In [ ]:
```
#!pip install pybaseball
#!pip install iterative-stratification
```

```
Collecting iterative-stratification
  Downloading iterative_stratification-0.1.9-py3-none-any.whl.metadata (1.3
kB)
Requirement already satisfied: numpy in /opt/anaconda3/lib/python3.12/site-p
ackages (from iterative-stratification) (1.26.0)
Requirement already satisfied: scipy in /opt/anaconda3/lib/python3.12/site-p
ackages (from iterative-stratification) (1.13.1)
Requirement already satisfied: scikit-learn in /opt/anaconda3/lib/python3.1
2/site-packages (from iterative-stratification) (1.4.2)
Requirement already satisfied: joblib>=1.2.0 in /opt/anaconda3/lib/python3.1
2/site-packages (from scikit-learn->iterative-stratification) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/anaconda3/lib/py
thon3.12/site-packages (from scikit-learn->iterative-stratification) (3.6.0)
Downloading iterative_stratification-0.1.9-py3-none-any.whl (8.5 kB)
Installing collected packages: iterative-stratification
Successfully installed iterative-stratification-0.1.9
```

In [ ]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from pybaseball import  playerid_lookup, statcast_pitcher, pitching_stats
import datetime as dt
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split, GroupShuffleSplit, Str
from tensorflow.keras.preprocessing.sequence import pad_sequences
import keras_tuner as kt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Masking, LSTM, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.layers import Embedding, GRU, Dense, Dropout, Masking,
from tensorflow.keras.optimizers import Adam, RMSprop
import tensorflow as tf
from keras_tuner import RandomSearch
from iterstrat.ml_stratifiers import MultilabelStratifiedShuffleSplit
from sklearn.preprocessing import MultiLabelBinarizer
from collections import Counter
from sklearn.model_selection import train_test_split
from collections import defaultdict
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau

import re
import time
from datetime import timedelta
from pandas.errors import ParserError
from pybaseball import statcast, playerid_lookup
```

```
2025-08-13 17:19:48.864447: I tensorflow/core/platform/cpu_feature_guard.cc:
210] This TensorFlow binary is optimized to use available CPU instructions i
n performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild
TensorFlow with the appropriate compiler flags.
```

In [2]:

```
## Grabbing pitch by pitch data for all pitchers with minimum qualifying inn
pitchers_2025 = pitching_stats(2025)
pitchers_2025[['first', 'last']] = pitchers_2025['Name'].str.split(' ', n=1,
pitchers_2025['first'] = pitchers_2025['first'].str.lower()
pitchers_2025['last'] = pitchers_2025['last'].str.lower()

player_ids = []

for _, row in pitchers_2025.iterrows():
    try:
        info = playerid_lookup(row['last'], row['first'])  # Note: last name
        if not info.empty:
            # Sort if multiple results, and grab just the top one
            mlbam_id = info.sort_values(by='key_mlbam', ascending=False).ilo
            player_ids.append(int(mlbam_id))
    except Exception as e:
        print(f"Error for {row['Name']}: {e}")
        continue

start_date = "2025-02-18"
end_date = "2025-08-10"  # today's date

all_pitch_data = []

for pid in player_ids:
    try:
        data = statcast_pitcher(start_date, end_date, pid)
        if not data.empty:
            all_pitch_data.append(data)
    except Exception as e:
        print(f"Error for pitcher {pid}: {e}")
        continue

# Combine all into one DataFrame
combined_pitch_data = pd.concat(all_pitch_data, ignore_index=True)
combined_pitch_data
```

```
Gathering player lookup table. This may take a moment.
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
Gathering Player Data
```

Out[2]:

	pitch_type	game_date	release_speed	release_pos_x	release_pos_z	player
0	FF	2025-08-08	97.3	1.58	6.17	Skuba
1	FF	2025-08-08	98.4	1.52	6.06	Skuba
2	CH	2025-08-08	86.9	1.93	5.97	Skuba
3	SI	2025-08-08	95.6	1.78	6.00	Skuba
4	CH	2025-08-08	87.6	1.77	5.95	Skuba
...	...	...	...	...	...	
123520	NaN	2025-02-26	NaN	NaN	NaN	An
123521	NaN	2025-02-26	NaN	NaN	NaN	An
123522	NaN	2025-02-26	NaN	NaN	NaN	An
123523	NaN	2025-02-26	NaN	NaN	NaN	An
123524	NaN	2025-02-26	NaN	NaN	NaN	An

123525 rows × 118 columns

In [ ]:
```
columns_to_keep = [
    ## Pitcher + pitch identity
    'player_name', 'pitch_type', 'pitch_number', 'at_bat_number', 'game_pk',
    'batter', 'pitcher', 'stand', 'p_throws',

    ## Pitch physics
    'release_speed', 'release_pos_x', 'release_pos_y', 'release_pos_z',
    'release_spin_rate', 'spin_axis', 'pfx_x', 'pfx_z',
    'plate_x', 'plate_z', 'sz_top', 'sz_bot',

    ## Count / game state
    'balls', 'strikes', 'outs_when_up', 'inning', 'inning_topbot',
    'bat_score', 'fld_score',

    ## Runners on base
    'on_1b', 'on_2b', 'on_3b',

    ## Score context
    'home_score', 'away_score', 'home_score_diff', 'bat_score_diff',

    ## Strategy / sequencing dynamics
    'n_thruorder_pitcher',
    'n_priorpa_thisgame_player_at_bat',

    ## Outcome labels (still useful for filtering)
    'description', 'events'
]


pitch_data = combined_pitch_data[columns_to_keep]
pitch_data
```

Out[ ]:

	player_name	pitch_type	pitch_number	at_bat_number	game_pk	batter
0	Skubal, Tarik	FF	4	39	776832	545361
1	Skubal, Tarik	FF	3	39	776832	545361
2	Skubal, Tarik	CH	2	39	776832	545361
3	Skubal, Tarik	SI	1	39	776832	545361
4	Skubal, Tarik	CH	4	38	776832	694384
...	...	...	...	...	...	...
123520	Anderson, Tyler	NaN	1	3	779160	682829
123521	Anderson, Tyler	NaN	1	2	779160	666158
123522	Anderson, Tyler	NaN	3	1	779160	680574
123523	Anderson, Tyler	NaN	2	1	779160	680574
123524	Anderson, Tyler	NaN	1	1	779160	680574

123525 rows × 39 columns

In [4]:

```
## Imputing NaN for events because it means the batter is still up
pitch_data.loc[:, 'events'] = pitch_data['events'].fillna('batter still up')

pitch_data.loc[:, 'runner_on_1b'] = pitch_data['on_1b'].notna().astype(int)
pitch_data.loc[:, 'runner_on_2b'] = pitch_data['on_2b'].notna().astype(int)
pitch_data.loc[:, 'runner_on_3b'] = pitch_data['on_3b'].notna().astype(int)

pitch_data.drop(columns=['on_1b', 'on_2b', 'on_3b'], inplace=True)
```

```
/var/folders/82/cfm89vg521n6ydcbprwxcw3w0000gn/T/ipykernel_68889/1589671432.
py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  pitch_data.loc[:, 'runner_on_1b'] = pitch_data['on_1b'].notna().astype(in
t)
/var/folders/82/cfm89vg521n6ydcbprwxcw3w0000gn/T/ipykernel_68889/1589671432.
py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  pitch_data.loc[:, 'runner_on_2b'] = pitch_data['on_2b'].notna().astype(in
t)
/var/folders/82/cfm89vg521n6ydcbprwxcw3w0000gn/T/ipykernel_68889/1589671432.
py:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  pitch_data.loc[:, 'runner_on_3b'] = pitch_data['on_3b'].notna().astype(in
t)
/var/folders/82/cfm89vg521n6ydcbprwxcw3w0000gn/T/ipykernel_68889/1589671432.
py:8: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  pitch_data.drop(columns=['on_1b', 'on_2b', 'on_3b'], inplace=True)
```

In [5]:
```
## Adding sequencing IDs
pitch_data = pitch_data.copy()

pitch_data['sequence_id'] = (
    pitch_data.groupby(['game_pk', 'at_bat_number']).ngroup()
)

pitch_data['pitch_number'] = (
    pitch_data.groupby('sequence_id').cumcount() + 1
)
```

In [6]:
```
pitch_data
```

Out[6]:

	player_name	pitch_type	pitch_number	at_bat_number	game_pk	batter
0	Skubal, Tarik	FF	1	39	776832	545361
1	Skubal, Tarik	FF	2	39	776832	545361
2	Skubal, Tarik	CH	3	39	776832	545361
3	Skubal, Tarik	SI	4	39	776832	545361
4	Skubal, Tarik	CH	1	38	776832	694384
...	...	...	...	...	...	...
123520	Anderson, Tyler	NaN	5	3	779160	682829
123521	Anderson, Tyler	NaN	1	2	779160	666158
123522	Anderson, Tyler	NaN	1	1	779160	680574
123523	Anderson, Tyler	NaN	2	1	779160	680574
123524	Anderson, Tyler	NaN	3	1	779160	680574

123525 rows × 40 columns

In [7]:
```
pitch_data.isna().sum()
```

Out[7]:
```
player_name                        0
pitch_type                      4595
pitch_number                       0
at_bat_number                      0
game_pk                            0
batter                             0
pitcher                            0
stand                              0
p_throws                           0
release_speed                   4601
release_pos_x                   4601
release_pos_y                   4601
release_pos_z                   4601
release_spin_rate               5068
spin_axis                       5069
pfx_x                           4734
pfx_z                           4601
plate_x                         4601
plate_z                         4601
sz_top                          4601
sz_bot                          4601
balls                              0
strikes                            0
outs_when_up                       0
inning                             0
inning_topbot                      0
bat_score                          0
fld_score                          0
home_score                         0
away_score                         0
home_score_diff                    0
bat_score_diff                     0
n_thruorder_pitcher                0
n_priorpa_thisgame_player_at_bat   0
description                        0
events                             0
runner_on_1b                       0
runner_on_2b                       0
runner_on_3b                       0
sequence_id                        0
dtype: int64
```

In [8]:
```
## Dealing with missing data
critical_columns = ['pitch_type', 'release_speed', 'plate_x', 'plate_z', 're
'release_spin_rate', 'spin_axis', 'pfx_x', 'pfx_z', 'sz_top', 'sz_bot']

