دانشگاه اصفهان
دانشکده مهندسی کامپیوتر

تمرین سوم: انبارداده **استاد:** دکتر احمد براآنی **درس:** سیستم‌های تصمیم‌یار

نام و نام خانوادگی: سید عمید اسدالهی مجد **شماره دانشجویی:** ۴۰۰۳۶۱۴۰۰۴

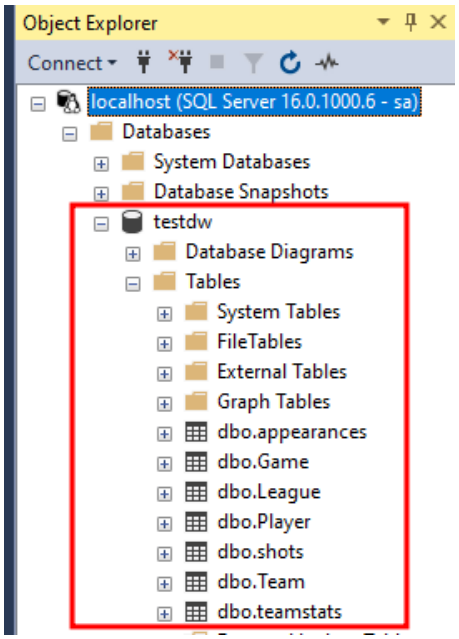
آدرس گیت تمرین: https://github.com/amidmajd/dss-datawarehouse

بخش اول: انتخاب داده‌ها

داده‌های فوتبال برای این تمرین انتخاب شده‌اند. این داده‌ها از بازیکنان، بازی‌ها و تیم‌های حاضر در ۵ لیگ برتر اروپا هستند. این داده‌ها در سایت Kaggle در اینجا قابل دسترسی هستند. این داده‌ها شامل ۴ جدول اصلی بازی‌ها، لیگ‌ها، بازیکنان و تیم‌ها است. همچنین ۳ رابطه داریم که جداول حضور بازیکن در بازی، شوت‌های بازیکن در بازی و اطلاعات بازی تیم‌ها است.

بخش دوم: ساخت انبارداده و نمایش ساختار داده‌ها

داده‌ها در SQL Server لود شدند و یک انبار داده ساخته‌شد. داده‌ها دارای ۴ بعد (Dimension) و ۳ حقیقت (Fact) هستند. داده‌های احتمالی از سایت‌های understat.com و football-data.co.uk که به ترتیب مربوط به پیشبینی‌های بازی و اطلاعات شرط‌بندی قبل بازی هستند استخراج شده‌اند.



