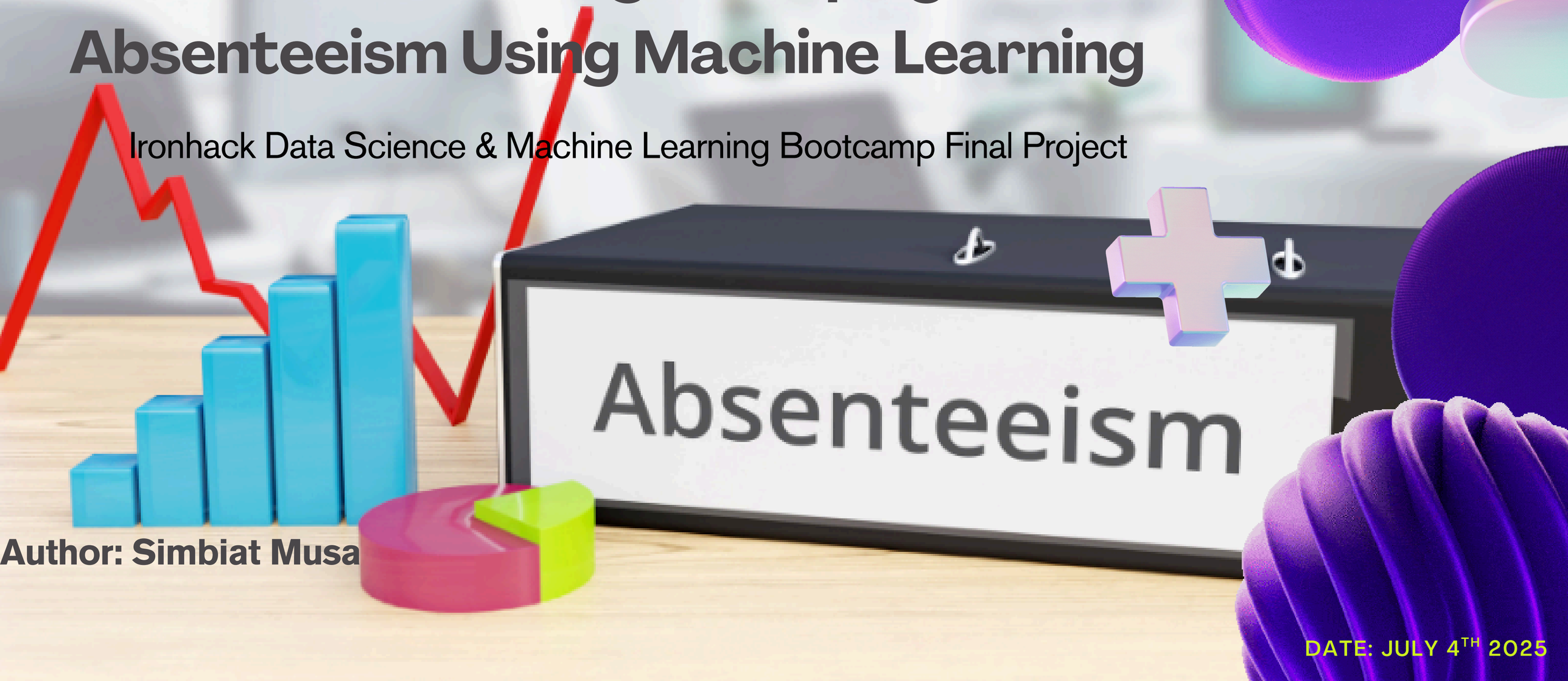
Predictive Modeling of Employee Absenteeism Using Machine Learning

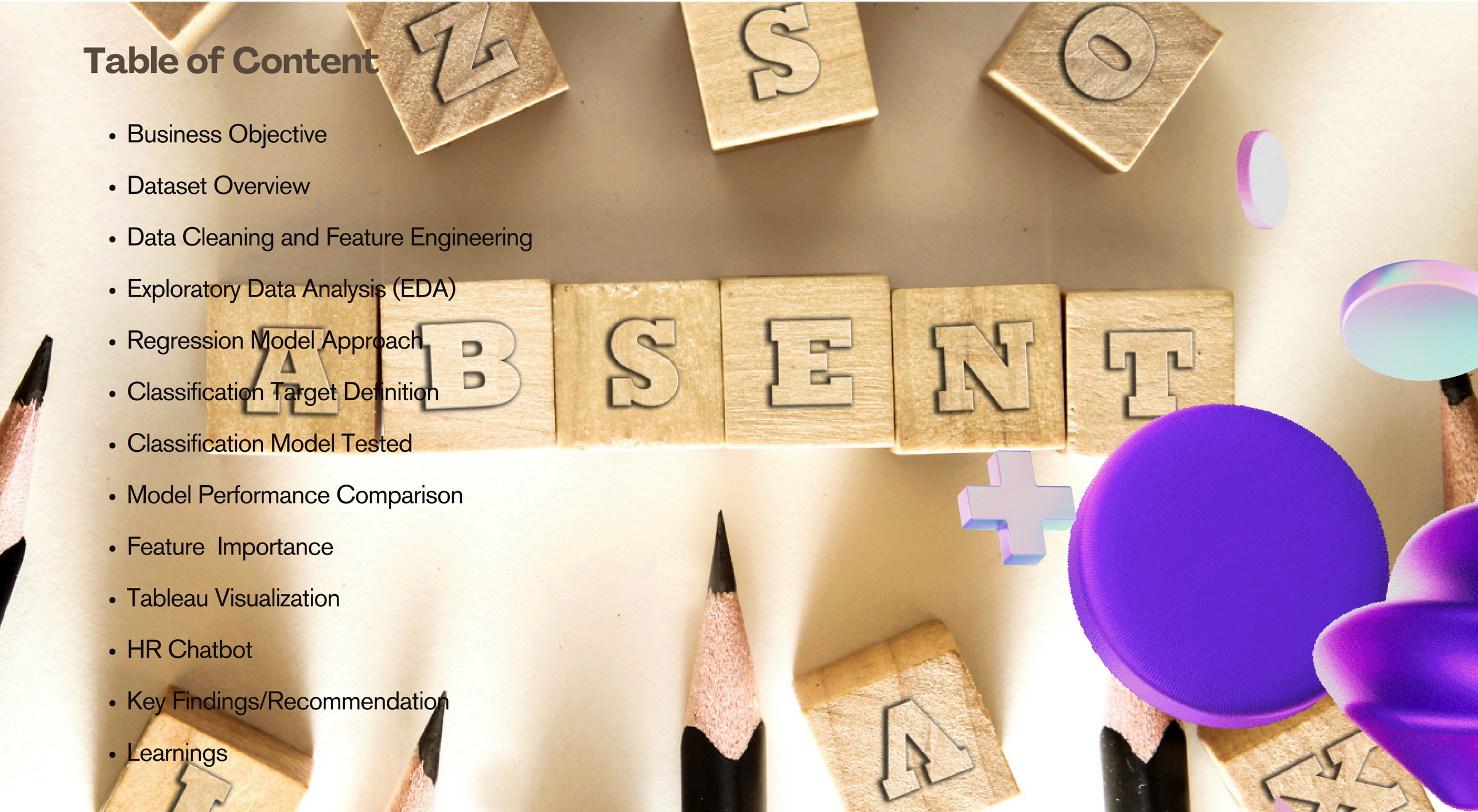
Ironhack Data Science & Machine Learning Bootcamp Final Project



