DEVELOPMENT OF A DATA BASIS FOR THE PREDICTION OF FOOTBALL BETTING

PROJECT OVERVIEW

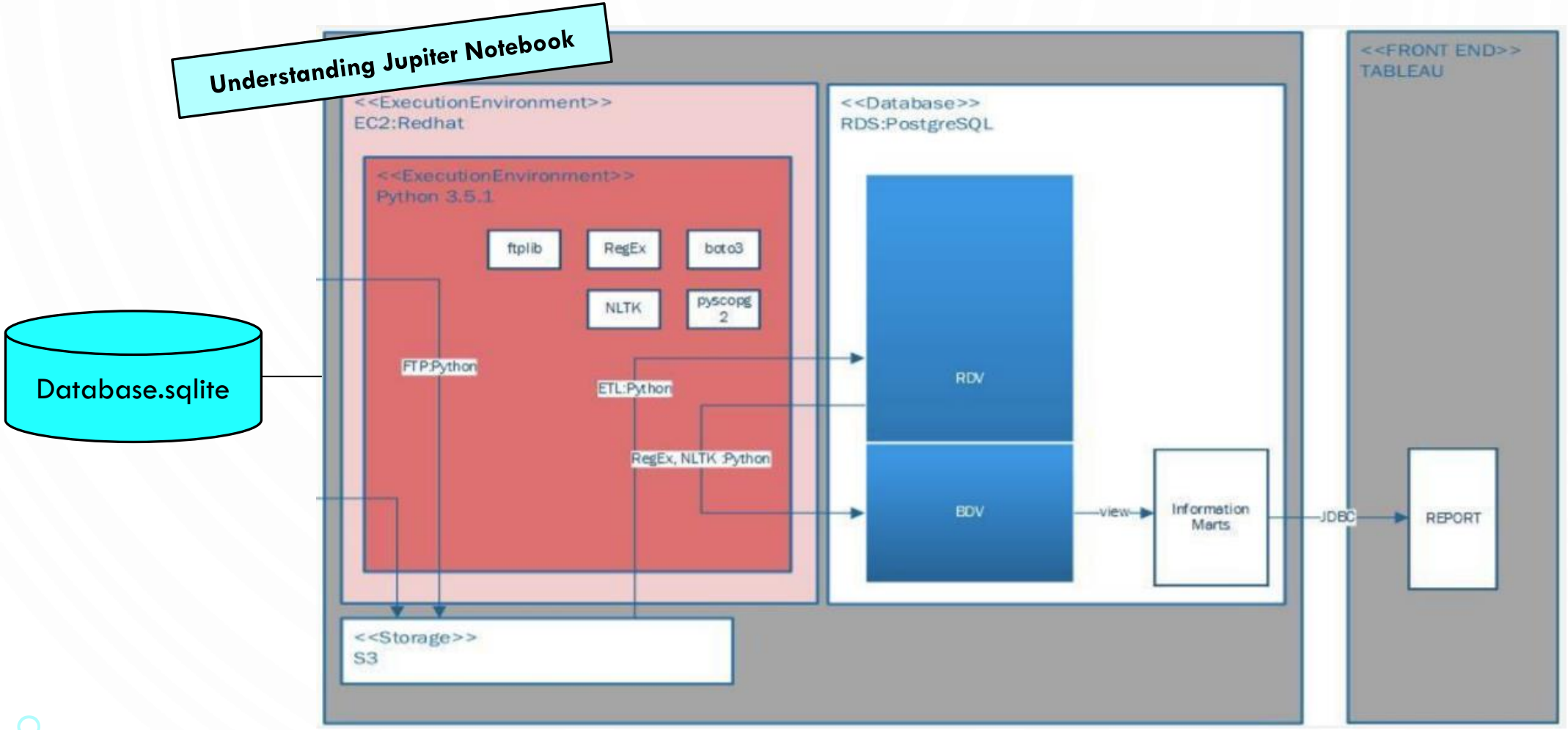
Project-Scope	Data Understanding	Data Model/ ETL Process	Use Case	Communication
• **Development ETL process** as basis for **soccer match forecasts,** which users can use against payment • **Goal:** Short-term liquidity protection of the company • **Period of time:** 3 days. Due to time, project focuses on a data extract • **Implementation in AWS**	• **Database consisted of 7 tables,** refers to 11 countries (1st league) • The **project focuses on 5 tables,** with emphasis on match and team tables • Within the tables the **focus is set on columns that are essential for the evaluation.** **Reason:** Time and budget constraints	• A **data vault schema is used** to ensure the expandability of the database or the forecast model • **Components:** 1 Hub, 1 Link and 12 Satellites	**Use Case** • **Implementation** of the **Home Advantage Approach** • **Forecast for victory of the home team,** because of home advantage • **Result:** Ranking list for clubs by league and season	• Presentation within the BDE course **Outlook:** • Completion of data model • Extension of the model by e.g. detailed attributes on player level • Evaluation of the forecast model during operation

