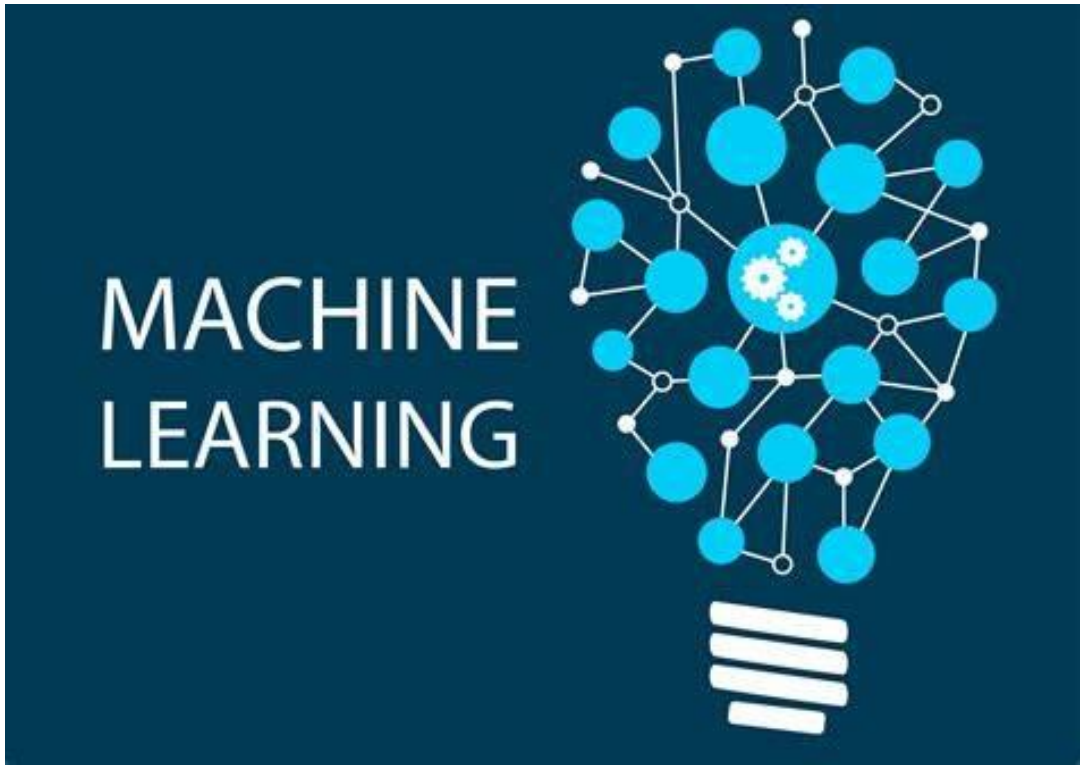
Position for NBA Players

YIWEI YU

2022 OCT



Purpose

By analyzing player data for the past three seasons (2019 – 2022) in the NBA, some models that can predict player positions will be created.

Models will be created by K-Nearest Neighbors (KNN), Support Vector Machine (SVM) and Random Forest.

Such models can provide support is provided for teams to find players in suitable positions. These models also help players recognize their own specific skills and find more suitable training methods.