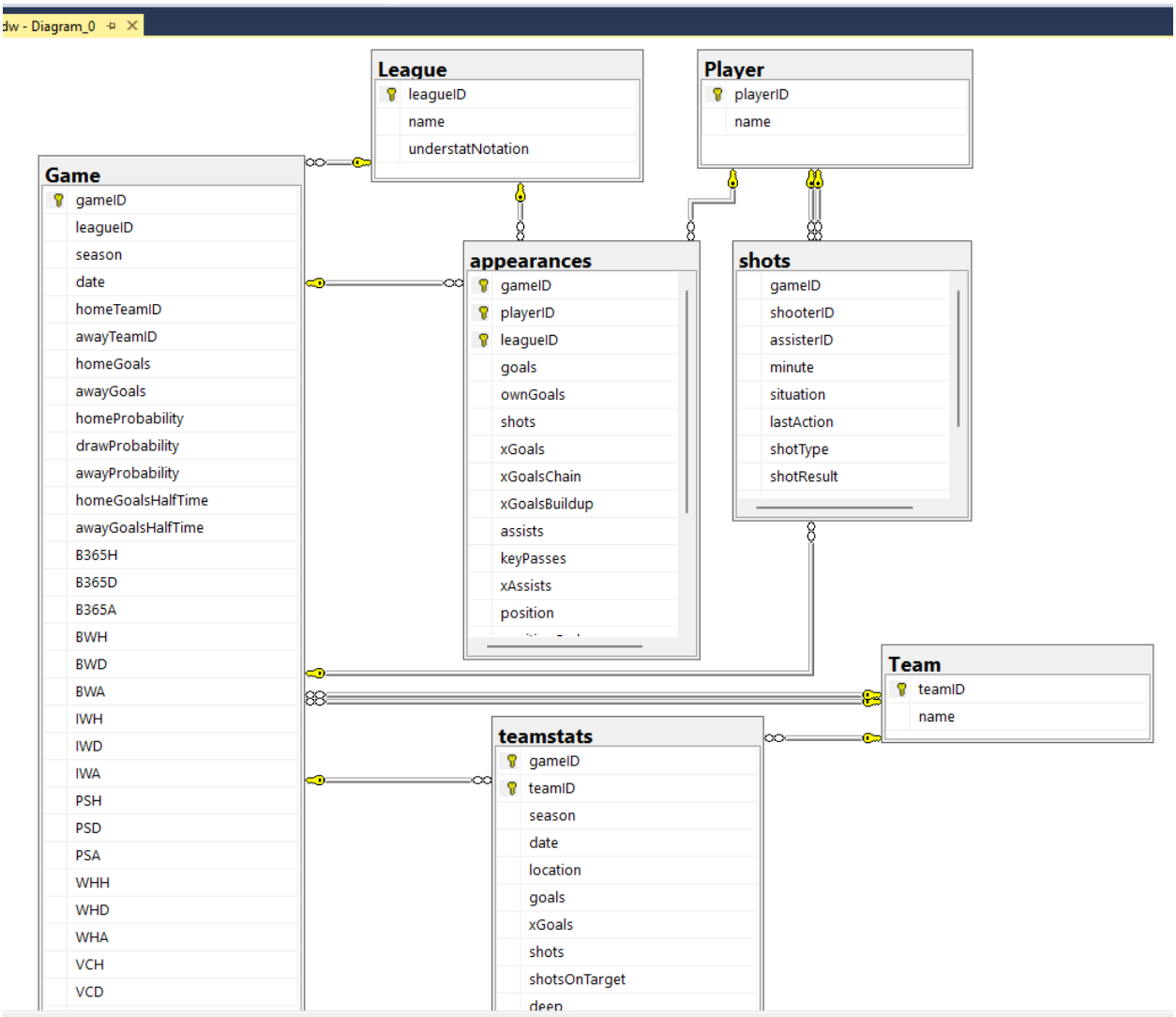
ابعاد:

- League: شامل نام لیگ‌ها و id هر لیگ
- Team: شامل نام تیم و id هر تیم
- Player: شامل نام هر بازیکن و id هر بازیکن
- Game: شامل id هر بازی انجام‌شده و اطلاعات آن مانند تاریخ، فصل، احتمال بر هر تیم و ... و اینکه متعلق به کدام لیگ است.

حقایق:

- appearances : شامل کلیدهای بازی، بازیکن و لیگ است. این حقیقت نشان‌دهنده حضور هر بازیکن در لیگ و بازی انجام شده است. همچنین شامل اطلاعات مربوطه تعداد شوت‌ها، پاس‌ها و ... هر بازیکن است.
- shots : شامل کلیدهای بازی، بازیکن شوت‌زنند و بازیکن همکاری‌کننده (دو کلید خارجی از بعد بازیکن) در شوت می‌باشد. همچنین اطلاعات مربوط به هر شوت مانند گل شدن، دقیقه و موقعیت را به ازای هر شوت نگهداری می‌کند.
- teamstats : شامل کلیدهای تیم و بازی می‌باشد. همچنین اطلاعات هر تیم به ازای هر بازی را دارد. این اطلاعات شامل برد و باخت تیم در بازی، تعداد کارت‌های دریافت‌شده، احتمال تعداد گل تیم و ... می‌باشد.

در جدول زیر یک ساختار **Fact Constellation** از انبارداده‌ی نهایی قابل مشاهده است.



