betbot

Release 0.7.2

Hermann Krumrey

Jul 30, 2021

CONTENTS

1 betbot package **3**
- 1.1 Subpackages . . . 3
- 1.2 Submodules . . . 10
- 1.3 betbot.main module . . . 10
- 1.4 Module contents . . . 10

2 betbot **11**

3 Indices and tables **13**

Python Module Index **15**

Index **17**

Contents:

CHAPTER

ONE

BETBOT PACKAGE

1.1 Subpackages

1.1.1 betbot.api package

Submodules

betbot.api.ApiConnection module

class betbot.api.ApiConnection.**ApiConnection**(*username: str*, *password: str*, *url: str*)

Bases: `object`

Class that handles API calls to the bundesliga-tippspiel instance

__init__(*username: str*, *password: str*, *url: str*)

Intitializes the API connection :param username: The username on bundesliga-tippspiel :param password: The password for that user :param url: The base url to the bundesliga-tippspiel instance

property auth_headers: Dict[str, str]

Returns Authorization headers for API calls

authorized() → bool

Checks if the stored API key is valid :return: True if valid, False if not (for example because it expired)

execute_api_call(*endpoint: str*, *method: str*, *authorization_required: bool = False*, *json_data: Optional[Dict[str, Any]] = None*) → Dict[str, Any]

Executes an API call :param endpoint: The API endpoint :param method: The request method :param authorization_required: Whether authoirzation is required :param json_data: The JSON data to send :return: The response JSON

get_active_leagues() → List[Tuple[str, int]]

Returns A list of tuples of leagues and seasons of active leagues

get_current_matchday_matches(*league: str*, *season: int*) → List[*betbot.api.Match.Match*]

Retrieves a list of matches for the current matchday :param league: The league to retrieve matches for :param season: The season to retrieve matches for :return: The list of matches

get_league_table(*season: str*, *league: int*) → List[str]

Retrieves the current league table order :param season: The league for which to retrieve the league table :param league: The season for which to retrieve the league table :return: The team abbreviations in order of their league rankings

login() → bool
Retrieves an API Key using a username and password if no key has been retrieved yet or if the existing key has expired :return: True if the login was successfulle

logout()
Logs out the bot by deleting the API key :return: None

place_bets(*bets: List[*betbot.api.Bet.Bet*]*)
Places a list of bets :param bets: The bets to place :return: None

betbot.api.Bet module

class betbot.api.Bet.**Bet**(*match:* betbot.api.Match.Match, *home_score: int*, *away_score: int*)
Bases: object

Class that encapsulates Bet information

__init__(*match:* betbot.api.Match.Match, *home_score: int*, *away_score: int*)
Initializes the Bet :param match: The associated match :param home_score: The score bet on the home team :param away_score: The score bet on the away team

to_dict() → Dict[str, Union[int, str]]

Returns A dictionary that can be used to place the bet using the API

betbot.api.Match module

class betbot.api.Match.**Match**(*league: str*, *season: str*, *matchday: int*, *home_team: str*, *away_team: str*, *finished: bool*)
Bases: object

Class that encapsulates information about a Match

__init__(*league: str*, *season: str*, *matchday: int*, *home_team: str*, *away_team: str*, *finished: bool*)
Initializes the Match :param _id: The ID of the match :param matchday: The matchday of the match :param home_team: The name of the home team :param away_team: The name of the away team :param finished: Whether the match is already finished or not

classmethod from_json(*json_data: Dict[str, Any]*)
Generates a Match object from JSON data :param json_data: The JSON data :return: The generated Match

Module contents

1.1.2 betbot.data package

Submodules

betbot.data.FootballDataCoUk module

class betbot.data.FootballDataCoUk.**FootballDataUk**(*data_path: str*)
Bases: object

