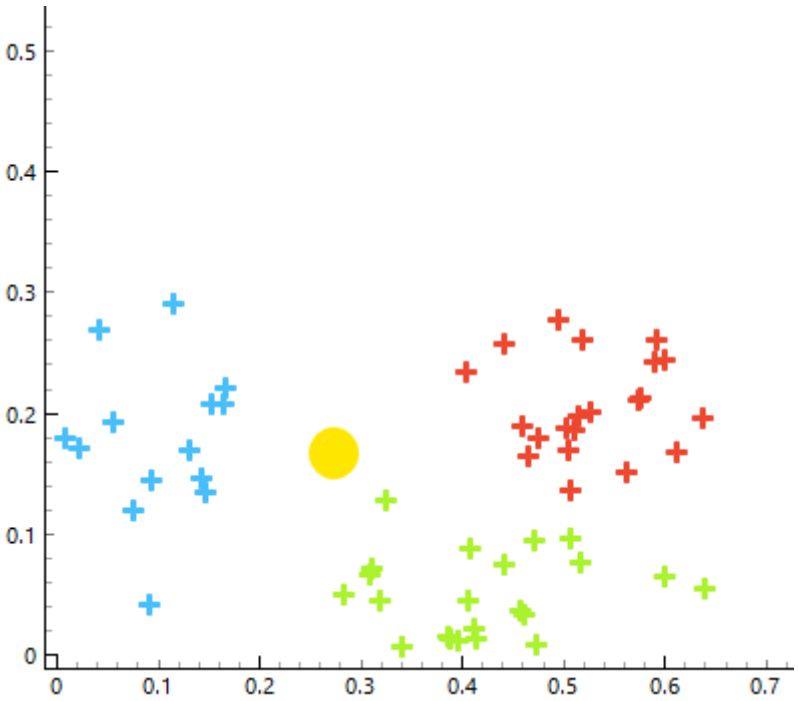
Observações e Considerações para o Modelo NBA

1. O que é o KNN (K-nearest neighbors)?

O KNN é um algoritmo supervisionado e não paramétrico que realiza classificações ou previsões com base na proximidade entre os dados, determinando o grupo ao
qual um ponto pertence ao analisar seus vizinhos mais próximos. Exemplo:



Qual grupo melhor se encaixa a bola amarela? O KNN consegue nos ajudar a classifica-lo.

2. Métricas e Distâncias para Classificação

Para determinar as distâncias entres os pontos, o KNN pode utilizar alguns tipos de cálculo para as distancias (Euclidiana, Manhattan**,** Minkowski), nesse projeto utilizei a mais conhecida, a Euclidiana.
Sua formula é bem simples, veja:

$$d(x,y) = \sqrt{\sum_i^n (Xi - xi)^2}$$

Basicamente ela mede uma linha reta entre dois pontos desejados.

3. Vantagens e Desvantagens

Aqui vamos passar de forma rápida pelas vantagens, pois o que nos interessa é as desvantagens.

3.1. Vantagens

- Facil de implementar
- Adapta-se facilmente
- Pouco parâmetros

3.2. Desvantagens

3.2.1. Consumo

O KNN é um algoritmo ‘preguiçoso’, ou seja, armazena todos os dados de treinamento e realiza os cálculos apenas no momento da predição. Isso faz com que consuma mais memória e espaço de armazenamento em comparação a outros classificadores.

3.2.2. Dimensionalidade

Não tem um bom desempenho com valores de dados de alta dimensionalidade, ou seja, dado um limite de dimensões o algoritmo começa a presentar mais erros na classificação ou na predição.

3.2.3. Overfitting

Devido a desvantagem anterior, o KNN está mais sujeito ao overfitting. É comum aplicar técnicas de seleção de características e redução de dimensionalidade. Além disso, a escolha do valor de k influencia diretamente o desempenho do modelo.

