```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import GradientBoostingRegressor, RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

LOAD DATASET

DATA PRE PROCESSING

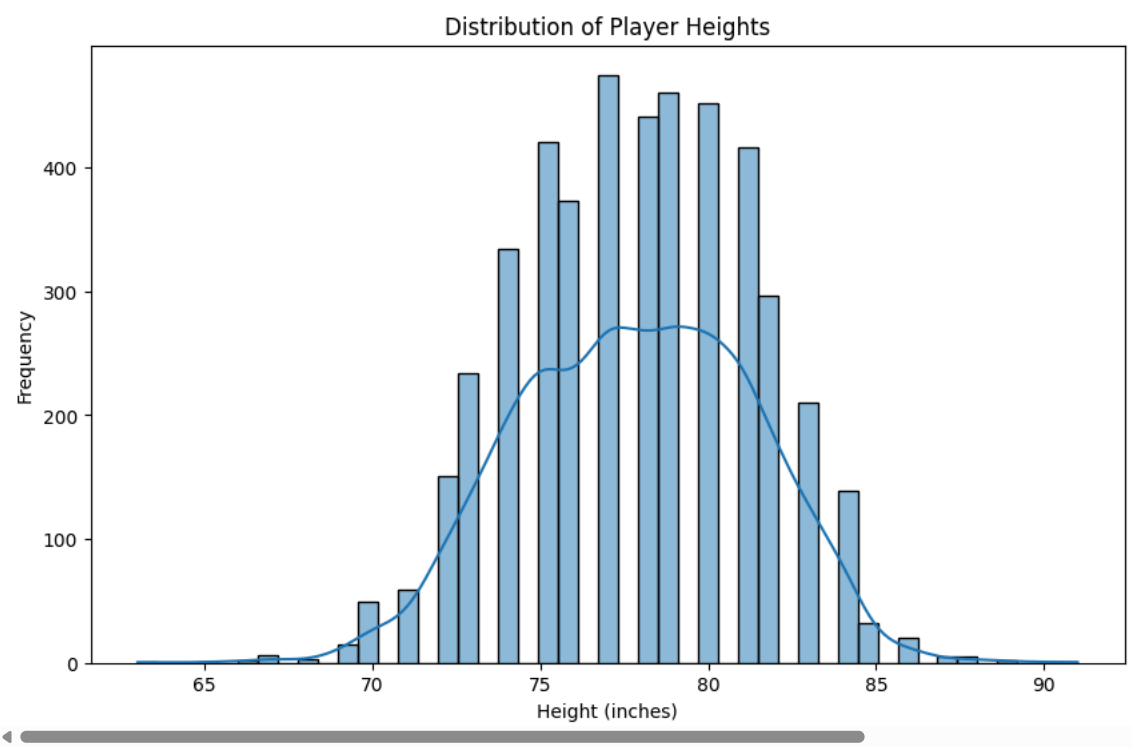
```
df = pd.read_csv('player_data.csv')
df.dropna(subset=['Ht', 'Wt', 'Pos', 'Birth Date', 'Colleges'], inplace=True)

def parse_ht(ht):
    feet, inches = ht.split('-')
    return int(feet) * 12 + int(inches)

df['Ht_inches'] = df['Ht'].apply(parse_ht)
df['Career_Length'] = df['To'].astype(int) - df['From'].astype(int)
```

LABEL ENCODING

