INSTITUTO UNIVERSITÁRIO DE LISBOA

Relatório Previsão de Sets de uma Partida de Ténis

Curso: Licenciatura em Ciência de Dados
Unidade Curricular: Projeto Aplicado em Ciência de Dados I
Docente: Sérgio Moro
Ano Letivo: 2024/2025

Grupo 1

Amanda Viapiana nº 123408,

Inês Machado nº 123435,

Leonor Gouveia nº 123428,

Sílvia Gentil nº 123426,

Tiago Woodger nº 123385.

Índice

Introdução

Este relatório, desenvolvido no âmbito da disciplina de **P**rojeto **A**plicado em **C**iência de **D**ados I, tem como objetivo aplicar conceitos e técnicas lecionadas ao longo do curso de Ciência de Dados, utilizando o *dataset* "atpplayers.json". Através da utilização de linguagens de programação como **R** e **Python**, e de *softwares* como **MongoDB**, bem como outras ferramentas de análise e tratamento de dados, pretende-se explorar padrões, tendências e *insights* relevantes no ténis, demonstrando a aplicação prática dos conhecimentos adquiridos.

O ténis, sendo um desporto com grande disponibilidade de dados, proporciona condições ideais para aplicar técnicas de análise preditiva. Este projeto visa analisar o perfil e o desempenho dos jogadores, condições de jogo e variáveis estratégicas, com o objetivo de desenvolver um modelo que prevê o número de sets de uma partida. Assim, alia a consolidação de conhecimentos teóricos a uma melhor compreensão *data-driven* do desporto.

A metodologia adotada neste projeto é o *Cross-Industry Standard Process for Data Mining* (**CRISP-DM**), um modelo amplamente reconhecido no âmbito da ciência de dados. Publicado em 1999 para uniformizar os processos de *data mining* em diferentes indústrias, o **CRISP-DM** tornou-se a metodologia mais utilizada em projetos de análise de dados, garantindo uma abordagem estruturada e eficiente. Devido ao seu caráter cíclico, permite retornar a etapas anteriores quando novos dados ou *insights* surgem. Este processo divide-se em seis fases principais:

1. Business Understanding - Compreensão do problema de negócio.
2. Data Understanding - Exploração e análise inicial dos dados.
3. Data Preparation - Processamento e limpeza dos dados.
4. Modeling - Construção e teste de modelos preditivos.
5. Evaluation - Avaliação dos resultados obtidos.
6. Deployment - Implementação do modelo, na prática.

Por fim, espera-se que este trabalho demonstre a importância da ciência de dados no desporto, destacando como a análise de dados pode ser utilizada para extrair conhecimento valioso e apoiar decisões estratégicas. A combinação de ferramentas tecnológicas e metodologias robustas, como o **CRISP-DM**, permitirá uma abordagem sistemática e replicável, alinhada com as melhores práticas da área.

1. Business Understanding

O objetivo principal do projeto consiste no desenvolvimento de um modelo preditivo capaz de estimar o número de sets necessários para a conclusão de um jogo de ténis. A previsão deverá ser feita com base num conjunto de variáveis descritivas relacionadas com os dois jogadores envolvidos na partida e com o contexto do jogo. O país atribuído ao nosso grupo para a realização deste trabalho foi a Alemanha, pelo que a análise será focada exclusivamente nos dados dos jogos realizados na Alemanha.

O ténis é um desporto individual (ou de duplas) jogado em campos de diferentes superfícies (duro, terra batida, relva, carpete). Um jogo é dividido em sets, e cada set é composto por vários jogos (*games*). Para vencer um set, um jogador precisa geralmente ganhar seis jogos, com pelo menos dois de vantagem. Caso haja empate (por exemplo, 6 - 6), joga-se um *tie-break* para decidir o vencedor do set.

Nos torneios profissionais, o número de sets necessários para vencer um jogo varia conforme o tipo de torneio. Na maioria dos torneios, os jogos são disputados à melhor de três sets, ou seja, vence quem ganhar dois sets primeiro. No entanto, nos torneios de maior prestígio, como os *Grand Slams*, os jogos masculinos são disputados à melhor de cinco sets, vencendo quem alcançar três sets primeiro. Esta variação tem impacto direto na duração, intensidade e exigência física das partidas.

Este tipo de modelo tem várias aplicações práticas que podem gerar valor para diferentes intervenientes ligados ao ténis profissional:

- Organizadores de torneios podem utilizar previsões da duração dos jogos (através da estimativa do número de sets) para otimizar a gestão de horários e infraestrutura.
- Casas de apostas podem enriquecer os seus modelos de previsão e oferecer mercados mais sofisticados relacionados com a duração dos jogos.
- Comentadores e analistas desportivos podem utilizar este tipo de informação como suporte para análises pré-jogo mais informadas.
- Treinadores e jogadores podem analisar padrões e antecipar o esforço exigido em diferentes condições de jogo.

Prevê-se, portanto, a necessidade de estimar com precisão a quantidade de sets jogados num determinado encontro, algo que depende de uma multiplicidade de fatores. O número de sets pode variar significativamente consoante o tipo de torneio, as características dos jogadores (como idade, altura, experiência ou estilo de jogo), e as condições do jogo (como tipo de superfície, prémio monetário e fase do torneio).

2. Data Understanding

2.1. Visão Geral da Base de Dados

A base de dados "atpplayers.json" possui um total de 1 308 835 observações e inicialmente contém informações sobre 9 960 jogadores distintos. Destas observações, 66 220 são de jogos realizados na Alemanha e incluem 4 101 jogadores distintos em torneios alemães.

Cada observação na base de dados "atpplayers.json" representa um jogo específico visto da perspetiva de um dos jogadores. As informações pessoais (como altura ou país de nascimento) referem-se sempre a esse jogador, com exceção do "GameRank" que se refere ao oponente, enquanto dados como "Location" ou "Tournament" caracterizam o jogo em si.

Esta estrutura faz com que cada partida apareça duas vezes na base de dados, uma vez sob a perspetiva de cada participante, o que permite comparar desempenhos individuais, identificar padrões de vitórias/derrotas e avaliar como as condições do jogo (como piso ou localização) influenciam o número de sets disputados.

Variáveis Relativas ao jogador "PlayerName":

- "PlayerName", "Born", "Height", "Hand", "LinkPlayer", "WL", "Score".

Variáveis Relativas ao jogador "Oponent":

- "Oponent", "GameRank".

Variáveis Relativas ao Jogo (que se repetem em ambas as perspetivas):

- "Tournament", "Location", "Date", "Ground", "Prize", "GameRound".

2.2. Descrição das Variáveis

Cada observação é composta por 15 campos, que são:

- PlayerName – Nome do jogador
- Born – Local (país e/ou cidade) de nascimento do jogador
- Height – Altura do jogador em centímetros
- Hand – Mão dominante e *backhand* do jogador
- LinkPlayer – Link de acesso ao jogador no site da *ATP*
- Tournament – Nome do torneio
- Location – Local do jogo
- Date – Data de início e fim do torneio
- Ground – Tipo de superfície do torneio (*Clay*, *Hard*, *Carpet* ou *Grass*)
- Prize – Prémio atribuído ao vencedor do torneio
- GameRound – Ronda do torneio

- GameRank – *Ranking* do oponente naquele jogo, no *ranking* do *ATP* Tour
- Oponent* – Nome do oponente do jogador
- WL – Resultado da partida: W, caso o jogador vença o jogo e L, caso contrário
- Score – O resultado registado para cada set

Então, podemos classificar estas variáveis para compreender melhor cada uma delas, podendo assim manipulá-las corretamente a fim de chegarmos ao objetivo pretendido:

Variáveis Categóricas Nominais: PlayerName, Born, Hand, LinkPlayer, Tournament, Location, Ground, GameRound, Oponent, Score.

Variáveis Categóricas Binárias: WL.

Variáveis Numéricas Discretas: GameRank.

Variáveis Numéricas Contínuas: Height, Prize.

Data: Date.

*A palavra "Oponent" é um erro de digitação para "Opponent", que será o termo utilizado a partir de agora.

2.3. Particularidades e Problemas da Base de Dados

Jogos espelhados

Um dos principais desafios desta base de dados é a duplicação de registos para cada jogo, uma vez que cada partida é registada a partir de duas perspetivas distintas. Essa redundância permite reunir mais informação sobre os jogadores, já que os dados completos apenas estão disponíveis quando o jogador aparece no campo "PlayerName". Se um jogador surge apenas como "Opponent", não é possível aceder a todas as suas características.

Jogadores diferentes com o mesmo "PlayerName"

Para os jogos realizados na Alemanha, existem 3283 valores distintos no campo "LinkPlayer", mas apenas 3282 valores distintos no campo "PlayerName". Isto indica a existência de um par de jogadores com nomes iguais. O nome em caso é "Andreas Weber".

Jogos terminados por meios que não uma vitória/derrota

Nem todos os jogos terminam através da marcação de pontos. Daqueles realizados na Alemanha, existem 3 tipos de resultados presentes nos dados:

- *Walkover* (W/O): jogos que nem começam, pois um dos jogadores não consegue jogar (não foi à partida, sofreu uma lesão antes do jogo, etc.).
- *Retirement* (RET): um jogo que começa, mas não termina devido à incapacidade de um dos jogadores (sofreu uma lesão, estava doente, etc.).

- *Default* (DEF): Um dos jogadores é desqualificado, levando à vitória por defeito do seu oponente.

3. Data Preparation

3.1. Tratamento de Dados

O tratamento de dados é uma etapa fundamental, na qual a informação é preparada para ser utilizada no modelo. Para este processo, recorreu-se ao **MongoDB**, uma base de dados NoSQL flexível que permite armazenar e gerir grandes volumes de dados não estruturados, facilitando consultas e atualizações. Além disso, utilizou-se o **RStudio**, um ambiente de desenvolvimento integrado para a linguagem **R**, ideal para manipulação, análise e visualização de dados de forma eficiente.

Nesta etapa, recorreu-se também ao **Python** e ao **Excel** para tarefas intermédias, que posteriormente seriam realizadas no **R** e no **MongoDB**, tais como a construção de tabelas de dados e a otimização da criação de comandos no **MongoDB**.

Esta etapa incluiu a adição de dados, criação de variáveis, e outros passos que asseguram a qualidade dos dados antes de serem aplicados no modelo.

3.1.1. MongoDB

As etapas iniciais do tratamento de dados foram realizadas no **MongoDB**.

1. **Correções Iniciais**

Antes de serem filtrados apenas os dados dos torneios que aconteceram na Alemanha, foram realizadas operações de *update* para padronizar os dados na coleção "player". Primeiro, os valores de "Score" com as *strings* "00 00 (W/O)" e "0 (W/O)" foram unificados para "(W/O)", afetando 3 documentos. Em seguida, todos os registos com "GameRank" igual a "-" (99819 documentos) tiveram esse campo atualizado para uma *string* vazia (""), garantindo consistência. Por fim, os valores numéricos no campo "Score" (41 documentos) foram convertidos para *strings*, assegurando que o tipo de dados fosse uniforme em toda a coleção. Estas alterações visam eliminar inconsistências e facilitar análises futuras.

2. **Variáveis Adicionadas pt.1**

Esta etapa consistiu em adicionar novas variáveis que serão úteis durante este processo.

➔ **ID**

O campo "ID" foi criado para atribuir uma identificação única a cada jogador, permitindo identificar casos duplicados ou evitar conflitos em casos onde jogadores compartilham as mesmas informações, como nome, local de nascimento, altura e mão dominante. Nesses casos,

o link é a única característica que diferencia os jogadores, por isso foi utilizado para gerar o "ID". Como o link completo é extenso e pouco prático, foi extraído o código de 4 caracteres presente nele, que também é usado pela *ATP* como identificação oficial do jogador.

➔ **LocationCountry**

A variável "LocationCountry" foi criada para identificar de forma clara o país de realização dos torneios, partindo da variável original "Location", que apresentava formatos inconsistentes (como apenas cidades, combinações de cidade/país ou somente países). Para uniformizar esta informação, desenvolveu-se um processo em duas etapas: quando o país estava explícito em "Location", extraiu-se diretamente; nos casos com apenas localidades, cruzou-se os dados com um dataset que continha cidades do mundo [1] para atribuir o país correto. Esta abordagem permitiu filtrar com precisão os torneios realizados na Alemanha ("LocationCountry": "Germany"), resolvendo as ambiguidades dos dados originais.

