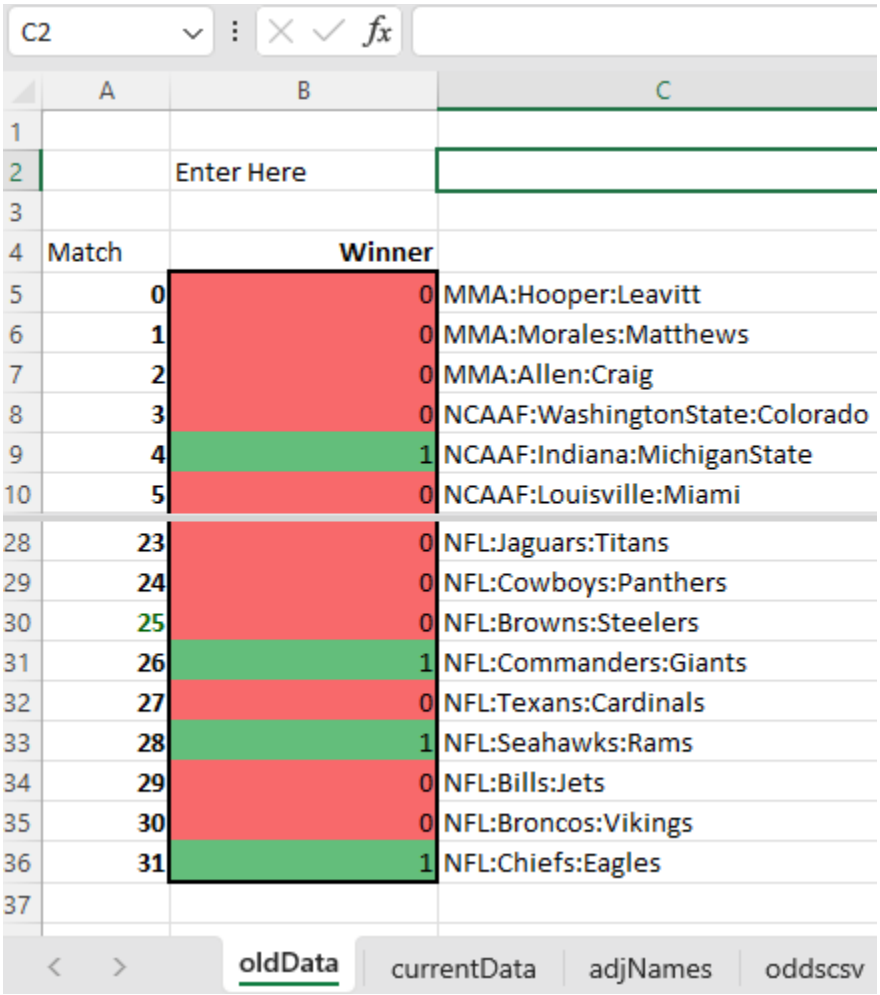
Settlement and new schedule

Three arrays:

- outcomes from prior week
- New who vs. whom
- New start times

Outcomes: zero for favorite win, one for dog win, two for tie or no contest. If events end on Sunday, one can send this on Monday, otherwise Tuesday works.

put outcome array into 'oldData' worksheet in createjsons.xlsm, B5:B36. I have it color coded here to highlight the anomalies, which is when the underdog wins. Ideally this can be completely automated, but at the least it is good to find a source to pull results into spreadsheets to generate outcomes as opposed to looking them up visually or via recollection.



C2

	A	B	C
1			
2		Enter Here	
3			
4	Match	Winner	
5	0	0	MMA:Hooper:Leavitt
6	1	0	MMA:Morales:Matthews
7	2	0	MMA:Allen:Craig
8	3	0	NCAAF:WashingtonState:Colorado
9	4	1	NCAAF:Indiana:MichiganState
10	5	0	NCAAF:Louisville:Miami
28	23	0	NFL:Jaguars:Titans
29	24	0	NFL:Cowboys:Panthers
30	25	0	NFL:Browns:Steelers
31	26	1	NFL:Commanders:Giants
32	27	0	NFL:Texans:Cardinals
33	28	1	NFL:Seahawks:Rams
34	29	0	NFL:Bills:Jets
35	30	0	NFL:Broncos:Vikings
36	31	1	NFL:Chiefs:Eagles
37			

oldData | currentData | adjNames | oddscsv

After posting the outcomes, pull data for the next weekend: who vs. whom and start times. This is where it is good to use a vendor that pulls odds, as these also give the event name and start times. There are several services that provide such data, at most of nominal cost. It will look like this:

	B	C	D	E	F
1					
2	**event_name**	**startTime**	**odd_0**	**odd_1**	**sport**
3	Bowling Green Falcons_Western Michigan Bro	2023-11-22T00:00:00.000Z	1.74	2.14	NCAAF
4	Bowling Green Falcons_Western Michigan Bro	2023-11-22T00:00:00.000Z	1.75	2.1	NCAAF
5	Bowling Green Falcons_Western Michigan Bro	2023-11-22T00:00:00.000Z	1.75	2.1	NCAAF
6	Bowling Green Falcons_Western Michigan Bro	2023-11-22T00:00:00.000Z	1.75	2.12	NCAAF
7	Bowling Green Falcons_Western Michigan Bro	2023-11-22T00:00:00.000Z	1.74	2.15	NCAAF
8	Bowling Green Falcons_Western Michigan Bro	2023-11-22T00:00:00.000Z	1.71	2.2	NCAAF
1064	Larissa Pacheco_Marina Mokhnatkina	2023-11-25T01:00:00.000Z	1.14	5.25	MMA
1065	Larissa Pacheco_Marina Mokhnatkina	2023-11-25T01:00:00.000Z	1.14	5.75	MMA
1066	Denis Goltsov_Renan Ferreira	2023-11-25T01:00:00.000Z	1.41	2.85	MMA
1067	Denis Goltsov_Renan Ferreira	2023-11-25T01:00:00.000Z	1.42	2.9	MMA
1068	Denis Goltsov_Renan Ferreira	2023-11-25T01:00:00.000Z	1.4	3	MMA
1069	Impa Kasanganay_Joshua Silveira	2023-11-25T01:00:00.000Z	1.56	2.38	MMA
1070	Impa Kasanganay_Joshua Silveira	2023-11-25T01:00:00.000Z	1.57	2.45	MMA
1071	Clay Collard_Olivier Aubin-Mercier	2023-11-25T01:30:00.000Z	3	1.38	MMA
1072	Clay Collard_Olivier Aubin-Mercier	2023-11-25T01:30:00.000Z	3.05	1.39	MMA
1073	Clay Collard_Olivier Aubin-Mercier	2023-11-25T01:30:00.000Z	3	1.36	MMA
1074	Clay Collard_Olivier Aubin-Mercier	2023-11-25T01:30:00.000Z	3.1	1.36	MMA
1075					
1076					
1077					

oldData | currentData | **adjNames** | oddscsv | settleRefreshcsv | +

The Access program uses the "_" delimiter to parse teams 0 and 1. If your data service uses a different delimiter, either replace it in the spreadsheet, or replace the specified delimiter in the Access database's Query1. In the data above, the first odds column refers to the team/fighter listed before the delimiter.

Betting add-ins usually generates the start times in Zulu time, aka GMT, as evidenced by the Z suffix. For example,

