<u>Employee Absenteeism</u>

<u>Project Report</u>

Sharang Shrivastava

28th February , 2019

Contents

1. Overview

2. Data Summary

3. Exploratory Data Analysis
    3.1 Missing Values Analysis
    3.2 Feature Engineering
    3.3 Outliers Analysis

4. Correlation Analysis

5. Preparing data for modelling

6. Model building
    6.1 Linear regression Model
    6.2 Random Forest Model
    6.3 GradientBoost Model
    6.4 XGBoost Model

7. Model Performance Comparison

8. Model Predictions

1.Overview

Employee Absenteeism is the absence of an employee from work. Its a major problem faced by almost all employers of today. Employees are absent from work and thus the work suffers. Absenteeism of employees from work leads to back logs, piling of work and thus work delay. There are various laws been enacted for safeguarding the interest of both Employers and Employees but they too have various constraints.

Absenteeism is of two types -

- **Innocent absenteeism -** Is one in which the employee is absent from work due to genuine cause or reason. It may be due to his illness or personal family problem or any other real reason
- **Culpable Absenteeism -** is one in which a person is absent from work without any genuine reason or cause. He may be pretending to be ill or just wanted a holiday and stay at home. The employers have got every right to enquire as to why an employee is absent from work. If an employee is absent because of illness he should be able to produce a doctor's letter as and when demanded.

As per the survey conducted by US-based human capital services provider Careerbuilder, 30% of workers have called in sick when not actually ill in the past year. The sick days, legitimate or otherwise, also become more frequent around the winter holidays, with nearly one-third of employers reporting more employees call in sick during the holiday season, the survey found. At the same time, 29 per cent of employers have checked up on an employee to verify that the illness is legitimate, usually by requiring a doctor's note or calling the employee later in the day.

2. Data Summary

As a first step let's do three simple steps on the dataset

- Size of the dataset
- Get a glimpse of data by printing few rows of it.
- What type of variables contribute our data

Shape of data : 740 rows , 21 columns

Sample Of First Few Rows

data.head()

	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	...	Disciplinary failure	Education	Son	Social drinker	Social smoker	Pet	Wei
0	11	26.0	7.0	3	1	289.0	36.0	13.0	33.0	239554.0	...	0.0	1.0	2.0	1.0	0.0	1.0	[illegible]
1	36	0.0	7.0	3	1	118.0	13.0	18.0	50.0	239554.0	...	1.0	1.0	1.0	1.0	0.0	0.0	[illegible]
2	3	23.0	7.0	4	1	179.0	51.0	18.0	38.0	239554.0	...	0.0	1.0	0.0	1.0	0.0	0.0	[illegible]
3	7	7.0	7.0	5	1	279.0	5.0	14.0	39.0	239554.0	...	0.0	1.0	2.0	1.0	1.0	0.0	[illegible]
4	11	23.0	7.0	5	1	289.0	36.0	13.0	33.0	239554.0	...	0.0	1.0	2.0	1.0	0.0	1.0	[illegible]

5 rows × 21 columns

`Data Analysis`

`Check the properties of the data`

RangeIndex:	740 entries, 0 to 739
Data columns	(total 21 columns):
ID	740 non-null int64
Reason for absence	737 non-null float64
Month of absence	739 non-null float64
Day of the week	740 non-null int64
Seasons	740 non-null int64
Transportation expense	733 non-null float64
Distance from Residence to Work	737 non-null float64
Service time	737 non-null float64
Age	737 non-null float64
Work load Average/day	730 non-null float64
Hit target	734 non-null float64
Disciplinary failure	734 non-null float64
Education	730 non-null float64
Son	734 non-null float64
Social drinker	737 non-null float64
Social smoker	736 non-null float64
Pet	738 non-null float64
Weight	739 non-null float64
Height	726 non-null float64
Body mass index	709 non-null float64
Absenteeism time in hours	718 non-null float64

dtypes: float64(18), int64(3)

what we can infer:

> There are null values in the dataset

> The data types are int and float

Check for any invalid data inputs

From above observations data does not seem to have any invalid data types to be handled

However feature 'Absence_Month' have an invalid value 0. Let’s drop it.

ALso, as we can see, 'Absent_Hours' are 0 in some places.

This could be result of cancelled or withdrawn leaves. We will drop all these rows.

3. Exploratory Data Analysis(EDA)

3.1 Missing Value Analysis

Calculating % of nulls

```
ID                                  0.000000
Reason for absence                  0.405405
Month of absence                    0.135135
Day of the week                     0.000000
Seasons                             0.000000
Transportation expense              0.945946
Distance from Residence to Work     0.405405
Service time                        0.405405
Age                                 0.405405
Work load Average/day               1.351351
Hit target                          0.810811
Disciplinary failure                0.810811
Education                           1.351351
Son                                 0.810811
Social drinker                      0.405405
Social smoker                       0.540541
Pet                                 0.270270
Weight                              0.135135
Height                              1.891892
Body mass index                     4.189189
Absenteeism time in hours           2.972973
dtype: float64
```

what we can infer:

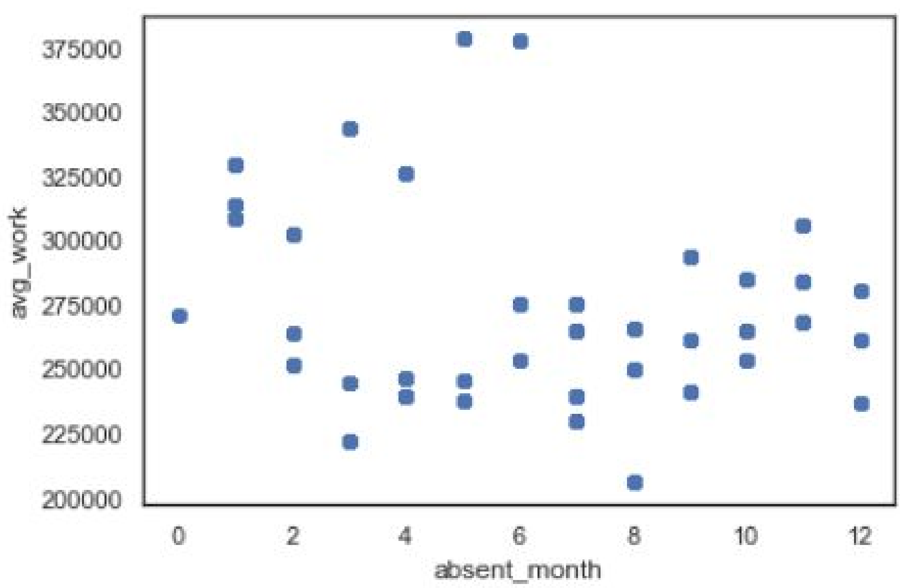
->There are null values in almost all the columns of the dataset, although in small amount.