Class that handles retrieving data from football-data.co.uk

TEAM_MAP = {'Augsburg': 'FCA', 'Bayern Munich': 'FCB', 'Bielefeld': 'DSC', 'Bochum': 'BOC', 'Darmstadt': 'D98', 'Dortmund': 'BVB', 'Dresden': 'SGD', 'Ein Frankfurt': 'SGE', 'Erzgebirge Aue': 'AUE', 'FC Koln': 'KOE', 'Fortuna Dusseldorf': 'F95', 'Freiburg': 'SCF', 'Greuther Furth': 'SGF', 'Hamburg': 'HSV', 'Hannover': 'H96', 'Hansa Rostock': 'FCH', 'Heidenheim': 'HDH', 'Hertha': 'BSC', 'Hoffenheim': 'TSG', 'Holstein Kiel': 'KIE', 'Ingolstadt': 'FCI', 'Karlsruhe': 'KSC', 'Leverkusen': 'B04', "M'gladbach": 'BMG', 'Mainz': 'M05', 'Nurnberg': 'FCN', 'Paderborn': 'SCP', 'RB Leipzig': 'RBL', 'Regensburg': 'SSV', 'Sandhausen': 'SVS', 'Schalke 04': 'S04', 'St Pauli': 'STP', 'Stuttgart': 'VFB', 'Union Berlin': 'FCU', 'Werder Bremen': 'BRE', 'Wolfsburg': 'WOB'}

__init__(*data_path: str*)
Initializes the object :param data_path: The path in which to store the data files

download_history()
Download history for the first and second bundesliga :return: None

get_history_matches() → List[Dict[str, Union[str, int, float]]]
Retrieves data on historical bundesliga and bundesliga 2 matches :return: A list of match dictionaries

get_odds() → Dict[Tuple[str, str], Tuple[float, float, float]]
Loads currently available odds :return: The current odds (home, draw, away) mapped to home/away teams

static load_history_file(*history_file: str*) → List[Dict[str, str]]
Loads the contents of a history file :param history_file: The history file to load :return: A list of match dictionaries

load_matches(*data_file: str*) → List[Dict[str, Union[str, int, float]]]
Loads match data from a single CSV file :param data_file: The path to the CSV file :return: The match data

betbot.data.OddsPortal module

class betbot.data.OddsPortal.**OddsPortal**
Bases: object

Class that handles retrieving bet data using oddsportal.com

TEAM_MAP = {'Arminia Bielefeld': 'DSC', 'Aue': 'AUE', 'Augsburg': 'FCA', 'B. Monchengladbach': 'BMG', 'Bayer Leverkusen': 'B04', 'Bayern Munich': 'FCB', 'Bochum': 'BOC', 'Braunschweig': 'BRA', 'Darmstadt': 'D98', 'Dortmund': 'BVB', 'Dortmund II': 'BV2', 'Duisburg': 'MSV', 'Dusseldorf': 'F95', 'Eintracht Frankfurt': 'SGE', 'FC Koln': 'KOE', 'Freiburg': 'SCF', 'Freiburg II': 'SC2', 'Greuther Furth': 'SGF', 'Hallescher': 'HFC', 'Hamburger SV': 'HSV', 'Hannover': 'H96', 'Hansa Rostock': 'FCH', 'Heidenheim': 'HDH', 'Hertha Berlin': 'BSC', 'Hoffenheim': 'TSG', 'Holstein Kiel': 'KIE', 'Ingolstadt': 'FCI', 'Kaiserslautern': 'FCK', 'Karlsruher': 'KSC', 'Magdeburg': 'FCM', 'Mainz': 'M05', 'Mannheim': 'MAN', 'Meppen': 'SVM', 'Munich 1860': 'MÜN', 'Nurnberg': 'FCN', 'Paderborn': 'SCP', 'RB Leipzig': 'RBL', 'Regensburg': 'SSV', 'SG Dynamo Dresden': 'SGD', 'Saarbrucken': 'FCS', 'Sandhausen': 'SVS', 'Schalke': 'S04', 'St. Pauli': 'STP', 'Stuttgart': 'VFB', 'TSV Havelse': 'HAV', 'Turkgucu Munchen': 'TÜR', 'Union Berlin': 'FCU', 'Verl': 'SCV', 'VfL Osnabruck': 'OSN', 'Viktoria Berlin': 'VBE', 'Viktoria Koln': 'VKO', 'Wehen': 'WIE', 'Werder Bremen': 'BRE', 'Wolfsburg': 'WOB', 'Wurzburger Kickers': 'WÜR', 'Zwickau': 'ZWI'}

__init__()
Initializes the oddsportal scraper

get_odds(*league: str*) → Dict[Tuple[str, str], Tuple[float, float, float]]
Retrieves the odds for upcoming matches from oddsportal.com :param league: The league to search for :return: Odds for the matches (home, draw, odds)