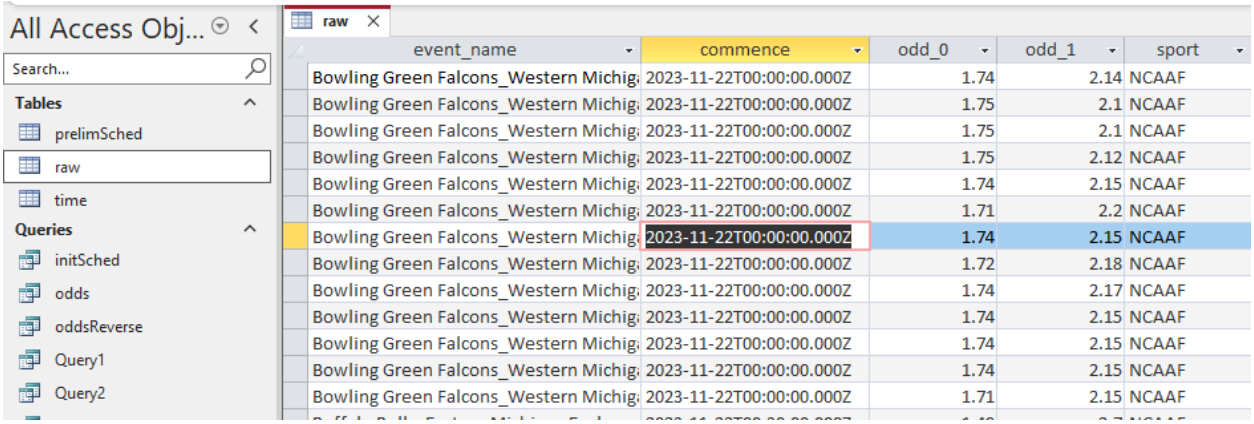
2023-11-22T00:00:00.000Z

is GMT. Access extracts the date and time, and then translates it into UTC

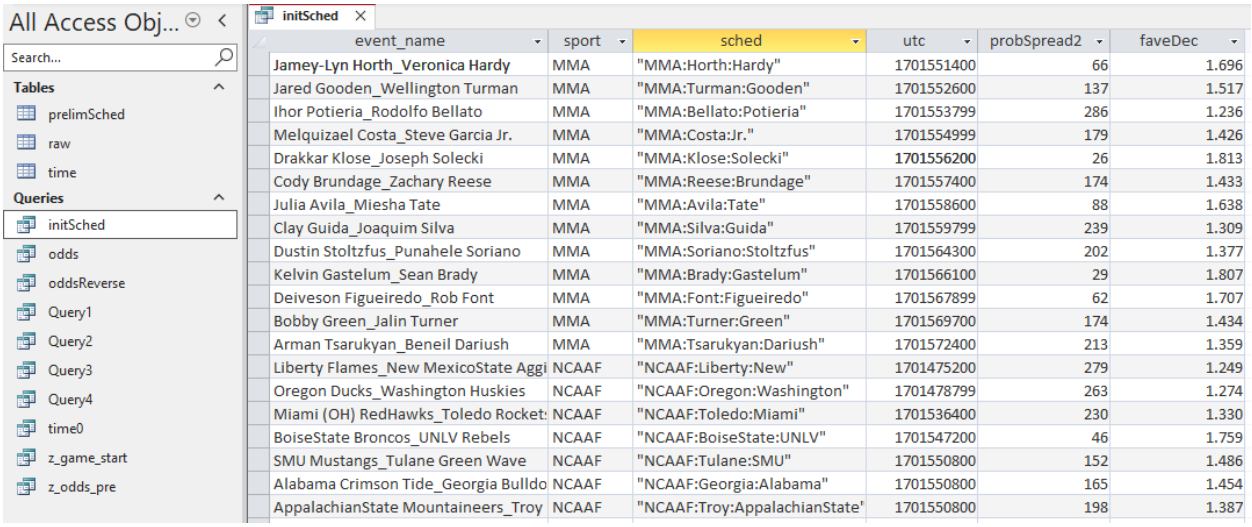
How to send Oracle data

The final column is the sport, which must be added manually. It is good to be consistent, and initially this used the notation MMA, NFL, and NCAAF for mixed martial arts, pro US football, and college US football. Boxing is an obvious addition that work.

Remove all spaces from team/player names. For example, West Virginia should be WestVirginia or WVirginia, not West or Viginia, which is what Access will generate if you do not remove the space between West and Virginia. Once done, take the 1000ish × 5 array of raw data and paste it into MSAccess database's **raw** table (delete the old raw table first). The queries will automatically exclude events commencing prior to the next Friday at 9:00 PM GMT or after next Tuesday at 9 PM GMT, so you do not have to filter these out manually.