-> We'll drop all the null value rows for target variable and

-> We'll will impute null values for all other features.

Replace missing of any any employee with information of same employee from other instances

Example if 'Age' of employee 1 is missing, then impute it with 'Age' from other instance of employee 1



From above, we can deduce that 'Average_Workload' is distributed mostly by month.

So, let's impute missing 'Average_Workload' by mode of that month

now only absent reason and hit target are left

ID	0
absent_reason	3

```
absent_month         0
day                  0
Seasons              0
transport_expense    0
dist_work            0
serv_time            0
age                  0
avg_work             0
hit_targ             6
displn_failure       0
education            0
son                  0
drinker              0
smoker               0
pet                  0
weight               0
height               0
bmi                  0
absent_hours        22
```

We will impute hit_target values by grouping by season month and day of the week and delete those rows which have null values in absent reason as these rows are only 3 in numbers. We will fill NA values in target with 0's.

Missing Value handling ENDS here

3.2 Feature extraction .

Extract any new features from existing features if required

Converting data to proper formats

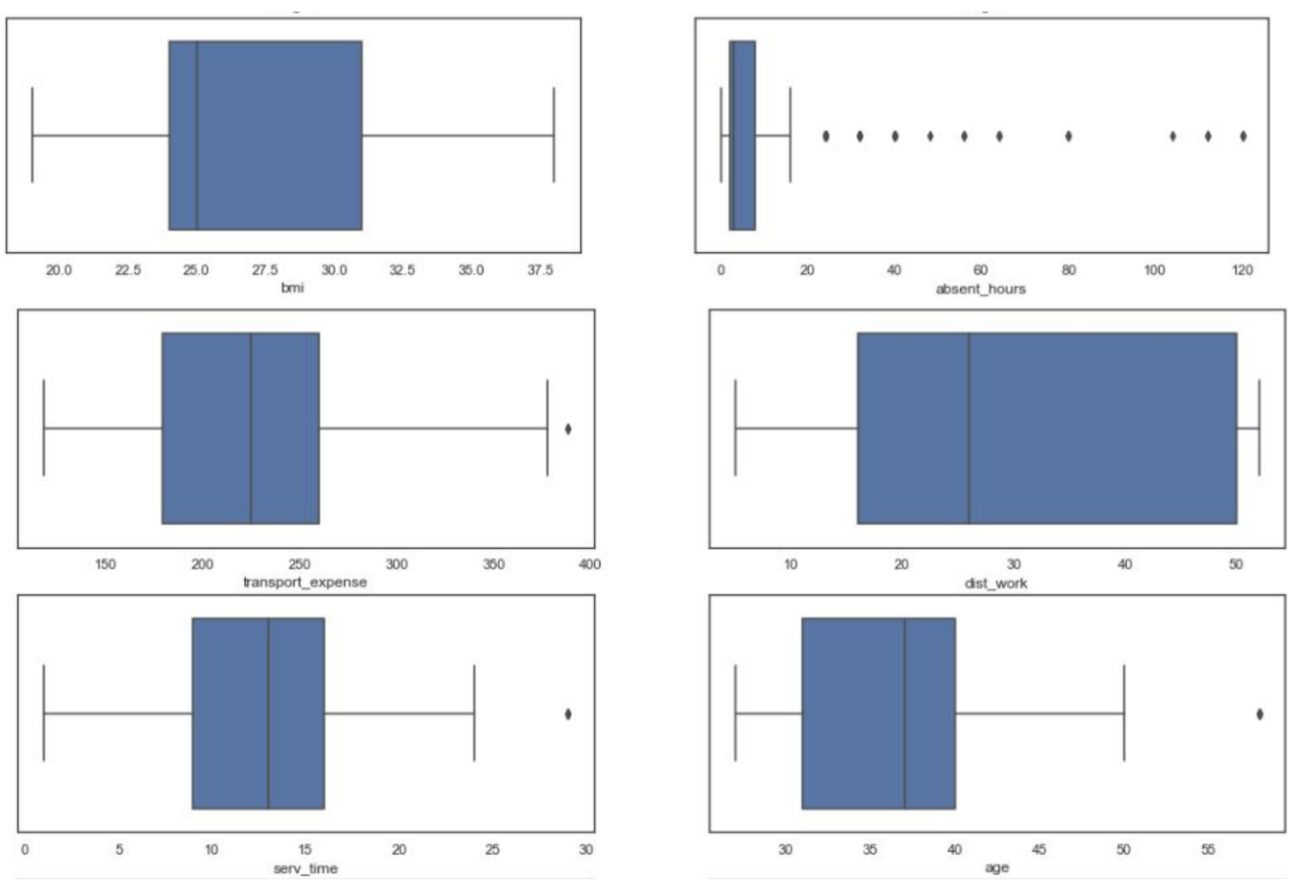
features like 'Absence_Month','Education',etc are categories here. Lets convert these variables to categories.

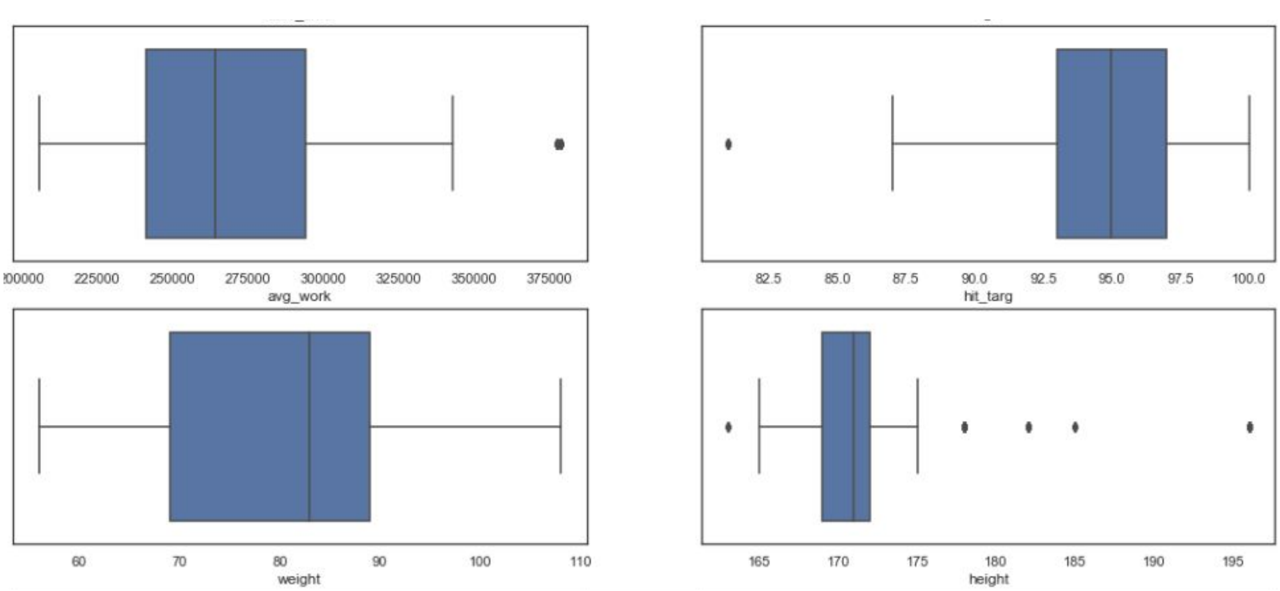
```
ID                        677 non-null int64
Absence_Reason            677 non-null category
Absence_Month             677 non-null category
Absence_Day               677 non-null category
Seasons                   677 non-null category
Transportation_Expense    677 non-null float64
Work_Distance             677 non-null float64
Service_Time              677 non-null float64
Age                       677 non-null float64
Average_Workload          677 non-null float64
Hit_Target                677 non-null float64
Disciplinary_Failure      677 non-null category
Education                 677 non-null category
Son                       677 non-null category
Drinker                   677 non-null category
Smoker                    677 non-null category
Pet                       677 non-null category
Weight                    677 non-null float64
Height                    677 non-null float64
BMI                       677 non-null float64
Absent_Hours              677 non-null float64
dtypes: category(10), float64(10), int64(1)
```

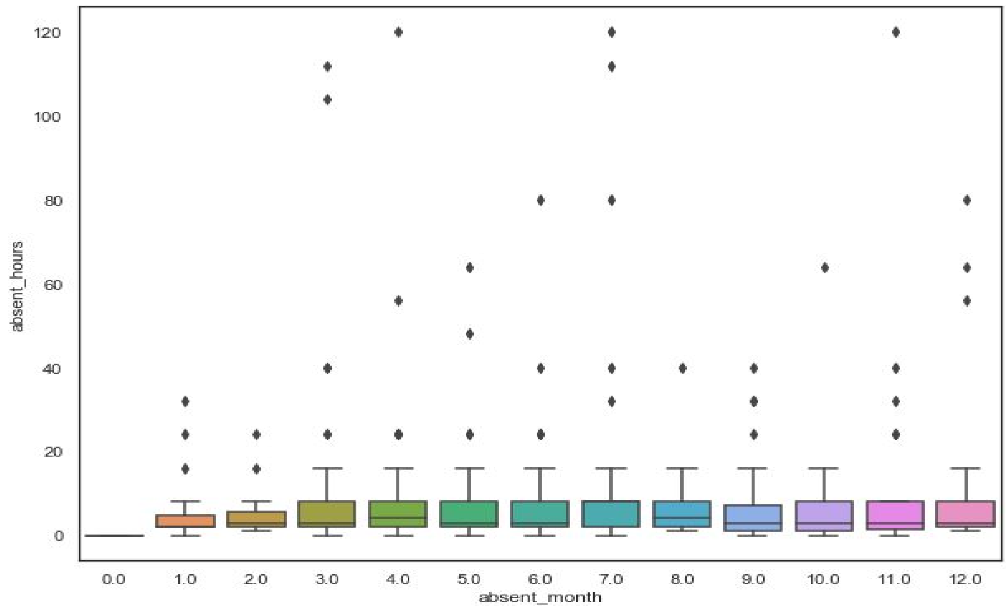
Here we do not need any feature extraction.

However, before feeding to model, we might need to aggregate the data.

3.3 Outlier Analysis







serv_time(Service_time) is not an outlier
now we will check for dist_work(distance from work)
data[data["dist_work"]==52] is having the highest dist_work and this is justified by looking at the transportation expense for that row
so dist_work is also not an outlier

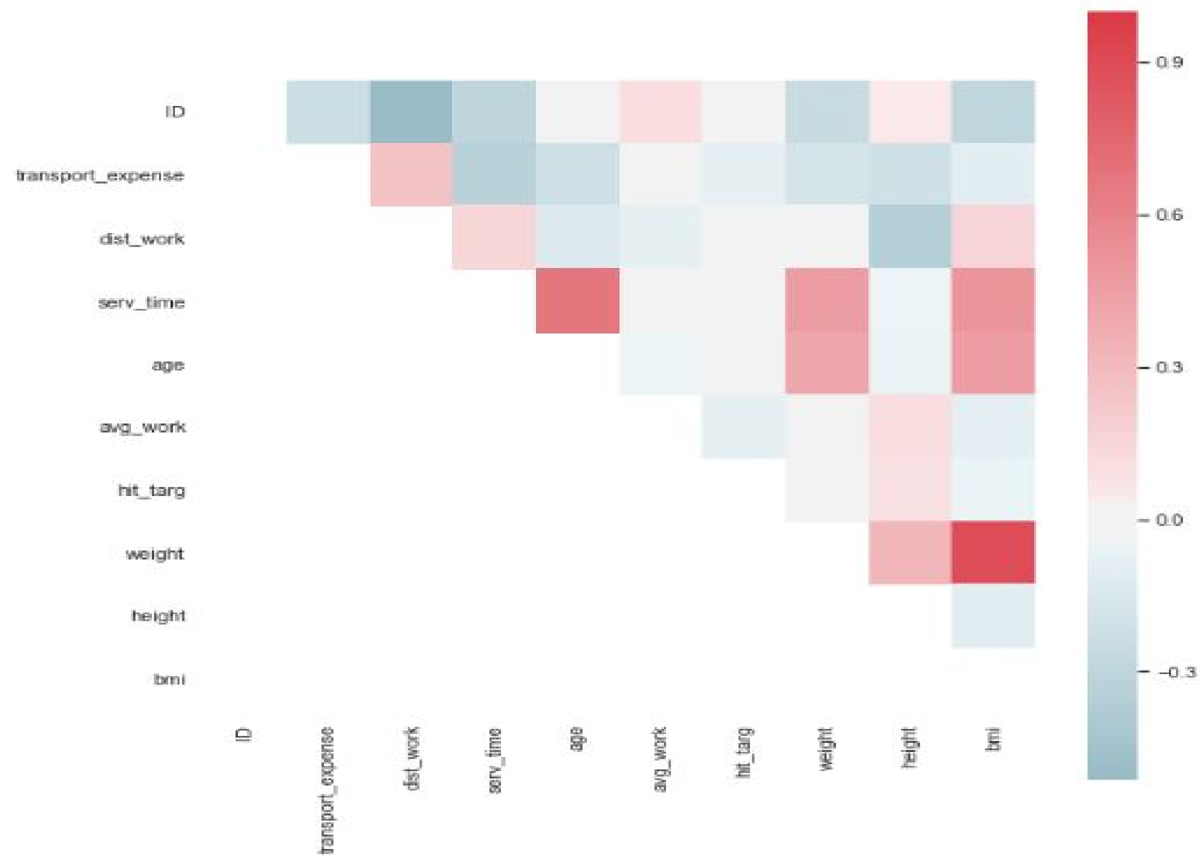
what we can infer from above boxplots:

-> Target feature 'Absent_hours', has many outliers. It needs to be handled(will handle it after exploratory analysis)

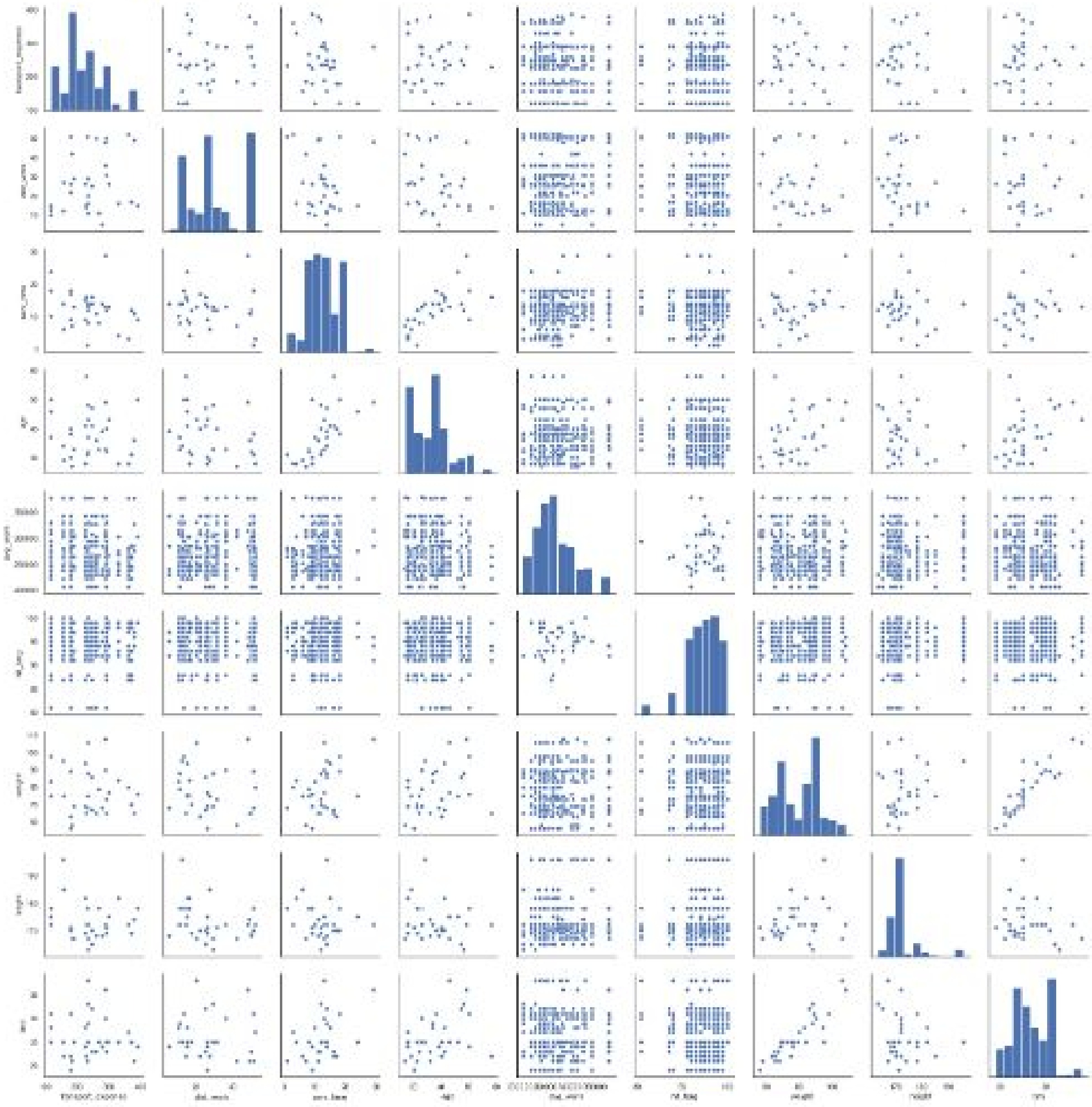
-> Not many outliers in independent features. Data seems balanced.

4. Correlation Analysis

correlation between the independent continuous features with target variable

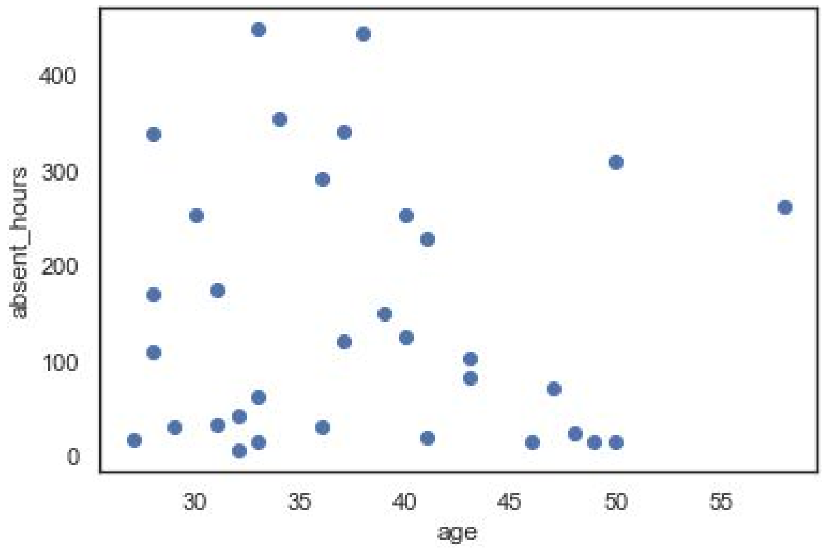


This shows that there is multicollinearity in the dataset. BMI and Weight are highly correlated. 'Service_Time' and 'Age' are also correlated
Will have to deal with multicollinearity by removing few features from the dataset.

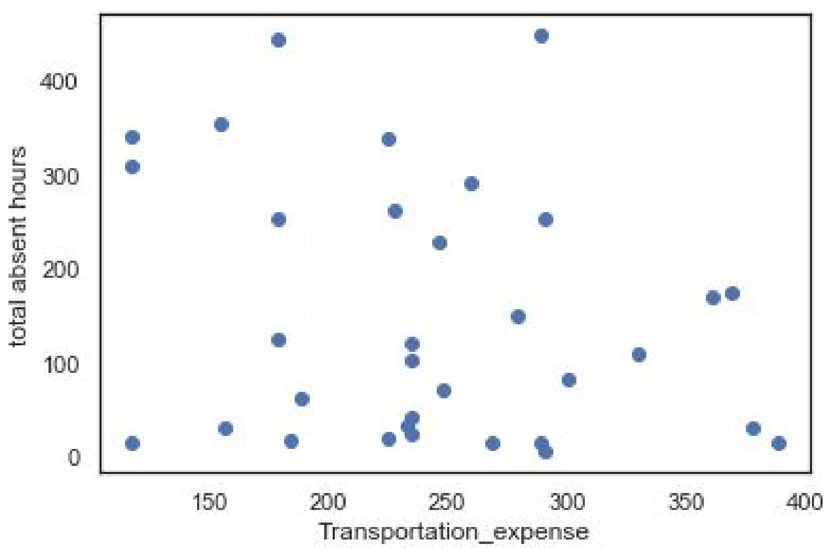


The linear relationship could be seen by looking at pair plot of (serv_time,age) and (bmi,weight).

`now we will look at the plot of age and absent hours to better understand the relationship`

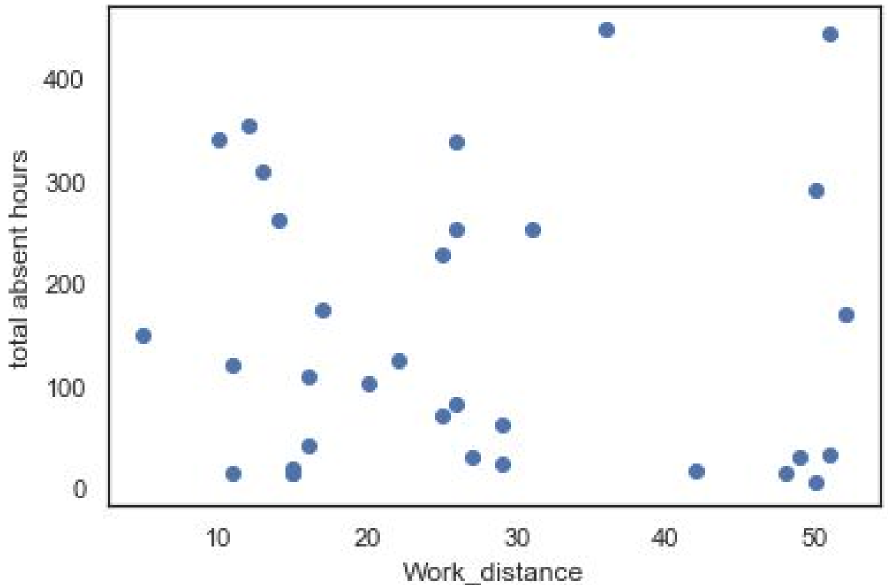


we can deduce from the following that employee having more than 40 years age use to take less leaves
now like this we will aggregate data and check for others independent var with target var
Transport expense with absent hours plot



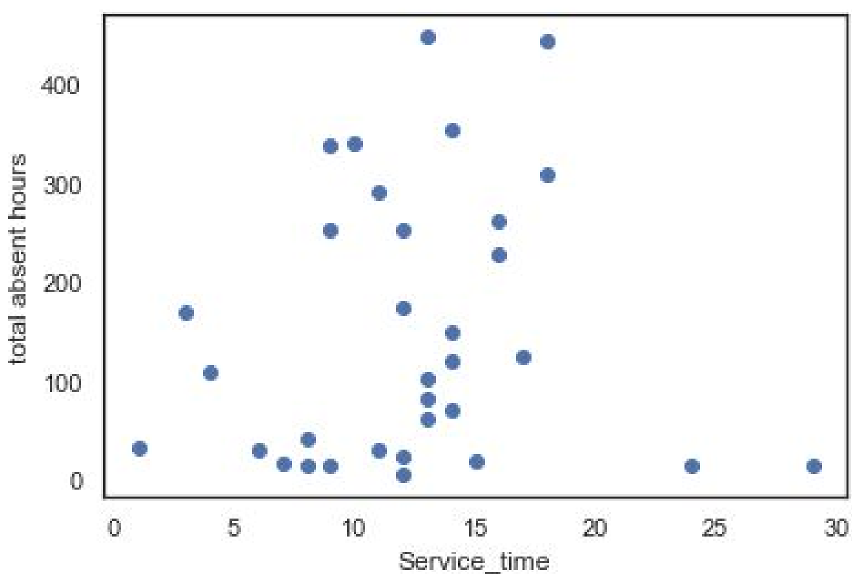
This clearly shows concentration of leaves more where the Transportation_Expense is between 150-300

now Work_distance and absent_hours plot



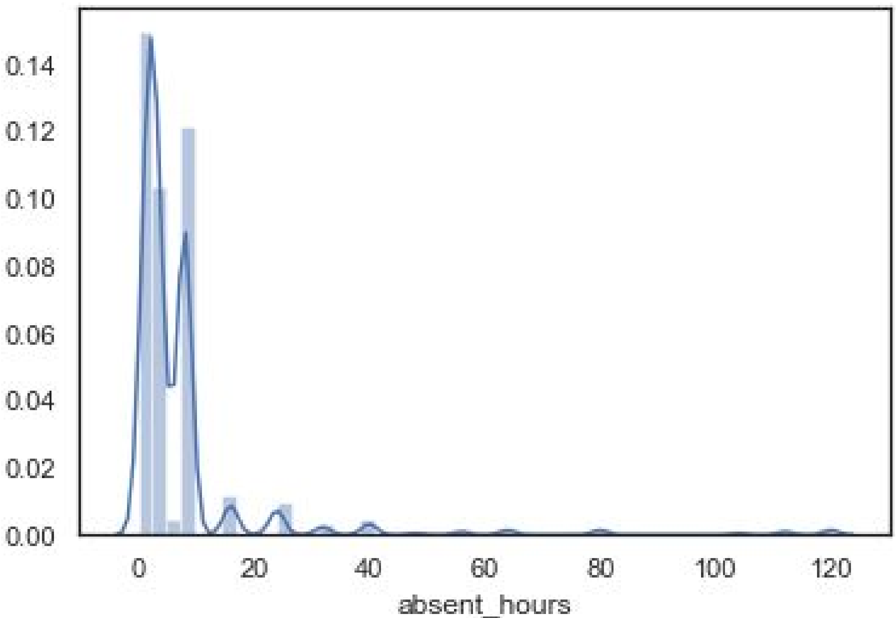
This clearly shows concentration of leaves more where the distance from work is between 10-30 km

Checking the effect of 'Service_Time' on 'Absence_hours



employees with service years < 8 and >18 tends to take less leaves
and employees with serv_time between 8-18 have been absent for most number of hours

Now we will see distribution of target variable.



