Aprendizado profundo para analisar e predizer corridas de cavalo

Eduardo Augusto Saito - 140524  
Instituto de Ciência e TecnologiaUniversidade Federal de São PauloSão José dos Campos, Brasil  
easaito@unifesp.br

Wallace dos Santos Costa - 142548  
Instituto de Ciência e TecnologiaUniversidade Federal de São PauloSão José dos Campos, Brasil  
wcosta@unifesp.br

# Introdução

Quando se pensa em apostas esportivas, comumente vem à cabeça as modalidades: futebol, basquete, E-sports, vôlei entre outras de maior contato, com pessoas atuando no esporte, e não através do uso de animais. Isso ocorre pois no Brasil temos essas modalidades como as principais em nossa cultura, no entanto uma modalidade muito popular nos Estados Unidos da América, Reino Unido, entre outros países, nos chamou a atenção, as corridas de cavalos; Tendo um número mínimo de 2 cavalos disputando entre si, podendo chegar a 40 cavalos (Prêmio Grand National – Reino Unido), variando de acordo com as regras de cada pista para que ocorra a alteração na quantidade de cavalos, e também no estilo que será corrida onde estão inscritos, com obstáculos, em terra batida ou gramado. Esse esporte movimenta milhões todos os dias, como a betfair que só em 2014 no reino unido teve uma movimentação de 2.1 bilhões de libras[1], sendo um esporte com adeptos no Brasil também, mas sem nenhum algoritmo, mesmo que experimental para que possa ocorrer uma predição de quais cavalos podem vir a ganhar ou perder, tornando a experiência do usuário que acompanha o esporte e aposta, única e exclusivamente baseada em análises humanas, sendo essa passível de falha, já que pode passar despercebido para os telespectadores, podendo ocasionar em perdas de valores apostados. Desta forma achamos interessante trazer a proposta de através da inteligência artificial, realizar uma predição simples com a finalidade de identificar qual a porcentagem de chance de determinado cavalo ganhar uma corrida, ou qual não ganharia de forma alguma, com aplicações em aprendizagem profunda, método esse que vai auxiliar e possivelmente identificar quais serão os cavalos da corrida com chances de vencer.

# Trabalhos relacionados

Para que possamos identificar as classes que iremos trabalhar, pessoas e cavalos, utilizaremos um modelo pré-treinado que disponibiliza, as bounding boxes das classes, com suas respectivas coordenadas x e y, tanto para imagens quanto para vídeos. Esse modelo foi disponibilizado[2] em um vídeo no Youtube, e usaremos inicialmente como suporte para iniciar o projeto tendo alguma referência. Também temos como referência breve, o estudo complementar[3], em que foi feito a pesquisa dos eixos de quando um cavalo está em uma curva, o qual usamos no início dos estudos para fins de possíveis definições de qual modelo usaríamos, esse projeto não terá aplicabilidade no nosso estudo, no entanto achamos interessante citá-lo, pois serviu de pontapé inicial para que buscássemos outros que já realizaram algo similar. Também foram feitas pesquisas com outros graduandos, em agradecimento especial ao Matheus, que nos disponibilizou seu modelo de YOLO para estacionamentos, projeto realizado em 2023[4], que mostra que é possível utilizar imagens em tempo real, ou vídeos para fazer predições, o que nos motivou a continuar com o estudo e criação de um algoritmo atender nosso problema.

# Conceitos fundamentais

Para que possamos aplicar os conceitos e torná-los de simples compreensão para quem realizar a leitura deste artigo, explicaremos alguns conceitos que já estamos utilizando em nosso projeto, e conceitos que serão futuramente implementados

