World Cup Prediction Using Machine Learning and Python

Team Member: Cao Qinyu, Cui Xiaomei, Gong Chen, Li Chen, Ning Yu, Wang Ruikun, Sun Dianyong, Tong Pai

Group Project for PKU Python Programming Summer School

Instructor: He Jibo

Introduction

Project Background

- **Like the previous FIFA World Cup 2014, the tournament in Russia has also caught the attention of several modelers who try to predict the tournament winner.** (Leitner, Zeileis, and Hornik, 2010b, Zeileis, Leitner, and Hornik, 2012, 2014, 2016)

- Existing approaches:
- *1. Poisson Regression Models*
- *2. Ranking Methods*
- *3. Tree-based Methods*

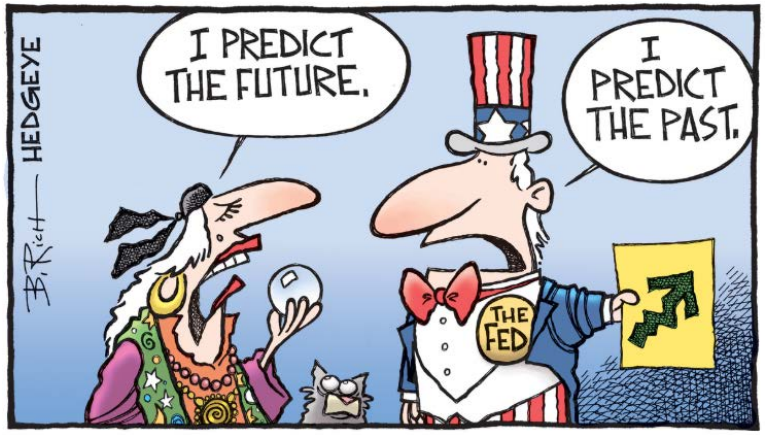
Introduction

Project Goal

★ **Predict 2022 Qatar FIFA World Cup result.** (Which team will likely to win in the Round 16, Quarter Finals, Semi-Finals, and Finals.)

Possible Application

1. **Bet based on calculated probability**
2. **Show prediction ability to client**
3. **Practice Python programming**



Contents

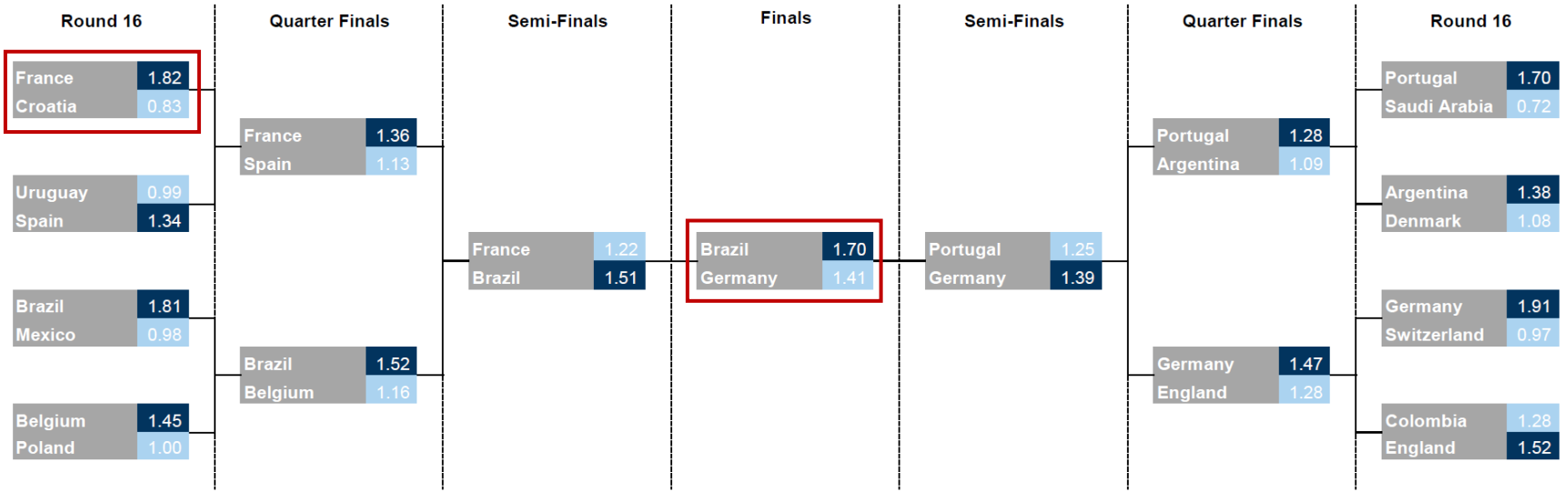
Source: Team Analysis

Contents

Existing Work

Investment Banks Really Like to Predict World Cups (2018)…

Goldman Sachs



Key Conclusion:

1. **Brazil win the world cup**
2. **Russia cannot stand out from group stage**
3. **Germany will be in the final**

Wrong!