What we can infer from above analysis of continuous variables:

-> Target variable 'Absent_Hours' is not normally distributed, which is not a good thing.

-> We have to look in to this, before feeding the data to model.

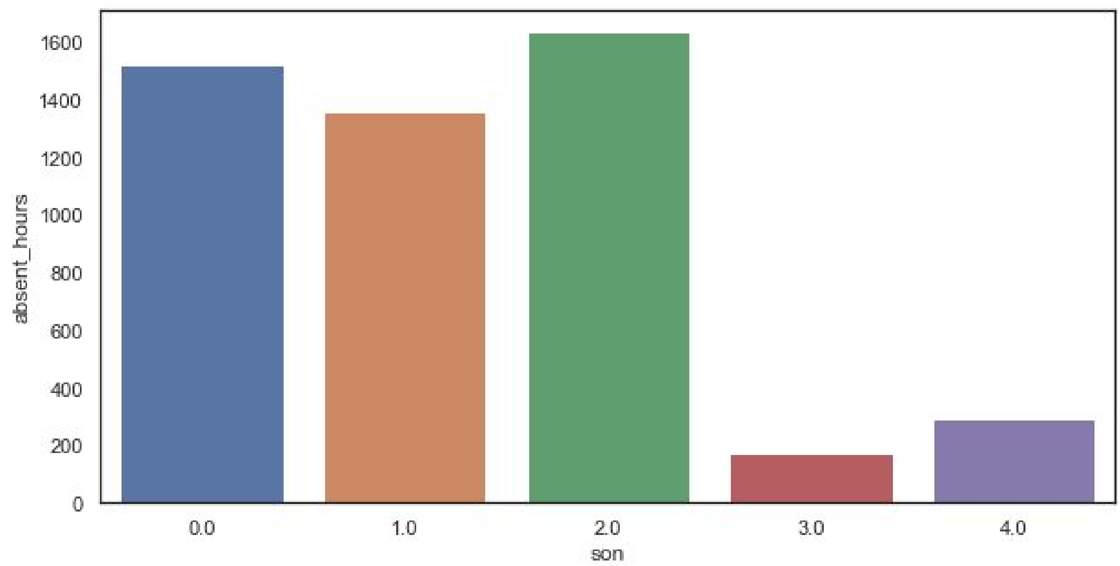
-> 'Work_Distance','Age','Average_Workload' has good correlation with target feature 'Absent_Hours'.

> Let's drop others from further analysis.

-> There is multicollinearity in dataset. 'Work_Distance' and 'Transportation_Expense' are correlated.

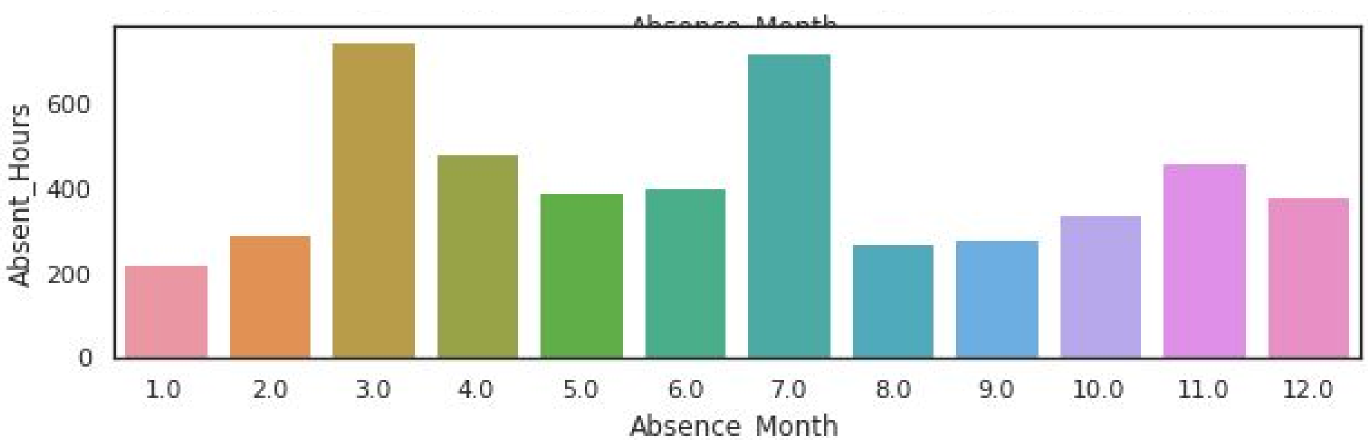
-> However, dist_work is more correlated towards target var,hence we will drop transport_expense analysis.

Analyzing absence dependency of no of kids



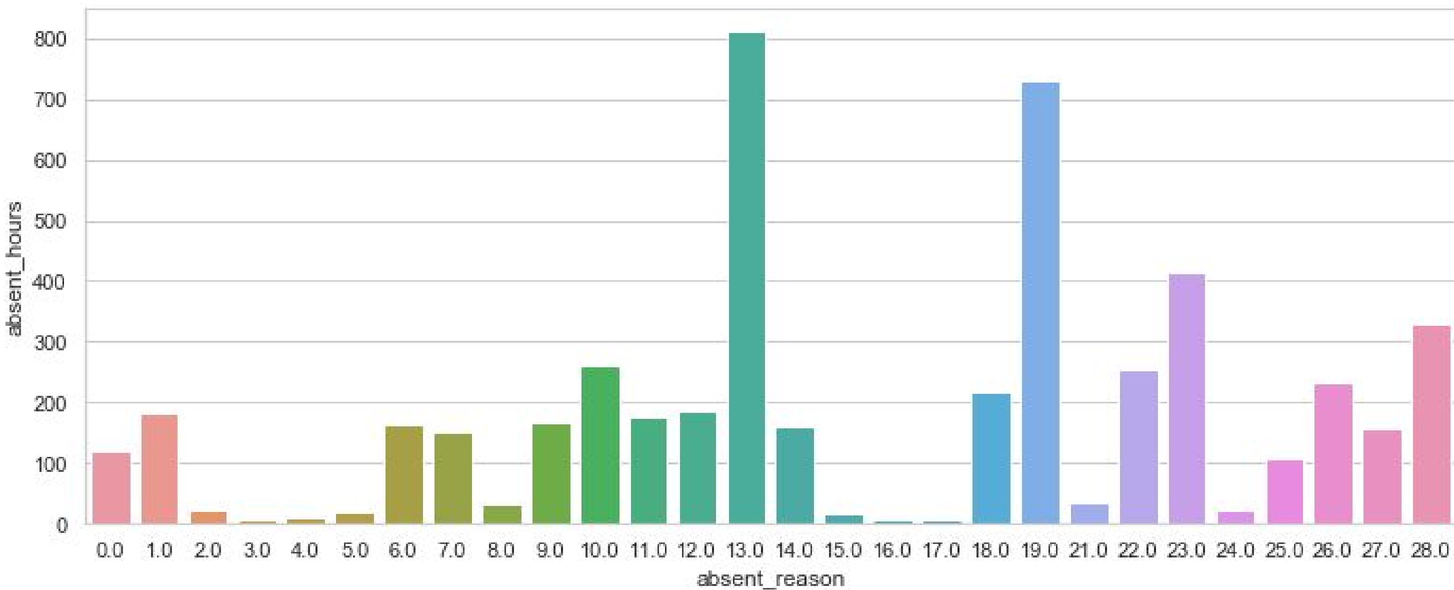
Clearly, employee with 3-4 kids tend to take less hours of absence

Analyzing absence dependency of month of year



March and july month clearly tops the list having most numbers of absent hours

Analyzing reason of absence with respect to sum of absent hours for that reason



Longest hours of absences for reason 13,19,23,28

#23 - medical consultation

#28 - dental consultation

#13 - Diseases of the musculoskeletal system and connective tissue

#19 - Injury, poisoning and certain other consequences of external causes

Seems like employee takes most absences for medical consultations/dental consultation and physiotherapy.These hours can be reduced by setting up a medical consultation/dental consultation/physiotherapy booth(with visiting doctors may be) at office/facility.In long term, introducing exercise/yoga sessions in office once/twice a week will help reduce physiotherapy issues

5. Preparing data for modelling

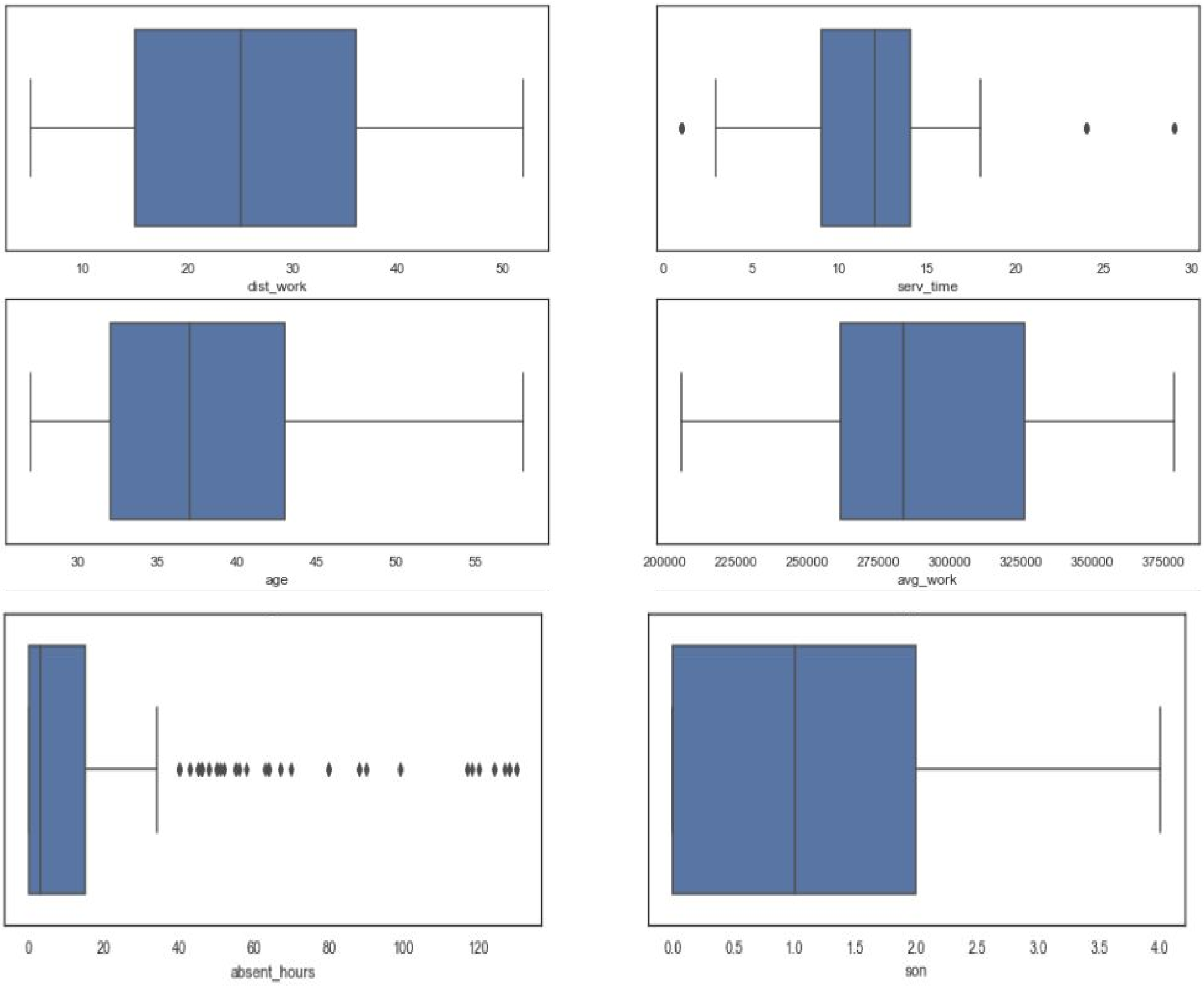
Now, since we need to predict the losses per month, Lets aggregate the data on month and ID before feeding the data to model.

	ID	absent_month	dist_work	serv_time	age	avg_work	drinker	son	absent_hours
0	1	1.0	11.0	14.0	37.0	330061.0	0.0	1.0	1.0
1	1	2.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	1	3.0	11.0	14.0	37.0	244387.0	0.0	1.0	16.0
3	1	4.0	11.0	14.0	37.0	326452.0	0.0	1.0	11.0
4	1	5.0	11.0	14.0	37.0	246074.0	0.0	1.0	16.0

```
Lets deal with Nans introduced(same way already done above, by imputing)
We will impute Nan values with max each value present for a particular id.
eg. Age will always be same for any id.
And update workload with the mode of corresponding month's workload, as we
did earlier.
Now the shape of the data got reduced to (396 rows, 9 columns) after
aggregating the data on basis of month and ID.
```

	ID	absent_month	dist_work	serv_time	age	avg_work	drinker	son	absent_hours
0	1	1.0	11.0	14.0	37.0	330061.0	0.0	1.0	1.0
1	1	2.0	11.0	14.0	37.0	302585.0	0.0	1.0	0.0
2	1	3.0	11.0	14.0	37.0	244387.0	0.0	1.0	16.0
3	1	4.0	11.0	14.0	37.0	326452.0	0.0	1.0	11.0
4	1	5.0	11.0	14.0	37.0	246074.0	0.0	1.0	16.0

Let’s check for any outliers in the aggregated data



Clearly, 'Absent_Hours' has so many outliers, this will affect model. So, extreme outliers needs to be removed to make the model more generic. We are not removing outliers in service time, since the input data for 2011 is going to be same as 2010(except 'Age' and 'Service Time')

Standardization of data

As we can clearly see that the dataset has different features of different range/scale.

Lets standardise the range/scale for better performance of model

	ID	absent_month	dist_work	serv_time	age	avg_work	drinker	son	absent_hours
0	1	1.0	0.87234	0.535714	0.677419	0.282268	0.0	1.0	1.0
1	1	2.0	0.87234	0.535714	0.677419	0.441119	0.0	1.0	0.0
2	1	3.0	0.87234	0.535714	0.677419	0.777588	0.0	1.0	16.0
3	1	4.0	0.87234	0.535714	0.677419	0.303133	0.0	1.0	11.0
4	1	5.0	0.87234	0.535714	0.677419	0.767834	0.0	1.0	16.0

Now we are done preparing data for modelling, we will start building our models on top of it

6. Model building

6.1 Linear regression Model

lrm_regressor = LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=True)

6.2 Random Forest Model (Ensemble method using Bagging technique)

forest_reg = RandomForestRegressor(n_estimators=2000, criterion='mse', max_depth=10, min_samples_split=5, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=20, min_impurity_decrease=0.00, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=1, random_state=1, verbose=0, warm_start=False)

6.3 GradientBoost Model (Ensemble method using Boosting technique)

 # without parameter tuning

gb_reg = GradientBoostingRegressor(ranxgb = xgboost.XGBRegressor(n_estimators=100, learning_rate=0.08, gamma=0, subsample=0.75,

colsample_bytree=1, max_depth=7)

dom_state=1)

 # Following model is with parameter tuning

gb_reg = GradientBoostingRegressor(loss='ls', learning_rate=0.2, n_estimators=2000, subsample=1.0, criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=3, min_impurity_decrease=0.0, min_impurity_split=None, init=None, random_state=1, max_features=None, alpha=0.9, verbose=0, max_leaf_nodes=15, warm_start=False, presort='auto')

6.4 XGBoost Model

xgb = xgboost.XGBRegressor(n_estimators=100, learning_rate=0.08, gamma=0, subsample=0.75,

colsample_bytree=1, max_depth=7)

7. Model Performance Comparison

	Model	Mean Squared Error	Root Mean Squared Error	Mean Absolute Error
0	Linear Reg	0.000103	0.010131	0.007699
1	Random Forest	0.000076	0.008743	0.006912
2	GradientBoost	0.000081	0.009016	0.006252
3	XgBoost	0.000081	0.008992	0.006579

It is cleared that random forest is predicting target values with least root mean squared error and other metrics. Now we will try Grid Search to fine tune the parameters of random forest model.

```
param_grid = {
    'n_estimators': [800, 1600,2400],
    'max_features': ['auto', 'sqrt', 'log2']
}
CV_rfc = GridSearchCV(estimator=forest_reg, param_grid=param_grid, cv= 5)
```

Best parameters - {'max_features': 'sqrt', 'n_estimators': 2400}

Root mean squared error for new random forest model 0.008639010819575816

Previous RMSE for random forest was **0.008743**

So it has clearly performed better after tuning the parameters

8. Model Predictions

Predictions of our model with respect to data provided.

Predicted absence hours of 2010 - 2493.418314528776

Actual absence hours of 2010   - 1984.0

" Predicted and actual absence hours per month "

	absent_month	absent_hours	Predicted_Absent_Hours
0	1	108.0	137.702743
1	2	185.0	183.782749
2	3	139.0	174.035775
3	4	237.0	179.420688
4	5	249.0	271.309499
5	6	119.0	204.438903
6	7	211.0	250.070921
7	8	129.0	261.361245
8	9	101.0	200.702479
9	10	146.0	206.724767
10	11	205.0	205.898869
11	12	155.0	217.969678

Since, random forest model is our final model to be used for prediction, We'll use this model to predict the losses of 2011. We will now prepare data for 2011

To prepare data for 2011,assuming that all the employees are retained in 2011 and all other condition remains and same trends continues, we need to add +1 to 'Service_Time' and 'Age'(keeping all other features same)

Predictions for 2011

New Prepared Data for 2011

	absent_month	dist_work	serv_time	age	avg_work	drinker	son
0	1	0.87234	9.535714	9.677419	0.282268	0.0	1.0
1	2	0.87234	9.535714	9.677419	0.441119	0.0	1.0
2	3	0.87234	9.535714	9.677419	0.777588	0.0	1.0
3	4	0.87234	9.535714	9.677419	0.303133	0.0	1.0
4	5	0.87234	9.535714	9.677419	0.767834	0.0	1.0

Predicted absent_hours for 2011 data

	absent_month	dist_work	serv_time	age	avg_work	drinker	son	Predicted_Absent_Hours
0	1	0.12766	0.464286	0.322581	0.717732	0.0	1.0	8.296749
1	2	0.12766	0.464286	0.322581	0.558881	0.0	1.0	7.178514
2	3	0.12766	0.464286	0.322581	0.222412	0.0	1.0	6.765780
3	4	0.12766	0.464286	0.322581	0.696867	0.0	1.0	8.836768
4	5	0.12766	0.464286	0.322581	0.232166	0.0	1.0	7.563202

Predicted absent_hours per month in 2011

	absent_month	Predicted_Absent_Hours
0	1	272.170448
1	2	242.631656
2	3	243.232376
3	4	267.658245
4	5	285.469534
5	6	340.420058
6	7	259.372971
7	8	284.068893
8	9	255.903953
9	10	268.951578
10	11	266.506938

MONTHLY LOSSES PREDICTED FOR YEAR 2011 PER MONTH

Let's say in a month excluding weekend 22 days are working days.

Total working hours of 36 employees will be 22*8*36.

total losses % = (absent_hours / Total_Hours)*100

tot_Monthly_hours = 22*8*36

	absent_month	Predicted_Absent_Hours	monthly_loss_percentage
0	1	272.170448	4.295619
1	2	242.631656	3.829414
2	3	243.232376	3.838895
3	4	267.658245	4.224404
4	5	285.469534	4.505517
5	6	340.420058	5.372791
6	7	259.372971	4.093639
7	8	284.068893	4.483411
8	9	255.903953	4.038888
9	10	268.951578	4.244817
10	11	266.506938	4.206233
11	12	259.585973	4.097001

THE END