Module contents

1.1.3 betbot.prediction package

Submodules

betbot.prediction.BettingOddsPredictor module

class betbot.prediction.BettingOddsPredictor.**BettingOddsPredictor**(*api:* betbot.api.ApiConnection.ApiConnection, *league: str*, *season: int*)

Bases: *betbot.prediction.Predictor.Predictor*

Class that determinalistically predicts matches based on Tipico quotes

generate_scores(*home_odds: float*, *draw_odds: float*, *away_odds: float*) → Tuple[int, int]
Generates a scoreline based on the quote data :param home_odds: The odds that the home team wins :param draw_odds: The odds that the match ends in a draw :param away_odds: The odds that the away team wins :return: The generated bet

classmethod name() → str

Returns The name of the predictor

predict(*matches: List[*betbot.api.Match.Match*]*) → List[*betbot.api.Bet.Bet*]
Performs the prediction :param matches: The matches to predict :return: The predictions as Bet objects

betbot.prediction.DrawPredictor module

class betbot.prediction.DrawPredictor.**DrawPredictor**(*api:* betbot.api.ApiConnection.ApiConnection, *league: str*, *season: int*)

Bases: *betbot.prediction.Predictor.Predictor*

Class that always predicts 0:0, 1:1, 2:2 or 3:3

classmethod name() → str

Returns The name of the predictor

predict(*matches: List[*betbot.api.Match.Match*]*) → List[*betbot.api.Bet.Bet*]
Performs the prediction :param matches: The matches to predict :return: The predictions as Bet objects

betbot.prediction.HomeTeamPredictor module

class betbot.prediction.HomeTeamPredictor.**HomeTeamPredictor**(*api:* betbot.api.ApiConnection.ApiConnection, *league: str*, *season: int*)

Bases: *betbot.prediction.Predictor.Predictor*

Class that always predicts 2:1 for the home team

classmethod name() → str

Returns The name of the predictor

predict(*matches: List[*betbot.api.Match.Match*]*) → List[*betbot.api.Bet.Bet*]

Performs the prediction :param matches: The matches to predict :return: The predictions as Bet objects

betbot.prediction.LeagueTablePredictor module

class betbot.prediction.LeagueTablePredictor.**LeagueTablePredictor**(*api:* betbot.api.ApiConnection.ApiConnection, *league: str*, *season: int*)

Bases: *betbot.prediction.Predictor.Predictor*

Class that always 2:1 for the team higher up in the league table

classmethod name() → str

Returns The name of the predictor

predict(*matches: List[*betbot.api.Match.Match*]*) → List[*betbot.api.Bet.Bet*]

Performs the prediction :param matches: The matches to predict :return: The predictions as Bet objects

betbot.prediction.NameAndOddsPredictor module

class betbot.prediction.NameAndOddsPredictor.**NameAndOddsPredictor**(*api:* betbot.api.ApiConnection.ApiConnection, *league: str*, *season: int*)

Bases: *betbot.prediction.SKLearnPredictor.SKLearnPredictor*

scikit-learn powered predictor that uses the name and betting odds

interpret_results(*home_result: float*, *away_result: float*) → Tuple[int, int]

Interprets the raw results :param home_result: The home goals result :param away_result: The away goals result :return: The home goals, the away goals

classmethod name() → str

Returns The name of the predictor

betbot.prediction.NamePredictor module

class betbot.prediction.NamePredictor.**NamePredictor**(*api:* betbot.api.ApiConnection.ApiConnection, *league: str*, *season: int*)

Bases: *betbot.prediction.SKLearnPredictor.SKLearnPredictor*

scikit-learn powered predictor that uses only the team names

encode_result(*home_score: int*, *away_score: int*) → numpy.ndarray

Encodes a result vector This is done as a normalization step. :param home_score: The home score to encode :param away_score: The away score to encode :return: The encoded result vector

interpret_results(*home_result: float*, *away_result: float*) → Tuple[int, int]

Interprets the raw results :param home_result: The home goals result :param away_result: The away goals result :return: The home goals, the away goals

classmethod name() → str

Returns The name of the predictor

predict_match(*match:* betbot.api.Match.Match) → Optional[Tuple[int, int]]