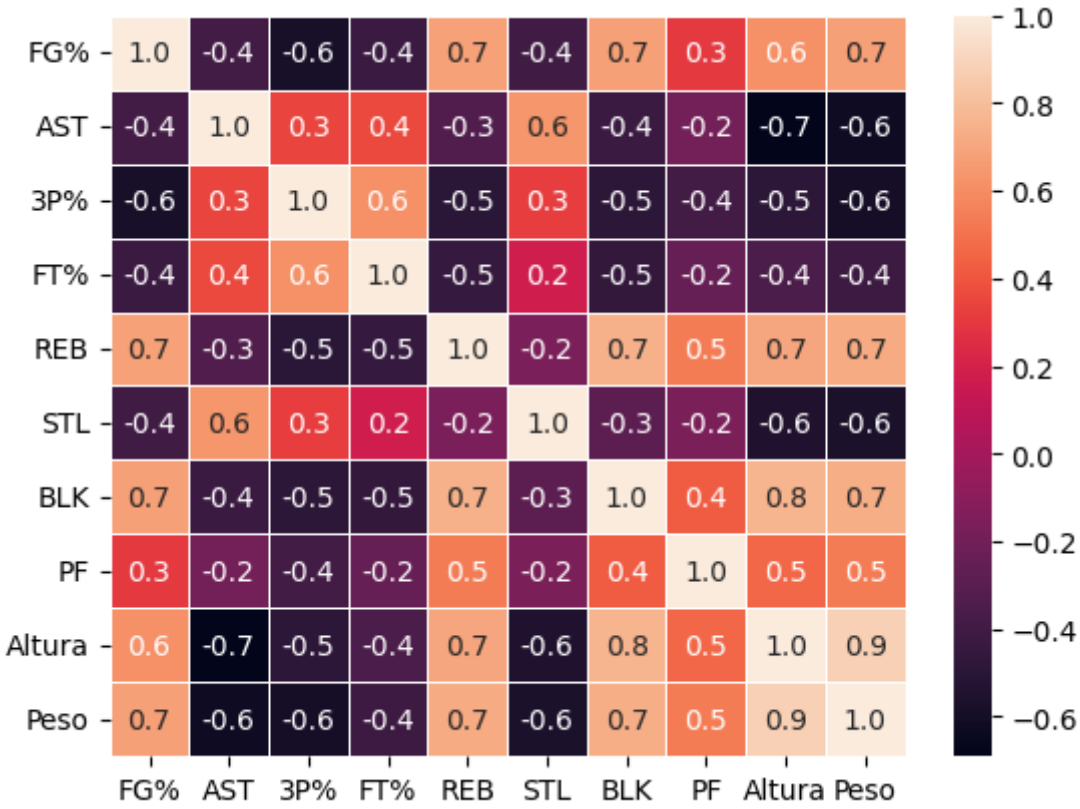
4. Considerações no nosso modelo NBA

4.1. Variáveis para treinar nosso modelo

Aqui nos deparamos com um problema, quais variáveis usar? Na NBA os jogadores possuem inúmeras estatísticas relacionada a pontos, faltas, rebotes e tudo mais. Mas qual delas melhor descreve uma possível clusterização desses jogadores? Não podemos usar todas elas?

Respondendo minhas próprias perguntas, não podemos usar todas devido a uma desvantagem do KNN que é a dimensionalidade, então vamos tentar diminuir ao máximo a