# Step 1: Find sequence_ids with ANY missing value in those columns
sequences_with_na = (
    pitch_data[critical_columns + ['sequence_id']]
    .groupby('sequence_id')
    .apply(lambda df: df[critical_columns].isnull().any().any())
)

# Step 2: Get only the sequence_ids with missing values
bad_sequences = sequences_with_na[sequences_with_na].index.tolist()
```

```
# Step 3: Filter out those sequences
pitch_data_cleaned = pitch_data[~pitch_data['sequence_id'].isin(bad_sequence
```

```
/var/folders/82/cfm89vg521n6ydcbprwxcw3w0000gn/T/ipykernel_68889/1683873461.
py:9: DeprecationWarning: DataFrameGroupBy.apply operated on the grouping co
lumns. This behavior is deprecated, and in a future version of pandas the gr
ouping columns will be excluded from the operation. Either pass `include_gro
ups=False` to exclude the groupings or explicitly select the grouping column
s after groupby to silence this warning.
  .apply(lambda df: df[critical_columns].isnull().any().any())
```

In [9]:
```
pitch_data_cleaned.isna().sum()
```

Out[9]:
```
player_name                       0
pitch_type                        0
pitch_number                      0
at_bat_number                     0
game_pk                           0
batter                            0
pitcher                           0
stand                             0
p_throws                          0
release_speed                     0
release_pos_x                     0
release_pos_y                     0
release_pos_z                     0
release_spin_rate                 0
spin_axis                         0
pfx_x                             0
pfx_z                             0
plate_x                           0
plate_z                           0
sz_top                            0
sz_bot                            0
balls                             0
strikes                           0
outs_when_up                      0
inning                            0
inning_topbot                     0
bat_score                         0
fld_score                         0
home_score                        0
away_score                        0
home_score_diff                   0
bat_score_diff                    0
n_thruorder_pitcher               0
n_priorpa_thisgame_player_at_bat  0
description                       0
events                            0
runner_on_1b                      0
runner_on_2b                      0
runner_on_3b                      0
sequence_id                       0
dtype: int64
```

Feature Engineering

One-Hot Encoding

In [10]:
```
columns_to_encode = ['inning_topbot', 'stand', 'p_throws']
def one_hot_encode (data, columns):
  for column in columns:
    data = pd.get_dummies(data, columns=[column], dtype=int)
  return data

pitch_data_encoded = one_hot_encode(pitch_data_cleaned, columns_to_encode)
```

In [ ]:
```
pitch_types = pitch_data_encoded['pitch_type']

## Initialize the label encoder
le = LabelEncoder()

## Fit the encoder and transform the pitch type strings to integers
pitch_data_encoded['pitch_type_encoded'] = le.fit_transform(pitch_types)

## Saving the mapping from pitch type string to integer
pitch_type_mapping = dict(zip(le.classes_, le.transform(le.classes_)))

print("Pitch type mapping:", pitch_type_mapping)

## 'pitch_type_encoded' is an integer column ready for model training
pitch_data_encoded = pitch_data_encoded.copy().drop(columns = ['pitch_type']
pitch_data_encoded
```

```
Pitch type mapping: {'CH': 0, 'CS': 1, 'CU': 2, 'FC': 3, 'FF': 4, 'FS': 5,
'KC': 6, 'PO': 7, 'SI': 8, 'SL': 9, 'ST': 10, 'SV': 11}
```

Out[ ]:

	player_name	pitch_number	at_bat_number	game_pk	batter	pitcher	rel
0	Skubal, Tarik	1	39	776832	545361	669373	
1	Skubal, Tarik	2	39	776832	545361	669373	
2	Skubal, Tarik	3	39	776832	545361	669373	
3	Skubal, Tarik	4	39	776832	545361	669373	
4	Skubal, Tarik	1	38	776832	694384	669373	
...	...	...	...	...	...	...	...
123222	Anderson, Tyler	3	6	778503	575929	542881	
123223	Anderson, Tyler	1	5	778503	663457	542881	
123224	Anderson, Tyler	2	5	778503	663457	542881	
123225	Anderson, Tyler	3	5	778503	663457	542881	
123226	Anderson, Tyler	4	5	778503	663457	542881	

118170 rows × 43 columns

Column Dropping

In [12]:
```
columns_to_drop = ['game_pk', 'pitcher', 'at_bat_number', 'bat_score', 'fld_
pitch = pitch_data_encoded.drop(columns = columns_to_drop)
```

Normalizing Data

In [ ]:
```
non_numeric_cols = pitch.select_dtypes(include=['object', 'category']).colum

## Specify non-scaled columns explicitly
non_scaled_cols = ['pitch_number', 'sequence_id', 'player_name', 'pitch_type

## Columns to exclude from scaling
all_excluded_cols = list(set(non_scaled_cols + non_numeric_cols))

## Scale only numeric columns
scaler = StandardScaler()
scaled_features = scaler.fit_transform(pitch.drop(columns=all_excluded_cols)

## Convert scaled values to DataFrame
scaled_df = pd.DataFrame(scaled_features,
                         columns=pitch.drop(columns=all_excluded_cols).colum
                         index=pitch.index)

## Reattach excluded (non-scaled) columns
```

```
pitch_scaled = pd.concat([scaled_df, pitch[all_excluded_cols]], axis=1)
cols = ['sequence_id', 'pitch_number', 'player_name'] + [col for col in pitc
pitch_scaled = pitch_scaled[cols]
pitch_scaled
```

Out[ ]:

	sequence_id	pitch_number	player_name	batter	release_speed	release_
0	567	1	Skubal, Tarik	-2.129812	1.373089	1.2
1	567	2	Skubal, Tarik	-2.129812	1.560861	1.1
2	567	3	Skubal, Tarik	-2.129812	-0.402210	1.4
3	567	4	Skubal, Tarik	-2.129812	1.082896	1.3
4	566	1	Skubal, Tarik	0.818188	-0.282719	1.3
...	...	...	...	...	...	
123222	27428	3	Anderson, Tyler	-1.525111	-0.214438	1.1
123223	27427	1	Anderson, Tyler	0.206384	-2.126298	1.1
123224	27427	2	Anderson, Tyler	0.206384	-1.016736	1.3
123225	27427	3	Anderson, Tyler	0.206384	-0.197368	1.0
123226	27427	4	Anderson, Tyler	0.206384	-0.299789	1.1

118170 rows × 37 columns

In [14]:
```
pitch_final = pitch_scaled.copy()
```

In [15]:
```
pitch_final = pitch_final.loc[:, ~pitch_final.columns.duplicated()]
pitch_final['player_name']
```

Out[15]:
```
0            Skubal, Tarik
1            Skubal, Tarik
2            Skubal, Tarik
3            Skubal, Tarik
4            Skubal, Tarik
                ...
123222    Anderson, Tyler
123223    Anderson, Tyler
123224    Anderson, Tyler
123225    Anderson, Tyler
123226    Anderson, Tyler
Name: player_name, Length: 118170, dtype: object
```

In [ ]:
```
## Step 1: Group sequences by (pitcher, sequence), ordered by pitch_number
grouped = pitch_final.sort_values(['sequence_id', 'pitch_number']).groupby([
pitcher_to_sequences = defaultdict(list)
```

```
for (pitcher, sequence_id), group in grouped:
    pitcher_to_sequences[pitcher].append((sequence_id, group))

## Step 2: Initializing storage
X_train_final, y_train_final, train_pitcher_ids = [], [], []
X_val, y_val, val_pitcher_ids = [], [], []
X_test, y_test, test_pitcher_ids = [], [], []

## Step 3: Per-pitcher split into train/val/test
for pitcher, seq_data in pitcher_to_sequences.items():
    if len(seq_data) < 3:  # Skip pitchers with too few sequences
        continue