Predicts the result of a single match using the trained model :param match: The match to predict :return: The home goals and away goals or None

if no prediction took place

vectorize(*match_data: Dict[str, Union[str, float]]*) → numpy.ndarray

Defines how a match is vectorized :param match_data: The match data to vectorize :return: The vector for the match

betbot.prediction.Predictor module

class betbot.prediction.Predictor.**Predictor**(*api:* betbot.api.ApiConnection.ApiConnection, *league: str*, *season: int*)

Bases: `object`

Class that specifies required methods for predictor objects

__init__(*api:* betbot.api.ApiConnection.ApiConnection, *league: str*, *season: int*)

Initializes the model directory if it does not exist :param api: The bundesliga-tippspiel API connection :param league: The league for which to predict matches :param season: The season for which to predict matches

classmethod name() → str

Returns The name of the predictor

predict(*matches: List[*betbot.api.Match.Match*]*) → List[*betbot.api.Bet.Bet*]

Performs the prediction :param matches: The matches to predict :return: The predictions as Bet objects

betbot.prediction.RandomPredictor module

class betbot.prediction.RandomPredictor.**RandomPredictor**(*api: betbot.api.ApiConnection.ApiConnection, league: str, season: int*)

Bases: *betbot.prediction.Predictor.Predictor*

Class that always predicts random results

classmethod name() → str

Returns The name of the predictor

predict(*matches: List[betbot.api.Match.Match]*) → List[*betbot.api.Bet.Bet*]

Performs the prediction :param matches: The matches to predict :return: The predictions as Bet objects

betbot.prediction.SKLearnPredictor module

class betbot.prediction.SKLearnPredictor.**SKLearnPredictor**(*api: betbot.api.ApiConnection.ApiConnection, league: str, season: int*)

Bases: *betbot.prediction.Predictor.Predictor*

Abstract class that defines how a scikit-learn-based predictor should operate

__init__(*api: betbot.api.ApiConnection.ApiConnection, league: str, season: int*)

Initializes the scikit-learn model :param api: The bundesliga-tippspiel API connection :param league: The league for which to predict matches :param season: The season for which to predict matches

encode_result(*home_score: int, away_score: int*) → numpy.ndarray

Encodes a result vector This is done as a normalization step. :param home_score: The home score to encode :param away_score: The away score to encode :return: The encoded result vector

interpret_results(*home_result: float, away_result: float*) → Tuple[int, int]

Interprets the raw results :param home_result: The home goals result :param away_result: The away goals result :return: The home goals, the away goals

property odds: Dict[Tuple[str, str], Tuple[float, float, float]]

Retrieves current odds using a mix of football-data.co.uk and oddsportal.com :return: The odds for each match in the selected league

predict(*matches: List[betbot.api.Match.Match]*) → List[*betbot.api.Bet.Bet*]

Performs the prediction :param matches: The matches to predict :return: The predictions as Bet objects

predict_match(*match: betbot.api.Match.Match*) → Optional[Tuple[int, int]]

Predicts the result of a single match using the trained model :param match: The match to predict :return: The home goals and away goals or None

if no prediction took place

classmethod regressor() → sklearn.neural_network._multilayer_perceptron.MLPRegressor

Defines the regressor used during the prediction process :return: The predictor

train()

Trains the prediction model

vectorize(*match_data: Dict[str, Union[str, float]]*) → numpy.ndarray

Defines how a match is vectorized :param match_data: The match data to vectorize :return: The vector for the match

Module contents

1.1.4 betbot.test package

Module contents

1.2 Submodules

1.3 betbot.main module

`betbot.main.main`(*predictor_name: str*, *username: str*, *password: str*, *url: str*, *loop: bool = False*)

The main function of the betbot Predicts results and places bets accordingly :param predictor_name: The name of the predictor to use :param username: The username for bundesliga-tippspiel :param password: The password for bundesliga-tippspiel :param url: The base URL for the bundesliga-tippspiel instance :param loop: If true will place new bets once an hour :return: None

`betbot.main.multi_main`(*url: str*, *config: List[Tuple[str, str, str]]*, *loop: bool = False*)

Predicts using multiple procedures simultaneously :param url: The base URL for the bundesliga-tippspiel site :param config: The predictor names and credentials for the API :param loop: Whether or not to loop this once an hour :return: None

1.4 Module contents

`betbot.sentry_dsn = 'https://bf9dc130ba4c4284a42813a0032b63fe@sentry.namibsun.net/21'`