Data Source

https://www.basketball-reference.com/leagues/NBA_2022_per_game.html

https://www.basketball-reference.com/leagues/NBA_2021_per_game.html

https://www.basketball-reference.com/leagues/NBA_2020_per_game.html

2168 rows, 32 columns

Clean-up Data

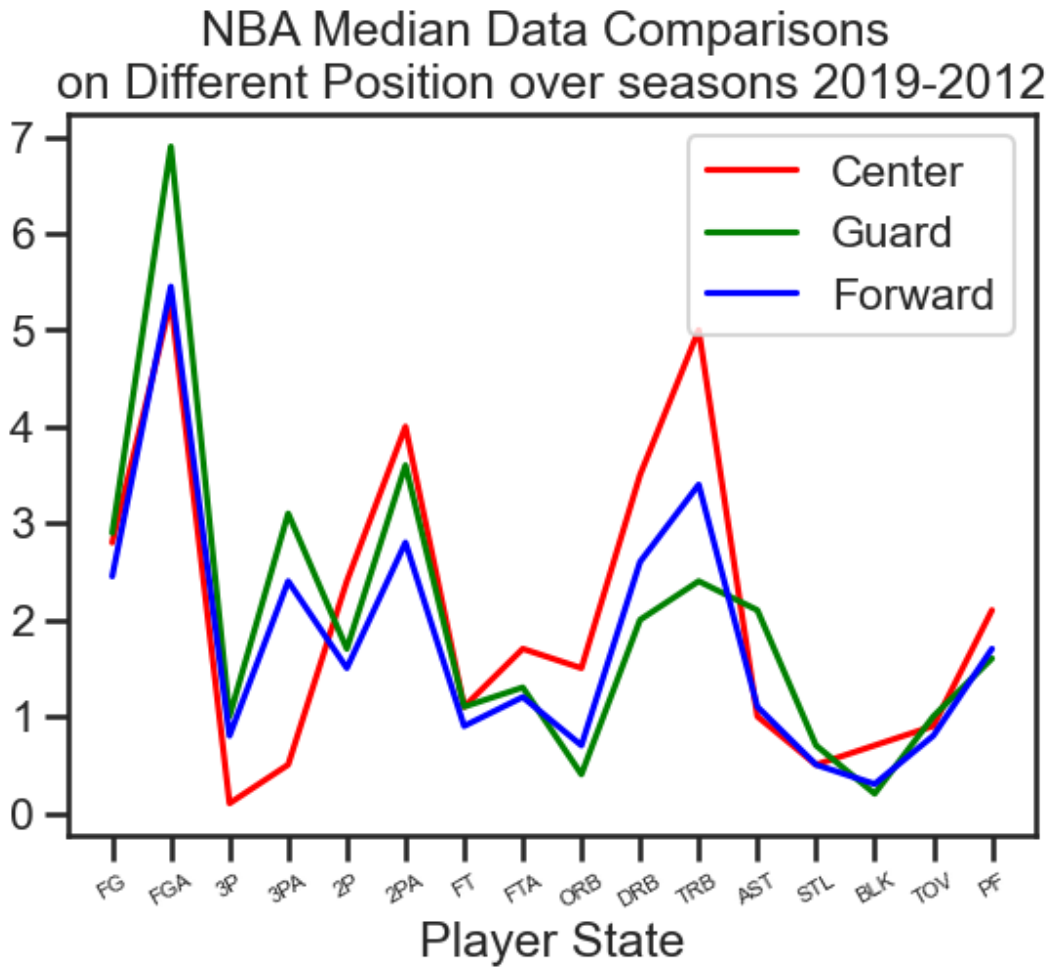
1. Remove the players whose time / game < 5 minutes, to avoid playing time is too little to form the technical statistics of the corresponding position as outlines.

2. Merge positions

Center (C)	Guard (G)	Forward (F)
C, PF-C, SF-C	SG, PG, SF-SG, SG-PG, PG-SG	PF, SF, PF-SF, SF-PF, C-PF, SG-PG-SF

3. Choose necessary features

Target →	Pos – Position	
	FG -- Field Goals Per Game	ORB -- Offensive Rebounds Per Game
	FGA -- Field Goal Attempts Per Game	DRB -- Defensive Rebounds Per Game
	3P -- 3-Point Field Goals Per Game	TRB -- Total Rebounds Per Game
	3PA -- 3-Point Field Goal Attempts Per Game	AST -- Assists Per Game
	2P -- 2-Point Field Goals Per Game	STL -- Steals Per Game
	2PA -- 2-Point Field Goal Attempts Per Game	BLK -- Blocks Per Game
	FT -- Free Throws Per Game	TOV -- Turnovers Per Game
	FTA -- Free Throw Attempts Per Game	PF -- Personal Fouls Per Game