```
label_encoder = LabelEncoder()
df['Pos_Encoded'] = label_encoder.fit_transform(df['Pos'])
df['College_Encoded'] = label_encoder.fit_transform(df['Colleges'])
```

DISTRIBUTION OF PLAYER HEIGHTS

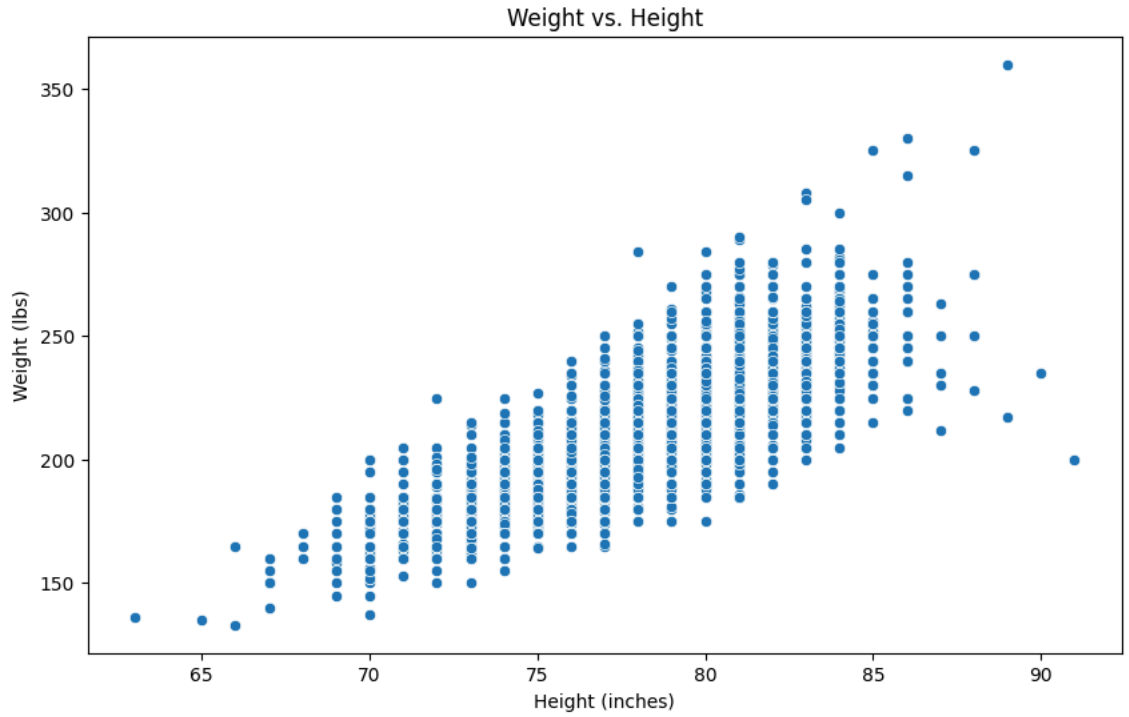
```
plt.figure(figsize=(10, 6))
sns.histplot(df['Ht_inches'], kde=True)
plt.title('Distribution of Player Heights')
plt.xlabel('Height (inches)')
plt.ylabel('Frequency')
plt.show()
```



WEIGHT VS HEIGHT SCATTER PLOT

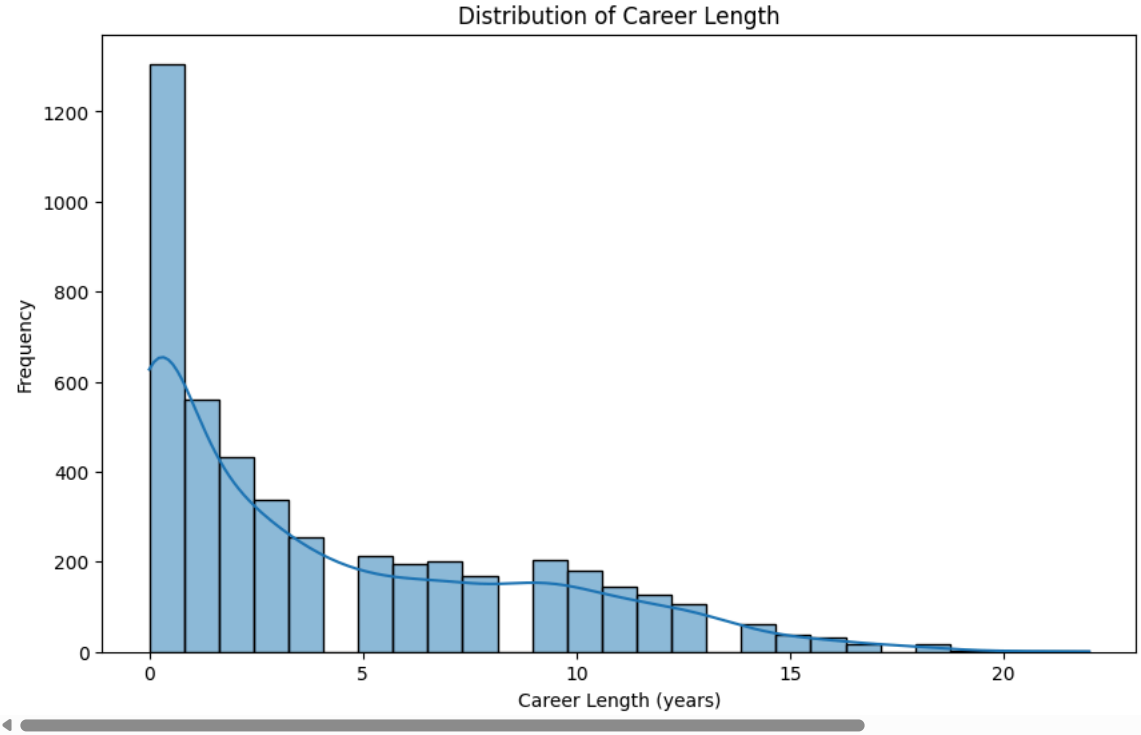
1. List item
2. List item

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Ht_inches', y='Wt', data=df)
plt.title('Weight vs. Height')
plt.xlabel('Height (inches)')
plt.ylabel('Weight (lbs)')
plt.show()
```