* Para as predições e aplicação dos resultados, utilizaremos o site betfair[5], site em que é possível realizar uma “aposta contra”(lay) um cavalo, para casos de cavalos que provavelmente não ganharão a corrida.
* Lay, como brevemente mencionado no ponto anterior, se refere a inserir um valor em um cavalo, supondo que este não ganhará a corrida, portanto, todos os demais cavalos podem ganhar, exceto esse cavalo, o Lay serve para aplicar, a execução final do projeto, caso o objetivo de nossa predição apenas diga quais são os cavalos que não ganharão de forma alguma, mas esteja funcionando para apostas.
* ODDs, nome dado aos valores pré-fixados pelo site da betfair, as ODDs podem ter variação e poderão de forma muito expressiva, auxiliar na identificação de possíveis ganhadores/perdedores, uma vez que sua variação ocorre de forma ao vivo, e na medida que a corrida se aproxima do fim, cavalos com maiores chances de perder tem ODDs mais altas ao mesmo tempo que melhores cavalos tem ODDs mais baixas, e isso poderá ser cruzado com nossa análise do próprio vídeo do cavalo, dando mais notoriedade ao que estamos afirmando na predição
* A IDE que será utilizada, será Python, por Anaconda, já que existe diversas funcionalidades que substituem o Google Colab, sem a perda da qualidade do programa. Inicialmente pensamos em utilizar o Google Colab, no entanto após determinadas execuções, ele retorna uma limitação de uso, e o Anaconda até o presente momento se mostrou bastante promissor para execução dos algoritmos que usaremos.
* O modelo pré-treinado que será utilizado, será o modelo YOLO Ultralytics[7], neste já existem modelos de itens, itens esse que identificam: pessoas, cavalos, carros, placas entre outros tipos de itens, é um modelo muito bom, no entanto com alguns desafios.
* Obteremos as bounding boxes com os eixos (x,y) também através do YOLO, identificando os Jockeys que estão pegando impulso e não estão avançando para próximo dos primeiros, desta forma podemos prever que o Jockey está se movimentando muito, ou pouco, e futuramente nos estudos, isso pode servir de confirmação que a chance do cavalo não ganhar coincidiu com o resultado final da corrida.

# Objetivos

Nossos objetivos são identificar uma corrida que esteja acontecendo de forma ao vivo na Betfair, e durante o percurso da corrida o algoritmo verificar padrões que demonstre qual cavalo está perdendo impulso, ou qual jockey está se mexendo muito devido ao cansaço do animal. Pensamos que essa análise é muito viável, pois com o algoritmo bem calibrado não seria mais necessária análise humana, até certo ponto. Também é esperado que quando um cavalo ficar atrás de outro, sumir da câmera ou tenha seu ID resetado, possamos tratar para que eles continuem nos cálculos e nas predições. Desta forma não teríamos apenas cavalos que estejam em evidência, já que estes podem ser apenas os favoritos, o que tornaria nosso projeto muito mais difícil de passar alguma predição. Este problema foi um dos primeiros que identificamos no processo de elaboração do projeto, e para isso já estamos pensando que assim com que um cavalo que suma do ângulo da câmera por algum motivo, ainda esteja nas predições, e dessa forma porcentagens de cavalos sejam reportadas ao usuário do algoritmo.

# Metodologia experimental

No início deste projeto e optando por manter parte do primeiro material escrito, temos antes da execução, os passos inciais da nossa metodologia experimental, sendo em reuniões com o professor, identificamos que nossos maiores problemas na elaboração do projeto, serão com cavalos que não aparecerão constantemente na câmera, problema esse que envolvem diversos estudos em que um objeto de interesse é retirado por outro da imagem, ficando sobreposto na imagem.

Também teremos como entraves como identificar qual padrão corresponderá a um determinado Jockey, visto que podem haver semelhanças de uniformes ou cores de cavalos. Para isso precisaremos tratar como grafos, para que o algoritmo os armazene de acordo com probabilidades de distancia de cavalo X, para Y, visto que o cavalo probabilisticamente não terá mudado de posição, a menos que avance ou recue, sendo estático na imagem analisada pelo algoritmo.

Visando uma predição mais simples a priori faremos análises de vídeos já gravados, não sendo ao vivo, de corridas, de preferência com poucos Jockeys, uma vez que fica mais fácil para aprendizagem do algoritmo e calibragem, para posterior expansão em corridas mais complexas com mais corredores.

Após essa primeira parte concluída, nosso objetivo vai ser identificar quais são os eixos Y que mais sofrem alteração, dessa forma sabendo que o Jockey está pegando impulso, ou perdendo-o, denotando um jockey possivelmente não vencedor, essa análise feita com base no dataset citado em nosso objetivo.

Em seguida com o dataset dos dados gerados, uma função que deverá armazenar o progresso de todos os cavalos presentes, e ir passando probabilidades, feito essa devido a variação de Y, provavelmente em forma de porcentagem, do cavalo que está com melhores chances de vencer, e o cavalo que não está apresentando melhorias.