Source: Goldman Sachs Global Investment Research

Existing Work

Investment Banks Really Like to Predict World Cups (2018)…

Union Bank of Switzerland

	Winner	Runner-Up	Semi-Finalist	Quarter-Finalist	Winner Group Stage	Second Group Stage
Germany	**24.0**	**36.7**	**51.3**	**66.7**	**68.6**	**22.0**
Brazil	**19.8**	**31.9**	**44.1**	**60.5**	**66.8**	**23.1**
Spain	**16.1**	**28.0**	**50.5**	**68.5**	**60.6**	**26.5**
England	8.5	18.7	31.4	66.2	53.7	33.6
France	7.3	16.1	35.1	59.5	60.1	24.6
Belgium	5.3	11.6	23.8	56.9	38.3	43.7
Argentina	4.9	11.3	26.9	51.8	54.7	26.4
Portugal	3.1	8.0	21.8	39.8	25.2	38.2
Uruguay	1.8	5.5	15.8	32.0	42.5	34.3
Switzerland	1.8	5.0	11.5	22.9	19.7	39.6
Mexico	1.8	5.3	10.9	22.5	17.2	36.6
Italy	1.6	4.4	10.1	19.4	15.3	31.0
Russia	1.6	4.6	14.4	30.5	41.4	33.6
Poland	0.9	2.9	7.1	24.7	35.4	28.7
Colombia	0.5	1.8	5.0	20.0	28.2	27.9
Sweden	0.4	1.4	3.8	9.9	8.8	23.7
Iran	0.4	1.7	5.6	14.2	9.4	21.4
Nigeria	0.3	1.3	4.8	15.9	16.3	25.5
Peru	0.3	1.2	5.3	16.8	14.4	27.2
Serbia	0.2	1.0	2.8	7.7	8.7	22.8
Senegal	0.2	0.9	2.7	12.6	19.9	22.8
Iceland	0.2	0.7	3.7	13.6	13.8	23.5
Croatia	0.2	0.9	4.4	15.0	15.2	24.7
South Korea	0.2	0.6	1.9	6.0	5.4	17.7
Denmark	0.1	0.9	4.3	15.5	14.2	26.0
Australia	0.1	0.5	3.3	12.0	11.3	22.2
Morocco	0.1	0.3	2.2	6.8	4.9	13.9
Japan	0.1	0.4	1.6	9.8	16.6	20.6
Egypt	0.0	0.2	1.5	5.1	9.5	17.3
Tunisia	0.0	0.3	1.1	8.0	6.0	15.9
Costa Rica	0.0	0.2	0.9	3.9	4.7	14.5
Saudi Arabia	0.0	0.1	0.6	3.2	6.7	14.8
Panama	0.0	0.0	0.2	2.0	1.9	6.8

Key Conclusion:

1. **Germany win the world cup**
2. **Brazil, Spain likely to be runner-up**
3. **Croatia is not likely to stand out from group stage**

Wrong!

Source: UBS Global Investment Research

Existing Work

… But They Basically Get It Wrong

Correct prediction | Still playing | Wrong prediction

Predictor	2014	2018
Goldman Sachs	Brazil	Brazil
UBS	Brazil	Germany
ING	Spain	Spain
Nomura	N/A	France
Macquarie Bank	Germany	Spain
Actual Winner	Germany	France or Croatia

NOTE: Don't believe in investment bank's analysis; **Believe in data and Python!**