Devido ao grande volume de dados, não foi viável verificar, um a um, se existiam casos em que cidades potencialmente alemãs estavam associadas a países incorretos na variável "Location". Embora esta abordagem tenha permitido identificar a maioria dos torneios realizados em solo alemão, é possível que alguns jogos tenham sido classificados incorretamente devido a erros nos dados originais. Assim, é importante reconhecer que podem existir mais jogos na Alemanha do que os identificados através da variável "LocationCountry".

➔ **BornCountry**

A variável "BornCountry" foi criada para armazenar exclusivamente o país de nascimento do jogador, resolvendo o problema da heterogeneidade do campo original "Born", que, por vezes, incluía país e cidade, apenas localidades/territórios ou, noutros casos, só o país. A sua padronização é essencial para identificar claramente a nacionalidade de cada jogador, algo que seria inviável com os dados inconsistentes do campo "Born", dificultando análises e manipulações futuras. Em fases posteriores do tratamento de dados, foram adicionados dados a esta variável.

➔ **DOB**

Foi adicionada uma nova variável chamada "DOB" (*Date of Birth*), com o objetivo de enriquecer a análise com informações relativas à idade dos jogadores.

➔ **BirthYear**

A partir da variável "DOB", foi também criada uma variável numérica chamada "BirthYear", que representa apenas o ano de nascimento do jogador. Esta transformação auxilia numa análise mais simplificada e eficiente da idade dos atletas.

➔ **OpID, OpBornCountry, OpHand, OpHeight, OpDOB, OpGameRank, OpBirthYear, OpAge**

Para consolidar todas as informações relevantes de ambos os jogadores numa única observação, foram criadas variáveis adicionais relativas ao oponente, nomeadamente "OpID", "OpBornCountry", "OpHand", "OpHeight", "OpGameRank", "OpDOB", "OpBirthYear" e "OpAge". Estas variáveis permitem manter na mesma linha de dados não só as características do jogador em "PlayerName", mas também as do "Opponent". Esta estrutura é essencial para eliminar os "espelhos" dos jogos. Ao agregar os dados dos dois jogadores numa única entrada, torna-se possível realizar análises comparativas diretas entre os participantes de cada jogo, como a diferença de idade, altura, *ranking* ou país de origem, sem necessidade de cruzamentos adicionais entre observações.

3. Seedings

No ténis, existe um sistema designado por *seeding*, o qual consiste na isenção de um jogador de disputar uma determinada ronda de um torneio em virtude do seu posicionamento favorável no *ranking*. Nestas situações, o atleta avança automaticamente para a fase seguinte, uma vez que não lhe é atribuído um adversário. Na base de dados, estas ocorrências são assinaladas quando o campo relativo ao oponente apresenta a designação "*bye*", denotando a ausência de um jogo efetivamente realizado. No contexto dos torneios alemães, registaram-se um total de 1349 casos de *seeding*.

Uma vez que estas observações com "Opponent" = "*bye*" não correspondem a jogos reais, foram todas eliminadas da coleção "player" e colocadas na coleção "excluded".

4. Localizações Erradas - Location/LocationCountry

Após identificar instâncias com o formato {Location: "cidade, germany"}, verificou-se manualmente se cada localidade pertencia de fato à Alemanha. Detetaram-se registos incorretos, como:

- "**oberentfelden**, germany", este município situa-se na Suíça e não na Alemanha, logo, as observações com esta localização foram eliminadas.
- "**ostend**, germany", esta é uma cidade belga, por isso estas observações também foram apagadas.
- "**hong kong**, germany", Hong Kong é um território autónomo na China, no entanto, o jogo realmente aconteceu na Alemanha, por isso as entradas com esta "Location" foram mantidas, ficando a localização apenas "Germany" sem informação do município.

Após este passo, procedeu-se à filtragem das observações com "LocationCountry": "Germany". A coleção original, contendo todos os torneios, foi renomeada para "original", enquanto os dados filtrados, correspondentes exclusivamente a torneios realizados na Alemanha, passaram a constituir a nova coleção "player".

5. Separação da Variável Hand e Padronização das Categorias

Na base de dados original, a variável "Hand" continha observações no formato:

“Hand”: “Right-Handed, Two-Handed Backhand”

Ou seja, agregava duas informações distintas:

1. A mão dominante do jogador (“Hand”), que indica se este segura a raquete com a direita (*Right-Handed*) ou com a esquerda (*Left-Handed*) ou se é ambidestro (*Ambidextrous*);
2. O tipo de *backhand* (“BackHand”), que especifica se o jogador executa o revés com uma mão (*One-Handed*) ou com duas mãos (*Two-Handed*).

Para facilitar a análise, esta variável foi **dividida em duas**:

- **“Hand”** (Mão dominante): *Right-Handed*, *Left-Handed* ou *Ambidextrous*;
- **“BackHand”** (Tipo de revés): *One-Handed* ou *Two-Handed*.

A seguir, foi criada a variável **“OpBackHand”**.

Esta separação permite uma exploração mais clara e individualizada das características técnicas dos jogadores.

Também foi feita a padronização dos valores nas duas variáveis onde:

- Os valores omissos, escritos como “null”, “Unknown” e “Unknown BackHand” foram transformados em *strings* vazias (“”).
- Todos os valores que não estavam na formatação correta (como, por exemplo, “ambidextrous” em vez de “Ambidextrous”) foram adaptados às categorias existentes, para ambas as variáveis.

6. Valores Errados na Variável Height

Foram encontrados valores claramente incorretos nas alturas como 0, 15, 71 e 510 cm. Estes dados foram substituídos pelos valores corretos correspondentes.

7. Adição de Dados

Como mencionado anteriormente, a base de dados apresenta valores omissos, que foram preenchidos através de uma metodologia estruturada. Inicialmente, recorreu-se a fontes externas como o site da *ATP* [2], além de outras fontes presentes na bibliografia ([3], [4], [5], [6], [7], [8], [9], [10]), para recolher a informação em falta, utilizando o Excel para criar tabelas auxiliares. Estas tabelas foram preenchidas manualmente ou através de *web scraping* do site *ATP* Tour [2] com **Python**, garantindo a integração dos dados ausentes.

Para assegurar a correta identificação dos jogadores, utilizou-se o “ID” único e o “PlayerName”, o que permitiu distinguir atletas com nomes idênticos. No entanto, esta abordagem não foi possível para os jogadores fora do *ranking* da *ATP* (só presentes em “Opponent”) que não constam como “PlayerName” na base de dados e por isso não têm um “ID” atribuído através do “LinkPlayer”. Apesar disso, para viabilizar análises futuras e dada a relevância destes registos, optou-se por pesquisar esses jogadores com base apenas no seu nome (“Opponent”), a única informação disponível além do “GameRank” (que neste caso não

tem utilidade). Isso implica que, em casos de homónimos, não é possível diferenciar dois jogadores distintos, sendo esta uma limitação do método utilizado.

Para otimizar a geração dos comandos de *update* no **MongoDB**, recorreu-se ao **Python**, utilizando as tabelas auxiliares previamente criadas no **Excel** como fonte de dados. Este processo permitiu gerar automaticamente todos os comandos necessários, que foram depois exportados para ficheiros de texto (*.txt*). Dada a elevada quantidade de operações a realizar, esta abordagem automatizada revelou-se essencial para garantir eficiência e precisão na atualização em massa da base de dados.

Não foram adicionados dados à variável "BackHand" e "OpBackHand", uma vez que a grande maioria dos jogadores não possui dados disponíveis sobre o tipo de *backhand*.

A tabela seguinte apresenta as variáveis atualizadas com novos dados e os progressos feitos.

Variáveis	**Qtd. de dados omissos originalmente**	**Qtd. de dados adicionados**	**Qtd. de dados omissos que sobraram**
BornCountry*	3283	3283	0
Hand	674	673	1
Height	1239	45	1194
DOB	3283	3277	6
BirthYear	3283	3282	1

*Para preencher a tabela "BornCountry", optou-se por utilizar *web scraping*, uma vez que esta abordagem se revelou mais rápida e eficiente do que extrair a informação manualmente a partir do campo "Born".

A base de dados contém 818 adversários que nunca apareceram como "PlayerName". A tabela seguinte mostra os dados adicionados especificamente para estes oponentes.

Variáveis	Qtd. de dados omissos originalmente	Qtd. de dados adicionados	Qtd. de dados omissos que sobraram
OpBornCountry	818	817	1
OpHand	818	811	7
OpHeight	818	296	522
OpDOB	818	805	13
OpBirthYear	818	807	11

8. Alteração da Estrutura das Observações

Para resolver os espelhos, primeiro foi preciso fazer uma alteração na estrutura dos dados, para tal, foram seguidos os seguintes passos:

1. Consolidação de Dados dos Jogadores

- Para cada observação, foram inseridas as informações de ambos os jogadores ("PlayerName" e "Opponent").
- Os jogadores presentes na coluna "PlayerName" da base de dados original foram adicionados a uma tabela secundária.
- Utilizou-se um comando *lookup* para atualizar as informações desses jogadores nos jogos em que apareciam como "Opponent".

2. Criação da Tabela de Jogos

Foi criada uma tabela ("games") com a seguinte estrutura:

- Jogos em que o "PlayerName" venceu:
  - Os seus dados foram prefixos com "W" (*Winner*).
  - Os dados do "Opponent" (derrotado) foram prefixos com "L" (*Loser*).
- Jogos em que o "Opponent" venceu:
  - Os seus dados receberam "W", enquanto os do "PlayerName" (derrotado) ficaram com "L".

3. Base de dados

- A tabela "games" (ainda com espelhos) foi exportada do **MongoDB** e importada para o **R** para análise.
- A tabela "original" também foi importada para **R**, servindo como referência.

Desta forma, a variável "W/L" foi eliminada, uma vez que a nova estrutura já identifica claramente o vencedor e o perdedor em cada jogo.

3.1.2. RStudio

1. Observações Duplicadas

O conjunto de dados continha 305 observações duplicadas, isto é, registos com todos os campos exatamente iguais, o que é distinto dos chamados "jogos espelhados". Estes dados foram eliminados no **R**, mantendo apenas uma observação de cada, passando a ficar com 64 566 observações no total.

2. Observações Espelhadas

O conjunto de dados exportado do **MongoDB** ainda contém jogos espelhados, sendo que a transformação feita antes da importação permitiu uma fácil remoção deles. Quando todas as informações, dos jogadores e sobre o jogo/torneio específico, são as mesmas, menos o valor da coluna "Score", que se encontra espelhado (ex.: um jogo possui o valor "60 60" e o jogo na outra perspetiva possui o valor "06 06"), temos um caso de jogo espelhado.

Estas instâncias repetidas também foram eliminadas, certificando que nenhum jogo está representado mais que uma vez. Ao fim, a base de dados ficou com 34 026 observações.

3. Uniformização da Variável Location

A variável "Location", que inicialmente apresentava formatos heterogéneos, foi padronizada para seguir a estrutura "município", uma solução que teve em conta particularidades geográficas específicas, nomeadamente o caso alemão, onde todas as cidades são municípios, mas nem todos os municípios têm estatuto de cidade. Para garantir a consistência dos dados, nos casos em que a localização indicava apenas um bairro, procedeu-se à identificação do município correspondente.

Após esta padronização, o campo "Location" passou a exibir apenas o município, ficando como *NA* nos casos em que essa informação não estava disponível, uma vez que manter o país seria redundante, pois todos os jogos ocorreram na Alemanha.

4. Criação de IDs

Como alguns jogadores surgiam apenas como "Opponent" e nunca apareciam no campo "PlayerName", não havia acesso ao "LinkPlayer" desses jogadores, o que impedia a criação do identificador único "ID", foi necessário criar identificadores artificiais para esses jogadores, garantindo que todos os participantes tivessem um "ID" atribuído. Para resolver isso, criámos identificadores únicos para esses jogadores, atribuindo um "ID" sequencial com base no número de jogadores na base de dados. Após a criação de ID's novos, estes foram associados aos oponentes nos campos correspondentes, garantindo que todos os jogadores,

independentemente de serem o jogador principal (“PlayerName”) ou não, tivessem um “ID” único.