PROJECT SCOPE

Customer Requirements	• **Development of ETL process** as basis for soccer match forecasts • **Users** should be able to **use the forecast model against payment** • **Goal:** Ensure Short-term liquidity of the company XYZ based on the above business model (within 6-12 months) • **Period of time:** 3 days. Due to time constraints, project focuses on data extraction (i.e. match and team tables) • **Next project phases:** Use more detailed attributes (e.g. player performance) • **Implementation in AWS**

Interview with Stakeholders	• **2 interviews with head of department**
	  • 1st discussion: First in-depth discussions on the project scope
	  • 2nd discussion: Approval of project
	
	• **2 Interview with Tech-Expert**
	  • 1st discussion: First version of the Data Vault model agreed, first detailed questions about data model
	  • 2nd discussion: Confirmation of Data Vault model. Reconciliation of hash generation and insert conditions

DATA UNDERSTANDING



DATA UNDERSTANDING | SOURCE DATABASE

Data Understanding

- **Database consisted of *7* tables,** refers to 11 countries (1st league)
- The **project focuses on 5 tables,** with emphasis on match and team tables
- Within the tables the **focus is set on columns that are essential for the evaluation.**
  Reason: Time and budget shortage

team_attributes
id
team_fifa_api_id
team_api_id
date
buildUpPlaySpeed
buildUpPlaySpeedClass
buildUpPlayDribbling
buildUpPlayDribblingClass
buildUpPlayPassing
buildUpPlayPassingClass
buildUpPlayPositioningClass
chanceCreationPassing
chanceCreationPassingClass
chanceCreationCrossing
chanceCreationCrossingClass
chanceCreationShooting
chanceCreationShootingClass
chanceCreationPositioningClass
defencePressure
defencePressureClass
defenceAggression
defenceAggressionClass
defenceTeamWidth
defenceTeamWidthClass
defenceDefenderLineClass

team
id
team_api_id
team_fifa_api_id
team_long_name
team_short_name

League
id
country_id
name

Country
id
name

Match
id
country_id
league_id
season
stage
date
match_api_id
home_team_api_id
away_team_api_id
home_team_goal
away_team_goal
…
…
B365H = Bet365 Heimsiegchancen
B365D = Bet365-Gewinnchancen
B365A = Bet365 Auswärtsgewinnchancen
BWH = Bet & Win Heimsiegquoten
BWD = Bet & Win Draw Odds
BWA = Bet & Win Away Gewinnchancen
IWH = Interwetten Heimsiegchancen
IWD = Interwetten Draw Odds
IWA = Interwetten Auswärtsgewinnchancen
…

DATA UNDERSTANDING | SOURCE DATABASE

```
# Show existing tables with number of columns
db = sqlite3.connect('./database.sqlite')

tables = pd.read_sql("""SELECT * FROM sqlite_master WHERE type='table'""", db)
tables.head
```

```
<bound method NDFrame.head of     type                name            tbl_name  rootpage  \
0  table     sqlite_sequence     sqlite_sequence          4
1  table  Player_Attributes  Player_Attributes         11
2  table              Player              Player         14
3  table               Match               Match         18
4  table              League              League         24
5  table             Country             Country         26
6  table                Team                Team         29
7  table     Team_Attributes     Team_Attributes          2

                                                 sql
0               CREATE TABLE sqlite_sequence(name,seq)
1  CREATE TABLE "Player_Attributes" (\n\t`id`\tIN...
2  CREATE TABLE `Player` (\n\t`id`\tINTEGER PRIMA...
3  CREATE TABLE `Match` (\n\t`id`\tINTEGER PRIMAR...
4  CREATE TABLE `League` (\n\t`id`\tINTEGER PRIMA...
5  CREATE TABLE `Country` (\n\t`id`\tINTEGER PRIM...
6  CREATE TABLE "Team" (\n\t`id`\tINTEGER PRIMARY...
7  CREATE TABLE `Team_Attributes` (\n\t`id`\tINTE...  >
```

