Concordia University
Engineering and Computer Science

Big Data Analytics SOEN 691

Dr. Tristan Glatard
Winter 2020
Team ID 12

NBA Playoff Prediction

YING RAO

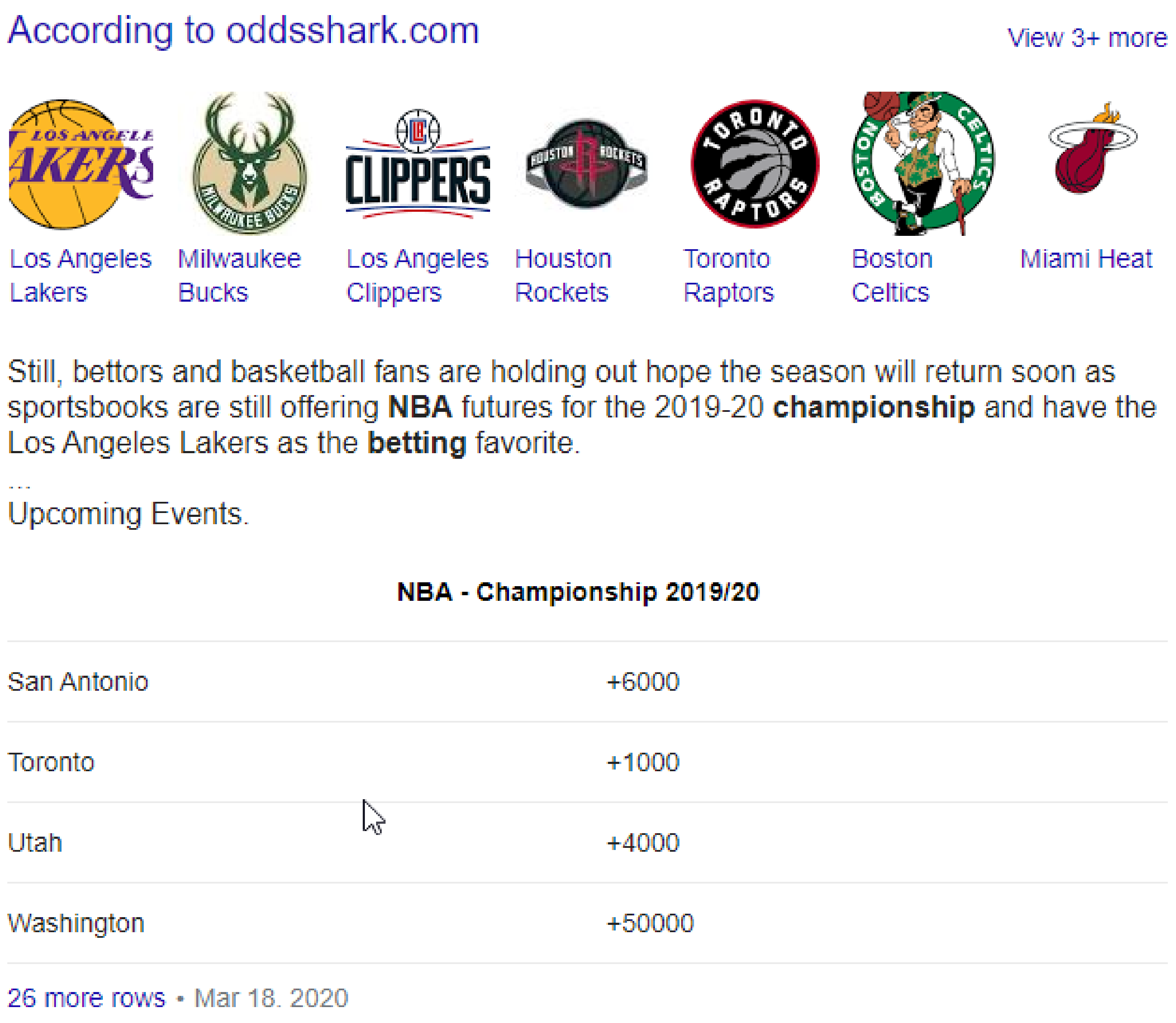
ELIE DOSH

PROJECT REPO

HTTPS://GITHUB.COM/SKIERWING/SOEN691_PROJECT_ED_YR.GIT

Introduction

- The goal of the project is to predict if an NBA Team will make it to the playoff or NOT



