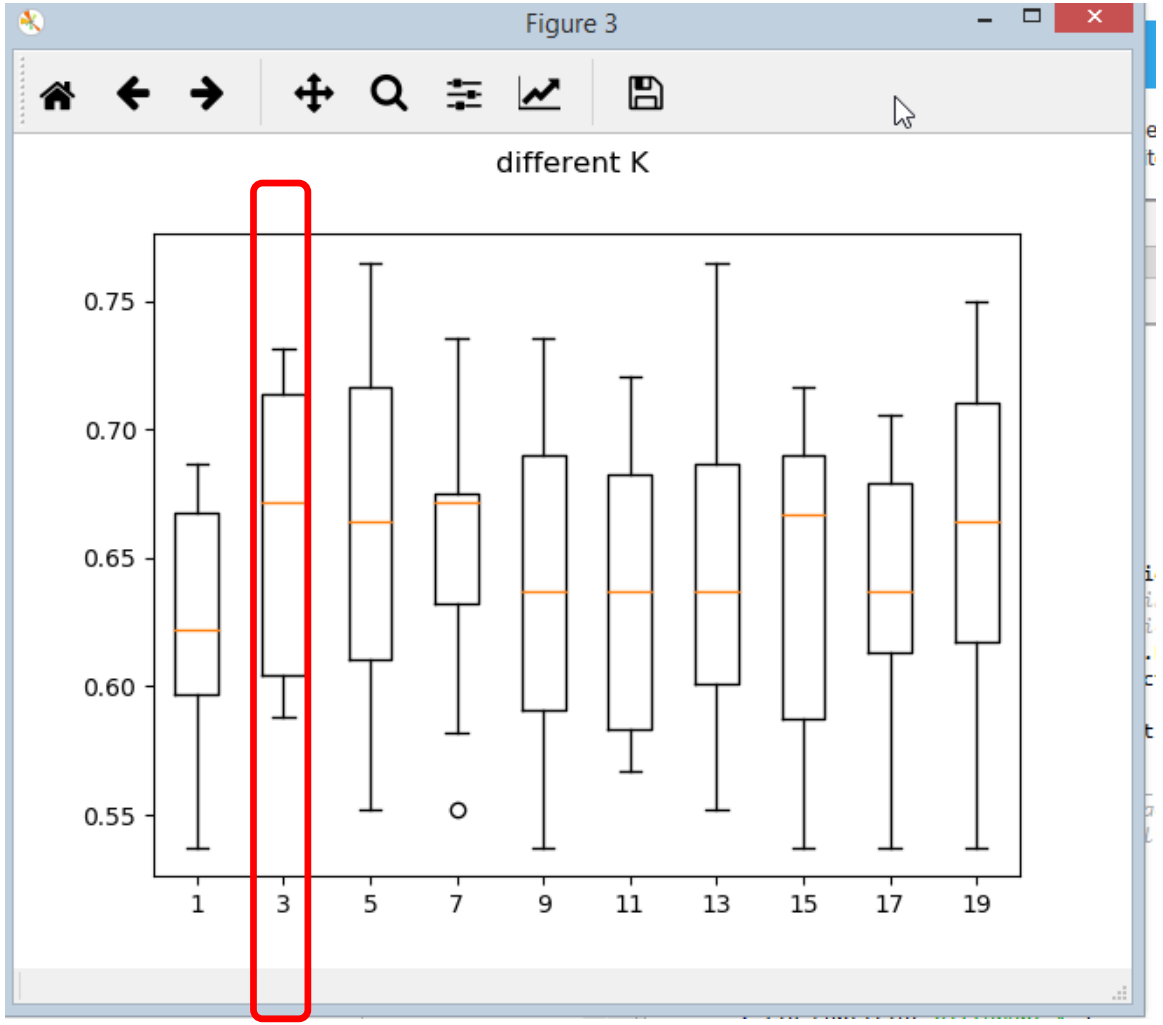

Data with normalization
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1
0
1
2

K - te klein: overfitting
K - te groot ?

weights = 'distance'



Cross validation voor KNN
K: 1..19



Confusion matrix

Accuracy: **0.60**

	Predicted	
Actual	**0**	**1**
0	TN: 59	FP: 33
1	FN: 41	TP: 54

`precision	recall	f1-score	support
0 0.59	0.64	0.61	92
1 **0.62**	0.57	0.59	95
avg / total 0.61	0.60	0.60	187

Metric	Formula
True positive rate, recall	$\frac{TP}{TP+FN}$
False positive rate	$\frac{FP}{FP+TN}$
Precision	$\frac{TP}{TP+FP}$
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
F-measure	$\frac{2 \cdot precision \cdot recall}{precision + recall}$

Accuracy

It is the number of correct predictions made divided by the total number of predictions made

Precision

the number of positive predictions divided by the total number of positive class values predicted. It is also called the Positive Predictive Value (PPV).

Precision can be thought of as a measure of a classifiers exactness. A low precision can also indicate a large number of False Positives.

Recall

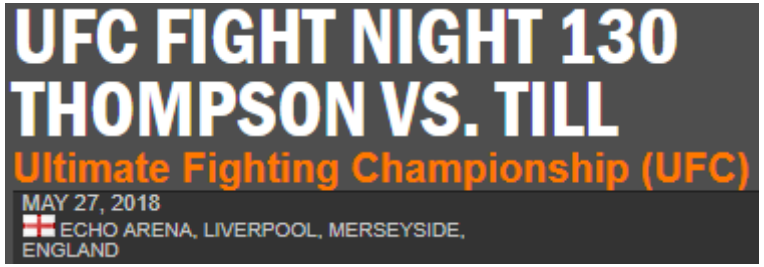
Recall is the number of True Positives divided by the number of True Positives and the number of False Negatives. Put another way it is the number of positive predictions divided by the number of positive class values in the test data. It is also called Sensitivity or the True Positive Rate.