بخش سوم: یادگیری ماشین

**** فایل ml.ipynb شامل کد مربوط به این بخش است. همچنین فایل ml.pdf نمایش از کد و اجرای آن است ****

در این بخش داده‌های مورد نیاز پس از انجام join برروی ابعاد League، Team، Game و teamstats از انبار داده در پایتون و توسط کتابخوانه‌ی pyodbc دریافت می‌شوند. به شکل زیر به انبارداده‌ی مستقر در sql server متصل می‌شویم:

```
sql_conn = pyodbc.connect(
    f"DRIVER={{ODBC Driver 18 for SQL Server}};SERVER={SERVER_ADDRESS};DATABASE={DATABASE};UID={USER};PWD={PASSWORD};TrustServerCertificate=yes;"
)
cursor = sql_conn.cursor()
```

سپس عمل join را انجام داده و داده‌ها را فراخوانی می‌کنیم. متغیر select_cols مربوط به ستون‌های انتخابی است. متغیر query مربوط به کوئری دریافت از انبارداده‌ها است. در شکل زیر تعداد داده‌ها پس از join و ستون‌های نهایی دریافتی قابل مشاهده است.

```
Length on selected data:  12680
Data labels:  ['gameID', 'leagueName', 'homeTeamName', 'homeTeamID', 'awayTeamID', 'season', 'homeProbability', 'drawProbability', 'awayProbability',
'homeGoalsProbability', 'B365H', 'B365D', 'B365A', 'BWH', 'BWD', 'BWA', 'IWH', 'IWD', 'IWA', 'PSH', 'PSD', 'PSA', 'WHH', 'WHD', 'WHA', 'VCH', 'VCD', 'VCA', 'PSCH',
'PSCD', 'PSCA', 'result']
```

سپس داده‌ها را با استفاده از کتابخوانه‌ی pandas به یک dataframe تبدیل می‌کنیم تا کار کردن با آن راحت‌تر باشد. سپس نوع هر ستون را به نوع داده‌ای صحیح تبدیل می‌کنیم و سطرهایی که مقدار null دارند را حذف می‌کنیم (زیرا جمعا حدود ۳۰ سطر بودند). داده‌های نهایی به شکل زیر هستند:

```
df = pd.DataFrame(data=np.array(data), columns=data_labels)

df = df.dropna()
df = df.convert_dtypes()

df
```

	gameID	leagueName	homeTeamName	homeTeamID	awayTeamID	season	homeProbability	drawProbability	awayProbability	homeGoalsProbability	...	WHH	WHD	WHA	VCH	VCD	VCA	PSCH	PSCD	PSCA	result
0	81	Premier League	Manchester United	89	82	2015	0.2843	0.3999	0.3158	0.627539	...	1.62	3.6	6.0	1.67	4.0	5.75	1.64	4.07	6.04	W
1	82	Premier League	Bournemouth	73	71	2015	0.3574	0.35	0.2926	0.876106	...	1.91	3.5	4.0	2.0	3.5	4.2	1.82	3.88	4.7	L
2	83	Premier League	Everton	72	90	2015	0.2988	0.4337	0.2675	0.604226	...	1.73	3.5	5.0	1.73	3.9	5.4	1.75	3.76	5.44	D
3	84	Premier League	Leicester	75	77	2015	0.6422	0.2057	0.1521	2.56803	...	2.0	3.1	2.7	2.0	3.4	4.33	1.79	3.74	5.1	W
4	85	Premier League	Norwich	79	78	2015	0.1461	0.2159	0.638	1.13076	...	2.6	3.1	2.88	2.6	3.25	3.0	2.46	3.39	3.14	L
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
12675	16131	Ligue 1	Nantes	168	166	2020	0.2812	0.2671	0.4517	1.41119	...	1.5	4.5	6.0	1.5	4.33	6.0	1.58	4.36	6.18	L
12676	16132	Ligue 1	Reims	177	176	2020	0.3367	0.2999	0.3634	1.19819	...	2.5	2.9	3.2	2.4	3.1	3.0	2.66	3.28	2.93	L
12677	16133	Ligue 1	Rennes	163	235	2020	0.6719	0.2502	0.0779	1.33269	...	1.32	5.25	9.0	1.3	5.25	9.0	1.23	6.85	12.59	W
12678	16134	Ligue 1	Saint-Etienne	175	181	2020	0.3541	0.301	0.3449	1.4605	...	1.29	5.25	11.0	1.29	5.25	9.5	1.29	5.97	10.8	L
12679	16135	Ligue 1	Strasbourg	225	179	2020	0.1748	0.4863	0.3389	0.32396	...	2.62	2.2	4.33	2.63	2.2	3.25	2.69	2.3	4.18	D

