Edwisor

Data Scientist Career Path

Machine Learning Regression Model

Topic: Absenteeism Dataset Prediction

By: Arvind Kumar

Contents:

Introduction

1.1 Problem Statement

As we know the company ABC Pvt. Ltd. Have the data of their employees. They want to some important details from the data. Like Absenteeism of the employees. The company has shared it dataset and requested to have an answer on the following areas:

1. What changes company should bring to reduce the number of absenteeism?
2. How much losses every month can be project in 2011 if same trend of absenteeism continues?

1.2 Data description

There are 21 variable in the dataset in which 20 are independent and 1 is dependent (Absenteeism.time.in.hours).Since out target variable is continuous in nature so we can say that this is the problem is regression problem.

Variable transformation:

1. Individual identification
2. Reason for absence
3. Month of absence
4. Day of week(Monday(2),Tueseday(3),Wednesday(4),Thursday(5),Friday(6))
5. 5.Seasons(Summer(1), autumn(2), winter(3), spring(4))
6. Transportation expense
7. Distance from residence to work(km)
8. Service time
9. Age
10. Work load Average/day
11. Hit target
12. Disciplinary failure
13. Education
14. Son(num of children)
15. Social smoker (yes=1,no 0)
16. Social drinker(yes=1,no=0)
17. Pet(number of pet)
18. Weight
19. Height
20. Body mass index
21. Absenteeism time in hours (target)

1.3 Exploratory data analysis

Exploratory data analysis(EDA) is an approach to analyzing data sets to summarize their main characteristics. In the given dataset there are 21 variables and data types of all variables are either float64 or int64 .there are 740 number of rows(observations) and 21 variables or columns. And missing values are present in our dataset .

Column names and their data types

```
: data.dtypes
```

```
: ID                                int64
  Reason for absence              float64
  Month of absence                float64
  Day of the week                   int64
  Seasons                           int64
  Transportation expense          float64
  Distance from Residence to Work float64
  Service time                    float64
  Age                             float64
  Work load Average/day           float64
  Hit target                      float64
  Disciplinary failure            float64
  Education                       float64
  Son                             float64
  Social drinker                  float64
  Social smoker                   float64
  Pet                             float64
  Weight                          float64
  Height                          float64
  Body mass index                 float64
  Absenteeism time in hours       float64
  dtype: object
```

Column name and their number of unique values

```
# number of unique value in each variable
data.nunique()
```

```
ID                                36
Reason for absence                28
Month of absence                  13
Day of the week                    5
Seasons                            4
Transportation expense            24
Distance from Residence to Work   25
Service time                      18
Age                               22
Work load Average/day             38
Hit target                        13
Disciplinary failure               2
Education                          4
Son                                5
Social drinker                     2
Social smoker                      2
Pet                                6
Weight                            26
Height                            14
Body mass index                   17
Absenteeism time in hours         19
dtype: int64
```

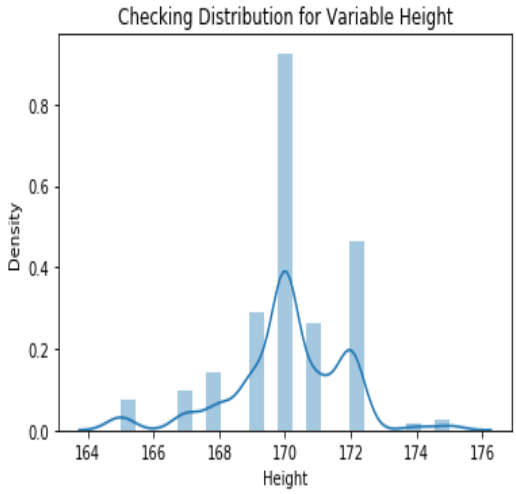
From the dataset we found there are 10 continuous variables and 11 categorical vars.

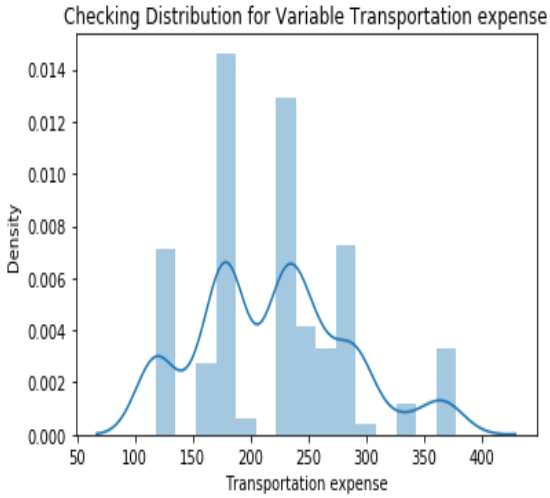
Methodology

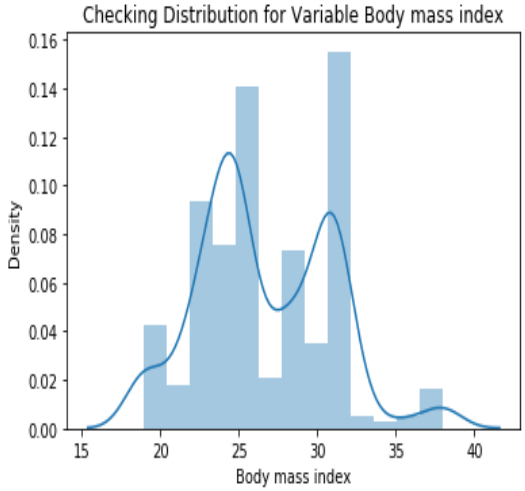
before feeding the data to the model we need to clean the data and convert it to a proper format. It is the most crucial part of the machine learning project we spend almost 80% of time in it.

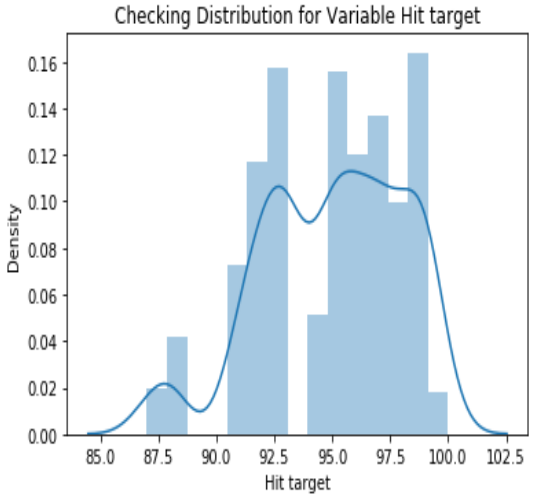
2.1 Pre Processing

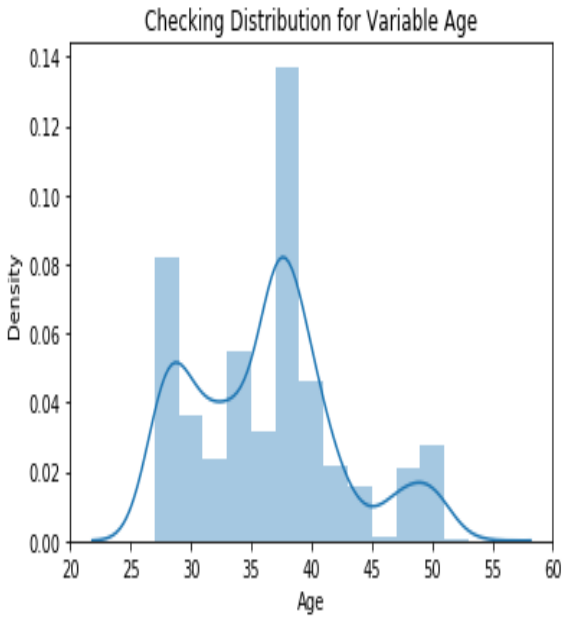
Any predictive modeling requires that we look at the data before we start modeling. However, in data mining terms looking at data refers to so much more than just looking. Looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots .this is often called as exploratory data analysis to start this process we will look at the all the probability distributed. We can visualize that in a glance by looking at the probability distributions or probability density functions of the variable.

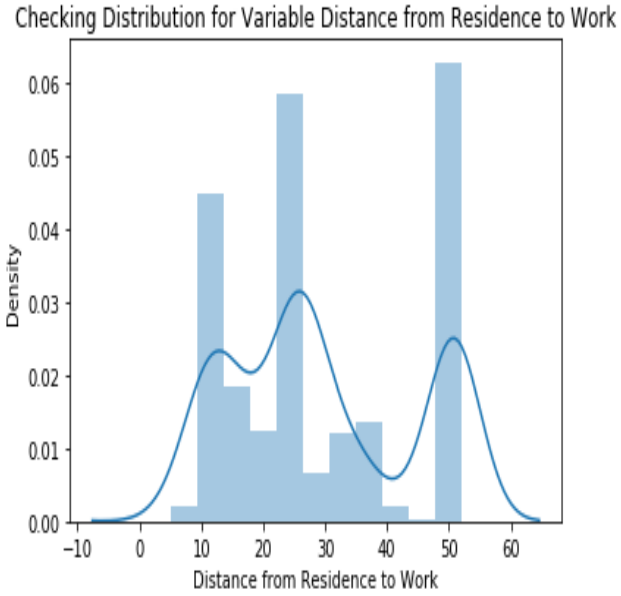


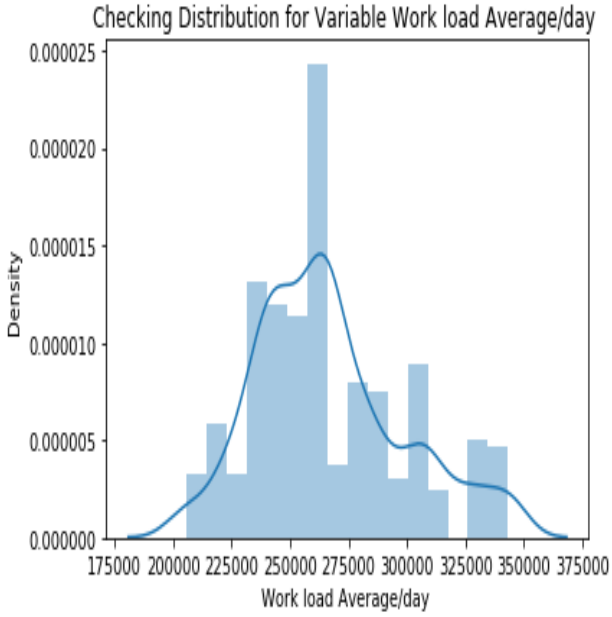


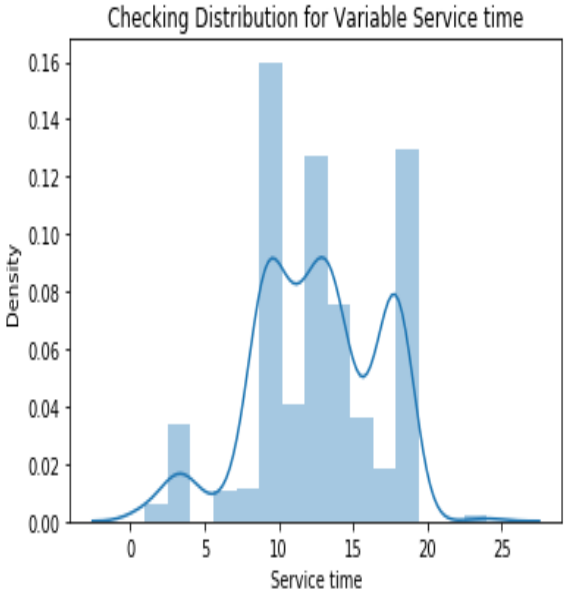





Checking Distribution for Variable Age
Density
Age


Checking Distribution for Variable Distance from Residence to Work
Density
Distance from Residence to Work


Checking Distribution for Variable Work load Average/day
Density
Work load Average/day


Checking Distribution for Variable Service time
Density
Service time

2.2.1 Missing Value analysis

In this data missing value occur when no data value stored for the variable in an observation. Missing data are a common occurrence and can have significant effect on the conclusions that can be drawn from the data. If a columns has more than 30% of data as missing value either we ignore the entire column or we ignore those observations. In the given data the maximum percentage of missing value 4.189% for the column Body mass index. So we compute the missing value for all the missing value column.

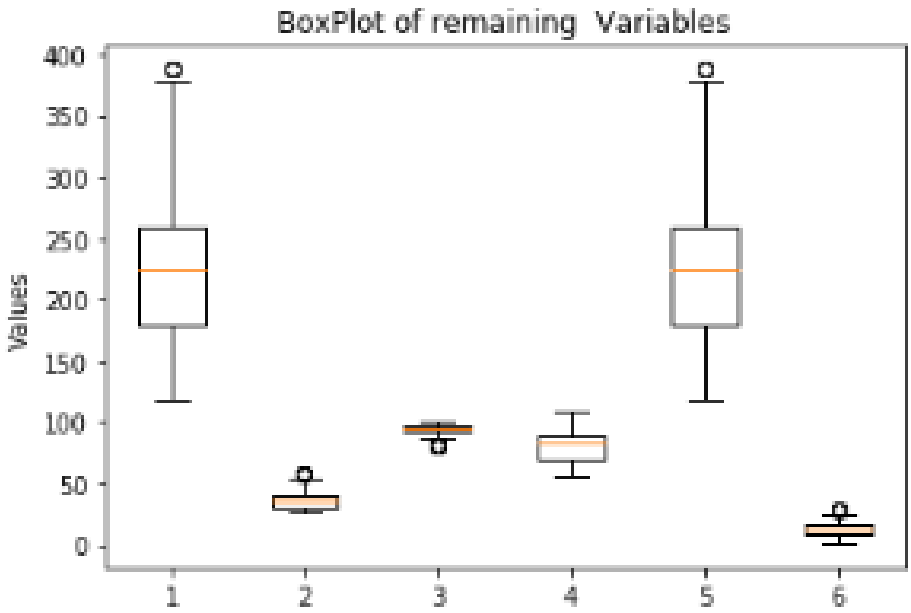
```
#descending order
missing_val = missing_val.sort_values('Missing_percentage', ascending = False).reset_index(drop = True)

missing_val
```

	Variables	Missing_percentage
0	Body mass index	4.189189
1	Absenteeism time in hours	2.972973
2	Height	1.891892
3	Work load Average/day	1.351351
4	Education	1.351351
5	Transportation expense	0.945946
6	Son	0.810811
7	Disciplinary failure	0.810811
8	Hit target	0.810811
9	Social smoker	0.540541
10	Age	0.405405
11	Reason for absence	0.405405
12	Service time	0.405405
13	Distance from Residence to Work	0.405405
14	Social drinker	0.405405
15	Pet	0.270270
16	Weight	0.135135
17	Month of absence	0.135135
18	Seasons	0.000000
19	Day of the week	0.000000
20	ID	0.000000

2.2.2 Outlier Analysis

We can clearly observe from these probability distributaions that most of the variables are skewed. Skew in these distributions can be most likely explained by the presence of outliers and extreme values in the data. One of the other steps of pre processing apart from checking for normality is the presence of outliers. In this we use a classic approach of removing outliers. We visualize the outliers using boxplots.

In figure we have plotted the boxplots of the predictore variables which are the continuous variables with respect to the absenteeism time in hours. A lot of useful inferences can be made from these plots. First as you can see, we have a lot of outliers and extreme values in each of the dataset.

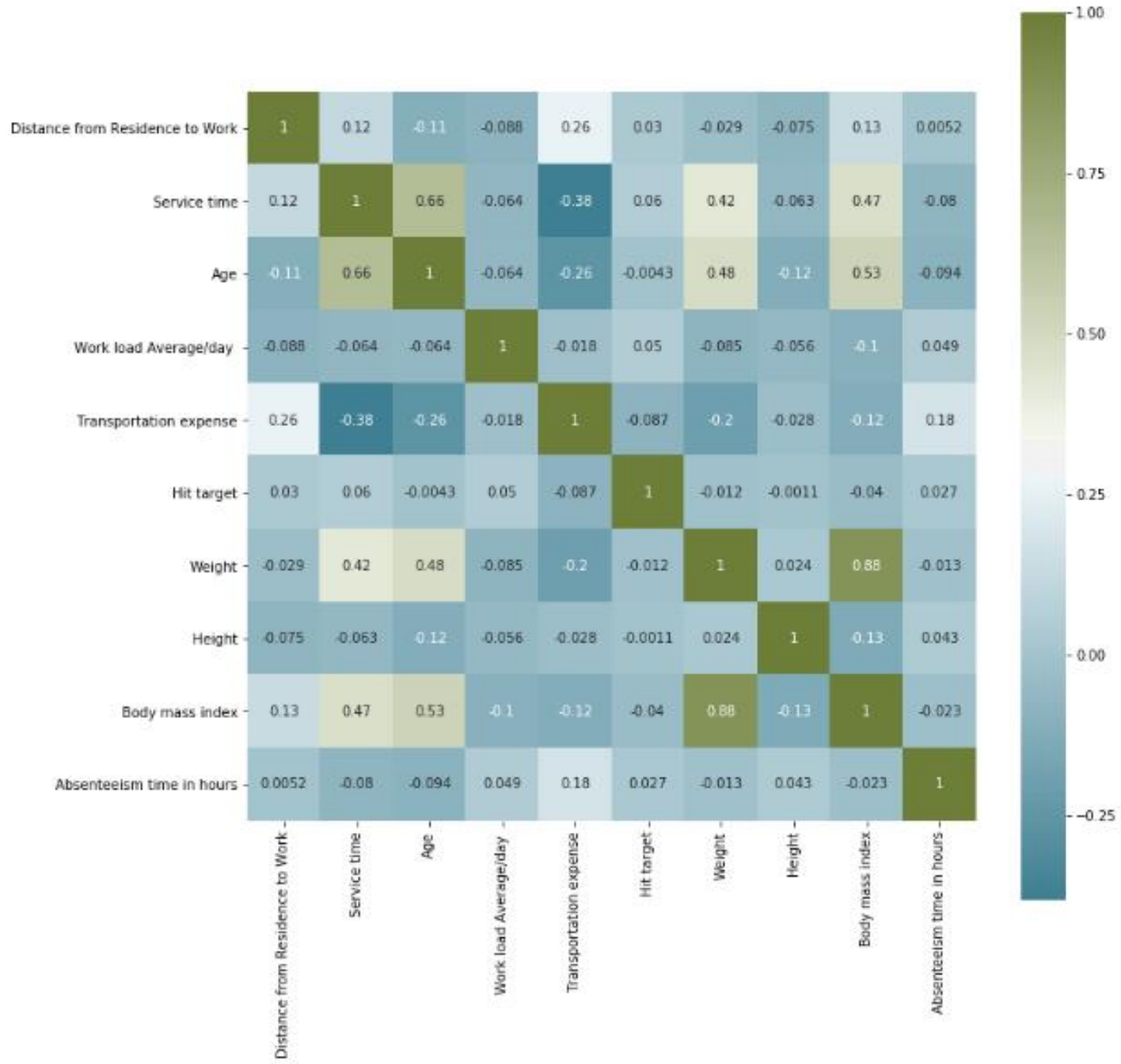


Outlier sample diagram for some of the variables

After the outliers removal we fill the outliers with NAs then we fill the NAs with the median value of the column.

2.2.3 feature selection

Before performing any type of modeling we need to access the importance of each predictor variable in our analysis. There is a possibility that many variables in our analysis are not important at all to the problem of class prediction. Selecting subset of relevant columns for the model construction is known as feature selection. We can not use all the features because some features may be carrying the same information or irrelevant information which can increase overload. To reduce overload we adopt feature selection technique to extract meaningful feature of the data. This in turn help us to avoid the problem of multi collinearity. In this project we have selected correlation analysis of numerical variables and anova analysis for categorical variables.



from correlation analysis we have found that Weight and Body mass index has high correlation (>0.7), so we have excluded the Weight column.

2.2.4 Feature Scaling

Feature scaling is a method used to standardize the range of independent variables or feature of data. In data pre processing, it is also known as data normalization and is generally performed during the data pre processing step. Since the range of values of raw data varies widely, in some machine learning algorithms, objective function will not work properly without normalization. For example, the majority of classiers calculate the distance between two points by the euclidean distance. If one of the feature has broad range of values, the distance will be governed by this particular feature. Therefore, the range of all features should be normalized so that each feature contributions approximately proportionately to that distance since our data is not uniformly distributed.

In this we will used standardization method.

2.2.5 Feature Splitting

Before devide the dataset into the train and test case we do sampling of the data if we have long data like millions of rows but we have little data like 714 rows so there is no need to do sampling we applied splitting the orginal dataset and devide it into the train and test case.

2.2 Modeling

After pre processing we will be using Regressor models on our processed data to predict the target variable. Following are the models which we have built-

2.2.1 Decision tree Regressor

A decision tree is a support tool that uses a tree –like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. Each branch connects nodes with “and” and multiple branches are connected by “or” .It can be used for classification and regresson it is a supervised machine learning algorithm. Accept continuous and categorical variables as independent variables. Extremely easy to understand by the business users. Split of decision tree is seen in the below tree. The RMSE value and time taken by the algorithm is shown in the output.

Decision Tree Regressor

```
# Importing libraries for Decision Tree
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_squared_error

# Building model on top of training dataset
fit_DT = DecisionTreeRegressor(max_depth = 2).fit(X_train,y_train)

# Calculating RMSE for training data to check for over fitting
pred_train = fit_DT.predict(X_train)
rmse_for_train = np.sqrt(mean_squared_error(y_train,pred_train))

# Calculating RMSE for test data to check accuracy
t5=time.time()
pred_test = fit_DT.predict(X_test)
t6=time.time()-t5
rmse_for_test =np.sqrt(mean_squared_error(y_test,pred_test))

print("RMSE For Train = "+str(rmse_for_train))
print("RMSE For Test = "+str(rmse_for_test))
print("time in second ::::::::::::::::::::::::::::::::::"+str(t6))
```

```
RMSE For Train = 2.9373971388422278
RMSE For Test = 3.58357719381736
time in second ::::::::::::::::::::::::::::::::::0.00099945068359375
```

2.2.2 Random Forest Regressor

Random forest is an ensemble technique that consists of many decision tree. The idea behind random forest is to build n number of trees to have more accuracy in dataset. It is called random forest as we are building n no. of trees randomly. In the other words, to build the decision trees it selects randomly n no. of variables and n no of observations to build each decision tree. It means to build each decision tree on random forest we are not going to use the same data. The Code of the Random forest in Python is given below.

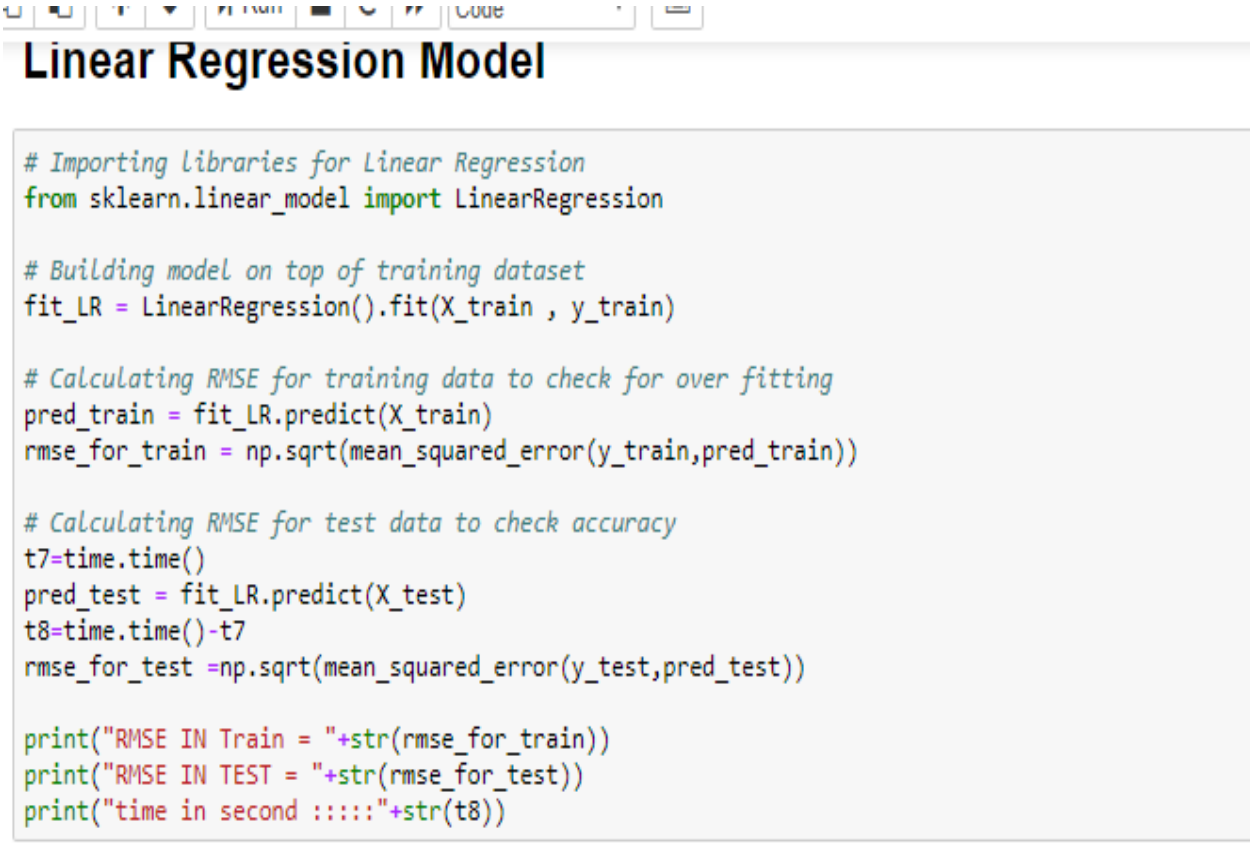
Random Forest Regressor

```
# Importing libraries for Random Forest
from sklearn.ensemble import RandomForestRegressor
import time

# Building model on top of training dataset
fit_RF = RandomForestRegressor(n_estimators = 500).fit(X_train,y_train)

# Calculating RMSE for training data to check for over fitting
pred_train = fit_RF.predict(X_train)
rmse_for_train = np.sqrt(mean_squared_error(y_train,pred_train))

# Calculating RMSE for test data to check accuracy
t1=time.time()
pred_test = fit_RF.predict(X_test)
t2=time.time()-t1
rmse_for_test =np.sqrt(mean_squared_error(y_test,pred_test))

print("RMSE  ForTrain = "+str(rmse_for_train))
print("RMSE for Test = "+str(rmse_for_test))
print("time in second :::::"+str(t2))
```

```
RMSE  ForTrain = 1.0575845469864322
RMSE for Test = 3.1714535770264067
time in second :::::0.09023404121398926
```

2.2.3 Linear regression

Linear Regression is one of the statistical methods of prediction. It is applicable only on continuous data. To build any model we have some assumption to put on data and model.



2.2.4 Gradient Boosting

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of ensemble model of weak prediction models, typically decision tree. It builds the model in a stage wise fashion like other boosting methods do, and it generalize them by assigning them by allowing optimization of an arbitrary differentiable loss function.

Gradient Boosting Regressor

```
# Importing library for GradientBoosting
from sklearn.ensemble import GradientBoostingRegressor

# Building model on top of training dataset
fit_GB = GradientBoostingRegressor().fit(X_train, y_train)

# Calculating RMSE for training data to check for over fitting
pred_train = fit_GB.predict(X_train)
rmse_for_train = np.sqrt(mean_squared_error(y_train,pred_train))

# Calculating RMSE for test data to check accuracy
t3=time.time()
pred_test = fit_GB.predict(X_test)
t4=time.time()-t3
rmse_for_test =np.sqrt(mean_squared_error(y_test,pred_test))

print("RMSE IN TRAIN = "+str(rmse_for_train))
print("RMSE IN Test = "+str(rmse_for_test))
print("time in second ::::::::::::::::::::::::::::::::::::"+str(t4))
```

```
RMSE IN TRAIN = 2.0359050616501393
RMSE IN Test = 3.2221479433906284
time in second ::::::::::::::::::::::::::::::::::::0.0020012855529785156
```

Best Model to Select on the basis of RMSE and time

Both R and Python

Algo. Name	Parameter Name	Python	R
Decision Tree	RMSE	3.2	FILE
	R^2	0.098	RULES.TXT
	MAE		RULES.TXT
Random Forest			
	RMSE	3.11	1.44
	R^2	0.17	0.85
	MAE		0.99

Linear Regresson	RMSE	41630980841	2.86
	R^2	-1.47	0.269
	MAE		2.06
gbR	RMSE	2.95	2.77
	R^2	0.255	0.27
	MAE		1.9

References:

[1] Missing Value Imputation

https://video.edwisor.com/video/z59imM8Isw

[2] Data Pre Processing

https://video.edwisor.com/video/z59imM8Isw

[3] Standardization and Normalization

https://www.dataschool.io/15-hours-of-expert-machine-learning-videos/

[4] https://github.com/arvindkumar09/Data-Science

[5] https://www.kaggle.com/learn/overview

[6] https://machinelearningmastery.com/machine-learning-in-python-step-by-step/

[7] https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/

[8] https://www.analyticsvidhya.com/blog/2016/07/deeper-regression-analysis-assumptions-plots-solutions/

[9] https://www.kaggle.com/dansbecker/learning-materials-on-kaggle

[10] https://scikit-learn.org/