DATA UNDERSTANDING | SOURCE DATABASE

Check for unique in columns "team_fifa_api_id" und "team_api_id"

```
df_team['team_fifa_api_id'].value_counts().head(5) # Bei gleichen "team_fifa_api_id" unterschiedli
```

```
111429.0    2
111560.0    2
301.0       2
100879.0    1
247.0       1
Name: team_fifa_api_id, dtype: int64
```

```
# bei gleichen long_name unterschiedliche short_name vorhanden ?
```

```
df_team[df_team['team_fifa_api_id'] == 111429] # Recodr ID=31445 muss
```

	id	team_api_id	team_fifa_api_id	team_long_name	team_short_name
182	31444	8031	111429.0	Polonia Bytom	POB
183	31445	8020	111429.0	Polonia Bytom	GOR

```
df_team['team_short_name'].value_counts().head(5)
```

```
MON    3
VAL    3
POR    3
BEL    3
GEN    3
Name: team_short_name, dtype: int64
```

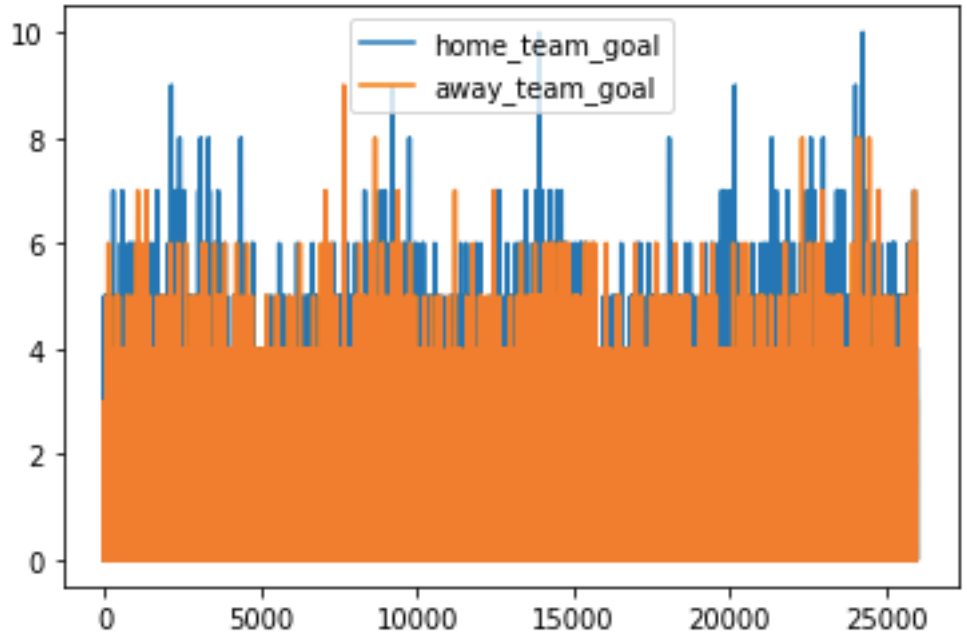
```
df_team[df_team['team_short_name'] == 'MON']
```

	id	team_api_id	team_fifa_api_id	team_long_name	team_short_name
7	8	9998	1747.0	RAEC Mons	MON
69	9547	9829	69.0	AS Monaco	MON
80	10309	10249	70.0	Montpellier Hérault SC	MON

DATA UNDERSTANDING | SOURCE DATABASE

```
df_m = pd.read_sql_query("SELECT id,country_id,league_id,season,stage,home_team_goal,away_team_goal FROM Match", db)
df_m.head(3)
```

	id	country_id	league_id	season	stage	home_team_goal	away_team_goal
0	1	1	1	2008/2009	1	1	1
1	2	1	1	2008/2009	1	0	0
2	3	1	1	2008/2009	1	0	3