12605 rows × 32 columns

مقادیر یکتای ستون‌های با نوع داده‌ای string به شکل زیر هستند:

```
print("* Unique leagues:", df.leagueName.unique(), end="\n\n")
print("* Unique teams:", df.homeTeamName.unique(), end="\n\n")
print("* Unique results:", df.result.unique(), end="\n\n")
```

```
* Unique leagues: <StringArray>
['Premier League', 'Serie A', 'Bundesliga', 'La Liga', 'Ligue 1']
Length: 5, dtype: string

* Unique teams: <StringArray>
[   'Manchester United',          'Bournemouth',              'Everton',
           'Leicester',              'Norwich',              'Chelsea',
     'Newcastle United',              'Arsenal',                'Stoke',
 'West Bromwich Albion',
 ...
     'Sheffield United',             'Mallorca',         'Union Berlin',
              'Brest',                'Lecce',              'Brescia',
              'Leeds',               'Spezia',                'Cadiz',
    'Arminia Bielefeld']
Length: 146, dtype: string

* Unique results: <StringArray>
['W', 'L', 'D']
Length: 3, dtype: string
```

ستون‌های leagueName و homeTeamName برای انجام یادگیری ماشین حذف می‌شوند زیرا نام لیگ و تیم‌ها هستند و فقط مقادیر id آن‌ها یعنی ستون‌های leagueID و homeTeamID نگهداری می‌شوند.

همچنین ستون result بیانگر نتیجه بازی با سه حالت برد تیم خانه، باخت تیم خانه و مساوی است. تیم خانه همان تیم اصلی و تیم مهمان همان تیم حریف در نظر گرفته شده است. این ستون به مقادیر صفر یا باخت، یک یا برد و دو یا مساوی تبدیل می‌شود.

در نهایت ستون‌های gameID و season هم از داده‌ها حذف می‌شوند و یک دیتافریم جدید با نام df_for_ml آماده‌ی یادگیری ماشین ساخته می‌شود.

دیتافریم نهایی آماده برای یادگیری ماشین به شکل زیر است:

	homeTeamID	awayTeamID	homeProbability	drawProbability	awayProbability	homeGoalsProbability	B365H	B365D	B365A	BWH	...	WHH	WHD	WHA	VCH	VCD	VCA	PSCH	PSCD	PSCA	result
0	89	82	0.2843	0.3999	0.3158	0.627539	1.65	4.0	6.0	1.65	...	1.62	3.6	6.0	1.67	4.0	5.75	1.64	4.07	6.04	1
1	73	71	0.3574	0.35	0.2926	0.876106	2.0	3.6	4.0	2.0	...	1.91	3.5	4.0	2.0	3.5	4.2	1.82	3.88	4.7	0
2	72	90	0.2988	0.4337	0.2675	0.604226	1.7	3.9	5.5	1.7	...	1.73	3.5	5.0	1.73	3.9	5.4	1.75	3.76	5.44	2
3	75	77	0.6422	0.2057	0.1521	2.56803	1.95	3.5	4.33	2.0	...	2.0	3.1	2.7	2.0	3.4	4.33	1.79	3.74	5.1	1
4	79	78	0.1461	0.2159	0.638	1.13076	2.55	3.3	3.0	2.6	...	2.6	3.1	2.88	2.6	3.25	3.0	2.46	3.39	3.14	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
12675	168	166	0.2812	0.2671	0.4517	1.41119	1.45	4.5	7.0	1.5	...	1.5	4.5	6.0	1.5	4.33	6.0	1.58	4.36	6.18	0
12676	177	176	0.3367	0.2999	0.3634	1.19819	2.55	2.87	3.25	2.45	...	2.5	2.9	3.2	2.4	3.1	3.0	2.66	3.28	2.93	0
12677	163	235	0.6719	0.2502	0.0779	1.33269	1.3	5.75	9.0	1.34	...	1.32	5.25	9.0	1.3	5.25	9.0	1.23	6.85	12.59	1
12678	175	181	0.3541	0.301	0.3449	1.4605	1.3	5.5	10.0	1.33	...	1.29	5.25	11.0	1.29	5.25	9.5	1.29	5.97	10.8	0
12679	225	179	0.1748	0.4863	0.3389	0.32396	2.6	2.0	4.33	2.75	...	2.62	2.2	4.33	2.63	2.2	3.25	2.69	2.3	4.18	2