Source: Team Analysis

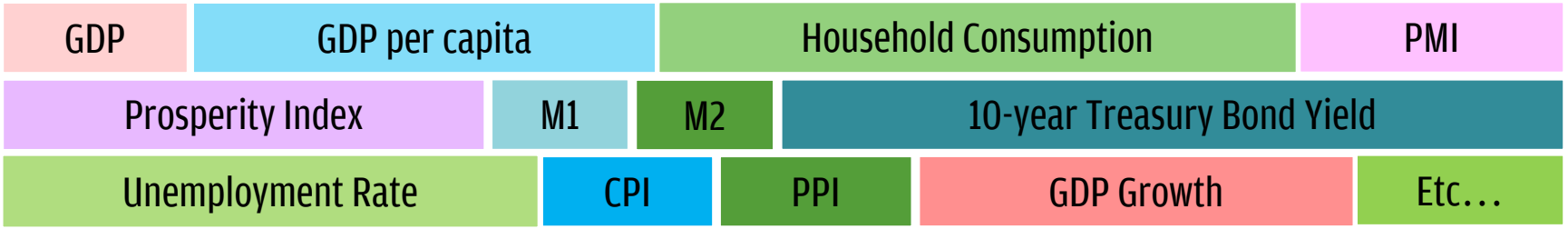
Contents

Data Collection

Collected Data from Various Dimension

Dataset Description

Data	Explanation
Major football tournament data from 1930 to 2018	Including World Cup, World Cup qualification, regional championship et al.
World Cup ranking data from 1930 to 2018	Including each team's rank in the history of World Cup
Political, Economic, climate data	Auxiliary data.



Source: Kaggle, World Bank

Initial Data Visualization

Graph Representation and Interpretation

Graph Representation

Task 1:

Find out how many games each team won and show the result in a bar chart.

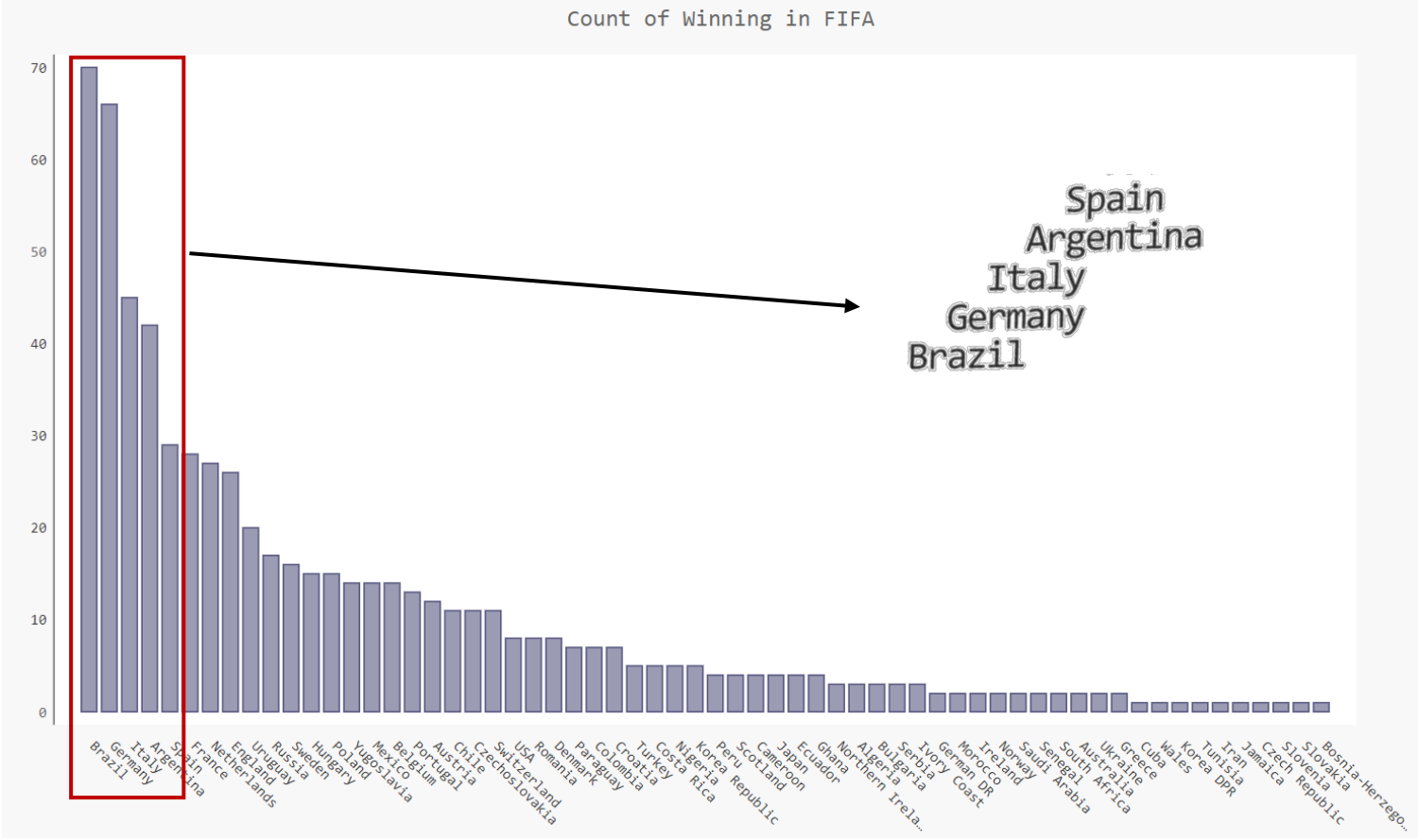
The top five teams are Germany, Argentina, Brazil, France and Spain.