    # Split: 64% train, 16% val, 20% test
    train_seqs, temp_seqs = train_test_split(seq_data, test_size=0.36, rando
    val_seqs, test_seqs = train_test_split(temp_seqs, test_size=5/9, random_

    def extract_sequences(seqs, X_bucket, y_bucket, pid_bucket):
        for _, group in seqs:
            pitches = group['pitch_type_encoded'].tolist()
            for i in range(1, len(pitches)):
                X_bucket.append(pitches[:i])
                y_bucket.append(pitches[i])
                pid_bucket.append(pitcher)

    extract_sequences(train_seqs, X_train_final, y_train_final, train_pitche
    extract_sequences(val_seqs, X_val, y_val, val_pitcher_ids)
    extract_sequences(test_seqs, X_test, y_test, test_pitcher_ids)
```

In [ ]:
```
## FILTERING SHORT SEQUENCES (≥ 3 pitches)
X_train_final, y_train_final, train_pitcher_ids = zip(*[
    (x, y, p) for x, y, p in zip(X_train_final, y_train_final, train_pitcher
])
X_val, y_val, val_pitcher_ids = zip(*[
    (x, y, p) for x, y, p in zip(X_val, y_val, val_pitcher_ids) if len(x) >=
])
X_test, y_test, test_pitcher_ids = zip(*[
    (x, y, p) for x, y, p in zip(X_test, y_test, test_pitcher_ids) if len(x)
])
```

In [18]:
```
## Padding sequences
max_len = max(len(seq) for seq in X_train_final)

X_train_pad = pad_sequences(X_train_final, padding='pre', maxlen=max_len)
X_val_pad = pad_sequences(X_val, padding='pre', maxlen=max_len)
X_test_pad = pad_sequences(X_test, padding='pre', maxlen=max_len)

y_train_arr = np.array(y_train_final)
y_val_arr = np.array(y_val)
y_test_arr = np.array(y_test)

vocab_size = np.max(y_train_arr) + 1  # or len(np.unique(y_train_arr))
```

Modeling

In [ ]:
```
vocab_size = np.max(y_train_arr) + 1
input_length = X_train_pad.shape[1]

def build_model(hp):
    model = Sequential()

    ## Embedding with masking
    model.add(Embedding(
        input_dim=vocab_size,
        output_dim=hp.Int('embed_dim', min_value=32, max_value=64, step=16),
        mask_zero=True
    ))

    ## First GRU layer returns sequences so I can stack another
    model.add(Bidirectional(GRU(
        units=hp.Int('gru_units_1', min_value=64, max_value=128, step=32),
        return_sequences=True,
        dropout=hp.Float('dropout_1', 0.2, 0.5, step=0.1),
        recurrent_dropout=hp.Float('recurrent_dropout_1', 0.1, 0.5, step=0.1
    )))

    ## Second GRU layer processes the sequence
    model.add(Bidirectional(GRU(
        units=hp.Int('gru_units_2', min_value=32, max_value=64, step=16),
        return_sequences=False,
        dropout=hp.Float('dropout_2', 0.2, 0.5, step=0.1),
        recurrent_dropout=hp.Float('recurrent_dropout_2', 0.1, 0.5, step=0.1
    )))

    ## Dense hidden layer before output
    model.add(Dense(
        hp.Int('dense_units', min_value=32, max_value=128, step=32),
        activation='relu'
    ))
    model.add(Dropout(hp.Float('dense_dropout', 0.2, 0.5, step=0.1)))

    ## Output layer
    model.add(Dense(vocab_size, activation='softmax'))

    model.compile(
        optimizer=hp.Choice('optimizer', ['adam', 'rmsprop']),
        loss='sparse_categorical_crossentropy',
        metrics=['accuracy']
    )

    return model
```

In [ ]:
```
## Initializing tuner
tuner = kt.RandomSearch(
    build_model,
    objective='val_accuracy',
    max_trials=5,
    executions_per_trial=1,
    directory='pitch_sequence_tuning',
    project_name='gru_rnn_random_v2',
    overwrite=True,
    seed=42
)
```

In [21]:
```
lr_scheduler = ReduceLROnPlateau(monitor='val_loss', factor=0.5,
                                 patience=2, min_lr=1e-5, verbose=1)

early_stop_cb = EarlyStopping(monitor='val_loss', patience=3, restore_best_w

tuner.search(X_train_pad, y_train_arr,
             validation_data=(X_val_pad, y_val_arr),
             epochs=10,
             batch_size=32,
             callbacks=[early_stop_cb, lr_scheduler])
```

```
Trial 5 Complete [00h 05m 35s]
val_accuracy: 0.38135191798210144

Best val_accuracy So Far: 0.3848345875740051
Total elapsed time: 00h 24m 49s
```

In [22]:
```
best_model = tuner.get_best_models(1)[0]
best_hp = tuner.get_best_hyperparameters(1)[0]
```

```
/opt/anaconda3/lib/python3.12/site-packages/keras/src/saving/saving_lib.py:7
57: UserWarning: Skipping variable loading for optimizer 'adam', because it
has 2 variables whereas the saved optimizer has 36 variables.
  saveable.load_own_variables(weights_store.get(inner_path))
```

Testing on pitchers

In [23]:
```
pitcher_scores = {}

for pitcher in set(test_pitcher_ids):
    indices = [i for i, pid in enumerate(test_pitcher_ids) if pid == pitcher
    if not indices:
        continue

    X_p = X_test_pad[indices]
    y_p = y_test_arr[indices]

    loss, acc = best_model.evaluate(X_p, y_p, verbose=0)
    num_pitch_types = len(np.unique(y_p))

    if num_pitch_types > 1:  # avoid divide-by-zero or meaningless 1-pitch c
        random_baseline = 1 / num_pitch_types
        predictability_score = (acc - random_baseline) / (1 - random_baselir
    else:
        predictability_score = 0  # or np.nan

    pitcher_scores[pitcher] = {
        'test_accuracy': acc,
        'num_pitch_types': num_pitch_types,
        'predictability_score': predictability_score
    }

predictability_df = pd.DataFrame.from_dict(pitcher_scores, orient='index')
predictability_df = predictability_df.sort_values("predictability_score", as
predictability_df.reset_index(inplace=True)
predictability_df.rename(columns={"index": "pitcher_name"}, inplace=True)
```

In [24]:
```
predictability_df.sort_values(by='predictability_score', ascending=False)
```

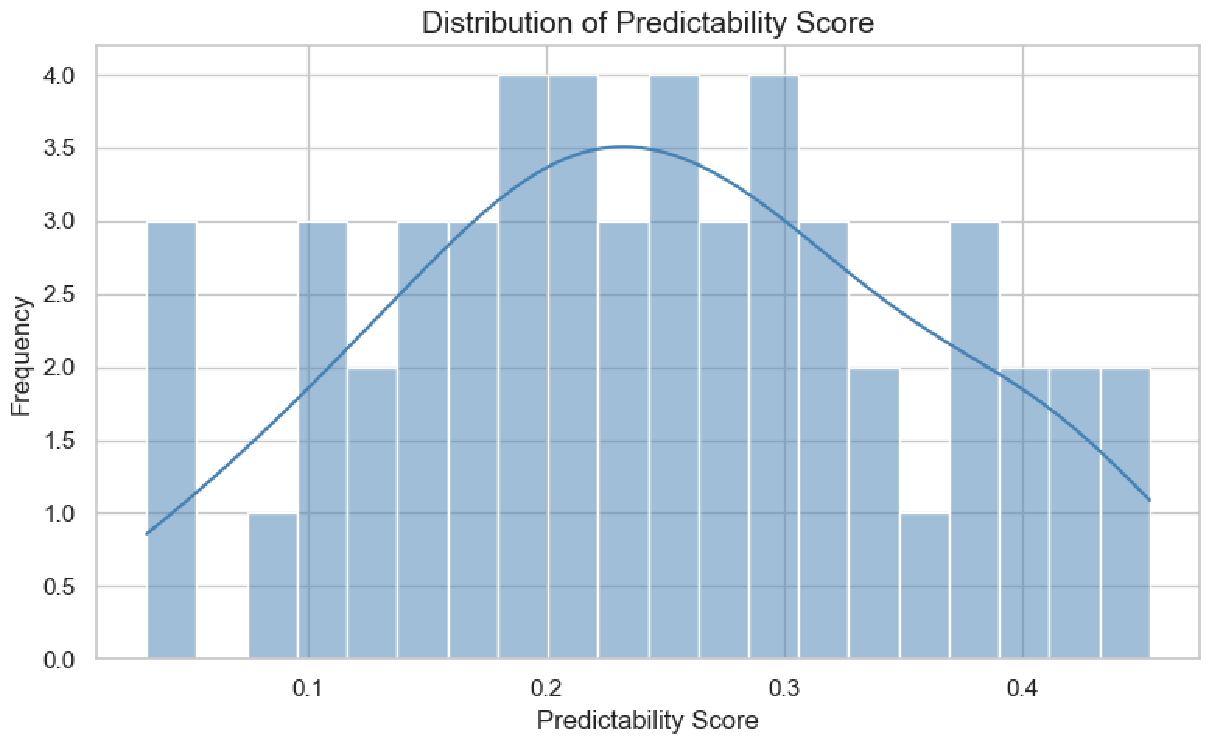
Out[24]:

	pitcher_name	test_accuracy	num_pitch_types	predictability_score
0	Pepiot, Ryan	0.544776	6	0.453731
1	Ryan, Joe	0.540000	6	0.448000
2	Gore, MacKenzie	0.542857	5	0.428571
3	Cease, Dylan	0.512500	6	0.415000
4	Brown, Hunter	0.508876	6	0.410651
5	Baz, Shane	0.523179	5	0.403974
6	Peralta, Freddy	0.540698	4	0.387597
7	Gallen, Zac	0.488372	6	0.386047
8	Parker, Mitchell	0.529412	4	0.372549
9	Ray, Robbie	0.493976	5	0.367470
10	Flaherty, Jack	0.472000	5	0.340000
11	Paddack, Chris	0.445205	6	0.334247
12	Springs, Jeffrey	0.457143	5	0.321429
13	Pivetta, Nick	0.416667	7	0.319444
14	Warren, Will	0.452381	5	0.315476
15	Abbott, Andrew	0.444444	5	0.305556
16	Gausman, Kevin	0.475000	4	0.300000
17	Wheeler, Zack	0.413408	6	0.296089
18	Falter, Bailey	0.432432	5	0.290541
19	Keller, Mitch	0.403727	6	0.284472
20	Crochet, Garrett	0.416149	5	0.270186
21	Yamamoto, Yoshinobu	0.389610	6	0.267532
22	Pallante, Andre	0.444444	4	0.259259
23	Williams, Gavin	0.381579	6	0.257895
24	Bibee, Tanner	0.377778	6	0.253333
25	Skenes, Paul	0.352601	7	0.244701
26	Woo, Bryan	0.391304	5	0.239130
27	Castillo, Luis	0.426829	4	0.235772
28	Anderson, Tyler	0.358974	6	0.230769
29	deGrom, Jacob	0.415094	4	0.220126
30	Allen, Logan	0.376000	5	0.220000
31	Kikuchi, Yusei	0.409836	4	0.213115

	pitcher_name	test_accuracy	num_pitch_types	predictability_score
32	Webb, Logan	0.363636	5	0.204545
33	Irvin, Jake	0.333333	6	0.200000
34	Bassitt, Chris	0.298013	8	0.197729
35	Pfaadt, Brandon	0.328947	6	0.194737
36	Kelly, Merrill	0.322148	6	0.186577
37	Bello, Brayan	0.342105	5	0.177632
38	Wacha, Michael	0.314685	6	0.177622
39	Singer, Brady	0.333333	5	0.166667
40	Kremer, Dean	0.326531	5	0.158163
41	Severino, Luis	0.294118	6	0.152941
42	Holmes, Clay	0.290780	6	0.148936
43	Littell, Zack	0.299145	5	0.123932
44	Sugano, Tomoyuki	0.267241	6	0.120690
45	Gray, Sonny	0.240310	7	0.113695
46	Peterson, David	0.284615	5	0.105769
47	Lugo, Seth	0.200000	9	0.100000
48	Valdez, Framber	0.316129	4	0.088172
49	Fried, Max	0.187500	7	0.052083
50	Lodolo, Nick	0.276730	4	0.035639
51	Skubal, Tarik	0.225806	5	0.032258

In [ ]:
```
## Set a consistent style
sns.set(style="whitegrid", palette="muted")