Comparison on Different Position

1. The technical characteristics of the Center are obvious: few three-point shots, high rebounding.

2. Guard own more Assists Per Game.

3. Most of the technical features of F and G overlap and are difficult to distinguish directly.

(There are too many talented and very good athletes in the NBA, and their statistics are often far beyond the average athlete. Therefore, in order to reflect the generality, only the median is used for comparison)

Train / Test Datasets

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

# Set up X and y variables
y, X = df['Pos'], df.drop(columns='Pos')
# Split the data into training and test samples
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4, random_state=12345)

# encoder the labels
enc = LabelEncoder()
enc = enc.fit(['C', 'F', 'G'])

y_train = enc.transform(y_train)
y_test = enc.transform(y_test)
```

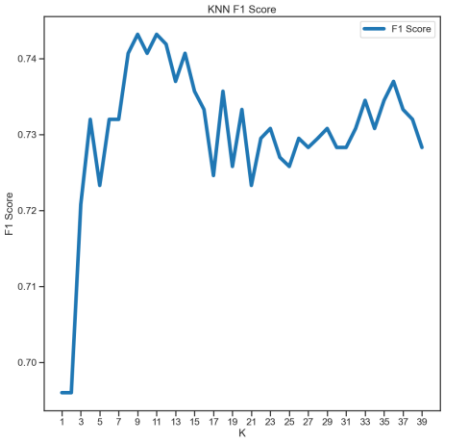
KNN Model

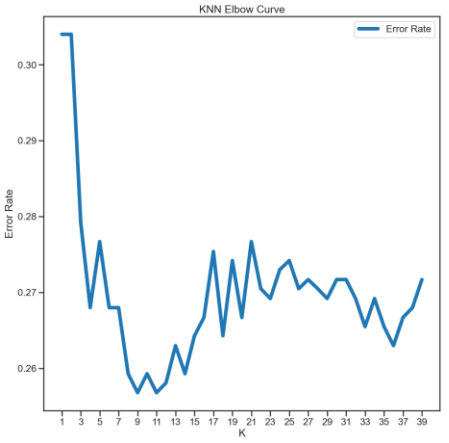
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report, f1_score
```

KNN Elbow

```
max_k = 40
f1_scores = list()
error_rates = list() # 1-accuracy

for k in range(1, max_k):
    knn = KNeighborsClassifier(n_neighbors=k, weights='distance')
    knn = knn.fit(X_train, y_train)

    y_pred = knn.predict(X_test)
    f1 = f1_score(y_pred, y_test, average='micro')
    f1_scores.append((k, round(f1_score(y_test, y_pred, average='micro'), 4)))
    error = 1-round(accuracy_score(y_test, y_pred), 4)
    error_rates.append((k, error))

f1_results = pd.DataFrame(f1_scores, columns=['K', 'F1 Score'])
error_results = pd.DataFrame(error_rates, columns=['K', 'Error Rate'])
```