Após a identificação correta dos cavalos em vídeos já gravados, iremos então tentar aplicar de forma ao vivo para o próprio site Betfair, onde faremos testes e adaptação do algoritmo para que a análise ocorra em tempo real, fazendo então com que o algoritmo tenha aplicabilidade no mundo real, por apostadores.

Como ocorrerá variação da nossa base de testes para o mundo real, acreditamos que seja o ponto mais desafiador do projeto, visto que teremos que adaptar o algoritmo para a tela com URL aleatória para cada corrida que será gerada, sendo esse o penúltimo passo e também crucial para um algoritmo que funcione em programação ao vivo.

Nos Lays, como último passo, utilizaremos a variação das ODDs para em conjunto com nossa análise da posição do cavalo e variação do eixo (X,Y), dar um resultado mais assertivo.

Após esse último passo, faremos então um algoritmo que emita a predição, dando sugestões ao utilizador do melhor cavalo para se realizar uma bet.

Após esses pontos definidos e previstos, daremos então início ao projeto em sua parte final, para que tenhamos conhecimento e controle do que foi dito e para Metodologia efetivamente apresentada.

Para o código, optamos inicialmente por gerar e preparar nosso próprio dataset do tipo CSV, com valores contendo: ID do vídeo, ID da cena, lista de coordenadas X e Y do centroide da Bounding Box do montador do cavalo (jockey), além do valor 1 para ganhador e 0 para não ganhador, utilizando o modelo YOLO8 do Ultralytics e o CV2 para que consigamos percorrer vídeos de corrida detectando e rastreando os cavalos a cada frame, pois com a característica da base contendo a variação do eixo Y. Teremos assim uma possível previsão do ganhador conforme a corrida vai avançando, pois o Jockey tende a se movimentar mais em cima do cavalo, devido ao cansaço do animal e também para tentar acelerar o animal ao fim da corrida.

Iniciando o pré-processamento tivemos que encontrar diferentes vídeos com o mínimo de variação de câmera possível, pois isso poderia interferir nos nossos resultados, feito isso pegamos os arquivos gerados destes nossos vídeos, pois não existia uma base pronta para treinamento.

Nos encontros realizados, fomos orientados pelos professor e tivemos como primeiro objetivo selecionar por volta de noventa vídeos e assim extrair os dados dos vídeos como posição do Jockey, movimentações, direções, velocidade e afins. Entretanto, a quantidade de vídeos proposta não foi possível, pois neste ponto, já identificamos um problema de processamento por parte dos computadores que tínhamos a disposição, já que para executar o processamento requerido, era necessário um grande poder computacional, além da ferramenta utilizada Google Collab perder seu tempo gratuito de uso e nos induzir a adquirir a versão paga, algo que não pudemos por conta de limitação financeira. Desta forma, selecionamos apenas sete vídeos localizados com as melhores características para o projeto e que foram possíveis de extrair, tratar e gerar os dados que almejamos. A partir deste ponto, com o Python, linguagem que escolhemos para criar nosso modelo preditivo, e desta forma além da criação, geração e tratamento dos dados de cada vídeo, também passou a ser possível baixar e utilizar ferramentas de Stream para visualizar os vídeos localmente e realizar as extrações de seus dados.

Como abordado anteriormente, conseguimos criar o nosso dataset utilizando o modelo YOLO8, modelo mais recente de detecção de objetos do Ultralytics. Com este modelo foi possível que pudéssemos selecionar as diferentes classes que nos interessava rastrear no frame ou imagem gerada, que no nosso caso, seriam especificamente as classes: Cavalo e Pessoa, de forma que os dois estivessem juntos na imagem, já que algumas pessoas podem aparecer na imagem, como torcedores, ou desenhos de cavalos, dependendo do vídeo exportado. O modelo YOLO após inicialização e especificação das classes de interesse, nos retorna um objeto com diferentes informações, essas que podemos acessar frame por frame do vídeo ou então um processamento do vídeo por completo. Para ambos os casos, o objeto que é retornado pelo modelo, nos fornece a geometria da imagem e outros dados muito relevantes, como o ID do Objeto detectado, nome, sua bounding box e outras. Após termos partido desta etapa inicial, e também com ajustes e adaptações que fomos realizando para que os dados fossem os mais precisos e úteis para nosso interesse, pudemos identificar já então alguns possíveis problemas que surgiriam durante o desenvolvimento sendo eles os seguintes: reset de ID dos cavalos e dos Jockeys; mudança das câmeras (aproximações, afastamentos, ângulos inesperados, entre outros).