Source: Team Analysis

Initial Data Visualization

Graph Representation and Interpretation

Graph Representation

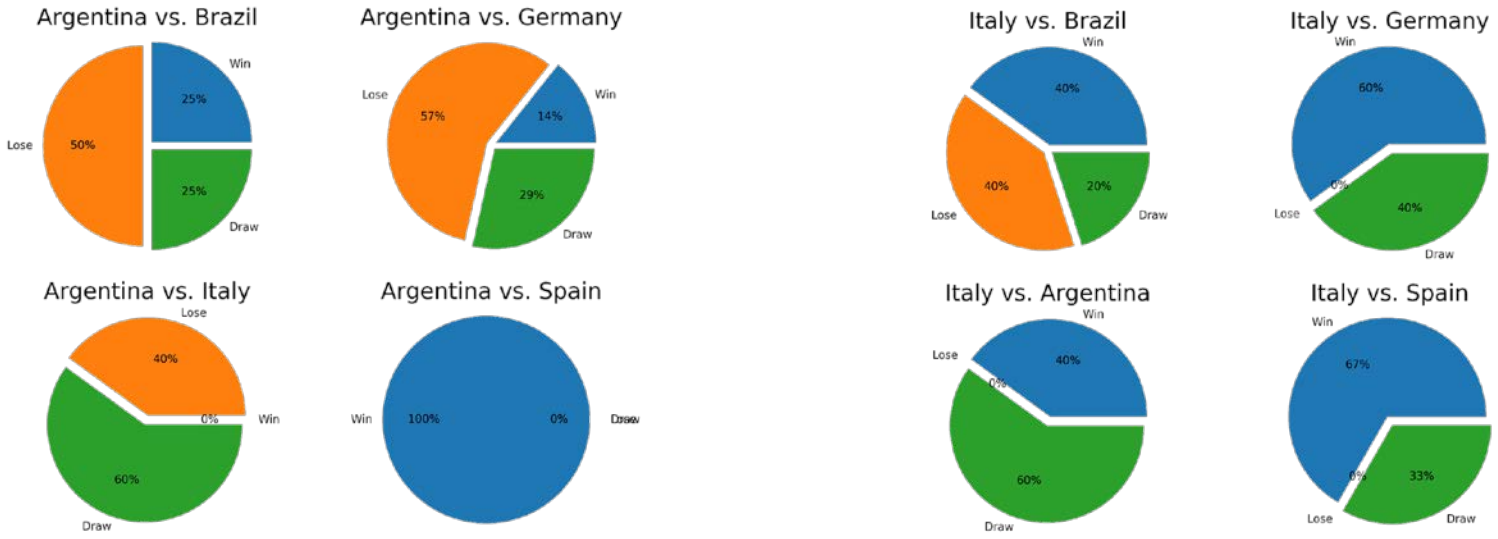


Source: Team Analysis

Initial Data Visualization

Graph Representation and Interpretation

Graph Representation



Task 2:

Find out the matches between the top five teams in World Cup.