Author: Simbiat Musa

DATE: JULY 4TH 2025

Table of Content



Business Objective

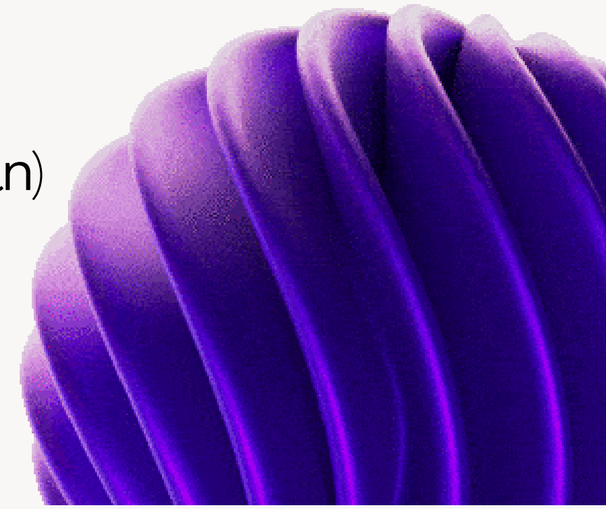
The business aim is to help HR departments proactively manage absenteeism.

Goal: Predict whether an employee will have High or Low absenteeism based on various features.

This assists in identifying key risk factors and implementing preemptive HR strategies.

Dataset Overview

- Source: Kaggle (Real-world HR records from a courier company in Brazil)
- Dataset: Employee Absenteeism
- Total Records: 740
- Features: 21
- Includes personal (age, BMI), work (seasons, workload), and health-related (reason for absence) attributes.
- Target Variable:
  - Initially: Number of absenteeism hours (regression)
  - Transformed: High (above median) or Low (below or equal to median) absenteeism (classification)



Data Cleaning and Feature Engineering

Removed irrelevant columns (e.g., Weight, Height had low correlation)

Grouped similar reasons for absence into categories (Reason group)

Encoded categorical variables (e.g., Seasons, Day of week)

Scaled numerical features using StandardScaler

Created binary target based on median absenteeism hours

```
data.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 740 entries, 0 to 739
Data columns (total 21 columns):
 #   Column                           Non-Null Count  Dtype  
---  ------                           --------------  -----  
 0   ID                               740 non-null    int64  
 1   Reason for absence               737 non-null    float64
 2   Month of absence                 739 non-null    float64
 3   Day of the week                  740 non-null    int64  
 4   Seasons                          740 non-null    int64  
 5   Transportation expense           733 non-null    float64
 6   Distance from Residence to Work  737 non-null    float64
 7   Service time                     737 non-null    float64
 8   Age                              737 non-null    float64
 9   Work load Average/day            730 non-null    object 
 10  Hit target                       734 non-null    float64
 11  Disciplinary failure             734 non-null    float64
 12  Education                        730 non-null    float64
 13  Son                              734 non-null    float64
 14  Social drinker                   737 non-null    float64
 15  Social smoker                    736 non-null    float64
 16  Pet                              738 non-null    float64
 17  Weight                           739 non-null    float64
 18  Height                           726 non-null    float64
 19  Body mass index                  709 non-null    float64
 20  Absenteeism time in hours        718 non-null    float64
dtypes: float64(17), int64(3), object(1)
memory usage: 121.5+ KB
```

Imbalanced Dataset

```
data.isnull().sum().sort_values(ascending=False)
```

[10] ✓ 0.0s

```
Body mass index                  31
Absenteeism time in hours        22
Height                           14
Work load Average/day            10
Education                        10
Transportation expense            7
Son                               6
Disciplinary failure              6
Hit target                        6
Social smoker                     4
Age                               3
Reason for absence                3
Service time                      3
Distance from Residence to Work   3
Social drinker                    3
Pet                               2
Weight                            1
Month of absence                  1
Seasons                           0
Day of the week                   0
ID                                0
dtype: int64
```

```
# Drop Missing Target Rows; dropped rows where the target column (Absenteeism time in hours) is missing, otherwise i can't train the model.
data = data.dropna(subset=['Absenteeism time in hours'])

# Confirm the drop worked
data['Absenteeism time in hours'].isnull().sum()
```

[11] ✓ 0.0s

```
0
```

```
numerical_cols = data.select_dtypes(include=['int64', 'float64']).columns.tolist()
categorical_cols = data.select_dtypes(include=['object']).columns.tolist()
# Print the numerical and categorical columns
print("Numerical Columns:", numerical_cols)
print("Categorical Columns:", categorical_cols)
```

2] ✓ 0.0s

```
Numerical Columns: ['ID', 'Reason for absence', 'Month of absence', 'Day of the week', 'Seasons', 'Transportation expense
Categorical Columns: []
```

```
# Separate true categorical columns from numerical ones
cat_cols = [
    'Reason for absence',        # 0–28
    'Month of absence',          # 1–12
    'Day of the week',           # 2–6
    'Seasons',                   # 1–4
    'Education',                 # 1–4
    'Disciplinary failure',      # 0–1
    'Social drinker',            # 0–1
    'Social smoker',             # 0–1
    'Pet',                       # Discrete count
    'Son'                        # Discrete count
]
```

3] ✓ 0.0s

```
# Separate numerical vs categorical columns for EDA
numerical_cols = [col for col in data.columns if col not in cat_cols and col != 'Absenteeism time in hours']
categorical_cols = cat_cols
target_col = 'Absenteeism time in hours'

print("Numerical Columns:", numerical_cols)
print("Categorical Columns:", categorical_cols)
```

FEATURE ENGINEERING

```python
# Group “Reason for Absence” into Categories
# Instead of 28+ individual reasons, I’ll group them into 4 main categories based on UCI’s metadata
def reason_group(reason_code):
    if 1 <= reason_code <= 14:
        return 1  # Disease-related
    elif 15 <= reason_code <= 17:
        return 2  # Pregnancy / Family
    elif 18 <= reason_code <= 21:
        return 3  # Work / Travel
    elif 22 <= reason_code <= 28:
        return 4  # Other
    else:
        return 0  # No absence (code 0)

# Create a new column
data['Reason_group'] = data['Reason for absence'].apply(reason_group)

# Check result
data[['Reason for absence', 'Reason_group']].head()
```

✓ 0.0s

	Reason for absence	Reason_group
0	26.0	4
1	0.0	0
2	23.0	4
3	7.0	1
4	23.0	4

```python
# Create a Risk Category for Absenteeism
# Classify absentee hours into Low, Medium, and High categories.
# This is to enable classification modeling in addition to regression.
def risk_level(hours):
    if hours <= 8:
        return 'Low'
    elif hours <= 16:
        return 'Medium'
    else:
        return 'High'

data['Absenteeism_level'] = data['Absenteeism time in hours'].apply(risk_level)

# Check value counts
data['Absenteeism_level'].value_counts()
```

✓ 0.0s

```
Absenteeism_level
Low       656
High       43
Medium     19
Name: count, dtype: int64
```

