Observation of dataset

1) At 1st glance three columns are not useful. EmployeeCount, EmployeeNumber and ID. EmployeeCount is same for all. EmployeeNumber and ID are not characteristics on which attrition of an employee will depend.

2) All columns with textual data is categorical.

3) About correlation!

   ➔ PercentSalaryHike and PerformanceRating are highly correlated

   "scipy.stats.pearsonr (data.PerformanceRating, data.PercentSalaryHike)" **result**: (**0.7713657093111319**, 1.3500749445473152e-203)

   Spearman **Result**: (correlation **= 0.6268477**, pvalue=2.49431543e-113)

   ➔ Similarly some more columns are interrelated, but only columns of x, attrition rate(y) is not related to any of x value.

   ➔ Montly income is highly related to Job level, Job Level and total working years, total working years and monthly income

Pre-Processing Methods:

1) **Data Understanding:**

   Glanced through data to understand various columns in data. Dropped columns which obviously do not affect y. For eg. EmployeeCount, EmployeeNumber and ID.

2) **Converting text to numbers:**

   Since I am using scikit-learn library, it only processes numbers.

   Converted columns having text to numbers:

   BusinessTravel, Department, EducationField, Gender, JobRole, MaritalStatus, OverTime

   Since all these columns were categorical , so easily converted them into numbers

3) **Normalisation:**

   Following given is range of dataset:

Age	DailyRate	DistanceFromHome	Education	EnvironmentSatisfaction
42	1397	28	4	3

HourlyRate	JobInvolvement	JobLevel	JobSatisfaction	MonthlyIncome
70	3	4	3	18990

Monthly Rate	NumCompanies Worked	PercentSalary Hike	Performance Rating	Relationship Satisfaction
24905	9	14	3	3

Since data has wide variation in range and therefore algorithm wasn't converging for logistic regression and linear SVC, and hence normalisation was necessary.

Also note for decison tree, normalisation is not necessary, therefore it converged.

4) PCA

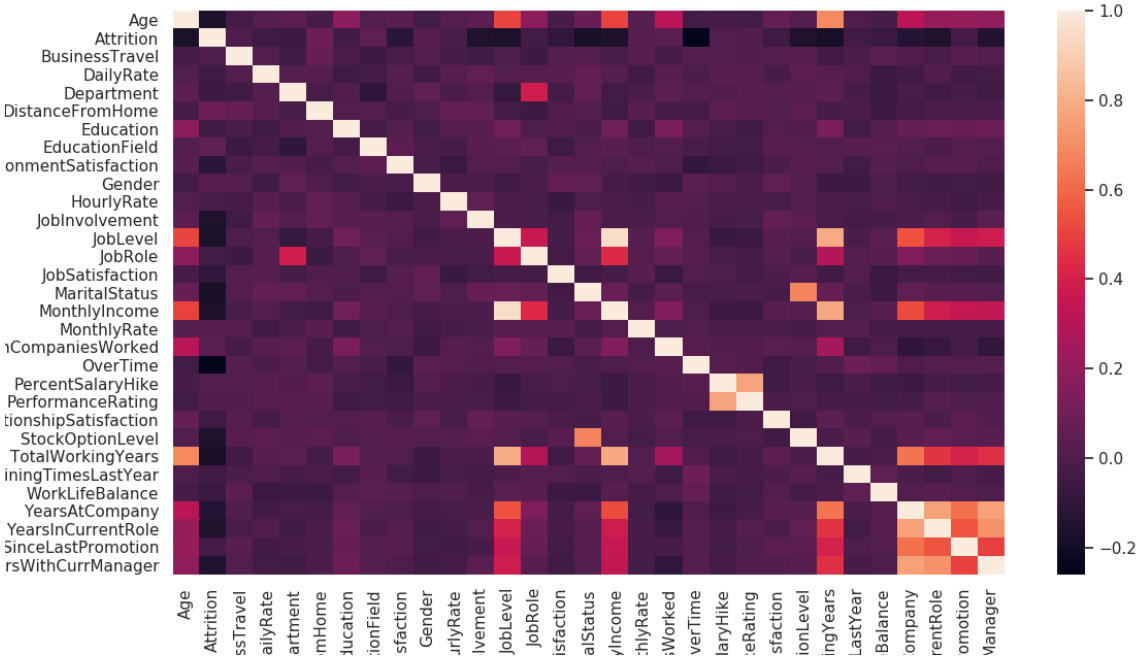
If number of components is reduced using PCA to any number below 29, for any number n the accuracy is comes out to 85.92%

Since it is less than 86.86%, it is not useful.

This can also be understood from correlation – heat map, as most of data columns are not highly correlated

Rather removing one or two column manually might work!

5) Visualisation and understanding correlation in data:



Approaches

1) After performing pre-processing methods 1 & 2, applied Logistic Regression and linear Support Vector Classification, both of these failed to converge even after using 10000 iterations.
2) However, Decision Tree algorithm worked giving accuracy of about 80% (submission 2 and 3 on Kaggle)
3) Did normalisation of data and applied Logistic Regression giving accuracy of around 86%.
4) and Linear SVC giving accuracy of about 89% (after normalisation)
5) Tried PCA to see whether accuracy is less due to it but was not successful
6) Random Forest: depth = 4 gave accuracy of 86.86%
7) Gradient Boosting: manipulated parameter: learning rates and n_estimator. This method gave highest accuracy of 91.4%

   where lr_rate:0.11 , n_estimator: 70

   Reason of Gradient Boosting being highest in accuracy:

   Ensemble machine learning methods are ones in which a number of predictors are aggregated to form a final prediction, which has lower bias and variance than any of the individual predictors. There are 2 type of ensembles: Bagging and Boosting.

   Gradient boosting produces an ensemble of decision trees that, on their own, are weak decision models.

n_estimator	learning_rate	accuracy_train	accuracy_validation
20	0.06	0.857	0.867
20	0.5	0.95	0.871
20	0.4	0.953	0.878
20	0.3	0.936	0.864
20	0.45	0.959	0.876
30	0.05	0.867	0.869
30	0.06	0.875	0.871
30	0.55	0.873	0.873
50	0.05	0.883	0.871
50	0.1	0.923	0.889

50	0.11	0.928	0.898
50	0.12	0.928	0.887
60	0.1	0.936	0.891
60	0.11	0.938	0.894
60	0.12	0.942	0.887
60	0.15	0.955	0.887
70	0.05	0.899	0.878
70	0.1	0.941	0.891
70	0.15	0.965	0.885
70	0.2	0.974	0.88
70	0.08	0.926	0.887
70	0.09	0.93	0.889
70	0.11	0.946	0.896
70	0.12	0.952	0.882
80	0.05	0.906	0.882
80	0.1	0.948	0.891
80	0.15	0.969	0.882
80	0.08	0.936	0.885
80	0.09	0.936	0.887
80	0.11	0.954	0.898
80	0.12	0.962	0.878

Gradient Boosting came out to be best after normalisation

****Removing monthly income column is improving result in each of the methods above and result of SVC and GradientBoosting is after removing Monthly Income.

Reason: it is highly Job level and TotalWorkingYears

Results and Final learning

Results:

Method	Accuracy
Decision Tree	80%
Logistic Regression (After Normalisation)	86%
Random Forest	86.86%
SVC (After Normalisation)	89%
Gradient Boosting	91.4%

Final Learning:

1) For the first time particilated in kaggle competitions
2) Learnt to use scikit-learn library
3) Learnt to use various python libraries such as pandas and matplotlib
4) Practical implementation of correlation graphs.
5) Tried Logistic Regression, Support Vector Classification, Decision Tree, Random Forest and Gradient Boosting; understood various parameters of these methods
6) Learnt Bagging and Boosting also
7) Understood importance of correlation, and importance of normalisation