Pontifícia Universidade Católica do Rio de Janeiro

Pós-graduação em Ciência de Dados e Analytics

Diogo Oliveira Paiva Mattos

Pipeline de Dados Utilizando DataBricks para Análise de Resultados do Futebol Europeu e Apostas na Temporada de 2023-24

MVP DA DISCIPLINA DE ENGENHARIA DE DADOS

Rio de Janeiro

Julho de 2024

Índice

Objetivo

O mercado de apostas esportivas tem crescido consideravelmente. Segundo dados da Anbima (Associação Brasileira das Entidades dos Mercados Financeiro e de Capitais), 14% dos brasileiros fizeram ao menos uma aposta esportiva no ano de 2023. Este número supera com folga o de investidores de títulos privados, fundos de investimento e títulos públicos (respectivamente, por 5%, 4% e 2% dos brasileiros).

Dentro deste mercado, a categoria que mais movimenta recursos e apostadores é a de futebol.

A proposta deste trabalho é utilizar uma base de dados com dados de todas as partidas das cinco principais ligas de futebol europeias (alemã, espanhola, francesa, inglesa e italiana), na temporada de 2023-24. Além dos placares de cada partida, esta base também contém dados da partida, como chutes a gol e escanteios.

A ideia seria cruzar estes dados com as cotações ("*odds*") que as principais empresas de apostas esportivas do mercado apontavam antes do início de cada partida.

A princípio, iremos nos concentrar apenas nas apostas de probabilidade do resultado final de cada partida, mas podemos tentar explorar também outras modalidades no desenvolvimento deste trabalho.

Algumas perguntas que este trabalho se propõe a responder:

- Qual o índice de acerto das previsões das casas de aposta para o resultado da partida?

- Alguma liga segue mais o padrão destas previsões? Alguma está mais propensa a "zebras"?

- Em algum período da temporada, as probabilidades de apostas se confirmam mais ou menos do que em outros períodos?

- Há alguma distorção entre as probabilidades que cada empresa trabalha antes das partidas, ou elas seguem o mesmo padrão?

Plataforma

Para a elaboração deste MVP, seguimos a orientação dos professores da disciplina e utilizamos o Databricks Community Edition. Chegamos a prospectar a utilização de ferramentas AWS S3, como o Glue Studio e o Redshift, mas desistimos após o feedback obtido nos encontros para tirar dúvidas. Para criar, tratar a base e fazer as consultas, foram utilizadas as linguagens PySpark e SQL.

Os datasets e notebooks criados estão disponibilizados no GitHUb: https://github.com/diogomattos1/mvp-engenharia-dados e o dataset está publicado no Databricks: https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/955592756983036/4174348929971334/103274989165388/latest.html

Detalhamento

Busca pelos dados

Para os dados deste problema, foi utilizada um dataset obtido no site Kaggle (https://www.kaggle.com/datasets/audreyhengruizhang/european-soccer-data), de autoria de Audrey Hengrui Zhang. O autor utilizou o site https://www.football-data.co.uk/ para obter os dados que compõem este dataset.

Este dataset possui todos os resultados das 5 principais ligas de futebol da Europa (Alemanha, Espanha, França, Inglaterra e Itália).

Coleta

Inicialmente não será necessária uma atividade de coleta. Utilizaremos apenas os dados do dataset disponível no Kaggle, pois julgamos que ele apresenta bastante informação em suas 104 colunas para responder as perguntas propostas.

Cogitamos buscar dados sobre volume de apostas e valor financeiro envolvido nelas, para cada uma das partidas, o que tornaria a base de dados mais rica e que até mudaria o enfoque das perguntas inicialmente feitas. Mas estes não são disponibilizados pelas empresas, provavelmente por ser uma informação estratégica.

Modelagem

Para este trabalho, o modelo que trabalharemos será o flat. O próprio dataset que foi escolhido já está nesta estrutura, desta forma será necessário apenas fazer alguns tratamentos posteriores.

O dataset traz em um arquivo .txt a descrição detalhada de cada campo das tabelas, que utilizaremos como catálogo de dados de referência para o MVP:

Key to results data:

Div	League Division
Date	Match Date (dd/mm/yy)
Time	Time of match kick off
HomeTeam	Home Team
AwayTeam	Away Team
FTHG and HG	Full Time Home Team Goals
FTAG and AG	Full Time Away Team Goals
FTR and Res	Full Time Result (H=Home Win, D=Draw, A=Away Win)
HTHG	Half Time Home Team Goals
HTAG	Half Time Away Team Goals
HTR	Half Time Result (H=Home Win, D=Draw, A=Away Win)

Match Statistics (where available)

Referee	Match Referee
HS	Home Team Shots
AS	Away Team Shots
HST	Home Team Shots on Target
AST	Away Team Shots on Target
HHW	Home Team Hit Woodwork
AHW	Away Team Hit Woodwork
HC	Home Team Corners
AC	Away Team Corners
HF	Home Team Fouls Committed
AF	Away Team Fouls Committed
HFKC	Home Team Free Kicks Conceded
AFKC	Away Team Free Kicks Conceded
HO	Home Team Offsides
AO	Away Team Offsides
HY	Home Team Yellow Cards
AY	Away Team Yellow Cards
HR	Home Team Red Cards
AR	Away Team Red Cards
HBP	Home Team Bookings Points (10 = yellow, 25 = red)
ABP	Away Team Bookings Points(10 = yellow, 25 = red)

Key to 1X2 (match) betting odds data:

B365H Bet365 home win odds
B365D Bet365 draw odds
B365A Bet365 away win odds
BSH Blue Square home win odds
BSD Blue Square draw odds
BSA Blue Square away win odds
BWH Bet&Win home win odds
BWD Bet&Win draw odds
BWA Bet&Win away win odds
GBH Gamebookers home win odds
GBD Gamebookers draw odds
GBA Gamebookers away win odds
IWH Interwetten home win odds
IWD Interwetten draw odds
IWA Interwetten away win odds
LBH Ladbrokes home win odds
LBD Ladbrokes draw odds
LBA Ladbrokes away win odds
PSH and PH Pinnacle home win odds
PSD and PD Pinnacle draw odds

Bb1X2 Number of BetBrain bookmakers used to calculate match odds averages and maximums
BbMxH Betbrain maximum home win odds
BbAvH Betbrain average home win odds

MaxH Market maximum home win odds
MaxD Market maximum draw win odds
MaxA Market maximum away win odds

PSA and PA Pinnacle away win odds
SOH Sporting Odds home win odds
SOD Sporting Odds draw odds
SOA Sporting Odds away win odds
SBH Sportingbet home win odds
SBD Sportingbet draw odds
SBA Sportingbet away win odds
SJH Stan James home win odds
SJD Stan James draw odds
SJA Stan James away win odds
SYH Stanleybet home win odds
SYD Stanleybet draw odds
SYA Stanleybet away win odds
VCH VC Bet home win odds
VCD VC Bet draw odds
VCA VC Bet away win odds
WHH William Hill home win odds
WHD William Hill draw odds
WHA William Hill away win odds

BbMxD Betbrain maximum draw odds
BbAvD Betbrain average draw win odds
BbMxA Betbrain maximum away win odds
BbAvA Betbrain average away win odds

AvgH Market average home win odds
AvgD Market average draw win odds
AvgA Market average away win odds

Key to total goals betting odds:

BbOU Number of BetBrain bookmakers used to calculate over/under 2.5 goals (total goals) averages and maximums
BbMx>2.5 Betbrain maximum over 2.5 goals
GB>2.5 Gamebookers over 2.5 goals
GB<2.5 Gamebookers under 2.5 goals
B365>2.5 Bet365 over 2.5 goals