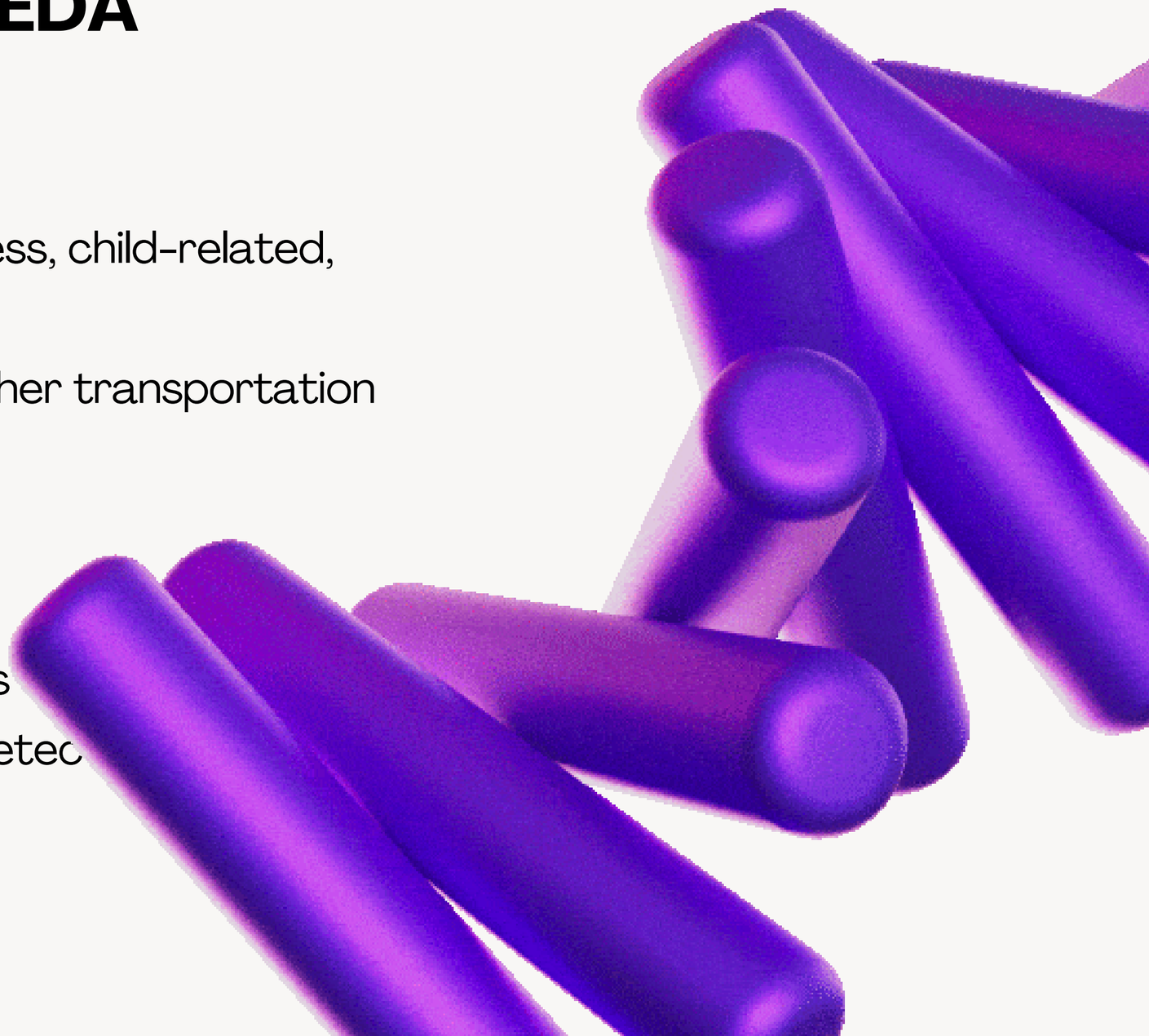
EDA

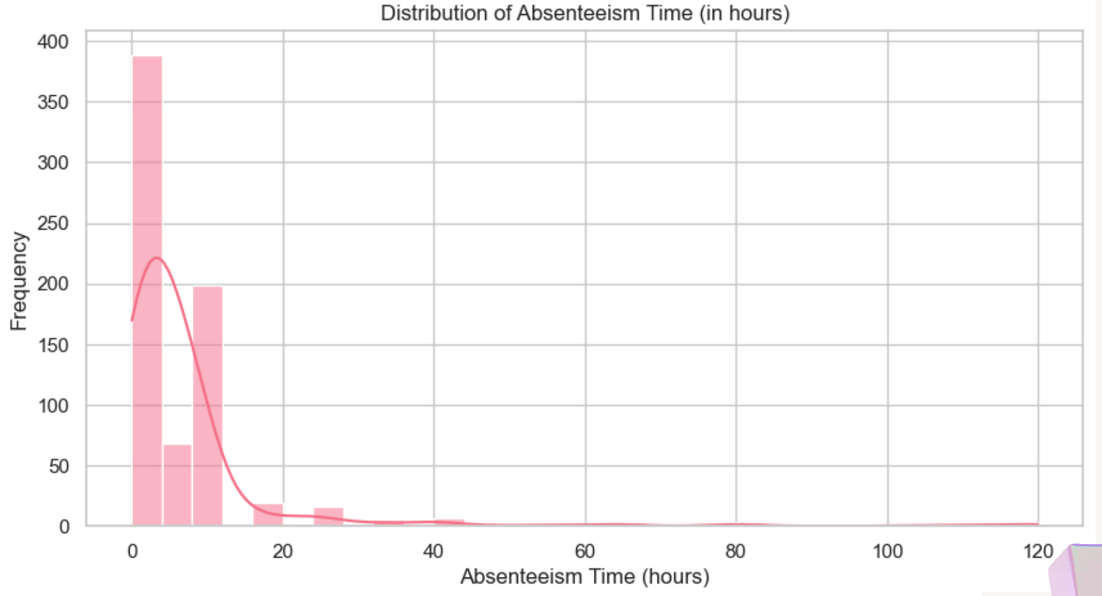
Findings:

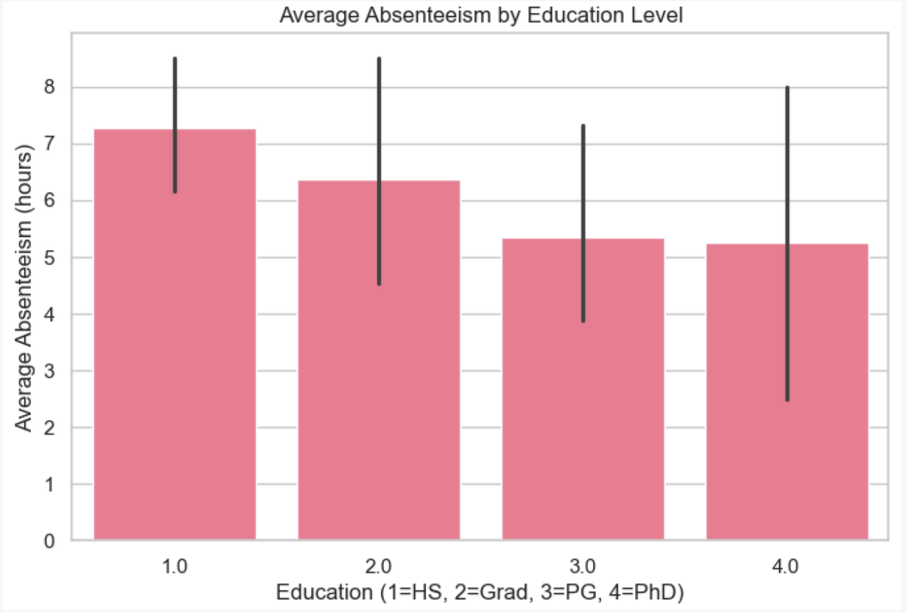
- Top reasons for absence: Sickness, child-related, dental
- High absenteeism linked with higher transportation cost and long distances

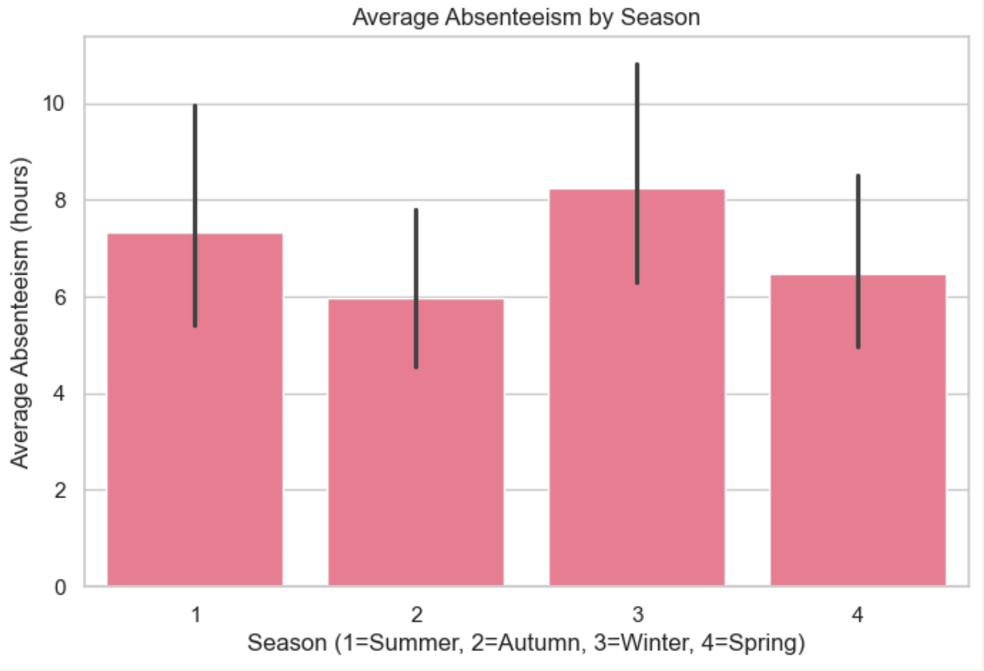
Visual Tools Used:

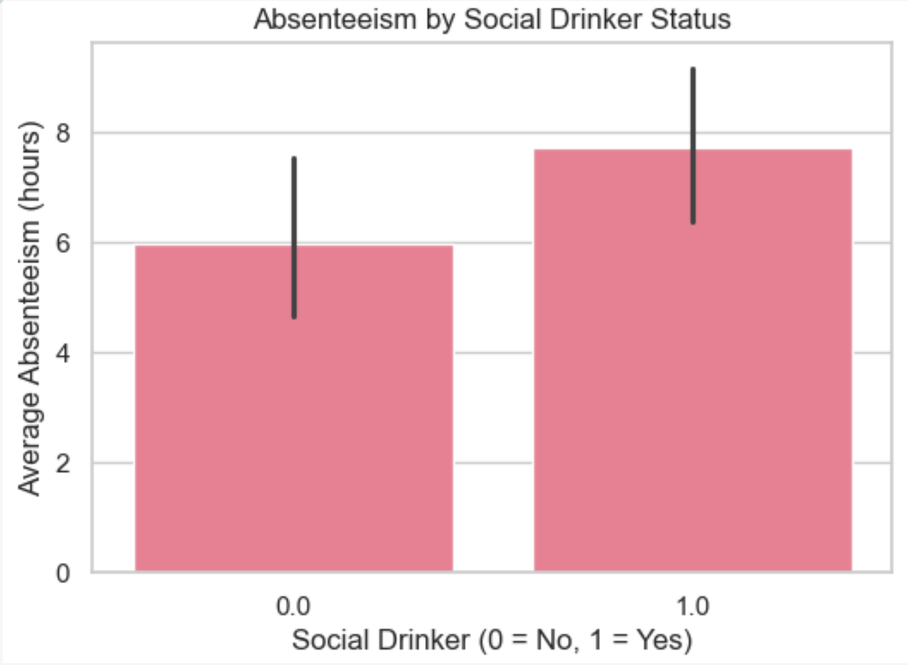
- Histograms, Heatmaps, Boxplots
- Pearson correlation matrix to detec multicollinearity