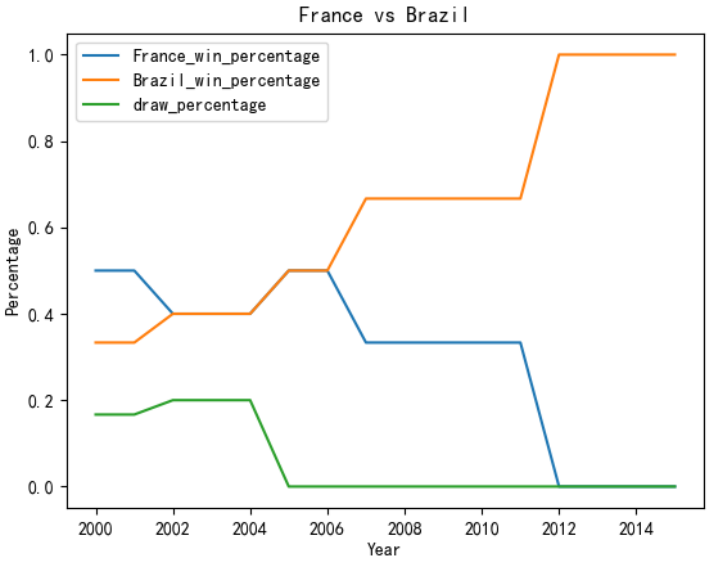
Interesting note: Argentina always win over Spain in the World Cup history.

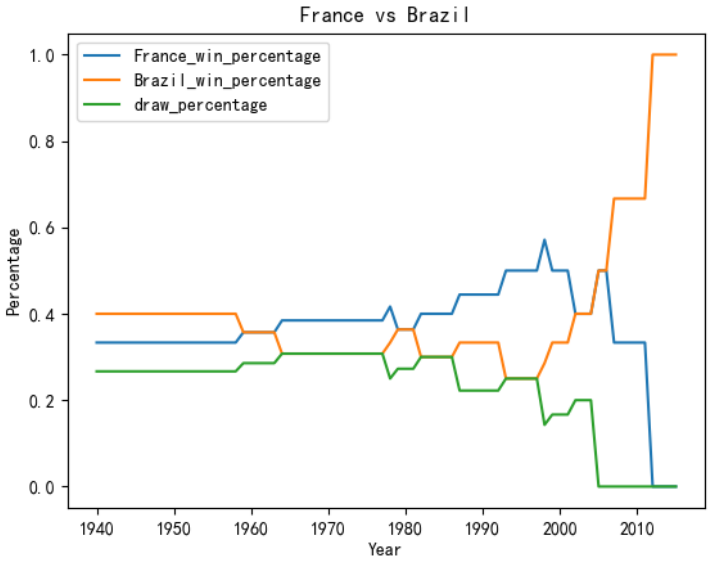
Source: Team Analysis

Initial Data Visualization

Graph Representation and Interpretation

Graph Representation





Task 3:

Find out the winning percentage between arbitrary two teams in the given interval.

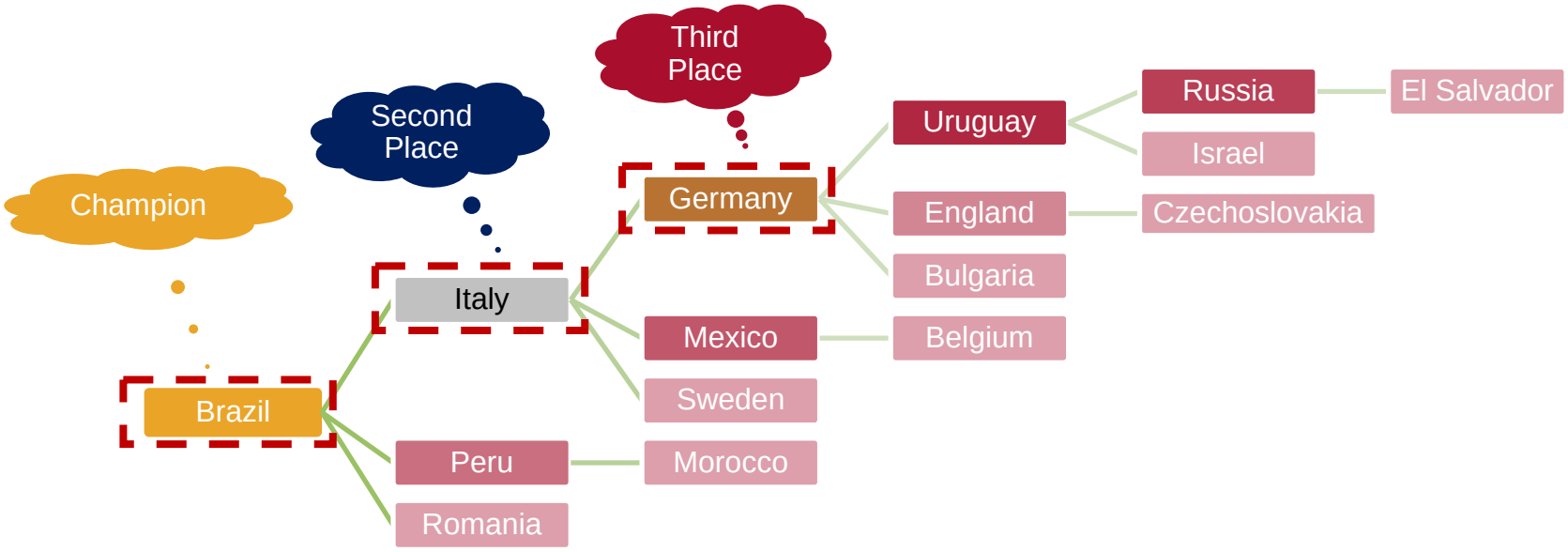
Source: Team Analysis

Contents

Data Preprocessing (Feature Engineering)

Tree-Based Representation of Game

Example



Source: Team Analysis

Data Preprocessing (Feature Engineering)

Tree-Based Features

Details

Further Explanation:

- We made the annual situation of the game into the shape of a tree. The father of each node is the team that defeated itself.

For example, Germany was defeated by Italy, Italy was defeated by Brazil, and Brazil was the champion.

- Extract some features from it, such as depth, number of descendants, etc.

```
def analysis(self):

    # init
    self.feature = {}
    self.feature[""] = {}
    self.feature["neither"] = {}
    for i in self.li:
        self.feature[i.home_team] = {}
        self.feature[i.away_team] = {}

    # depth (recursive)
    def get_depth(name, curd):
        self.feature[name]["depth"] = curd
        for i in self.beatee[name]:
            get_depth(i, curd + 1)

    get_depth("", 0)
```

```
    # n_children (recursive)
    def get_n_children(name):
        n = 0
        for i in self.beatee[name]:
            n += get_n_children(i) + 1
        self.feature[name]["n_children"] = n
        return n

    get_n_children("")

    # n_country
    n = len(country_involved(self.li))
    self.feature["_year"] = {"n_country": n}
    # print("%s: %s" % (self.time, n))
```

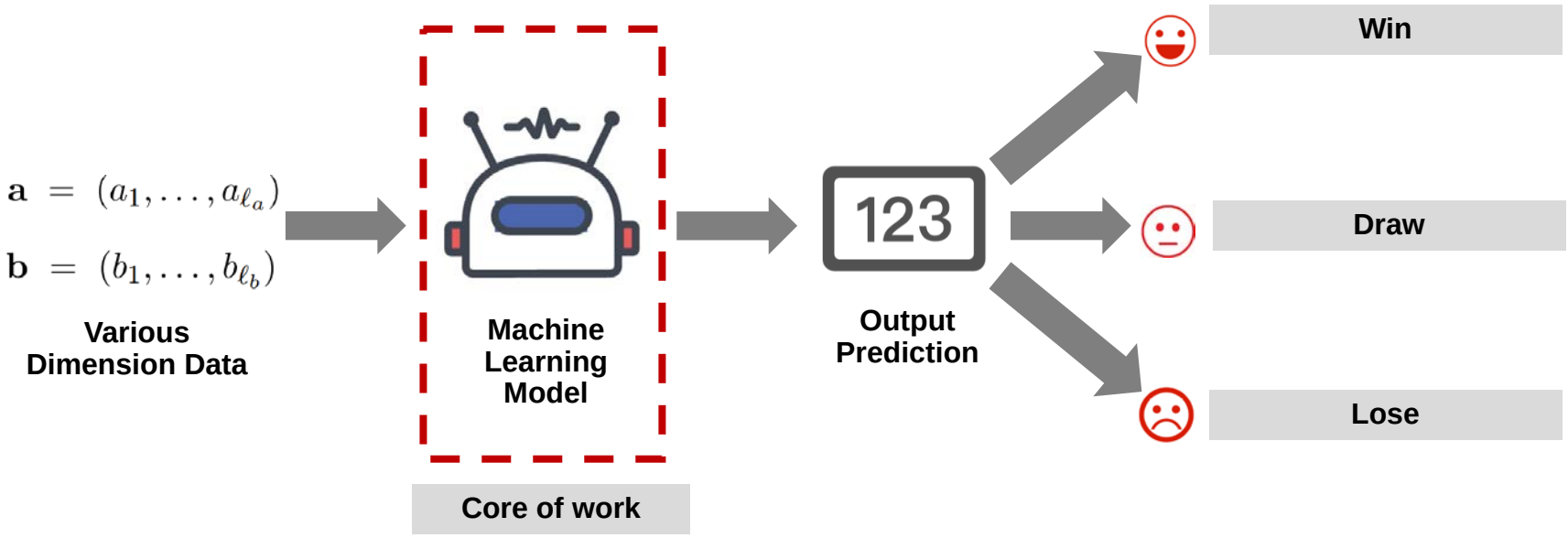
Source: Team Analysis

Contents

Our Proposal: Machine Learning Method

Model Input and Output

Approach Overview



Source: Team Analysis

Our Proposal: Machine Learning Method

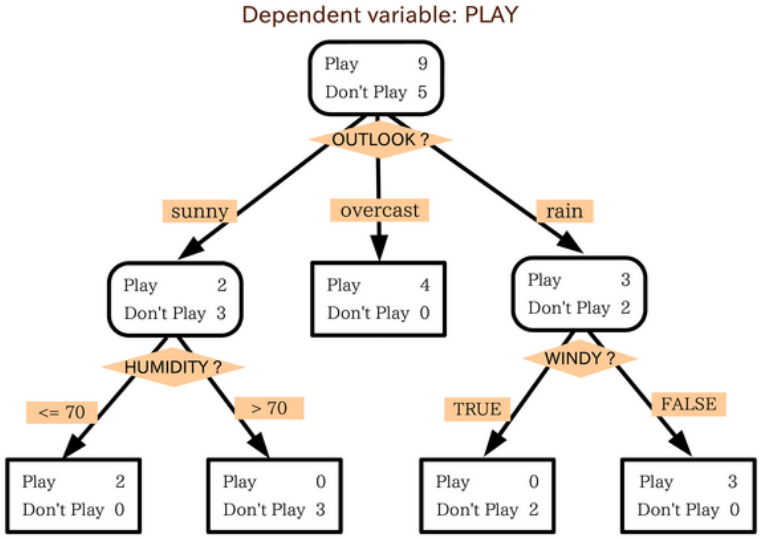
Multi-Grained Cascade Forest

Introduction

An example:

According to environmental variables, we want to predict whether or not to play golf.

The parameters here are all learned during the training process, and if the number of layers and the number of branches are much larger, complex problems can be predicted.

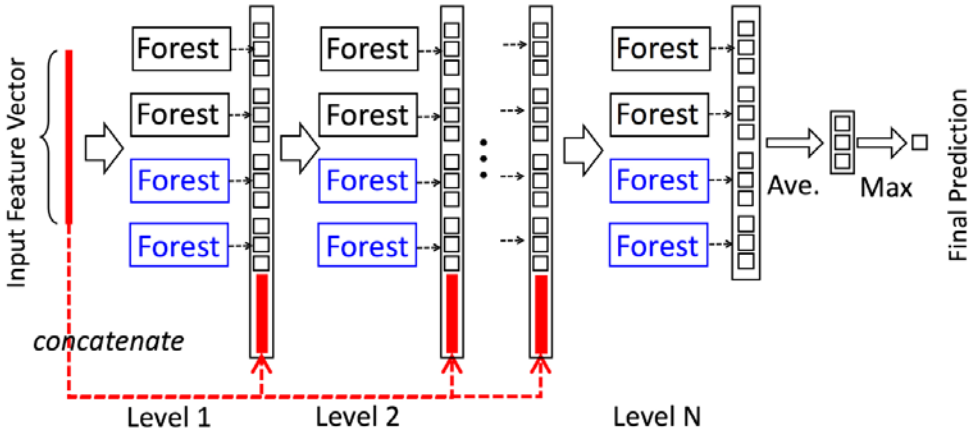


Source: Team Analysis

Our Proposal: Machine Learning Method

Multi-Grained Cascade Forest

Further Illustration



Multiple trees with different parameters form a **random forest**, and the output of random forests is determined by all trees.

The output of the random forest is regarded as the extraction of the input features, and iteratively continues, which is the meaning of the **cascade**.

Source: Team Analysis

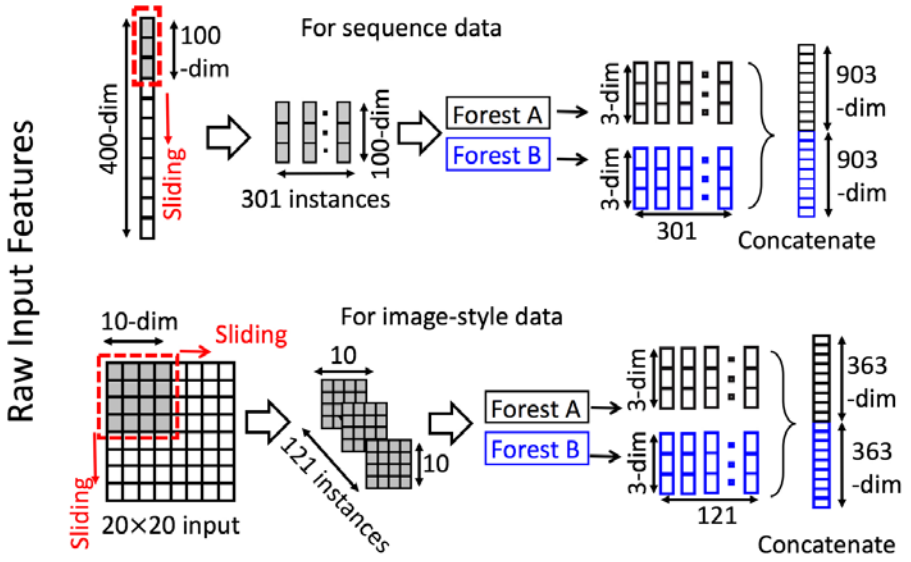
Our Proposal: Machine Learning Method

Multi-Grained Cascade Forest

Further Illustration

At the same time, we will sample the input multiple times, such as 400-dimensional input, and take every 100 consecutive dimensions to get 301 new instances.

This sampling can better identify the connections between the data, which is good for prediction.



Source: Team Analysis

Our Proposal: Machine Learning Method

Multi-Grained Cascade Forest

Python Code

```
class BeaterTree(object):

    def __init__(self, li):
        self.li = li
        self.time = self.li[-1].date
        self.beater = {}
        self.beatee = {}
        self.build()
        self.feature = {}
        self.special = {"", "neither", "_year"}
        self.datumTable = {}

    def build(self):
        self.beater = {}
        self.beater["neither"] = ""
        for i in self.li:
            self.beater[i.home_team] = ""
            self.beater[i.away_team] = ""
        # later loss record will overwirte earlier ones auto
        for i in self.li:
            self.beater[i.loser()] = i.winner()
        self.beater[champion(year(self.time))] = ""
        del (self.beater["neither"])
```

```
        self.beatee = {}
        self.beatee["neither"] = []
        self.beatee[""] = []  # root
        for i in self.li:
            self.beatee[i.home_team] = []
            self.beatee[i.away_team] = []
        for i in self.beater:
            self.beatee[self.beater[i]].append(i)
        del (self.beatee["neither"])

        return
```

Source: Team Analysis

Contents

Result Graph Presentation

2022 Qatar World Cup Prediction

Graph Representation

Source: Team Analysis

Contents

Other Group Trials

Everything worthwhile takes time and commitment.

Group Brainstorm of Several Topics to Try

We failed many times, finally arrived at this work…

Crawler of “Arena of Valor”	Machine learning based on weather data crawler	Beijing travel ticket prices (Succeeded)

It is our great pleasure to have met Professor He Jibo and all of you here @PKU!

Source: Team Analysis

Contents

Contents

References

1. Goodfellow, Y. Bengio, and A. Courville. Deep Learning. MIT Press, Cambridge, MA, 2016.

2. H. Mhaskar, Q. Liao, and T. A. Poggio. When and why are deep networks better thanshallow ones? In Proceedings of the 31st AAAI Conference on Artificial Intelligence,pages 2343{2349, San Francisco, CA, 2017.

3. Zhou Z H, Feng J. Deep forest: Towards an alternative to deep neural networks[J]. arXiv preprint arXiv:1702.08835, 2017.

4. D. L. Richmond, D. Kainmueller, M. Y. Yang, E. W. Myers, and C. Rother. Relating cascaded random forests to deep convolutional neural networks for semantic segmentation. arXiv:1507.07583, 2015.

Scan QR code to get our Python code on GitHub!

Thank you!