HR Analytics Absenteeism Prediction

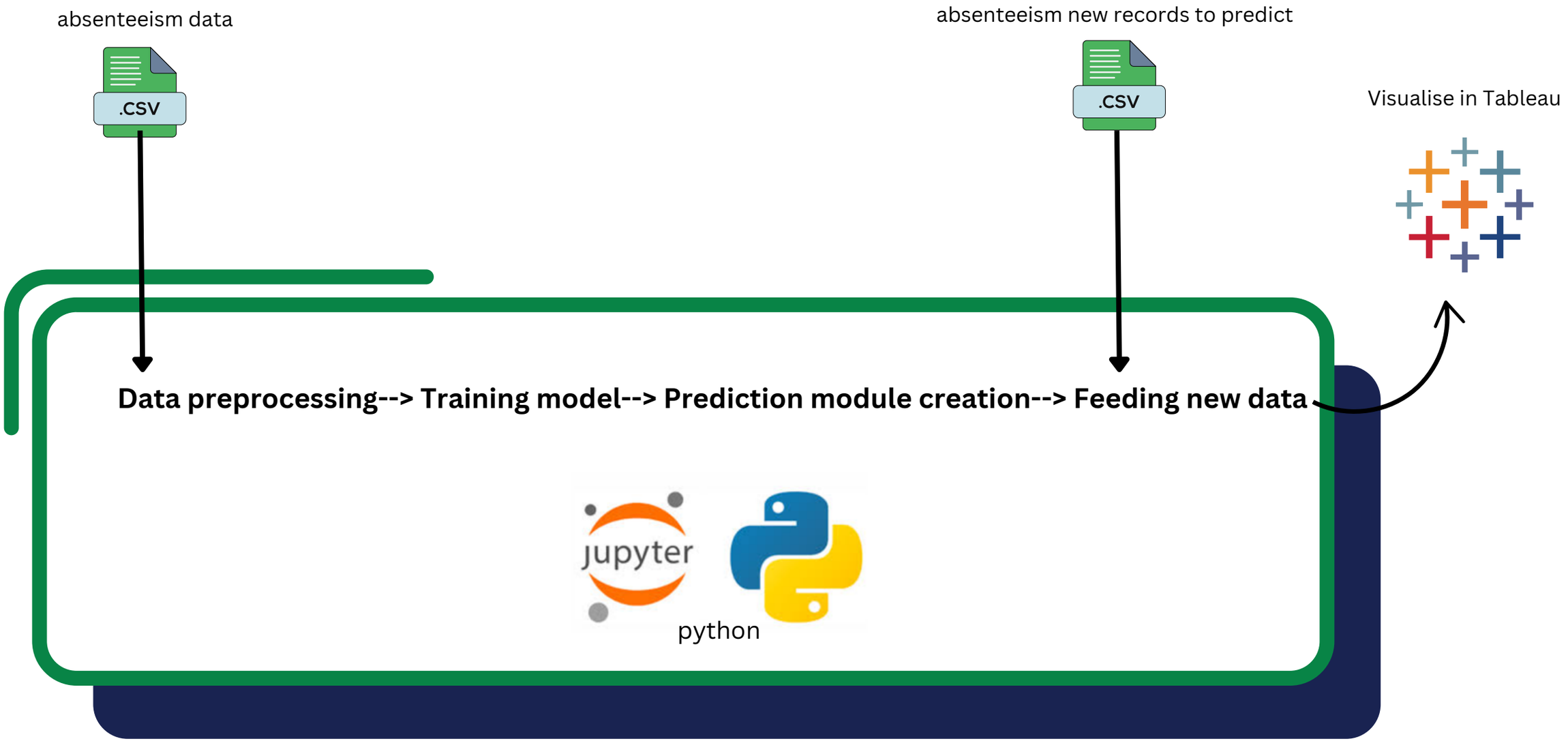
using Python and Tableau

This project’s objective:

Predicted excessive absenteeism at work for employee using a modified version of a multivariate time-series dataset from UCI machine learning repository.

Github

Full Project Architecture



Data Pre-processing Process using pandas

Objective (Data Preprocessing Part)

To design and implement a data preprocessing pipeline that cleans, transforms, and prepares the company's employee absenteeism dataset for predictive modeling. The pipeline will include handling missing data, feature engineering (including categorizing absence reasons), and scaling relevant features to ensure the data is optimized for a linear regression model to predict absenteeism time in hours.