## Histogram of predictability_score
plt.figure(figsize=(8, 5))
sns.histplot(predictability_df['predictability_score'], bins=20, kde=True, c
plt.title('Distribution of Predictability Score', fontsize=14)
plt.xlabel('Predictability Score', fontsize=12)
plt.ylabel('Frequency', fontsize=12)
plt.tight_layout()
plt.show();
```

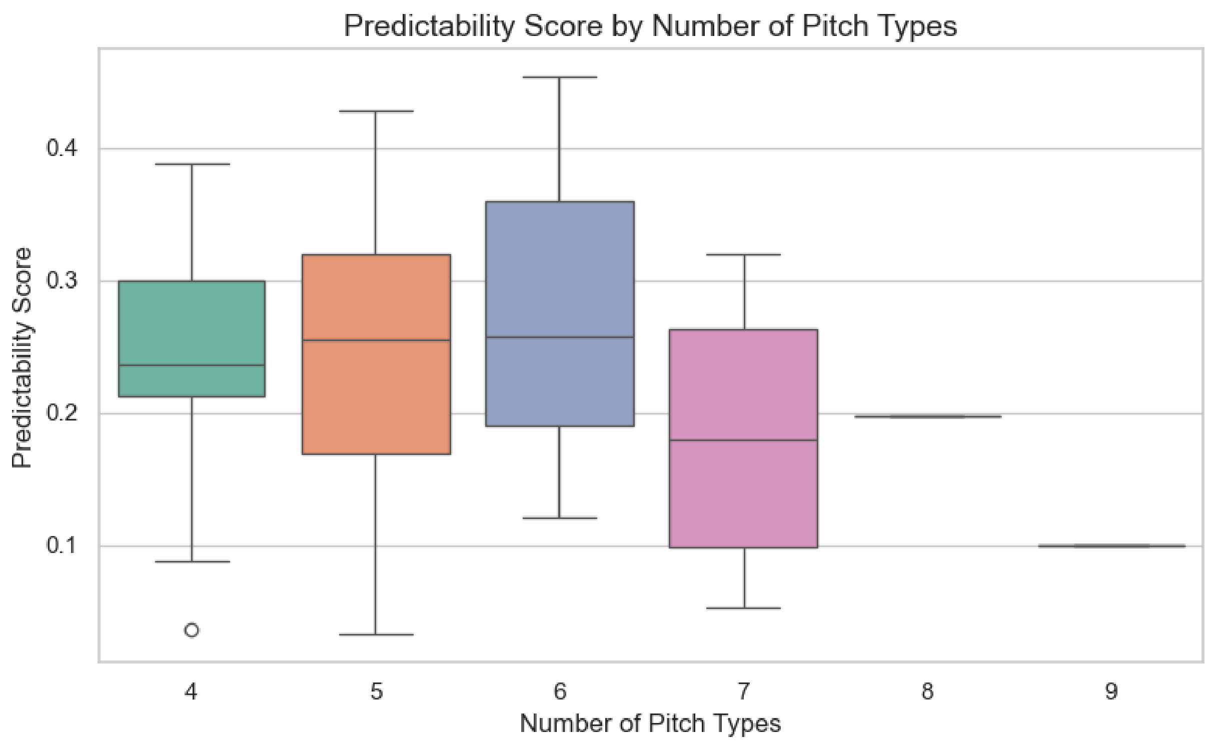


```
In [26]: plt.figure(figsize=(8, 5))
         sns.boxplot(x='num_pitch_types', y='predictability_score', data=predictabili
         plt.title('Predictability Score by Number of Pitch Types', fontsize=14)
         plt.xlabel('Number of Pitch Types', fontsize=12)
         plt.ylabel('Predictability Score', fontsize=12)
         plt.tight_layout()
         plt.show();
```

```
/var/folders/82/cfm89vg521n6ydcbprwxcw3w0000gn/T/ipykernel_68889/1570620204.
py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.boxplot(x='num_pitch_types', y='predictability_score', data=predictabi
lity_df, palette="Set2")
```



In [ ]:
```python
## Configuring winodow to get Statcast metrics
start_date = "2025-02-18"
end_date   = "2025-08-10"
WINDOW_DAYS = 7  # chunk size for Statcast pulls

## Helper functions

# Normalizaed Accuracy Gain (NAG) calculation for preditability score
def nag_uniform(acc: float, k: int) -> float:
    """Normalized Accuracy Gain with a uniform (1/k) baseline."""
    if k is None or k < 2 or acc is None or not (0.0 <= acc <= 1.0):
        return np.nan
    base = 1.0 / k
    return (acc - base) / (1.0 - base)

SUFFIXES = {"jr", "sr", "ii", "iii", "iv", "v"}

## Cleaning names
def clean_suffixes(s: str) -> str:
    s = re.sub(r"[^\w\s\-',]", " ", str(s)).strip()
    parts = [p for p in s.replace(".", "").split() if p.lower() not in SUFFI
    return " ".join(parts)

## Parsing names
def parse_name(raw: str):
    """Return (first, last) from 'Last, First' or 'First Last'."""
    s = clean_suffixes(raw)
    if "," in s:  # "Last, First"
        last, first = [p.strip() for p in s.split(",", 1)]
        return first, last
    toks = s.split()
    if len(toks) == 1:
        return toks[0], ""
    return toks[0], " ".join(toks[1:])

## Selecting best candidate from playerid_lookup results, just in case one i
def best_candidate(df: pd.DataFrame):
    """Pick a likely MLBAM id from playerid_lookup results."""
    if df is None or df.empty:
        return None
    for col in ["mlb_played_last", "mlb_played_first", "key_mlbam"]:
        if col not in df.columns:
            df[col] = np.nan
        df[col] = pd.to_numeric(df[col], errors="coerce")
    df = df.sort_values(
        ["mlb_played_last", "mlb_played_first", "key_mlbam"],
        ascending=[False, False, False],
        kind="mergesort"
    )
    val = df.iloc[0]["key_mlbam"]
    return int(val) if pd.notna(val) else None

## ID lookup function
def lookup_mlbam_id(name: str):
    """Lookup MLBAM id from a name string."""
    first, last = parse_name(name)
    if not first and not last:
        return None
    for L, R in [
        (last, first),
        (last.split()[-1] if last else "", first),
        (first, last),
    ]:
        try:
            c = playerid_lookup(L, R)
            pid = best_candidate(c)
            if pid:
                return pid
        except Exception:
            pass
    return None

