In [1]:

```
# Load the Data
import numpy as np
import pandas as pd
```

In [2]:

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
```

In [3]:

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import accuracy_score
```

In [4]:

```
Team1vsTeam2 = pd.read_excel(r'Team1vsTeam2_2019-2023.xlsx')
df = Team1vsTeam2
df
```

Out[4]:

	index	WR	KD	CKPM	GPR	GSPD	EGR	MLR	FB	FT	...
0	T1	0.119447	0.13	-0.02	0.37	0.014	1.1	10.9	-0.01	0.05	...
1	Cloud9	0.196685	0.35	-0.18	0.33	0.045	4.0	15.2	0.00	-0.07	...
2	Gen.G	0.030134	0.25	-0.23	0.27	0.025	7.0	-4.0	0.15	0.12	...
3	Team BDS	-0.236601	-0.55	-0.05	-1.07	-0.089	-9.7	-16.2	0.01	-0.07	...
4	G2 Esports	0.099203	0.01	0.35	0.27	0.030	-0.4	10.4	-0.10	0.08	...
...	...	...	...	...	...	...	...	...	...	...	...
363	J Team	-0.109790	-0.30	-0.20	-0.47	-0.027	-5.8	-5.7	-0.12	-0.09	...
364	Dplus KIA	-0.070833	-0.09	-0.02	-0.10	0.019	-7.0	-0.1	-0.15	-0.03	...
365	Invictus Gaming	0.103757	-0.03	0.38	0.17	0.022	1.5	9.5	-0.23	0.04	...
366	Royal Never Give Up	0.177327	0.30	0.12	0.60	0.019	8.0	11.2	0.03	-0.01	...
367	Fnatic	-0.037646	-0.28	0.23	-0.09	-0.013	-0.8	-3.5	0.01	-0.02	...

368 rows × 21 columns

In [5]:

```
df.IsWin.value_counts()
df.describe()
```

Out[5]:

	WR	KD	CKPM	GPR	GSPD	EGR
count	368.000000	368.000000	368.000000	368.000000	368.000000	368.000000
mean	0.005476	0.011848	0.002174	0.021630	0.001641	0.260598
std	0.112255	0.268764	0.158620	0.527456	0.039215	7.766588
min	-0.293907	-0.940000	-0.430000	-1.640000	-0.125000	-18.500000
25%	-0.078084	-0.170000	-0.110000	-0.330000	-0.024250	-5.825000
50%	0.007489	0.020000	-0.010000	0.065000	0.002500	0.300000
75%	0.089830	0.180000	0.110000	0.360000	0.027000	6.425000
max	0.279326	0.890000	0.430000	1.640000	0.125000	18.500000

In [ ]:

```

```

Dummy Encoding

Encoding scheme to 'index' variable

In [6]:

```
df_enc = pd.get_dummies(df, columns = ['index'])
df_enc2 = pd.get_dummies(df, columns = ['index'], drop_first = True)
```