Max>2.5 Market maximum over 2.5 goals
Max<2.5 Market maximum under 2.5 goals

BbAv>2.5 Betbrain average over 2.5 goals
BbMx<2.5 Betbrain maximum under 2.5 goals
BbAv<2.5 Betbrain average under 2.5 goals
B365<2.5 Bet365 under 2.5 goals
P>2.5 Pinnacle over 2.5 goals
P<2.5 Pinnacle under 2.5 goals

Avg>2.5 Market average over 2.5 goals
Avg<2.5 Market average under 2.5 goals

Key to Asian handicap betting odds:

BbAH Number of BetBrain bookmakers used to Asian handicap averages and maximums
BbAHh Betbrain size of handicap (home team)
AHh Market size of handicap (home team) (since 2019/2020)
GBAHH Gamebookers Asian handicap home team odds
GBAHA Gamebookers Asian handicap away team odds
GBAH Gamebookers size of handicap (home team)
LBAHH Ladbrokes Asian handicap home team odds
LBAHA Ladbrokes Asian handicap away team odds
LBAH Ladbrokes size of handicap (home team)

MaxAHH Market maximum Asian handicap home team odds
MaxAHA Market maximum Asian handicap away team odds

BbMxAHH Betbrain maximum Asian handicap home team odds
BbAvAHH Betbrain average Asian handicap home team odds
BbMxAHA Betbrain maximum Asian handicap away team odds
BbAvAHA Betbrain average Asian handicap away team odds
B365AHH Bet365 Asian handicap home team odds
B365AHA Bet365 Asian handicap away team odds
B365AH Bet365 size of handicap (home team)
PAHH Pinnacle Asian handicap home team odds
PAHA Pinnacle Asian handicap away team odds

AvgAHH Market average Asian handicap home team odds
AvgAHA Market average Asian handicap away team odds

Além dos resultados e estatísticas da partida, o arquivo traz também as probabilidades de vitória de cada time ou empate nas partidas e outros tipos de apostas, como por exemplo:

- over/under 2,5 gols: é possível apostar se uma determinada partida terá um placar final de até 2 gols; ou de 3 ou mais gols.

- asian handicap: As apostas em handicap asiático são uma forma de apostar no futebol em que os times mais fortes já começam a partida “prejudicados”, de modo que devam vencer por placares maiores para ter sua aposta considerada vencedora.

Não vamos utilizar todos estes dados para o MVP, mas estas últimas categorias permitem realizar mais perguntas além das previamente planejadas.

Observação: verificamos que o arquivo do catálogo de dados apresenta mais informações do que os arquivos .csv. Iremos tratar isso posteriormente, na etapa de ETL.

Carga

Para realizar o processo de extração, transformação e carga (ETL), foi utilizado um notebook no DataBricks. O primeiro passo foi importar o dataset do Kaggle.

```
%sh
kaggle datasets download -d audreyhengruizhang/european-soccer-data
```

```
Dataset URL: https://www.kaggle.com/datasets/audreyhengruizhang/european-soccer-data
License(s): other
Downloading european-soccer-data.zip to /databricks/driver
100%|██████████| 230k/230k [00:00<00:00, 6.54MB/s]
```

O dataset estava em um arquivo .zip, e foi preciso descompactá-lo

```
%sh
unzip /databricks/driver/european-soccer-data.zip
```

Analisando os arquivos descompactados, vemos que ele possui um arquivo .csv para cada liga, além de um arquivo chamado notes.txt, que contém o catálogo de dados exibido na seção anterior.

```
dbutils.fs.ls('dbfs:/FileStore/')
```

```
[FileInfo(path='dbfs:/FileStore/Bundesliga23.csv', name='Bundesliga23.csv', size=135825, modificationTime=1720021690000),
 FileInfo(path='dbfs:/FileStore/LaLiga23.csv', name='LaLiga23.csv', size=168164, modificationTime=1720021691000),
 FileInfo(path='dbfs:/FileStore/Ligue1.csv', name='Ligue1.csv', size=135000, modificationTime=1720021691000),
 FileInfo(path='dbfs:/FileStore/PL23.csv', name='PL23.csv', size=172196, modificationTime=1720021691000),
 FileInfo(path='dbfs:/FileStore/SerieA23.csv', name='SerieA23.csv', size=166581, modificationTime=1720021691000),
 FileInfo(path='dbfs:/FileStore/notes.txt', name='notes.txt', size=6702, modificationTime=1720021692000),
 FileInfo(path='dbfs:/FileStore/tables/', name='tables/', size=0, modificationTime=0)]
```

Ao analisar os arquivos baixados, foi observado que o arquivo PL23.csv possui uma coluna a mais que os demais. Este ponto precisará ser tratado posteriormente.

```
df1 = spark.read.format('csv').option('header', 'true').load('/FileStore/Bundesliga23.csv')
df2 = spark.read.format('csv').option('header', 'true').load('/FileStore/SerieA23.csv')
df3 = spark.read.format('csv').option('header', 'true').load('/FileStore/Ligue1.csv')
df4 = spark.read.format('csv').option('header', 'true').load('/FileStore/LaLiga23.csv')
df5 = spark.read.format('csv').option('header', 'true').load('/FileStore/PL23.csv')
```

```
▸ (5) Spark Jobs
▸ df1: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 103 more fields]
▸ df2: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 103 more fields]
▸ df3: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 103 more fields]
▸ df4: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 103 more fields]
▸ df5: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 104 more fields]
```

A coluna a mais que o arquivo possuía era a “Referee” (árbitro). Esta não será necessária para as perguntas do nosso MVP, então vamos unir os 5 arquivos em um só, desconsiderando esta informação.

```
dfbl = spark.read.format('csv').option('header', 'true').load('/FileStore/Bundesliga23.csv')
dfsa = spark.read.format('csv').option('header', 'true').load('/FileStore/SerieA23.csv')
dfl1 = spark.read.format('csv').option('header', 'true').load('/FileStore/Ligue1.csv')
dfll = spark.read.format('csv').option('header', 'true').load('/FileStore/LaLiga23.csv')
dfpl = spark.read.format('csv').option('header', 'true').load('/FileStore/PL23.csv').drop('Referee')
```

```
▸ (5) Spark Jobs
▸ dfbl: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 103 more fields]
▸ dfsa: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 103 more fields]
▸ dfl1: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 103 more fields]
▸ dfll: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 103 more fields]
▸ dfpl: pyspark.sql.dataframe.DataFrame = [Div: string, Date: string ... 103 more fields]
```

Outro ponto que precisará ser trabalhado são os tipos de dados. Todas as colunas dos arquivos .csv estão no formato string. Vamos converter os campos dos arquivos em formatos que possam ser trabalhados posteriormente (data, hora, int e double). E depois vamos fazer uma junção dos arquivos correspondentes a cada liga em uma única visão (o nosso modelo flat).

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col, to_date

# Inicialize a sessão do Spark
spark = SparkSession.builder.getOrCreate()

# Carregue o arquivo CSV como DataFrame
dfbl = spark.read.csv("/FileStore/Bundesliga23.csv", header=True, inferSchema=True)
# Converta a coluna "date" para o tipo "date"
dfbl = dfbl.withColumn("Date", to_date(col("Date"), "dd-MM-yyyy"))

# Carregue o arquivo CSV como DataFrame
dfsa = spark.read.csv("/FileStore/SerieA23.csv", header=True, inferSchema=True)
# Converta a coluna "date" para o tipo "date"
dfsa = dfsa.withColumn("Date", to_date(col("Date"), "dd-MM-yyyy"))

# Carregue o arquivo CSV como DataFrame
dfl1 = spark.read.csv("/FileStore/Ligue1.csv", header=True, inferSchema=True)
# Converta a coluna "date" para o tipo "date"
dfl1 = dfl1.withColumn("Date", to_date(col("Date"), "dd-MM-yyyy"))
```

```python
# Carregue o arquivo CSV como DataFrame
dfl1 = spark.read.csv("/FileStore/LaLiga23.csv", header=True, inferSchema=True)
# Converta a coluna "date" para o tipo "date"
dfl1 = dfl1.withColumn("Date", to_date(col("Date"), "dd-MM-yyyy"))

# Carregue o arquivo CSV como DataFrame
dfpl = spark.read.csv("/FileStore/PL23.csv", header=True, inferSchema=True).drop('Referee')
# Converta a coluna "date" para o tipo "date"
dfpl = dfpl.withColumn("Date", to_date(col("Date"), "dd-MM-yyyy"))

# Exiba o DataFrame resultante
df_final = dfbl.union(dfl1).union(dfl1).union(dfsa).union(dfpl)
display(df_final)
```

(12) Spark Jobs

dfbl: pyspark.sql.dataframe.DataFrame

```
Div: string
Date: date
Time: timestamp
HomeTeam: string
AwayTeam: string
FTHG: integer
FTAG: integer
FTR: string
HTHG: integer
HTAG: integer
HTR: string
```

	Div	Date	Time	HomeTeam	AwayTeam	FTHG	FTAG
18	D1	2023-08-27	2024-07-08T16:30:00.000+0...	Bayern Munich	Augsburg	3	1
19	D1	2023-09-01	2024-07-08T19:30:00.000+0...	Dortmund	Heidenheim	2	2
20	D1	2023-09-02	2024-07-08T14:30:00.000+0...	Augsburg	Bochum	2	2
21	D1	2023-09-02	2024-07-08T14:30:00.000+0...	Hoffenheim	Wolfsburg	3	1
22	D1	2023-09-02	2024-07-08T14:30:00.000+0...	Leverkusen	Darmstadt	5	1
23	D1	2023-09-02	2024-07-08T14:30:00.000+0...	Stuttgart	Freiburg	5	0
24	D1	2023-09-02	2024-07-08T14:30:00.000+0...	Werder Bremen	Mainz	4	0
25	D1	2023-09-02	2024-07-08T17:30:00.000+0...	M'gladbach	Bayern Munich	1	2
26	D1	2023-09-03	2024-07-08T14:30:00.000+0...	Ein Frankfurt	FC Koln	1	1
27	D1	2023-09-03	2024-07-08T16:30:00.000+0...	Union Berlin	RB Leipzig	0	3
28	D1	2023-09-15	2024-07-08T19:30:00.000+0...	Bayern Munich	Leverkusen	2	2
29	D1	2023-09-16	2024-07-08T14:30:00.000+0...	FC Koln	Hoffenheim	1	3
30	D1	2023-09-16	2024-07-08T14:30:00.000+0...	Freiburg	Dortmund	2	4
31	D1	2023-09-16	2024-07-08T14:30:00.000+0...	Mainz	Stuttgart	1	3
32	D1	2023-09-16	2024-07-08T14:30:00.000+0...	RB Leipzig	Augsburg	3	0

1,751 rows | 18.57 seconds runtime

Refreshed 5 hours ag

Por fim, foi observado que algumas colunas dos arquivos .csv apresentavam nomes que poderiam vir a causar erro nas etapas posteriores, seja porque são palavras que já pertencem à linguagem SQL ou por apresentarem espaços ou caracteres inválidos.

```python
# Renomeie as colunas usando a função withColumnRenamed
df_final = df_final.withColumnRenamed('Div', 'League')
df_final = df_final.withColumnRenamed('Date', 'DateMatch')
df_final = df_final.withColumnRenamed('Time', 'TimeMatch')
df_final = df_final.withColumnRenamed('AS', 'ASS')
df_final = df_final.withColumnRenamed('FTHG and HG', 'FTHGandHG')
df_final = df_final.withColumnRenamed('FTAG and AG', 'FTAGandAG')
df_final = df_final.withColumnRenamed('FTR and Res', 'FTRandRes')
df_final = df_final.withColumnRenamed('B365>2.5', 'B365O25')
df_final = df_final.withColumnRenamed('B365<2.5', 'B365U25')
df_final = df_final.withColumnRenamed('P>2.5', 'PO25')
df_final = df_final.withColumnRenamed('P<2.5', 'PU25')
df_final = df_final.withColumnRenamed('Max>2.5', 'MaxO25')
df_final = df_final.withColumnRenamed('Max<2.5', 'MaxU25')
df_final = df_final.withColumnRenamed('Avg>2.5', 'AvgO25')
df_final = df_final.withColumnRenamed('Avg<2.5', 'AvgU25')

# Exiba o DataFrame com a coluna renomeada
display(df_final)
```

Já tendo os dados e no formato desejado, vamos providenciar a criação do banco de dados “bronze”, para salvar os dados dos 5 arquivos em uma única tabela “europa”. Nela, os dados serão salvos já no formato adequado para o trabalho (seja string, date, timestamp, int ou double).

```
%sql
CREATE DATABASE IF NOT EXISTS bronze
```

OK

03:34 AM (7s) 17 Python

```
dbutils.fs.rm('dbfs:/user/hive/warehouse/bronze.db/europa',True)
df_final.write.format("delta").mode("append").saveAsTable("bronze.europa")
```

(6) Spark Jobs

03:34 AM (2s) 18

```
%sql
select * from bronze.europa
```

(3) Spark Jobs

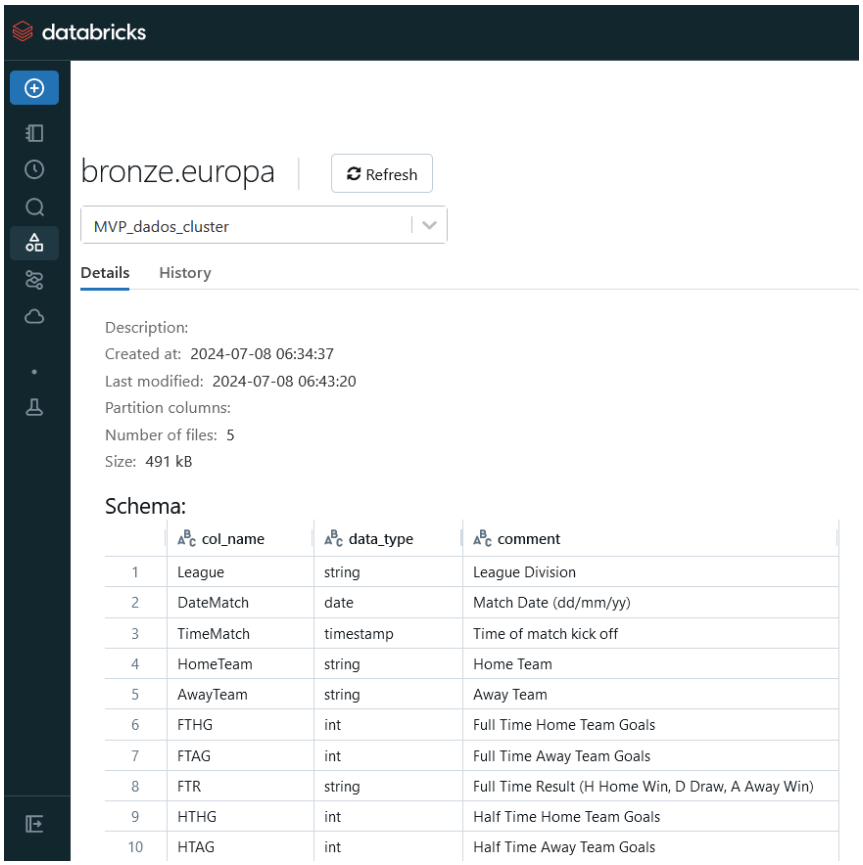
_sqldf: pyspark.sql.dataframe.DataFrame = [League: string, DateMatch: date ... 103 more fields]

Table

	League	DateMatch	TimeMatch	HomeTeam	AwayTeam	FTHG	FTAG
1	E0	2023-08-11	2024-07-08T20:00:00.000+0...	Burnley	Man City	0	
2	E0	2023-08-12	2024-07-08T12:30:00.000+0...	Arsenal	Nott'm Forest	2	
3	E0	2023-08-12	2024-07-08T15:00:00.000+0...	Bournemouth	West Ham	1	
4	E0	2023-08-12	2024-07-08T15:00:00.000+0...	Brighton	Luton	4	
5	E0	2023-08-12	2024-07-08T15:00:00.000+0...	Everton	Fulham	0	
6	E0	2023-08-12	2024-07-08T15:00:00.000+0...	Sheffield United	Crystal Palace	0	

Para manter a o catálogo de dados atualizado também no Databricks, vamos incluir nos metadados da tabela “europa” a descrição de cada campo que foi informada no arquivo “notes.txt”, já desconsiderando as informações do arquivo que não possuem correspondente na tabela.