## Chunked Statcast pull with retry/backoff
def statcast_all_safe(start_date, end_date, window_days=7, max_retries=3, ba
    """Chunked Statcast pull with simple retry/backoff."""
    start = pd.to_datetime(start_date)
    end = pd.to_datetime(end_date)
    frames = []
    cur = start
    while cur <= end:
        chunk_start = cur
        chunk_end = min(cur + timedelta(days=window_days - 1), end)
        tries = 0
        while tries < max_retries:
            try:
                df = statcast(chunk_start.strftime("%Y-%m-%d"),
                              chunk_end.strftime("%Y-%m-%d"))
                if df is not None and not df.empty:
                    frames.append(df)
                break
            except (ParserError, ValueError) as e:
                tries += 1
                wait = backoff ** (tries - 1)
                print(f"[statcast] parse error {chunk_start:%Y-%m-%d}–{chunk
                      f"retry {tries}/{max_retries} in {wait:.1f}s ({e})")
                time.sleep(wait)
            except Exception as e:
                tries += 1
                wait = backoff ** (tries - 1)
                print(f"[statcast] {type(e).__name__}: {e}; retry {tries}/{m
                time.sleep(wait)
        cur = chunk_end + timedelta(days=1)

    if not frames:
        return pd.DataFrame()
    out = pd.concat(frames, ignore_index=True)
    if "pitcher" in out.columns:
        out["pitcher"] = pd.to_numeric(out["pitcher"], errors="coerce").asty
    return out

## Computing barrel mask
def compute_barrel_mask(ev: pd.Series, la: pd.Series) -> pd.Series:
    """EV/LA band per Statcast-style heuristic (no need for a precomputed 'b
    ev = pd.to_numeric(ev, errors="coerce")
    la = pd.to_numeric(la, errors="coerce")
    mask_ev = ev >= 98
    min_la = np.maximum(26 - (ev - 98), 8)
    max_la = np.minimum(30 + (ev - 98), 50)
    mask_la = (la >= min_la) & (la <= max_la)
    return mask_ev & mask_la

# ==== BUILDING pred_metrics_df FROM predictability_df =====================
# Expect predictability_df to have: pitcher_name, num_pitch_types, test_accu
if 'predictability_df' not in globals():
    raise NameError("`predictability_df` not found. Create it first, then ru

pred_metrics_df = predictability_df.copy()

## Compute NAG (uniform baseline) as your predictability score
required = {'pitcher_name', 'num_pitch_types', 'test_accuracy'}
missing = required - set(pred_metrics_df.columns)
if missing:
    raise KeyError(f"predictability_df missing required columns: {missing}")

pred_metrics_df["predictability_score"] = [
    nag_uniform(a, k) for a, k in zip(pred_metrics_df["test_accuracy"],
                                      pred_metrics_df["num_pitch_types"])
]

## Map pitcher_name -> mlbam_id (for merging Statcast metrics)
pred_metrics_df["mlbam_id"] = (
    pred_metrics_df["pitcher_name"]
    .astype(str)
    .map(lambda nm: lookup_mlbam_id(nm))
).astype("Int64")
```

```
# ==== STATCAST METRICS FOR THE DATE WINDOW ==================================
print(f"[info] downloading Statcast for {start_date} → {end_date} (window={W
sc_all = statcast_all_safe(start_date, end_date, window_days=WINDOW_DAYS)

if not sc_all.empty:
    ## Contact-only rows for contact-quality metrics
    bbe_mask = sc_all.get("type").eq("X") if "type" in sc_all.columns else p
    bbe_df = sc_all.loc[bbe_mask].copy()

    ## xwOBA against on contact
    if "estimated_woba_using_speedangle" in bbe_df.columns:
        xw = (bbe_df.groupby("pitcher")["estimated_woba_using_speedangle"]
              .mean().rename("xwoba_against"))
    else:
        xw = pd.Series(dtype=float, name="xwoba_against")

    ## Barrel% against via EV/LA band
    if {"launch_speed", "launch_angle"}.issubset(bbe_df.columns):
        bbe_df["_barrel"] = compute_barrel_mask(bbe_df["launch_speed"], bbe_
        barrel_pct = (bbe_df.groupby("pitcher")["_barrel"].mean().mul(100.0)
                      .rename("barrel_percent_against"))
    else:
        barrel_pct = pd.Series(dtype=float, name="barrel_percent_against")

    metrics = (pd.concat([xw, barrel_pct], axis=1)
                 .reset_index()
                 .rename(columns={"pitcher": "mlbam_id"}))
else:
    metrics = pd.DataFrame(columns=["mlbam_id", "xwoba_against", "barrel_per

## Round/clean
for c, nd in [("xwoba_against", 6), ("barrel_percent_against", 3)]:
    if c in metrics.columns:
        metrics[c] = pd.to_numeric(metrics[c], errors="coerce").round(nd)

# ==== MERGE METRICS INTO pred_metrics_df ====================================
for c in ["xwoba_against", "barrel_percent_against"]:
    if c not in pred_metrics_df.columns:
        pred_metrics_df[c] = np.nan

left = pred_metrics_df.set_index("mlbam_id")
right = metrics.set_index("mlbam_id")[["xwoba_against", "barrel_percent_agai
left.update(right)  # only overwrite where right has non-nulls
pred_metrics_df = left.reset_index()

## Final polish / ordering
for c, nd in [("xwoba_against", 6), ("barrel_percent_against", 3), ("predict
    if c in pred_metrics_df.columns:
        pred_metrics_df[c] = pd.to_numeric(pred_metrics_df[c], errors="coerc

pred_metrics_df = pred_metrics_df.sort_values("predictability_score", ascend

pred_metrics_df
```

```
[info] downloading Statcast for 2025-02-18 → 2025-08-10 (window=7d)
This is a large query, it may take a moment to complete
Skipping offseason dates
0it [00:00, ?it/s]
This is a large query, it may take a moment to complete
Skipping offseason dates
0it [00:00, ?it/s]
This is a large query, it may take a moment to complete
Skipping offseason dates
0it [00:00, ?it/s]
This is a large query, it may take a moment to complete
Skipping offseason dates
  0%|          | 0/3 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 3/3 [00:03<00:00,  1.16s/it]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:02<00:12,  2.15s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
```

```
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:06<00:00,  1.14it/s]
This is a large query, it may take a moment to complete
```

```
 14%|█         | 1/7 [00:00<00:00,  6.01it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██▊       | 2/7 [00:01<00:02,  1.71it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 43%|████▎     | 3/7 [00:01<00:02,  1.94it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 57%|█████▋    | 4/7 [00:02<00:01,  1.79it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:04<00:00,  1.73it/s]
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/statcast.py:85: Futur
eWarning: The behavior of DataFrame concatenation with empty or all-NA entri
es is deprecated. In a future version, this will no longer exclude empty or
all-NA columns when determining the result dtypes. To retain the old behavio
r, exclude the relevant entries before the concat operation.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
```

```
This is a large query, it may take a moment to complete
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:06,  1.01s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|███       | 2/7 [00:01<00:04,  1.13it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 43%|████      | 3/7 [00:02<00:02,  1.41it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:04<00:00,  1.41it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:08,  1.35s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|███       | 2/7 [00:02<00:05,  1.02s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.31it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:07,  1.29s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██▊       | 2/7 [00:01<00:04,  1.15it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 43%|████▎     | 3/7 [00:03<00:05,  1.36s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 57%|█████▋    | 4/7 [00:04<00:03,  1.08s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
```

```
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:06<00:00,  1.09it/s]
This is a large query, it may take a moment to complete
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:08,  1.48s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██▊       | 2/7 [00:01<00:04,  1.11it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 43%|████▎     | 3/7 [00:02<00:03,  1.31it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.29it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:09,  1.58s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██▊       | 2/7 [00:02<00:04,  1.06it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 43%|████▎     | 3/7 [00:02<00:03,  1.20it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
```

```
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.28it/s]
This is a large query, it may take a moment to complete
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:08,  1.47s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██▊       | 2/7 [00:02<00:04,  1.02it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.24it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:06,  1.04s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██▊       | 2/7 [00:02<00:05,  1.12s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 43%|████▎     | 3/7 [00:02<00:03,  1.18it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.26it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:11,  1.84s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██▊       | 2/7 [00:02<00:06,  1.32s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
```

```
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.21it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:09,  1.66s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██▊       | 2/7 [00:02<00:04,  1.08it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 43%|████▎     | 3/7 [00:03<00:04,  1.02s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.19it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:11,  1.88s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██        | 2/7 [00:02<00:06,  1.21s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 43%|████      | 3/7 [00:03<00:03,  1.03it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
```

```
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.18it/s]
This is a large query, it may take a moment to complete
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:07,  1.20s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|███       | 2/7 [00:01<00:03,  1.29it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.36it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:10,  1.70s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
```

```
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.18it/s]
This is a large query, it may take a moment to complete
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:11,  1.89s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|███       | 2/7 [00:02<00:06,  1.20s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:06<00:00,  1.13it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
```

```
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:06<00:00,  1.16it/s]
This is a large query, it may take a moment to complete
```

```
 14%|█         | 1/7 [00:00<00:01,  4.96it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|██        | 2/7 [00:01<00:04,  1.03it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 43%|████      | 3/7 [00:02<00:03,  1.20it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.37it/s]
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/statcast.py:85: Futur
eWarning: The behavior of DataFrame concatenation with empty or all-NA entri
es is deprecated. In a future version, this will no longer exclude empty or
all-NA columns when determining the result dtypes. To retain the old behavio
r, exclude the relevant entries before the concat operation.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
```