Execute the query **initSched** and pull the table into a spreadsheet



The only thing needed at this point are the **sched** and **utc** columns. The other columns are there primarily as a sanity check. The sched field is in the form:

- "sport:favorite:underdog"

How to send Oracle data

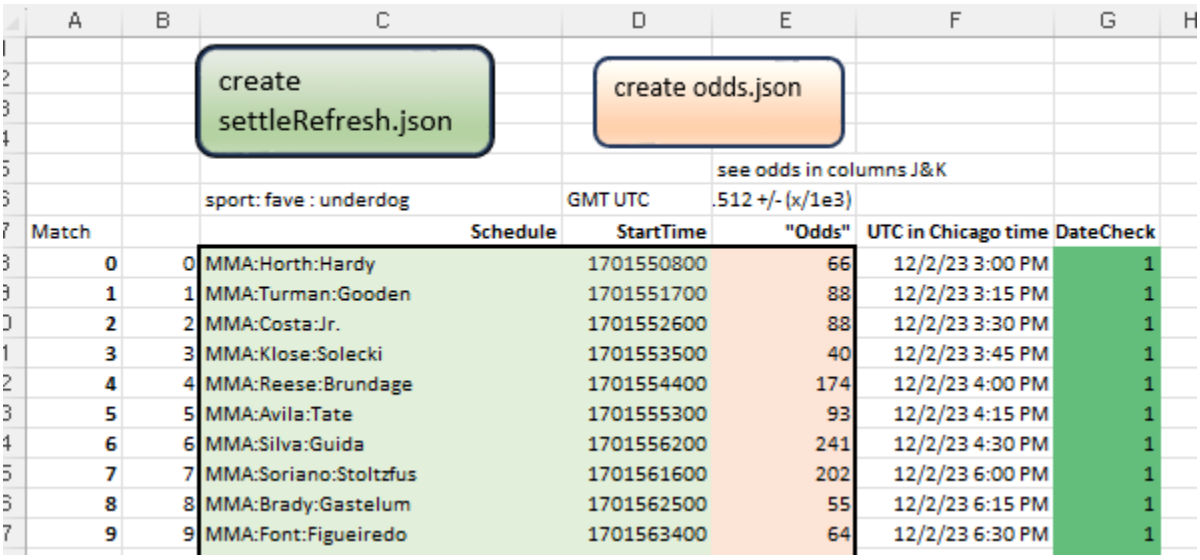
- eg, “NFL:Chiefs:Raiders”

The quotes are useful hear because pasting into excel without quotes often puts data separated by colons into separate columns, and the quotes tells excel to keep it all in the same column. However, different users will have different Excel defaults.

The start times for NFL and NCAA football are usually correct, but not always. One must check times against an independent source (pulling diverse data from one add-in is one source). MMA start time data is more frequently wrong. This problem is especially bad when fights are in Abu Dhabi, and fights start around 1 pm USA time.

The odds data are not important, though Access excludes contests where the favorite has decimal odds lower than 1.17 or greater than 5.42. If the contest is in the schedule for next week, one cannot delete the contest, or replace it with a new one. An event with new odds outside of what the contract accepts will have to take stale odds of 335, which corresponds to a 1.17-5.42 event. If these odds are more than 2.5% off the current odds, the oracles should stop betting on the contract immediately after the odds are posted, and the result should be coded as a 2 for ‘no contest’ regardless of outcome. This effectively cancels the event. It is probably best to exclude contests on the probability edge to avoid this problem

Once you have your list, paste it into the currentdata worksheet.



Match		Schedule	StartTime	"Odds"	UTC in Chicago time	DateCheck
0	0	MMA:Horth:Hardy	1701550800	66	12/2/23 3:00 PM	1
1	1	MMA:Turman:Gooden	1701551700	88	12/2/23 3:15 PM	1
2	2	MMA:Costa:Jr.	1701552600	88	12/2/23 3:30 PM	1
3	3	MMA:Klose:Solecki	1701553500	40	12/2/23 3:45 PM	1
4	4	MMA:Reese:Brundage	1701554400	174	12/2/23 4:00 PM	1
5	5	MMA:Avila:Tate	1701555300	93	12/2/23 4:15 PM	1
6	6	MMA:Silva:Guida	1701556200	241	12/2/23 4:30 PM	1
7	7	MMA:Soriano:Stoltzfus	1701561600	202	12/2/23 6:00 PM	1
8	8	MMA:Brady:Gastelum	1701562500	55	12/2/23 6:15 PM	1
9	9	MMA:Font:Figueiredo	1701563400	64	12/2/23 6:30 PM	1