Let’s understand the dataset first

Dataset is a multivariate time-series comma-separated value file (700 records) taken from UCI machine learning repository
Column 1: Unique Individual identification (ID)
Column 2: Reason for absence attested by the International Code of Diseases (ICD) stratified into 21 categories (I to XXI) as follows:

Category 0 : No reason stated
Category 1 to 14 : Serious issues
Category 15 to 17 : Pregnancy and childbirth related issues
Category 18 to 21 : Fatal issues
Category 22 to 28 : Light issues.

[1 Certain infectious and parasitic diseases 2 Neoplasms 3 Diseases of the blood and blood-forming organs and certain disorders involving the immune mechanism 4 Endocrine, nutritional and metabolic diseases 5 Mental and behavioural disorders 6 Diseases of the nervous system 7 Diseases of the eye and adnexa 8 Diseases of the ear and mastoid process 9 Diseases of the circulatory system 10 Diseases of the respiratory system 11 Diseases of the digestive system 12 Diseases of the skin and subcutaneous tissue 13 Diseases of the musculoskeletal system and connective tissue 14 Diseases of the genitourinary system 15 Pregnancy, childbirth and the puerperium 16 Certain conditions originating in the perinatal period 17 Congenital malformations, deformations and chromosomal abnormalities 18 Symptoms, signs and abnormal clinical and laboratory findings, not elsewhere classified 19 Injury, poisoning and certain other consequences of external causes 20 External causes of morbidity and mortality 21 Factors influencing health status and contact with health services 22 patient follow-up , 23 medical consultation , 24 blood donation , 25 laboratory examination , 26 unjustified absence , 27 physiotherapy , dental consultation.]

Column 3: Month of absence
Column 4: Day of the week (Monday (2), Tuesday (3), Wednesday (4), Thursday (5), Friday (6))
Column 5: Seasons (summer (1), autumn (2), winter (3), spring (4))
Column 6: Transportation expense
Column 7: Distance from Residence to Work (kilometers)
Column 8: Service time
Column 9: Age
Column 10: Work load Average/day
Column 11: Hit target
Column 12: Disciplinary failure (yes=1; no=0)
Column 13: Education (high school (1), graduate (2), postgraduate (3), master and doctor (4))
Column 14: Son (number of children)
Column 15: Social drinker (yes=1; no=0)
Column 16: Social smoker (yes=1; no=0)
Column 17: Pet (number of pet)
Column 18: Weight
Column 19: Height
Column 20: Body mass index
Column 21: Absenteeism time in hours (target)

Absenteeism_data.csv : 700 records

ID	Reason for Absence	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours
11	26	07/07/2015	289	36	33	239.554	30	1	2	1	4
36	0	14/07/2015	118	13	50	239.554	31	1	1	0	0
3	23	15/07/2015	179	51	38	239.554	31	1	0	0	2
7	7	16/07/2015	279	5	39	239.554	24	1	2	0	4
11	23	23/07/2015	289	36	33	239.554	30	1	2	1	2
3	23	10/07/2015	179	51	38	239.554	31	1	0	0	2
10	22	17/07/2015	361	52	28	239.554	27	1	1	4	8

Absenteeism_Preprocessing.ipynb

Step 1 of preprocessing > we copy the dataframe from raw csv file into a pandas dataframe object and then drop the ID column since this is just an index column and it does not contribute to absenteeism.

In [2]:

```
# This code involves the preprocessing of csv data used in this project
#Features of the raw dataset: Reason for Absence,Transportation Expense,Distance to Work,Age,
#Daily Work Load,Average,Body Mass Index,Education,Children,Pets,Absenteeism Time in Hours

import pandas as pd

# pandas defaults
pd.options.display.max_columns = 500
pd.options.display.max_rows = 1000

raw_csv_data = pd.read_csv("C:/Users/User/AppData/Local/Programs/Python/Python36/Scripts/Absenteism Project/Absenteeism_data

# type(raw_csv_data)
# raw_csv_data

df = raw_csv_data.copy()
```

```
df = df.drop(['ID'], axis = 1)

df
```

:

	Reason for Absence	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours
0	26	07/07/2015	289	36	33	239.554	30	1	2	1	4
1	0	14/07/2015	118	13	50	239.554	31	1	1	0	0
2	23	15/07/2015	179	51	38	239.554	31	1	0	0	2
3	7	16/07/2015	279	5	39	239.554	24	1	2	0	4

We have 28 unique values for reason for absence column as it should be.