DATA UNDERSTANDING | SOURCE DATABASE

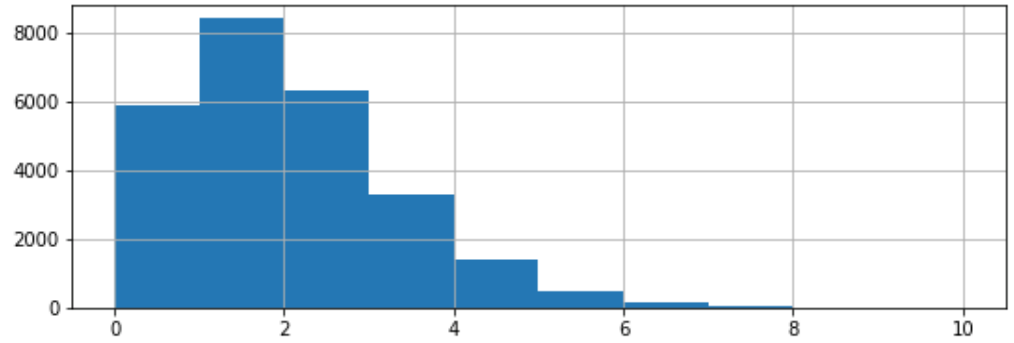
```
df_m.groupby('season')['home_team_goal'].mean()
```

```
season
2008/2009    1.505412
2009/2010    1.541176
2010/2011    1.548466
2011/2012    1.572671
2012/2013    1.550000
2013/2014    1.578826
2014/2015    1.520301
2015/2016    1.543897
Name: home_team_goal, dtype: float64
```

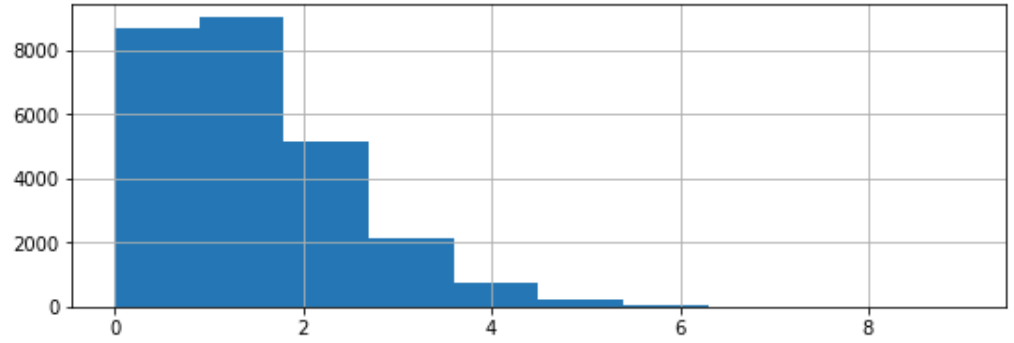
```
df_m.groupby('season')['away_team_goal'].mean()
```

```
season
2008/2009    1.101924
2009/2010    1.131269
2010/2011    1.135276
2011/2012    1.143789
2012/2013    1.222699
2013/2014    1.187995
2014/2015    1.155489
2015/2016    1.210764
Name: away_team_goal, dtype: float64
```

```
df_m['home_team_goal'].hist(figsize=(9,3));
```