In [7]:

```
df_enc.info()
df_enc.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 368 entries, 0 to 367
Data columns (total 66 columns):
 #   Column                      Non-Null Count  Dtype  
---  ------                      --------------  -----  
 0   WR                          368 non-null    float64
 1   KD                          368 non-null    float64
 2   CKPM                        368 non-null    float64
 3   GPR                         368 non-null    float64
 4   GSPD                        368 non-null    float64
 5   EGR                         368 non-null    float64
 6   MLR                         368 non-null    float64
 7   FB                          368 non-null    float64
 8   FT                          368 non-null    float64
 9   F3T                         368 non-null    float64
 10  HLD                         368 non-null    float64
 11  FD                          368 non-null    float64
 12  DRG                         368 non-null    float64
 13  ELD                         368 non-null    float64
 14  BN                          368 non-null    float64
 15  LNE                         368 non-null    float64
 16  JNG                         368 non-null    float64
 17  WPM                         368 non-null    float64
 18  CWPM                        368 non-null    float64
 19  IsWin                       368 non-null    int64  
 20  index_100 Thieves           368 non-null    uint8  
 21  index_Bilibili Gaming       368 non-null    uint8  
 22  index_CTBC Flying Oyster    368 non-null    uint8  
 23  index_Cloud9                368 non-null    uint8  
 24  index_Clutch Gaming         368 non-null    uint8  
 25  index_DRX                   368 non-null    uint8  
 26  index_DWG KIA               368 non-null    uint8  
 27  index_DetonatioN FocusMe    368 non-null    uint8  
 28  index_Dplus KIA             368 non-null    uint8  
 29  index_Dplus Kia             368 non-null    uint8  
 30  index_EDward Gaming         368 non-null    uint8  
 31  index_Evil Geniuses         368 non-null    uint8  
 32  index_FlyQuest              368 non-null    uint8  
 33  index_Fnatic                368 non-null    uint8  
 34  index_FunPlus Phoenix       368 non-null    uint8  
 35  index_G2 Esports            368 non-null    uint8  
 36  index_GAM Esports           368 non-null    uint8  
 37  index_GEN.G                 368 non-null    uint8  
 38  index_Gen.G                 368 non-null    uint8  
 39  index_Griffin               368 non-null    uint8  
 40  index_Hanwha Life Esports   368 non-null    uint8  
 41  index_Hong Kong Attitude    368 non-null    uint8  
 42  index_Invictus Gaming       368 non-null    uint8  
 43  index_J Team                368 non-null    uint8  
 44  index_JD Gaming             368 non-null    uint8  
 45  index_KT Rolster            368 non-null    uint8  
 46  index_LGD Gaming            368 non-null    uint8  
 47  index_LNG Esports           368 non-null    uint8  
 48  index_MAD Lions             368 non-null    uint8  
 49  index_Machi Esports         368 non-null    uint8  
 50  index_NRG                   368 non-null    uint8  
 51  index_PSG Talon             368 non-null    uint8
 52  index_RNG                   368 non-null    uint8
 53  index_Rogue                 368 non-null    uint8
 54  index_Royal Never Give Up   368 non-null    uint8
 55  index_SK Telecom T1         368 non-null    uint8
 56  index_Splyce                368 non-null    uint8
 57  index_Suning                368 non-null    uint8
 58  index_T1                    368 non-null    uint8
 59  index_TSM                   368 non-null    uint8
 60  index_Team BDS              368 non-null    uint8
 61  index_Team Liquid           368 non-null    uint8
 62  index_Top Esports           368 non-null    uint8
 63  index_Unicorns of Love.CIS  368 non-null    uint8
 64  index_Weibo Gaming          368 non-null    uint8
 65  index_ahq eSports Club      368 non-null    uint8
dtypes: float64(19), int64(1), uint8(46)
memory usage: 74.2 KB
```

Out[7]:

	WR	KD	CKPM	GPR	GSPD	EGR	MLR	FB	FT	F3T	...	inde
0	0.119447	0.13	-0.02	0.37	0.014	1.1	10.9	-0.01	0.05	0.03	...	
1	0.196685	0.35	-0.18	0.33	0.045	4.0	15.2	0.00	-0.07	-0.03	...	
2	0.030134	0.25	-0.23	0.27	0.025	7.0	-4.0	0.15	0.12	0.06	...	
3	-0.236601	-0.55	-0.05	-1.07	-0.089	-9.7	-16.2	0.01	-0.07	-0.19	...	
4	0.099203	0.01	0.35	0.27	0.030	-0.4	10.4	-0.10	0.08	0.03	...	