The sentry DSN used for exception logging

CHAPTER

TWO

BETBOT

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

b

betbot, 10
betbot.api, 4
betbot.api.ApiConnection, 3
betbot.api.Bet, 4
betbot.api.Match, 4
betbot.data, 6
betbot.data.FootballDataCoUk, 4
betbot.data.OddsPortal, 5
betbot.main, 10
betbot.prediction, 10
betbot.prediction.BettingOddsPredictor, 6
betbot.prediction.DrawPredictor, 6
betbot.prediction.HomeTeamPredictor, 7
betbot.prediction.LeagueTablePredictor, 7
betbot.prediction.NameAndOddsPredictor, 7
betbot.prediction.NamePredictor, 8
betbot.prediction.Predictor, 8
betbot.prediction.RandomPredictor, 9
betbot.prediction.SKLearnPredictor, 9
betbot.test, 10

INDEX

Symbols

__init__() (*betbot.api.ApiConnection.ApiConnection method*), 3
__init__() (*betbot.api.Bet.Bet method*), 4
__init__() (*betbot.api.Match.Match method*), 4
__init__() (*betbot.data.FootballDataCoUk.FootballDataUk method*), 5
__init__() (*betbot.data.OddsPortal.OddsPortal method*), 5
__init__() (*betbot.prediction.Predictor.Predictor method*), 8
__init__() (*betbot.prediction.SKLearnPredictor.SKLearnPredictor method*), 9

A

ApiConnection (*class in betbot.api.ApiConnection*), 3
auth_headers (*betbot.api.ApiConnection.ApiConnection property*), 3
authorized() (*betbot.api.ApiConnection.ApiConnection method*), 3

B

Bet (*class in betbot.api.Bet*), 4
betbot
    module, 10
betbot.api
    module, 4
betbot.api.ApiConnection
    module, 3
betbot.api.Bet
    module, 4
betbot.api.Match
    module, 4
betbot.data
    module, 6
betbot.data.FootballDataCoUk
    module, 4
betbot.data.OddsPortal
    module, 5
betbot.main
    module, 10
betbot.prediction
    module, 10
betbot.prediction.BettingOddsPredictor
    module, 6
betbot.prediction.DrawPredictor
    module, 6
betbot.prediction.HomeTeamPredictor
    module, 7
betbot.prediction.LeagueTablePredictor
    module, 7
betbot.prediction.NameAndOddsPredictor
    module, 7
betbot.prediction.NamePredictor
    module, 8
betbot.prediction.Predictor
    module, 8
betbot.prediction.RandomPredictor
    module, 9
betbot.prediction.SKLearnPredictor
    module, 9
betbot.test
    module, 10
BettingOddsPredictor (*class in betbot.prediction.BettingOddsPredictor*), 6

D

download_history() (*betbot.data.FootballDataCoUk.FootballDataUk method*), 5
DrawPredictor (*class in betbot.prediction.DrawPredictor*), 6

E

encode_result() (*betbot.prediction.NamePredictor.NamePredictor method*), 8
encode_result() (*betbot.prediction.SKLearnPredictor.SKLearnPredictor method*), 9
execute_api_call() (*betbot.api.ApiConnection.ApiConnection method*), 3

F

FootballDataUk (*class in betbot.data.FootballDataCoUk*), 4
from_json() (*betbot.api.Match.Match class method*), 4

G

generate_scores() (*betbot.prediction.BettingOddsPredictor.BettingOddsPredictor method*), 6
get_active_leagues() (*betbot.api.ApiConnection.ApiConnection method*), 3
get_current_matchday_matches() (*betbot.api.ApiConnection.ApiConnection method*), 3
get_history_matches() (*betbot.data.FootballDataCoUk.FootballDataUk method*), 5
get_league_table() (*betbot.api.ApiConnection.ApiConnection method*), 3
get_odds() (*betbot.data.FootballDataCoUk.FootballDataUk method*), 5
get_odds() (*betbot.data.OddsPortal.OddsPortal method*), 5

H

HomeTeamPredictor (*class in betbot.prediction.HomeTeamPredictor*), 7

I

interpret_results() (*betbot.prediction.NameAndOddsPredictor.NameAndOddsPredictor method*), 7
interpret_results() (*betbot.prediction.NamePredictor.NamePredictor method*), 8
interpret_results() (*betbot.prediction.SKLearnPredictor.SKLearnPredictor method*), 9