In [51]:

```
########## 'Reason for Absence' ##################
##################################################

# df['Reason for Absence'].min()
# df['Reason for Absence'].max()
# pd.unique(df['Reason for Absence'])
# df['Reason for Absence'].unique()
# len(df['Reason for Absence'].unique())
sorted(df['Reason for Absence'].unique())
```

Out[51]:

```
[0,
 1,
 2,
 3,
 4,
 5,
 6,
 7,
 8,
 9,
 10,
 11,
 12,
 13,
 14,
 15,
 16,
 17,
 18,
 19,
 21,
 22,
 23,
 24,
 25,
 26,
 27,
 28]
```

Step 2 of preprocessing > We get 0 and 1 values to indicate reason for absence for each record.

In [52]:

```
########## '.get_dummies()' ######################
##################################################


reason_columns = pd.get_dummies(df['Reason for Absence'])
reason_columns

# reason_columns['check'] = reason_columns.sum(axis=1)
# # reason_columns

# # reason_columns['check'].sum(axis=0)
# # reason_columns['check'].unique()

# reason_columns = reason_columns.drop(['check'], axis = 1)
# # reason_columns

# reason_columns = pd.get_dummies(df['Reason for Absence'], drop_first = True)
# # reason_columns
```

Out[52]:

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	21	22	23	24	25	26	27	28
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
3	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

```
########## Group the Reasons for Absence##########
##################################################


# df.columns.values
# reason_columns.columns.values
df = df.drop(['Reason for Absence'], axis = 1)

# reason_columns.loc[:, 1:15].max(axis=1)
reason_type_1 = reason_columns.iloc[:, 1:15].max(axis=1)
reason_type_2 = reason_columns.iloc[:, 15:18].max(axis=1)
reason_type_3 = reason_columns.iloc[:, 18:22].max(axis=1)
reason_type_4 = reason_columns.iloc[:, 22:].max(axis=1)
```

```
########## Concatenate Column Values #############
##################################################

df = pd.concat([df, reason_type_1, reason_type_2, reason_type_3, reason_type_4], axis = 1)
# df

# df.columns.values
column_names = ['Date', 'Transportation Expense', 'Distance to Work', 'Age',
       'Daily Work Load Average', 'Body Mass Index', 'Education',
       'Children', 'Pets', 'Absenteeism Time in Hours', 'Reason_1', 'Reason_2', 'Reason_3', 'Reason_4']

df.columns = column_names
df.head()
```

	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Reason_1	Reason_2	Reason_3
0	07/07/2015	289	36	33	239.554	30	1	2	1	4	0	0	0
1	14/07/2015	118	13	50	239.554	31	1	1	0	0	0	0	0
2	15/07/2015	179	51	38	239.554	31	1	0	0	2	0	0	0
3	16/07/2015	279	5	39	239.554	24	1	2	0	4	1	0	0
4	23/07/2015	289	36	33	239.554	30	1	2	1	2	0	0	0

```
########## Reorder Columns #######################
##################################################


column_names_reordered = ['Reason_1', 'Reason_2', 'Reason_3', 'Reason_4',
                          'Date', 'Transportation Expense', 'Distance to Work', 'Age',
       'Daily Work Load Average', 'Body Mass Index', 'Education',
       'Children', 'Pets', 'Absenteeism Time in Hours']

df = df[column_names_reordered]
df.head()
```

	Reason_1	Reason_2	Reason_3	Reason_4	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Ab
0	0	0	0	1	07/07/2015	289	36	33	239.554	30	1	2	1	
1	0	0	0	0	14/07/2015	118	13	50	239.554	31	1	1	0	
2	0	0	0	1	15/07/2015	179	51	38	239.554	31	1	0	0	
3	1	0	0	0	16/07/2015	279	5	39	239.554	24	1	2	0	
4	0	0	0	1	23/07/2015	289	36	33	239.554	30	1	2	1	

The ‘Date’ column must be segregated into ‘month’ and ‘day of the week’

In [70]:
```
list_months=[]
for i in range(0,700):
    list_months.append(df_reason_mod['Date'][i].month)
```

```
df_reason_mod['Month Value']= list_months
```

```
df_reason_mod
```

Reason_2	Reason_3	Reason_4	Date	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Month Value
0	0	1	2015-07-07	289	36	33	239.554	30	1	2	1	4	7
0	0	0	2015-07-14	118	13	50	239.554	31	1	1	0	0	7
0	0	1	2015-07-15	179	51	38	239.554	31	1	0	0	2	7
0	0	0	2015-07-16	279	5	39	239.554	24	1	2	0	4	7