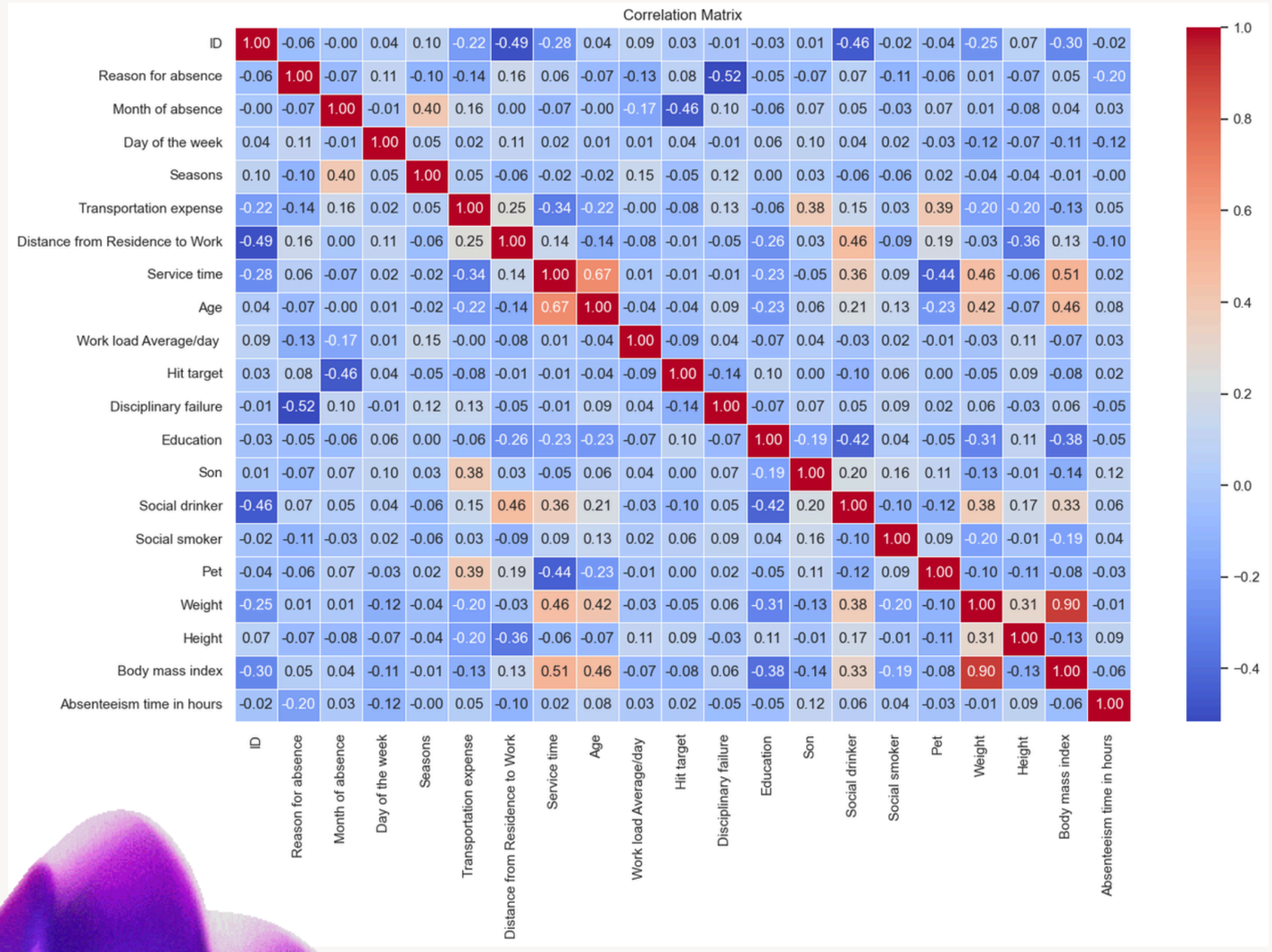


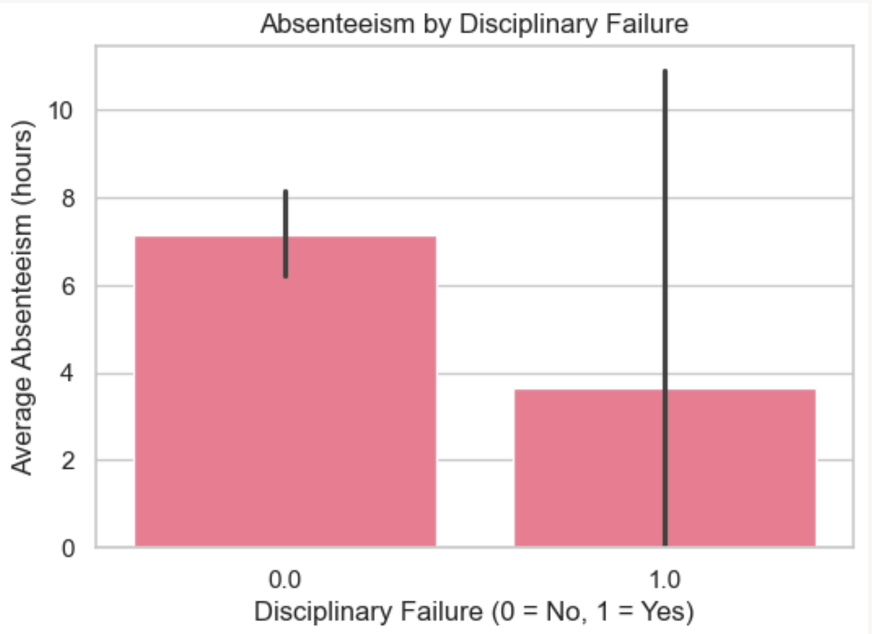

Distribution of Absenteeism Time (in hours)
Frequency
Absenteeism Time (hours)
0
50
100
150
200
250
300
350
400
0
20
40
60
80
100
120

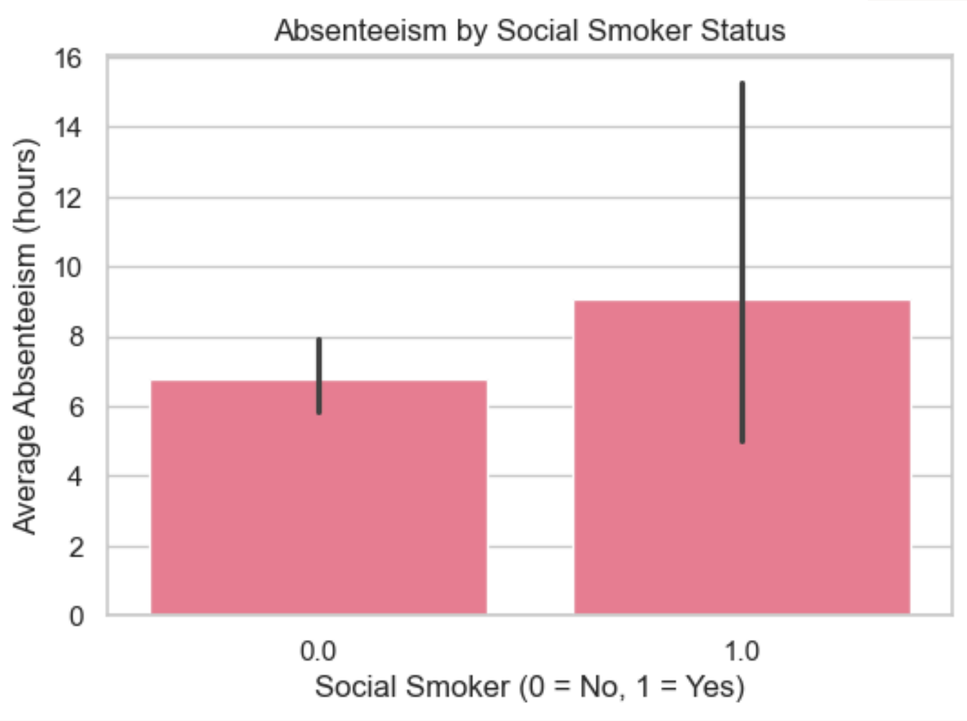

Average Absenteeism by Education Level
Average Absenteeism (hours)
Education (1=HS, 2=Grad, 3=PG, 4=PhD)
0
1
2
3
4
5
6
7
8
1.0
2.0
3.0
4.0


Average Absenteeism by Season
Average Absenteeism (hours)
Season (1=Summer, 2=Autumn, 3=Winter, 4=Spring)
0
2
4
6
8
10
1
2
3
4


Absenteeism by Social Drinker Status
Average Absenteeism (hours)
Social Drinker (0 = No, 1 = Yes)
0
2
4
6
8
0.0
1.0