```
This is a large query, it may take a moment to complete
 14%|█         | 1/7 [00:00<00:01,  4.73it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 57%|█████     | 4/7 [00:01<00:01,  2.11it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:03<00:00,  1.94it/s]
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/statcast.py:85: Futur
eWarning: The behavior of DataFrame concatenation with empty or all-NA entri
es is deprecated. In a future version, this will no longer exclude empty or
all-NA columns when determining the result dtypes. To retain the old behavio
r, exclude the relevant entries before the concat operation.
  final_data = pd.concat(dataframe_list, axis=0).convert_dtypes(convert_stri
ng=False)
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:01<00:08,  1.39s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
```

```
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:06<00:00,  1.07it/s]
This is a large query, it may take a moment to complete
```

```
  0%|          | 0/7 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 14%|█         | 1/7 [00:00<00:04,  1.48it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 29%|███       | 2/7 [00:02<00:06,  1.26s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
```

```
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 7/7 [00:05<00:00,  1.24it/s]
This is a large query, it may take a moment to complete
  0%|          | 0/6 [00:00<?, ?it/s]/opt/anaconda3/lib/python3.12/site-pack
ages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: errors='ign
ore' is deprecated and will raise in a future version. Use to_datetime witho
ut passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 17%|█▋        | 1/6 [00:00<00:04,  1.03it/s]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
 33%|███▎      | 2/6 [00:02<00:04,  1.11s/it]/opt/anaconda3/lib/python3.12/s
ite-packages/pybaseball/datahelpers/postprocessing.py:59: FutureWarning: err
ors='ignore' is deprecated and will raise in a future version. Use to_dateti
me without passing `errors` and catch exceptions explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
/opt/anaconda3/lib/python3.12/site-packages/pybaseball/datahelpers/postproce
ssing.py:59: FutureWarning: errors='ignore' is deprecated and will raise in
a future version. Use to_datetime without passing `errors` and catch excepti
ons explicitly instead
  data_copy[column] = data_copy[column].apply(pd.to_datetime, errors='ignor
e', format=date_format)
100%|██████████| 6/6 [00:05<00:00,  1.15it/s]
```

Out[ ]:

	mlbam_id	pitcher_name	test_accuracy	num_pitch_types	predictability_score	x
0	686752	Pepiot, Ryan	0.544776	6	0.453731	
1	657746	Ryan, Joe	0.540000	6	0.448000	
2	669022	Gore, MacKenzie	0.542857	5	0.428571	
3	656302	Cease, Dylan	0.512500	6	0.415000	
4	686613	Brown, Hunter	0.508876	6	0.410651	
5	669358	Baz, Shane	0.523179	5	0.403974	
6	642547	Peralta, Freddy	0.540698	4	0.387597	
7	668678	Gallen, Zac	0.488372	6	0.386047	
8	680730	Parker, Mitchell	0.529412	4	0.372549	
9	592662	Ray, Robbie	0.493976	5	0.367470	
10	656427	Flaherty, Jack	0.472000	5	0.340000	
11	663978	Paddack, Chris	0.445205	6	0.334247	
12	605488	Springs, Jeffrey	0.457143	5	0.321429	
13	601713	Pivetta, Nick	0.416667	7	0.319444	
14	701542	Warren, Will	0.452381	5	0.315476	
15	671096	Abbott, Andrew	0.444444	5	0.305556	
16	592332	Gausman, Kevin	0.475000	4	0.300000	
17	554430	Wheeler, Zack	0.413408	6	0.296089	
18	663559	Falter, Bailey	0.432432	5	0.290541	
19	656605	Keller, Mitch	0.403727	6	0.284472	
20	676979	Crochet, Garrett	0.416149	5	0.270186	
21	808967	Yamamoto, Yoshinobu	0.389610	6	0.267532	
22	669467	Pallante, Andre	0.444444	4	0.259259	
23	668909	Williams, Gavin	0.381579	6	0.257895	
24	676440	Bibee, Tanner	0.377778	6	0.253333	
25	694973	Skenes, Paul	0.352601	7	0.244701	

	mlbam_id	pitcher_name	test_accuracy	num_pitch_types	predictability_score	x
26	693433	Woo, Bryan	0.391304	5	0.239130	
27	622379	Castillo, Luis	0.426829	4	0.235772	
28	542881	Anderson, Tyler	0.358974	6	0.230769	
29	594798	deGrom, Jacob	0.415094	4	0.220126	
30	671106	Allen, Logan	0.376000	5	0.220000	
31	579328	Kikuchi, Yusei	0.409836	4	0.213115	
32	657277	Webb, Logan	0.363636	5	0.204545	
33	663623	Irvin, Jake	0.333333	6	0.200000	
34	605135	Bassitt, Chris	0.298013	8	0.197729	
35	694297	Pfaadt, Brandon	0.328947	6	0.194737	
36	518876	Kelly, Merrill	0.322148	6	0.186577	
37	678394	Bello, Brayan	0.342105	5	0.177632	
38	608379	Wacha, Michael	0.314685	6	0.177622	
39	663903	Singer, Brady	0.333333	5	0.166667	
40	665152	Kremer, Dean	0.326531	5	0.158163	
41	622663	Severino, Luis	0.294118	6	0.152941	
42	605280	Holmes, Clay	0.290780	6	0.148936	
43	641793	Littell, Zack	0.299145	5	0.123932	
44	608372	Sugano, Tomoyuki	0.267241	6	0.120690	
45	543243	Gray, Sonny	0.240310	7	0.113695	
46	656849	Peterson, David	0.284615	5	0.105769	
47	607625	Lugo, Seth	0.200000	9	0.100000	
48	664285	Valdez, Framber	0.316129	4	0.088172	
49	608331	Fried, Max	0.187500	7	0.052083	
50	666157	Lodolo, Nick	0.276730	4	0.035639	
51	669373	Skubal, Tarik	0.225806	5	0.032258	

```
In [29]: pred_metrics_df = pred_metrics_df.dropna(axis=1, how='any')
```

```
pred_metrics_df
```

Out[29]:

	mlbam_id	pitcher_name	test_accuracy	num_pitch_types	predictability_score	x
0	686752	Pepiot, Ryan	0.544776	6	0.453731	
1	657746	Ryan, Joe	0.540000	6	0.448000	
2	669022	Gore, MacKenzie	0.542857	5	0.428571	
3	656302	Cease, Dylan	0.512500	6	0.415000	
4	686613	Brown, Hunter	0.508876	6	0.410651	
5	669358	Baz, Shane	0.523179	5	0.403974	
6	642547	Peralta, Freddy	0.540698	4	0.387597	
7	668678	Gallen, Zac	0.488372	6	0.386047	
8	680730	Parker, Mitchell	0.529412	4	0.372549	
9	592662	Ray, Robbie	0.493976	5	0.367470	
10	656427	Flaherty, Jack	0.472000	5	0.340000	
11	663978	Paddack, Chris	0.445205	6	0.334247	
12	605488	Springs, Jeffrey	0.457143	5	0.321429	
13	601713	Pivetta, Nick	0.416667	7	0.319444	
14	701542	Warren, Will	0.452381	5	0.315476	
15	671096	Abbott, Andrew	0.444444	5	0.305556	
16	592332	Gausman, Kevin	0.475000	4	0.300000	
17	554430	Wheeler, Zack	0.413408	6	0.296089	
18	663559	Falter, Bailey	0.432432	5	0.290541	
19	656605	Keller, Mitch	0.403727	6	0.284472	
20	676979	Crochet, Garrett	0.416149	5	0.270186	
21	808967	Yamamoto, Yoshinobu	0.389610	6	0.267532	
22	669467	Pallante, Andre	0.444444	4	0.259259	
23	668909	Williams, Gavin	0.381579	6	0.257895	
24	676440	Bibee, Tanner	0.377778	6	0.253333	
25	694973	Skenes, Paul	0.352601	7	0.244701	

	mlbam_id	pitcher_name	test_accuracy	num_pitch_types	predictability_score	x
26	693433	Woo, Bryan	0.391304	5	0.239130	
27	622379	Castillo, Luis	0.426829	4	0.235772	
28	542881	Anderson, Tyler	0.358974	6	0.230769	
29	594798	deGrom, Jacob	0.415094	4	0.220126	
30	671106	Allen, Logan	0.376000	5	0.220000	
31	579328	Kikuchi, Yusei	0.409836	4	0.213115	
32	657277	Webb, Logan	0.363636	5	0.204545	
33	663623	Irvin, Jake	0.333333	6	0.200000	
34	605135	Bassitt, Chris	0.298013	8	0.197729	
35	694297	Pfaadt, Brandon	0.328947	6	0.194737	
36	518876	Kelly, Merrill	0.322148	6	0.186577	
37	678394	Bello, Brayan	0.342105	5	0.177632	
38	608379	Wacha, Michael	0.314685	6	0.177622	
39	663903	Singer, Brady	0.333333	5	0.166667	
40	665152	Kremer, Dean	0.326531	5	0.158163	
41	622663	Severino, Luis	0.294118	6	0.152941	
42	605280	Holmes, Clay	0.290780	6	0.148936	
43	641793	Littell, Zack	0.299145	5	0.123932	
44	608372	Sugano, Tomoyuki	0.267241	6	0.120690	
45	543243	Gray, Sonny	0.240310	7	0.113695	
46	656849	Peterson, David	0.284615	5	0.105769	
47	607625	Lugo, Seth	0.200000	9	0.100000	
48	664285	Valdez, Framber	0.316129	4	0.088172	
49	608331	Fried, Max	0.187500	7	0.052083	
50	666157	Lodolo, Nick	0.276730	4	0.035639	
51	669373	Skubal, Tarik	0.225806	5	0.032258	

In [30]:
```
pred_metrics_df['barrel_percent_against'] = pred_metrics_df['barrel_percent_
```