Devido a isto, durante algumas reuniões entre os membros do projeto, e também com o monitor Christian que muito nos ajudou pensamos e buscamos diferentes alternativas para o cálculo, assertividade e correção destes pontos que eram importantes tratar. Assim, optamos por utilizar o cálculo de intersecção das bounding box através do Índice de Jaccard (também conhecido como Intersection-over-Union), este que retorna um valor de confiança de que uma bounding box está sobrepondo outra. A partir daí, definimos uma margem de 0,4 (40%) de score para que pudéssemos obter corretamente os Jockeys, ou seja, a intersecção da bbox do montador com o cavalo. Ao adotar esta técnica, evitamos poluir o nosso dataset com funcionários e pessoas que não estavam efetivamente em cima dos cavalos, como a plateia para eventos grandes, ou membros e familiares que estavam próximos antes ou durante as corridas, visto que o modelo YOLO8 rastreia estas pessoas por sua base ter essas classes previamente definidas, mas não ser importante para o nosso projeto. Além disso, como existem ângulos e corridas em que alguns Jockeys estão próximos a outros em determinados momentos, mantemos durante as iterações o maior valor do índice de Jaccard (o melhor cavalo para o Jockey), tentando garantir que até um certo ponto o Jockey que está sendo rastreado está montado no cavalo logo abaixo.

Para tentar solucionar os problemas de câmeras e suas movimentações suaves ou bruscas que é o principal causador da perda e reset do ID dos corredores identificados, optamos por fazer algumas verificações entre o frame anterior e o frame atual (também conversado durante as reuniões sobre essa possibilidade) e desta forma realizar um cruzamento e identificar a similaridade de ambos. Buscando na literaturas e outros conteúdos, encontramos e aplicamos através da biblioteca do CV2, ao pegar dois frames durante as iterações (o atual e seu anterior) e os estilizamos para que ambos ficassem em tons de preto-e-branco, eliminando desta forma qualquer informação desnecessária. Após a utilização deste método, utilizamos a técnica de Similaridade Estrutural (Structural Similarity Index Measure ou SSIM) para compará-los através de uma pontuação; Ao testar esta funcionalidade, notamos que utilizar um limite de muito alto como 0,8 (80%), estava ocasionando problemas especialmente dependendo da velocidade da corrida. O índice acabava por identificar pequenas alterações como frames completamente diferentes (por exemplo num trecho de um dos vídeos em que havia um um grande público de pessoas ao redor e acabava contabilizando como um frame totalmente diferente). Sendo assim, optamos por deixar em 0,5 (50%) e este afrouxamento dos parâmetros melhorou o nosso resultado de forma satisfatória. A ideia de utilizar esta técnica era para que pudéssemos criar um novo dicionário, uma cena, com novos IDs, ou seja, um novo rastreio, visto que se mantivéssemos os IDs anteriores, o valor de X e Y das bounding boxes dos objetos seriam inconsistentes, tornando-os muito distintos entre si podendo gerar um ruído que afetasse os resultados esperados mesmo que na etapa de treinamento do modelo preditivo.

Por fim, uma das etapa finais para a conclusão da metodologia final aplicada e criação de um dataset que pudéssemos utilizar nos treinamentos era definir quem é o vencedor nos vídeos que utilizamos como base para treino e teste. Haviam duas opções, assistir o vídeo e fazer isto manualmente gerando esse ruído com aprendizagem supervisionada ou tentar encontrar alguma outra forma de calcular o vencedor com os dados que já havíamos extraído.

Por conseguinte, acabamos por na mesma ideia de comparar um frame com o seguinte, fizemos isso para cada final de cena, comparando os último cinco centróides do Jockey e estimando a direção em que as bounding boxes estavam caminhando durante o vídeo, através do vetor de movimento. Descobrindo assim o ângulo em que o vetor está se direcionando, e assim conseguimos então estimar a direção em que a bounding box está indo e gerar um valor de ‘esquerda’, ‘direita’, ‘esquerda-baixo’, etc. para o objeto. Como supostamente todos os objetos se direcionam para a mesma direção, fizemos um cálculo simples de distância até a margem, estabelecendo a pessoa mais próxima da margem, nestes cálculos, como a vencedora de uma determinada cena.