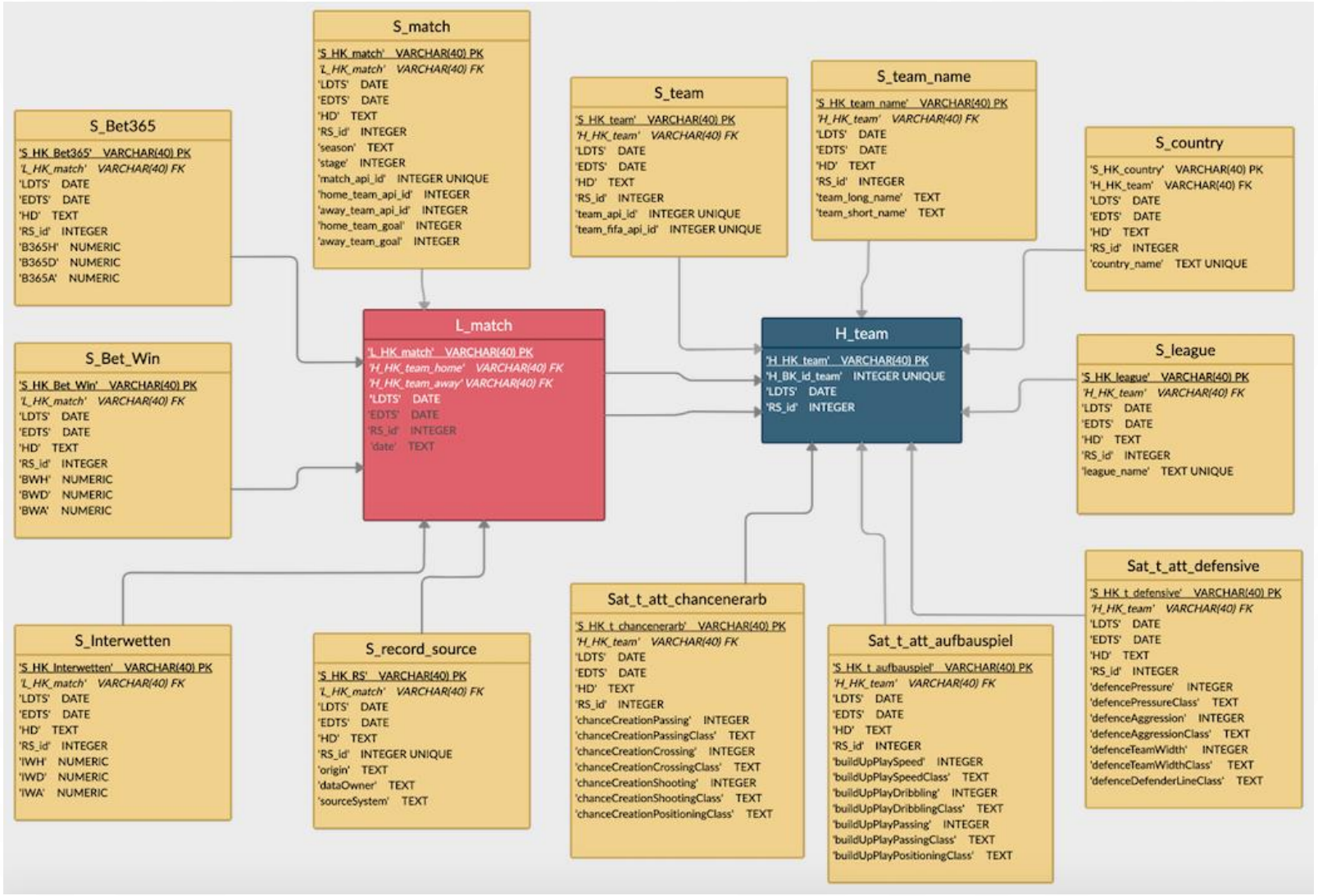



```
df_m['away_team_goal'].hist(figsize=(9,3));
```



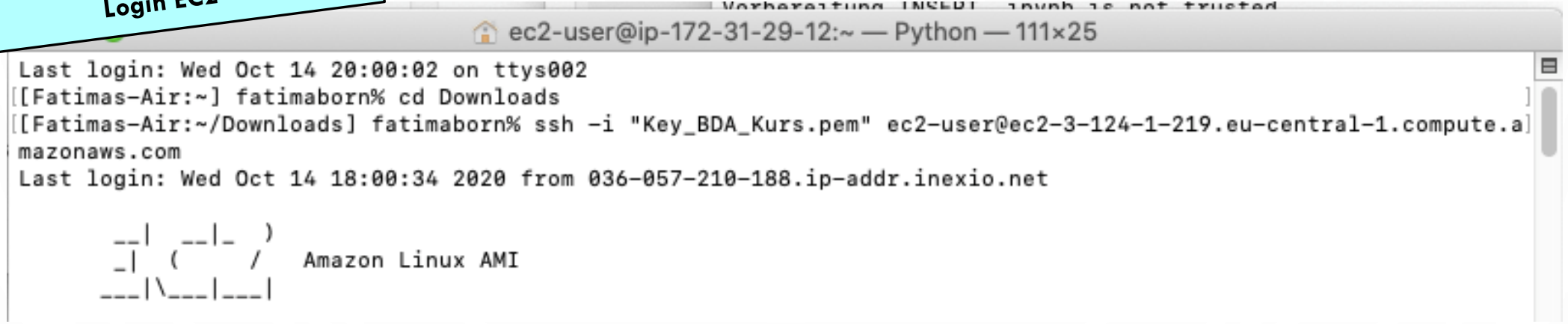
DATA VAULT MODELLING

Based on our approach, we keep the history of the data. Moreover, all data up to its source system are traceable.