```
%sql
ALTER TABLE bronze.europa CHANGE COLUMN   League  League  STRING  COMMENT 'League Division';
ALTER TABLE bronze.europa CHANGE COLUMN   DateMatch   DateMatch   DATE  COMMENT 'Match Date (dd/mm/yy)';
ALTER TABLE bronze.europa CHANGE COLUMN   TimeMatch   TimeMatch   TIMESTAMP COMMENT 'Time of match kick off';
ALTER TABLE bronze.europa CHANGE COLUMN   HomeTeam  HomeTeam  STRING  COMMENT 'Home Team';
ALTER TABLE bronze.europa CHANGE COLUMN   AwayTeam  AwayTeam  STRING  COMMENT 'Away Team';
ALTER TABLE bronze.europa CHANGE COLUMN   FTHG  FTHG  INT COMMENT 'Full Time Home Team Goals';
ALTER TABLE bronze.europa CHANGE COLUMN   FTAG  FTAG  INT COMMENT 'Full Time Away Team Goals';
ALTER TABLE bronze.europa CHANGE COLUMN   FTR FTR STRING  COMMENT 'Full Time Result (H Home Win, D Draw, A Away Win)';
ALTER TABLE bronze.europa CHANGE COLUMN   HTHG  HTHG  INT COMMENT 'Half Time Home Team Goals';
ALTER TABLE bronze.europa CHANGE COLUMN   HTAG  HTAG  INT COMMENT 'Half Time Away Team Goals';
ALTER TABLE bronze.europa CHANGE COLUMN   HTR   HTR   STRING  COMMENT 'Half Time Result (H Home Win, D Draw, A Away Win)
';
ALTER TABLE bronze.europa CHANGE COLUMN   HS  HS  INT COMMENT 'Home Team Shots';
ALTER TABLE bronze.europa CHANGE COLUMN   ASS   ASS   INT COMMENT 'Away Team Shots';
ALTER TABLE bronze.europa CHANGE COLUMN   HST   HST   INT COMMENT 'Home Team Shots on Target';
ALTER TABLE bronze.europa CHANGE COLUMN   AST   AST   INT COMMENT 'Away Team Shots on Target';
ALTER TABLE bronze.europa CHANGE COLUMN   HC  HC  INT COMMENT 'Home Team Corners';
ALTER TABLE bronze.europa CHANGE COLUMN   AC  AC  INT COMMENT 'Away Team Corners';
ALTER TABLE bronze.europa CHANGE COLUMN   HF  HF  INT COMMENT 'Home Team Fouls Committed';
ALTER TABLE bronze.europa CHANGE COLUMN   AF  AF  INT COMMENT 'Away Team Fouls Committed';
ALTER TABLE bronze.europa CHANGE COLUMN   HY  HY  INT COMMENT 'Home Team Yellow Cards';
ALTER TABLE bronze.europa CHANGE COLUMN   AY  AY  INT COMMENT 'Away Team Yellow Cards';
ALTER TABLE bronze.europa CHANGE COLUMN   HR  HR  INT COMMENT 'Home Team Red Cards';
```



	col_name	data_type	comment
1	League	string	League Division
2	DateMatch	date	Match Date (dd/mm/yy)
3	TimeMatch	timestamp	Time of match kick off
4	HomeTeam	string	Home Team
5	AwayTeam	string	Away Team
6	FTHG	int	Full Time Home Team Goals
7	FTAG	int	Full Time Away Team Goals
8	FTR	string	Full Time Result (H Home Win, D Draw, A Away Win)
9	HTHG	int	Half Time Home Team Goals
10	HTAG	int	Half Time Away Team Goals

Por fim, apenas para fins didáticos neste MVP, vamos transformar os valores da nova coluna “League”, que no dataset original continha códigos, para o nome do país de cada liga.

2 minutes ago (41s)

```
%sql
UPDATE bronze.europa
SET League = 'Inglaterra'
WHERE League = 'E0';

UPDATE bronze.europa
SET League = 'Itália'
WHERE League = 'I1';

UPDATE bronze.europa
SET League = 'Espanha'
WHERE League = 'SP1';

UPDATE bronze.europa
SET League = 'Alemanha'
WHERE League = 'D1';

UPDATE bronze.europa
SET League = 'França'
WHERE League = 'F1';
```

Análise

Qualidade dos Dados

Alguns problemas encontrados no dataset foram tratados, conforme mostrado em sessões anteriores: exclusão de uma coluna (“referee”), transformação dos tipos de dados, inserção de metadados, alteração do nome das colunas. Também foi verificado que nem todas as informações contidas no arquivo de catálogo de dados não constavam na base, mas essas ausências não trouxeram perda da qualidade para responder as perguntas propostas.

Continuando a análise das informações contidas no banco de dados, concluímos que as colunas que trazem os valores máximo e média das “odds” de cada partida não correspondem aos valores do dataset.

Aqui o exemplo de uma partida entre as equipes Lyon e Le Havre, onde podemos observar uma diferença entre o maior valor para a vitória do time da casa entre todas as 6 casas disponíveis (2,71) é diferente do maior valor (2,75) que o dataset trouxe consolidado na coluna MaxH, e que segundo o catálogo de dados, corresponderia ao “Market maximum home win odds”. Note que também há diferença para a cotação máxima para empate e para vitória do time visitante.

Podemos ver que o mesmo problema ocorre quando comparamos as médias:

```
%sql
SELECT
    ROUND((B365H + BWH + IWH + PSH + WHH + VCH) / 6, 2) AS media_H,
    AvgH,
    ROUND((B365D + BWD + IWD + PSD + WHD + VCD) / 6, 2) AS media_D,
    AvgD,
    ROUND((B365H + BWA + IWA + PSA + WHA + VCA) / 6, 2) AS media_A,
    AvgA
FROM
    bronze.europa
WHERE
    HomeTeam = "Lyon"
    AND AwayTeam = "Le Havre";
```

Just now (2s) 23

(2) Spark Jobs

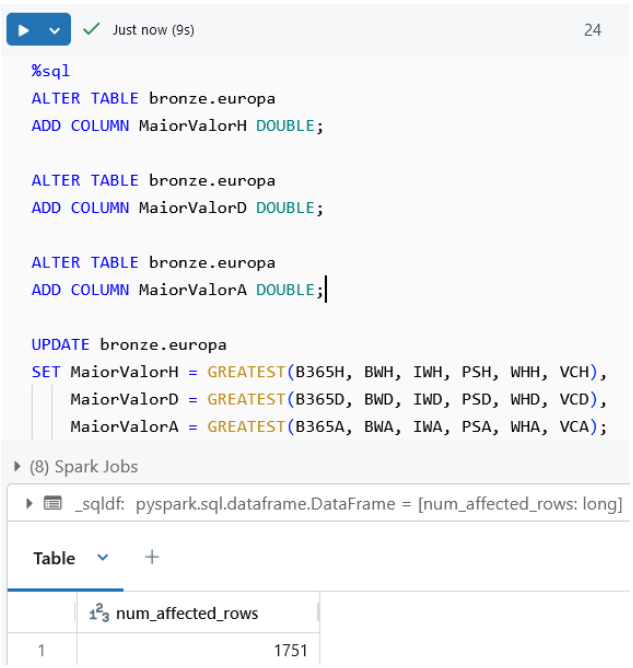
_sqldf: pyspark.sql.dataframe.DataFrame = [media_H: double, AvgH: double ... 4 more fields]

Table

	1.2 media_H	1.2 AvgH	1.2 media_D	1.2 AvgD	1.2 media_A	1.2 AvgA
1	1.67	1.69	3.97	4.12	4.11	4.68

Os arquivos apresentam as cotações de 6 casas de apostas, enquanto o arquivo de catálogo de informações mostra dados de 12 casas. Acreditamos que os máximos e médias tenham sido calculados em cima da base completa com estas 12 casas, mas não podemos garantir essa hipótese. O mesmo problema com máximos e médias ocorre para as cotações de “over/under” e “asian handicap” (mas estas, a princípio, não serão necessárias para responder às perguntas, por isso não iremos ajustar neste MVP).

Desta forma, a sugestão do trabalho será criar colunas para registrar os valores de máximo e média que sejam necessários para responder às perguntas propostas, considerando apenas das empresas cujos dados constam no dataset.



```
%sql
ALTER TABLE bronze.europa
ADD COLUMN MaiorValorH DOUBLE;

ALTER TABLE bronze.europa
ADD COLUMN MaiorValorD DOUBLE;

ALTER TABLE bronze.europa
ADD COLUMN MaiorValorA DOUBLE;

UPDATE bronze.europa
SET MaiorValorH = GREATEST(B365H, BWH, IWH, PSH, WHH, VCH),
    MaiorValorD = GREATEST(B365D, BWD, IWD, PSD, WHD, VCD),
    MaiorValorA = GREATEST(B365A, BWA, IWA, PSA, WHA, VCA);
```

Just now (9s) 24

(8) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [num_affected_rows: long]

Table

	num_affected_rows
1	1751

Como foi observado que, para algumas partidas, o valor das cotações de alguma casa de apostas poderia estar indisponível (e estivesse “null” no banco de dados), usamos um tratamento com a função COALESCE para que as médias desconsiderassem esses casos.

Solução do Problema

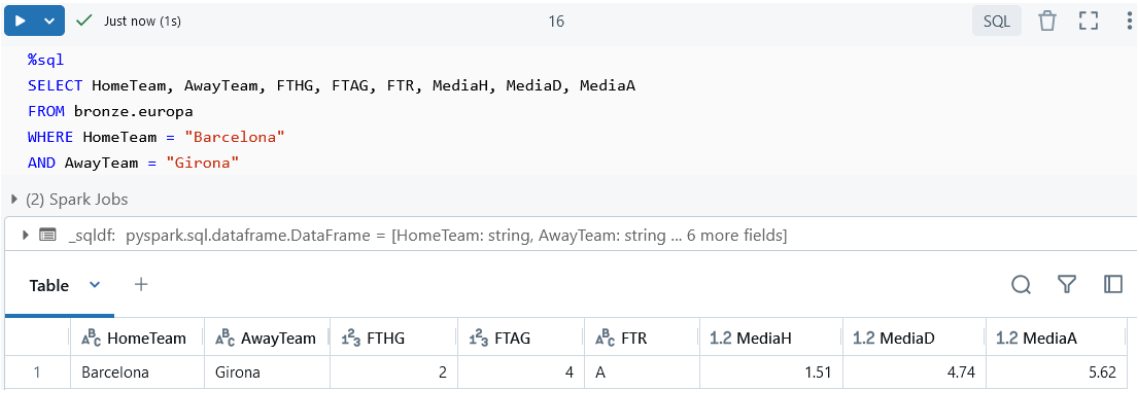
Vamos repetir aqui as perguntas inicialmente feitas, e para responde-las, precisaremos dar algumas explicações e possivelmente realizar novas transformações de dados antes de buscar a resposta adequada.

- Qual o índice de acerto das previsões das casas de aposta para o resultado da partida?

À medida que estudamos como funcionam as apostas, chegamos à conclusão de que as cotações das apostas não fornecem subsídios para tentar prever o placar final das partidas. Então vamos trabalhar apenas com o resultado final: se a partida terá como vencedora a equipe da casa, a visitante ou terminará em empate.

Para tentar responder à esta pergunta, pensamos em duas linhas: uma com valores absolutos e outra com relativos.

Vamos selecionar uma partida do dataset para exemplificar as duas abordagens:



```
%sql
SELECT HomeTeam, AwayTeam, FTHG, FTAG, FTR, MediaH, MediaD, MediaA
FROM bronze.europa
WHERE HomeTeam = "Barcelona"
AND AwayTeam = "Girona"
```

(2) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [HomeTeam: string, AwayTeam: string ... 6 more fields]

Table

	HomeTeam	AwayTeam	FTHG	FTAG	FTR	MediaH	MediaD	MediaA
1	Barcelona	Girona	2	4	A	1.51	4.74	5.62

Para esta partida, as melhores “odds” (1,51) indicavam o time da casa, o Barcelona, como vencedor da partida. Como o placar final foi 4x 2 para o visitante, o Girona, podemos considerar que as casas de apostas simplesmente erraram o vencedor. Esta seria a abordagem absoluta.

Porém, vamos ver como as “odds” são calculadas:

Quando, no exemplo acima, a casa indica que a cotação para o Barcelona vencer é de 1,51, podemos transpor isso para um valor percentual:

Vitória do Barcelona: $\frac{1}{1{,}51} * 100 = 66{,}23\%$

Logo, o Barcelona teria 66,23% de chances de vencer a partida.

Analogamente, as probabilidades de empate e de vitória do Girona seriam de, respectivamente, 21,1% e 17,79%:

Empate: $\frac{1}{4{,}74} * 100 = 21{,}1\%$

Vitória do Girona: $\frac{1}{5{,}62} * 100 = 17{,}79\%$

Podemos observar que, neste exemplo, o somatório ultrapassa os 100%. Neste caso, fica em 105,12%. Isso é o que, no mercado de apostas, é chamado de “margem”. Na prática, seria uma espécie de comissão que as casas levam.

Se normalizarmos este valor, para que o total seja de 100%, podemos considerar que o percentual de probabilidade de cada resultado seria:

Vitória do Barcelona: 63%

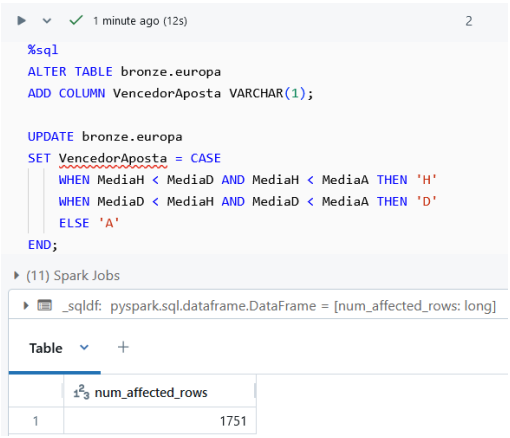
Empate: 20,08%

Vitória do Girona: 16,92%

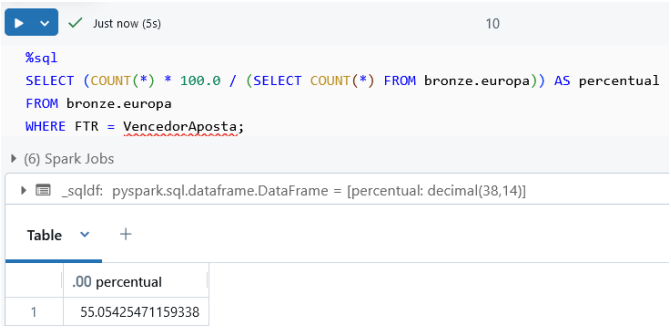
Enquanto, na primeira abordagem, consideramos que a casa de apostas acertou 0% o resultado final, nesta segunda consideramos que ela estimou em 16,92% a chance de vitória da equipe que ganhou a partida.

Para ambas as abordagens, será necessário voltar a trabalhar com novas etapas de transformação da nossa tabela flat, acrescentando colunas e dados nela.

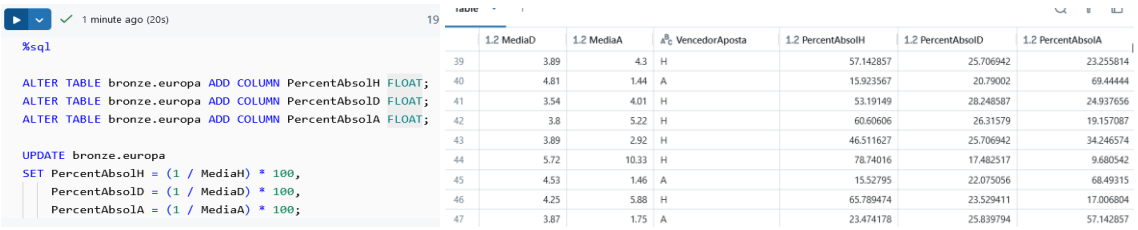
Desta forma, incluímos uma nova coluna chamada VencedorAposta. Ela indica através de um caractere (H = home, time da casa; D = draw, empate; A = away, visitante), qual o mais provável vencedor segundo a média das cotações das casas de apostas.



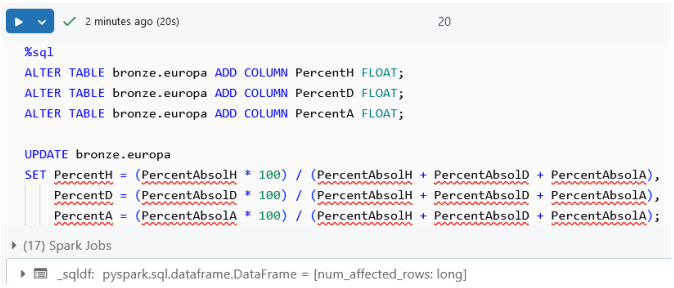
A partir daí podemos tentar responder à pergunta sobre o índice de acerto absoluto das casas de apostas. Neste caso, em 55,05% dos casos a equipe que tinha melhores cotações de fato venceu a partida.



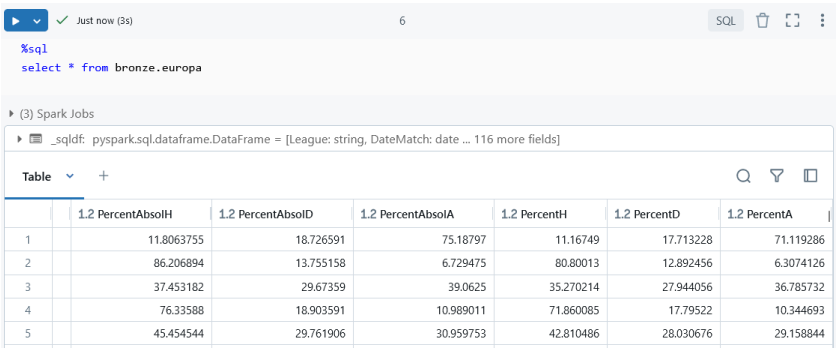
Para tratarmos a segunda abordagem, ponderando o peso de cada cotação, vamos precisar novamente enriquecer a tabela flat, incluindo novas informações vindas de cálculos com base na própria tabela. Primeiro os percentuais absolutos. Aquele que no exemplo da partida Barcelona x Girona ficou em 105,12%, incluindo a margem.



Agora criamos novas 3 colunas, desta vez com o percentual com a probabilidade de cada resultado segundo a média das casas de apostas.



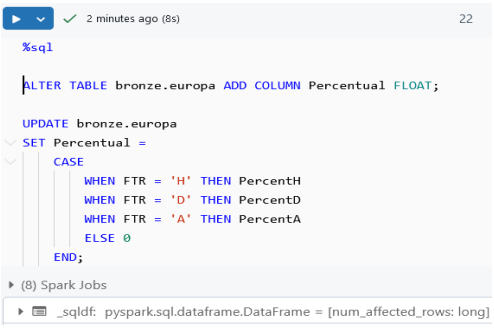
```
%sql
ALTER TABLE bronze.europa ADD COLUMN PercentH FLOAT;
ALTER TABLE bronze.europa ADD COLUMN PercentD FLOAT;
ALTER TABLE bronze.europa ADD COLUMN PercentA FLOAT;

UPDATE bronze.europa
SET PercentH = (PercentAbsolH * 100) / (PercentAbsolH + PercentAbsolD + PercentAbsolA),
    PercentD = (PercentAbsolD * 100) / (PercentAbsolH + PercentAbsolD + PercentAbsolA),
    PercentA = (PercentAbsolA * 100) / (PercentAbsolH + PercentAbsolD + PercentAbsolA);
```

```
%sql
select * from bronze.europa
```

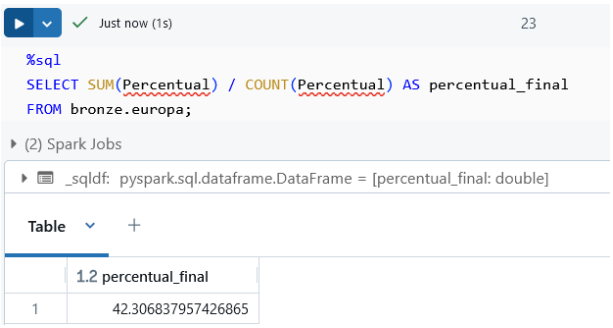

	1.2 PercentAbsolH	1.2 PercentAbsolD	1.2 PercentAbsolA	1.2 PercentH	1.2 PercentD	1.2 PercentA
1	11.8063755	18.726591	75.18797	11.16749	17.713228	71.119286
2	86.206894	13.755158	6.729475	80.80013	12.892456	6.3074126
3	37.453182	29.67359	39.0625	35.270214	27.944056	36.785732
4	76.33588	18.903591	10.989011	71.860085	17.79522	10.344693
5	45.454544	29.761906	30.959753	42.810486	28.030676	29.158844

Para facilitar as operações, mais uma coluna, consolidando em percentual a cotação do resultado efetivamente ocorrido. Apesar de ser possível fazer esta operação sem a necessidade de criação desta coluna, achamos que isso traria mais clareza na visualização dos resultados.



```
%sql

ALTER TABLE bronze.europa ADD COLUMN Percentual FLOAT;

UPDATE bronze.europa
SET Percentual =
    CASE
        WHEN FTR = 'H' THEN PercentH
        WHEN FTR = 'D' THEN PercentD
        WHEN FTR = 'A' THEN PercentA
        ELSE 0
    END;
```


Finalmente, chegamos ao índice de acerto na predição das casas de apostas, considerando o peso de suas cotações: 42,31%.