Take K=11 as the best value

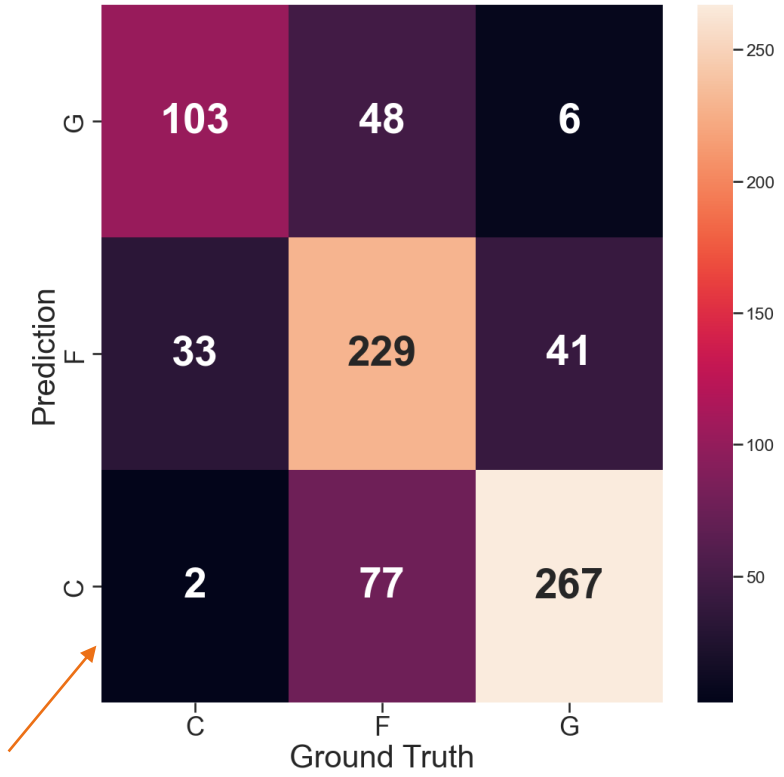




Prediction by KNN Model(K=11)

```
# Take K = 11 as the best value
knn = KNeighborsClassifier(n_neighbors=11, weights='distance')
knn = knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
# Preciision, recall, f-score from the multi-class support function
print(classification_report(y_test, y_pred))
print('Accuracy score: ', round(accuracy_score(y_test, y_pred), 2))
print('F1 Score: ', round(f1_score(y_test, y_pred, average='micro'), 2))
```

```
              precision    recall  f1-score   support

           0       0.75      0.66      0.70       157
           1       0.65      0.76      0.70       303
           2       0.85      0.77      0.81       346

    accuracy                           0.74       806
   macro avg       0.75      0.73      0.73       806
weighted avg       0.75      0.74      0.75       806

Accuracy score:  0.74
F1 Score:  0.74
```



The technical characteristics of the Center are obvious

Manual Check

Suppose same player on different year will get same prediction

```
# Take some examples
df_c = df_org[df_org['Pos'] == 'C'].sample(3, random_state=12345)
df_g = df_org[df_org['Pos'] == 'G'].sample(3, random_state=54321)
df_f = df_org[df_org['Pos'] == 'F'].sample(3, random_state=11223)
```

```
df_c['Player'].unique().tolist()
```

```
['Bruno Fernando', 'Serge Ibaka', 'Gorgui Dieng']
```

```
df_g['Player'].unique().tolist()
```

```
['Wesley Matthews', 'Derrick Rose', 'Mason Jones']
```

```
df_f['Player'].unique().tolist()
```

```
['Lauri Markkanen', 'Taurean Prince', 'Jaren Jackson Jr.']
```

```
# Get all data for the players whose position is Center in the examples over the years 2019-2012
df_c_more = df_org[df_org['Player'].isin([s for s in df_c['Player']])]
```

```
# Get all data for the players whose position is Guard in the examples over the years 2019-2012
df_g_more = df_org[df_org['Player'].isin([s for s in df_g['Player']])]
```

```
# Get all data for the players whose position is Forward in the examples over the years 2019-2012
df_f_more = df_org[df_org['Player'].isin([s for s in df_f['Player']])]
```

Predict Position Center

```
x_test = df_c_more.drop(columns=['Pos', 'Player', 'Rk', 'Season'])
y_pred = knn.predict(x_test)

pred = [labels[i] for i in y_pred]
result = df c more.copy().reset index().drop('index', axis=1)
result.insert(0, 'Prediction', pred)
```