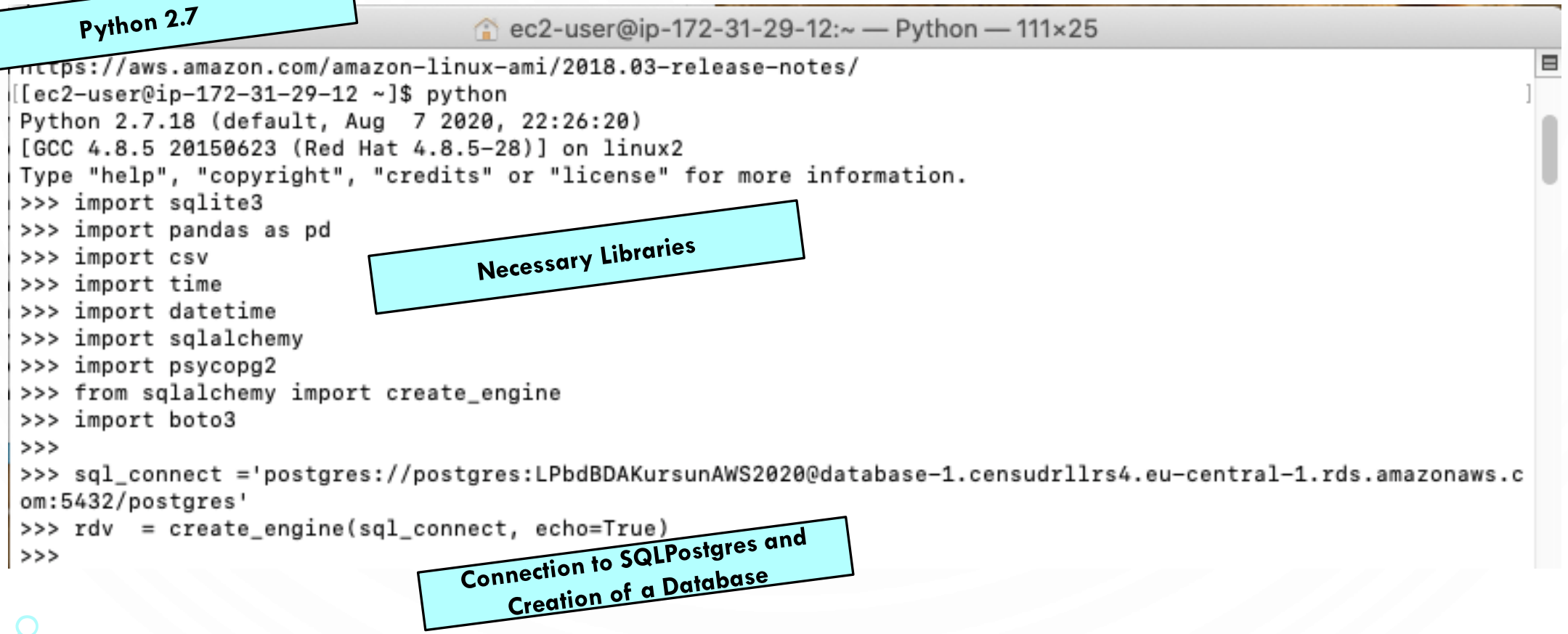
ETL-PROCESS | PREPARATION

Login EC2



ETL-PROCESS | PREPARATION

Python 2.7



Necessary Libraries

Connection to SQLPostgres and Creation of a Database

ETL-PROCESS | PREPARATION

Load Data into S3

```
ec2-user@ip-172-31-29-12:~ — ssh -i Key_BDA_Kurs.pem ec2-user@ec2-3-122-247-28.eu-central-1.compute.amazonaw...
>>> import boto3
>>> s3 = boto3.client('s3')
>>> response = s3.list_buckets()
>>> for bucket in response['Buckets']:
...     print(bucket)
...
{u'CreationDate': datetime.datetime(2020, 10, 8, 7, 2, 38, tzinfo=tzlocal()), u'Name': 'bucketbdakurs2020'}
>>> s3.download_file('bucketbdakurs2020','database.sqlite', ' database.sqlite ')
>>>
```

ETL-PROCESS | DATA CLEANSING

Show Data Frames

```
# Create the connection and create df for all tables
db = sqlite3.connect('./database.sqlite')

df_team = pd.read_sql_query("SELECT * FROM Team", db)
print(df_team.head())

db = sqlite3.connect('./database.sqlite')

df_team_attributes = pd.read_sql_query("SELECT * FROM Team_Attributes", db)
print(df_team_attributes.head())

db = sqlite3.connect('./database.sqlite')

df_match = pd.read_sql_query("SELECT * FROM Match WHERE season='2015/2016'", db)
print(df_match.head())

db = sqlite3.connect('./database.sqlite')

df_country = pd.read_sql_query("SELECT * FROM Country", db)
print(df_country.head())

db = sqlite3.connect('./database.sqlite')

df_league = pd.read_sql_query("SELECT * FROM League", db)
print(df_league.head())
```