According to oddsshark.com

View 3+ more

Still, bettors and basketball fans are holding out hope the season will return soon as sportsbooks are still offering **NBA** futures for the 2019-20 **championship** and have the Los Angeles Lakers as the **betting** favorite.

...

Upcoming Events.

NBA - Championship 2019/20

San Antonio	+6000
Toronto	+1000
Utah	+4000
Washington	+50000

26 more rows • Mar 18, 2020

Data preparation and analysis

- ❑Manually gather data from basketball-reference.com
  - ❑ Between Seasons 1980-2018
- ❑Analyzing the data:
  - ❑Number of teams changed over the years 23, 27,29 and lately 30
  - ❑ Seasons 1980-1983 12 Teams 1984-2018 16 Teams to playoff
  - ❑Each team plays different number of minutes in total
  - ❑Used the Minutes Played field as common denominator for all Classifiers
- ❑Using Spark framework and the Dataframe Library to generate the dataset

Points_Per_minute	3Points_Per_minute	2Points_Per_minute	FThrow_Per_minute
Rebound_Per_minute	Assists_Per_minute	Steals_Per_minute	Blocks_Per_minute
TurnOvers_Per_minute			
Playoff			

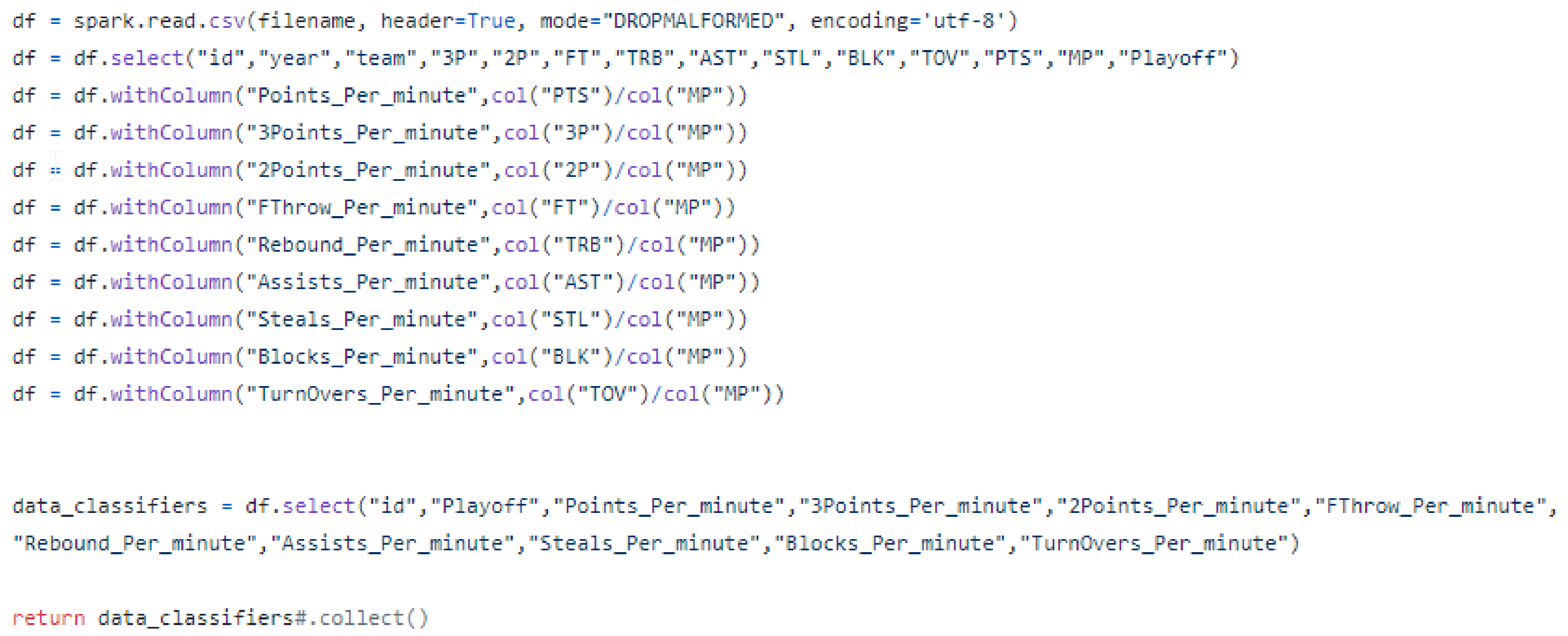
Machine Learning Libraries

Small Dataset 1074 records & 9 Classifiers