Recall can be thought of as a measure of a classifiers completeness. A low recall indicates many False Negatives.

F1 Score

The F1 Score is the 2*((precision*recall)/(precision+recall)). It is also called the F Score or the F Measure. Put another way, the F1 score conveys the balance between the precision and the recall.

Real data…



fighter_name	LOSSES_DECISIONS	LOSSES_KOTKO	LOSSES_SUBMISSION	WINS_DECISIONS	WINS_KOTKO	WINS_SUBMISSIONS
Jason Knight	**3**	**1**	**0**	**4**	**3**	**13**
fighter_name_o	LOSSES_DECISIONS_o	LOSSES_KOTKO_o	LOSSES_SUBMISSION	WINS_DECISIONS_o	WINS_KOTKO_o	WINS_SUBMISSIONS_o
Makwan Amirkhani	**2**	**0**	**1**	**3**	**1**	**9**

- knn = KNeighborsClassifier(n_neighbors=3, weights = 'distance')
- knn.fit(rescaledX, Y)
- predictions = knn.predict(X_real)
- print(predictions)
- **[0]**

MMA Classifier: Amirkhani wins !

Beveiligd | https://www.unibet.eu/betting#bethistory/2018-06-08

POPULAR

ATP

International Friendly Matches

MLB NBA NHL

WTA French Open

French Open Women

Formula 1 World Cup 2018

World Cup Boosts

ALL SPORTS NUMBER OF BETS

American Football	261
Australian Rules	793
Baseball	558
Basketball	1223
Beach Volley	16
Boxing	31
Chess	2

Home Live My Bets

8 Jun 2018

27 May 2018 07:36:17 — Won

Stake: €9.00 Odds: 2.25 Knight, Jason - Amirkhani, Makwan

Payout: €20.25

Bout Odds: Amirkhani, Makwan

26 May 2018 14:27:40 — Won

Stake: €10.00 Odds: 1.42 Al

Payout: €14.20

Bout Odds: Allen, Arnold

26 May 2018 14:10:54 — Won

Stake: €1.00 Odds: 1.16 Magny, Neil - White, Craig

Payout: €1.16

Bout Odds: Magny, Neil

26 May 2018 13:56:09 — Won

Stake: €10.00 Odds: 1.95 Thompson, Stephen - Till, Darren

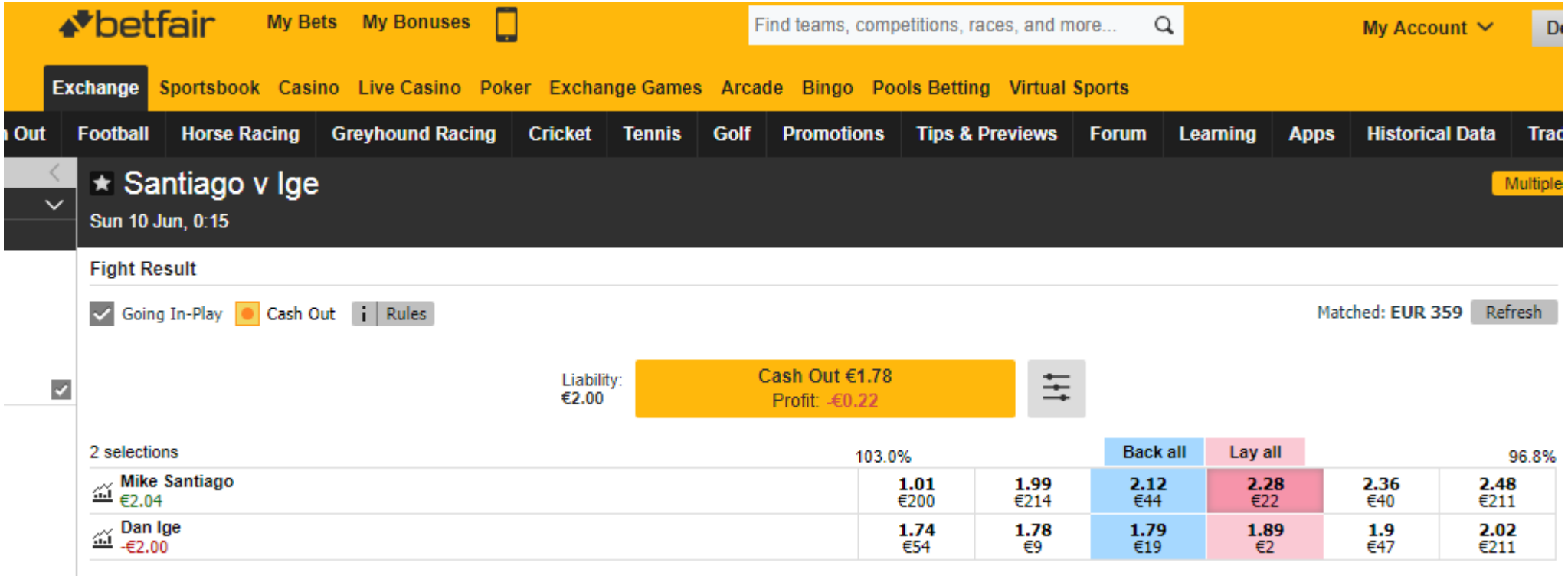
Payout: €19.50

Bout Odds: Till, Darren

Jason

```
{
   "params": {
      "filter": {
         "eventTypeIds": [1]
      }
   },
   "jsonrpc": "2.0",
   "method": "SportsAPING/v1.0/listCompetitions",
   "id": 1
}
```

- Application key
- Session token
- Certificates



Conclusie

Python rules!

Long live Internet 😊

OSTAGRAM.RU