Replace Null Values

```
# 1. CODE fuer Tabelle TEAM

df_team["team_fifa_api_id"] = df_team["team_fifa_api_id"].fillna(0)

# 2. CODE fuer Tabelle Team_Attributes

df_team_attributes["buildUpPlayDribbling"] = df_team_attributes["buildUpPlayDribbling"].fillna(9999)

# 3. CODE fuer Tabelle Match

df_match['B365H'].isna().sum()
df_match['B365H'] = df_match['B365H'].fillna(9999)

df_match['B365D'].isna().sum()
df_match['B365D'] = df_match['B365D'].fillna(9999)

df_match['B365A'].isna().sum()
df_match['B365A'] = df_match['B365A'].fillna(9999)

df_match['BWH'].isna().sum()
df_match['BWH'] = df_match['BWH'].fillna(9999)

df_match['BWD'].isna().sum()
df_match['BWD'] = df_match['BWD'].fillna(9999)

df_match['IWH'].isna().sum()
df_match['IWH'] = df_match['IWH'].fillna(9999)

df_match['IWD'].isna().sum()
df_match['IWD'] = df_match['IWD'].fillna(9999)

df_match['IWA'].isna().sum()
```

ETL-PROCESS | CREATE DATA VAULT

Create Data Vault Scheme

```
import sqlite3
import pandas as pd
import csv
import time
import datetime
import sqlalchemy
import psycopg2
from sqlalchemy import create_engine
import boto3

sql_connect ='postgres://postgres:LPbdBDAKursunAWS2020@database-1.censudrllrs4.eu-central-1.rds.amazonaws.com:5432/postgres'
rdv  = create_engine(sql_connect, echo=True)
# DATAVAULT ERSTELLEN
# Hub Team
rdv.execute("""DROP TABLE IF EXISTS HUB_Team; CREATE TABLE HUB_Team (

    HK_Hub_Team varchar(100) NOT NULL, H_BK_id_team INTEGER, LDTS varchar(20), record_source varchar(50), PRIMARY KEY (HK_Hub_Team)

    )""")

# Satellite Team
rdv.execute("""DROP TABLE IF EXISTS Link_Sales; CREATE TABLE Link_Sales (

    S_HK_team varchar(100) NOT NULL, H_HK_team varchar(100), LDTS varchar(100), EDTS varchar(100), HD varchar(100), record_source varchar(50),

    )""")
```

ETL-PROCESS | FILLING DATA VAULT

Create Hash Function

```
def create_hash(key_to_convert):

    seperator = ", "
    key_to_convert = seperator.join(map(str, key_to_convert))

    if key_to_convert == None: # Change NULLS to empty strings
        return("")
    else:
        key_to_convert_str = str(key_to_convert) # Convert all to String data type
        key_to_convert_no_spaces = key_to_convert_str.replace(' ', '') #.replace(',', '') # Trim all Fields, NON-Space delimiter between fields
        key_to_convert_capital_letters = key_to_convert_no_spaces.upper() # UPPERcase the string
        hash_object = hashlib.md5(key_to_convert_capital_letters.encode('utf-8'))
        #print(key_to_convert_capital_letters)
        #print(hash_object.hexdigest())

    return hash_object.hexdigest()
```

Filling Data Vault

```
HK_raw = []
for row in range(len(df_team)):
    RS = "1"
    HK_raw.append(df_team.loc[row,"id"])
    HK_raw.append(RS)
    HK = create_hash(HK_raw)
    BK = df_team.loc[row,"id"]
    ts = time.time()
    LDTS = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d %H:%M:%S')

    insert = "INSERT INTO HUB_Team(HK_Hub_Team,H_BK_id_team,LDTS,record_source) SELECT ('"+ str(HK)+ "')
    rdv.execute(insert)
    HK_raw = []
```

PREDICTION: HOME ADVANTAGE APPROACH (1)

Dummy Matrices

```
df_visitor = pd.get_dummies(df_prognose_bundesliga["away_team_api_id"], dtype=np.int64)
df_home = pd.get_dummies(df_prognose_bundesliga["home_team_api_id"], dtype=np.int64)
```