Decision to use Gaussian Naïve Bayes & SVM supervised learning models

Implemented "scikit-learn" library (Sklearn)

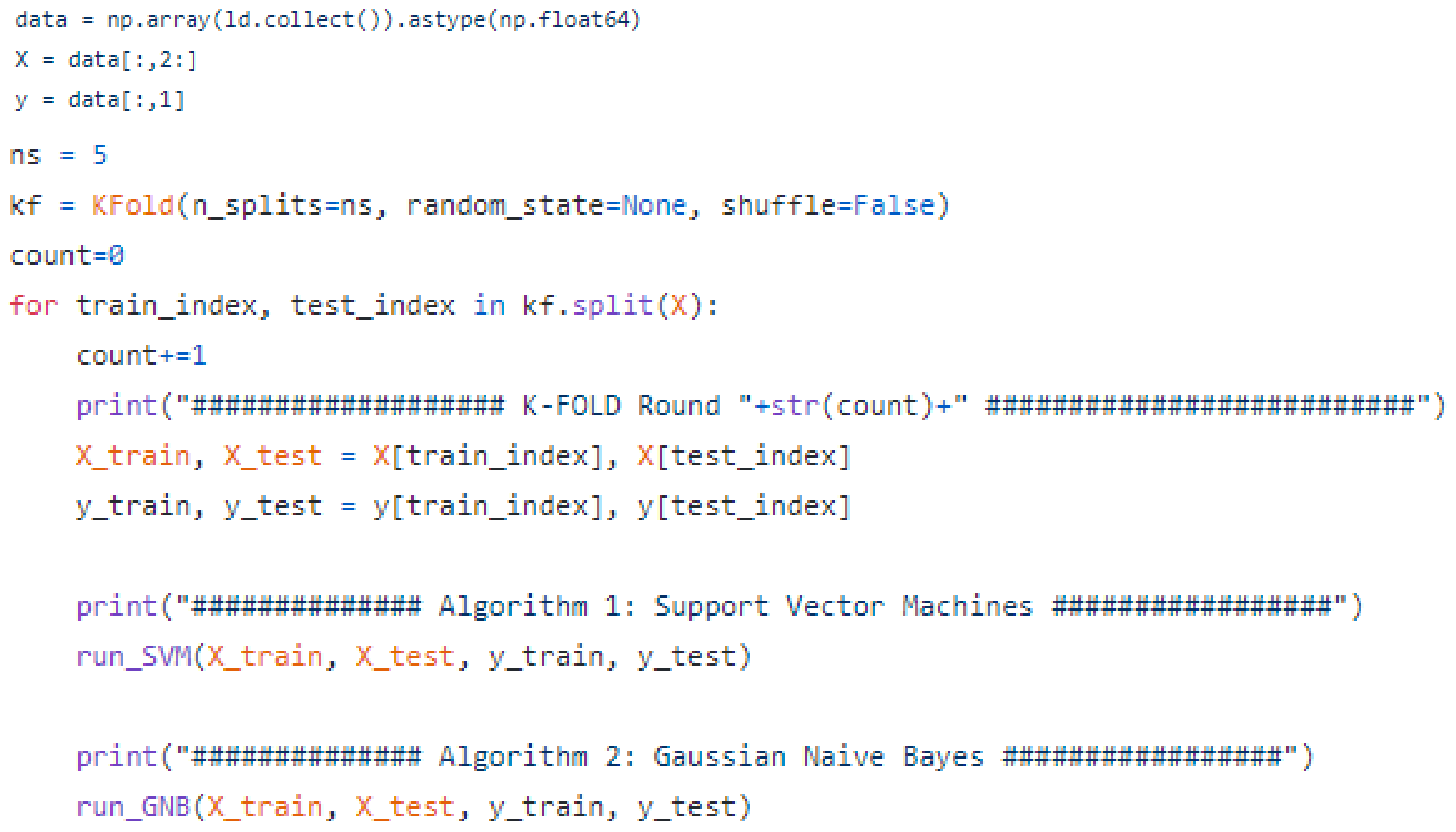
Used K-fold technique for training and testing



```
df = spark.read.csv(filename, header=True, mode="DROPMALFORMED", encoding='utf-8')
df = df.select("id","year","team","3P","2P","FT","TRB","AST","STL","BLK","TOV","PTS","MP","Playoff")
df = df.withColumn("Points_Per_minute",col("PTS")/col("MP"))
df = df.withColumn("3Points_Per_minute",col("3P")/col("MP"))
df = df.withColumn("2Points_Per_minute",col("2P")/col("MP"))
df = df.withColumn("FThrow_Per_minute",col("FT")/col("MP"))
df = df.withColumn("Rebound_Per_minute",col("TRB")/col("MP"))
df = df.withColumn("Assists_Per_minute",col("AST")/col("MP"))
df = df.withColumn("Steals_Per_minute",col("STL")/col("MP"))
df = df.withColumn("Blocks_Per_minute",col("BLK")/col("MP"))
df = df.withColumn("TurnOvers_Per_minute",col("TOV")/col("MP"))


data_classifiers = df.select("id","Playoff","Points_Per_minute","3Points_Per_minute","2Points_Per_minute","FThrow_Per_minute",
"Rebound_Per_minute","Assists_Per_minute","Steals_Per_minute","Blocks_Per_minute","TurnOvers_Per_minute")

return data_classifiers#.collect()
```