5 rows × 66 columns

In [8]:

```
df_enc2.info()
df_enc2.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 368 entries, 0 to 367
Data columns (total 65 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   WR                           368 non-null    float64
 1   KD                           368 non-null    float64
 2   CKPM                         368 non-null    float64
 3   GPR                          368 non-null    float64
 4   GSPD                         368 non-null    float64
 5   EGR                          368 non-null    float64
 6   MLR                          368 non-null    float64
 7   FB                           368 non-null    float64
 8   FT                           368 non-null    float64
 9   F3T                          368 non-null    float64
 10  HLD                          368 non-null    float64
 11  FD                           368 non-null    float64
 12  DRG                          368 non-null    float64
 13  ELD                          368 non-null    float64
 14  BN                           368 non-null    float64
 15  LNE                          368 non-null    float64
 16  JNG                          368 non-null    float64
 17  WPM                          368 non-null    float64
 18  CWPM                         368 non-null    float64
 19  IsWin                        368 non-null    int64
 20  index_Bilibili Gaming        368 non-null    uint8
 21  index_CTBC Flying Oyster     368 non-null    uint8
 22  index_Cloud9                 368 non-null    uint8
 23  index_Clutch Gaming          368 non-null    uint8
 24  index_DRX                    368 non-null    uint8
 25  index_DWG KIA                368 non-null    uint8
 26  index_DetonatioN FocusMe     368 non-null    uint8
 27  index_Dplus KIA              368 non-null    uint8
 28  index_Dplus Kia              368 non-null    uint8
 29  index_EDward Gaming          368 non-null    uint8
 30  index_Evil Geniuses          368 non-null    uint8
 31  index_FlyQuest               368 non-null    uint8
 32  index_Fnatic                 368 non-null    uint8
 33  index_FunPlus Phoenix        368 non-null    uint8
 34  index_G2 Esports             368 non-null    uint8
 35  index_GAM Esports            368 non-null    uint8
 36  index_GEN.G                  368 non-null    uint8
 37  index_Gen.G                  368 non-null    uint8
 38  index_Griffin                368 non-null    uint8
 39  index_Hanwha Life Esports    368 non-null    uint8
 40  index_Hong Kong Attitude     368 non-null    uint8
 41  index_Invictus Gaming        368 non-null    uint8
 42  index_J Team                 368 non-null    uint8
 43  index_JD Gaming              368 non-null    uint8
 44  index_KT Rolster             368 non-null    uint8
 45  index_LGD Gaming             368 non-null    uint8
 46  index_LNG Esports            368 non-null    uint8
 47  index_MAD Lions              368 non-null    uint8
 48  index_Machi Esports          368 non-null    uint8
 49  index_NRG                    368 non-null    uint8
 50  index_PSG Talon              368 non-null    uint8
 51  index_RNG                  368 non-null    uint8
 52  index_Rogue                368 non-null    uint8
 53  index_Royal Never Give Up  368 non-null    uint8
 54  index_SK Telecom T1        368 non-null    uint8
 55  index_Splyce               368 non-null    uint8
 56  index_Suning               368 non-null    uint8
 57  index_T1                   368 non-null    uint8
 58  index_TSM                  368 non-null    uint8
 59  index_Team BDS             368 non-null    uint8
 60  index_Team Liquid          368 non-null    uint8
 61  index_Top Esports          368 non-null    uint8
 62  index_Unicorns of Love.CIS 368 non-null    uint8
 63  index_Weibo Gaming         368 non-null    uint8
 64  index_ahq eSports Club     368 non-null    uint8
dtypes: float64(19), int64(1), uint8(45)
memory usage: 73.8 KB
```

Out[8]:

	WR	KD	CKPM	GPR	GSPD	EGR	MLR	FB	FT	F3T	...	inde
0	0.119447	0.13	-0.02	0.37	0.014	1.1	10.9	-0.01	0.05	0.03	...	
1	0.196685	0.35	-0.18	0.33	0.045	4.0	15.2	0.00	-0.07	-0.03	...	
2	0.030134	0.25	-0.23	0.27	0.025	7.0	-4.0	0.15	0.12	0.06	...	
3	-0.236601	-0.55	-0.05	-1.07	-0.089	-9.7	-16.2	0.01	-0.07	-0.19	...	
4	0.099203	0.01	0.35	0.27	0.030	-0.4	10.4	-0.10	0.08	0.03	...	