For the eduction records: 1 = high school, 2 = graduate, 3 = post graduate, 4 = doctorate

In [80]:

```
def date_to_weekday(date_value):
    return date_value.weekday()
```

In [81]:

```
df_reason_mod['Day of the Week'] = df_reason_mod['Date'].apply(date_to_weekday)
```

In [85]:

```
df_reason_mod.head(46)
```

Out[85]:

Reason_2	Reason_3	Reason_4	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours	Month Value	Day of the Week
0	0	1	289	36	33	239.554	30	1	2	1	4	7	1
0	0	0	118	13	50	239.554	31	1	1	0	0	7	1
0	0	1	179	51	38	239.554	31	1	0	0	2	7	2
0	0	0	279	5	39	239.554	24	1	2	0	4	7	3
0	0	1	289	36	33	239.554	30	1	2	1	2	7	3
0	0	1	179	51	38	239.554	31	1	0	0	2	7	4
0	1	0	361	52	28	239.554	27	1	1	4	8	7	4
0	0	1	260	50	36	239.554	23	1	4	0	4	7	4
0	1	0	155	12	34	239.554	25	1	2	0	40	7	0

Shuffle the columns

In [83]:
```
df_reason_mod = df_reason_mod.drop(['Date'], axis=1)
df_reason_mod.columns
```

Out[83]:
```
Index(['Reason_1', 'Reason_2', 'Reason_3', 'Reason_4',
       'Transportation Expense', 'Distance to Work', 'Age',
       'Daily Work Load Average', 'Body Mass Index', 'Education', 'Children',
       'Pets', 'Absenteeism Time in Hours', 'Month Value', 'Day of the Week'],
      dtype='object')
```

In [126…
```
date_columns_mod= ['Reason_1', 'Reason_2', 'Reason_3', 'Reason_4', 'Month Value', 'Day of the Week',
       'Transportation Expense', 'Distance to Work', 'Age',
       'Daily Work Load Average', 'Body Mass Index', 'Education', 'Children',
       'Pets', 'Absenteeism Time in Hours']
df_reason_date_mod = df_reason_mod[date_columns_mod]
df_reason_date_mod
```

Out[126…

	Reason_1	Reason_2	Reason_3	Reason_4	Month Value	Day of the Week	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pet
0	0	0	0	1	7	1	289	36	33	239.554	30	1	2	
1	0	0	0	0	7	1	118	13	50	239.554	31	1	1	
2	0	0	0	1	7	2	179	51	38	239.554	31	1	0	
3	1	0	0	0	7	3	279	5	39	239.554	24	1	2	
4	0	0	0	1	7	3	289	36	33	239.554	30	1	2	
5	0	0	0	1	7	4	179	51	38	239.554	31	1	0	
6	0	0	1	0	7	4	361	52	28	239.554	27	1	1	
7	0	0	0	1	7	4	260	50	36	239.554	23	1	4	

We have all the columns with numeric values, ready for next stage input to train supervised learning model.

final note on pre-processing

In [133…

```
df_preprocessed = df_reason_date_mod.copy()
df_preprocessed.head(10)
```

Out[133…

Reason_2	Reason_3	Reason_4	Month Value	Day of the Week	Transportation Expense	Distance to Work	Age	Daily Work Load Average	Body Mass Index	Education	Children	Pets	Absenteeism Time in Hours
0	0	1	7	1	289	36	33	239.554	30	0	2	1	4
0	0	0	7	1	118	13	50	239.554	31	0	1	0	0
0	0	1	7	2	179	51	38	239.554	31	0	0	0	2
0	0	0	7	3	279	5	39	239.554	24	0	2	0	4
0	0	1	7	3	289	36	33	239.554	30	0	2	1	2
0	0	1	7	4	179	51	38	239.554	31	0	0	0	2
0	1	0	7	4	361	52	28	239.554	27	0	1	4	8
0	0	1	7	4	260	50	36	239.554	23	0	4	0	4
0	1	0	7	0	155	12	34	239.554	25	0	2	0	40
0	1	0	7	0	235	11	37	239.554	29	1	1	1	8

In [134…

```
df_preprocessed.to_csv('Absenteeism_preprocessed.csv', index=False)
```

Final Dashboard after model training and new data prediction of absenteeism probability.