Passando então ao Reconhecimento de Padrões temos as tarefas que serão realizadas nessa próxima etapa, as de extração de mais corridas, além de aplicação de gráficos e modelos de classificação, rotulando um possível padrão de cavalos ganhadores, essa parte da tarefa tende a ser mais desafiadora pois conviveremos com a sobreposição de cavalos e reset do ID na medida que a câmera altera de posição.

O algoritmo que utilizamos é o Random Forest por ter boa precisão como visto em algumas literaturas, porém SVM pode vir a ser utilizado caso os dados precisem de um modelo mais geral e que funciona em primeiros testes.

O teste de significância que esperamos diz respeito ao teste trazer um padrão de ganhador, após X período de tempo, logo no início da corrida, com uma predição que possa ser aplicado ao mundo real, com base que nunca vimos.

Com o dado da coordenada do jockey no eixo y, criamos novas colunas através do cálculo de média, desvio padrão, valor mínimo e valor máximo (mean, std, min e max, respectivamente). Extraindo assim dados extras baseado nos valores de y de cada jockey.

E por fim no pós-processamento e este é o mais importante, após preparar os dados de treino e criar um dataset com diversas coordenadas referentes a posição do jockey no vídeo, conseguimos enfim treinar o modelo utilizando algoritmo Random Forest. Através deste, separamos os dados de treino e teste e selecionamos apenas as colunas que nos eram interessantes, até então, ‘rider\_id’, ‘y\_values’ e as outras colunas que previamente criamos para analisar possíveis padrões, que se referem ao eixo y do jockey no frame.

Para este algorimo, após o treino, geramos dois valores diferentes de acurácia, a acurácia normal e a AUC (Area Under Curve). Para a acurácia normal, obtivemos um resultado de 0.8 (80%) e para a AUC um valor de 0.85 (85%). Além disso, para que pudéssemos analisar realmente as diferentes configurações de parametrização no algoritmo de Random Forest, optamos por utilizar também o Refinamento por Hiperparâmetros com Busca Aleatória (Hiperparameter Tuning with Random Search). Desta forma, ao executar este modelo, obtemos como retorno um grupo de melhores parâmetros para testar nosso modelo. Então, com estes parâmetros, realizamos novamente o cálculo e obtivemos o valor de 0.78 (78%) de acurácia.

# Resultados finais esperados

Esperamos que ao final desse projeto, consigamos executar o algoritmo na própria betfair, ao vivo, e trazer em tempo real a variação e chance de cada cavalo, fazendo assim com que nosso algoritmo consiga predizer de forma assertiva quais cavalos ganharão ou não baseado nas metodologias aplicadas, tornando assim possível para que o usuário da aplicação possa ganhar dinheiro utilizando o algoritmo, já que os cavalos potencialmente perdedores ou ganhadores podem passar despercebidos aos olhos, mas com uma boa conclusão desse projeto, esse entrave possa ser superado, visto que as apostas em corridas de cavalos podem gerar uma renda extra a quem vier atuar neste mercado tão competitivo e rentável.

Ainda segue sendo nossa ambição, que ocorra uma predição tornando possível retornos financeiros acerca do nosso projeto, no entanto este modelo ficará para estudos além da disciplina, ou reaproveitamento por outros membros acadêmicos, pois o resultado se limitou a parte dos vídeos gravados, mas que existe um potencial enorme de estudo, aprimoramento e aperfeiçoamento do projeto que idealizamos e iniciamos.

# Referências

1. https://www.flutter.com/media/ih5pnrww/annual-report-2014.pdf, pg 4.
2. https://github.com/itberrios/CV\_tracking/blob/main/setup\_tutorials/YOLOv8\_video\_tutorial.ipynb.
3. https://www.kaggle.com/code/jpmiller/horse-racing-welcome-to-the-machine/
4. https://github.com/math-silva/YOLO-Parking-Spot .
5. https://www.betfair.com/exchange/plus/pt/corrida-de-cavalos-apostas-7.
6. https://www.youtube.com/watch?v=F7uw3oPC1wQ&list=PLsyobOqUhkthjvmA\_s7tTjb7V2EiwYYGC&index=2.
7. https://yolov8.com/
8. https://medium.com/@Yaga987/hyperparameter-tuning-with-random-forest-ff62882fd5bb

[9]https://www.kaggle.com/code/funxexcel/p3-random-forest-tuning-randomizedsearchcv

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.**