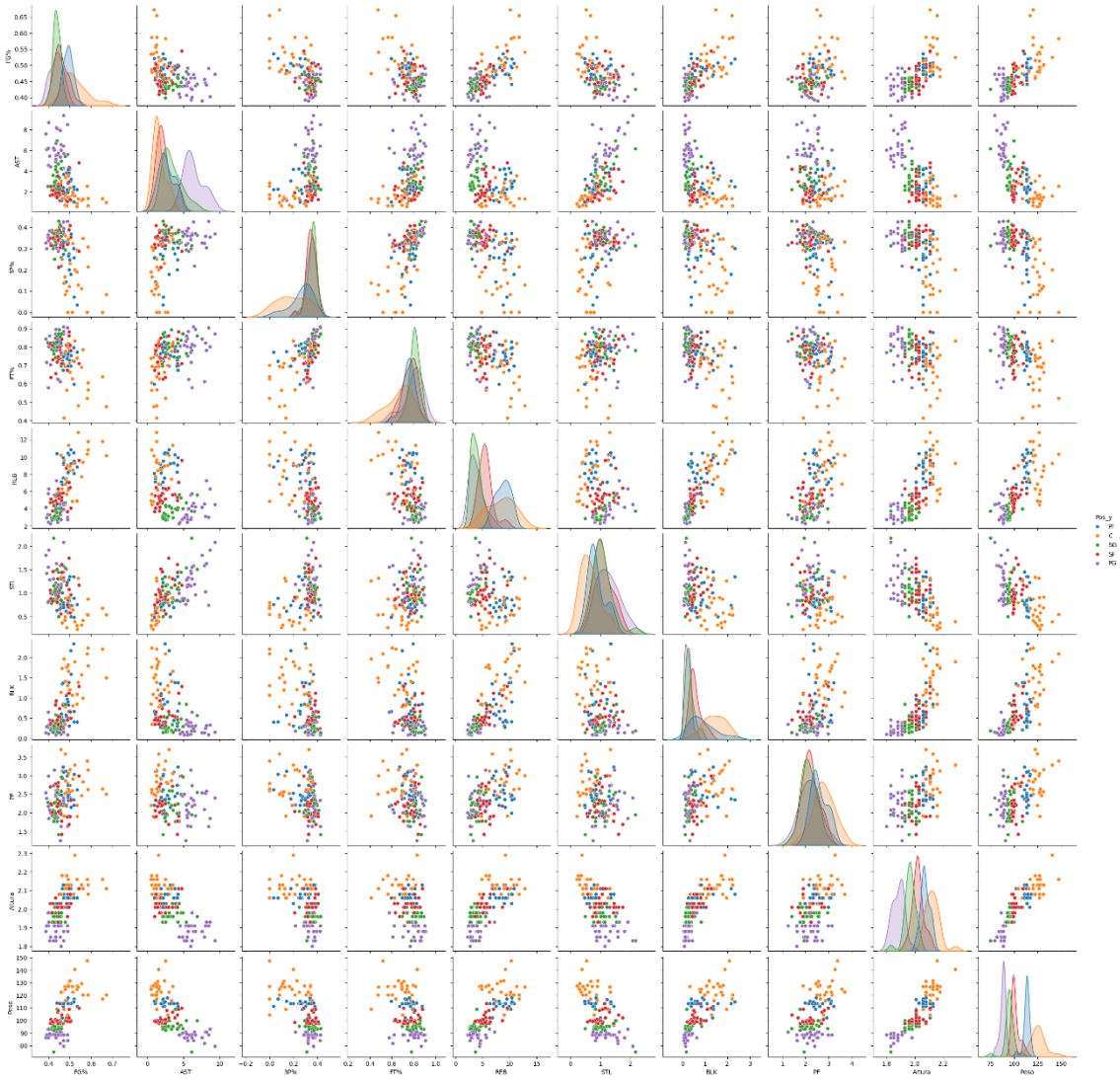
quantidade de tal forma que não sobrecarregue tanto nosso modelo. Vamos analisar um gráfico de dispersão e de correlação:



Válido ressaltar que a correlação está entre -1 e +1 e quanto mais próximo de zero mais fraca é sua correlação.

Vemos que algumas variáveis são bem fracas comparando a relação dela com as demais, exemplo é ‘FT%’ que é a porcentagem de lances livres.

Pegando a mesma amostra, vemos o gráfico de dispersão:



Obs.: está no git a imagem à parte.