The contract can handle Against the Spread (ATS) bets by conspicuously putting the point spread by the favorite in the ‘schedule’ text field. This is often useful in games where there is a large spread and extreme odds, as otherwise the game cannot fit the contract. A good ATS spread implies a probSpread/2x1000 number of zero, in that each team has the same chance of beating the spread. For example, this weekend Michigan is a big favorite against Iowa, with 1.05 decimal odds. This translates to a 95% win probability. It also has a 21.5 point spread.

How to send Oracle data

14	NCAAF:Tulane:SMU	1701550800	164
15	NCAAF:Georgia:Alabama	1701550800	153
16	NCAAF:Michigan-20.5:Iowa+20.5	1701565200	10

One can enter the data as above for the Michigan-Iowa game. In the above ATS case, if Michigan wins by 21 or more, team 0 wins match 16, else team 1 wins. I lowered the spread to 20.5 from 21.5 market odds. This moves the odds from even, which corresponds to a prSpread/2x1000 = 0, to a number like 10, which implies 1% higher-than-even odds of winning. This way, if the spread moves down to 20.5, I can capture the contest with an input of zero; if the spread stays the same, I can use the odds number 10 or 20, and if the odds number moves up, and can use the prSpread/2 number of 30 or 40.

Historically, the relation between the probabilitySpread/2, spread, and odds is as follows (NFL games from 2014-2022)

	Prob(fave=win)		
Spread	**historical**	**odds implied**	**prSpread/2x10000**
-12	92%	85%	342
-11	82%	84%	326
-10	82%	82%	305
-9	81%	80%	287
-8	76%	78%	266
-7	76%	75%	235
-6	69%	71%	198
-5	62%	68%	172
-4	60%	65%	141
-3	57%	60%	88
-2	53%	55%	40
-1	51%	53%	20

Note the probability of a favorite winning moves by about 5% for each point when between 2 and 7. 5% corresponds to a 50 in the probSpread/2 number (50/1000). As bettors need a 2.5% edge in predicting winners, a 5% error creates a problem for the house, in that bettors would then have the edge over the house. For this reason, and because contests with 3-point spreads are almost even odds, it is best to use ATS on games where the spreads are higher, say above 7. However, for special events, like Superbowl weekend, it would be useful to offer an ATS *and* straight up bets, as there will not be many alternatives and lots of demand.

If there are not 32 events, the contract still needs non-null data for all 32 elements of the array. Make the ‘sport’ as ‘AAA’, and the front end will not display these events (eg, AAA:bla:foo). If someone mistakenly bets on these games, make sure to record an outcome of 2, so that anyone will get their money back.

Click the ‘create settleRefresh.json’ button in the worksheet currentData, and it will throw the settleRefresh.json file into the python directory.

How to send Oracle data

An oracle account can send the data to the oracle contract between midnight and 3 AM GMT. One can do this using the python program settleRefresh.py. Once a submission is received, the oracle contract's 'subnumber' will change from 0 to 1. After a submission, only the same oracle account can send replacement data, and only until 3 AM GMT. After 3 AM GMT the other oracle accounts can vote. The python programs voteNo.py and voteYes.py can automate this. After 3 PM GMT that same day, anyone can execute the processVote() function, and oracle accounts can automate executing the processVote.py program. If the vote is majority yes, it is then sent to the betting contract, settling the prior week's bets, and setting up the contract for the next weekend.

Sending Odds

one array:

- probSpread/2 x 1000

Once the settleRefresh() function has been processed, the next step is applying odds to the events in the contract. The first step is to grab those events from the oracle contract, as the precise order is essential. This can be done via getOracleData.py. Paste the data into the MSAccess table relimSched. Make sure the first column is numbered from 0 to 31 (autonumbering created problems). This is the target for getting the odds.

