Баскетбол бывает по 5, 10 12 и 20. Прогноз общего тотала(суммы очков) ОДНОЙ КОМАНДЫ!! за четверть игры:

Очки команда = ОК, Б, ТЧ – тотал(сумма очков) четверти,

СВЧ–сыгранное время четверти, ОВЧ–оставшееся время четверти, ВЧ–время четверти,

Прогноза очков относительно их количества за время четверти:

Средний максимум очков команды за время четверти:

Время: [300, 600, 720, 1200 ]

Множитель: [0.06, 0.055, 0.0625, 0.045 ]

Очки: [18, 33, 45, 72 ]

Средне-максимальный прогноз очков, относительно множителя и времени:

за всю четверть:

на текущее время:

возможное количество за оставшееся время:

Модифицированный прогноз очков (одной команда):

Если ВЧ > 0, но ОК = 0 то ТЧпоОК = ;

Динамический знаменатель относительно времени четверти:

ДЗ = 2 + Хвостик; Хвостик = 0,6 при СВЧ = 0;

изменения ДЗ относительно СВЧ = ДЗ = ДЗ – ;

МПО одной команда =

Общий тотал очков ИГРЫ!!

ТЧпоОК общее = ТЧпоОК1 + ТЧпоОК2;

МПО общее = МПО1 + МПО2;

Находим разницу в процентах и уровень (что <, >?) между ТЧпоОК общее и МПО общее

function suppressionJumpsDifferences(percentDifferenceAndLevels, forecast, percentLastTime){

let suppressionForecast = forecast; let transitionalPercentDifference = 11.5;

let dwTransitionalPercentDifference = 2.2; let dynamicDenominatorForLevelTwo = 2.7;

let limitDynamicPercentLastTime = 60;

if (percentLastTime <= limitDynamicPercentLastTime){

dynamicDenominatorForLevelTwo =

rationalDependenceNumberBonA(percentLastTime, limitDynamicPercentLastTime, dynamicDenominatorForLevelTwo, 1.95);

};

switch (true) {

case ((percentDifferenceAndLevels[1]== 2)&&(percentDifferenceAndLevels[0] > transitionalPercentDifference)):

suppressionForecast =

forecast - ( ( forecast / 100 ) \* (percentDifferenceAndLevels[0] / dynamicDenominatorForLevelTwo ) );

break;

case ((percentDifferenceAndLevels[1]== 2)&&(percentDifferenceAndLevels[0] < transitionalPercentDifference)):

suppressionForecast = forecast - ( ( forecast / 100 ) \* (percentDifferenceAndLevels[0]) );

break;

case ((percentDifferenceAndLevels[1]== 0) &&

(percentDifferenceAndLevels[0] < (transitionalPercentDifference - dwTransitionalPercentDifference) ) ):

suppressionForecast = forecast + ( ( forecast / 100 ) \* (percentDifferenceAndLevels[0] ) );

break;

};

return suppressionForecast;

}

function rationalDependenceNumberBonA(numberA, MaxNumberA, MinNumberB, MaxNumberB){

let numberBonA =

((MaxNumberB - MinNumberB)/100) \* ((numberA \* 100) / MaxNumberA) + (MinNumberB - MaxNumberB) + MaxNumberB;

return numberBonA;

}