5. Separação da Variável Date

Inicialmente, a variável “Date”, que tinha as datas dos torneios no formato de texto **(ex.: Date: ‘2021.06.28 - 2021.07.11’)**, foi reformulada para melhorar a sua utilidade analítica. Como algumas observações continham apenas a data de início (ex.: **Date: “1978.07.03”**), criaram-se duas novas colunas: “StartDate” para a data início de torneio e “EndDate” para a data final. Além desta separação, os valores foram convertidos para o formato de data padrão (AAAA-MM-DD), permitindo operações temporais mais eficientes. Os casos sem data final foram mantidos como *NA* (*Not Available*).

6. Tournament

Foram realizadas alterações na variável “Tournament” com o objetivo de uniformizar os nomes dos torneios e corrigir inconsistências que dificultavam a análise.

Em primeiro lugar, todos os jogos da *Davis Cup*, que apareciam sob diversos formatos e descrições, frequentemente contendo abreviações, nomes dos países envolvidos e fases da competição, foram padronizados para o nome único “DAVIS CUP”. Esta harmonização foi essencial para permitir a filtragem correta dos jogos deste torneio e evitar a fragmentação das análises.

Além disso, foram unificados nomes de torneios que apareciam com variações ou múltiplas designações para a mesma competição. Por exemplo, “Munich-2” foi renomeado para “Munich”, e o torneio “Dusseldorf-2” foi agregado sob a designação “Dusseldorf”, uma vez que se referem ao mesmo evento.

Do mesmo modo, “Ulm” foi renomeado para “Neu Ulm”, e “Stuttgart” e “*ATP Masters 1000 Stuttgart*” foram diferenciados como “Stuttgart-1” e “Stuttgart-2” para distinguir edições ou categorias distintas do mesmo local.

7. Valores Errados na Variável Score

Foram corrigidos 14 valores incorretos na variável “Score”. Algumas observações continham erros de formatação, como a ausência de espaços entre os números, ou sequências que não correspondiam a resultados válidos. Além disso, casos em que os jogos foram interrompidos passaram a incluir a marcação “(RET)” para indicar desistência.

Ex.: o valor “67 75 40” foi corrigido para “67 75 40 (RET)”, assinalando que o jogo não acabou.

8. Variáveis Adicionadas pt.2

➔ **Sets**

Com o objetivo de prever o número de sets jogados em cada partida, foi criada a variável "Sets", derivada da variável "Score". Esta transformação foi essencial para transformar a informação textual do "Score" num formato numérico mais adequado para análise e modelação. Para a maioria dos jogos, o número de sets foi calculado contando o número de espaços na *string* da pontuação e somando 1, já que cada espaço separa dois sets distintos. A criação da variável "Sets" permitiu, assim, padronizar esta informação e torná-la diretamente utilizável como variável-alvo na tarefa de previsão do número de sets em cada jogo.

➔ **NF**

Foi também criada a variável binária "NF" (*Not Finished*), que assume o valor 1 quando o jogo não foi concluído de forma normal — ou seja, quando o resultado corresponde a um dos casos especiais: *Retirement* (RET), *Walkover* (W/O) ou *Default* (DEF) — e 0 nos restantes jogos. Esta variável foi desenvolvida com o propósito de simplificar o processo de filtragem dos dados, permitindo identificar de forma rápida e eficiente os jogos que não terminaram. A criação de "NF" é especialmente relevante porque esses jogos incompletos não fornecem informação fiável sobre a dinâmica completa dos jogos, como o número real de sets disputados, sendo por isso inadequados para o treino do nosso modelo preditivo. Com esta variável, torna-se possível excluir, no futuro, esses casos da base de dados.

➔ **Year**

Foi criada a variável "Year" a partir do ano da variável "StartDate". Esta variável vai ser importante, para fazer as alterações necessárias no "Prize", identificar os jogos disputados à melhor de 5, entre outras transformações necessárias.

➔ **WAge e LAge**

Com base na variável "DOB" e "StartDate", foram criadas as variáveis "WAge" e "LAge", que representam a idade aproximada do jogador no momento em que cada jogo ocorreu. Para esse cálculo, foi utilizado especificamente a data de início do torneio, o que significa que a idade atribuída a cada jogador corresponde à sua idade no início de cada competição. A idade foi obtida subtraindo a data de nascimento à data de início do torneio.

Winner	WDOB	StartDate	WAge
Roger Federer	1981-08-08	1999-01-25	17
Roger Federer	1981-08-08	2000-06-12	18
Roger Federer	1981-08-08	2000-10-30	19

Figura 1 - Exemplo da variável "WAge"

Para alguns jogadores, apenas havia disponível o ano de nascimento ("BirthYear"), sem informação sobre o dia e mês de nascimento ("DOB"). Nestes casos, foi considerado que o jogador faz anos a 1 de janeiro, ou seja, foi calculada a idade máxima possível que ele poderia ter no ano da partida, este cálculo é a diferença entre "Year" e o "BirthYear" do jogador.

Este método é uma alternativa à imputação da média nos valores omissos, que permite um valor mais próximo do valor real.

➔ WGameRankMean e LGameRankMean

A variável "GameRankMean" foi criada com o objetivo de fornecer uma estimativa mais robusta e contínua do nível de desempenho recente de um jogador, baseada no histórico de *rankings* nos anos anteriores ao jogo. Esta variável foi calculada separadamente para o vencedor ("WGameRankMean") e para o perdedor ("LGameRankMean") de cada jogo.

Para o seu cálculo, considerou-se a média dos *rankings* do jogador nos três anos anteriores ao ano do jogo atual. Por exemplo, se uma partida ocorreu em 2015, o cálculo da média foi feito com base nos *rankings* disponíveis entre 2012 e 2014. Esta média foi obtida apenas com os valores disponíveis (ignorando *NA*) e atribuída à nova variável.

Além disso, nos casos em que o "GameRank" de um jogador estava em falta no *dataset* original (*NA*), foi feita a imputação com o valor correspondente de "GameRankMean", garantindo que o modelo preditivo tenha uma estimativa de *ranking* para todos os jogos.

9. Tratamento da Variável Prize

➔ Uniformização da Variável "Prize"

A variável "Prize" regista os valores monetários dos prémios, tendo sido identificadas 8006 observações com o símbolo "□", que correspondem a erros de registo onde deveria constar o símbolo "€" [2]. Este problema revelou uma questão mais ampla: a coexistência de valores em dólares e euros na base de dados, o que impossibilita comparações diretas entre observações devido às diferentes moedas.

Para resolver esta limitação, recorreu-se a uma tabela de conversão* com as médias anuais de câmbio, calculadas a partir de um ficheiro **Excel** (.xlsx) de valores diários publicados pelo Banco de Portugal [11] entre 04/01/1999 e 29/12/2023. Para criar o novo ficheiro com a média calculada foi utilizado **Python**. Optou-se por converter todos os valores para dólares, uma vez que o euro só entrou em circulação em 1 de janeiro de 1999, tornando inviável a conversão inversa para períodos anteriores. O cálculo de conversão seguiu seguinte fórmula:

Prize em euros x Média da Taxa de câmbio anual = Prize em dólares

Contudo, é importante salientar algumas limitações deste método:

1. A desvalorização monetária ao longo do tempo - por exemplo, de acordo com *US Inflation Calculator* [12], $100 em 1968 equivaliam aproximadamente a $840,96 em 2022, devido aos efeitos cumulativos da inflação. Ou seja, um produto comprado a $100 em 1968, custaria $840,96 em 2022.
2. Caso se resolvesse a primeira limitação, seria necessário considerar a diferença nas taxas de inflação entre o euro e o dólar.

➔ Prize na Davis Cup

O valor monetário associado aos jogos da *Davis Cup* é diferente dos restantes torneios, uma vez que, neste caso, o valor é atribuído à federação nacional. Noutros torneios, o prémio monetário é entregue diretamente ao vencedor, enquanto na *Davis Cup*, cada federação decide quanto cada jogador irá receber. As federações podem, ou não, distribuir parte do prémio aos jogadores e esse valor pode variar bastante entre países, dependendo das políticas internas, dos acordos com os jogadores e do respetivo orçamento [13].

Por isso, o valor não é fixo e, na maioria das vezes, não é divulgado publicamente, o que dificulta saber o valor exato que cada jogador recebe por jogo. Assim, optámos por indicar o valor como 0, de forma a evitar a inclusão de dados imprecisos ou especulativos.

Além disso, a participação na *Davis Cup* tem, acima de tudo, um significado simbólico e patriótico, sendo motivada principalmente pela honra de representar o país, mais do que por razões financeiras.

- ➔ **Prize do torneio "Munich"**

Nos anos de 1968 e 1969, atribuímos o valor do prémio monetário do torneio de "Munich" como 0. Esta decisão deve-se ao facto de, na época, não ter sido atribuído qualquer prémio financeiro aos participantes [14].

Durante esse período, o circuito profissional de ténis ainda se encontrava em desenvolvimento e muitos torneios, especialmente os de menor dimensão, não ofereciam prémios monetários. O torneio de "Munich" é um desses casos, em que a participação não era associada a um valor económico direto para os jogadores.

3.1.3. Estrutura Final das Observações

Na nova estrutura, cada observação representa um jogo com a perspetiva de ambos os jogadores. As características de cada um estão agora presentes na mesma linha, junto com os dados do jogo. As variáveis com o prefixo "W" referem-se ao vencedor (*Winner*), enquanto as com "L" correspondem ao perdedor (*Loser*).

Figura 2 - Transformação na estrutura das observações

Assim, após o tratamento de dados as variáveis passam a ter os seguintes de tipos:

Variáveis Categóricas Nominais: Winner, Loser, WHand, LHand, WBackHand, LBackHand, WID, LID, WBornCountry, LBornCountry, Location, GameRound, Ground, Score, Tournament.

Variáveis Categóricas Ordinais: Sets.

Variáveis Categóricas Binárias: NF.

Variáveis Numéricas Discretas: WAge, LAge, WGameRank, LGameRank, WGameRankMean, LGameRankMean, WBirthYear, LBirthYear.

Variáveis Numéricas Contínuas: WHeight, LHeight, Prize.

Data: StartDate, EndDate, WDOB, LDOB.

3.2. Limpeza de Dados

Nesta fase, os dados serão limpos e preparados com o objetivo de serem usados no modelo. Ou seja, serão selecionados, de forma geral, os dados que forem relevantes para o modelo.

3.2.1. Remoção de Jogos à Melhor de 5

A variável "Sets" foi definida como variável-alvo do modelo a construir. No entanto, como existem jogos disputados à melhor de 5 sets e outros à melhor de 3, estamos perante duas estruturas de jogo distintas. Por exemplo, num encontro à melhor de 5, a ocorrência de 3 sets pode significar que o jogo terminou por *retirement* ou que o mesmo jogador venceu os três primeiros sets. Já num jogo à melhor de 3, a presença de 3 sets indica necessariamente que o jogo estava empatado, sendo o terceiro set decisivo para determinar o vencedor. Assim, o mesmo número de sets pode ter significados diferentes consoante o formato do jogo, o que compromete a consistência da variável-alvo. Dado que tal situação inviabiliza a construção de um modelo fiável com uma única variável-alvo, optou-se por considerar apenas os jogos disputados à melhor de 3 sets, assegurando a homogeneidade dos dados e a coerência na interpretação do alvo.

Verificou-se a utilização do formato à melhor de cinco sets em diversos torneios de ténis, com variações consoante o ano e a fase do torneio.

Davis Cup:

- Todos os encontros realizados entre 1968 e 2017 foram disputados à melhor de cinco sets.

Torneio de Munich:

- 1971: Todas as rondas.
- 1975: Apenas nas meias-finais e na final.
- 1978, 1982 e 1984: Apenas a final.

Torneio de Hamburg:

- 1968 - 1973: Todas as rondas.
- 1974 e 1982: Apenas nas meias-finais.
- 1975, 1979 - 1981, 1983: Meias-finais e final.
- 1978, 1984, 1986 - 1989: Apenas na final.

Cologne WCT:

- Apenas a final da edição de 1973 foi disputada à melhor de cinco.

Torneio de Dusseldorf:

- 1974: Meia-final.

- 1975 e 1976: Final.

ATP Masters 1000 Hamburg:

- A final foi disputada à melhor de cinco nas edições nos anos de 1990, 1991, 1993 - 1997 e 1999 - 2006.

ATP Tour World Championship:

- A final foi disputada à melhor de cinco sets entre 1991 e 1999.