```
pred_metrics_df
```

Out[30]:

	mlbam_id	pitcher_name	test_accuracy	num_pitch_types	predictability_score	x
0	686752	Pepiot, Ryan	0.544776	6	0.453731	
1	657746	Ryan, Joe	0.540000	6	0.448000	
2	669022	Gore, MacKenzie	0.542857	5	0.428571	
3	656302	Cease, Dylan	0.512500	6	0.415000	
4	686613	Brown, Hunter	0.508876	6	0.410651	
5	669358	Baz, Shane	0.523179	5	0.403974	
6	642547	Peralta, Freddy	0.540698	4	0.387597	
7	668678	Gallen, Zac	0.488372	6	0.386047	
8	680730	Parker, Mitchell	0.529412	4	0.372549	
9	592662	Ray, Robbie	0.493976	5	0.367470	
10	656427	Flaherty, Jack	0.472000	5	0.340000	
11	663978	Paddack, Chris	0.445205	6	0.334247	
12	605488	Springs, Jeffrey	0.457143	5	0.321429	
13	601713	Pivetta, Nick	0.416667	7	0.319444	
14	701542	Warren, Will	0.452381	5	0.315476	
15	671096	Abbott, Andrew	0.444444	5	0.305556	
16	592332	Gausman, Kevin	0.475000	4	0.300000	
17	554430	Wheeler, Zack	0.413408	6	0.296089	
18	663559	Falter, Bailey	0.432432	5	0.290541	
19	656605	Keller, Mitch	0.403727	6	0.284472	
20	676979	Crochet, Garrett	0.416149	5	0.270186	
21	808967	Yamamoto, Yoshinobu	0.389610	6	0.267532	
22	669467	Pallante, Andre	0.444444	4	0.259259	
23	668909	Williams, Gavin	0.381579	6	0.257895	
24	676440	Bibee, Tanner	0.377778	6	0.253333	
25	694973	Skenes, Paul	0.352601	7	0.244701	

	mlbam_id	pitcher_name	test_accuracy	num_pitch_types	predictability_score	x
26	693433	Woo, Bryan	0.391304	5	0.239130	
27	622379	Castillo, Luis	0.426829	4	0.235772	
28	542881	Anderson, Tyler	0.358974	6	0.230769	
29	594798	deGrom, Jacob	0.415094	4	0.220126	
30	671106	Allen, Logan	0.376000	5	0.220000	
31	579328	Kikuchi, Yusei	0.409836	4	0.213115	
32	657277	Webb, Logan	0.363636	5	0.204545	
33	663623	Irvin, Jake	0.333333	6	0.200000	
34	605135	Bassitt, Chris	0.298013	8	0.197729	
35	694297	Pfaadt, Brandon	0.328947	6	0.194737	
36	518876	Kelly, Merrill	0.322148	6	0.186577	
37	678394	Bello, Brayan	0.342105	5	0.177632	
38	608379	Wacha, Michael	0.314685	6	0.177622	
39	663903	Singer, Brady	0.333333	5	0.166667	
40	665152	Kremer, Dean	0.326531	5	0.158163	
41	622663	Severino, Luis	0.294118	6	0.152941	
42	605280	Holmes, Clay	0.290780	6	0.148936	
43	641793	Littell, Zack	0.299145	5	0.123932	
44	608372	Sugano, Tomoyuki	0.267241	6	0.120690	
45	543243	Gray, Sonny	0.240310	7	0.113695	
46	656849	Peterson, David	0.284615	5	0.105769	
47	607625	Lugo, Seth	0.200000	9	0.100000	
48	664285	Valdez, Framber	0.316129	4	0.088172	
49	608331	Fried, Max	0.187500	7	0.052083	
50	666157	Lodolo, Nick	0.276730	4	0.035639	
51	669373	Skubal, Tarik	0.225806	5	0.032258	

In [31]:
```
pred_metrics_df['PS_xwoba'] = pred_metrics_df['predictability_score'] + pred
pred_metrics_df['PS_barrel'] = pred_metrics_df['predictability_score'] + pre
```

```
pred_metrics_df['PS_xwoba'] = pred_metrics_df['PS_xwoba'].round(3)
pred_metrics_df['PS_barrel'] = pred_metrics_df['PS_barrel'].round(3)

pred_metrics_df
```

Out[31]:

	mlbam_id	pitcher_name	test_accuracy	num_pitch_types	predictability_score	x
0	686752	Pepiot, Ryan	0.544776	6	0.453731	
1	657746	Ryan, Joe	0.540000	6	0.448000	
2	669022	Gore, MacKenzie	0.542857	5	0.428571	
3	656302	Cease, Dylan	0.512500	6	0.415000	
4	686613	Brown, Hunter	0.508876	6	0.410651	
5	669358	Baz, Shane	0.523179	5	0.403974	
6	642547	Peralta, Freddy	0.540698	4	0.387597	
7	668678	Gallen, Zac	0.488372	6	0.386047	
8	680730	Parker, Mitchell	0.529412	4	0.372549	
9	592662	Ray, Robbie	0.493976	5	0.367470	
10	656427	Flaherty, Jack	0.472000	5	0.340000	
11	663978	Paddack, Chris	0.445205	6	0.334247	
12	605488	Springs, Jeffrey	0.457143	5	0.321429	
13	601713	Pivetta, Nick	0.416667	7	0.319444	
14	701542	Warren, Will	0.452381	5	0.315476	
15	671096	Abbott, Andrew	0.444444	5	0.305556	
16	592332	Gausman, Kevin	0.475000	4	0.300000	
17	554430	Wheeler, Zack	0.413408	6	0.296089	
18	663559	Falter, Bailey	0.432432	5	0.290541	
19	656605	Keller, Mitch	0.403727	6	0.284472	
20	676979	Crochet, Garrett	0.416149	5	0.270186	
21	808967	Yamamoto, Yoshinobu	0.389610	6	0.267532	
22	669467	Pallante, Andre	0.444444	4	0.259259	
23	668909	Williams, Gavin	0.381579	6	0.257895	
24	676440	Bibee, Tanner	0.377778	6	0.253333	
25	694973	Skenes, Paul	0.352601	7	0.244701	

	mlbam_id	pitcher_name	test_accuracy	num_pitch_types	predictability_score	x
26	693433	Woo, Bryan	0.391304	5	0.239130	
27	622379	Castillo, Luis	0.426829	4	0.235772	
28	542881	Anderson, Tyler	0.358974	6	0.230769	
29	594798	deGrom, Jacob	0.415094	4	0.220126	
30	671106	Allen, Logan	0.376000	5	0.220000	
31	579328	Kikuchi, Yusei	0.409836	4	0.213115	
32	657277	Webb, Logan	0.363636	5	0.204545	
33	663623	Irvin, Jake	0.333333	6	0.200000	
34	605135	Bassitt, Chris	0.298013	8	0.197729	
35	694297	Pfaadt, Brandon	0.328947	6	0.194737	
36	518876	Kelly, Merrill	0.322148	6	0.186577	
37	678394	Bello, Brayan	0.342105	5	0.177632	
38	608379	Wacha, Michael	0.314685	6	0.177622	
39	663903	Singer, Brady	0.333333	5	0.166667	
40	665152	Kremer, Dean	0.326531	5	0.158163	
41	622663	Severino, Luis	0.294118	6	0.152941	
42	605280	Holmes, Clay	0.290780	6	0.148936	
43	641793	Littell, Zack	0.299145	5	0.123932	
44	608372	Sugano, Tomoyuki	0.267241	6	0.120690	
45	543243	Gray, Sonny	0.240310	7	0.113695	
46	656849	Peterson, David	0.284615	5	0.105769	
47	607625	Lugo, Seth	0.200000	9	0.100000	
48	664285	Valdez, Framber	0.316129	4	0.088172	
49	608331	Fried, Max	0.187500	7	0.052083	
50	666157	Lodolo, Nick	0.276730	4	0.035639	
51	669373	Skubal, Tarik	0.225806	5	0.032258	

In [33]:
```
pred_metrics_df.to_csv('pred_metrics_df.csv', index=False)
```

```
In [67]: PS_xwoba_sorted = predictability_df.sort_values(by='PS_xwoba', ascending=Fal
         PS_barrel_sorted = predictability_df.sort_values(by='PS_barrel', ascending=F
         PS_hardhit_sorted = predictability_df.sort_values(by='PS_hardhit', ascending

