In [520…
```
# Import pandas.

import pandas as pd
```

In [524…
```
data = pd.read_csv(r'C:\Users\HP\Desktop\Advance Data Analyst\6. The nuts and bolts of ML\2. Module 2\1. PACE in ML\Files\nba-players.csv')
data.head(10)
```

Out[524…

	Unnamed: 0		name	gp	min	pts	fgm	fga	fg	3p_made	3pa	3p	ftm	fta	ft	oreb	dreb	reb	ast	stl	blk	tov	target_5yrs
0	0	0	Brandon Ingram	36	27.4	7.4	2.6	7.6	34.7	0.5	2.1	25.0	1.6	2.3	69.9	0.7	3.4	4.1	1.9	0.4	0.4	1.3	0
1	1	1	Andrew Harrison	35	26.9	7.2	2.0	6.7	29.6	0.7	2.8	23.5	2.6	3.4	76.5	0.5	2.0	2.4	3.7	1.1	0.5	1.6	0
2	2	2	JaKarr Sampson	74	15.3	5.2	2.0	4.7	42.2	0.4	1.7	24.4	0.9	1.3	67.0	0.5	1.7	2.2	1.0	0.5	0.3	1.0	0
3	3	3	Malik Sealy	58	11.6	5.7	2.3	5.5	42.6	0.1	0.5	22.6	0.9	1.3	68.9	1.0	0.9	1.9	0.8	0.6	0.1	1.0	1
4	4	4	Matt Geiger	48	11.5	4.5	1.6	3.0	52.4	0.0	0.1	0.0	1.3	1.9	67.4	1.0	1.5	2.5	0.3	0.3	0.4	0.8	1
5	5	5	Tony Bennett	75	11.4	3.7	1.5	3.5	42.3	0.3	1.1	32.5	0.4	0.5	73.2	0.2	0.7	0.8	1.8	0.4	0.0	0.7	0
6	6	6	Don MacLean	62	10.9	6.6	2.5	5.8	43.5	0.0	0.1	50.0	1.5	1.8	81.1	0.5	1.4	2.0	0.6	0.2	0.1	0.7	1
7	7	7	Tracy Murray	48	10.3	5.7	2.3	5.4	41.5	0.4	1.5	30.0	0.7	0.8	87.5	0.8	0.9	1.7	0.2	0.2	0.1	0.7	1
8	8	8	Duane Cooper	65	9.9	2.4	1.0	2.4	39.2	0.1	0.5	23.3	0.4	0.5	71.4	0.2	0.6	0.8	2.3	0.3	0.0	1.1	0
9	9	9	Dave Johnson	42	8.5	3.7	1.4	3.5	38.3	0.1	0.3	21.4	1.0	1.4	67.8	0.4	0.7	1.1	0.3	0.2	0.0	0.7	0

In [526…
```
# Display number of rows, number of columns.

data.shape
```

Out[526… (1340, 22)

In [532…
```
# Display all column names.

data.columns
# Column Name       Column Description
# name              Name of NBA player
# gp                Number of games played
# min               Number of minutes played per game
# pts               Average number of points per game
# fgm               Average number of field goals made per game
# fga               Average number of field goal attempts per game
# fg                Average percent of field goals made per game
# 3p_made               Average number of three-point field goals made per game
# 3pa               Average number of three-point field goal attempts per game
# 3p                Average percent of three-point field goals made per game
# ftm               Average number of free throws made per game
# fta               Average number of free throw attempts per game
# ft                Average percent of free throws made per game
# oreb              Average number of offensive rebounds per game
# dreb              Average number of defensive rebounds per game
# reb               Average number of rebounds per game
# ast               Average number of assists per game
# stl               Average number of steals per game
# blk               Average number of blocks per game
# tov               Average number of turnovers per game
# target_5yrs       1 if career duration >= 5 yrs, 0 otherwise
```

Out[532…
```
Index(['Unnamed: 0', 'name', 'gp', 'min', 'pts', 'fgm', 'fga', 'fg', '3p_made',
       '3pa', '3p', 'ftm', 'fta', 'ft', 'oreb', 'dreb', 'reb', 'ast', 'stl',
       'blk', 'tov', 'target_5yrs'],
      dtype='object')
```

In [534…
```
# Use .info() to display a summary of the DataFrame.

data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1340 entries, 0 to 1339
Data columns (total 22 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Unnamed: 0   1340 non-null   int64
 1   name         1340 non-null   object
 2   gp           1340 non-null   int64
 3   min          1340 non-null   float64
 4   pts          1340 non-null   float64
 5   fgm          1340 non-null   float64
 6   fga          1340 non-null   float64
 7   fg           1340 non-null   float64
 8   3p_made      1340 non-null   float64
 9   3pa          1340 non-null   float64
 10  3p           1340 non-null   float64
 11  ftm          1340 non-null   float64
 12  fta          1340 non-null   float64
 13  ft           1340 non-null   float64
 14  oreb         1340 non-null   float64
 15  dreb         1340 non-null   float64
 16  reb          1340 non-null   float64
 17  ast          1340 non-null   float64
 18  stl          1340 non-null   float64
 19  blk          1340 non-null   float64
 20  tov          1340 non-null   float64
 21  target_5yrs  1340 non-null   int64
dtypes: float64(18), int64(3), object(1)
memory usage: 230.4+ KB
```

In [536…
```
# Display the number of missing values in each column.
# Check whether each value is missing.
#Aggregate the number of missing values per column.

data.isna().sum()
```

Out[536…
```
Unnamed: 0      0
name            0
gp              0
min             0
pts             0
fgm             0
fga             0
fg              0
3p_made         0
3pa             0
3p              0
ftm             0
fta             0
ft              0
oreb            0
dreb            0
reb             0
ast             0
stl             0
blk             0
tov             0
target_5yrs     0
dtype: int64
```

In [538…
```
# Display percentage (%) of values for each class (1, 0) represented in the target column of this dataset.

data["target_5yrs"].value_counts(normalize=True)*100
```

Out[538…
```
target_5yrs
1    62.014925
0    37.985075
Name: proportion, dtype: float64
```

In [540…
```
# Select the columns to proceed with and save the DataFrame in new variable `selected_data`.
# Include the target column, `target_5yrs`.

selected_data = data[["gp", "min", "pts", "fg", "3p", "ft", "reb", "ast", "stl", "blk", "tov", "target_5yrs"]]


# Display the first few rows.

selected_data.head()
```

Out[540…

	gp	min	pts	fg	3p	ft	reb	ast	stl	blk	tov	target_5yrs
0	36	27.4	7.4	34.7	25.0	69.9	4.1	1.9	0.4	0.4	1.3	0
1	35	26.9	7.2	29.6	23.5	76.5	2.4	3.7	1.1	0.5	1.6	0
2	74	15.3	5.2	42.2	24.4	67.0	2.2	1.0	0.5	0.3	1.0	0
3	58	11.6	5.7	42.6	22.6	68.9	1.9	0.8	0.6	0.1	1.0	1
4	48	11.5	4.5	52.4	0.0	67.4	2.5	0.3	0.3	0.4	0.8	1

In [542…
```
# Display the first few rows of `selected_data` for reference.

selected_data.head()
```

Out[542…

	gp	min	pts	fg	3p	ft	reb	ast	stl	blk	tov	target_5yrs
0	36	27.4	7.4	34.7	25.0	69.9	4.1	1.9	0.4	0.4	1.3	0
1	35	26.9	7.2	29.6	23.5	76.5	2.4	3.7	1.1	0.5	1.6	0
2	74	15.3	5.2	42.2	24.4	67.0	2.2	1.0	0.5	0.3	1.0	0
3	58	11.6	5.7	42.6	22.6	68.9	1.9	0.8	0.6	0.1	1.0	1
4	48	11.5	4.5	52.4	0.0	67.4	2.5	0.3	0.3	0.4	0.8	1

In [544…
```
# Extract two features that would help predict target_5yrs.
# Create a new variable named `extracted_data`.

# Make a copy of `selected_data`
extracted_data = selected_data.copy()

# Add a new column named `total_points`;
# Calculate total points earned by multiplying the number of games played by the average number of points earned per game
extracted_data["total_points"] = extracted_data["gp"] * extracted_data["pts"]

# Add a new column named `efficiency`. Calculate efficiency by dividing the total points earned by the total number
# of minutes played, which yields points per minute. (Note that `min` represents avg. minutes per game.)
extracted_data["efficiency"] = extracted_data["total_points"] / (extracted_data["min"] * extracted_data["gp"])

# Display the first few rows of `extracted_data` to confirm that the new columns were added.
extracted_data.head()
```

Out[544…

	gp	min	pts	fg	3p	ft	reb	ast	stl	blk	tov	target_5yrs	total_points	efficiency
0	36	27.4	7.4	34.7	25.0	69.9	4.1	1.9	0.4	0.4	1.3	0	266.4	0.270073
1	35	26.9	7.2	29.6	23.5	76.5	2.4	3.7	1.1	0.5	1.6	0	252.0	0.267658
2	74	15.3	5.2	42.2	24.4	67.0	2.2	1.0	0.5	0.3	1.0	0	384.8	0.339869
3	58	11.6	5.7	42.6	22.6	68.9	1.9	0.8	0.6	0.1	1.0	1	330.6	0.491379
4	48	11.5	4.5	52.4	0.0	67.4	2.5	0.3	0.3	0.4	0.8	1	216.0	0.391304

In [546…
```
# Remove any columns from `extracted_data` that are no longer needed.

# Remove `gp`, `pts`, and `min` from `extracted_data`.
extracted_data = extracted_data.drop(columns=["gp", "pts", "min"])

# Display the first few rows of `extracted_data` to ensure that column drops took place.

extracted_data.head()
```

Out[546…

	fg	3p	ft	reb	ast	stl	blk	tov	target_5yrs	total_points	efficiency
0	34.7	25.0	69.9	4.1	1.9	0.4	0.4	1.3	0	266.4	0.270073
1	29.6	23.5	76.5	2.4	3.7	1.1	0.5	1.6	0	252.0	0.267658
2	42.2	24.4	67.0	2.2	1.0	0.5	0.3	1.0	0	384.8	0.339869
3	42.6	22.6	68.9	1.9	0.8	0.6	0.1	1.0	1	330.6	0.491379
4	52.4	0.0	67.4	2.5	0.3	0.3	0.4	0.8	1	216.0	0.391304

In [548…
```
# Export the extracted data.

extracted_data.to_csv("extracted_nba_players_data.csv", index=0)
```

In [ ]:
```
# key takeaways

# It is important to check for class balance in a dataset, particularly in the context of feature engineering and predictive modeling.
# If the target column in a dataset has more than 90% of its values belonging to one class, it is recommended to redistribute the data; otherwise, once a model is trained on the imbalanced data and predictions are made, the prediction
# Feature selection involves choosing features that help predict the target variable and removing columns that may not be helpful for prediction.
# In this process, and throughout feature engineering, it is important to make ethical considerations.
# Feature transformation involves transforming features so that they are more usable for future modeling purposes, which includes encoding categorical features to turn them into numerical features.
# Feature extraction involves combining existing columns meaningfully to construct new features that would help improve prediction.

# What summary would you provide to stakeholders?
```

```
# The following attributes about player performance could help predict their NBA career duration and should be included in a presentation to stakeholders: field goals, three-point field goals, free throws, rebounds, assists, steals, b
# It would be important to explain that these attributes, along with a relevant dataset, will be used in the next stage of the project.
# At that point, a model will be built to predict a player's career duration. Insights gained will be shared with stakeholders once the project is complete.
# Stakeholders would also appreciate being provided with a timeline and key deliverables that they can expect to receive.
```