Accurate = 100%

	Prediction	Pos	Player	Rk	Season	FG	FGA	3P	3PA	2P	...	FT	FTA	ORB	DRB	TRB	AST	STL	BLK	TOV	PF
0	C	C	Gorgui Dieng	172	2019_20	2.6	5.8	0.9	2.4	1.8	...	1.2	1.6	1.4	4.2	5.6	1.2	0.8	0.9	1.0	2.0
1	C	C	Gorgui Dieng	173	2019_20	2.7	6.0	1.0	2.6	1.7	...	1.0	1.3	1.6	4.0	5.6	1.3	0.8	0.9	1.0	1.9
2	C	C	Gorgui Dieng	174	2019_20	2.5	5.1	0.5	1.9	2.0	...	1.8	2.5	1.1	4.7	5.8	0.9	0.8	1.0	0.9	2.1
3	C	C	Bruno Fernando	206	2019_20	1.8	3.6	0.1	0.7	1.8	...	0.5	0.9	1.2	2.3	3.5	0.9	0.3	0.3	0.8	1.9
4	C	C	Serge Ibaka	304	2019_20	6.2	12.2	1.3	3.3	5.0	...	1.6	2.3	2.1	6.1	8.2	1.4	0.5	0.8	2.0	2.8
5	C	C	Gorgui Dieng	817	2020_21	2.2	4.3	0.8	1.8	1.4	...	1.5	1.8	1.1	2.6	3.7	1.3	0.7	0.4	0.8	1.5
6	C	C	Gorgui Dieng	818	2020_21	2.5	4.9	1.0	2.2	1.5	...	1.7	2.0	1.3	3.2	4.5	1.3	0.8	0.6	1.0	1.7
7	C	C	Gorgui Dieng	819	2020_21	1.8	3.4	0.4	1.4	1.4	...	1.3	1.5	0.8	1.9	2.6	1.2	0.6	0.1	0.6	1.1
8	C	C	Bruno Fernando	848	2020_21	0.5	1.3	0.0	0.1	0.5	...	0.5	0.7	0.5	1.9	2.4	0.3	0.1	0.1	0.6	0.7
9	C	C	Serge Ibaka	963	2020_21	4.5	8.9	1.0	2.8	3.6	...	1.0	1.3	1.8	4.9	6.7	1.8	0.2	1.1	1.1	1.9
10	C	C	Gorgui Dieng	1527	2021_22	1.2	2.5	0.7	1.5	0.5	...	0.4	0.6	0.7	2.0	2.8	0.8	0.3	0.3	0.5	1.2
11	C	C	Bruno Fernando	1572	2021_22	1.2	1.8	0.0	0.1	1.2	...	0.5	0.8	0.6	1.2	1.8	0.2	0.0	0.4	0.5	0.8
12	C	C	Bruno Fernando	1574	2021_22	2.9	4.1	0.0	0.1	2.9	...	1.1	1.9	1.1	2.9	4.0	0.3	0.1	0.8	0.9	1.8
13	C	C	Serge Ibaka	1706	2021_22	2.7	5.5	0.7	1.8	2.1	...	0.6	0.8	1.1	3.5	4.6	0.9	0.2	0.6	0.8	1.9
14	C	C	Serge Ibaka	1707	2021_22	2.7	5.5	0.7	1.8	2.0	...	0.6	0.8	1.1	3.1	4.3	1.0	0.2	0.7	0.9	1.6
15	C	C	Serge Ibaka	1708	2021_22	2.8	5.5	0.7	1.9	2.2	...	0.6	0.8	1.2	4.1	5.3	0.7	0.2	0.4	0.5	2.4

16 rows × 21 columns

Predict position Forward

Very interesting that model predict Jaren Jackson Jr. in season (2020-21) is C and it's correct!

He is on C in seasons (2019-20 and 2020-21) and changed to F on season 2021-22.

Model can capture his technical characteristic change. (although model delivers wrong value in 2019-20).