Correlation Matrix
ID 1.00 -0.06 -0.00 0.04 0.10 -0.22 -0.49 -0.28 0.04 0.09 0.03 -0.01 -0.03 0.01 -0.46 -0.02 -0.04 -0.25 0.07 -0.30 -0.02
Reason for absence -0.06 1.00 -0.07 0.11 -0.10 -0.14 0.16 0.06 -0.07 -0.13 0.08 -0.52 -0.05 -0.07 0.07 -0.11 -0.06 0.01 -0.07 0.05 -0.20
Month of absence -0.00 -0.07 1.00 -0.01 0.40 0.16 0.00 -0.07 -0.00 -0.17 -0.46 0.10 -0.06 0.07 0.05 -0.03 0.07 0.01 -0.08 0.04 0.03
Day of the week 0.04 0.11 -0.01 1.00 0.05 0.02 0.11 0.02 0.01 0.01 0.04 -0.01 0.06 0.10 0.04 0.02 -0.03 -0.12 -0.07 -0.11 -0.12
Seasons 0.10 -0.10 0.40 0.05 1.00 0.05 -0.06 -0.02 -0.02 0.15 -0.05 0.12 0.00 0.03 -0.06 -0.06 0.02 -0.04 -0.04 -0.01 -0.00
Transportation expense -0.22 -0.14 0.16 0.02 0.05 1.00 0.25 -0.34 -0.22 -0.00 -0.08 0.13 -0.06 0.38 0.15 0.03 0.39 -0.20 -0.20 -0.13 0.05
Distance from Residence to Work -0.49 0.16 0.00 0.11 -0.06 0.25 1.00 0.14 -0.14 -0.08 -0.01 -0.05 -0.26 0.03 0.46 -0.09 0.19 -0.03 -0.36 0.13 -0.10
Service time -0.28 0.06 -0.07 0.02 -0.02 -0.34 0.14 1.00 0.67 0.01 -0.01 -0.01 -0.23 -0.05 0.36 0.09 -0.44 0.46 -0.06 0.51 0.02
Age 0.04 -0.07 -0.00 0.01 -0.02 -0.22 -0.14 0.67 1.00 -0.04 -0.04 0.09 -0.23 0.06 0.21 0.13 -0.23 0.42 -0.07 0.46 0.08
Work load Average/day 0.09 -0.13 -0.17 0.01 0.15 -0.00 -0.08 0.01 -0.04 1.00 -0.09 0.04 -0.07 0.04 -0.03 0.02 -0.01 -0.03 0.11 -0.07 0.03
Hit target 0.03 0.08 -0.46 0.04 -0.05 -0.08 -0.01 -0.01 -0.04 -0.09 1.00 -0.14 0.10 0.00 -0.10 0.06 0.00 -0.05 0.09 -0.08 0.02
Disciplinary failure -0.01 -0.52 0.10 -0.01 0.12 0.13 -0.05 -0.01 0.09 0.04 -0.14 1.00 -0.07 0.07 0.05 0.09 0.02 0.06 -0.03 0.06 -0.05
Education -0.03 -0.05 -0.06 0.06 0.00 -0.06 -0.26 -0.23 -0.23 -0.07 0.10 -0.07 1.00 -0.19 -0.42 0.04 -0.05 -0.31 0.11 -0.38 -0.05
Son 0.01 -0.07 0.07 0.10 0.03 0.38 0.03 -0.05 0.06 0.04 0.00 0.07 -0.19 1.00 0.20 0.16 0.11 -0.13 -0.01 -0.14 0.12
Social drinker -0.46 0.07 0.05 0.04 -0.06 0.15 0.46 0.36 0.21 -0.03 -0.10 0.05 -0.42 0.20 1.00 -0.10 -0.12 0.38 0.17 0.33 0.06
Social smoker -0.02 -0.11 -0.03 0.02 -0.06 0.03 -0.09 0.09 0.13 0.02 0.06 0.09 0.04 0.16 -0.10 1.00 0.09 -0.20 -0.01 -0.19 0.04
Pet -0.04 -0.06 0.07 -0.03 0.02 0.39 0.19 -0.44 -0.23 -0.01 0.00 0.02 -0.05 0.11 -0.12 0.09 1.00 -0.10 -0.11 -0.08 -0.03
Weight -0.25 0.01 0.01 -0.12 -0.04 -0.20 -0.03 0.46 0.42 -0.03 -0.05 0.06 -0.31 -0.13 0.38 -0.20 -0.10 1.00 0.31 0.90 -0.01
Height 0.07 -0.07 -0.08 -0.07 -0.04 -0.20 -0.36 -0.06 -0.07 0.11 0.09 -0.03 0.11 -0.01 0.17 -0.01 -0.11 0.31 1.00 -0.13 0.09
Body mass index -0.30 0.05 0.04 -0.11 -0.01 -0.13 0.13 0.51 0.46 -0.07 -0.08 0.06 -0.38 -0.14 0.33 -0.19 -0.08 0.90 -0.13 1.00 -0.06
Absenteeism time in hours -0.02 -0.20 0.03 -0.12 -0.00 0.05 -0.10 0.02 0.08 0.03 0.02 -0.05 -0.05 0.12 0.06 0.04 -0.03 -0.01 0.09 -0.06 1.00
ID
Reason for absence
Month of absence
Day of the week
Seasons
Transportation expense
Distance from Residence to Work
Service time
Age
Work load Average/day
Hit target
Disciplinary failure
Education
Son
Social drinker
Social smoker
Pet
Weight
Height
Body mass index
Absenteeism time in hours
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4


Absenteeism by Disciplinary Failure
Average Absenteeism (hours)
0
2
4
6
8
10
0.0
1.0
Disciplinary Failure (0 = No, 1 = Yes)


Absenteeism by Social Smoker Status
Average Absenteeism (hours)
0
2
4
6
8
10
12
14
16
0.0
1.0
Social Smoker (0 = No, 1 = Yes)

Regression Model Approach

Initial Goal

Predict exact absenteeism hours using Linear Regression

Evaluation Metrics

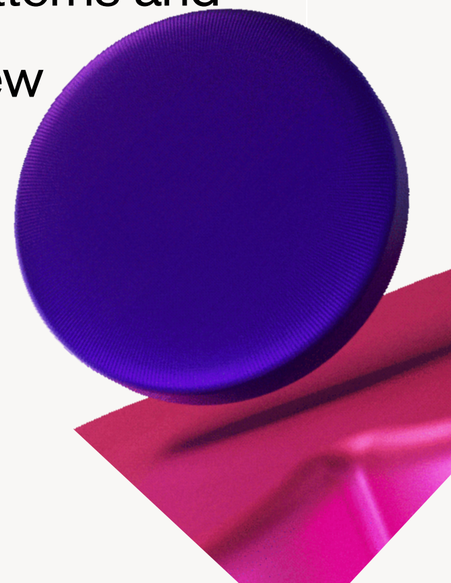
MAE: 7.88

RMSE: 17.84

R^2 Score: -0.97 (poor explanatory power)

Conclusion

Regression not suitable due to non-linear patterns and data skew



Classification Target Definition

- Initial Variable Created: Low, Medium and High Absenteeism level
- Created a new binary variable:
    - High Absenteeism
    - Low Absenteeism
- Ensured label balance and addressed class imbalance using stratified splitting

Classification Models Tested

Models

Logistic Regression

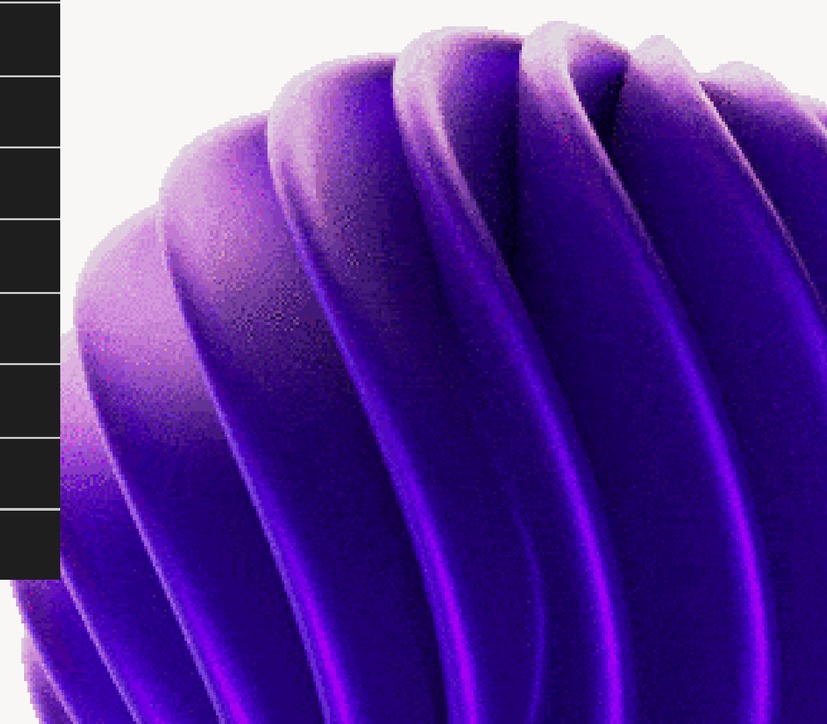
Random Forest

XGBoost

- Random Forest was chosen due to best performance on validation metrics
- Used GridSearchCV for hyperparameter optimization (432 combinations)