5 rows × 65 columns

In [9]:

```
X = df_enc2[['WR', 'KD', 'CKPM','GPR','GSPD','EGR','MLR','FB','FT','F3T','HL
Y = df_enc2[['IsWin']]
```

In [10]:

```
model = DecisionTreeClassifier()
model.fit(X,Y)

feature_importance = model.feature_importances_

feature_importance_df_enc2 = pd.DataFrame({'Feature': X.columns, 'Importance

feature_importance_df_enc2 = feature_importance_df_enc2.sort_values(by='Impo

feature_importance_df_enc2.head(25)
```

Out[10]:

	Feature	Importance
17	WPM	0.201932
13	ELD	0.094971
18	CWPM	0.080782
2	CKPM	0.080069
7	FB	0.076451
1	KD	0.065964
0	WR	0.065082
11	FD	0.052038
16	JNG	0.050143
4	GSPD	0.050075
10	HLD	0.047507
6	MLR	0.035129
14	BN	0.031032
8	FT	0.025621
5	EGR	0.019369
12	DRG	0.013958
15	LNE	0.004968
9	F3T	0.004907
3	GPR	0.000000

In [24]:

```
features_to_remove = ['F3T', 'LNE','GPR']
df1=df.drop(features_to_remove, axis=1)
df1
```

Out[24]:

	index	WR	KD	CKPM	GSPD	EGR	MLR	FB	FT	HLD	F
0	T1	0.119447	0.13	-0.02	0.014	1.1	10.9	-0.01	0.05	0.02	0.0
1	Cloud9	0.196685	0.35	-0.18	0.045	4.0	15.2	0.00	-0.07	0.12	0.1
2	Gen.G	0.030134	0.25	-0.23	0.025	7.0	-4.0	0.15	0.12	-0.05	-0.0
3	Team BDS	-0.236601	-0.55	-0.05	-0.089	-9.7	-16.2	0.01	-0.07	-0.03	0.0
4	G2 Esports	0.099203	0.01	0.35	0.030	-0.4	10.4	-0.10	0.08	0.10	-0.0
...	...	...	...	...	...	...	...	...	...	...	
363	J Team	-0.109790	-0.30	-0.20	-0.027	-5.8	-5.7	-0.12	-0.09	-0.05	-0.1
364	Dplus KIA	-0.070833	-0.09	-0.02	0.019	-7.0	-0.1	-0.15	-0.03	0.04	-0.1
365	Invictus Gaming	0.103757	-0.03	0.38	0.022	1.5	9.5	-0.23	0.04	-0.10	-0.1
366	Royal Never Give Up	0.177327	0.30	0.12	0.019	8.0	11.2	0.03	-0.01	-0.10	0.1
367	Fnatic	-0.037646	-0.28	0.23	-0.013	-0.8	-3.5	0.01	-0.02	-0.15	-0.0

368 rows × 18 columns

In [25]:

```
X = df1[['WR','KD','CKPM','GSPD','EGR','MLR','FT','FB','HLD','FD','DRG','ELD

Y = df1[['IsWin']]
```

In [26]:

```
# Split Data

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, ran
X_train.shape, X_test.shape
```

Out[26]:

```
((257, 16), (111, 16))
```

In [27]:

```
# BASELINE
negative = np.sum(Y_train == 0)
positive = np.sum(Y_train == 1)
print(pd.Series({'0': negative, '1': positive}))
```

```
0    IsWin    128
dtype: int64
1    IsWin    129
dtype: int64
dtype: object
```

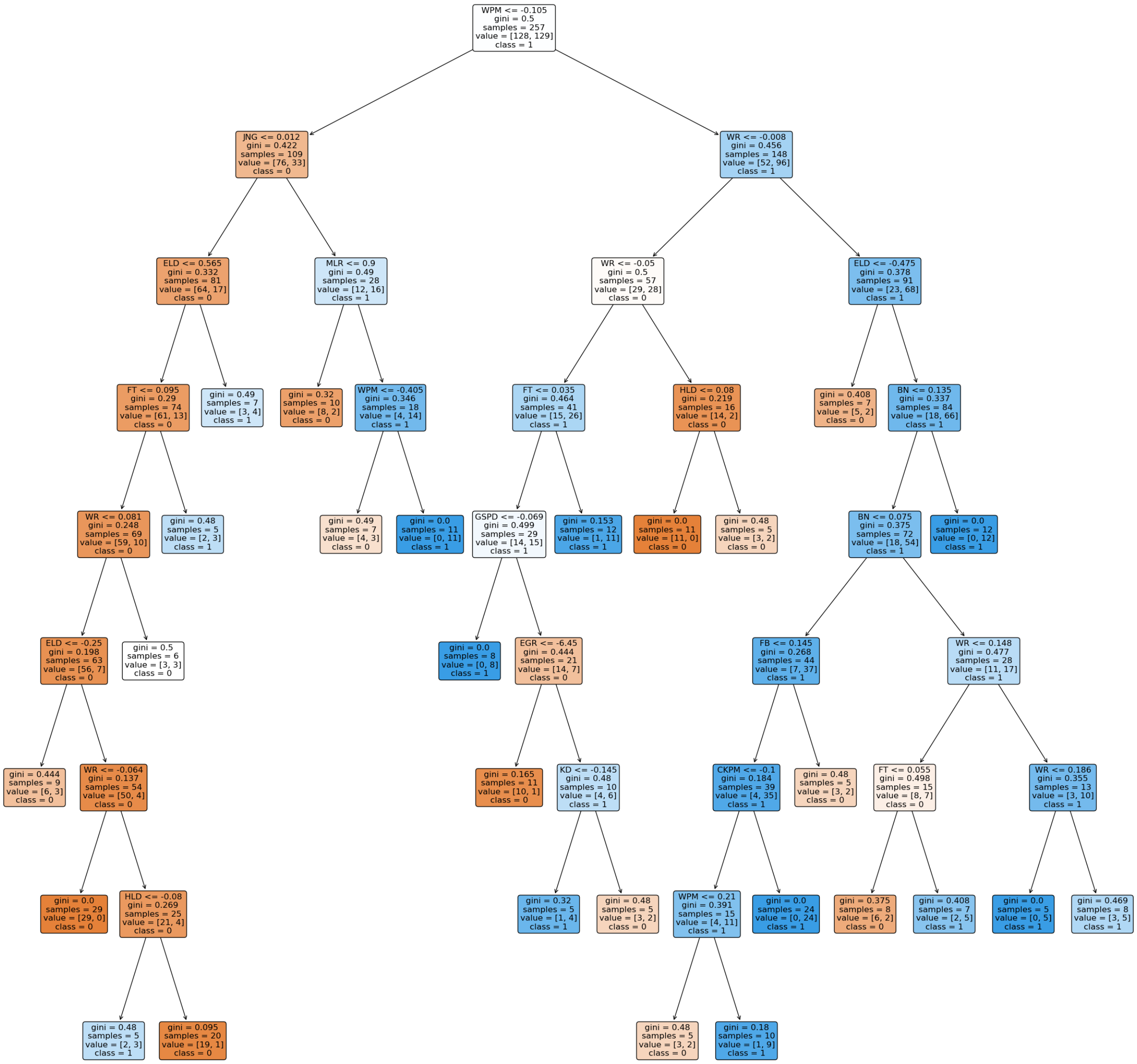
In [28]:

```
dtc = DecisionTreeClassifier(min_samples_leaf=5,ccp_alpha=0.003,random_state

dtc = dtc.fit(X_train, Y_train)
```

In [29]:
```
print('Node count =', dtc.tree_.node_count)
plt.figure(figsize=(30,30))
plot_tree(dtc,
          feature_names=list(X_train.columns),
          class_names=['0','1'],
          filled=True,
          impurity=True,
          rounded=True,
          fontsize=12)
plt.show()
```

```
Node count = 53
```