CAREER LENGTH DISTRIBUTION

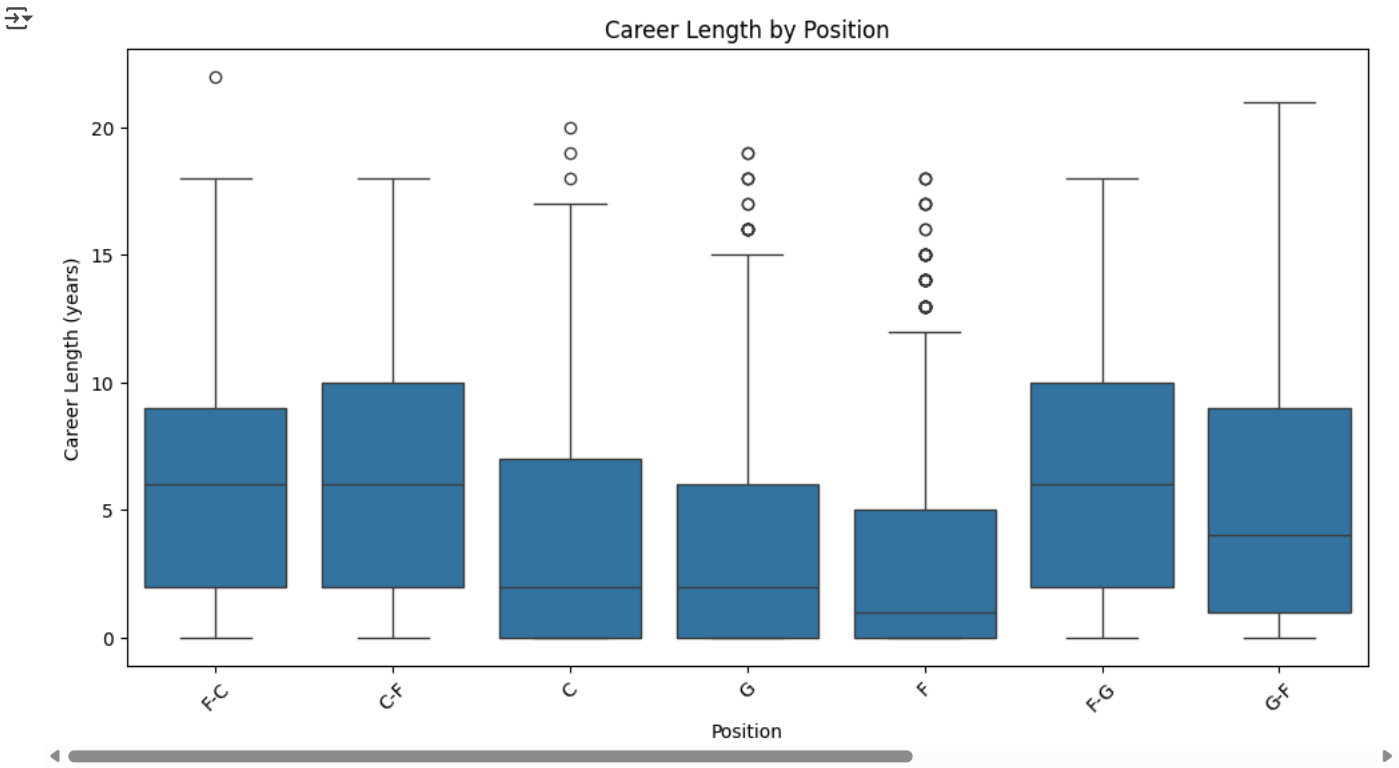
```
plt.figure(figsize=(10, 6))
sns.histplot(df['Career_Length'], kde=True)
plt.title('Distribution of Career Length')
plt.xlabel('Career Length (years)')
plt.ylabel('Frequency')
plt.show()
```



BOXPLOT OF CAREER LENGHT BY POSITION

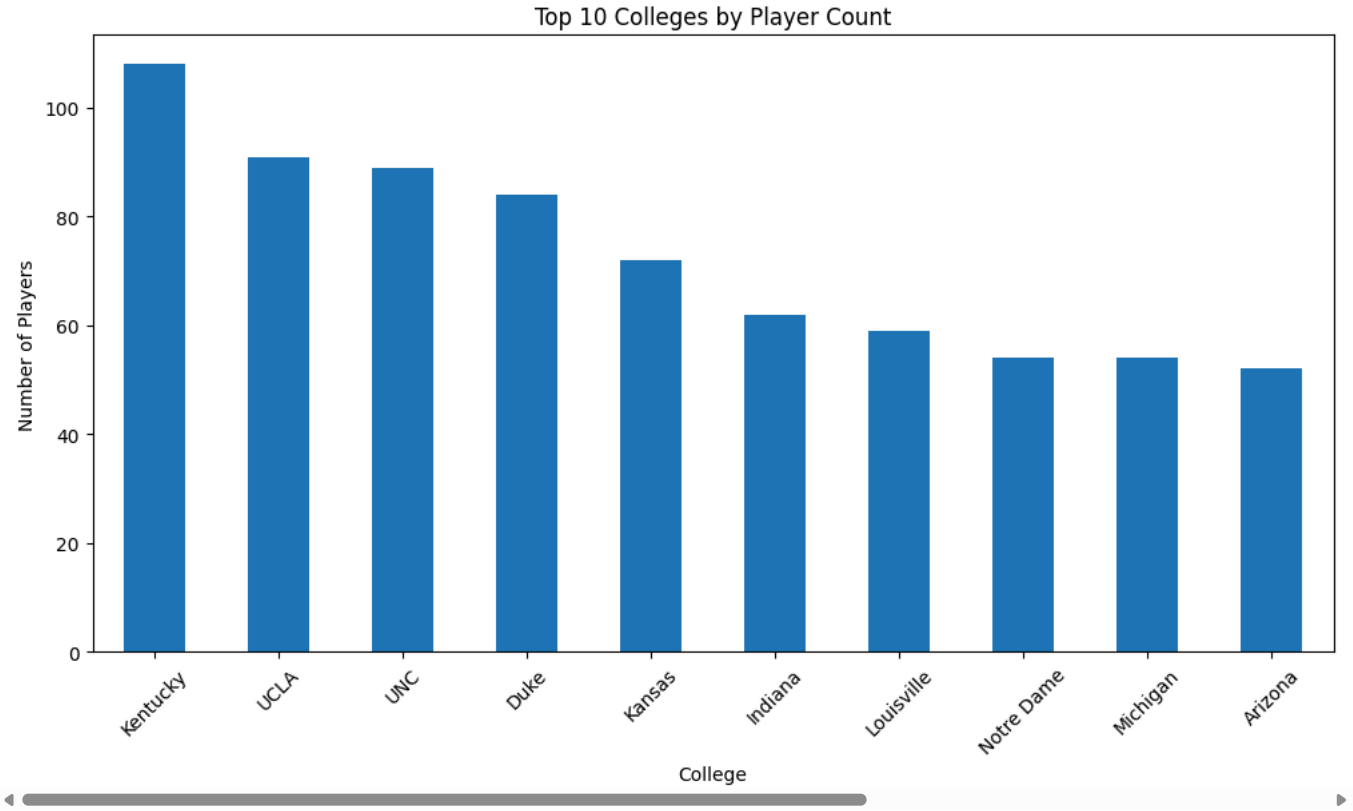
- List item
- List item

```
plt.figure(figsize=(12, 6))
sns.boxplot(x='Pos', y='Career_Length', data=df)
plt.title('Career Length by Position')
plt.xlabel('Position')
plt.ylabel('Career Length (years)')
plt.xticks(rotation=45)
plt.show()
```



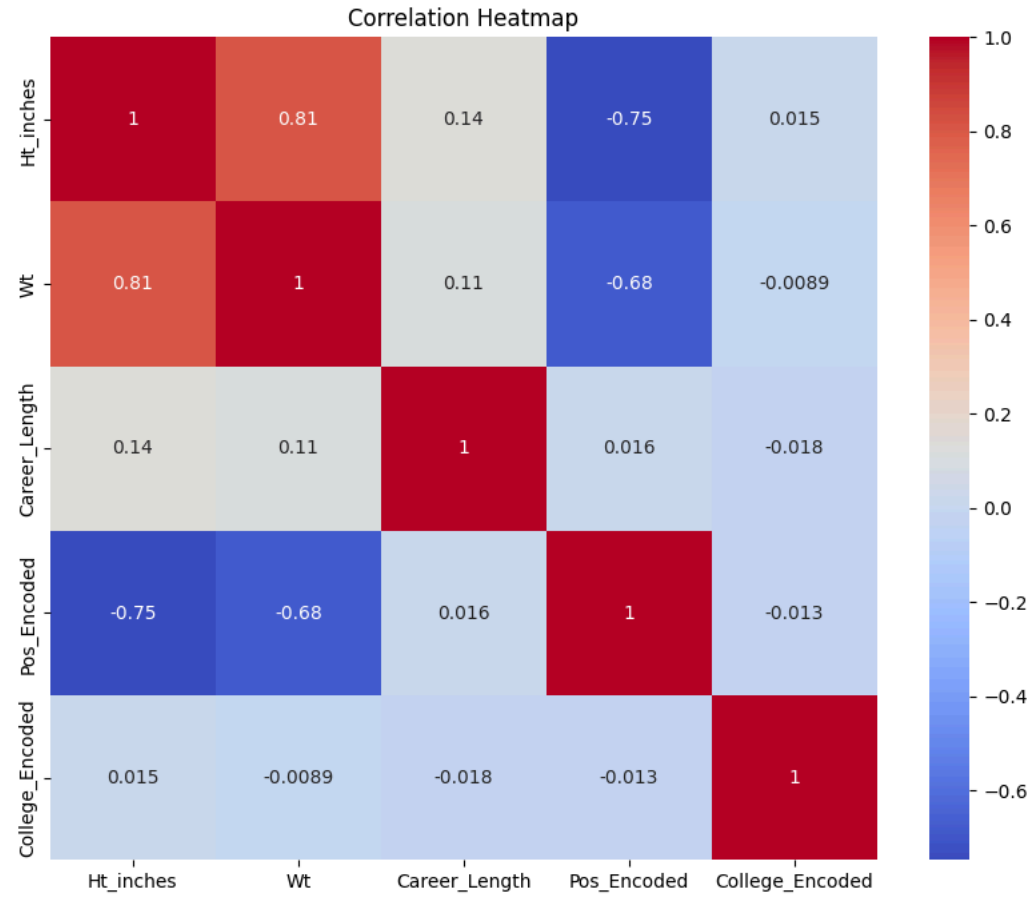
TOP COLLEGES BY PLAYER COUNT

```
top_colleges = df['Colleges'].value_counts().nlargest(10)
plt.figure(figsize=(12, 6))
top_colleges.plot(kind='bar')
plt.title('Top 10 Colleges by Player Count')
plt.xlabel('College')
plt.ylabel('Number of Players')
plt.xticks(rotation=45)
plt.show()
```



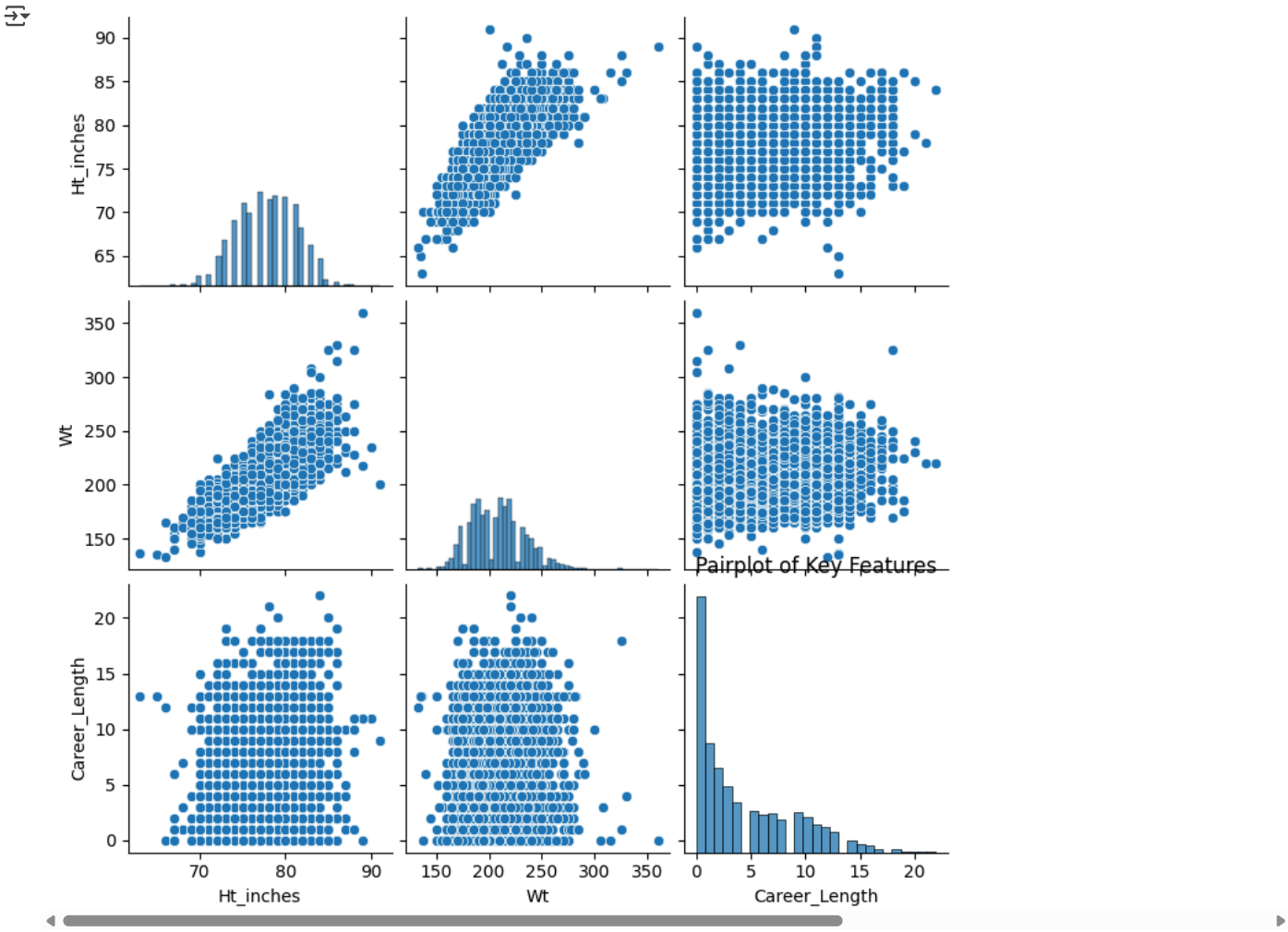
Correlation Heatmap of Numerical Features

```
numerical_features = ['Ht_inches', 'Wt', 'Career_Length', 'Pos_Encoded', 'College_Encoded']
corr_matrix = df[numerical_features].corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



Pairplot of Key Features

```
pair_features = ['Ht_inches', 'Wt', 'Career_Length']
sns.pairplot(df[pair_features])
plt.title('Pairplot of Key Features')
plt.show()
```



Regression Models for Career Length Prediction

```
X = df[['Ht_inches', 'Wt', 'Pos_Encoded', 'College_Encoded']]
y = df['Career_Length']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

LINEAR REGRESSION

```
lr_model = LinearRegression()
lr_model.fit(X_train, y_train)
lr_pred = lr_model.predict(X_test)
lr_mse = mean_squared_error(y_test, lr_pred)
lr_r2 = r2_score(y_test, lr_pred)
print(f'Linear Regression MSE: {lr_mse}, R2: {lr_r2}')
```

```
Linear Regression MSE: 18.30041215054787, R2: 0.04288401779568518
```

GRADIENT REGRESSION

```
gb_model = GradientBoostingRegressor(random_state=42)
gb_model.fit(X_train, y_train)
gb_pred = gb_model.predict(X_test)
```

```
gb_mse = mean_squared_error(y_test, gb_pred)
gb_r2 = r2_score(y_test, gb_pred)
```

```
Gradient Boosting MSE: 17.432291447304824, R2: 0.08828694056709907
```

Random Forest Regression

```
rf_model = RandomForestRegressor(random_state=42)
rf_model.fit(X_train, y_train)
rf_pred = rf_model.predict(X_test)
rf_mse = mean_squared_error(y_test, rf_pred)
rf_r2 = r2_score(y_test, rf_pred)
print(f'Random Forest MSE: {rf_mse}, R2: {rf_r2}')
```

```
Random Forest MSE: 19.825701025068195, R2: -0.03688895929751812
```

Visualization of Regression Results

```
plt.figure(figsize=(14, 8))
plt.scatter(y_test, rf_pred, alpha=0.5, label='Random Forest')
plt.scatter(y_test, gb_pred, alpha=0.5, label='Gradient Boosting')
plt.scatter(y_test, lr_pred, alpha=0.5, label='Linear Regression')
plt.xlabel('Actual Career Length')
plt.ylabel('Predicted Career Length')
plt.title('Actual vs. Predicted Career Length (Multiple Models)')
plt.legend()
plt.show()
```