Vemos que algumas variáveis possuem uma diferença dado as posições definidas. São elas: altura, peso, rebotes, assistências, faltas pessoais e bloqueios. Aqui vale deixar um comentário, em pensamentos mais ‘basquetebolisticos ‘, considerar pontos e porcentagens é bem irregular como mostra nos gráficos, mas também após os anos 2000 a gente tem inúmeros jogadores que possuem bons arremessos independentemente da posição que jogam. Se analisarmos bem, as assistências tem o mesmo caminho, porém é uma característica muito forte dos PG, basta olhar o gráfico de dispersão e vemos também a correlação negativa com peso e altura, ou seja, quanto mais altos e pesados os jogadores, menos assistências eles têm de modo geral. Então vamos utilizar essas 6 variáveis.

4.2. Amostra para treinar o modelo

Nesse tópico nos deparamos com outro problema. Para entendermos melhor, toda a mostra que eu consegui parte dos anos 2000, são aproximadamente 25 temporadas. Dentro disso, tem jogadores que começaram e terminara, apenas começara,

apenas terminaram, então resolvi fazer um corte considerando algumas condições pra que o jogador fosse elegível, são elas:

- Médias mais que 20 jogos por temporada;
- Não ter 0 em nenhuma estatística (menos 3PT%);
- Ter jogado mais que 4 temporadas.

Apesar de serem dois, já foi uma galera embora.

Uma segunda preocupação foi o fato que jogadores durante suas carreiras podem ter jogados em mais de uma posição (exemplo o Lebron que jogou em todas). Então para minha amostra um tanto quanto aleatória selecionei jogadores que são reconhecidos, em sua maior parte da carreira, por jogar em certa posição, exemplo é Dwight Howard que nos seus primeiros dois anos de liga era considerado PF, porém após isso ficou conhecido com C até o fim da sua carreira.

E por último, não menos importante, deixei para cada posição aproximadamente 25 jogadores, ou seja, para treinar nosso modelo temos uma amostra de aproximadamente 130 jogadores.

Para escolher eles, utilizei um pouco do conceito do KNN, vou explicar por partes:

1. Perguntei pra DeepSeek (tive que aproveitar esse projeto para conhece-la) quais eram os 5 jogadores mais emblemáticos das 5 posições após os anos 2000)
2. Peguei esses jogadores e pra cada posição calculei a distância entre esse grupo e toda minha base e selecionei os 20 mais próximos deles.
3. Exemplificando para os PG: a DeepSeek me disse: "Stephen Curry", "Chris Paul", "Jason Kidd", "Steve Nash" e "Russell Westbrook"
4. Calculei a distância entre eles e todos os PG e selecionei os 20 mais próximos. No fim, os 5 selecionados pela IA e os 20 mais próximos seria a amostra dos PG pra treinar o modelo.

É bem discutível esse número, porém creio que dentro do possível ele vai representar bem as características de cada grupo. E outro ponto, o grande intuito desse projeto não é acertar em cheio, mas sim ter o contato com todos esses processos.

5. Modelo Final

Agora depois de todo esse trabalho chegou à parte mais fácil, graças as bibliotecas de analise de dados que temos hoje, podemos escrever o comando em poucas linhas, o arquivo com os códigos estão no git, mas vou deixar uma foto aqui:

```
from sklearn.neighbors import KNeighborsClassifier
classe  = KNeighborsClassifier(n_neighbors=3)

X,y = df_foto[['AST','REB','BLK','PF','Peso','Altura']],df_foto['Pos_y']

result = classe.fit(X, y)

base = {'AST': [!],
       'REB': [!],
       'BLK': [!],
       'PF' : [!],
       'Peso': [!],
       'Altura': [!]}
player_escolhido = pd.DataFrame(base)

result.predict(player_escolhido)
```

Foi bem divertido a experiência!

Válido ressaltar alguns sites que me ajudaram:

https://www.basketball-reference.com/

https://www.ibm.com/br-pt/topics/knn