	Prediction	Pos	Player	Rk	Season	FG	FGA	3P	3PA	2P	...	FT	FTA	ORB	DRB	TRB	AST	STL	BLK	TOV	PF
0	F	C	Jaren Jackson Jr.	313	2019_20	6.2	13.2	2.5	6.5	3.6	...	2.4	3.3	1.0	3.6	4.6	1.4	0.7	1.6	1.7	4.1
1	F	F	Lauri Markkanen	392	2019_20	5.0	11.8	2.2	6.3	2.8	...	2.5	3.1	1.2	5.1	6.3	1.5	0.8	0.5	1.6	1.9
2	F	F	Taurean Prince	506	2019_20	4.3	11.5	2.3	6.7	2.1	...	1.1	1.4	0.8	5.2	6.0	1.8	0.9	0.4	2.0	2.5
3	C	C	Jaren Jackson Jr.	973	2020_21	4.8	11.4	1.5	5.5	3.3	...	3.2	3.8	1.5	4.1	5.6	1.1	1.1	1.6	1.4	3.8
4	F	F	Lauri Markkanen	1064	2020_21	4.9	10.2	2.3	5.8	2.6	...	1.5	1.8	0.7	4.6	5.3	0.9	0.5	0.3	1.0	1.5
5	F	F	Taurean Prince	1188	2020_21	3.2	8.0	1.5	3.8	1.7	...	1.6	1.9	0.5	3.0	3.5	1.9	0.7	0.6	1.1	1.8
6	F	F	Taurean Prince	1189	2020_21	2.5	6.2	1.1	3.1	1.4	...	2.0	2.3	0.2	2.7	2.8	0.6	0.7	0.7	0.9	1.8
7	F	F	Taurean Prince	1190	2020_21	3.5	8.7	1.7	4.1	1.8	...	1.4	1.7	0.6	3.1	3.7	2.4	0.7	0.5	1.2	1.9
8	F	F	Jaren Jackson Jr.	1716	2021_22	5.5	13.3	1.6	5.1	3.9	...	3.6	4.4	1.5	4.3	5.8	1.1	0.9	2.3	1.7	3.5
9	F	F	Lauri Markkanen	1829	2021_22	5.1	11.5	2.2	6.2	2.9	...	2.3	2.6	1.0	4.7	5.7	1.3	0.7	0.5	0.9	2.1
10	F	F	Taurean Prince	1967	2021_22	2.6	5.7	1.2	3.3	1.3	...	0.9	1.2	0.4	2.1	2.5	1.0	0.7	0.3	0.8	1.7

11 rows × 21 columns

Predict position Guard

Accurate = 8 / 11
~= 72.73% which is similar with the accurate score of model.

Maybe Wesley Matthews plays like Forward, maybe ...

	Prediction	Pos	Player	Rk	Season	FG	FGA	3P	3PA	2P	...	FT	FTA	ORB	DRB	TRB	AST	STL	BLK	TOV	PF
0	G	G	Wesley Matthews	399	2019_20	2.5	6.3	1.6	4.4	0.9	...	0.8	1.0	0.3	2.1	2.5	1.4	0.6	0.1	0.6	1.5
1	G	G	Derrick Rose	527	2019_20	7.4	15.1	0.9	2.9	6.5	...	2.4	2.8	0.5	1.9	2.4	5.6	0.8	0.3	2.5	1.0
2	G	G	Mason Jones	1001	2020_21	1.6	3.8	0.8	2.1	0.9	...	1.3	2.0	0.2	1.5	1.7	1.3	0.2	0.0	0.9	0.8
3	F	G	Mason Jones	1002	2020_21	1.8	4.4	0.9	2.5	0.9	...	1.3	2.2	0.2	1.8	2.0	1.5	0.2	0.0	1.1	0.8
4	F	G	Wesley Matthews	1074	2020_21	1.5	4.3	1.1	3.4	0.4	...	0.6	0.7	0.3	1.3	1.6	0.9	0.7	0.3	0.4	1.4
5	G	G	Derrick Rose	1221	2020_21	5.7	12.2	1.0	2.6	4.7	...	2.2	2.5	0.4	2.2	2.6	4.2	1.0	0.4	1.6	1.1
6	G	G	Derrick Rose	1222	2020_21	5.3	12.3	0.9	2.6	4.4	...	2.8	3.3	0.4	1.5	1.9	4.2	1.2	0.3	1.9	1.1
7	G	G	Derrick Rose	1223	2020_21	5.9	12.2	1.1	2.6	4.9	...	1.9	2.2	0.4	2.5	2.9	4.2	0.9	0.4	1.4	1.1
8	G	G	Mason Jones	1756	2021_22	1.8	3.8	0.3	1.0	1.5	...	3.0	3.8	0.3	2.3	2.5	1.0	0.5	0.0	0.3	2.0
9	F	G	Wesley Matthews	1839	2021_22	1.8	4.5	1.1	3.2	0.7	...	0.4	0.6	0.6	1.3	1.9	0.7	0.5	0.2	0.4	1.8
10	G	G	Derrick Rose	1997	2021_22	4.7	10.5	1.4	3.5	3.3	...	1.2	1.2	0.8	2.2	3.0	4.0	0.8	0.5	1.5	0.6