Grand Slam Cup:

- As meias-finais e a final foram jogadas à melhor de cinco entre 1990 e 1999.

Torneio Stuttgart-1:

- A final foi disputada à melhor de cinco sets nas seguintes edições: 1978, 1979, 1981, 1982, 1984, 1991 - 1996 e 1999 - 2006.

Torneio Stuttgart-2:

- A final foi disputada neste formato nos anos de 1981, 1990 - 1994 e 1996 - 2001.

Torneio de Berlin:

- Todas as rondas foram disputadas à melhor de cinco nas edições de 1968 e 1973.

ATP Masters 1000 Essen:

- Apenas a final da edição de 1995 foi disputada à melhor de cinco sets

Todos os jogos disputados à melhor de 5 foram eliminados da base de dados, com base no nome do torneio ("Tournament"), o ano ("Year") e a fase do torneio ("GameRound").

3.2.2. Eliminação de Jogos Não Concluídos

Para prever o número de sets, serão considerados apenas os jogos que foram concluídos. Supondo que os jogos sejam disputados à melhor de 3 sets, temos as seguintes possibilidades:

- **0 sets**: nenhum set foi concluído o jogo não começou ou foi interrompido muito cedo.
- **1 set**: o jogo terminou após apenas um set, ou seja, não foi concluído corretamente.
- **2 sets**: o jogador venceu os dois primeiros sets, encerrando o jogo sem necessidade de um terceiro.
- **2 sets**: o jogo estava empatado 1 a 1, mas foi interrompido antes do terceiro set, ou seja, também não foi concluído.
- **3 sets:** o jogo foi interrompido a meio do set decisivo, ou seja, não foi concluído.

- **3 sets**: o jogo foi concluído normalmente, com o vencedor definido no terceiro set.

No modelo, apenas os jogos finalizados serão utilizados. Os jogos não concluídos foram eliminados porque, de acordo com o *Business Understanding*, o objetivo é prever apenas os jogos concluídos. Esses casos foram identificados pela variável "NF", que assume valor 1 quando o jogo não foi concluído.

Por exemplo:

- As casas de apostas não lucram com jogos interrompidos.
- Os jogadores querem avaliar o seu desempenho com base em partidas completas, e não em situações excecionais, como lesões ou abandonos.

Foram eliminados 1152 jogos não terminados.

3.2.3. Eliminação de Erros Cronológicos

Foram eliminados da base de dados 2 jogos onde o ano do jogo não condiz com a idade reportada de um dos jogadores, pois o ano de nascimento deles é depois do ano do jogo (Francesco Scarpa e Florian Merkel). Também foram eliminados jogos atribuídos erroneamente a um jogador por uma das funções corridas no **MongoDB**, por terem o mesmo nome.

3.2.4. Dados Omissos

Todos os valores identificados como omissos, tanto nas variáveis numéricas como nas variáveis categóricas, foram codificados como *NAs*.

A tabela abaixo mostra a quantidade de valores que estão em falta (*missing values*) na base de dados após o tratamento, por variável.

Variáveis Relativas ao Jogador						
Hand	**BackHand**	**BornCountry**	**Height**	**DOB**	**BirthYear**	**Age**
7	2790	0	1685	17	11	11

Tabela 1 - Dados Omissos em relação às variáveis relativas ao jogador

Variáveis Relativas ao Jogador e ao Jogo	
WGameRank	**LGameRank**
2644	3405

Tabela 2 - Dados Omissos em relação às variáveis relativas ao jogador e ao jogo

Variáveis Relativas ao Jogo							
Location	**GameRound**	**Ground**	**Tournament**	**Prize**	**StartDate**	**EndDate**	**Score/Sets**
5612	0	0	0	0	0	660	0

Tabela 3 - Dados Omissos em relação às variáveis relativas ao jogo

GameRank: Imputação da média
Para os jogadores que já tinham valores registados de "GameRank" noutros jogos, foi utilizado o valor da variável "GameRankMean", correspondente à média dos *rankings* dos 3 anos anteriores. Para os jogadores sem qualquer informação disponível sobre "GameRank", foi imputada a média das variáveis "WGameRank" e "LGameRank" presentes na base de dados.

Location: Não vai ser utilizada no modelo
A variável "Location" foi removida do modelo devido ao elevado número de categorias, o que a tornava redundante para a análise preditiva. A ausência de variação geográfica significativa e a dificuldade em agrupar localizações de forma lógica justificam a sua exclusão.

Hand: Imputação da Moda
Dado o reduzido número de casos com valores omissos e a elevada predominância de jogadores destros (88,74%) no conjunto de dados, decidiu-se imputar a moda da variável, que corresponde à categoria "Right-Handed".

StartDate, EndDate, DOB e BirthYear: Não serão utilizadas no modelo.
Estas variáveis não serão utilizadas diretamente no modelo, pois não é indicada a utilização de datas nestes tipos de modelos.

Age: Imputação da média
Para a variável "Age", os poucos valores em falta (11 jogadores) foram imputados utilizando a média da variável, que é 23,87 anos.

BackHand: Foi eliminada a variável "BackHand" por apresentar 68,75% de valores omissos. Segundo Dorian Pyle, no livro *Data Preparation for Data Mining* (1999) [15], variáveis com

elevada ausência de dados devem ser removidas, pois a sua utilidade analítica é limitada e podem introduzir ruído ou enviesar os modelos. Como o objetivo do modelo é prever o número de sets com base em características consistentes dos jogadores e do jogo, manter uma variável tão incompleta comprometeria a integridade da análise. Além disso, o esforço de imputação neste caso não se justifica, dado o baixo valor informativo remanescente dessa variável. Assim, a decisão de remoção alinha-se com boas práticas de preparação de dados na ciência de dados.

3.3. Variáveis Novas Para o Modelo

Para o modelo, foram criadas novas variáveis com base nas variáveis originais da base de dados, com o objetivo de enriquecer a informação disponível, permitir a comparação entre os dois tenistas em jogo e agrupar categorias.

➔ **WinRate**

Esta variável representa o desempenho geral dos jogadores ao longo do tempo, servindo como uma medida de eficácia histórica até ao ponto de cada jogo. Ao incorporar este histórico, é possível avaliar se um jogador geralmente ganha mais do que perde, o que pode ser um forte indicador de performance e consistência.

O jogo da observação atual não é utilizado para o cálculo do "WinRate", então para evitar problemas como *NaN* (*Not a Number*), que ocorre quando não há jogos anteriores à observação atual*,* é utilizado um valor *default* que é 50%, este valor é utilizado na seguinte situação:

- No primeiro jogo do jogador da base de dados, logo não existe nenhuma observação de um jogo do jogador com data anterior à observação atual. Isto também inclui situações onde o jogador aparece apenas uma vez na base de dados.

O valor 0,5 foi escolhido porque:

- Em contextos reais de previsão, o resultado do jogo atual não está disponível, pelo que este valor não deve ser incluído no cálculo da métrica, descartando-se esta opção de usar o valor do jogo atual como *default*, o que seria 0, caso o jogador perdesse nesse jogo, ou 1, caso ganhasse.
- O valor 0,5 representa uma expectativa neutra, ou seja, uma probabilidade de vitória de 50%. Este valor não assume nem o sucesso, nem o fracasso, é uma posição imparcial que indica que, sem histórico, não sabemos se o jogador tende a ganhar ou a perder.

➔ **WinRateDiff**

Com base na variável "WinRate", foi criada a variável "WinRateDiff", que representa a diferença de desempenho entre os dois jogadores de cada jogo. Ela é calculada como a diferença entre a WinRate do jogador do lado "W" ("WWinRate") e o do jogador do lado "L" ("LWinRate").

O valor da diferença é mantido em módulo (valor absoluto), pois o modelo que será desenvolvido não deve considerar quem venceu ou perdeu a partida, já que essa informação ainda não está disponível no momento da previsão.

➔ **GameRankMeanMean**

Esta variável representa a média das posições médias dos *rankings* dos dois jogadores envolvidos no jogo, calculada a partir das variáveis "WGameRankMean" e "LGameRankMean", que indicam a média das classificações dos jogadores vencedores e perdedores, respetivamente, considerando os seus desempenhos anteriores até ao momento do jogo.

O valor da variável "GameRankMeanMean" permite avaliar o nível competitivo do jogo, ao indicar se o encontro envolve jogadores com posições elevadas na classificação (mais próximos do topo) ou com classificações mais baixas, o que pode influenciar a duração e a intensidade do jogo.

➔ **HandDiff**

Foi criada uma nova variável denominada "HandDiff", que representa o cruzamento entre as mãos dominantes dos dois jogadores em cada jogo. Esta variável foi construída a partir das colunas "WHand" (mão do vencedor) e "LHand" (mão do perdedor), categorizando os jogos em três situações distintas:

- Both Right-Handed: ambos os jogadores são destros;
- Both Left-Handed: ambos são esquerdinos;
- Opposite Hands: os jogadores têm mãos dominantes diferentes.

A variável "HandDiff" foi posteriormente transformada em duas variáveis *dummy* (indicadoras), utilizando "Both Right-Handed" como categoria de referência. Esta transformação permite a sua utilização nos modelos preditivos, facilitando a interpretação dos efeitos comparativos de combinações alternativas de mãos dominantes sobre o número de sets de um jogo.

Além das vantagens em termos de simplificação e interpretação, optou-se por utilizar a variável "HandDiff" em vez das variáveis originais "WHand" e "LHand", porque, no contexto de previsão, não se sabe quem será o vencedor ou o perdedor de um jogo. Como "WHand" refere-se à mão do vencedor e "LHand" à do perdedor, estas variáveis só estão disponíveis após o resultado do jogo. Portanto, não faria sentido usá-las como variáveis explicativas num modelo preditivo, pois estariam a introduzir informação futura.

➔ **CountryDiff**

Seguindo a mesma lógica aplicada na criação da variável "HandDiff", foi construída a variável "CountryDiff" a partir das variáveis originais "WBornCountry" e "LBornCountry", que indicam os países de nascimento do vencedor e do perdedor do jogo.

A variável "CountryDiff" indica o número de jogadores alemães presentes em cada partida, assumindo três categorias:

- "Both Germany": ambos os jogadores são alemães;
- "One Germany": apenas um jogador é alemão;
- "None Germany": nenhum jogador é alemão.

Esta variável serve essencialmente para captar a possível vantagem de jogar em casa, uma vez que todos os jogos ocorreram na Alemanha.

Tal como foi feito com a "HandDiff", a "CountryDiff" foi posteriormente transformada em duas variáveis *dummy*, utilizando "Both Germany" como categoria de referência. Esta transformação permite a sua utilização direta nos modelos.

➔ **AgeDiff, Height Diff, GameRankDiff e GameRankMeanDiff**

As variáveis "AgeDiff", "HeightDiff", "GameRankDiff" e "GameRankMeanDiff" foram construídas com base na diferença absoluta entre os atributos dos dois jogadores em cada jogo. Seguindo a mesma lógica aplicada à variável "WinRateDiff", estas variáveis não têm em consideração quem venceu ou perdeu a partida. A utilização do valor absoluto permite captar a magnitude da diferença entre os jogadores, sem introduzir viés direcional. Desta forma, é possível analisar o impacto da diferença de idade, altura ou *ranking* na probabilidade de vitória, mantendo a imparcialidade necessária para o desenvolvimento de um modelo preditivo.

➔ **TournamentGroup**

A agrupação dos torneios foi realizada com o objetivo de simplificar a análise, permitindo agrupar torneios com características semelhantes e reduzir o número de categorias da variável. A redução do número de categorias contribui para uma maior diferenciação entre elas, o que é vantajoso para os modelos.

ATP Tour: Esta categoria reúne os torneios de mais alto nível do circuito masculino, incluindo *ATP Masters 1000*, *ATP 500* e *ATP 250*, que são organizados diretamente pela *ATP* e oferecem maior prémio monetário e mais pontos para o *ranking*. Participam os jogadores mais bem classificados e o nível competitivo é muito elevado, justificando uma categoria exclusiva para estes eventos.

ATP Challenger: Os torneios *Challenger* constituem o segundo escalão do ténis profissional masculino, servindo de ponte entre os torneios *Futures* e o *ATP Tour*. São frequentados por jogadores em ascensão ou por atletas a tentar recuperar posições no *ranking*. Estes torneios oferecem menos pontos e prémios, mas são fundamentais para o desenvolvimento de carreiras e para o acesso ao circuito principal. A sua inclusão numa categoria própria reflete o seu papel intermédio e o perfil dos participantes.

ATP Futures: Inclui os torneios de entrada no circuito profissional, com prémios e pontos reduzidos, destinados a jogadores em início de carreira ou em transição do circuito juvenil para o profissional. O nível competitivo é mais baixo comparado com os *Challengers* e *ATP Tour*, sendo fundamental para a formação e progressão dos tenistas, por isso, justifica-se a sua separação das restantes categorias mais avançadas.

Other: Esta categoria engloba torneios que não pertencem aos circuitos principais da *ATP*, como a *Davis Cup*, *Grand Slam Cup*, *World Team Cup*, *Nations Cup* e outros eventos especiais. Estes torneios têm formatos, regras e objetivos distintos dos torneios regulares, podendo envolver seleções nacionais ou formatos por equipas, o que justifica a sua distinção das restantes categorias.

Esta agregação permite analisar o desempenho dos jogadores e as características dos jogos em diferentes contextos competitivos, respeitando a hierarquia e as especificidades de cada nível do circuito profissional de ténis.

➔ **GameRoundGroup**

Usando a mesma lógica utilizada para o "TournamentGroup", foram agrupadas as rondas dos torneios da seguinte forma:

Qualifying: *1st* e *2nd Round Qualifying*
Estas rondas representam o início do torneio, e são compostas por jogadores que ainda não têm acesso direto ao quadro principal e precisam vencer várias partidas para garantir a entrada. O nível competitivo é geralmente mais heterogéneo, com jogadores de *ranking* inferior, a pressão é elevada, mas o contexto é distinto do quadro principal, justificando uma categoria própria para estas fases.

Early Rounds: *Round of 64* e *Round of 32*
Correspondem às primeiras eliminatórias do quadro principal. Aqui já participam jogadores qualificados e cabeças de série, normalmente existe maior disparidade de nível entre adversários, o que se reflete em jogos mais previsíveis e, muitas vezes, mais curtos. A importância destas rondas reside no facto de serem a porta de entrada para as fases mais competitivas, mas ainda sem a intensidade e equilíbrio das rondas seguintes.

Mid Rounds: *Round of 16* e *Round Robin*
Estas rondas marcam a transição para a parte mais competitiva do torneio. Os jogadores já superaram as primeiras fases e o nível técnico é mais elevado, os encontros tendem a ser mais equilibrados e exigentes, tanto física como mentalmente. A inclusão do *Round Robin* (formato usado em alguns torneios, como as *ATP Finals*) justifica-se porque, apesar da diferença de formato, o grau de dificuldade e a importância dos jogos é comparável ao das oitavas de final, sendo decisivo para o acesso às fases finais.

Final Rounds: *Quarter-Finals*, *Semi-Finals*, *Finals* e *3rd Round Qualifying*
Estas fases são as mais decisivas do torneio, onde a pressão competitiva é máxima e estão em jogo os maiores prémios e pontos para o *ranking*. Os jogadores presentes são, em geral, os melhores do torneio, e cada jogo pode ser determinante para a carreira dos atletas.

A *3rd Round Qualifying* foi incluída neste grupo porque é a última fase antes do quadro principal, tem uma carga emocional e competitiva semelhante às rondas finais do torneio principal. Vencer este jogo pode mudar o percurso de um jogador e garantir-lhe acesso a prémios, pontos e visibilidade, tal como acontece nos quartos de final, nas meias-finais e nas finais.

3.4. Avaliação das Variáveis para os Modelos

Nesta fase avaliaram-se as variáveis que podem ser utilizadas no modelo. Foram consideradas apenas variáveis relacionadas com o jogo e comparações entre os jogadores (como "AgeDiff", "HeightDiff"), excluindo-se variáveis individuais como "WHeight" ou "LHeight". Isto porque, ao usar variáveis separadas, o modelo não capta a relação ou diferença entre as duas. Esta lógica aplica-se a todas as variáveis individuais dos jogadores, que foram transformadas em variáveis de comparação.

3.4.1. Correlação de Pearson

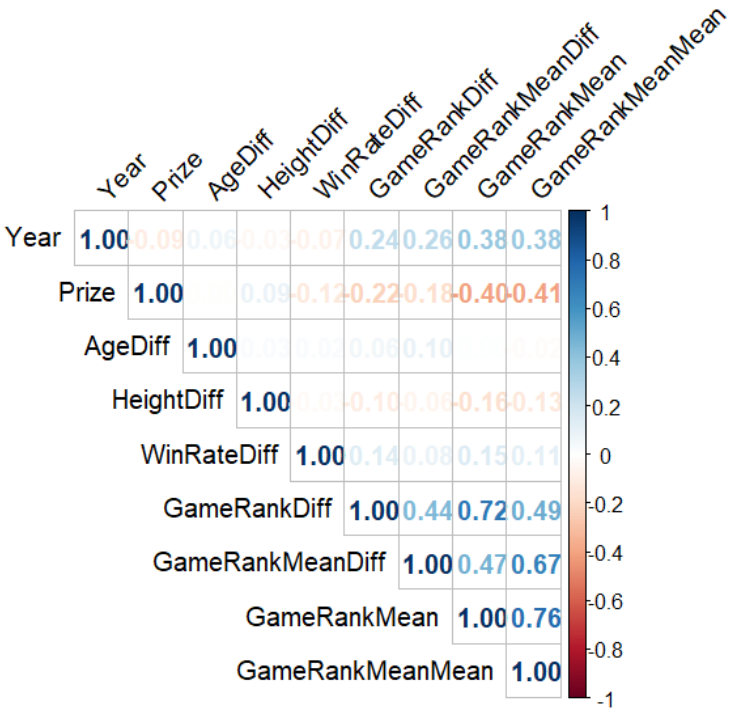

Figura 3 - Matriz de Correlação

As variáveis "GameRankMeanMean" e "GameRankMean" obtiveram a maior correlação, com o valor de 0,76. O segundo maior valor de correlação foi entre as variáveis "GameRankMean" e "GameRankDiff", com 0,72. Ambos "GameRankMean" e "GameRankMeanMean" possuem uma correlação de 0,38 com a variável "Year". Basicamente, todas essas variáveis oscilam no mesmo sentido, só que em proporções diferentes.

As correlações moderadas negativas entre "GameRankMean" e "GameRankMeanMean" com a variável "Prize" são de - 0,40 e - 0,41, o que significa que quanto maior a média entre os *rankings* dos jogadores, menor é o prémio do torneio.

As quatro variáveis derivadas do *ranking* dos jogadores, a diferença de *ranking* entre os jogadores ("GameRankDiff"); a média dos *rankings* de ambos jogadores ("GameRankMean"); a diferença da média dos *rankings* dos 3 anos anteriores entre os jogadores ("GameRankMeanDiff") e a média da média dos *rankings* dos 3 anos anteriores de ambos jogadores ("GameRankMeanMean"); não devem ser utilizadas juntas em modelos lineares pela forte correlação entre si, podendo atrapalhar a performance dos modelos. Essa correlação já era esperada, tendo em conta que as quatro provêm do mesmo par de variáveis, "WGameRank" e "LGameRank".

As restantes correlações são consideradas fracas, apresentando valores abaixo de 0,3, em absoluto. Essa baixa correlação entre variáveis preditoras é importante para a criação dos modelos, garantindo a hipótese de não haver problemas de multicolinearidade.

3.4.2. Associação entre as variáveis numéricas e Sets (Eta Quadrado)

	Variável	Eta_Squared
5	GameRankMean	0.057939460
4	GameRankDiff	0.057837453
9	GameRankMeanMean	0.046055661
7	WinRateDiff	0.031525605
8	GameRankMeanDiff	0.027009049
3	HeightDiff	0.023584786
1	Year	0.018012316
6	Prize	0.017658142
2	AgeDiff	0.002644906

Figura 4 - *Eta Squared* por variável

Foi feita a associação entre as variáveis numéricas e a variável alvo "Sets". Entre as 14 associações calculadas, somente 2 obtiveram valores superiores a 0,05, com as variáveis "GameRankMean" e "GameRankDiff". Essas variáveis são as únicas que apresentaram algum impacto na variância de "Sets", mesmo que mínimo.

Outras variáveis, como "HeightDiff", obtiveram valores levemente abaixo de 0,05, mas a maioria não atingiu mais que 0,03, mostrando-se irrelevantes para explicar a variância da variável dependente. Os valores apresentados foram transformados para a mesma escala de medida da correlação de Pearson, através da raiz quadrada, para facilitar a sua interpretação.

3.4.3. Associação entre as variáveis categóricas e Sets (V de Cramér)

	Variável	Cramers_V
4	TournamentGroup	0.04471430
5	GameRoundGroup	0.02895641
3	CountryDiff	0.02636559
1	Ground	0.02473525
2	HandDiff	0.01469874

A medida de associação V de Cramér foi utilizada entre as variáveis categóricas e a variável dependente "Sets". Os valores encontrados estão dispostos na tabela ao lado.

Figura 5 - Associação entre Variáveis Categóricas (V de Cramér)

Um ponto importante é que apenas uma variável apresenta valores acima de 0,04 (considerado fraco), "TournamentGroup", enquanto os restantes valores são considerados muito fracos.

*Valores indicativos para a correlação, associação por Eta Quadrado e V de Cramér foram retirados da Universidade de Cambridge [16].

3.4.4. Gráficos entre as Variáveis e Sets

Variável Ground

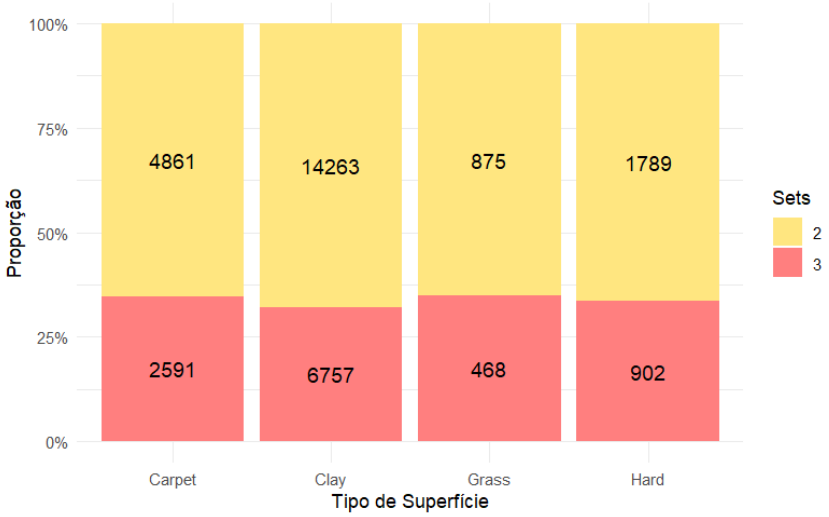

Figura 6 - Número de Sets em função do tipo de superfície

Observa-se uma variação mínima na proporção de jogos que terminam em 2 ou 3 sets, consoante a superfície. Por exemplo, em superfícies de piso duro (*Hard*), há uma tendência a jogos mais curtos (2 sets). A superfície onde ocorrem mais jogos de 3 sets é o *Clay*, ainda que a maioria dos jogos em terra batida termine em 2 sets. Em todas as superfícies há uma tendência maior para os jogos de 2 sets. Este padrão sugere que a superfície não tem influência no número de sets.

Variável HeightDiff

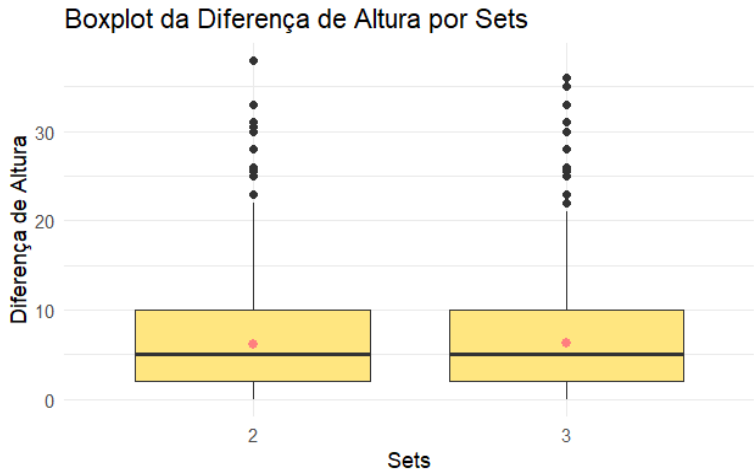

Figura 7 - *Boxplots* da diferença de alturas de jogadores por número de Sets

Aqui é possível ver que a diferença de altura não tem um efeito notável no número de sets. Apesar da existência de alguns *outliers*, em ambos os *boxplots*, os jogadores têm mediana de 5 cm de diferença e intervalos interquartil semelhantes.

Variável AgeDiff

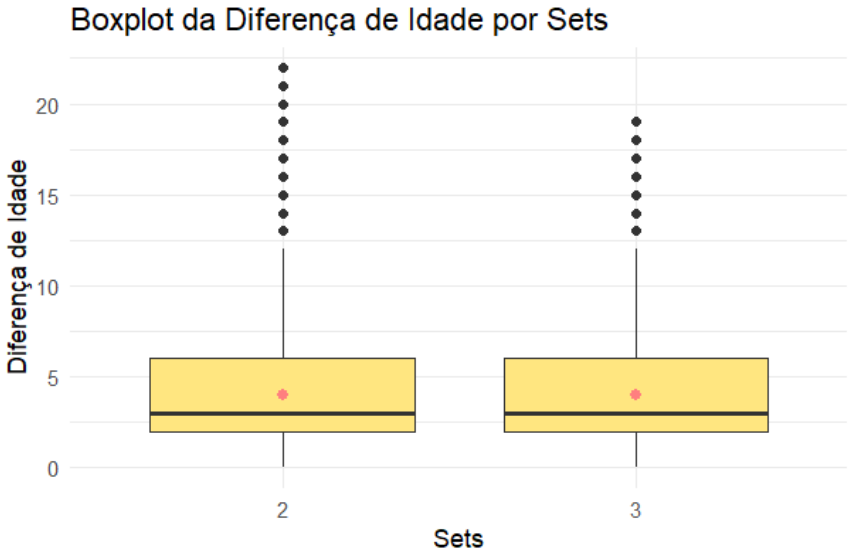

Figura 8 - *Boxplots* da diferença de idades dos jogadores por número de Sets

Como é possível ver no *boxplot* acima, a distribuição da diferença de idades entre os jogadores é bastante semelhante nos jogos que terminam em 2 e 3 sets. A mediana é ligeiramente inferior nas partidas com 2 sets, mas a dispersão dos dados, incluindo a presença de *outliers*, mantém-se consistente em ambos os *boxplots*. Mesmo verificando-se uma ligeira tendência para encontros decididos em 3 sets quando os jogadores têm idades mais próximas, esta variável não demonstra ter uma influência significante na previsão do número de sets num jogo.

Variável CountryDiff

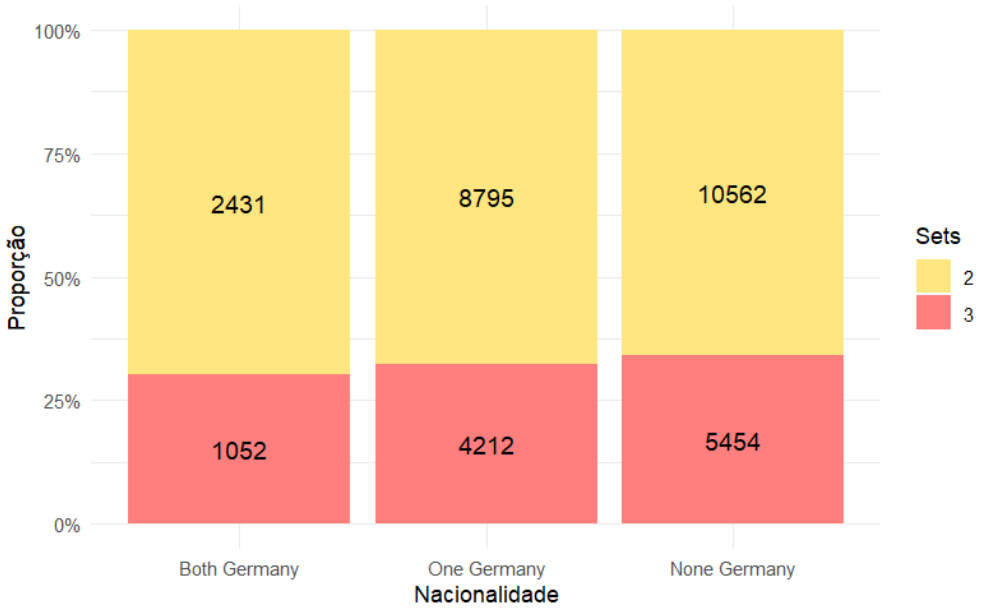

Figura 9 - Diferenças de nacionalidades entre os jogadores em função da proporção e número de Sets

Jogos em que ambos os jogadores têm nacionalidade alemã tendem a terminar em 2 sets com mais frequência, enquanto jogos sem qualquer jogador alemão apresentam uma maior proporção de jogos decididos em 3 sets. Esta tendência poderá refletir uma maior familiaridade dos jogadores alemães com as condições locais (ex.: o apoio do público, o ambiente, etc.), o que favorece esses jogadores. Assim, a presença de jogadores com nacionalidade alemã parece estar associada a jogos mais curtos, o que indica que o fator "jogar em casa" pode ter impacto no número de sets.

Variável GameRound

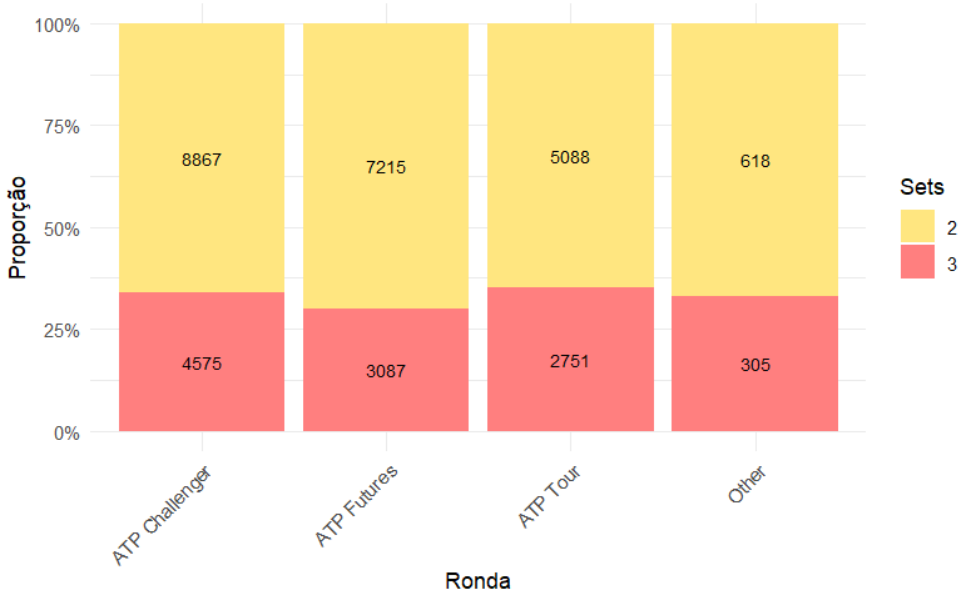

Figura 10 - Tipos de torneio em função do número/proporção de jogos de 2 ou 3 sets

Nos torneios *ATP Futures*, que normalmente envolvem jogadores com *ranking* mais baixos e em fase inicial da carreira, observa-se uma menor proporção de jogos decididos a 3 sets. Isto indica uma maior diferença de qualidade entre os adversários, resultando em vitórias mais rápidas e com menor equilíbrio. Por outro lado, os torneios *ATP Challenger* e *ATP Tour*, verificam uma maior percentagem de jogos a 3 sets, o que indica jogos mais equilibrados entre os adversários. Nestes casos, a semelhança entre o *ranking*, a capacidade técnica e a competitividade dos jogadores levam a jogos mais equilibrados, exigindo frequentemente um terceiro set. Nos torneios "Other" (que incluem a *World Team Cup*, a *Davis Cup*, etc.) é também frequente que os jogos sejam decididos em 3 sets. Estes torneios, apresentam formatos diferentes, envolvem a competição entre seleções nacionais e participam os melhores jogadores, resultando em jogos mais equilibrados.

Variável HandDiff

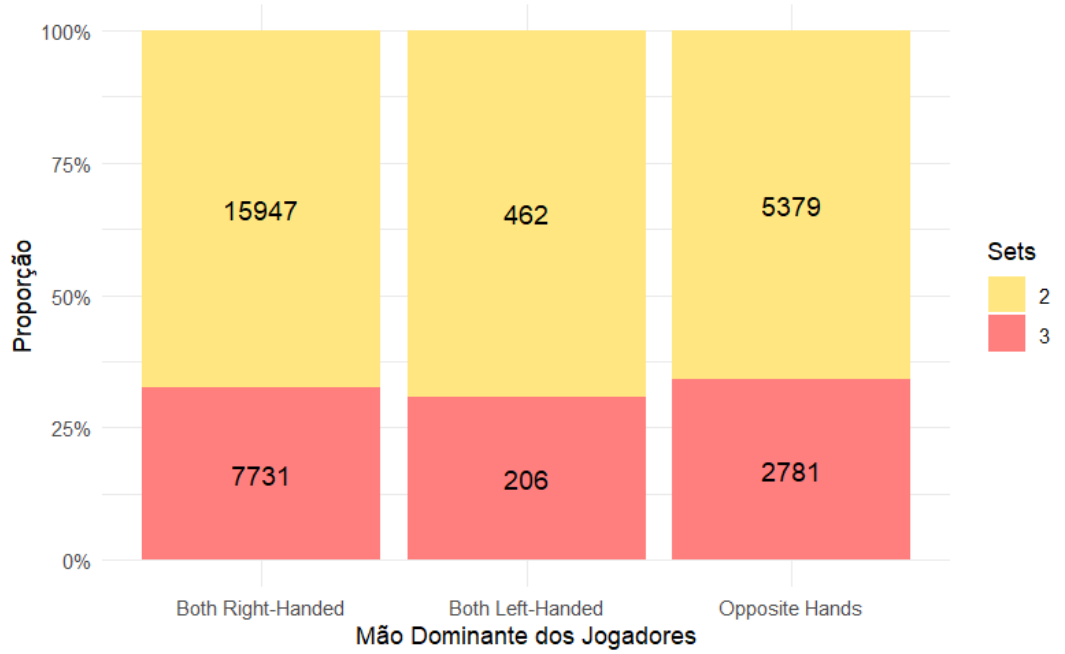

Figura 11 - Diferenças na mão de preferência em função da proporção e número de sets

Os jogadores com mãos dominantes opostas têm maior probabilidade de acabar o jogo em 3 sets. Isto é possivelmente devido ao facto de ser mais fácil um jogador canhoto dominar, pois o oponente destro em princípio não estará habituado a jogar contra canhotos.

Variável Prize

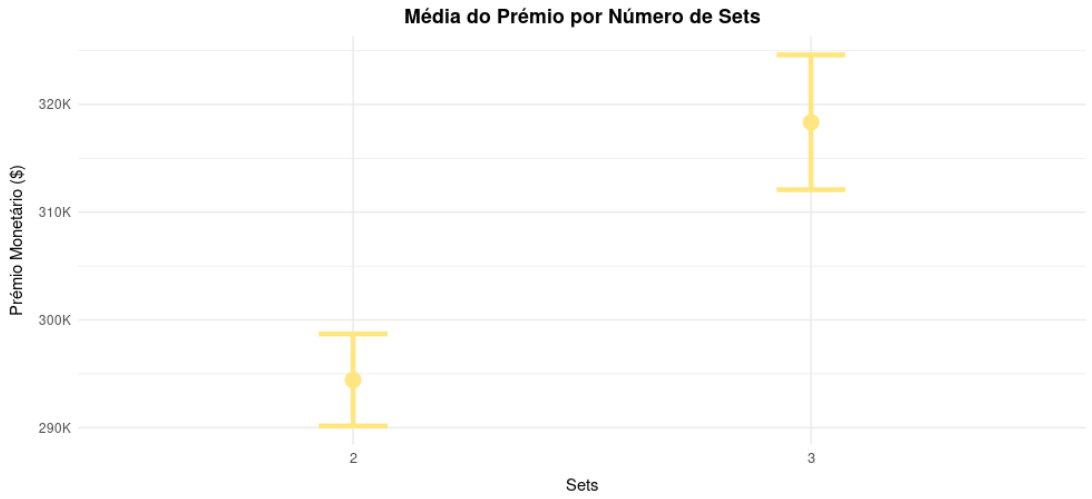

Figura 12 - Estatísticas do prémio (média e erro padrão) divididos por número de sets.

O valor do prémio afeta notavelmente o número de sets, sendo os jogos com 3 sets mais comuns quando o prémio é maior, sugerindo que torneios de nível mais alto podem estar melhor equilibrados do que aqueles com prémios mais modestos.

4. Modeling

Nesta fase, são aplicados algoritmos de *Machine Learning* aos dados já limpos e tratados para construir modelos preditivos. O objetivo dos modelos é prever o número de sets em que a partida de ténis vai acabar, com base nas características do jogo e dos jogadores.

Este é um problema de classificação binária, no qual cada partida de ténis é categorizada em duas classes possíveis: classe "2" (jogos que terminam em 2 sets) ou classe "3" (jogos que terminam em 3 sets).

4.1. Variáveis Para os Modelos

Nos modelos, foram utilizadas apenas variáveis relacionadas ao jogo em si e às comparações entre as características dos jogadores em cada partida. As variáveis originais específicas de cada jogador individualmente não foram incluídas.

No caso das variáveis numéricas, trabalhou-se com valores absolutos, enquanto as variáveis categóricas foram construídas de forma neutra, sem qualquer influência do desfecho do jogo. Esta abordagem assegurou que o modelo se baseasse apenas em padrões objetivos do jogo e nas comparações entre os tenistas, sem utilizar informações relacionadas com resultados conhecidos.

Foram então selecionadas as 8 variáveis a serem utilizadas nos modelos:

* As variáveis são a comparação entre os jogadores.
** As variáveis são sobre o torneio/jogo específico.

Variáveis Numéricas:

- ➔ GameRankDiff *
- ➔ GameRankMeanMean *
- ➔ Prize **
- ➔ WinRateDiff *

Variáveis Categóricas:

- ➔ CountryDiff *
- ➔ HandDiff *
- ➔ GameRoundGroup **
- ➔ TournamentGroup **

As variáveis categóricas acima foram transformadas em variáveis *dummies* através do método de *one-hot encoding* com remoção de uma categoria base (também conhecido como *k - 1 encoding*).

Por exemplo, a variável "CountryDiff", que originalmente continha três categorias, "Both Germany", "One Germany", e "None Germany", foi convertida em duas variáveis binárias:

One Germany:

- 1, quando apenas um dos jogadores é alemão.
- 0, caso contrário.

None Germany:

- 1, quando nenhum dos jogadores é alemão.
- 0, caso contrário.

Quando ambas as variáveis (*One Germany* e *None Germany*) assumem o valor 0, significa que ambos os jogadores são alemães (*Both Germany*, que serve como categoria de referência).

O mesmo processo foi aplicado às restantes variáveis categóricas, de forma a garantir que todas as categorias fossem representadas por variáveis numéricas, exceto a categoria base, que é implicitamente interpretada quando todas as *dummies* associadas são zero.

Este procedimento é essencial porque muitos modelos de *Machine Learning* e regressão exigem que as variáveis de entrada sejam numéricas. Além disso, ao remover uma das categorias (a base), evita-se a multicolinearidade entre as variáveis *dummies*.

4.2. Oversampling

O *Oversampling* é útil neste caso porque os dados estão desequilibrados: a classe "2" tem cerca do dobro de observações (21.788) em comparação com a classe "3" (10.718). Quando uma classe é muito mais frequente que outra, os modelos tendem a aprender melhor os padrões da classe dominante, prejudicando a precisão nas previsões da classe menos representada. O *Oversampling* ajuda a resolver esse problema ao equilibrar a distribuição das classes, e assim, o modelo consegue aprender padrões mais equilibrados, melhorando o seu desempenho geral, especialmente para a classe com menos dados.

Para os modelos, foi utilizada a técnica **SMOTE** (*Synthetic Minority Oversampling Technique*). Para equilibrar a base de dados, esta técnica usa observações sintéticas.

Observações sintéticas são dados artificiais gerados pelo **SMOTE**, que em vez de simplesmente copiar amostras da classe minoritária, cria novas observações interpolando características entre pontos reais vizinhos, preservando a distribuição original dos dados. De forma mais simples, o **SMOTE** pega num exemplo real da classe minoritária (neste caso, uma partida de tênis específica da classe "3"), encontra outra partida semelhante a ela e depois mistura as características das duas para criar uma nova partida sintética. Isto permite que o modelo aprenda padrões mais diversificados da classe minoritária sem recorrer à duplicação pura de dados.

O **SMOTE** foi aplicado de duas formas diferentes:

1º Opção - Balanceamento Total

- Igualou o número de observações em ambas as classes, ou seja, eliminou observações da classe "2" e criou observações sintéticas para a classe "3"
- Resultado: 16 253 observações para cada classe ("2" e "3").
- Elimina completamente o viés a favor da classe maioritária e é ideal quando precisamos de igual precisão em ambas as classes

2º Opção - Balanceamento Parcial

- Aumentou apenas a classe minoritária ("3"), duplicando a quantidade de observações através da criação de observações sintéticas.
- Resultado: 21 436 observações para "3" e mantém as 21 788 de "2".
- Preservou os dados originais da classe maioritária, logo, é útil quando se quer manter a informação original da classe dominante.

As principais diferenças dos dois métodos utilizados são:

- O Balanceamento Total cria igualdade perfeita, mas descarta dados da classe "2".
- O Balanceamento Parcial mantém mais dados originais da classe "2", enquanto melhora a representação da classe "3".

4.3. Cross-Validation

Para avaliar de forma rigorosa o desempenho dos modelos, optou-se por utilizar validação cruzada com 10 *folds* (*10-fold cross-validation*). Esta técnica é fundamental para garantir que os resultados obtidos não estão dependentes de uma única divisão aleatória entre treino e teste, mas sim que refletem a verdadeira capacidade de generalização do modelo. O processo consiste em dividir o conjunto de dados completo em 10 subconjuntos aproximadamente iguais. Em cada uma das dez iterações, o modelo é treinado com 9 desses subconjuntos e testado com o subconjunto restante. Este processo é repetido até que cada subconjunto tenha servido exatamente uma vez como conjunto de teste, ou seja, 10 vezes.

A principal vantagem deste método é que todos os dados são usados tanto para treino como para teste, embora em momentos diferentes. Isto maximiza a utilização da informação disponível. Além disso, a validação cruzada reduz a variância na estimativa de desempenho do modelo, apresentando uma média mais estável e representativa dos resultados.

4.4. Divisão de Treino e Teste

Durante a validação cruzada, em cada *fold* é realizada uma divisão explícita entre dados de treino e dados de teste. Esta separação serve para simular a aplicação real do modelo, que será

usado para prever o número de sets de jogos futuros a partir de dados nunca vistos. Assim, em cada uma das 10 iterações, foram utilizados 90% dos dados para treinar o modelo e reservados os 10% restantes para testá-lo.

4.5. Modelos

Nesta etapa foram criados 6 modelos diferentes de previsão, utilizando diferentes métodos de *Oversampling*.

1. Modelo K-Nearest Neighbors

Para o modelo *K-Nearest Neighbors* (**KNN**) foi necessário fazer a normalização dos dados com a função *scale*, porque o algoritmo calcula a distância euclidiana entre observações e é sensível à escala das variáveis. Variáveis com valores maiores influenciam mais a distância total, o que pode distorcer os resultados. Por isso, os dados foram transformados para que cada variável tivesse média 0 e desvio padrão 1, com base no conjunto de treino, evitando a perda de informação.

Após isso, foram testados vários valores de k. O valor de k escolhido foi 5, pois foi o que teve maior acurácia média.

Método de Oversampling: 2ª Opção - Balanceamento Parcial

Nº de Observações da Classe 2: 21 788 **Nº de Observações da Classe 3:** 21 436

Interpretação dos Resultados:

Matriz de confusão

Previsto/Referência	2	3
2	15326	9406
3	6462	12030

Acurácia - 63,29%, o modelo acerta 63% das previsões, (**NIR** = 50,41%).
**No information rate* (Prever sempre a classe maioritária)

Sensibilidade - 70,34%, o modelo identifica corretamente a classe 2 em 7 de cada 10 casos.

Especificidade - 56,12%, o modelo acerta pouco mais da metade das vezes ao identificar a classe 3.

Precisão - Quando prevê a classe 2 o modelo está certo em 61,97% dos casos, e quando prevê a classe 3 o modelo está certo em 65,06% dos casos.

2. Modelo Random Forest (XGBoost)

Para o modelo *Random Forest* (*XGBoost*), os dados foram convertidos em matrizes numéricas e em objetos *xgb.DMatrix*, formato exigido pelo algoritmo. Os modelos foram treinados com os hiperparâmetros *max.depth=6* (que controla a profundidade máxima das árvores) e *nround=10* (número de iterações/árvores). A escolha de *max.depth* visa evitar o *overfitting*, mantendo as árvores suficientemente complexas para captar padrões relevantes, mas não tão profundas que memorizem o treino. O número de iterações (*nround*) relativamente baixo foi usado para reduzir o custo computacional durante a validação cruzada.

Para equilibrar as duas classes, ao invés de usar o *Oversampling*, foi utilizado um peso com o hiperparâmetro *scale_pos_weight* com o valor [0,9 * (Nº de observações "2" / Nº de observações "3")], idealmente o valor seria o somente o rácio, mas experimentos feitos com outros pesos revelaram resultados mais satisfatórios.

O modelo binário funciona prevendo a probabilidade da classe "3" e a probabilidade contrária é a da classe "2".

Método de Oversampling: Sem *Oversampling*

Nº de Observações da Classe 2: 21 788 **Nº de Observações da Classe 3:** 10 718

Interpretação dos Resultados:

Matriz de confusão

Previsto/Referência	2	3
2	14771	6803
3	7017	3915

Acurácia - 57,48%, proporção total de acertos. A acurácia esperada se o modelo sempre previsse a classe mais comum (classe "2"), seria **NIR** = 67,03. Logo, o modelo é pior que um modelo que prevê sempre a classe dominante.

Sensibilidade - 67,79%, o modelo é relativamente bom ao detetar a classe 2.

Especificidade - 36,53%, o modelo tem dificuldade em identificar bem a classe 3.

Precisão - quando o modelo prevê a classe 2 acerta 68,47% das vezes, mas quando prevê a classe "3", apenas acerta 35,81% das vezes.

3. Modelo K-Nearest Neighbors

Para esse modelo **KNN**, as mesmas mudanças do modelo 1 foram feitas. A única diferença entre eles é o método de *Oversampling*. Também como antes, foram testados vários valores de k, sendo escolhido novamente o 5.

Método de Oversampling: 2ª Opção - Balanceamento Total

Nº de Observações da Classe 2: 16 253 **Nº de Observações da Classe 3:** 16 253

Interpretação dos Resultados:

Matriz de confusão

Previsto/Referência	2	3
2	10307	7629
3	5946	8624

Acurácia - 58,24%, o modelo acerta ligeiramente mais do que o acaso (**NIR***=50%), o que indica um desempenho modesto.
* No information rate

Sensibilidade - 63,42%, a capacidade do modelo para identificar corretamente os casos da classe 2 é razoável.

Especificidade - 53,06%, o modelo tem mais dificuldade em identificar corretamente os casos da classe 3 (casos negativos), isto pode levar a falsos positivos, ou seja, o modelo acaba por prever como classe 2, observações que realmente pertencem à classe 3.

Precisão - A precisão do modelo a prever a classe 2 é 57,47%, isto é, entre todas as previsões feitas como classe 2, apenas 57,47% estavam corretas. A precisão da classe 3 é 59,19%, logo o modelo é mais preciso para prever a classe 3.

4. Modelo Random Forest (XGBoost)

Para esse modelo *Random Forest*, os dados também foram transformados no formato exigido pelo *XGBoost*. Diferente do modelo 2, não possui pesos, porque utiliza o método de *Oversampling* com balanceamento total.

Método de Oversampling: 2ª Opção - Balanceamento Total

Nº de Observações da Classe 2: 16 253 **Nº de Observações da Classe 3:** 16 253

Interpretação dos Resultados:

Matriz de confusão

Previsto/Referência	2	3
2	15241	11028
3	1012	5225

Acurácia - 62,96%, modestamente acima do **NIR** (**NIR**=50%), o que indica um desempenho relativamente bom nesta métrica.

Sensibilidade - 93,77%, este modelo tem a maior sensibilidade, logo, tem uma baixa tendência a prever uma observação realmente da classe "2", como classe "3".

Especificidade - 32,15%, este modelo tem a maior dificuldade, entre todos os modelos testados, de identificar observações da classe "3". Este facto, combinado com a alta sensibilidade, pode significar que o modelo tem uma tendência de prever a classe "2" em detrimento da 3. Isto põe em questão a utilidade da sua acurácia relativamente alta.

Precisão - Quando a previsão do modelo foi a classe "2" ele acertou 58,02%, das vezes. A precisão da classe "3" é 83,77%. Isto significa que, das observações previstas como sendo da classe "3", mais de 4 em 5 estão corretas.

5. Modelo Combinado (K-NN e XGBoost)

O **KNN** classifica cada observação e devolve a probabilidade da classe predita. O *XGBoost* (modelo binário) dá a probabilidade de a observação ser da classe "3".

Faz-se a média entre a probabilidade do **KNN** para a classe 3, e a probabilidade do *XGBoost*. Se a média for ≥ 0,5, a classe final é "3"; caso contrário, é "2".

Método de Oversampling: 1ª Opção - Balanceamento Total

Nº de Observações da Classe 2: 16 253 **Nº de Observações da Classe 3:** 16 253

Interpretação dos Resultados:

Matriz de confusão

Previsto/Referência	2	3
2	11096	7329
3	5157	8924

Acurácia - 61,59%, o modelo é melhor que um modelo que prevê sempre a classe dominante (NIR = 50%).

Sensibilidade - 68, 27%, boa capacidade em identificar a classe "2".

Especificidade - 54,91%, o modelo acerta pouco mais da metade das vezes ao identificar a classe "3".

Precisão - Quando o modelo prevê a classe "2", ele está certo em ~ 60% dos casos, quando prevê a classe "3" está certo em 63,38% dos casos.

6. Modelo de Regressão Logística

Para o modelo de Regressão Logística, foi necessário criar um *data frame* com os valores já normalizados das variáveis. O modelo é o mais propenso a ter problemas com a multicolinearidade das variáveis preditoras, então, foi utilizado a métrica *Variance Inflation Factor* ou **VIF** para perceber se há multicolinearidade ou não, e, apesar de não reduzir o poder preditivo do modelo, pode atrapalhar na estimativa dos coeficientes da equação criada. Valores acima de 5 indicam que as variáveis não são independentes, o que não acontece no modelo, porque apresenta todos os valores abaixo de 4.

Método de Oversampling: 1ª Opção - Balanceamento Total

Nº de Observações da Classe 2: 16 253 **Nº de Observações da Classe 3:** 16 253

Interpretação dos Resultados:

Matriz de confusão

Previsto/Referência	2	3
2	7602	6395
3	8651	9858

Acurácia - 53,71%, o modelo tem um desempenho ligeiramente melhor do que um modelo *naive* (**NIR** = 50%), mas muito limitado em termos de capacidade preditiva.

Sensibilidade - 46,77%, o modelo falha em identificar corretamente mais da metade dos casos da classe "2".

Especificidade - 60,65%, um pouco melhor a identificar corretamente os casos de classe "3".

Precisão - 54,31%, apenas cerca de metade das previsões da classe “2” estão corretas, o que indica uma taxa significativa de falsos positivos. O modelo mostrou-se ainda pior ao prever a classe “3”, com uma precisão de 53,26%.

5. Evaluation

Comparação dos Modelos

Modelo	Acurácia	Precisão	Sensibilidade	Especificidade	Nº*
Modelo 1	0.6329	0.6197	0.7034	0.5612	**1**
Modelo 2	0.5748	0.6847	0.6779	0.3653	**4**
Modelo 3	0.5824	0.5747	0.6342	0.5306	**5**
Modelo 4	0.6296	0.5802	0.9377	0.3215	**3**
Modelo 5	0.6159	0.6022	0.6827	0.5491	**2**
Modelo 6	0.5371	0.5431	0.4677	0.6065	**6**

*Ordem do melhor para o pior modelo, calculada através da média do ranking do desempenho das métricas.

Quando se analisa a acurácia dos modelos desenvolvidos, nota-se que o melhor desempenho foi alcançado pelo modelo 1 (*K-Nearest Neighbors* com balanceamento parcial), atingindo uma taxa de acerto de cerca de 63,3%. Muito próximo está, também, o modelo 4 (*Random Forest* com balanceamento total) com 62,96%, e ainda o modelo 5 (Combinado entre **KNN** e *XGBoost*) que regista 61,59%. Estes três apresentam uma atuação acima dos restantes.

No que respeita à sensibilidade, que avalia a capacidade dos modelos em identificar corretamente os casos da classe “2”, o destaque vai para o modelo 4 (*Random Forest* com balanceamento total), que se mostra altamente sensível com uma taxa superior a 93%, revelando uma notável eficácia na deteção de positivos. Também o modelo 1 (**KNN** com balanceamento parcial) e o modelo 5 (Combinado) apresentam valores robustos, situando-se entre os 68% e 70%. Já o modelo 6, de Regressão Logística, revela dificuldades significativas nesta métrica, identificando corretamente menos da metade dos casos da classe “2”.

Quanto à especificidade, que mede a capacidade de detetar corretamente os casos da classe “3”, o cenário inverte-se. O modelo 6, de Regressão Logística, lidera neste aspeto, com mais de 60% de especificidade, sendo o único que se aproxima de um desempenho realmente satisfatório nesta métrica. Os modelos 5 e 1, Combinado e o **KNN** com balanceamento parcial, também mantêm valores aceitáveis, pouco acima dos 54% e 56%, respetivamente. Em

contrapartida, o modelo 4, que se destacou na sensibilidade, revela-se extremamente fraco na identificação da classe "3", com uma especificidade muito baixa, em torno dos 32%.

No que respeita à precisão na previsão da classe "2", verifica-se que o modelo 2 (*XGBoost* sem *Oversampling*) apresenta o valor mais elevado, com aproximadamente 68%, demonstrando uma boa capacidade de acerto sempre que prevê essa classe. Logo a seguir surge o **KNN** com balanceamento parcial (1), com cerca de 62%, e o modelo Combinado, que também mantém uma precisão razoável, na ordem dos 60%. Já o modelo 6 (Regressão Logística) revela um desempenho inferior neste parâmetro, com uma precisão de apenas 54%, o que indica uma elevada taxa de falsos positivos ao classificar como "2" observações que não o são.

Em suma, observa-se uma tensão constante entre sensibilidade e especificidade: modelos altamente sensíveis sacrificam a capacidade de detetar negativos, e vice-versa. O **KNN** com balanceamento parcial e o modelo Combinado revelam-se os mais equilibrados nas quatro métricas, enquanto os extremos, como o *Random Forest* com sensibilidade quase total ou a regressão com especificidade elevada, demonstram que otimizar uma métrica em excesso pode comprometer seriamente outras dimensões críticas da classificação.

Foi feito um *ranking* de melhor para pior em cada métrica, e depois a média dos *rankings* que cada modelo obteve. O Modelo 1, **KNN** com balanceamento parcial, teve a melhor média, por isso foi classificado como o melhor modelo a que se conseguiu chegar.

6. Deployment

O modelo de previsão do número de sets pode ser uma ferramenta valiosa para equipas técnicas e treinadores, uma vez que permite antecipar a duração e o desgaste físico esperado em cada partida. Ao estimar quantos sets um jogo poderá ter, os treinadores podem planear estratégias mais eficazes, gerir melhor a energia dos atletas e otimizar os tempos de recuperação pós-jogo. Além disso, a previsão ajuda a identificar encontros potencialmente equilibrados ou particularmente desgastantes, permitindo ajustes táticos e físicos com maior precisão.

Este modelo de previsão do número de sets também oferece uma vantagem competitiva às casas de apostas, especialmente no mercado *Over/Under* Sets, ao gerar estimativas mais precisas para o ajuste automático das *odds*. Ao incorporar variáveis de comparação dos jogadores e contexto do torneio/partida, o modelo permite uma definição de probabilidades mais dinâmica e informada, reduzindo margens de erro.

Além disso, a disponibilização das previsões através de uma **API** *(Application Programming Interface)* em tempo real facilita a integração em plataformas de apostas, permitindo que os utilizadores acedam a estatísticas avançadas e *insights* valiosos para decisões mais fundamentadas. Para a operadora, esta ferramenta não só melhora a atratividade das *odds* como auxilia na gestão de risco, identificando eventuais desequilíbrios no mercado e permitindo ações proativas para mitigar perdas.

No entanto, para que estas aplicações fossem viáveis, na prática, seria necessário que o modelo apresentasse um desempenho significativamente melhor. Apesar das diferentes abordagens testadas, os resultados obtidos revelaram níveis de acurácia, precisão e especificidade ainda insuficientes para uma aplicação real. Assim, embora o conceito tenha potencial, o modelo atual carece de melhorias antes de poder ser utilizado com confiança em contextos profissionais, como o planeamento desportivo ou mercados de apostas.

Conclusão

Este projeto teve como propósito explorar o potencial da ciência de dados na previsão do número de sets em partidas de ténis realizadas na Alemanha. Com base numa base de dados extensa e complexa, foi possível aplicar técnicas de tratamento, análise e *Machine Learning*, recorrendo a ferramentas como **R**, **Python** e **MongoDB**.

Através de uma abordagem estruturada e fundamentada no modelo **CRISP-DM**, foram testadas diferentes estratégias preditivas, destacando-se o modelo *K-Nearest Neighbors* com balanceamento parcial pelos seus resultados mais consistentes. Apesar de limitações no desempenho geral, os resultados obtidos evidenciam a relevância de soluções *data-driven* no contexto desportivo, com potencial de aplicação futura em áreas como planeamento estratégico, análise tática e mercados de apostas. Este trabalho reforça, assim, o valor da ciência de dados como instrumento de apoio à tomada de decisão em ambientes competitivos.

Bibliografia

[1] Simplemaps. (2025). *World Cities Database*. https://simplemaps.com/data/world-cities

[2] ATP Tour. (n.d.). *atptour.com*. https://www.atptour.com

[3] Tennis Explorer. (n.d.). *tennisexplorer.com*. https://www.tennisexplorer.com

[4] International Tennis Federation. (n.d.). *itftennis.com*. https://www.itftennis.com

[5] Tennis Belge. (n.d.). *tennis-belge.be*. http://www.tennis-belge.be

[6] TennisLive.net. (2025). *ATP – Resultados ao vivo, estatísticas e rankings*. https://www.tennislive.net/

[7] Steve G Tennis. (2025). *Previsões de ténis, estatísticas e análises H2H*. https://www.stevegtennis.com/

[8] Sackmann, J. (n.d.). *Jeff Sackmann - Tennis Data Projects*. https://www.jeffsackmann.com/

[9] Sackmann, J. (n.d.). *Jeff Sackmann on GitHub*. https://github.com/JeffSackmann

[10] Tennis Abstract. (n.d.). *tennisabstract.com*. https://www.tennisabstract.com/

[11] Banco de Portugal. (n.d.). *Quadros estatísticos – BPstat*. https://bpstat.bportugal.pt/conteudos/quadros/2033?utm_source

[12] USA Inflation Calculator. (n.d.). *usinflationcalculator.com*. https://www.usinflationcalculator.com/

[13] ATP Tour. (2025). *ATP Official Rulebook 2025*. https://www.atptour.com/en/corporate/rulebook

[14] TennisUpToDate. (2025). *BMW Munich Open 2025*. https://tennisuptodate.com/bmw-munich-open

[15] Pyle, D. (1999). *Data preparation for data mining*. Morgan Kaufmann Publishers. https://www.temida.si/~bojan/MPS/materials/Data_preparation_for_data_mining.pdf

[16] MRC Cognition and Brain Sciences Unit. (2025). *Regras gerais sobre magnitudes de tamanhos de efeito*. https://imaging.mrc-cbu.cam.ac.uk/statswiki/FAQ/effectSize