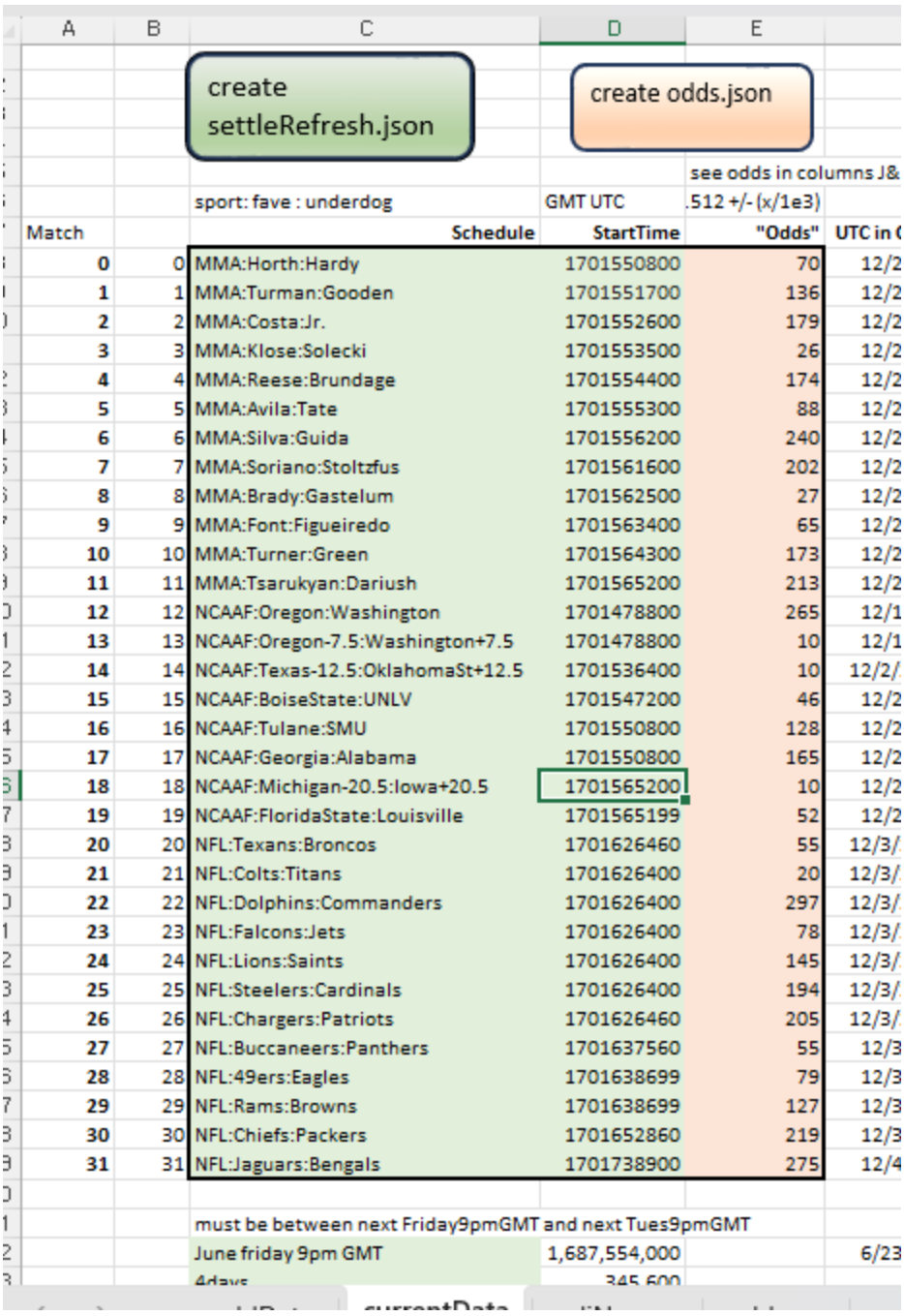
The next step is to pull the raw data as before, pasting into a cleaned 'raw' table in the MSAccess database. Execute the odds query, and the events and odds will be listed. If there are blanks, this can be due to a few reasons. First, the team may be typed differently. For example, One may have "OleMiss" instead of "OleMississippi", or "Texas" instead of "TexasA&M" (spaces!).