In [30]:
```
# MAKE PREDICTIONS
Y_pred = dtc.predict(X_test)
cm = confusion_matrix(Y_test, Y_pred)
Y_proba = dtc.predict_proba(X_test)
print ("Confusion Matrix : \n", cm)
print('Precision:',precision_score(Y_test, Y_pred))
print('Recall:',recall_score(Y_test, Y_pred))
print('Accuracy:',accuracy_score(Y_test, Y_pred))
display(Y_proba)
```

```
Confusion Matrix :
 [[24 25]
 [27 35]]
Precision: 0.5833333333333334
Recall: 0.5645161290322581
Accuracy: 0.5315315315315315
```

```
array([[0.        , 1.        ],
       [0.42857143, 0.57142857],
       [0.2       , 0.8       ],
       [1.        , 0.        ],
       [0.1       , 0.9       ],
       [0.90909091, 0.09090909],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.375     , 0.625     ],
       [0.6       , 0.4       ],
       [0.6       , 0.4       ],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.95      , 0.05      ],
       [0.        , 1.        ],
       [0.75      , 0.25      ],
       [0.        , 1.        ],
       [0.6       , 0.4       ],
       [0.2       , 0.8       ],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.1       , 0.9       ],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.28571429, 0.71428571],
       [0.2       , 0.8       ],
       [0.75      , 0.25      ],
       [1.        , 0.        ],
       [0.90909091, 0.09090909],
       [0.08333333, 0.91666667],
       [1.        , 0.        ],
       [0.6       , 0.4       ],
       [0.4       , 0.6       ],
       [0.95      , 0.05      ],
       [1.        , 0.        ],
       [1.        , 0.        ],
       [0.71428571, 0.28571429],
       [0.        , 1.        ],
       [0.2       , 0.8       ],
       [0.28571429, 0.71428571],
       [0.6       , 0.4       ],
       [0.375     , 0.625     ],
       [0.90909091, 0.09090909],
       [0.75      , 0.25      ],
       [0.        , 1.        ],
       [0.2       , 0.8       ],
       [0.28571429, 0.71428571],
       [0.8       , 0.2       ],
       [0.2       , 0.8       ],
       [1.        , 0.        ],
       [0.95      , 0.05      ],
       [0.08333333, 0.91666667],
       [1.        , 0.        ],
       [0.1       , 0.9       ],
       [0.        , 1.        ],
       [0.5       , 0.5       ],
       [0.42857143, 0.57142857],
       [0.6       , 0.4       ],
       [0.        , 1.        ],
       [0.2       , 0.8       ],
       [0.4       , 0.6       ],
       [0.        , 1.        ],
       [1.        , 0.        ],
       [0.        , 1.        ],
       [0.90909091, 0.09090909],
       [0.        , 1.        ],
       [0.90909091, 0.09090909],
       [0.57142857, 0.42857143],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.95      , 0.05      ],
       [1.        , 0.        ],
       [0.66666667, 0.33333333],
       [0.        , 1.        ],
       [0.1       , 0.9       ],
       [0.71428571, 0.28571429],
       [1.        , 0.        ],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.4       , 0.6       ],
       [0.2       , 0.8       ],
       [0.75      , 0.25      ],
       [1.        , 0.        ],
       [0.90909091, 0.09090909],
       [0.        , 1.        ],
       [0.8       , 0.2       ],
       [1.        , 0.        ],
       [0.2       , 0.8       ],
       [0.75      , 0.25      ],
       [1.        , 0.        ],
       [0.        , 1.        ],
       [0.6       , 0.4       ],
       [0.75      , 0.25      ],
       [0.        , 1.        ],
       [0.08333333, 0.91666667],
       [1.        , 0.        ],
       [0.        , 1.        ],
       [0.6       , 0.4       ],
       [0.1       , 0.9       ],
       [0.6       , 0.4       ],
       [0.6       , 0.4       ],
       [1.        , 0.        ],
       [0.        , 1.        ],
       [0.        , 1.        ],
       [0.75      , 0.25      ],
       [0.2       , 0.8       ],
       [0.71428571, 0.28571429],
       [0.66666667, 0.33333333],
       [0.        , 1.        ]])
```