Technical Workflow

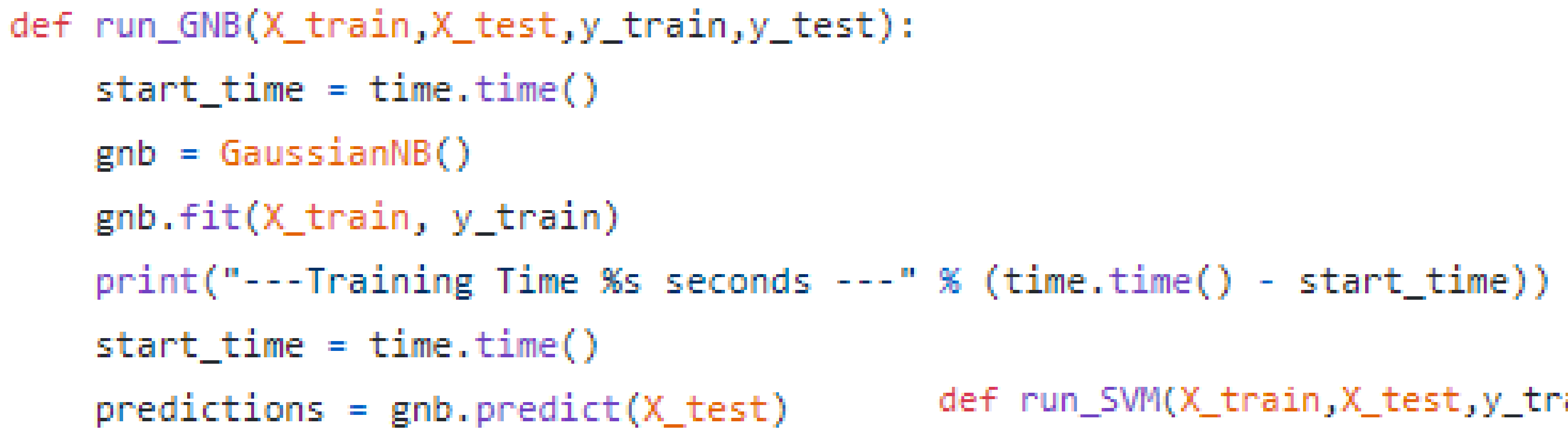
- Data loading and pre-processing: Spark dataframe