L

LeagueTablePredictor (*class in betbot.prediction.LeagueTablePredictor*), 7
load_history_file() (*betbot.data.FootballDataCoUk.FootballDataUk static method*), 5
load_matches() (*betbot.data.FootballDataCoUk.FootballDataUk method*), 5
login() (*betbot.api.ApiConnection.ApiConnection method*), 3
logout() (*betbot.api.ApiConnection.ApiConnection method*), 4

M

main() (*in module betbot.main*), 10
Match (*class in betbot.api.Match*), 4
module
- betbot, 10
- betbot.api, 4
- betbot.api.ApiConnection, 3
- betbot.api.Bet, 4
- betbot.api.Match, 4
- betbot.data, 6
- betbot.data.FootballDataCoUk, 4
- betbot.data.OddsPortal, 5
- betbot.main, 10
- betbot.prediction, 10
- betbot.prediction.BettingOddsPredictor, 6
- betbot.prediction.DrawPredictor, 6
- betbot.prediction.HomeTeamPredictor, 7
- betbot.prediction.LeagueTablePredictor, 7
- betbot.prediction.NameAndOddsPredictor, 7
- betbot.prediction.NamePredictor, 8
- betbot.prediction.Predictor, 8
- betbot.prediction.RandomPredictor, 9
- betbot.prediction.SKLearnPredictor, 9
- betbot.test, 10

multi_main() (*in module betbot.main*), 10

N

name() (*betbot.prediction.BettingOddsPredictor.BettingOddsPredictor class method*), 6
name() (*betbot.prediction.DrawPredictor.DrawPredictor class method*), 6
name() (*betbot.prediction.HomeTeamPredictor.HomeTeamPredictor class method*), 7
name() (*betbot.prediction.LeagueTablePredictor.LeagueTablePredictor class method*), 7
name() (*betbot.prediction.NameAndOddsPredictor.NameAndOddsPredicto class method*), 7
name() (*betbot.prediction.NamePredictor.NamePredictor class method*), 8
name() (*betbot.prediction.Predictor.Predictor class method*), 8
name() (*betbot.prediction.RandomPredictor.RandomPredictor class method*), 9
NameAndOddsPredictor (*class in betbot.prediction.NameAndOddsPredictor*), 7
NamePredictor (*class in betbot.prediction.NamePredictor*), 8

O

odds (*betbot.prediction.SKLearnPredictor.SKLearnPredictor*

property), 9
OddsPortal (*class in betbot.data.OddsPortal*), 5

P

place_bets() (*betbot.api.ApiConnection.ApiConnection method*), 4
predict() (*betbot.prediction.BettingOddsPredictor.BettingOddsPredictor method*), 6
predict() (*betbot.prediction.DrawPredictor.DrawPredictor method*), 6
predict() (*betbot.prediction.HomeTeamPredictor.HomeTeamPredictor method*), 7
predict() (*betbot.prediction.LeagueTablePredictor.LeagueTablePredictor method*), 7
predict() (*betbot.prediction.Predictor.Predictor method*), 8
predict() (*betbot.prediction.RandomPredictor.RandomPredictor method*), 9
predict() (*betbot.prediction.SKLearnPredictor.SKLearnPredictor method*), 9
predict_match() (*betbot.prediction.NamePredictor.NamePredictor method*), 8
predict_match() (*betbot.prediction.SKLearnPredictor.SKLearnPredictor method*), 9
Predictor (*class in betbot.prediction.Predictor*), 8

R

RandomPredictor (*class in betbot.prediction.RandomPredictor*), 9
regressor() (*betbot.prediction.SKLearnPredictor.SKLearnPredictor class method*), 9

S

sentry_dsn (*in module betbot*), 10
SKLearnPredictor (*class in betbot.prediction.SKLearnPredictor*), 9

T

TEAM_MAP (*betbot.data.FootballDataCoUk.FootballDataUk attribute*), 4
TEAM_MAP (*betbot.data.OddsPortal.OddsPortal attribute*), 5
to_dict() (*betbot.api.Bet.Bet method*), 4
train() (*betbot.prediction.SKLearnPredictor.SKLearnPredictor method*), 9

V

vectorize() (*betbot.prediction.NamePredictor.NamePredictor method*), 8
vectorize() (*betbot.prediction.SKLearnPredictor.SKLearnPredictor method*), 9