In [ ]:

```

```

Predict the Winner

8-in-4

In [31]:

```
year2023StatForWorldsTeam = pd.read_excel(r'year2023StatForWorldsTeam(1).xls
year2023StatForWorldsTeam.set_index("team", inplace = True)
year2023StatForWorldsTeam.rename(columns={"win rate": "WR"}, inplace=True)

selected_columns = year2023StatForWorldsTeam[['WR','KD','CKPM','GSPD','EGR',
selected_columns
```

Out[31]:

	WR	KD	CKPM	GSPD	EGR	MLR	FT	FB	HLD	FD	DRG
team											
Gen.G	0.726562	1.53	0.75	0.075	60.8	11.9	0.66	0.61	0.53	0.44	0.59
T1	0.611511	1.10	0.77	0.040	59.8	1.4	0.72	0.51	0.61	0.61	0.58
KT Rolster	0.694215	1.52	0.69	0.045	57.5	11.9	0.61	0.55	0.45	0.55	0.60
Dplus KIA	0.606061	1.35	0.67	0.039	60.5	0.1	0.53	0.58	0.54	0.62	0.62
JD Gaming	0.777778	1.58	0.86	0.083	58.5	21.5	0.55	0.52	0.43	0.50	0.60
Bilibili Gaming	0.641379	1.21	0.91	0.053	58.7	6.9	0.52	0.57	0.48	0.59	0.56
LNG Esports	0.651786	1.39	0.81	0.045	61.4	-1.4	0.40	0.61	0.60	0.60	0.54
Weibo Gaming	0.607843	1.18	0.82	0.044	52.3	-18.9	0.67	0.45	0.50	0.67	0.51
G2 Esports	0.705263	1.36	1.02	0.069	60.1	10.5	0.61	0.48	0.64	0.60	0.59
Fnatic	0.492754	0.97	0.92	-0.009	46.6	1.9	0.41	0.50	0.36	0.47	0.50
MAD Lions	0.483871	0.93	0.98	-0.002	52.2	-3.3	0.53	0.49	0.44	0.57	0.53
Team BDS	0.541176	1.03	0.81	-0.006	48.8	5.3	0.48	0.53	0.40	0.55	0.56
NRG	0.560976	1.14	0.84	-0.007	43.1	13.0	0.44	0.44	0.49	0.51	0.51
Cloud9	0.680556	1.28	0.80	0.043	56.2	11.9	0.46	0.49	0.56	0.67	0.59
Team Liquid	0.492063	0.97	0.79	0.026	58.7	-9.5	0.67	0.52	0.59	0.56	0.55
GAM Esports	0.696429	1.28	0.98	0.050	53.8	15.9	0.54	0.46	0.58	0.49	0.59

In [32]:

```
# GenG in Blue
GenG_vs_BLG = selected_columns.loc[["Gen.G"]].sub(selected_columns.loc[["Bil
# BlG in Blue
BLG_vs_GenG = selected_columns.loc[["Bilibili Gaming"]].sub(selected_columns
```

In [33]:

```
# Predict the Winning Rate
GenG_WR = dtc.predict_proba(GenG_vs_BLG)
BLG_WR = dtc.predict_proba(BLG_vs_GenG)
print('The Winning Rate for Gen.G in the blue side is ', GenG_WR,
      '; the Winning Rate for BLG in the blue side is ', BLG_WR)
```

```
The Winning Rate for Gen.G in the blue side is  [[0. 1.]] ; the Winning Rate
for BLG in the blue side is  [[0.2 0.8]]
```

GenG win

In [37]:

```
# NRG in Blue
NRG_vs_WBG = selected_columns.loc[["NRG"]].sub(selected_columns.loc[["Weibo
# WBG in Blue
WBG_vs_NRG = selected_columns.loc[["Weibo Gaming"]].sub(selected_columns.loc

# Predict the Winning Rate
NRG_WR = dtc.predict_proba(NRG_vs_WBG)
WBG_WR = dtc.predict_proba(WBG_vs_NRG)


print('The Winning Rate for NRG in the blue side is ', NRG_WR, '; the Winnir
```

```
The Winning Rate for NRG in the blue side is  [[1. 0.]] ; the Winning Rate f
or WBG in the blue side is  [[0.8 0.2]]
```