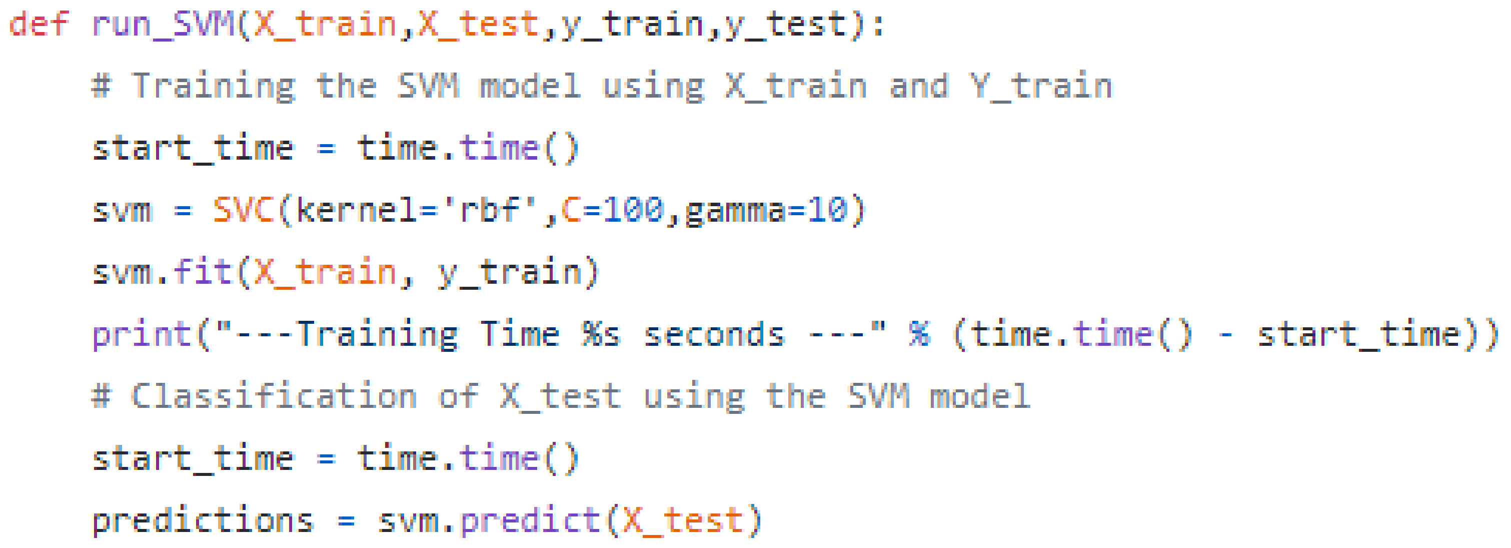
```
data = np.array(ld.collect()).astype(np.float64)
X = data[:,2:]
y = data[:,1]
ns = 5
kf = KFold(n_splits=ns, random_state=None, shuffle=False)
count=0
for train_index, test_index in kf.split(X):
    count+=1
    print("################### K-FOLD Round "+str(count)+" ##########################")
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]

    print("############## Algorithm 1: Support Vector Machines #################")
    run_SVM(X_train, X_test, y_train, y_test)

    print("############## Algorithm 2: Gaussian Naive Bayes #################")
    run_GNB(X_train, X_test, y_train, y_test)
```