```
%sql
SELECT SUM(Percentual) / COUNT(Percentual) AS percentual_final
FROM bronze.europa;
```


	1.2 percentual_final
1	42.306837957426865

- Alguma liga segue mais o padrão destas previsões? Alguma está mais propensa a "zebras"?

Novamente vamos usar tanto os resultados absolutos das partidas quanto os ponderados pelas cotações.

Just now (6s) 13

```
%sql
SELECT europa.League,
       (COUNT(*) * 100.0 / total_por_div) AS percentual_div,
       (COUNT(*) * 100.0 / (SELECT COUNT(*) FROM bronze.europa )) AS percentual_total
FROM bronze.europa
INNER JOIN (
    SELECT League AS subquery_Div, COUNT(*) AS total_por_div
    FROM bronze.europa
    GROUP BY League
) AS subquery ON europa.League = subquery.subquery_Div
WHERE FTR = VencedorAposta
GROUP BY europa.League, total_por_div
ORDER BY percentual_div DESC;
```

(8) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [League: string, percentual_div: decimal(38,14) ... 1 more field]

Table

	League	.00 percentual_div	.00 percentual_total
1	Inglaterra	59.21052631578947	12.84980011422045
2	Espanha	55.52631578947368	12.05025699600228
3	Itália	54.88126649076517	11.87892632781268
4	Alemanha	54.24836601307190	9.48029697315820
5	França	50.32679738562092	8.79497430039977

Para valores absolutos, as ligas espanhola, italiana e alemã ficaram mais próximas da média de 55,05% de acerto. Chama a atenção a liga inglesa, com índice de acerto bem acima da média, e a francesa que ficou quase 5% abaixo.

Just now (2s) 26

```
%sql
SELECT League, SUM(Percentual) / COUNT(Percentual) AS soma_dividida
FROM bronze.europa
GROUP BY League
ORDER BY soma_dividida DESC;
```

(2) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [League: string, soma_dividida: double]

Table

	League	1.2 soma_dividida
1	Inglaterra	45.17810327379327
2	Alemanha	43.32101502605513
3	Espanha	41.89480343115957
4	Itália	41.427165997688874
5	França	39.32824249828563

Para os valores ponderados, podemos ver que a tendência se mantém, com a liga inglesa acima e a francesa abaixo da média de 42,31%.

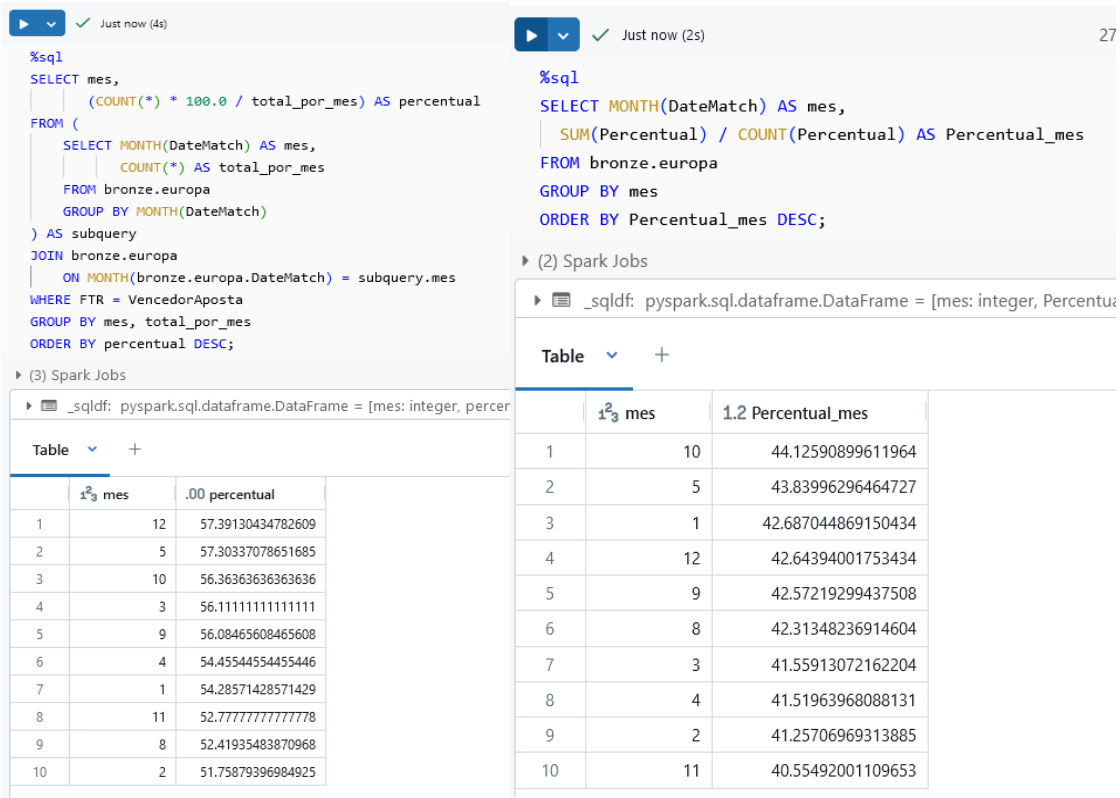
Entretanto, temos um outro dado interessante: desta vez a Alemanha ficou acima da média, e a Espanha abaixo, ao contrário do que ocorreu na análise anterior. Ou seja, embora as casas de apostas tenham mais vezes acertado o vencedor final das partidas da liga espanhola, na liga alemã os acertos foram mais assertivos.

Por fim, cabe mencionar que, se estivéssemos trabalhando em um Data Warehouse, também poderíamos usar uma dimensão para segmentar os resultados por localização (liga).

- Em algum período da temporada, as probabilidades de apostas se confirmam mais ou menos do que em outros períodos?

Nossa hipótese inicial era que, nos primeiros jogos da temporada, as casas de aposta teriam mais dificuldade em "prever" os resultados das partidas, pois haveria menos histórico para calcular estas chances. À medida que os meses passassem, o percentual de acerto aumentaria.

Portanto, nossa ideia será criar uma consulta para segmentar por mês os índices de acertos, tanto absolutos quanto relativos. Em um Data Warehouse, também poderíamos usar a dimensão tempo para trazer estes resultados.



```
%sql
SELECT mes,
       (COUNT(*) * 100.0 / total_por_mes) AS percentual
FROM (
    SELECT MONTH(DateMatch) AS mes,
           COUNT(*) AS total_por_mes
    FROM bronze.europa
    GROUP BY MONTH(DateMatch)
) AS subquery
JOIN bronze.europa
    ON MONTH(bronze.europa.DateMatch) = subquery.mes
WHERE FTR = VencedorAposta
GROUP BY mes, total_por_mes
ORDER BY percentual DESC;
```

(3) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [mes: integer, percer

Table

	mes	.00 percentual
1	12	57.39130434782609
2	5	57.30337078651685
3	10	56.36363636363636
4	3	56.11111111111111
5	9	56.08465608465608
6	4	54.45544554455446
7	1	54.28571428571429
8	11	52.77777777777778
9	8	52.41935483870968
10	2	51.75879396984925

Just now (2s)