WBG win

In [38]:

```
# JDG in Blue
JDG_vs_KT = selected_columns.loc[["JD Gaming"]].sub(selected_columns.loc[["K
# KT in Blue
KT_vs_JDG = selected_columns.loc[["KT Rolster"]].sub(selected_columns.loc[["

# Predict the Winning Rate
JDG_WR = dtc.predict(JDG_vs_KT)
KT_WR = dtc.predict(KT_vs_JDG)

print('The Winning Rate for JDG in the blue side is ', JDG_WR, '; the Winnir
```

```
The Winning Rate for JDG in the blue side is  [0] ; the Winning Rate for KT
in the blue side is  [1]
```

KT win

In [41]:

```
# LNG in Blue
LNG_vs_T1 = selected_columns.loc[["LNG Esports"]].sub(selected_columns.loc[[
# T1 in Blue
T1_vs_LNG = selected_columns.loc[["T1"]].sub(selected_columns.loc[["LNG Espo

# Predict the Winning Rate
LNG_WR = dtc.predict_proba(LNG_vs_T1)
```

```
T1_WR = dtc.predict_proba(T1_vs_LNG)
print('The Winning Rate for LNG in the blue side is ', LNG_WR, '; the Winnir
```

```
The Winning Rate for LNG in the blue side is  [[0.66666667 0.33333333]] ; th
e Winning Rate for T1 in the blue side is  [[1. 0.]]
```

LNG win

In [ ]:

```

```

Semifinals

In [43]:

```
# GenG in Blue
GenG_vs_WBG = selected_columns.loc[["Gen.G"]].sub(selected_columns.loc[["Wei
# WBG in Blue
WBG_vs_GenG = selected_columns.loc[["Weibo Gaming"]].sub(selected_columns.lc
# Predict the Winning Rate
GenG_WR = dtc.predict_proba(GenG_vs_WBG)
WBG_WR = dtc.predict_proba(WBG_vs_GenG)
print('The Winning Rate for GenG in the blue side is ', GenG_WR, '; the Winr
```

```
The Winning Rate for GenG in the blue side is  [[0.6 0.4]] ; the Winning Rat
e for WBG in the blue side is  [[0.90909091 0.09090909]]
```

GenG win

In [44]:

```
# KT in Blue
KT_vs_JDG = selected_columns.loc[["KT Rolster"]].sub(selected_columns.loc[["
# LNG in Blue
LNG_vs_T1 = selected_columns.loc[["LNG Esports"]].sub(selected_columns.loc[[

# Predict the Winning Rate
KT_WR = dtc.predict(KT_vs_JDG)
LNG_WR = dtc.predict(LNG_vs_T1)
print('The Winning Rate for KT in the blue side is ', KT_WR,
      '; the Winning Rate for LNG in the blue side is ', LNG_WR)
```

```
The Winning Rate for KT in the blue side is  [1] ; the Winning Rate for LNG
in the blue side is  [0]
```

KT win

In [ ]:

```

```

Final

In [47]:

```
# GenG  in Blue
GenG_vs_KT = selected_columns.loc[["Gen.G"]].sub(selected_columns.loc[["KT R
# KT in Blue
KT_vs_GenG = selected_columns.loc[["KT Rolster"]].sub(selected_columns.loc[[

# Predict the Winning Rate
GenG_WR = dtc.predict(GenG_vs_KT)
KT_WR = dtc.predict(KT_vs_GenG)
```

```
print('The Winning Rate for GenG in the blue side is ', GenG_WR,
      '; the Winning Rate for KT in the blue side is ', KT_WR)
```

```
The Winning Rate for GenG in the blue side is  [1] ; the Winning Rate for KT
in the blue side is  [0]
```

By CART model, the winner is GenG

```
In [ ]:
```

```
In [ ]:
```