Technical Workflow

- Split Data using K-fold: sklearn.model_selection
  - K-fold is a good cross-validation method that tackles overfitting or underfitting



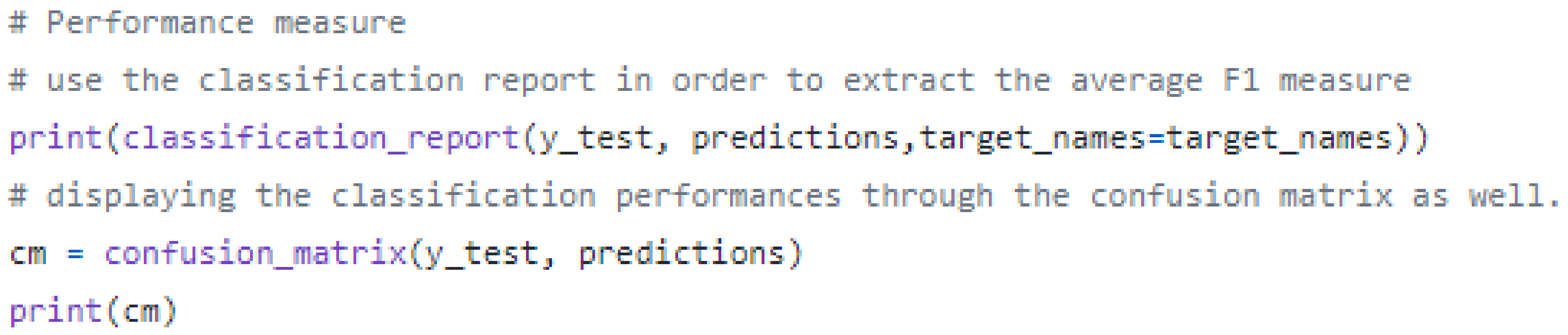
```
def run_GNB(X_train,X_test,y_train,y_test):
    start_time = time.time()
    gnb = GaussianNB()
    gnb.fit(X_train, y_train)
    print("---Training Time %s seconds ---" % (time.time() - start_time))
    start_time = time.time()
    predictions = gnb.predict(X_test)
```



```
def run_SVM(X_train,X_test,y_train,y_test):
    # Training the SVM model using X_train and Y_train
    start_time = time.time()
    svm = SVC(kernel='rbf',C=100,gamma=10)
    svm.fit(X_train, y_train)
    print("---Training Time %s seconds ---" % (time.time() - start_time))
    # Classification of X_test using the SVM model
    start_time = time.time()
    predictions = svm.predict(X_test)
```

Technical Workflow

- Data modeling and classification: scikit-learn
  - **library:** sklearn.naive_bayes GaussianNB and sklearn.svm SVC



```
# Performance measure
# use the classification report in order to extract the average F1 measure
print(classification_report(y_test, predictions,target_names=target_names))
# displaying the classification performances through the confusion matrix as well.
cm = confusion_matrix(y_test, predictions)
print(cm)
```

Technical Workflow

- ❑Performance evaluation: F1 score, confusion matrix
  - ❑**library:** sklearn.metrics classification_report and sklearn.metrics confusion_matrix

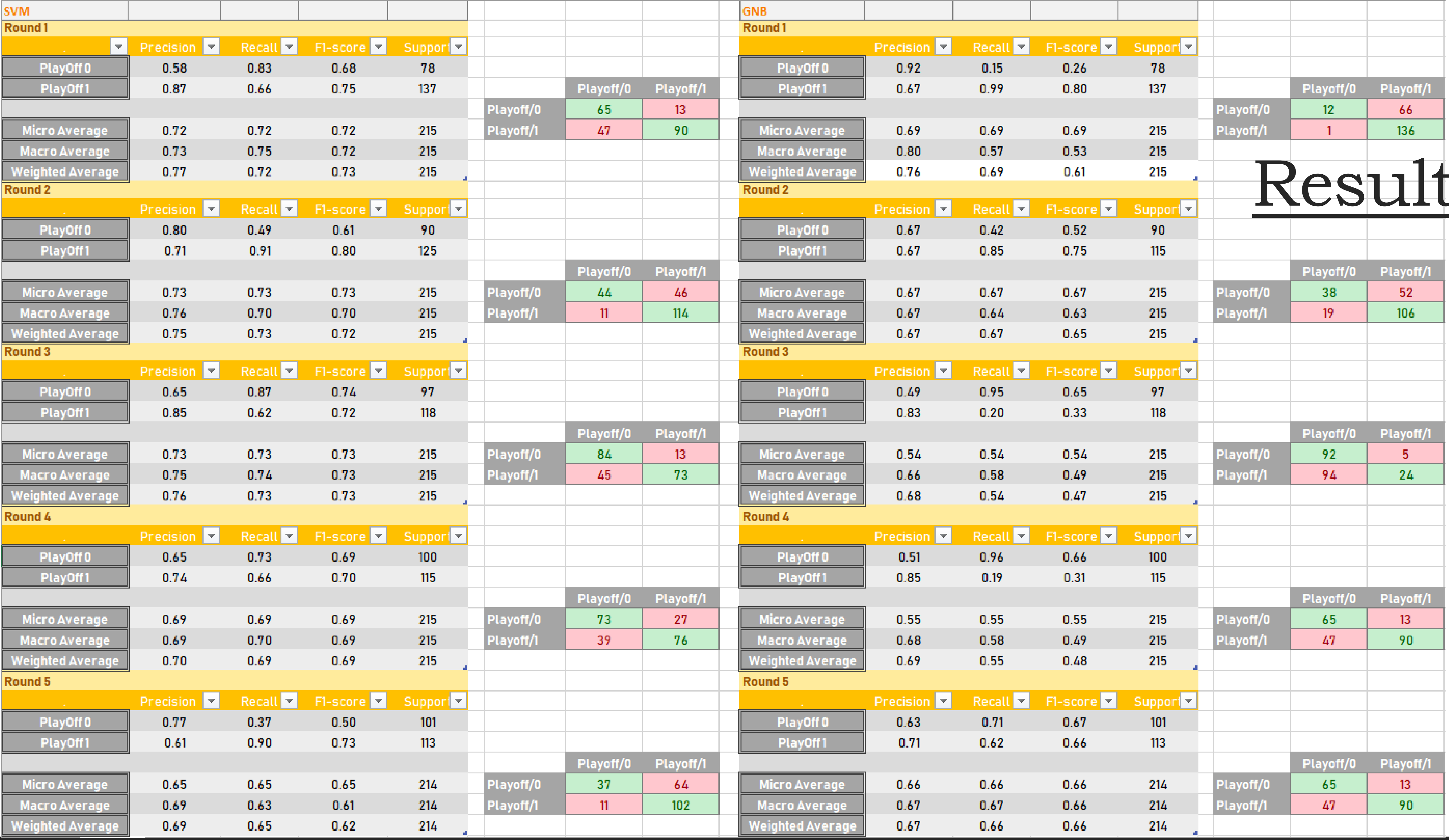
Observation

- ❑Base on our chosen dataset and classifiers
- ❑Naïve Bayes Classifier was quicker during execution
- ❑SVM Classifier is better at predicting the Playoff

❑ Average Weighted Precision :	[SVM=0.734]	>	[GNB=0.694]
❑ Average Weighted Recall :	[SVM=0.704]	>	[GNB=0.622]
❑ Average Weighted F1-Score :	[SVM=0.698]	>	[GNB=0.574]

- ❑ Results

Results



SVM

Round 1

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.58	0.83	0.68	78
PlayOff 1	0.87	0.66	0.75	137
Micro Average	0.72	0.72	0.72	215
Macro Average	0.73	0.75	0.72	215
Weighted Average	0.77	0.72	0.73	215

	Playoff/0	Playoff/1
Playoff/0	65	13
Playoff/1	47	90

Round 2

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.80	0.49	0.61	90
PlayOff 1	0.71	0.91	0.80	125
Micro Average	0.73	0.73	0.73	215
Macro Average	0.76	0.70	0.70	215
Weighted Average	0.75	0.73	0.72	215

	Playoff/0	Playoff/1
Playoff/0	44	46
Playoff/1	11	114

Round 3

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.65	0.87	0.74	97
PlayOff 1	0.85	0.62	0.72	118
Micro Average	0.73	0.73	0.73	215
Macro Average	0.75	0.74	0.73	215
Weighted Average	0.76	0.73	0.73	215

	Playoff/0	Playoff/1
Playoff/0	84	13
Playoff/1	45	73

Round 4

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.65	0.73	0.69	100
PlayOff 1	0.74	0.66	0.70	115
Micro Average	0.69	0.69	0.69	215
Macro Average	0.69	0.70	0.69	215
Weighted Average	0.70	0.69	0.69	215

	Playoff/0	Playoff/1
Playoff/0	73	27
Playoff/1	39	76

Round 5

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.77	0.37	0.50	101
PlayOff 1	0.61	0.90	0.73	113
Micro Average	0.65	0.65	0.65	214
Macro Average	0.69	0.63	0.61	214
Weighted Average	0.69	0.65	0.62	214

	Playoff/0	Playoff/1
Playoff/0	37	64
Playoff/1	11	102

GNB

Round 1

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.92	0.15	0.26	78
PlayOff 1	0.67	0.99	0.80	137
Micro Average	0.69	0.69	0.69	215
Macro Average	0.80	0.57	0.53	215
Weighted Average	0.76	0.69	0.61	215

	Playoff/0	Playoff/1
Playoff/0	12	66
Playoff/1	1	136

Round 2

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.67	0.42	0.52	90
PlayOff 1	0.67	0.85	0.75	115
Micro Average	0.67	0.67	0.67	215
Macro Average	0.67	0.64	0.63	215
Weighted Average	0.67	0.67	0.65	215

	Playoff/0	Playoff/1
Playoff/0	38	52
Playoff/1	19	106

Round 3

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.49	0.95	0.65	97
PlayOff 1	0.83	0.20	0.33	118
Micro Average	0.54	0.54	0.54	215
Macro Average	0.66	0.58	0.49	215
Weighted Average	0.68	0.54	0.47	215

	Playoff/0	Playoff/1
Playoff/0	92	5
Playoff/1	94	24

Round 4

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.51	0.96	0.66	100
PlayOff 1	0.85	0.19	0.31	115
Micro Average	0.55	0.55	0.55	215
Macro Average	0.68	0.58	0.49	215
Weighted Average	0.69	0.55	0.48	215

	Playoff/0	Playoff/1
Playoff/0	65	13
Playoff/1	47	90

Round 5

.	Precision	Recall	F1-score	Suppor
PlayOff 0	0.63	0.71	0.67	101
PlayOff 1	0.71	0.62	0.66	113
Micro Average	0.66	0.66	0.66	214
Macro Average	0.67	0.67	0.66	214
Weighted Average	0.67	0.66	0.66	214

	Playoff/0	Playoff/1
Playoff/0	65	13
Playoff/1	47	90

Conclusion

- ❑Given our dataset
  - ❑ Gaussian Naïve Bayes performed the worst
  - ❑ SVM Decent for playoffs prediction [~70%]

Thank You

Questions?

References

- Google
- https://www.basketball-reference.com/leagues/
- https://scikit-learn.org/stable/
- https://muthu.co/understanding-the-classification-report-in-sklearn/
- https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_recall_fscore_support.html
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html