         PS_xwoba_sorted
```

Out[67]:

	pitcher_name	num_pitch_types	predictability_score	xwoba_against	barrel_perce
1	Parker, Mitchell	4	0.134454	0.406460	
24	Pfaadt, Brandon	6	0.071096	0.454582	
6	Gore, MacKenzie	5	0.103896	0.406893	
10	Flaherty, Jack	5	0.097600	0.409486	
0	Peralta, Freddy	4	0.135174	0.368117	
9	Kikuchi, Yusei	4	0.098361	0.400926	
3	Gausman, Kevin	4	0.118750	0.375786	
18	Gallen, Zac	6	0.080426	0.411007	
4	deGrom, Jacob	4	0.116352	0.374496	
7	Castillo, Luis	4	0.103659	0.381107	
2	Pallante, Andre	4	0.123016	0.355365	
5	Baz, Shane	5	0.104636	0.372687	
13	Pepiot, Ryan	6	0.089552	0.382486	
20	Paddack, Chris	6	0.075342	0.395974	
23	Webb, Logan	5	0.071212	0.399249	
21	Cease, Dylan	6	0.073034	0.391251	
12	Ryan, Joe	6	0.093333	0.367629	
30	Pivetta, Nick	7	0.063187	0.397346	
44	Sugano, Tomoyuki	6	0.038793	0.419835	
16	Crochet, Garrett	5	0.085714	0.371970	
28	Littell, Zack	5	0.064706	0.392055	
8	Ray, Robbie	5	0.101205	0.355090	
11	Woo, Bryan	5	0.093548	0.362837	
31	Rea, Colin	7	0.061368	0.394387	
17	Springs, Jeffrey	5	0.085714	0.369151	
37	Irvin, Jake	6	0.053571	0.401653	

	pitcher_name	num_pitch_types	predictability_score	xwoba_against	barrel_perce
25	Singer, Brady	5	0.066667	0.386240	
15	Valdez, Framber	4	0.087097	0.365321	
14	Brown, Hunter	6	0.088757	0.361275	
34	Anderson, Tyler	6	0.059829	0.388540	
45	Gray, Sonny	7	0.037714	0.407810	
22	Lodolo, Nick	4	0.072327	0.370025	
47	Lugo, Seth	9	0.020635	0.418111	
40	Kelly, Merrill	6	0.048098	0.389808	
29	Kremer, Dean	5	0.063946	0.371058	
35	Peterson, David	5	0.055385	0.378786	
33	Williams, Gavin	6	0.060307	0.366822	
19	Wheeler, Zack	6	0.079861	0.346228	
26	Bibee, Tanner	6	0.065432	0.360772	
32	Keller, Mitch	6	0.061077	0.363839	
43	Bassitt, Chris	8	0.039735	0.384286	
41	Severino, Luis	6	0.047794	0.374660	
36	Wacha, Michael	6	0.054779	0.360070	
42	Skubal, Tarik	5	0.047742	0.366245	
38	Holmes, Clay	6	0.050827	0.360492	
27	Yamamoto, Yoshinobu	6	0.064935	0.329947	
46	Fried, Max	7	0.034053	0.355995	
39	Skenes, Paul	7	0.049546	0.322224	

```
In [68]: PS_barrel_sorted
```

Out[68]:

	pitcher_name	num_pitch_types	predictability_score	xwoba_against	barrel_perce
0	Peralta, Freddy	4	0.135174	0.368117	
1	Parker, Mitchell	4	0.134454	0.406460	
10	Flaherty, Jack	5	0.097600	0.409486	
4	deGrom, Jacob	4	0.116352	0.374496	
24	Pfaadt, Brandon	6	0.071096	0.454582	
3	Gausman, Kevin	4	0.118750	0.375786	
12	Ryan, Joe	6	0.093333	0.367629	
5	Baz, Shane	5	0.104636	0.372687	
2	Pallante, Andre	4	0.123016	0.355365	
6	Gore, MacKenzie	5	0.103896	0.406893	
18	Gallen, Zac	6	0.080426	0.411007	
11	Woo, Bryan	5	0.093548	0.362837	
9	Kikuchi, Yusei	4	0.098361	0.400926	
8	Ray, Robbie	5	0.101205	0.355090	
13	Pepiot, Ryan	6	0.089552	0.382486	
21	Cease, Dylan	6	0.073034	0.391251	
28	Littell, Zack	5	0.064706	0.392055	
20	Paddack, Chris	6	0.075342	0.395974	
34	Anderson, Tyler	6	0.059829	0.388540	
30	Pivetta, Nick	7	0.063187	0.397346	
37	Irvin, Jake	6	0.053571	0.401653	
17	Springs, Jeffrey	5	0.085714	0.369151	
25	Singer, Brady	5	0.066667	0.386240	
16	Crochet, Garrett	5	0.085714	0.371970	
22	Lodolo, Nick	4	0.072327	0.370025	

	pitcher_name	num_pitch_types	predictability_score	xwoba_against	barrel_perce
15	Valdez, Framber	4	0.087097	0.365321	
23	Webb, Logan	5	0.071212	0.399249	
31	Rea, Colin	7	0.061368	0.394387	
19	Wheeler, Zack	6	0.079861	0.346228	
14	Brown, Hunter	6	0.088757	0.361275	
44	Sugano, Tomoyuki	6	0.038793	0.419835	
7	Castillo, Luis	4	0.103659	0.381107	
26	Bibee, Tanner	6	0.065432	0.360772	
33	Williams, Gavin	6	0.060307	0.366822	
29	Kremer, Dean	5	0.063946	0.371058	
40	Kelly, Merrill	6	0.048098	0.389808	
36	Wacha, Michael	6	0.054779	0.360070	
32	Keller, Mitch	6	0.061077	0.363839	
45	Gray, Sonny	7	0.037714	0.407810	
43	Bassitt, Chris	8	0.039735	0.384286	
42	Skubal, Tarik	5	0.047742	0.366245	
38	Holmes, Clay	6	0.050827	0.360492	
27	Yamamoto, Yoshinobu	6	0.064935	0.329947	
35	Peterson, David	5	0.055385	0.378786	
47	Lugo, Seth	9	0.020635	0.418111	
41	Severino, Luis	6	0.047794	0.374660	
46	Fried, Max	7	0.034053	0.355995	
39	Skenes, Paul	7	0.049546	0.322224	

```
In [69]: PS_hardhit_sorted
```

Out[69]:

	pitcher_name	num_pitch_types	predictability_score	xwoba_against	barrel_perce
1	Parker, Mitchell	4	0.134454	0.406460	
15	Valdez, Framber	4	0.087097	0.365321	
4	deGrom, Jacob	4	0.116352	0.374496	
8	Ray, Robbie	5	0.101205	0.355090	
24	Pfaadt, Brandon	6	0.071096	0.454582	
37	Irvin, Jake	6	0.053571	0.401653	
6	Gore, MacKenzie	5	0.103896	0.406893	
18	Gallen, Zac	6	0.080426	0.411007	
10	Flaherty, Jack	5	0.097600	0.409486	
2	Pallante, Andre	4	0.123016	0.355365	
20	Paddack, Chris	6	0.075342	0.395974	
35	Peterson, David	5	0.055385	0.378786	
3	Gausman, Kevin	4	0.118750	0.375786	
9	Kikuchi, Yusei	4	0.098361	0.400926	
13	Pepiot, Ryan	6	0.089552	0.382486	
11	Woo, Bryan	5	0.093548	0.362837	
5	Baz, Shane	5	0.104636	0.372687	
12	Ryan, Joe	6	0.093333	0.367629	
22	Lodolo, Nick	4	0.072327	0.370025	
40	Kelly, Merrill	6	0.048098	0.389808	
30	Pivetta, Nick	7	0.063187	0.397346	
32	Keller, Mitch	6	0.061077	0.363839	
21	Cease, Dylan	6	0.073034	0.391251	
28	Littell, Zack	5	0.064706	0.392055	
41	Severino, Luis	6	0.047794	0.374660	
47	Lugo, Seth	9	0.020635	0.418111	
23	Webb, Logan	5	0.071212	0.399249	

	pitcher_name	num_pitch_types	predictability_score	xwoba_against	barrel_perce
25	Singer, Brady	5	0.066667	0.386240	
0	Peralta, Freddy	4	0.135174	0.368117	
31	Rea, Colin	7	0.061368	0.394387	
16	Crochet, Garrett	5	0.085714	0.371970	
33	Williams, Gavin	6	0.060307	0.366822	
38	Holmes, Clay	6	0.050827	0.360492	
27	Yamamoto, Yoshinobu	6	0.064935	0.329947	
17	Springs, Jeffrey	5	0.085714	0.369151	
26	Bibee, Tanner	6	0.065432	0.360772	
39	Skenes, Paul	7	0.049546	0.322224	
44	Sugano, Tomoyuki	6	0.038793	0.419835	
19	Wheeler, Zack	6	0.079861	0.346228	
29	Kremer, Dean	5	0.063946	0.371058	
45	Gray, Sonny	7	0.037714	0.407810	
34	Anderson, Tyler	6	0.059829	0.388540	
14	Brown, Hunter	6	0.088757	0.361275	
46	Fried, Max	7	0.034053	0.355995	
43	Bassitt, Chris	8	0.039735	0.384286	
36	Wacha, Michael	6	0.054779	0.360070	
7	Castillo, Luis	4	0.103659	0.381107	
42	Skubal, Tarik	5	0.047742	0.366245	

```
In [34]: np.mean(pred_metrics_df['predictability_score'])
```

```
Out[34]: 0.24673940384615384
```

```
In [35]: np.mean(pred_metrics_df['PS_xwoba'])
```

```
Out[35]: 0.6255961538461539
```

```
In [36]: np.mean(pred_metrics_df['PS_barrel'])
```

```
Out[36]:  0.32315384615384607
```