```
%sql
SELECT MONTH(DateMatch) AS mes,
  SUM(Percentual) / COUNT(Percentual) AS Percentual_mes
FROM bronze.europa
GROUP BY mes
ORDER BY Percentual_mes DESC;
```

(2) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [mes: integer, Percentua

Table

	mes	1.2 Percentual_mes
1	10	44.12590899611964
2	5	43.83996296464727
3	1	42.687044869150434
4	12	42.64394001753434
5	9	42.57219299437508
6	8	42.31348236914604
7	3	41.55913072162204
8	4	41.51963968088131
9	2	41.25706969313885
10	11	40.55492001109653

Ao agruparmos os resultados por mês, não vemos uma lógica na distribuição. Tanto na abordagem absoluta quanto na relativa pelas cotações, temos meses do início e do final da temporada tanto acima quanto abaixo da média, sem seguir um padrão. Logo, a resposta que os dados trouxeram para essa pergunta não corroborou a hipótese levantada.

- Há alguma distorção entre as probabilidades que cada empresa trabalha antes das partidas, ou elas seguem o mesmo padrão?

Apesar do catálogo de dados inicial indicar que o dataset trazia dados de 12 casas de apostas, verificamos na seção “Qualidade de dados” que na verdade apenas as cotações de 6 empresas constavam nos arquivos. São elas:

- Bet365
- Bet&Win
- Interwetten
- Pinnacle
- VC Bet
- William Hill

Para verificar se as empresas trabalham com cotações parecidas, fizemos uma análise do desvio padrão para a cotação de vitória do time da casa, empate e vitória do visitante para cada uma das empresas, calculando o seu desvio padrão da média das cotações.

```
12:54 PM (6s)
%sql
SELECT "Bet365" AS Casa_aposta,
       ROUND(STDDEV(B365H - MediaH), 3) AS DesvioPadraoH,
       ROUND(STDDEV(B365D - MediaD), 3) AS DesvioPadraoD,
       ROUND(STDDEV(B365A - MediaA), 3) AS DesvioPadraoA
FROM bronze.europa

UNION ALL

SELECT "Bet&Win" AS Casa_aposta,
       ROUND(STDDEV(BWH - MediaH), 3) AS DesvioPadraoH,
       ROUND(STDDEV(BWD - MediaD), 3) AS DesvioPadraoD,
       ROUND(STDDEV(BWA - MediaA), 3) AS DesvioPadraoA
FROM bronze.europa

UNION ALL

SELECT "Interwetten" AS Casa_aposta,
       ROUND(STDDEV(IWH - MediaH), 3) AS DesvioPadraoH,
       ROUND(STDDEV(IWD - MediaD), 3) AS DesvioPadraoD,
       ROUND(STDDEV(IWA - MediaA), 3) AS DesvioPadraoA
FROM bronze.europa

UNION ALL

SELECT "Pinnacle" AS Casa_aposta,
       ROUND(STDDEV(PSH - MediaH), 3) AS DesvioPadraoH,
       ROUND(STDDEV(PSD - MediaD), 3) AS DesvioPadraoD,
       ROUND(STDDEV(PSA - MediaA), 3) AS DesvioPadraoA
FROM bronze.europa

UNION ALL

SELECT "VC Bet" AS Casa_aposta,
       ROUND(STDDEV(VCH - MediaH), 3) AS DesvioPadraoH,
       ROUND(STDDEV(VCD - MediaD), 3) AS DesvioPadraoD,
       ROUND(STDDEV(VCA - MediaA), 3) AS DesvioPadraoA
FROM bronze.europa

UNION ALL

SELECT "William Hill" AS Casa_aposta,
       ROUND(STDDEV(WHH - MediaH), 3) AS DesvioPadraoH,
       ROUND(STDDEV(WHD - MediaD), 3) AS DesvioPadraoD,
       ROUND(STDDEV(WHA - MediaA), 3) AS DesvioPadraoA
FROM bronze.europa;
(7) Spark Jobs
_sqldf: pyspark.sql.dataframe.DataFrame = [Casa_aposta: string, DesvioPadraoH: double ... 2 more field
```

Table

	Casa_aposta	DesvioPadraoH	DesvioPadraoD	DesvioPadraoA
1	Bet365	0.138	0.161	0.413
2	Bet&Win	0.127	0.096	0.321
3	Interwetten	0.116	0.21	0.286
4	Pinnacle	0.102	0.211	0.271
5	VC Bet	0.15	0.167	0.429
6	William Hill	0.128	0.183	0.295

Podemos observar que o desvio padrão é menor e mais uniforme nas cotações de vitória do time da casa.

Isso ocorre porque as equipes que jogam em casa têm maior probabilidade de vencer a partida do que as equipes visitantes, conforme podemos verificar numa consulta a todas as partidas da temporada.

```
1 minute ago (1s)                                                    30
%sql
SELECT
      (SUM(CASE WHEN HTR = 'H' THEN 1 ELSE 0 END)/COUNT(*) * 100.0 ) AS vitoria_casa,
      (SUM(CASE WHEN HTR = 'D' THEN 1 ELSE 0 END)/COUNT(*) * 100.0 ) AS empate,
      (SUM(CASE WHEN HTR = 'A' THEN 1 ELSE 0 END)/COUNT(*) * 100.0 ) AS vitoria_visitante
FROM bronze.europa;
(2) Spark Jobs
_sqldf: pyspark.sql.dataframe.DataFrame = [vitoria_casa: double, empate: double ... 1 more field]
```

Table

	vitoria_casa	empate	vitoria_visitante
1	33.6950314106225	40.8909194745...	25.41404911479155

Já para vitória do visitante, o desvio padrão entre as cotações de cada empresa foi maior. Por ser um resultado mais improvável, cada empresa deve ter uma política diferente para definir estas “odds” e, por consequência, atrair apostadores em busca de ganhos maiores.

Autoavaliação

Inicialmente, a ideia deste trabalho seria trabalhar alguma forma de detectar indícios de fraude em apostas esportivas. Mas isso requereria uma base com dados de volumes de apostas e de quantias financeiras envolvidas. Estas informações, entretanto, não são abertas pelas empresas, e esta possibilidade foi descartada no início.

Entretanto, na procura por estes dados, encontrei este dataset, que embora pequeno, trouxe uma massa de dados suficiente para elaborar algumas perguntas e, a partir disso, levantar hipóteses sobre como funciona o mercado de apostas por parte das empresas que comercializam este serviço, comprovando-as ou não.

Para tentar extrair mais informações deste mesmo dataset, vejo a possibilidade de tentar correlacionar estatísticas de partidas (chutes a gol, escanteios, faltas etc.) com as cotações. Será que estes dados entram na equação onde elas calculam as probabilidades de vitória de cada equipe? Creio que isso iria demandar um conhecimento em estatística que iria além do básico, e por isso não foi possível seguir com perguntas relacionadas a estes dados para a entrega deste MVP. Mas isso pode ser explorado futuramente.

Outro motivo da escolha deste dataset foi o potencial dele ser utilizado nos outros MVPs do curso, podendo ser base para trabalhos de machine learning, por exemplo.

Apesar de tentar seguir o desenvolvimento do trabalho de acordo com a sequência proposta pelo enunciado do problema, percebi que em algumas vezes era necessário retornar a alguma etapa anterior. Por exemplo, quando já estava no trabalho da responder a uma pergunta, precisei retornar ao trabalho de transformação dos dados, para apurar alguma informação necessária. Achei mais didático manter uma ordem cronológica, mesmo que para isso estes pontos ficassem fora da seção do trabalho mais adequada. Espero que esta escolha ajude no entendimento de quem ler este trabalho.

Particularmente, a minha maior dificuldade no trabalho foi definir a plataforma, entender o básico de seu funcionamento e da configuração do ambiente. Diria que no mínimo 60% do tempo deste trabalho foi dedicado a estas atividades. Mas foi fundamental esse entendimento sobre Databricks para conseguir enfim realizar as atividades de ETL e explorar as consultas.