Initial Model Result Comparison - Balanced Data

Metric	Random Forest	XGBoost	Logistic Regression
Accuracy	88%	**90%**	67%
Recall (Low)	95%	**97%**	69%
Recall (Medium)	0%	0%	**25%**
Recall (High)	22%	**33%**	**56%**
F1-Score (Low)	94%	**95%**	80%
F1-Score (High)	24%	**40%**	27%
F1-Score (Medium)	0%	0%	**8%**
Precision (Low)	93%	**93%**	96%
Precision (High)	25%	**50%**	18%
Precision (Medium)	0%	0%	**5%**



New Binary Model Result Comparison

Metric	Logistic Regression	Random Forest	XGBoost
Accuracy	0.67	**0.75**	0.73
Precision (High)	0.63	**0.74**	0.72
Recall (High)	**0.71**	0.70	0.67
F1-Score (High)	0.67	**0.72**	0.69
Precision (Low)	0.72	0.76	**0.73**
Recall (Low)	0.64	**0.79**	0.78
F1-Score (Low)	0.68	**0.78**	0.76

Metric	Initial RF	Improved RF
Accuracy	0.75	**0.76**
Precision (High)	0.74	0.74
Recall (High)	0.70	**0.74**
F1-Score (High)	0.72	**0.74**
Precision (Low)	0.76	**0.78**
Recall (Low)	**0.79**	0.78
F1-Score (Low)	0.78	0.78