هدف از انجام یادگیری ماشین انجام classification یا طبقه‌بندی است. ستون result ستون مورد پیشبینی است که می‌خواهیم مدلی آموزش دهیم تا با استفاده از داده‌های بالا بتواند نتیجه بازی را پیشبینی کند. سه کلاس مورد پیشبینی برد، باخت و مساوی هستند.

از الگوریتم جنگل درخت‌های تصمیم استفاده برای ساخت مدل classification استفاده می‌شود. از معیار entropy و حداکثر عمق ۵۰ و تعداد درخت‌های ۲۲۰ استفاده شد. دقت مدل پس از آموزش ۶۱.۳ درصد و میزان خطای MSE یا یانگین مربعات خطا برار با ۰.۷ می‌باشد.

در شکل زیر مراحل آموزش مدل، میزان دقت و خطای MSE و همچنین ۳ سطر شانسی، پیشبینی مدل و مقدار واقعی نتجیه‌ی بازی قابل مشاهده است. میبینیم که مدل فقط برای داده‌ی با شماره 8949 (تیم ۱۶۳ دربرابر ۱۶۶) اشتباها بازی را مساوی پیشبینی کرده ولی برای دو داده‌ی دیگر یعنی تیم ۲۰۸ در برابر ۱۴۶ به ترتیب مقدار صفر یعنی باخت تیم خانه (۲۰۸) و برای تیم ۱۱۷ در برابر ۱۳۲ برد تیم خانه یعنی ۱۱۷ را پیشبینی کرده است.

با بررسی داده‌های مربوط به تیم‌ها می‌توان دریافت تیم‌ها با id های ذکرشده‌ی بالا به شرح زیر هستند:

Rennes :۱۶۳

Montpellier:۱۶۶

Almeria:۲۰۸

Valencia:۱۴۶

Bayern Munich:۱۱۷

Eintracht Frankfurt:۱۳۲

```
model = RandomForestClassifier(criterion='entropy', max_depth=50, n_estimators=220, n_jobs=20)
model.fit(x_train, y_train)

print(f'Score : {model.score(x_test, y_test) * 100:.2f}%')
print('Mean Squared Error :', mean_squared_error(y_test, model.predict(x_test)))
```

```
Score : 61.30%
Mean Squared Error : 0.7010309278350515
```

```
x_test.iloc[:3,]
```

	homeTeamID	awayTeamID	homeProbability	drawProbability	awayProbability	homeGoalsProbability	B365H	B365D	B365A	BWH	...	PSA	WHH	WHD	WHA	VCH	VCD	VCA	PSCH	PSCD	PSCA
8949	163	166	0.4369	0.4152	0.1479	0.768295	2.05	3.4	3.8	2.1	...	3.98	2.05	3.2	3.9	2.1	3.2	4.0	2.12	3.32	3.95
4752	208	146	0.1544	0.2525	0.5931	1.38838	4.75	4.2	1.67	4.75	...	1.66	5.0	3.6	1.7	5.0	4.5	1.67	5.78	4.54	1.6
8741	117	132	0.9841	0.0142	0.0017	4.62869	1.25	6.5	10.0	1.25	...	11.22	1.25	6.0	12.0	1.25	6.5	11.5	1.31	5.75	10.18

3 rows × 27 columns

```
model.predict(x_test.iloc[:3,])
```

```
array([1, 0, 1])
```

```
y_test.iloc[:3,]
```

```
8949    2
4752    0
8741    1
Name: result, dtype: int64
```

در نهایت در شکل زیر میزان تاثیر هر ستون از داده‌های آموزش قابل مشاهده است: