Classifier used: **LinearSVC(Training set score for KNN: 0.66,Test set score for KNN: 0.64)**

Other classifier used**:KNN(Training set score for KNN: 0.564,Test set score for KNN: 0.556)**

**Feature Selection:**

A screenshot of a computer

Description automatically generated

I performed analysis like above to know which features make the most difference for the target variable Pos. We could see that FGA,3PA, ORB,TRB,DRB,AST,STL,BLK,PTS makes difference for ‘Pos’

**Data Preprocessing:**

The dataset contains attributes such as player name and team name, age. We know that they are not useful for classification and thus do not include them as features.

data = data[data.MP >= 5]

data = data[data.PTS >= 5]

I also filtered the data with above conditions to choose players based on MP(Minutes Played) to avoid bench players (MP=0).

Based on PTS(average points scored per game) , I filtered games that are played atleast for 5 minutes.

**Chi square test:**

Here are the top 15 features based on chi square test:

A table of numbers with black text

Description automatically generated

Some of the features are redundant. For instance, field goal percentage (FG%) is defined by field goals made (FG) and field goal attempts (FGA): FG% = FG / FGA. It may not be beneficial to include all of them.

Hence I am removing the below columns from the data .

#using above analysis and domain knowledge removing following columns

columns\_to\_remove = ['G', 'GS', 'FG%', '3P%', '2P%','FT%', 'MP','FT', 'FG','eFG%']

data = data.drop(columns=columns\_to\_remove)

**First model:**

**KNN:**

A graph of a graph showing the results of a test accuracy

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

Cross validation results for KNN:

A screenshot of a computer code

Description automatically generated

**Method/Steps performed to improve accuracy:**

1. I wrote the code for all the different classifiers (Decision Tree, SVM, KNN, Naïve Bayes) and checked the initial accuracy of these models on the NBA data set by using all the feature columns. Only SVM and KNN gave me a better initial accuracy so I selected these 2 and worked on these.

**I used the following techniques and actions to increase the K-Nearest Neighbors (KNN) classifier's accuracy:**

**1. Selection of Features:**

I chose a set of characteristics that I thought would be more useful in determining the player's position. Field goals attempted (FGA), three-point attempts (3PA), two-point attempts (2PA), free throw attempts (FTA), assists (AST), steals (STL), blocks (BLK), total rebounds (TRB), offensive rebounds (ORB), defensive rebounds (DRB), and personal fouls (PF) were among the attributes I used. The significance of these numbers in basketball positions served as the basis for this choice.

**2. Preprocessing Data:**

I used `LabelEncoder` to encode the target variable 'Pos' by transforming the position labels into numeric labels. To use the KNN method with categorical target variables, this step is required.

**3.Feature Scaling:**

To make sure that every feature had the same scale, I used StandardScaler to standardize the feature values. Since KNN algorithms depend on the separation of data points, standardization improves their performance. Furthermore, it keeps large-scale characteristics from influencing the distance computations.

**Selecting the K-Neighbors:**

I experimented with various values of n\_neighbors, the number of neighbors to take into account, in order to do hyperparameter tweaking for the KNN classifier. I conducted tests on a range of numbers, 1 to 10. I trained and evaluated the KNN model for each value in order to determine the ideal number of neighbors that produced the best test accuracy. The figure I selected for best\_n\_neighbors was the one that produced the best test accuracy.The best\_knn is found to be 9.

**How to Use Euclidean Distance**

I employed the Euclidean distance (p=2), which is the KNN's default distance metric. Given that it calculates the geometric distance between data points, this is appropriate for the dataset.

**Assessment and Illustration:**

I calculated the test and training accuracies for various settings of n\_neighbors in order to assess the KNN models' performance. To see how the model performed at various K values, I plotted these accuracies versus the number of neighbors.

**Selecting the Ideal Model:**

I determined which KNN classifier performed best by analyzing the models for various n\_neighbors and choosing the model with the highest test accuracy. Best\_n\_neighbors was utilized to record the ideal K value, and the matching model was used to make predictions.

**Second Model:**

**SVM : Best Accuracy achieved:**

**Training set accuracy :66%**

**Test set accuracy :64%**

**First try:A white background with black numbers

Description automatically generated**

**Second Try:**

I filtered the data more and run SVM .Here is the accuracy then:

**data = data[data.PTS >= 8]**

**PTS: Average points scored per game.**

**A screenshot of a math test

Description automatically generated**

**Cross Validation results:**

**A screenshot of a computer program

Description automatically generated**

**Method/Steps performed to improve accuracy:**

In order to enhance the accuracy of Linear Support Vector Classifier (LinearSVC) model, I executed the below procedures and enhancements:

**Feature Selection:**

I have chosen a collection of characteristics that I think are important to consider when estimating a player's position. These characteristics include personal fouls (PF), steals (STL), assists (AST), points (PTS), two-point attempts (2PA), three-point attempts (3PA), total rebounds (TRB), defensive rebounds (DRB), and blocks (BLK). The significance of these numbers in basketball positions and other steps served as the basis for this choice.

**Preprocessing of Data:**

To make sure that every feature had the same scale, I used StandardScaler to standardize the feature values. Since the LinearSVC model is sensitive to feature scaling, standardization aids in its improved performance and faster convergence.

**Model Selection and Optimization of Hyperparameters:**

I employed a linear support vector classifier, or LinearSVC model. For binary and multi-class classification problems, such estimating the positions of basketball players, LinearSVC is suitable.Since it often

performs well for datasets with a lot of features (like this one), I set the dual option to True (the default value).To make sure the optimization process converges in the allotted amount of iterations, I set the max\_iter option to 10,000. To enhance the model's performance even further, we may think about additional optimizations and hyperparameter tweaking in addition to the actions already made. If there are imbalances in the classes, we may investigate other kernels, play about with different parameters like the penalty parameter C, and take class weighting into consideration.