Subtract Home from Visitors and add Goal Difference

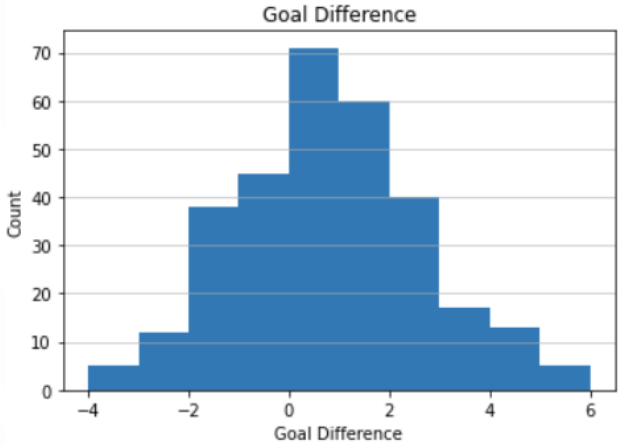
```
df_model = df_home.sub(df_visitor)
df_model['goal_difference'] = df_prognose['goal_difference‘]
```

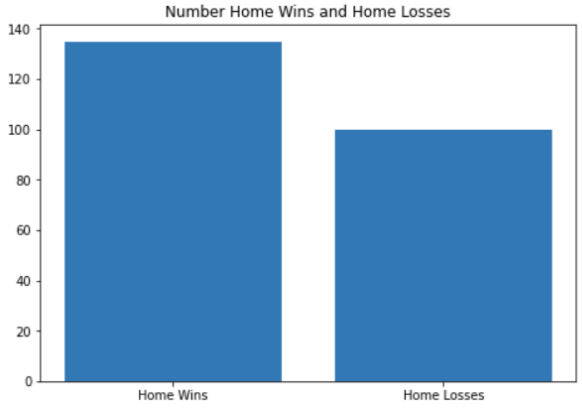
Apply Ridge Regression

```
df_train = df_model

lr = Ridge(alpha=0.001)
X = df_train.drop(['goal_difference'], axis=1)
y = df_train['goal_difference']

lr.fit(X, y)

df_ratings = pd.DataFrame(data={'Team': X.columns, 'Rating': lr.coef_})
df_ratings.astype(object)
```

TSG 1899 Hoffenheim	VfB Stuttgart	VfL Wolfsburg	goal_difference
0	0	0	5
0	0	0	4
-1	0	0	1
0	0	1	1
0	1	0	-2

PREDICTION: HOME ADVANTAGE APPROACH (2)





HEIMTABELLE

PL.	VEREIN	SP.	S	U	N	TORE	DIFF.	PUNKTE
1	Bayern München (M)	17	15	1	1	51:8	43	46
2	Borussia Dortmund	17	14	3	0	49:14	35	45
3	Bor. Mönchengladbach	17	13	1	3	42:18	24	40
4	Bayer 04 Leverkusen	17	10	3	4	31:17	14	33
5	VfL Wolfsburg (P)	17	9	5	3	32:17	15	32
6	Hertha BSC	17	9	5	3	24:15	9	32

	Team	Rating
0	1. FC Köln	-0.111108
1	1. FSV Mainz 05	0.111108
2	Bayer 04 Leverkusen	0.444432
3	Borussia Dortmund	1.3333
4	Borussia Mönchengladbach	0.472209
5	Eintracht Frankfurt	-0.499986
6	FC Augsburg	-0.27777
7	FC Bayern Munich	1.74995
8	FC Ingolstadt 04	-0.249993
9	FC Schalke 04	0.055554
10	Hamburger SV	-0.166662
11	Hannover 96	-0.861087
12	Hertha BSC Berlin	3.51989e-13
13	SV Darmstadt 98	-0.416655
14	SV Werder Bremen	-0.416655
15	TSG 1899 Hoffenheim	-0.416655
16	VfB Stuttgart	-0.694425
17	VfL Wolfsburg	-0.055554

CONCLUSION

Ambitious Real World Project

Goal:

- Short-term liquidity protection of the company XYZ based on identified business model (i.e. providing betting tool against payment). Planned ROI within 6-12 months

Challenge:

- Identify business case and set scope (i.e. focus on part of the data)

Outlook:

- Completion of the Data Vault Model
- Extension of the model by e.g. detailed attributes on player level
- Evaluation of the forecast model during operation