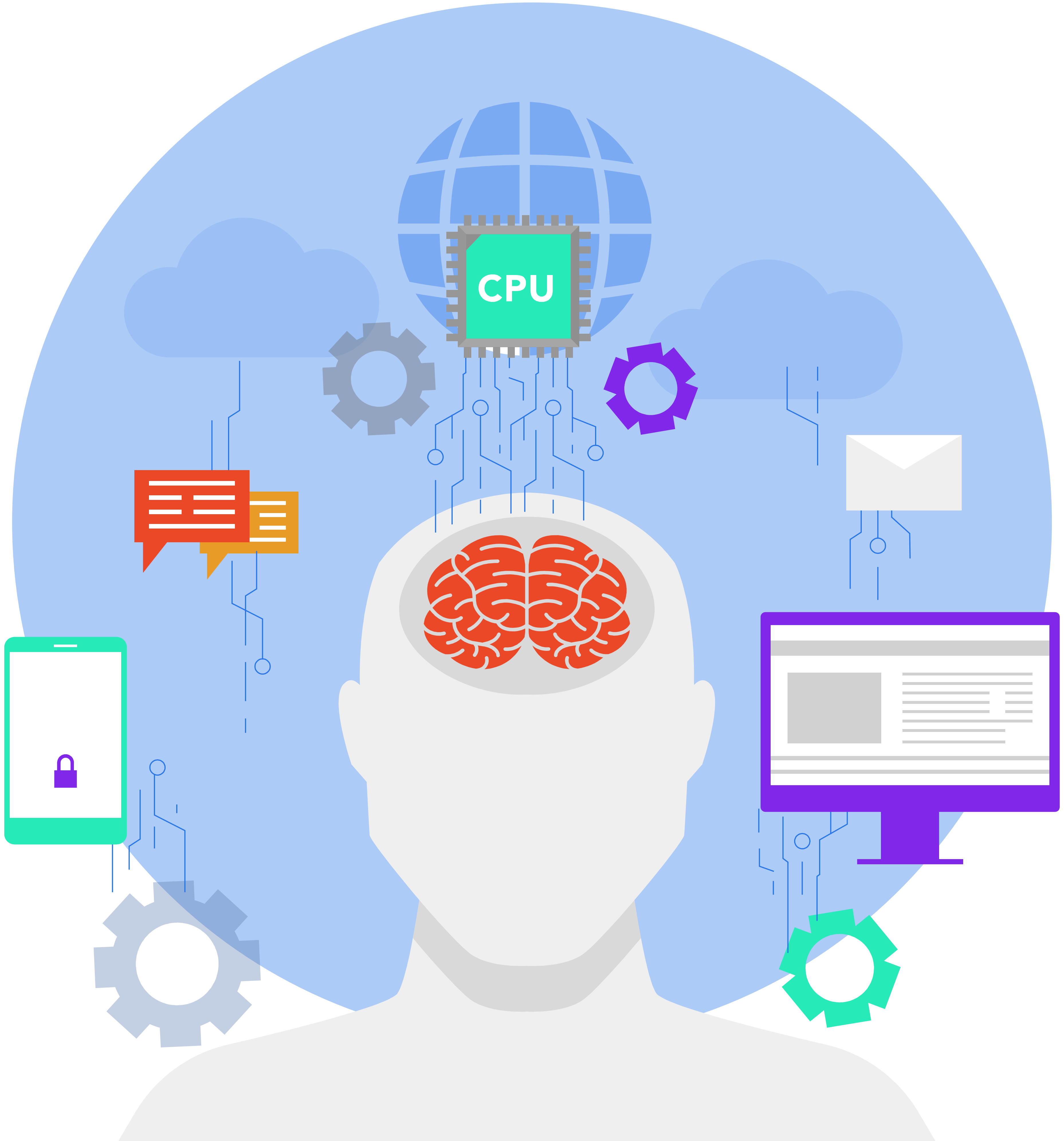


Project Machine Learning

Amina ADDI
Anaïs ASSOGANE
Yasmine GAOUI

Machine Learning

01 Dataset,
data cleaning &
transformation

02 Data visualization

03 Classification

04 Clustering

Dataset

Start

Diver
Nom du plongeur

Nationality
Nationalité du plongeur

Gender
Genre du plongeur (NOC)

Discipline
FIM : Immersion libre.
CNF : Brasse.
CWT : Poids constant (souvent avec monopalme).
CWT-B : Poids constant en bi-palmes.

AP
Announced performance

RP
Realized performance

Card
Blanc : Objectif atteint avec protocole respecté.
Jaune : Profondeur non atteinte mais protocole respecté.
Rouge : Protocole non respecté.

Points

Remarks

Title event

Event type

Day

Line
Colonne vide

Official top
Heure officiel du depart du plongeur

Data cleaning & transformation

Explore Diver

```
data.rename(columns={'Diver': 'Name'})
```

Explore Gender

M	**20310**
F	**11407**

```
data['Gender'].map({'M': 'Male', 'F': 'Female'})
```

Explore Nationality

KOR	3327
FRA	2259
JPN	1932
USA	1740
GBR	1277
...	
MDA	1
KEN	1

Explore Discipline

CWT	11757
FIM	9473
CNF	5673
CWTB	4814

Explore AP

```
Non-Null Count  Dtype
--------------  -----
31717 non-null  object
```

```
data['AP'].str.extract('(\d+)').astype(float)
```

Explore RP

```
Non-Null Count  Dtype
--------------  -----
31717 non-null  object
```

```
data['RP'].str.extract('(\d+)').astype(float)
```

Explore Day

2011-09-15	242
2013-09-15	211
...	
2022-07-25	1

```
pd.to_datetime(df['Day']).dt.month
pd.to_datetime(df['Day']).dt.year
```

Explore Card

WHITE	22635
YELLOW	5965
RED	3117

Explore Points

```
Non-Null Count  Dtype
--------------  -----
31717 non-null  float64
```

```
data['Points'].apply(lambda x: x if x >= 0 else None)
```

Data cleaning & transformation

	Name	Nationality	Gender	Discipline	AP	RP	Card	Points	Remarks	Event Type	Month	Year	Season	experience_dive	experience_discipline
0	Deborah Andollo	CUB	Female	CWT	61.0	61.0	WHITE	61.0	OK	Worldrecord attempt	6	1994	Summer	1	1
1	Umberto Pelizzari	ITA	Male	CWT	72.0	72.0	WHITE	72.0	OK	Worldrecord attempt	9	1995	Autumn	1	1
2	Deborah Andollo	CUB	Female	CWT	62.0	62.0	WHITE	62.0	OK	Worldrecord attempt	10	1996	Autumn	2	2
3	Michael Oliva	FRA	Male	CWT	72.0	72.0	WHITE	72.0	OK	Worldrecord attempt	10	1996	Autumn	1	1
4	Alejandro Ravelo	CUB	Male	CWT	73.0	73.0	WHITE	73.0	OK	Worldrecord attempt	8	1997	Summer	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
31366	Keenan Alexei Barrameda	PHL	Male	CWT	36.0	36.0	WHITE	36.0	OK	Depth Competition	11	2024	Autumn	6	2
31367	Ramon Paolo Robles	PHL	Male	CWT	33.0	33.0	WHITE	33.0	OK	Depth Competition	11	2024	Autumn	4	1
31368	James Bernard Gabriel	PHL	Male	CNF	25.0	25.0	WHITE	25.0	OK	Depth Competition	11	2024	Autumn	5	1
31369	Franklin Tabora	PHL	Male	CWT	25.0	25.0	WHITE	25.0	OK	Depth Competition	11	2024	Autumn	4	2

+ experience_dive

```
data.groupby('Name').cumcount() + 1
```

+ experience_discipline

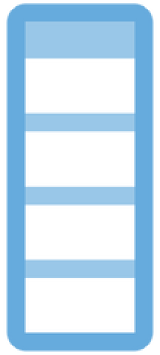
```
data.groupby(
['Name', 'Discipline']
).cumcount() + 1
```

+ Season

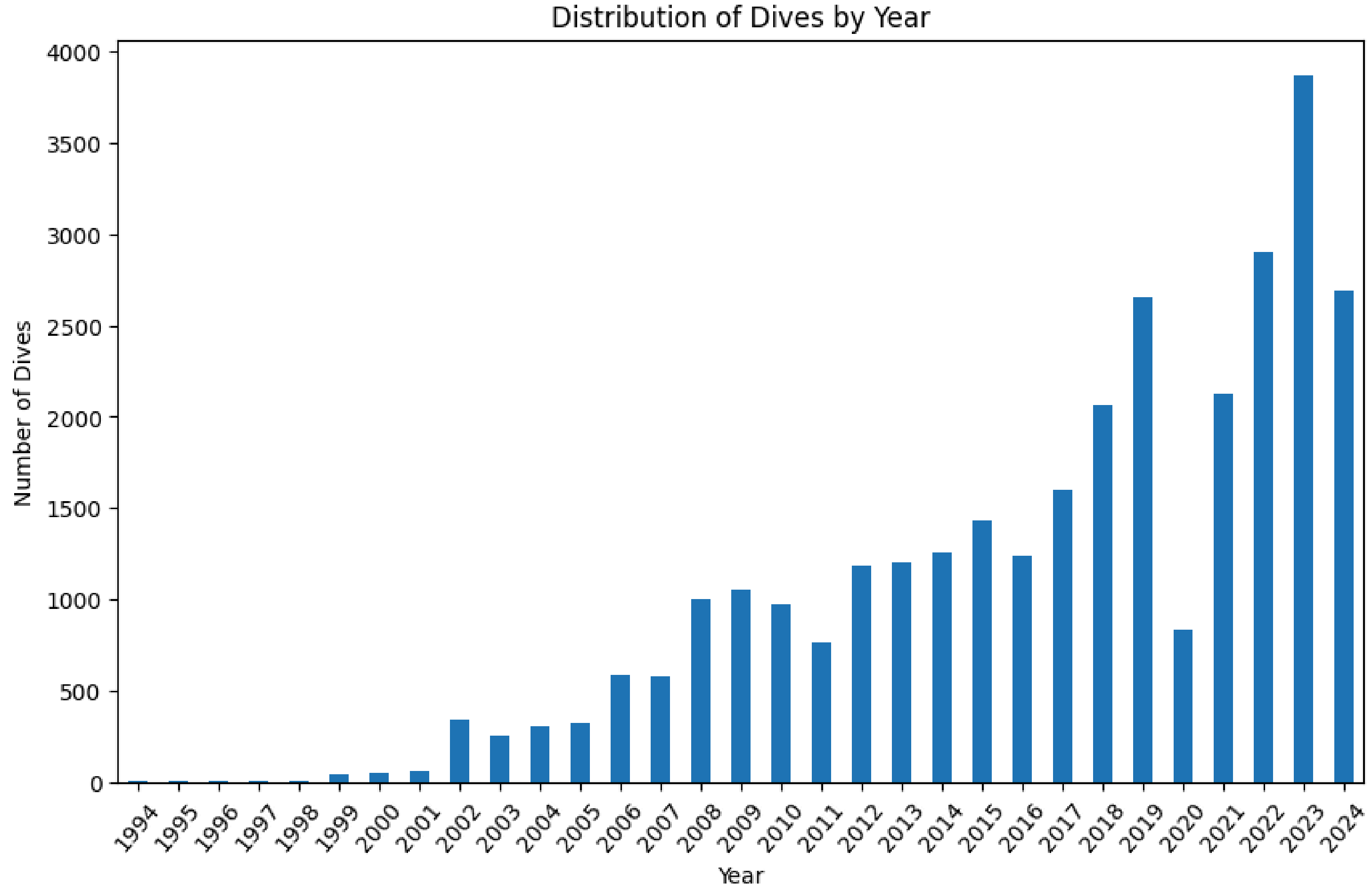
```
if month in [12, 1, 2]:
    return 'Winter'
elif month in [3, 4, 5]:
    return 'Spring'
elif month in [6, 7, 8]:
    return 'Summer'
else:
    return 'Autumn'
```

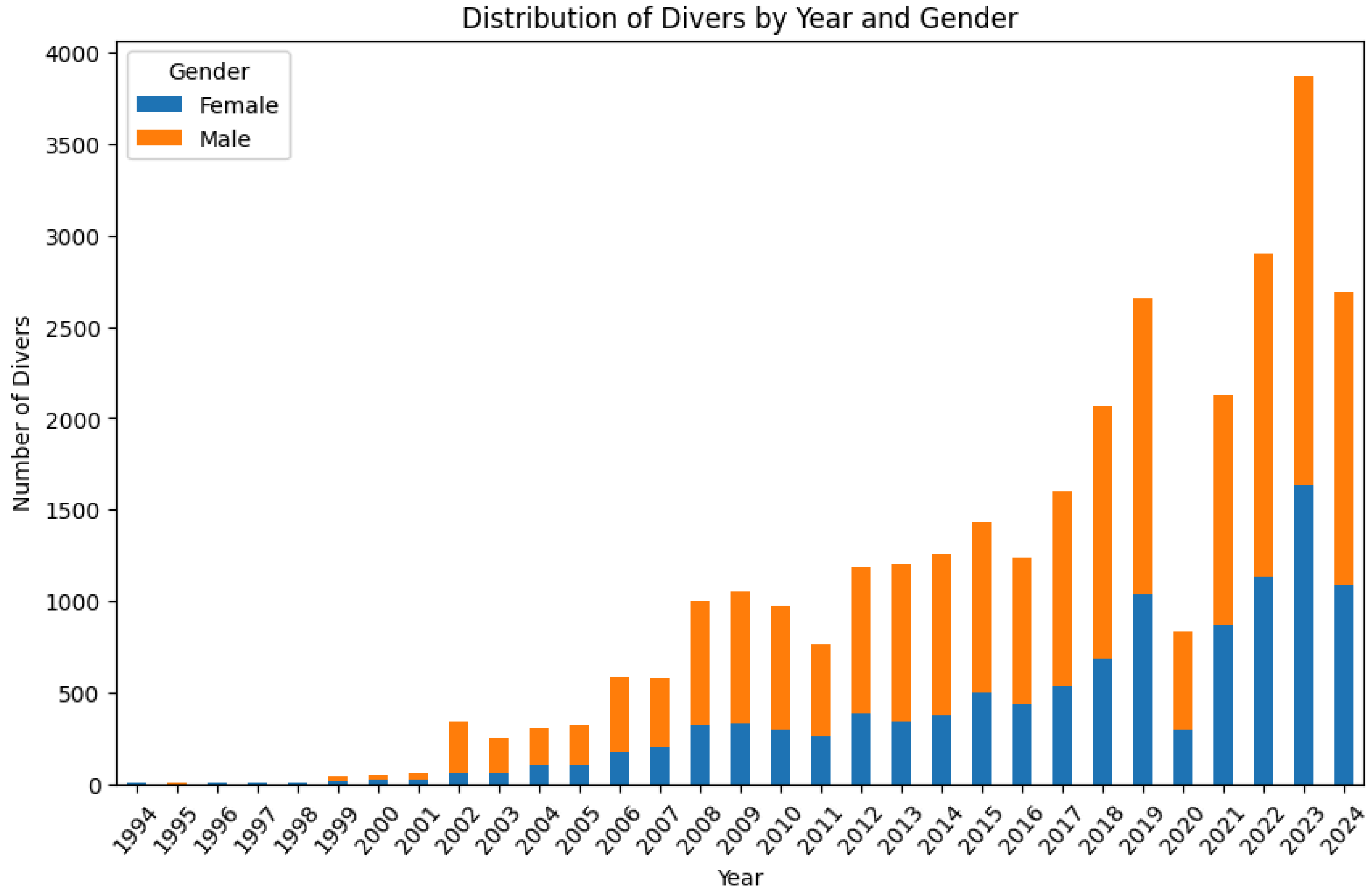
Data cleaning & transformation

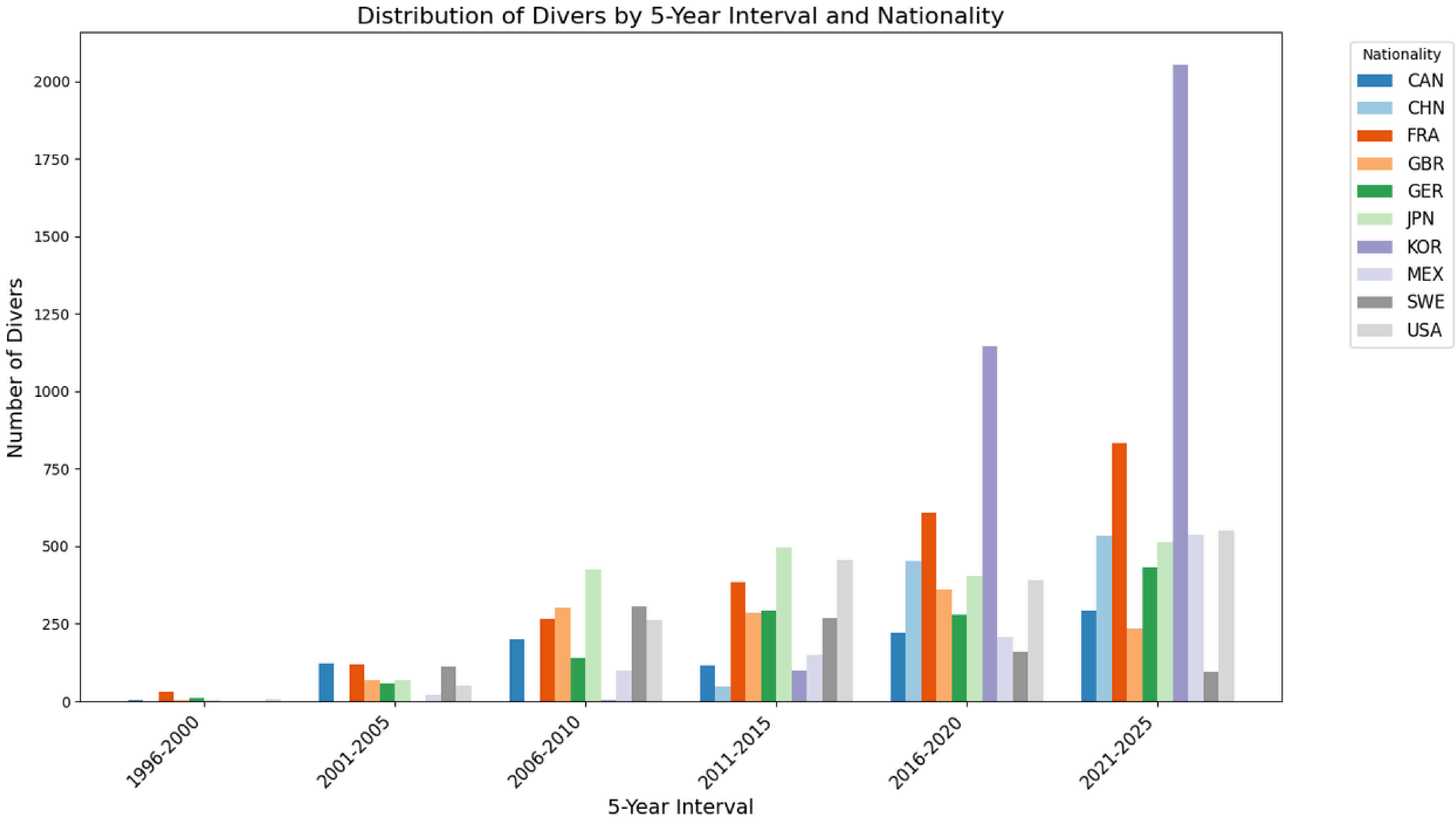
```
data.drop_duplicates(inplace=True)

data.drop(["Start", "Line", "Official Top", "Title Event"], axis=1)

data.drop(columns=['Unnamed: 0'], errors='ignore', axis=1)

data.drop(['Day'], axis=1)

data.to_csv('Dataset.csv')
```



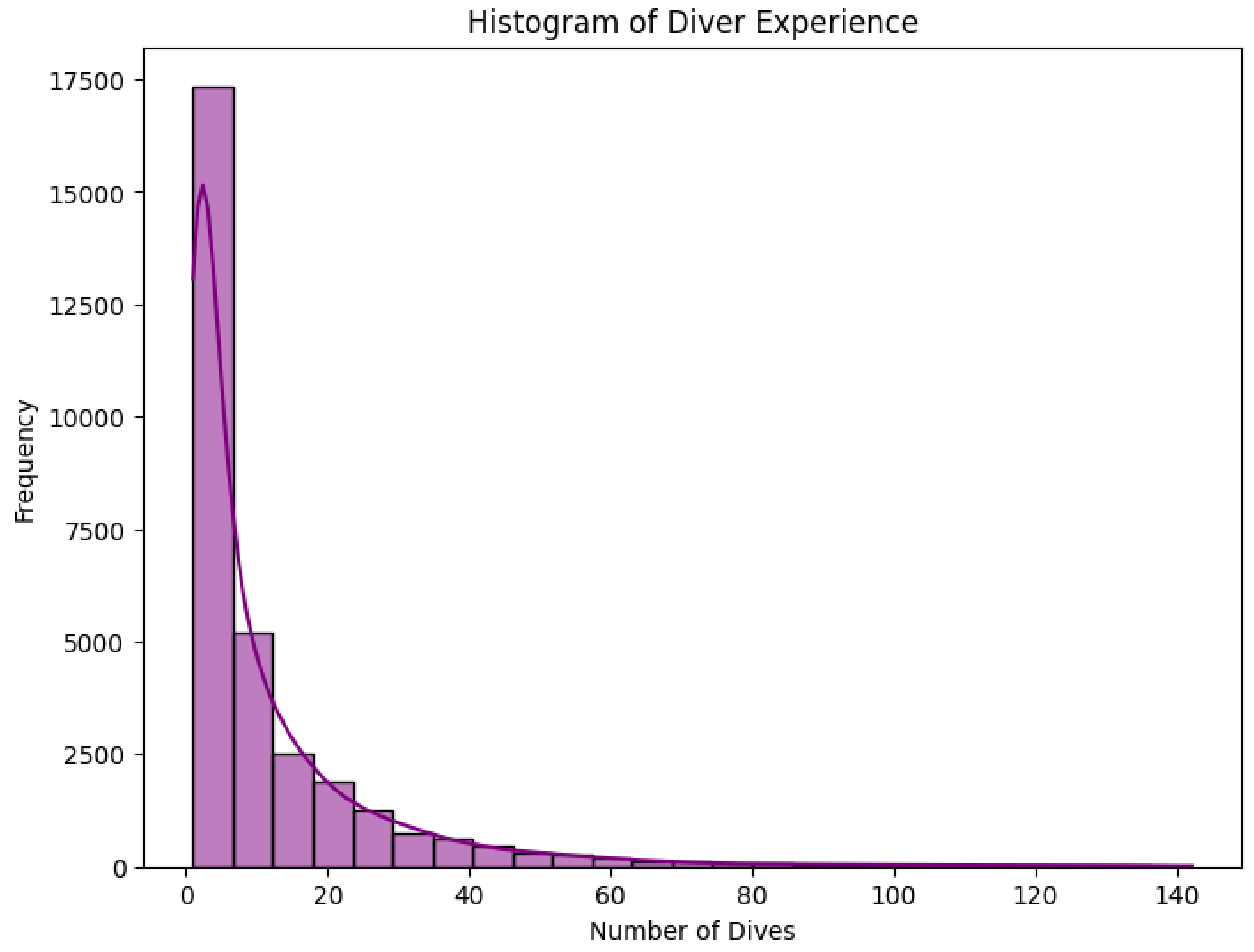
Data visualization

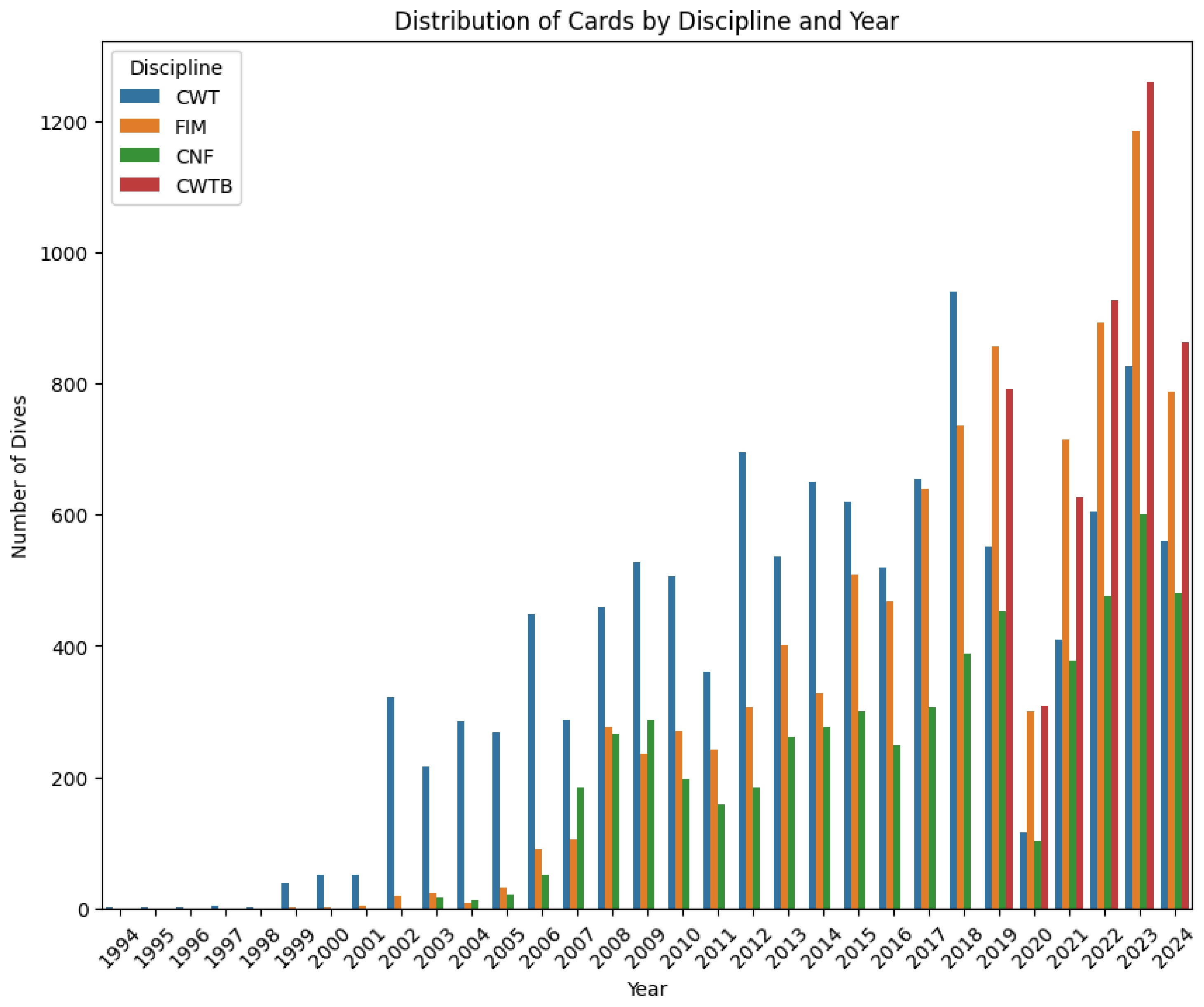


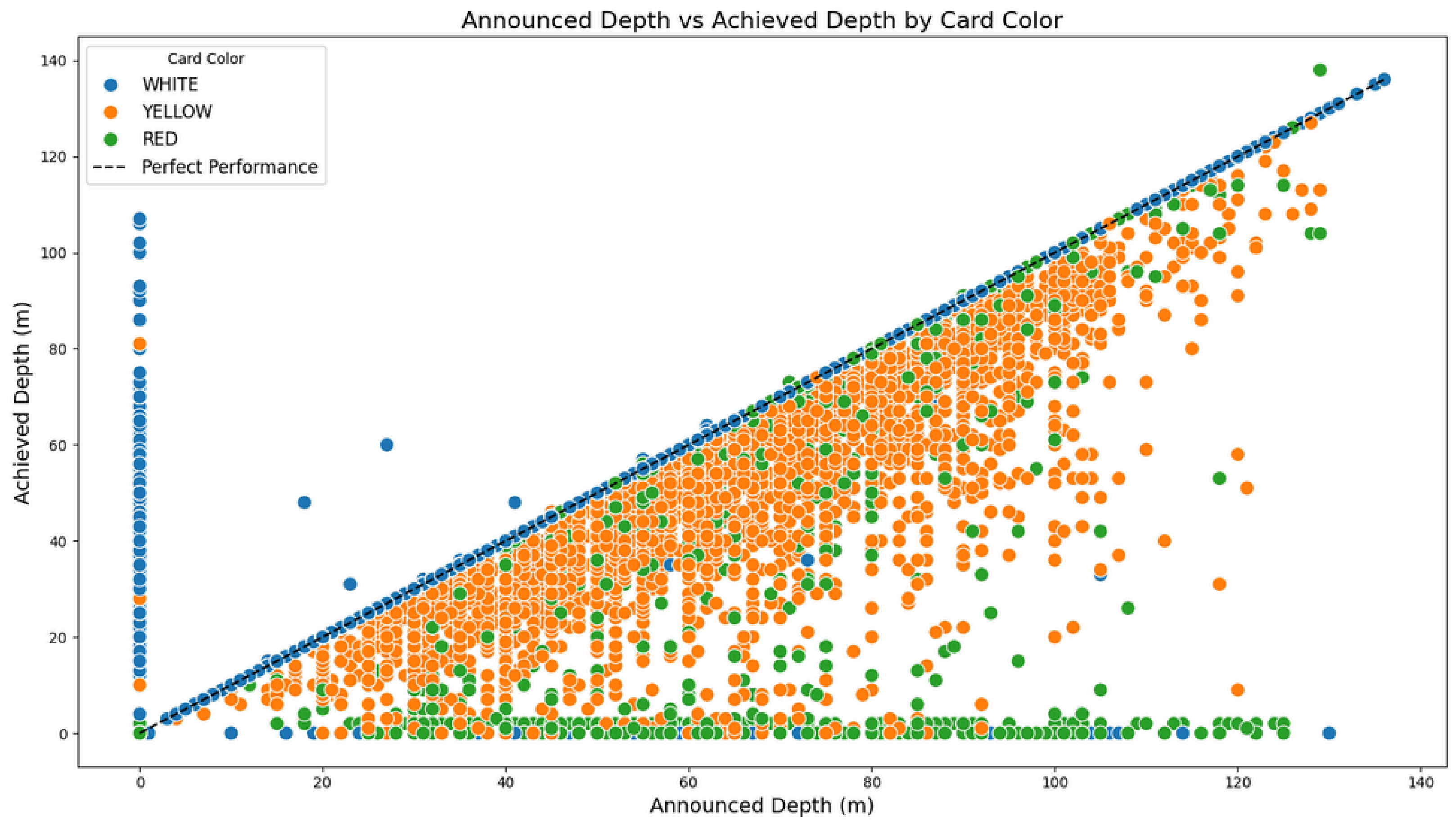


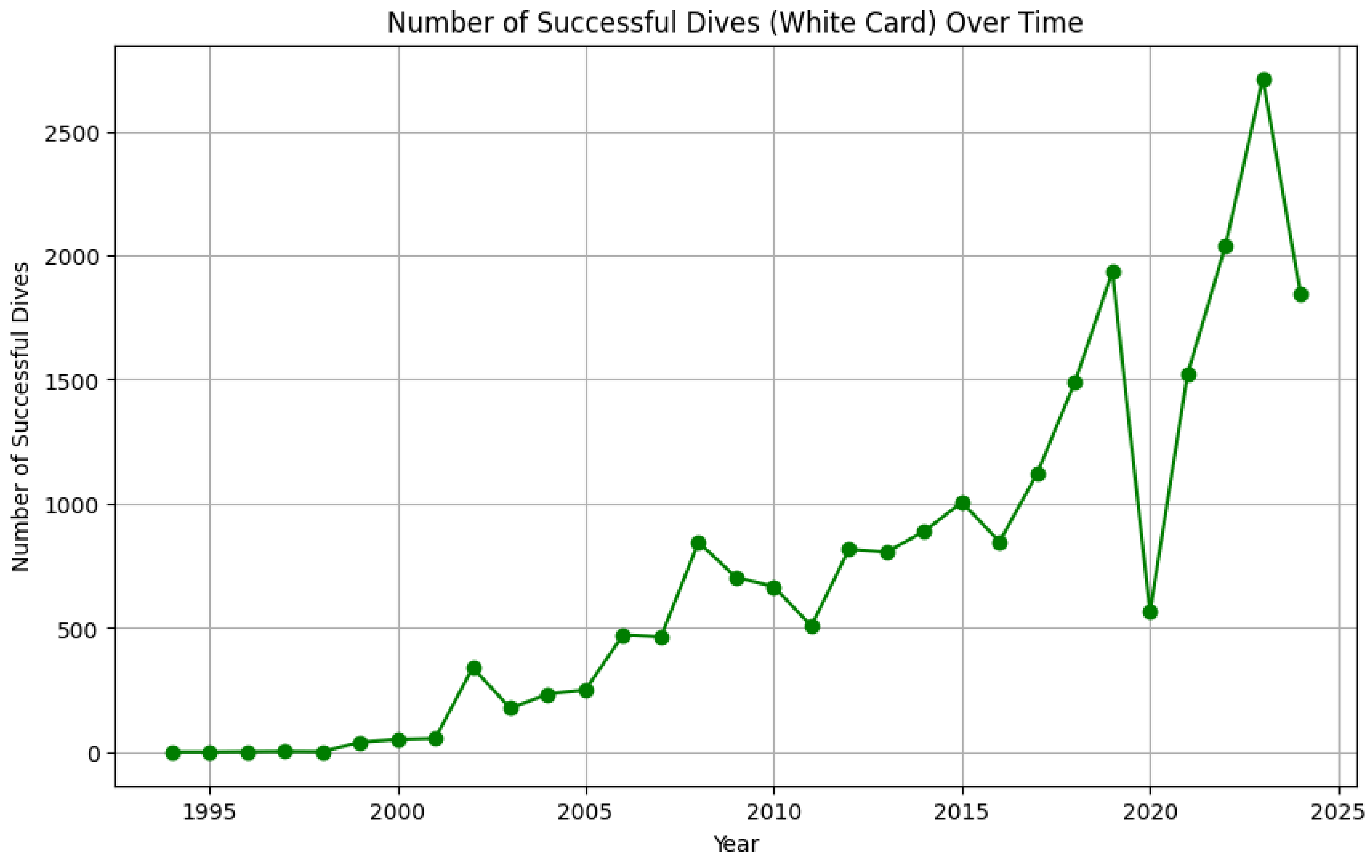


Data visualization

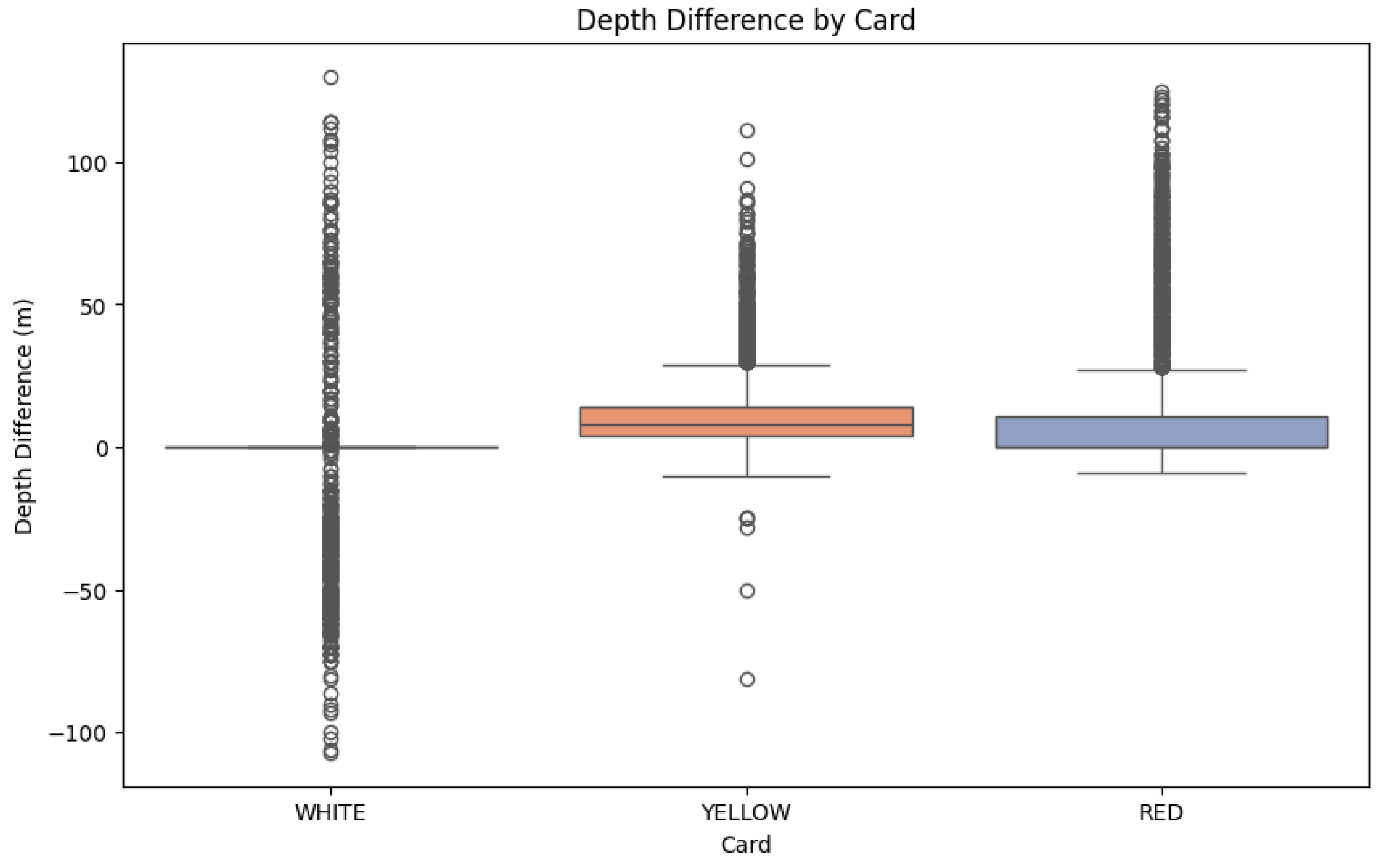


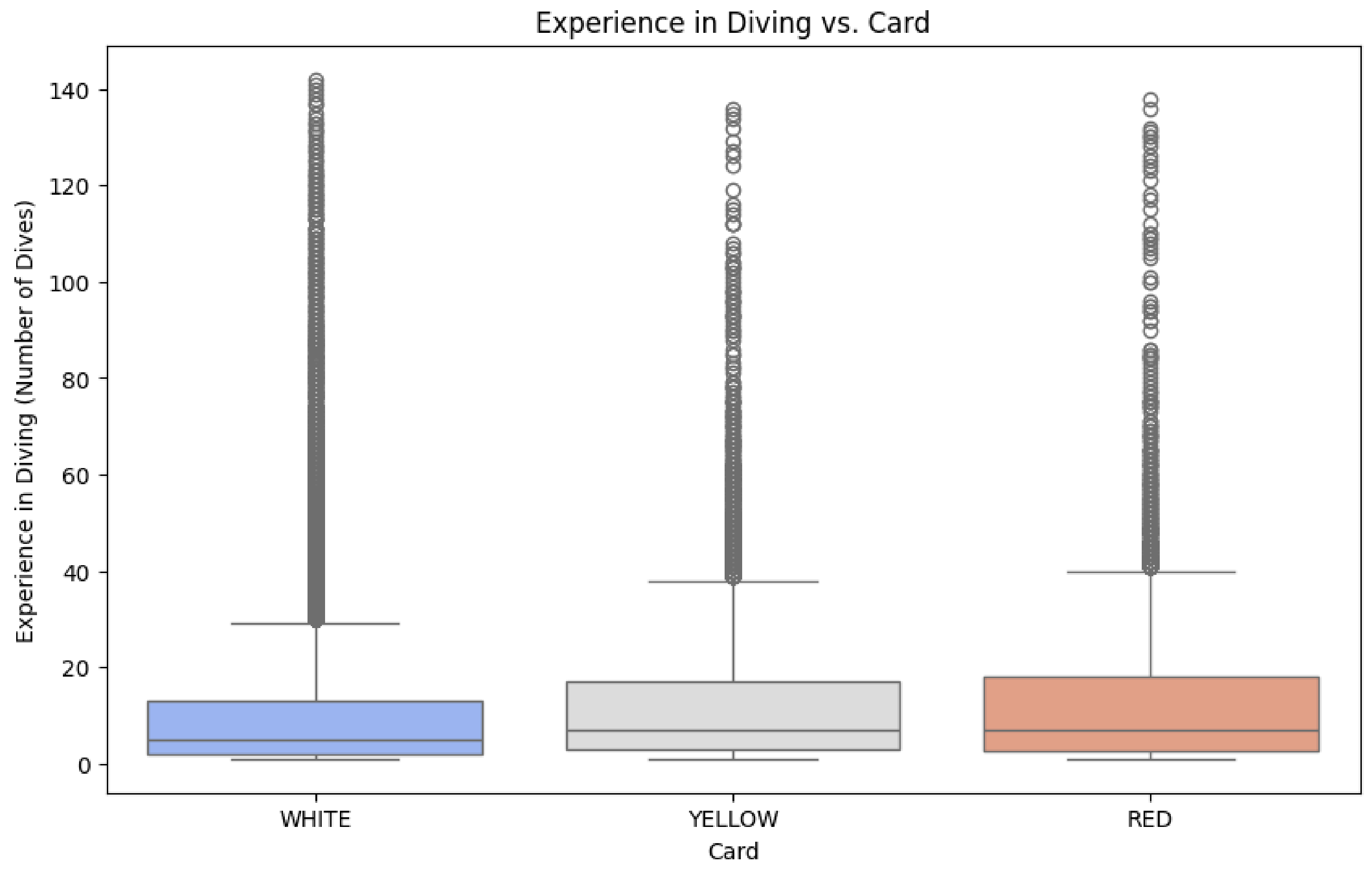


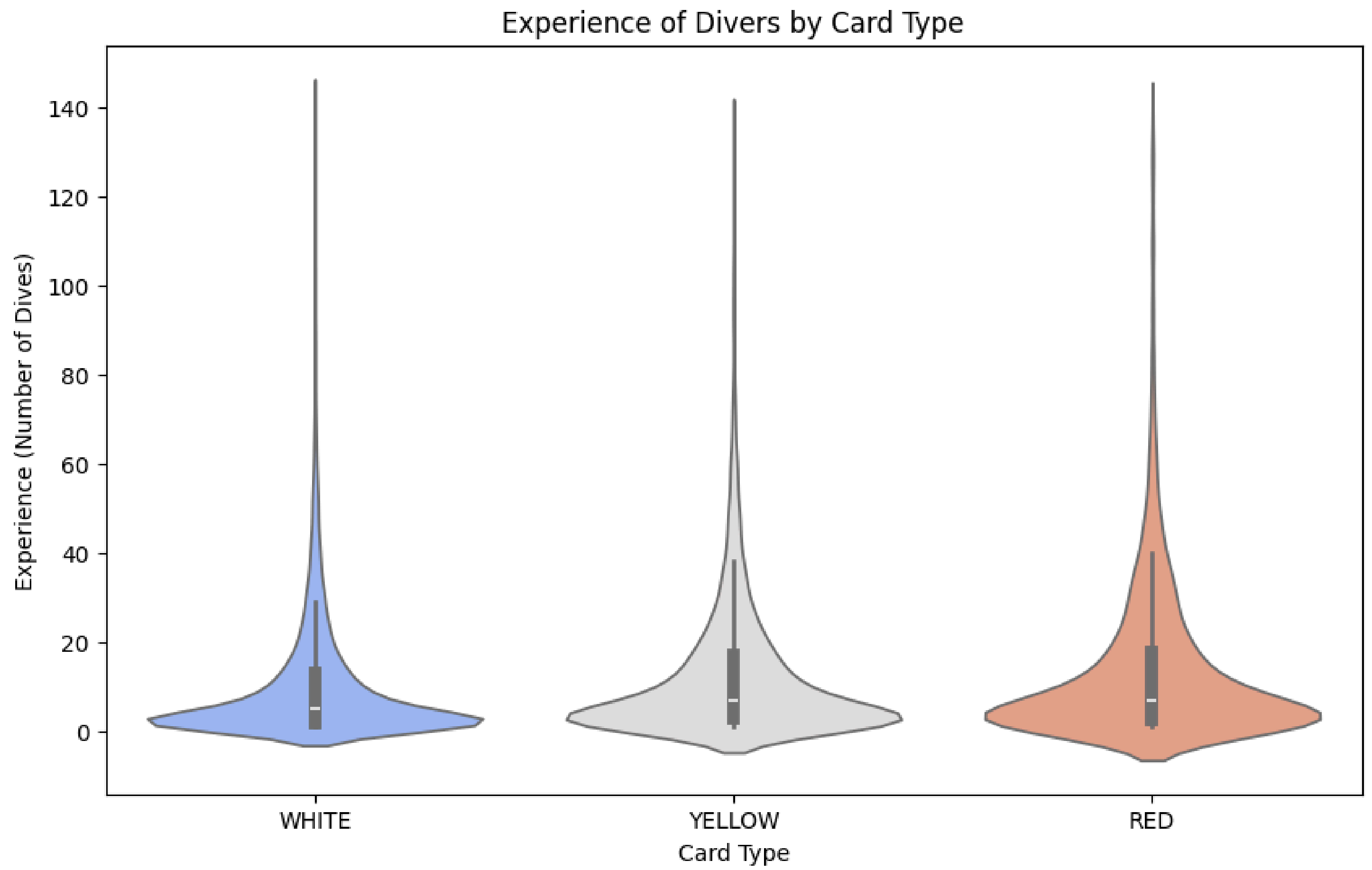


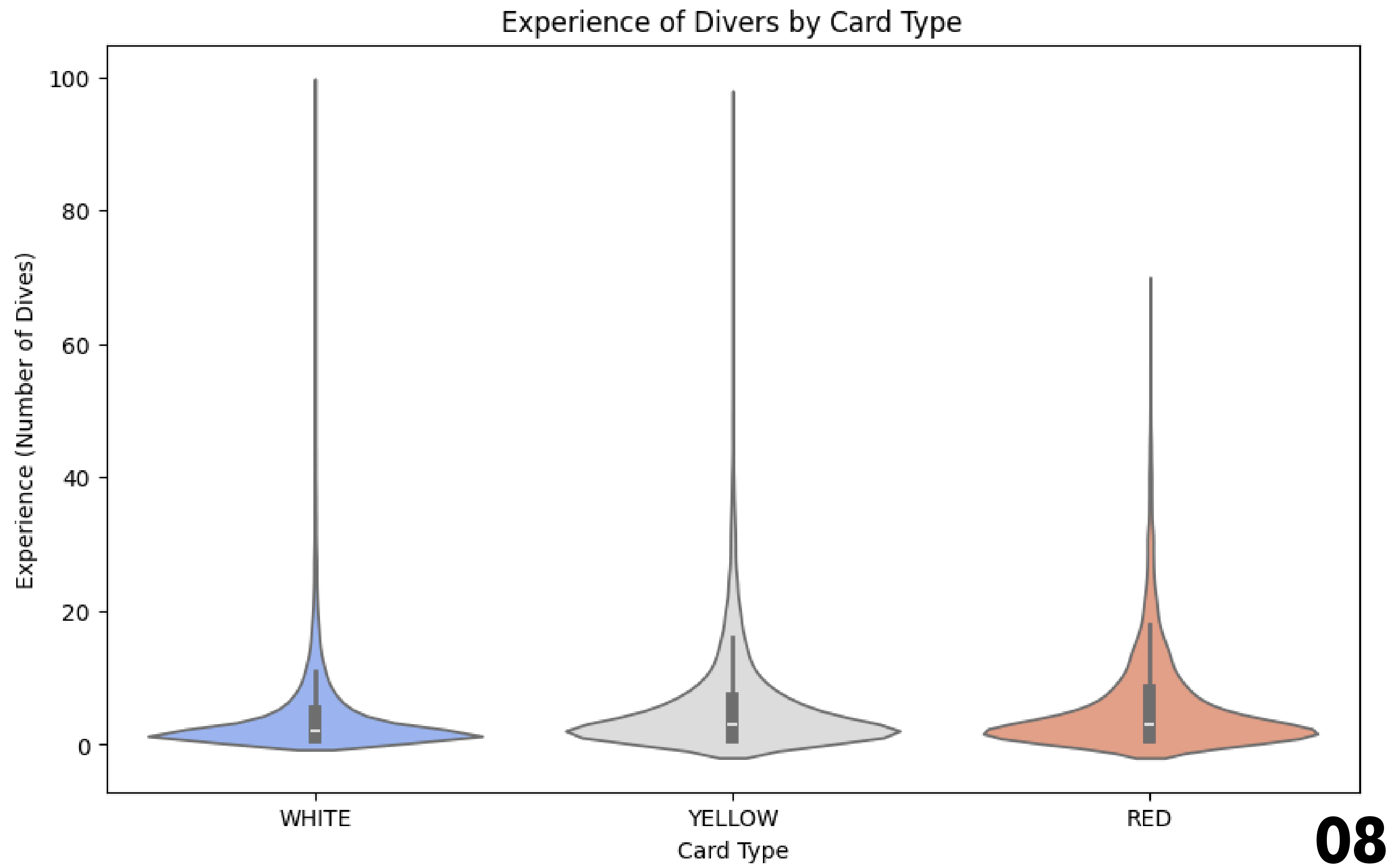


Data visualization









Classification

Features: AP, RP, experience_dive, experience_discipline, Month, Year

Target: Card

DATA

70% **Train**

30% **Test**

SVM

Training

Predictions

Why SVM?

Classification Report:

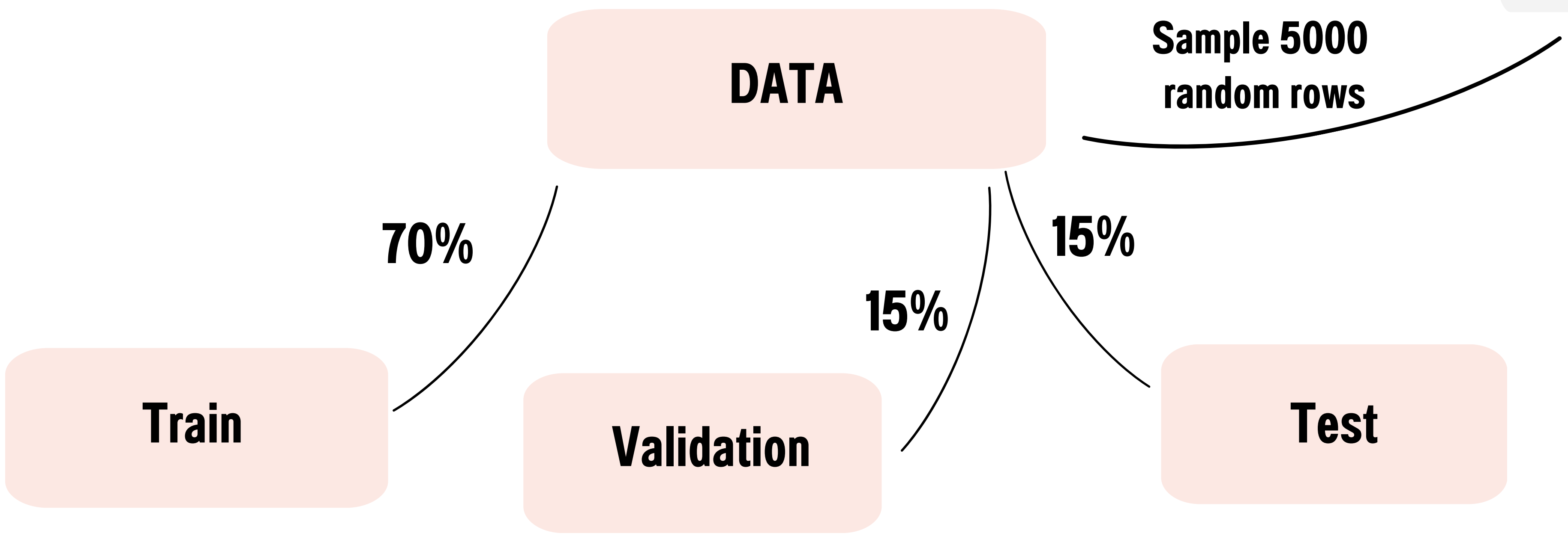
	precision	recall	f1-score	support
RED	0.00	0.00	0.00	953
WHITE	0.87	0.99	0.93	6692
YELLOW	0.80	0.81	0.80	1767
accuracy			0.86	9412
macro avg	0.56	0.60	0.58	9412
weighted avg	0.77	0.86	0.81	9412

Classification

Features: AP, RP, experience_dive, experience_discipline, Month, Year

Target: Card

WHITE	YELLOW	RED
22635	5965	3117



SVM (Linear, Poly, RBF)

Training

Predictions

C = 1.0

RED: 2.2
WHITE: 1.0
YELLOW: 2.0

Linear

Test Classification Report:

	precision	recall	f1-score	support
RED	0.33	0.46	0.39	460
WHITE	0.71	0.50	0.58	750
YELLOW	0.84	0.88	0.86	885
accuracy			0.65	2095
macro avg	0.63	0.61	0.61	2095
weighted avg	0.68	0.65	0.66	2095

Poly

Test Classification Report:

	precision	recall	f1-score	support
RED	0.39	0.42	0.40	460
WHITE	0.77	0.17	0.28	750
YELLOW	0.56	0.92	0.70	885
accuracy			0.54	2095
macro avg	0.58	0.50	0.46	2095
weighted avg	0.60	0.54	0.48	2095

Rbf

Test Classification Report:

	precision	recall	f1-score	support
RED	0.41	0.61	0.49	460
WHITE	0.69	0.54	0.61	750
YELLOW	0.87	0.81	0.84	885
accuracy			0.67	2095
macro avg	0.66	0.66	0.65	2095
weighted avg	0.71	0.67	0.68	2095

Classification

Features: AP, RP, experience_dive, experience_discipline, Month, Year

Target: Card

WHITE	YELLOW	RED
22635	5965	3117

Sample 5000 random rows

DATA

70% Train

15% Validation

15% Test

Random Forest Classifier

Training

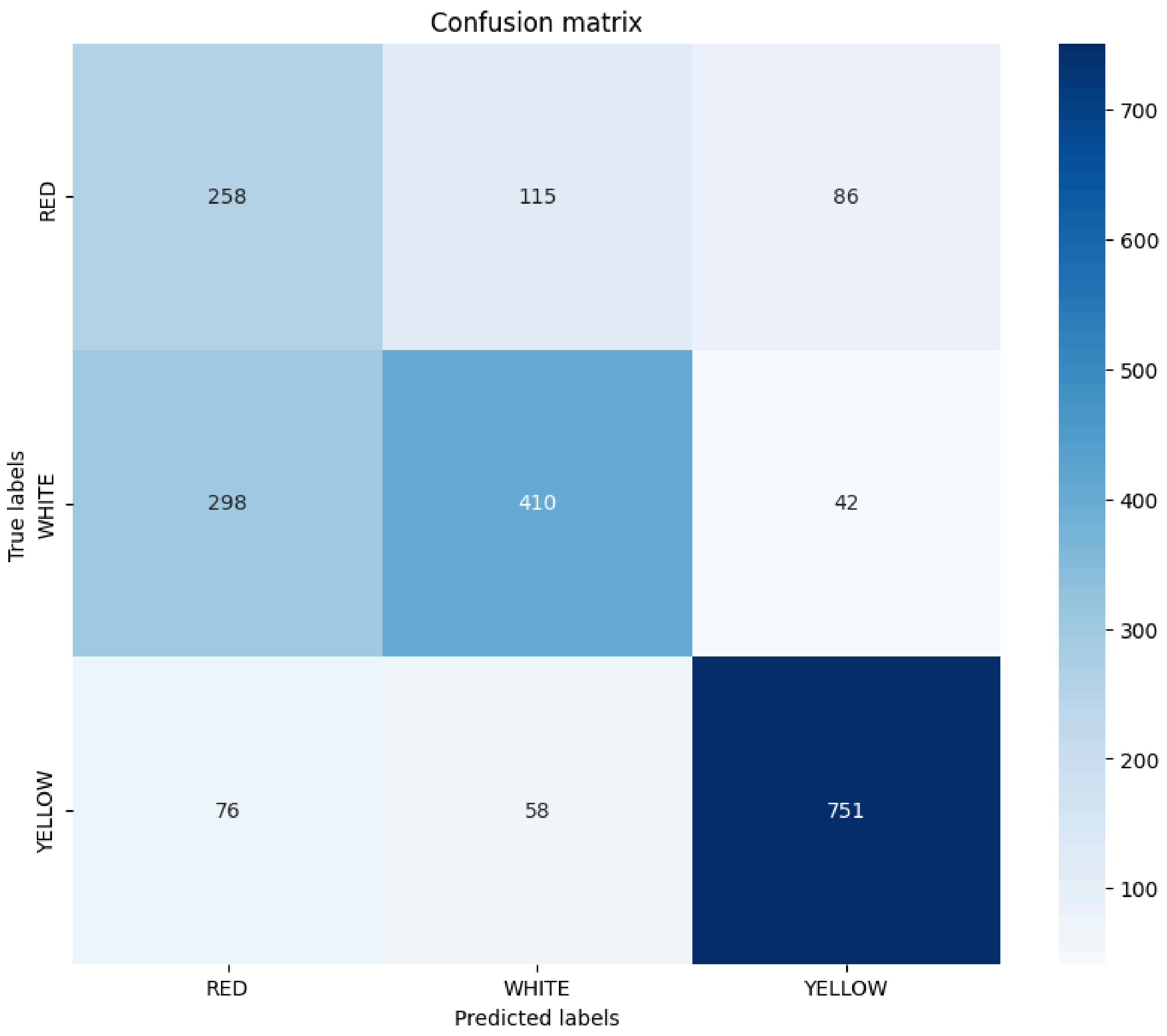
Predictions

Balanced

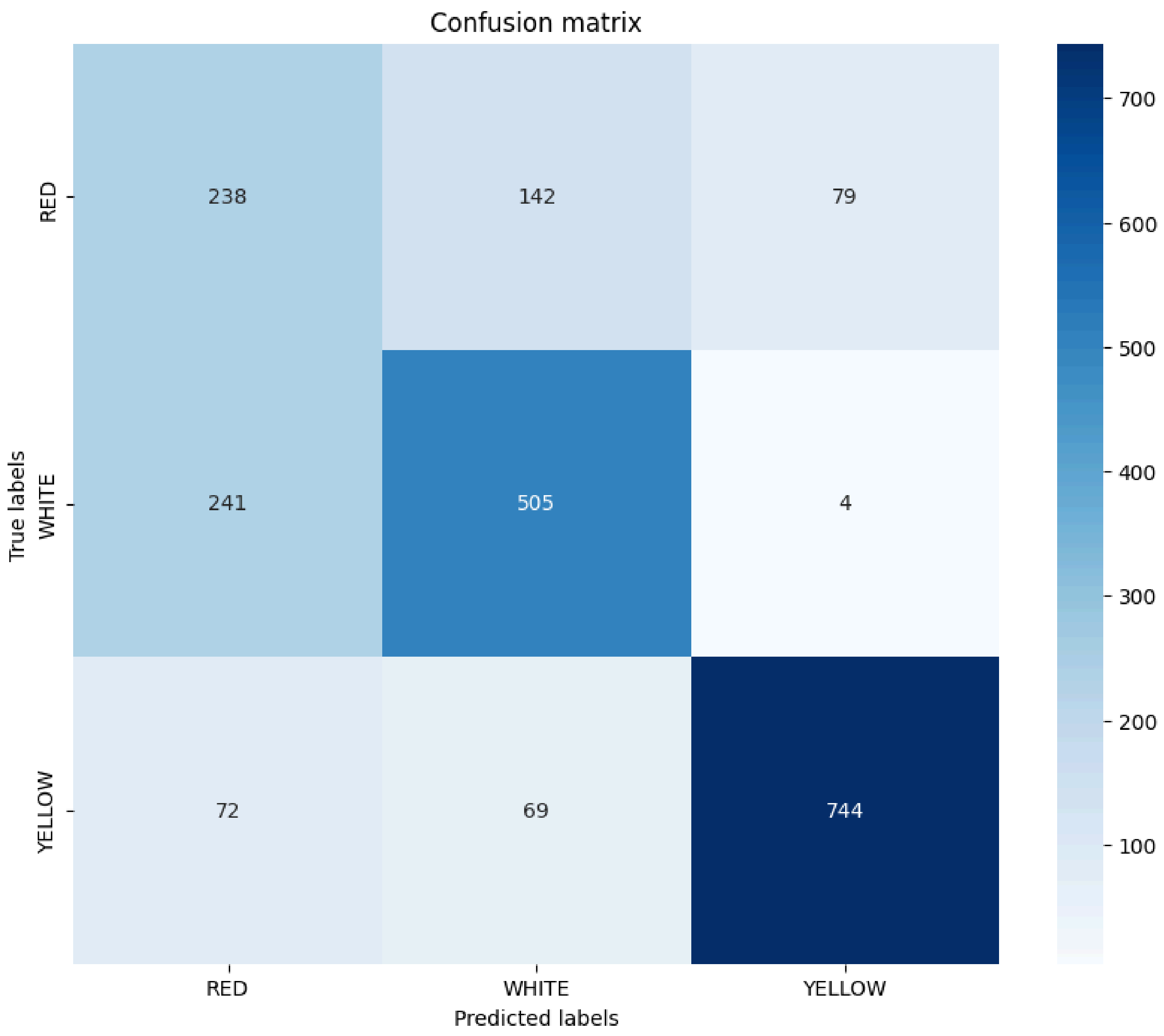
	precision	recall	f1-score	support
RED	0.52	0.44	0.48	459
WHITE	0.67	0.78	0.72	750
YELLOW	0.87	0.83	0.85	885
accuracy			0.72	2094
macro avg	0.69	0.68	0.68	2094
weighted avg	0.72	0.72	0.72	2094

Classification

With SVM (rbf kernel)



With Random Forest Classifier



	Model	Accuracy	Precision	Recall
0	Random Forest	0.710124	0.678962	0.677510
1	SVM (Linear)	0.655683	0.625709	0.614291
2	SVM (RBF)	0.677650	0.655289	0.652449
3	SVM (Polynomial)	0.543935	0.590666	0.506769

Clustering

```
# 1. Remplacer les NaN par 0
data.fillna(0, inplace=True)
```

et on change les NaN par des 0

on procède à un cleaning avant de commencer

```
print(data.isna().sum()) #plus de NaN car tout a été bien remplacé par 0
✓ 0.0s

Unnamed: 0          0
Name                0
AP                  0
RP                  0
Card                0
                   ..
Nationality_ZWE     0
Gender_Male         0
Discipline_CWT      0
Discipline_CWTB     0
Discipline_FIM      0
Length: 131, dtype: int64
```

	Unnamed: 0	Name	Nationality	Gender	Discipline	AP	RP	Card	Points	Remarks	Event Type	Month	Year	Season	experience_dive	experience_discipline
0	0	Deborah Andollo	CUB	Female	CWT	61.0	61.0	WHITE	61.0	OK	Worldrecord attempt	6	1994	Summer	1	1
1	1	Umberto Pelizzari	ITA	Male	CWT	72.0	72.0	WHITE	72.0	OK	Worldrecord attempt	9	1995	Autumn	1	1
2	2	Deborah Andollo	CUB	Female	CWT	62.0	62.0	WHITE	62.0	OK	Worldrecord attempt	10	1996	Autumn	2	2
3	3	Michael Oliva	FRA	Male	CWT	72.0	72.0	WHITE	72.0	OK	Worldrecord attempt	10	1996	Autumn	1	1
4	4	Alejandro Ravelo	CUB	Male	CWT	73.0	73.0	WHITE	73.0	OK	Worldrecord attempt	8	1997	Summer	1	1

Clustering

1. On change les données “catégoriques” en données “numériques”

TRANSFORMATION DES DONNEES CATEGORIE EN NUMERIQUE (car le clustering ne fonctionne pas sur des données numériques)

```python
from sklearn.preprocessing import LabelEncoder

# Colonnes à encoder
columns_to_encode_red = ['Discipline', 'Nationality', 'Gender']  # 'Discipline' existe dans red_dives
columns_to_encode_white = ['Nationality', 'Gender']  # 'Discipline' n'est pas dans white_dives


encoder = LabelEncoder()

numeric_data = encoder.fit_transform(data["Card"])
print(numeric_data)
```

✓ 1.4s

```
[1 1 1 ... 1 1 1]
```

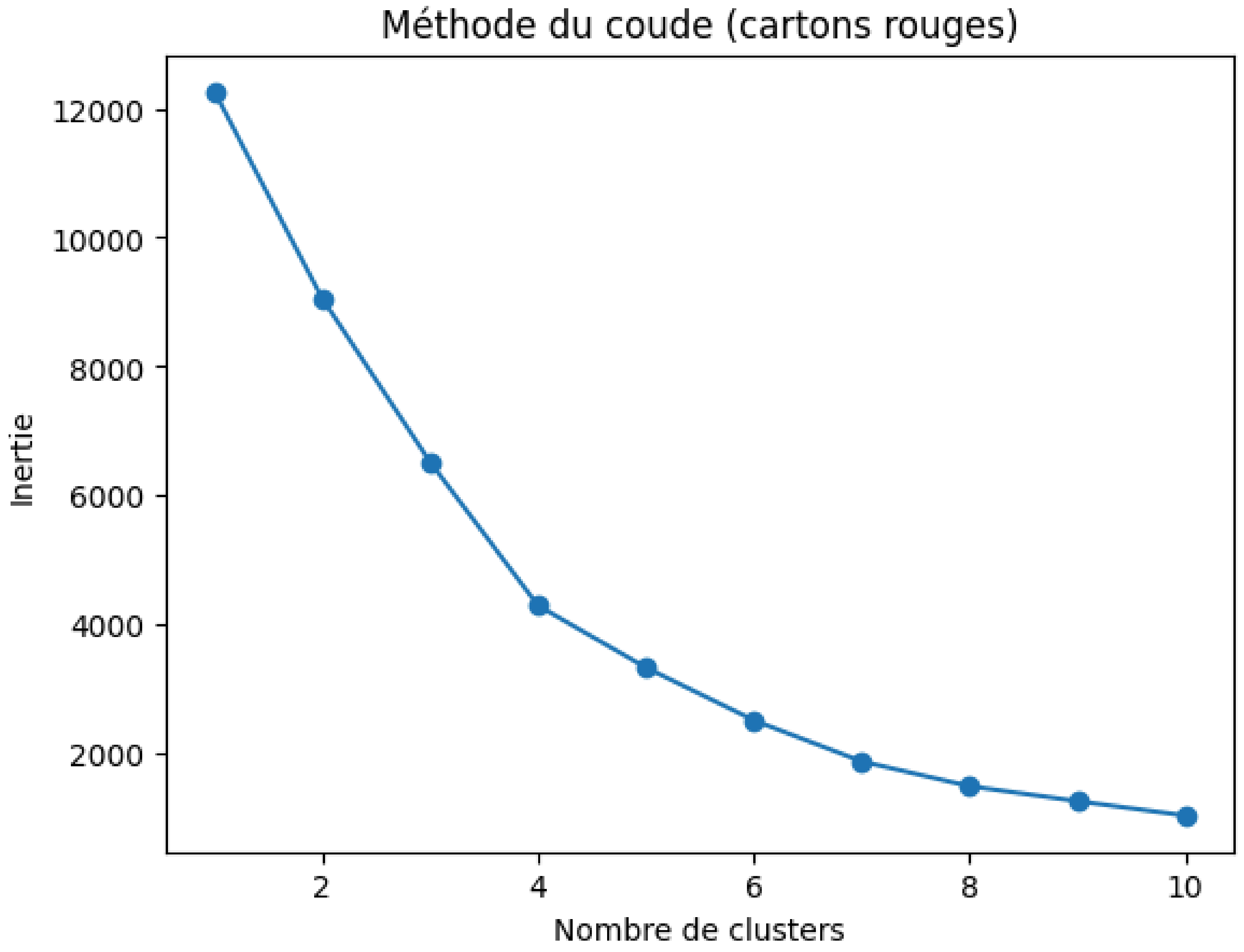
2. On normalise les données

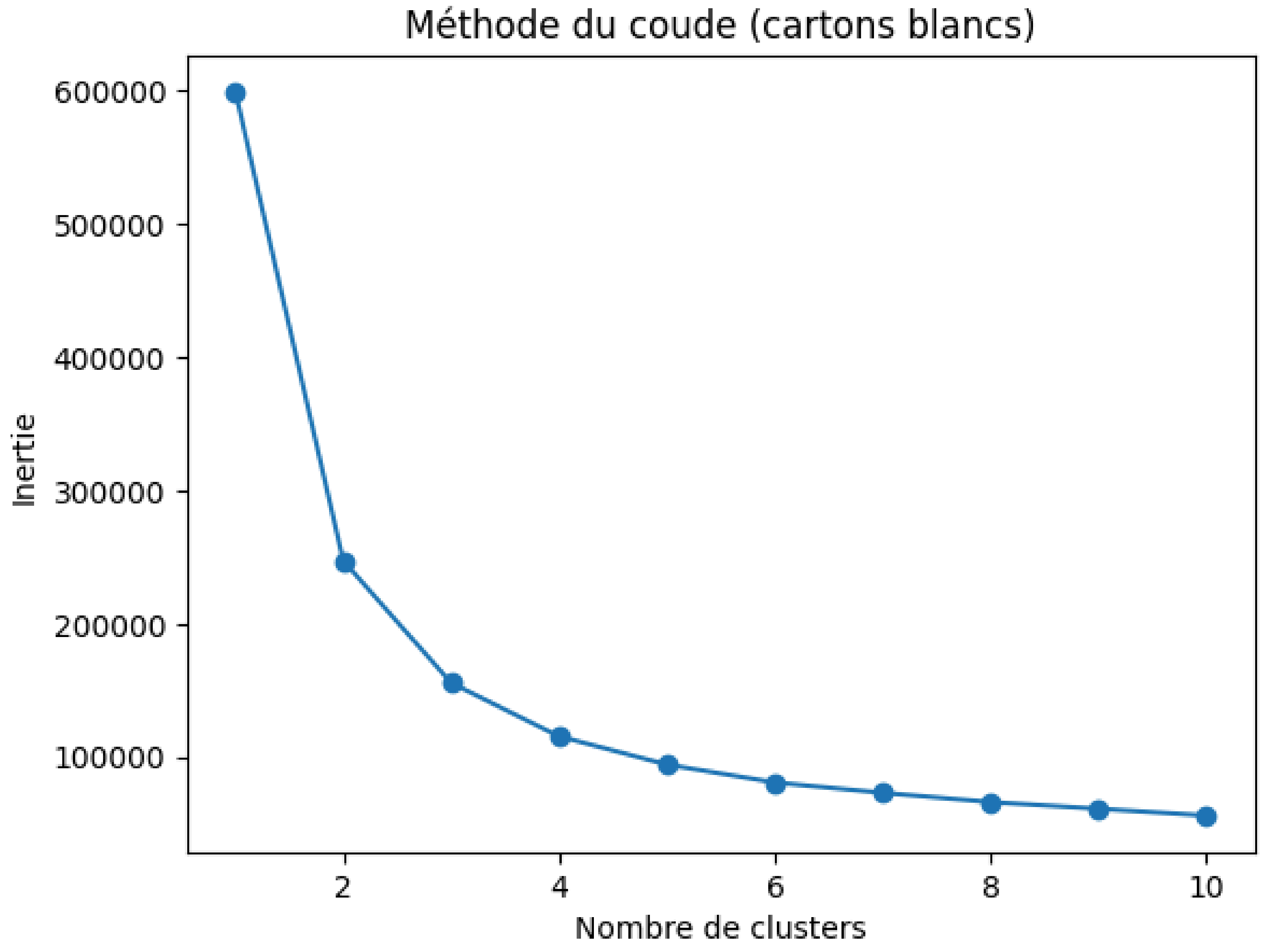
NORMALISATION DES DONNEES POUR QUE TOUTES LES COLONNES SOIENT COMPARABLES EN TERME DE DISTANCE EUCLIDIENNE

```python
from sklearn.preprocessing import StandardScaler

# Liste des colonnes à normaliser
features_to_normalize = ['Name', 'Nationality', 'Gender', 'Discipline', 'AP', 'RP',
        'Card', 'Points', 'Remarks', 'Event Type', 'Month', 'Year', 'Season',
        'experience_dive', 'experience_discipline']

# Normalisation
scaler = StandardScaler()
```

✓ 0.0s

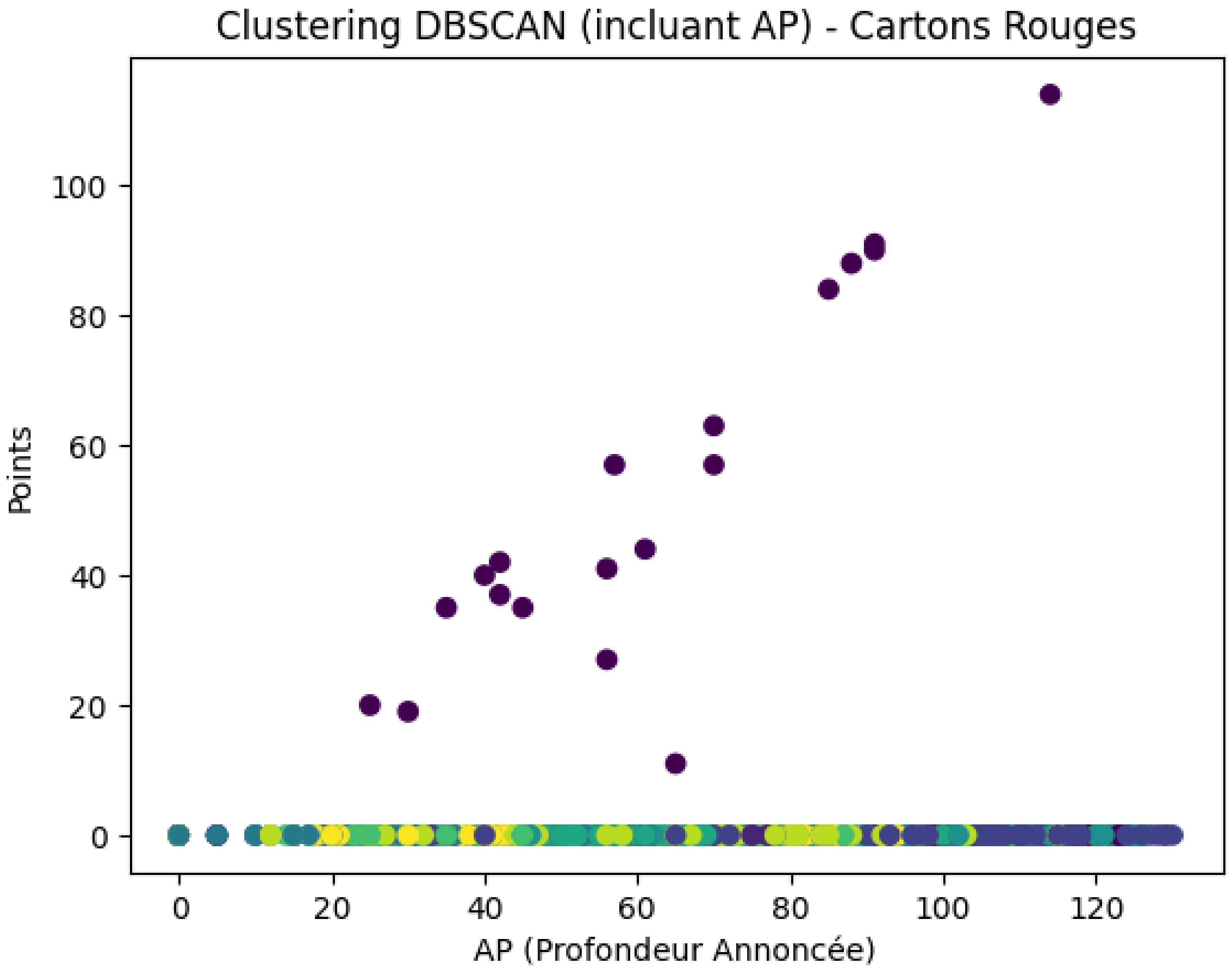
Clustering (visualisation)

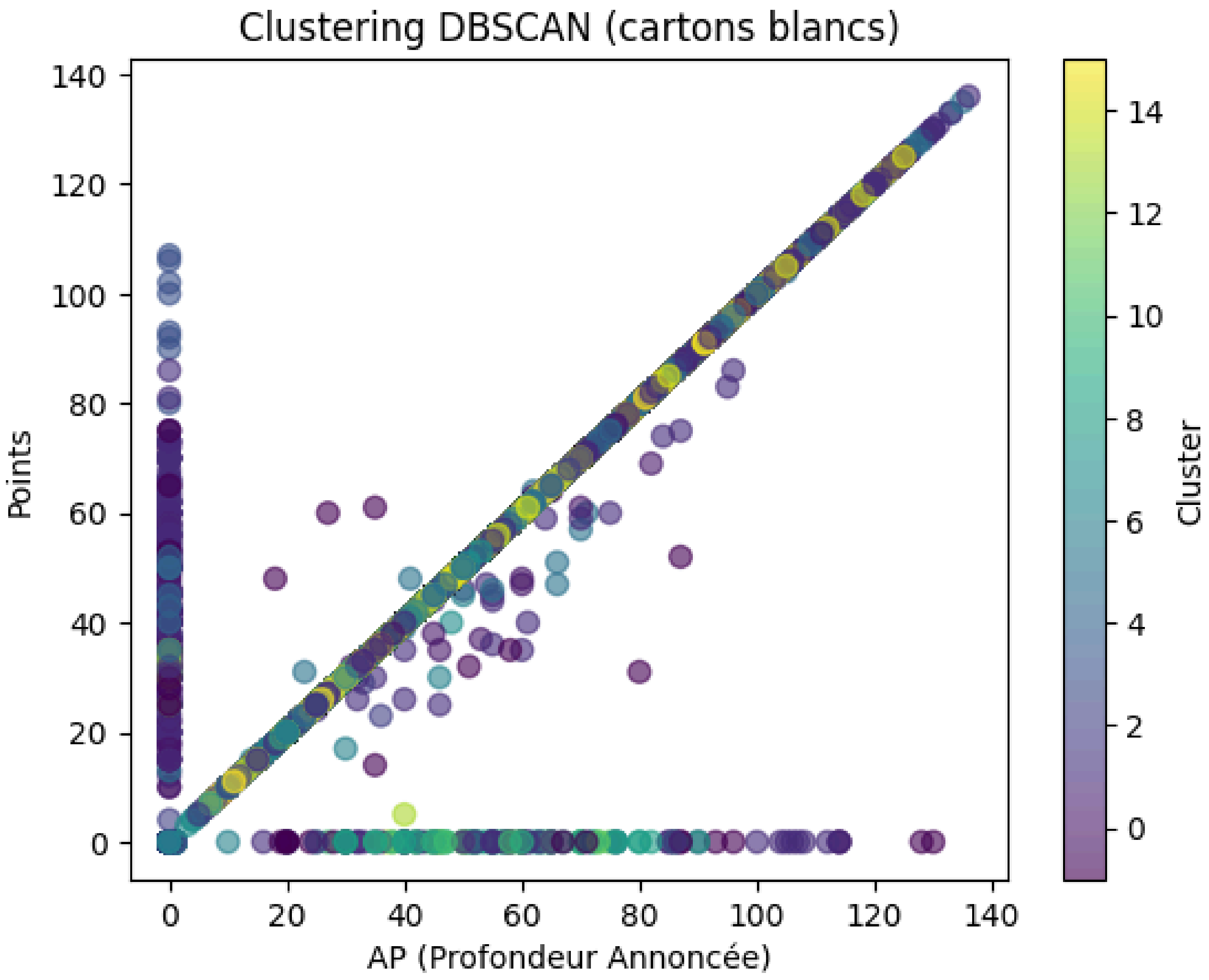




Clustering (visualisation)

inclure AP était plus concluant
quand on a exclut RP





Sources

pandas documentation – pandas 2.2.3 documentation
Matplotlib – Visualization with Python
seaborn: statistical data visualization – seaborn 0.13.2 documentation
scikit-learn: machine learning in Python – scikit-learn 1.5.2 documentation
SVC – scikit-learn 1.5.2 documentation
train_test_split – scikit-learn 1.5.2 documentation
confusion_matrix – scikit-learn 1.5.2 documentation
RandomForestClassifier – scikit-learn 1.5.2 documentation
StandardScaler – scikit-learn 1.5.2 documentation
LabelEncoder – scikit-learn 1.5.2 documentation