Just adjust the data in the 'raw' table to match what is in the contract. Another possibility is that the favorite changed, so that instead of

"MMA:Smith:Jones" it is now "MMA:Jones:Smith". This often happens when the match is nearly even, as odds changes by about 1.5% each week (in terms of a probability of a win). The query 'oddsReverse' captures these cases. If the new odds are close to even, one can still put the bet in there, with a SpreadProb/2 of zero, but if the new favorite is a large favorite, one might halt betting on that event right after posting the data to the contract.

Lastly, ATS bets described above will not be in the raw data. Yet, in that case the odds need a manual touch. If the spread in the text field equals the current spread one can input zero as the spreadProb/2. If the live spread is 1.0 greater than what is in the contract's text field, one might apply a number like 20 or 30. If the live spread is 1.0 lower, or generally leaves the contract with a non-equilibrium spread, one might use a zero and then be sure to instantly halt the contest before anyone bets on it. Note

Once these data are finalized, put them into the worksheet currentData, cells E8:E39, and click create odds button. This sends the odds.json file to the python directory, where the program oddsPost.py can grab it and send it to the Oracle contract.



Match		Schedule	StartTime	"Odds"	UTC in (
		sport: fave : underdog	GMT UTC	see odds in columns J& .512 +/- (x/1e3)	
0	0	MMA:Horth:Hardy	1701550800	70	12/2
1	1	MMA:Turman:Gooden	1701551700	136	12/2
2	2	MMA:Costa:Jr.	1701552600	179	12/2
3	3	MMA:Klose:Solecki	1701553500	26	12/2
4	4	MMA:Reese:Brundage	1701554400	174	12/2
5	5	MMA:Avila:Tate	1701555300	88	12/2
6	6	MMA:Silva:Guida	1701556200	240	12/2
7	7	MMA:Soriano:Stoltzfus	1701561600	202	12/2
8	8	MMA:Brady:Gastelum	1701562500	27	12/2
9	9	MMA:Font:Figueiredo	1701563400	65	12/2
10	10	MMA:Turner:Green	1701564300	173	12/2
11	11	MMA:Tsarukyan:Dariush	1701565200	213	12/2
12	12	NCAAF:Oregon:Washington	1701478800	265	12/1
13	13	NCAAF:Oregon-7.5:Washington+7.5	1701478800	10	12/1
14	14	NCAAF:Texas-12.5:OklahomaSt+12.5	1701536400	10	12/2/
15	15	NCAAF:BoiseState:UNLV	1701547200	46	12/2
16	16	NCAAF:Tulane:SMU	1701550800	128	12/2
17	17	NCAAF:Georgia:Alabama	1701550800	165	12/2
18	18	NCAAF:Michigan-20.5:Iowa+20.5	1701565200	10	12/2
19	19	NCAAF:FloridaState:Louisville	1701565199	52	12/2
20	20	NFL:Texans:Broncos	1701626460	55	12/3/
21	21	NFL:Colts:Titans	1701626400	20	12/3/
22	22	NFL:Dolphins:Commanders	1701626400	297	12/3/
23	23	NFL:Falcons:Jets	1701626400	78	12/3/
24	24	NFL:Lions:Saints	1701626400	145	12/3/
25	25	NFL:Steelers:Cardinals	1701626400	194	12/3/
26	26	NFL:Chargers:Patriots	1701626460	205	12/3/
27	27	NFL:Buccaneers:Panthers	1701637560	55	12/3
28	28	NFL:49ers:Eagles	1701638699	79	12/3
29	29	NFL:Rams:Browns	1701638699	127	12/3
30	30	NFL:Chiefs:Packers	1701652860	219	12/3
31	31	NFL:Jaguars:Bengals	1701738900	275	12/4
		must be between next Friday9pmGMT and next Tues9pmGMT			
		June friday 9pm GMT	1,687,554,000		6/23
		4days	345,600		

After this, just as in the settlement function, the other oracles should all evaluate the data, and then vote via voteYes.py or voteNo.py, and finally, processVote.py.

I walk through the processes in two instructional YouTube videos:

https://www.youtube.com/watch?v=zBaF7AV5TQ8

https://www.youtube.com/watch?v=s4h1DBIEgv0

How to send Oracle data