11 rows × 21 columns

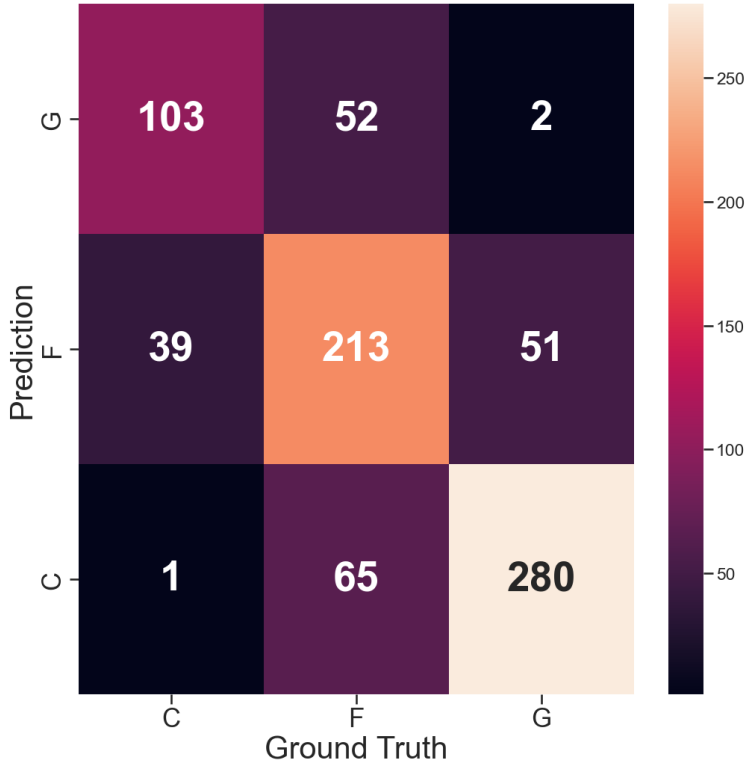
SVM Model

```
from sklearn.svm import SVC
```

```
rbfsvc = SVC(kernel='rbf', gamma='auto', C=10.0)
rbfsvc = rbfsvc.fit(X_train, y_train)
y_pred = rbfsvc.predict(X_test)
# Preciision, recall, f-score from the multi-class support function
print(classification_report(y_test, y_pred))
print('Accuracy score: ', round(accuracy_score(y_test, y_pred), 2))
print('F1 Score: ', round(f1_score(y_test, y_pred, average='micro'), 2))
```

```
              precision    recall  f1-score   support

           0       0.72      0.66      0.69       157
           1       0.65      0.70      0.67       303
           2       0.84      0.81      0.82       346

    accuracy                           0.74       806
   macro avg       0.74      0.72      0.73       806
weighted avg       0.74      0.74      0.74       806

Accuracy score:  0.74
F1 Score:  0.74
```

Result is similar with KNN model



Random Forest

```
from sklearn.ensemble import RandomForestClassifier
```

```
rc = RandomForestClassifier(n_estimators=200)
rc = rc.fit(X_train, y_train)
y_pred = rc.predict(X_test)
# Preciision, recall, f-score from the multi-class support function
print(classification_report(y_test, y_pred))
print('Accuracy score: ', round(accuracy_score(y_test, y_pred), 2))
print('F1 Score: ', round(f1_score(y_test, y_pred, average='micro'), 2))
```

```
              precision    recall  f1-score   support

           0       0.78      0.68      0.73       157
           1       0.66      0.73      0.69       303
           2       0.84      0.81      0.83       346

    accuracy                           0.75       806
   macro avg       0.76      0.74      0.75       806
weighted avg       0.76      0.75      0.76       806

Accuracy score:  0.75
F1 Score:  0.75
```

Result is similar with KNN model