RF Top 10 Feature Importance

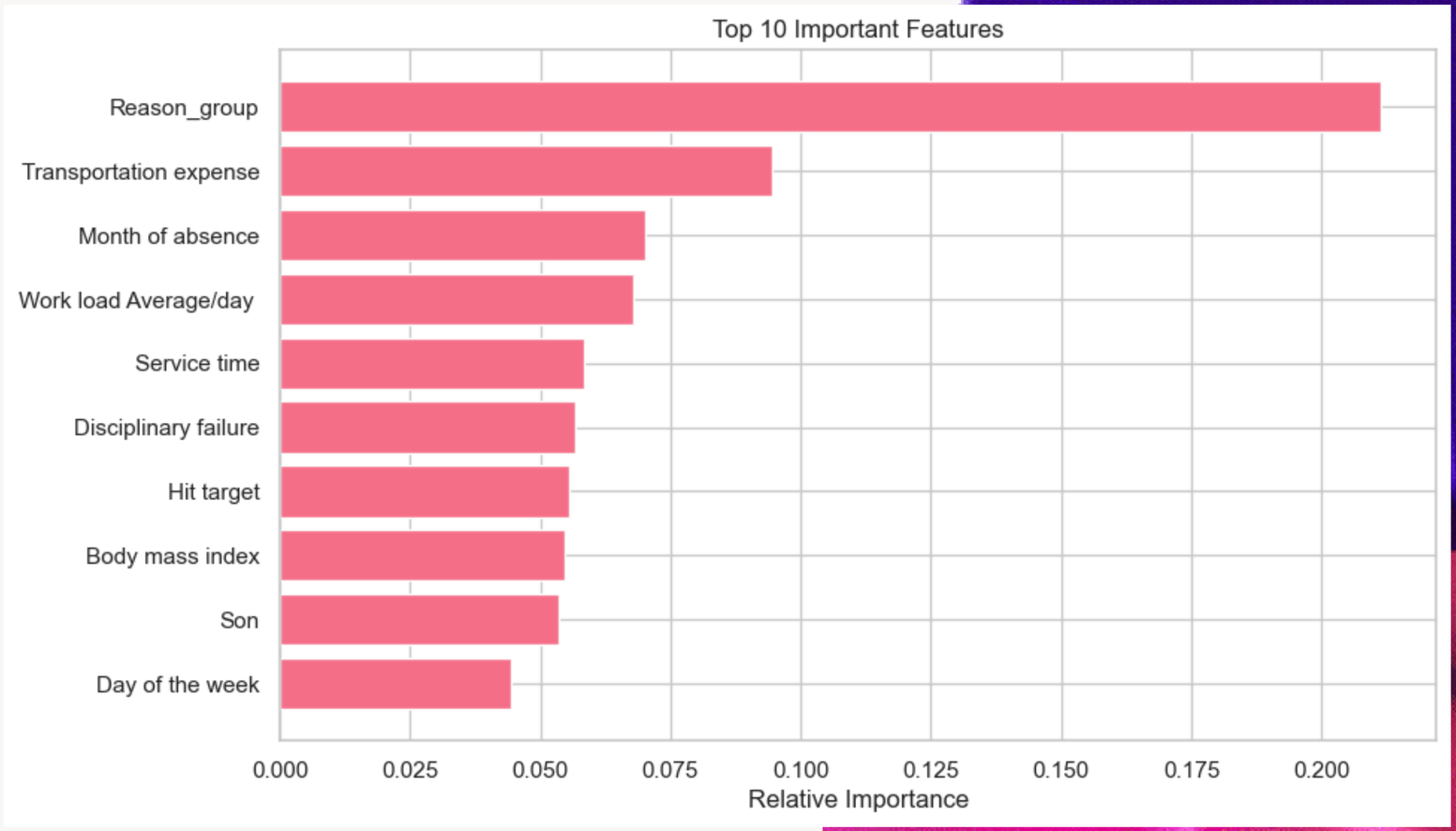


Tableau Visualization

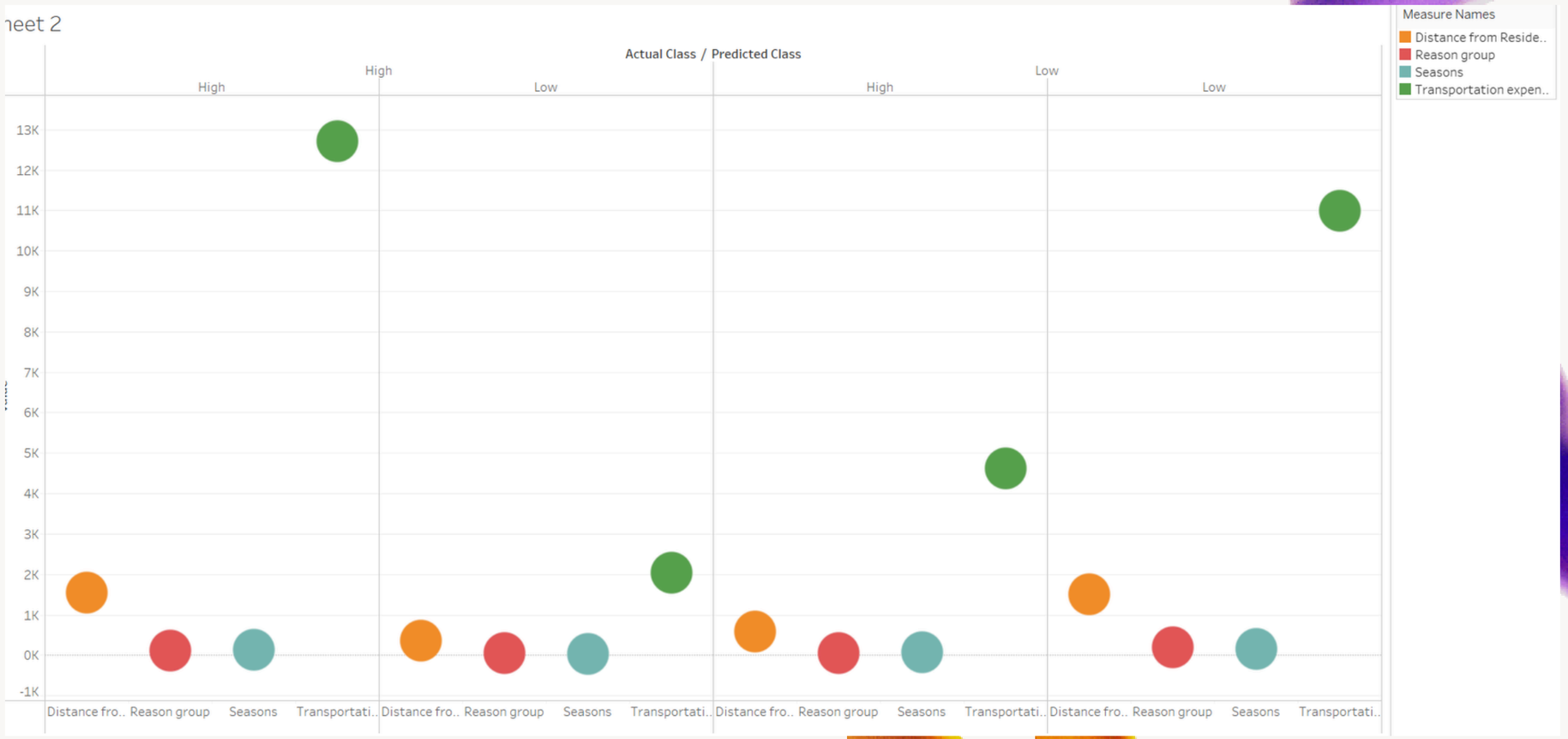


Tableau dashboard showed Actual vs Predicted Absenteeism

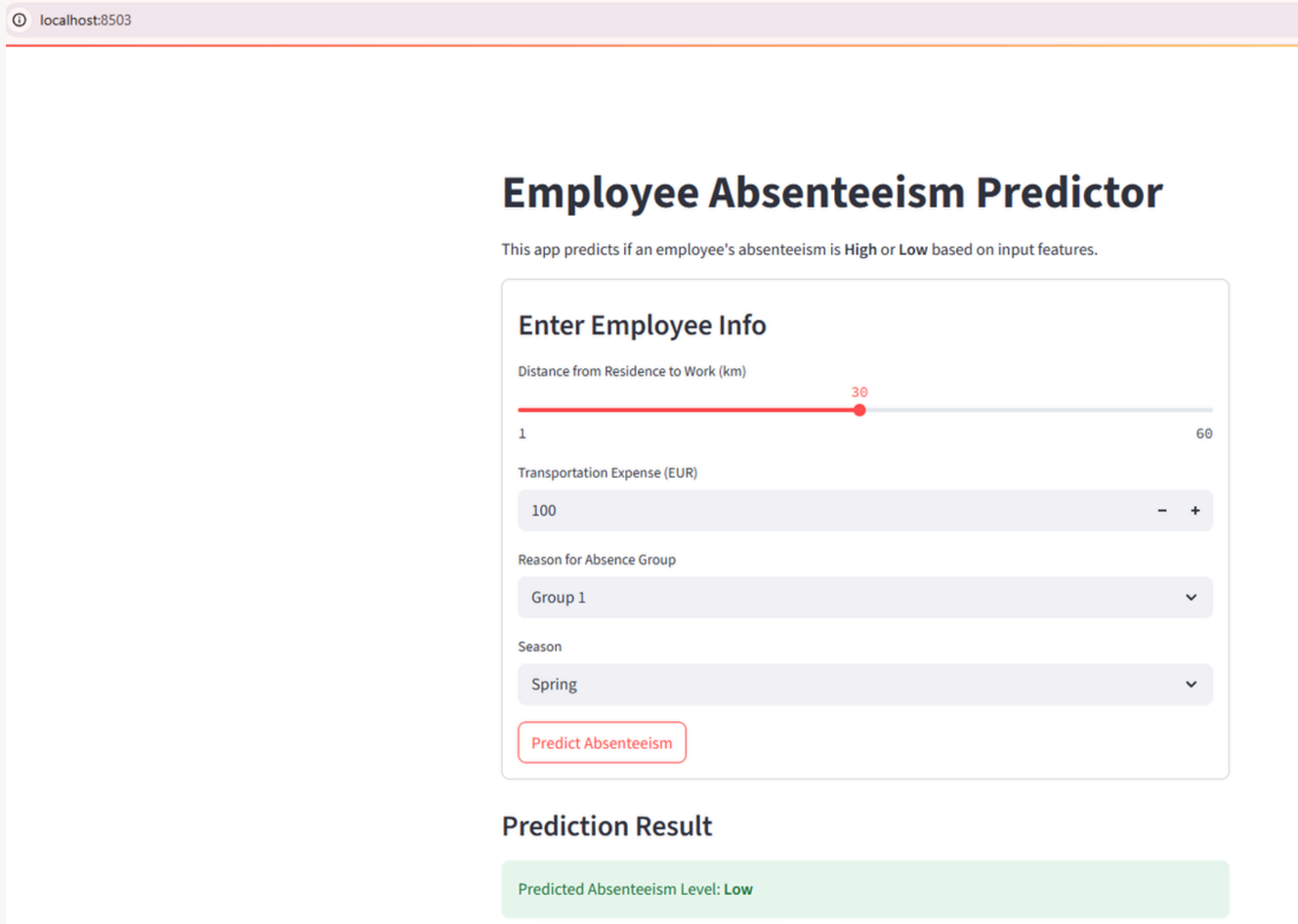
Compared to some of the top features

High transportation costs strongly align with actual High absenteeism

HR Chatbot - Streamlit

Deployed an HR-facing chatbot using Streamlit
Which allows HR managers to:

- Ask about absenteeism drivers
- Predict employee absenteeism
- Engage with an explainable AI assistant



Key Findings and Recommendation

Findings

- Regression model underperformed due to skewed data
- Classification with Random Forest gave a robust 76% accuracy
- Feature importance highlighted interpretable drivers
- Tableau visuals validated business relevance

Recommendation

- Monitor employees with:
  - Long commute distances
  - High transportation costs
  - Frequent medical absences
- Use model as a decision-support tool, not a standalone solution
- Consider adding behavioral and time-series data

Learnings

Trying multiple algorithms (Logistic Regression, XGBoost, Random Forest)

Streamlit App

Understanding the Evaluation Metrics

Translating technical outputs into strategic HR insights

Tableau visualization experience

Absenteeism

THANK YOU!