People Analytics and Employee Retention

Project Report

Submitted by

Arpit Khurana (15BCE0353)

Ishas Prasad Diskshit (15BCE0436)

Yash Goil (15BCE0467)

In fulfillment for the project of

Artificial Intelligence

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

School of Computer Science and Engineering

DECLARATION

We hereby declare that the project entitled People Analytics and Employee Retention submitted by us to the School of Computer Science and Engineering, VIT University, Vellore-14 in fulfillment of the requirements for the award of the project of Artificial Intelligence in Computer Science and Engineering is a record of bonafide work carried out by us under the supervision of Geraldine Bessie Amali D, Associate Professor. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other project of this institute or of any other institute or university.

Arpit Khurana (15BCE0353)

Ishas Prasad Diskshit (15BCE0436)

Yash Goil (15BCE0467)

School of Computer Science and Engineering

CERTIFICATE

The project report entitled People Analytics and Employee Retention is prepared and submitted by Arpit Khurana (Register No: 15BCE0353) Ishas Prasad Diskshit (15BCE0436) and Yash Goil (15BCE0467). It has been found satisfactory in terms of scope, quality and presentation as partial fulfillment of the requirements for the award of the project of Artificial Intelligence in Computer Science and Engineering in VIT University, India.

Geraldine Bessie Amali D

Associate Professor

Guide

Index

1. Abstract

Employee attrition is very important. Companies spend lots of resources, money and valuable time to train their employees to get the best results and hence if the employee leaves the company due to some reason it is a great loss of human resource for the company. Therefore, understanding why employees leave is very important for businesses to maximize efficiency and reduce training cost which translates into greater profit. The following question is the one we would like answered: Can we predict future terminations? If so, how well can we predict?

Dataset:

Link of dataset: (https://www.kaggle.com/ludobenistant/hr-analytics/discussion/26565)

2. Introduction

Organizations spend heaps of assets, cash and profitable time to prepare their representatives to get the best outcomes and thus if the worker leaves the organization because of some reason it is an extraordinary loss of human asset for the organization. In this manner, understanding why representatives leave is vital for organizations to expand proficiency and lessen preparing cost which converts into more prominent benefit. This additionally helps in making a friendlier and vivacious work space for representatives to work in which brings about better efficiency and imparts a feeling of gratefulness for what work they are doing. Through this examination, we might want to think of the most ideal orders which could be utilized to anticipate if a specific representative will take off. From our analysis we will find what type of people leave their job. And what are the values of different parameters for these employees like they have high salary or low. With the help of our data companies can take steps to prevent their employees from leaving the company.

HR truly needs to start thinking outside of its traditional thinking and methodologies to powerfully address the HR challenges and issues in the future. On a personal level we like to think of People Analytics as when the artificial intelligence process is applied to HR information.

3. Hardware/Software Requirements

This requires the following hardware

CPU: Core 2 Duo/Athlon X2 or better

RAM: 1.5GB

Graphic Card: 512MB of Graphics Memory

Storage: 12GB

The software requirements for a modeling and simulation of AI needed to do this include:

Language: Python

Version: Python 2.7

Libraries:-

- numpy
- pandas
- seaborn
- matplotlib.pyplot
- sklearn
- sklearn.metrics
- sklearn.tree
- sklearn.ensemble
- csv
- keras.models
- keras.layers
- random
- math

4. Algorithms and Approach:

In our given dataset, we have data about the relations of every employee with each other. So, we first plot a graph which helped us to understand the different communities or group that has been built within the organizations. In our graph, the circles would represent people, and an edge between two vertices signifies that those two individuals are related to each other. This graph helped us understand if any employee feels left alone or if some employee is working with those people who are not within his domain of work. Then we also have the data of the employee about their satisfaction level, Last evaluation, Time since last performance evaluation (in Years), Number of projects completed while at work, Average monthly hours at workplace, Number of years spent in the company, Whether the employee had a workplace accident, Whether the employee left the workplace or not, Whether the employee was promoted in the last five years, Department in which they work for. We have used pandas to load the data in tabular form.

Then we obtained the number of employees who have left the company. We plotted this in a pie chart. Then we found the satisfaction levels of employee and plot it on bar graph. We also, plotted the number of work accidents by users. We plotted the number of employees who were going to be promoted and who decided to leave their department. From, these graphs and plots we concluded what type of people left their job. And what are the values

of different parameters for these employees like they have high salary or low. Finally, from this data, companies can take steps to prevent their employees from leaving the company.

5. Existing Models

Authors and Year (Reference)	*Title (Study)*	*Concept / Theoretical model/ Framework*	*Methodology used/ Implementation*	*Dataset details/ Analysis*	*Relevant Finding*	*Limitations/ Future Research/ Gaps identified*
(http://ieeexplore.ieee.org/document/7873830/)	Simulation of marketplace customer satisfaction analysis based on machine learning algorithms	Sentiment analysis have influence and benefits, which is to obtain information about public sentiment towards companies . As previous research associated with Sentiment Analysis, among others, research detects fake websites or sites with the original classification of news articles on the website. Research [to analyze sentiment on twitter text, using the n-gram language characters and SVM models to cope with high lexical variation in Twitter text. Research developed a system that can identify and classify public sentiment to predict interesting products in marketing.	Sentiment analysis system developed as a system that can classify the type opinion into positive, negative, and neutral by using five methods. The methods used for different machine learning classifiers, consisting of K-Nearest Neighbor, Logistic Regression, Naïve Bayes, Support Vector Machine, and Random Forest	Twitter can be a source of public opinion data and sentiment. Such data can be used efficiently for marketing or social studies. In this research addresses this issue by measuring net sentiment based on customer satisfaction through customer's sentiment analysis from Twitter data.	Performance analysis for data service in third generation mobile telecommunication networks Design and performance modeling & simulation of self-healing mechanisms for wireless communication networks	The results showed Support Vector Machine has accuracy 80.8% with 1000 sampling dataset and 85.4% with 2000 sampling dataset. Logistic Regression has accuracy 78.8% with 1000 sampling dataset and 82.9% with 2000 sampling dataset. This shows that the more the number of training data will improve the accuracy of the system.
http://ieeexplore.ieee.org/document/4419482/	Evaluating customer satisfaction using fuzzy model based on flexible expert weight	A flexible expert weight fuzzy model for evaluating the customer satisfaction applied on any kind of industries is proposed, which can be used as a tool to survey customer satisfaction. This model applies the fuzzy sets theory which is first proposed by Zadeh in 1965 to deal with the vague situation while the experts evaluating the custom satisfaction. In the meantime, the flexible expert weight is also considered. In brief, the aims of this paper are (a) construct the fuzzy model based on flexible expert weight for estimating customer satisfaction, (b) Using numerical example to verify the results and show its advantages	The processes of fuzzy model based on flexible expert weight for estimating customer satisfaction are depicted as follows • Determine the evaluation factor. • Evaluate the relative importance among the factors. Since every factor may have different importance in different experts, so, it is necessary to compute the relative importance among the factors. - Rating customer satisfaction - Computing the aggregate score. Finally, the estimate of aggregate score of satisfaction level	Many literatures reveal that the statistical methods are used to assess the level of customer satisfaction; the common statistical methods can generate some useful statistical data and inferences. However, among all the ways of gathering statistics and analysing, we are unable to deal with the problems that facing evaluating linguistic terms, such as good, comfortable	Scenario-based modeling and its applications Optimal tactile sensor placement	Among all the ways of gathering statistics and analysing, we are unable to deal with the problems that facing evaluating linguistic terms, such as good, comfortable, etc. Furthermore, while evaluation is conducted, most researches assume that all expert's weights are equal. In fact, it is more reasonable that different expert should have different weight depending on there experiences

Authors and Year (Reference)	*Title (Study)*	*Concept / Theoretical model/ Framework*	*Methodology used/ Implementation*	*Dataset details/ Analysis*	*Relevant Finding*	*Limitations/ Future Research/ Gaps identified*
http://ieeexplore.ieee.org/document/5998761/	The Customer Satisfaction Assessment Research of YTO Express Wuchang Branch	Based on the aforementioned customer satisfaction theory [4] [5]and third party logistics industry customer satisfaction model, coupled with the views of Yuantong Express employees, some customer views on the scene, the customer satisfaction model of YEWB has been established	Considering the subject and multi-factor characteristics of customer satisfaction, it is difficult to make direct quantitative measurement, while AHP is a quantitative and qualitative method, highly logic and concise, so this paper uses AHP method to determine the weight of evaluation indicators.	The questionnaire objects are mainly customers of Yuantong Express, and site courier business consulting customers; use random sampling method to carry out data collection. Seen from the questionnaire collation, most customers aged from 20 to 35, showing the customers are mainly young people and the online shopping accounting for the vast majority, which reflects the tremendous courier business opportunities brought bye-commerce	A scheduling technique providing a strict isolation of real-time threads IT infrastructure management and standards	
http://ieeexplore.ieee.org/abstract/document/5460732/	Forecasting Employee Retention Probability	The goal is to identify potential employees who are likely to stay with the organization during the next year based on previous year data. Neural networks can help organizations to properly address the issue. To solve this problem a neural network should be trained to perform correct classification between employees. After the network has been properly trained, it can be used to identify employees who intent to leave and take the appropriate measures to retain them	All This Neural networks are most effective and appropriate artificial intelligence technology for pattern recognition. Superior results in pattern recognition can be directly applied for business purposes in forecasting, classification and data analysis [1]. This new approach gives an extra advantage in solving "real-world" problems in business and engineering.	The collected data using the Twitter public API which allows developers to extract tweets from twitter programmatically . The collected data, because of the random and casual nature of tweeting, need to be filtered to remove unnecessary information. Filtering out these and other problematic tweets such as redundant ones, and ones with no proper sentences was done next.		For the research we decided to focus on one nation, USA. We extracted tweets from seven major cities in the USA. The choice of location is very limited mainly due to data availability and language constraints. We decided to go with data from New York, Los Angeles, Boston, Chicago, Dallas, San Francisco and Philadelphia for the experiments.

Authors and Year (Reference)	*Title (Study)*	*Concept / Theoretical model/ Framework*	*Methodology used/ Implementation*	*Dataset details/ Analysis*	*Relevant Finding*	*Limitations/ Future Research/ Gaps identified*
http://ieeexplore.ieee.org/document/7916264/	A framework for evaluating customer satisfaction	The information transmission routine is becoming more and more mutually connected in social media rather than totally independent in traditional media [5], [6]. Social network is the fundamental of social interaction which provides a powerful platform social activities and social media analytics. The features of graph including degree, density, centrality, closeness, betweenness, geodesic, and matrix will be applied to implement social network analysis. In this paper, we will focus on the social media analytics which contains other more human interactions and activities.	To extract useful information from a large amount of data being collected in social medias and then to analyze, summarize and generalize it for providing useful information for particular users. Since opinion words are usually the dominating elements for sentiment classification, we choose lexicon-based approach that is an unsupervised method to implement our social analytic. Lexicon-based approach includes two methods as well: dictionary-based approach and corpus-based approach. The comments on each airline are classified into three categories: positive sentiment, neutral sentiment and negative sentiment by the method of sentiments classification	Propose a framework to evaluate customer satisfaction on the basis of the data from social media platform and the technology of sentiment analysis. For better demonstrate our approach, we take six American airlines as example to explain the process minutely and our source data is collected from Twitter. Representative conclusions and decisions must be based on high quality data. As a technique for enhancing data quality, data preprocess is also used in this work.	Improving smart card security using self-timed circuits IT infrastructure management and standards	

5. Developed Model (Design and Module Wise Description)

I) SVM

"Support Vector Machine" (SVM) is a supervised machine learning algorithm which can be used for both classification and regression challenges. However, it is mostly used in classification problems. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiate the two classes very well.

- It doesn't perform well, when we have large data set because the required training time is higher
- It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping
- SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is related SVC method of Python scikit-learn library.

II) Decision Tree

- ID3 algorithm begins with the original set S as the root node.
- On each iteration of the algorithm, it iterates through every unused attribute of the set S and calculates the entropy H(S) (or information gain IG(S)) of that attribute.
- It then selects the attribute which has the smallest entropy (or largest information gain) value.
- The set S is then split by the selected attribute (e.g. age is less than 50, age is between 50 and 100, age is greater than 100) to produce subsets of the data.
- The algorithm continues to recur on each subset, considering only attributes never selected before.
- Recursion on a subset may stop, when
- All the elements in the class belong to same class
- All instances does not belong to same class but there is no attribute to select
- There is no example in the subset
- Steps in ID3
- Calculate the entropy of every attribute using the data set S
- Split the set S into subsets using the attribute for which the resulting entropy (after splitting) is minimum (or, equivalently, information gain is maximum)
- Make a decision tree node containing that attribute
- Recurs on subsets using remaining attributes.

III) Random Forest:

Random Forests grows many classification trees. To classify a new object from an input vector, put the input vector down each of the trees in the forest. Each tree gives a classification, and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

Each tree is grown as follows:

1. If the number of cases in the training set is N, sample N cases at random - but with replacement, from the original data. This sample will be the training set for growing the tree.
2. If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
3. Each tree is grown to the largest extent possible. There is no pruning.

When the training set for the current tree is drawn by sampling with replacement, about one-third of the cases are left out of the sample. This oob (out-of-bag) data is used to get a running unbiased estimate of the classification error as trees are added to the forest. It is also used to get estimates of variable importance.

After each tree is built, all of the data are run down the tree, and proximities are computed for each pair of cases. If two cases occupy the same terminal node, their proximity is increased by one. At the end of the run, the proximities are normalized by dividing by the number of trees. Proximities are used in replacing missing data, locating outliers, and producing illuminating low-dimensional views of the data.

IV) Artificial Neural Network (Back Propagation):

Back Propagation has 2 phases

Forward pass phase: Computes 'functional signal', feed forward propagation of input pattern signals through network

Backward pass phase: Computes 'error signal', propagates the error backwards through network starting at output units (where the error is the difference between actual and desired output values)

- Step 1: Initialize all weights to small random values.
- Step 2: Choose an input-output training pair.
- Step 3: Calculate the actual output from each neuron in a layer by propagating the signal forward through the network layer by layer (forward propagation).
- Step 4: Compute the error value and error signals for output layer.
- Step 5: Propagate the errors backward to update the weights and compute the error signals for the preceding layers.
- Step 6: Check whether the whole set of training data have been cycled once, yes – go to step 7; otherwise go to step 2.
- Step 7: Check whether the current total error is acceptable; yes- terminate the training process and output the field weights, otherwise initiate a new training epoch by going to step 2.

6. Results and Discussion

In this project we have used various algorithms to analyse why employees leave. We have used SVC, Random Forest, Decision Tree and Artificial Neural Network. With SVC we obtained an accuracy of 95.6 %, with Random Forest, an accuracy of 98.8 %, with Decision Tree 98.16 % and with the Artificial Neural Network an accuracy of 96.62 %. Random Forest was the most accurate of all the algorithms.

Analysis of given Dataset

• Draw graphs for feature extraction

• Train model on the given dataset using SVC and Random forest Classifiers

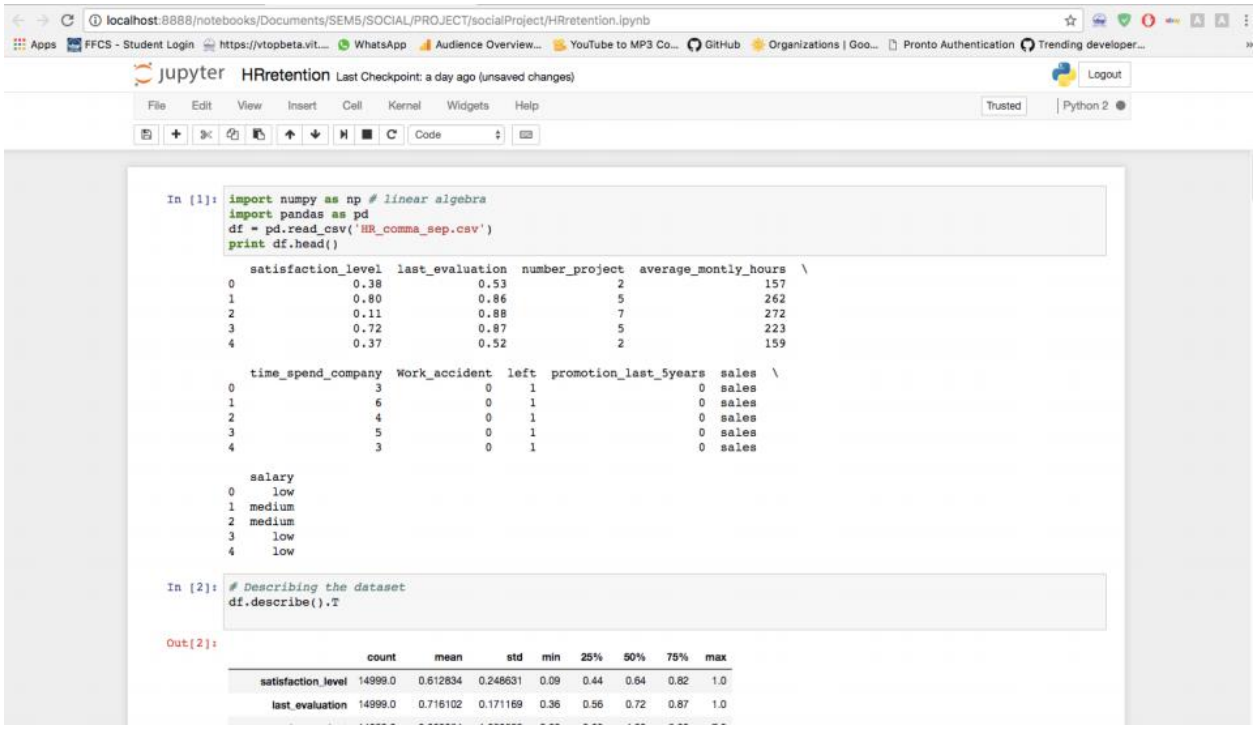
• Predict whether the employee is going to leave job or not, based on our trained model.

• Identified what may be the consequences, he is facing, which will be useful for employee retention.

7. Conclusion

From our analysis we found what type of people leave their job. And what are the values of different parameters for these employees like they have high salary or low. With the help of data companies can take steps to prevent their employees from leaving the company using algorithms in machine learning like SVC, Decision Tree, Random Forest and Artificial Neural Network.

Appendix



localhost:8888/notebooks/Documents/SEM5/SOCIAL/PROJECT/socialProject/HRretention.ipynb

jupyter HRretention Last Checkpoint: a day ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 2

Code

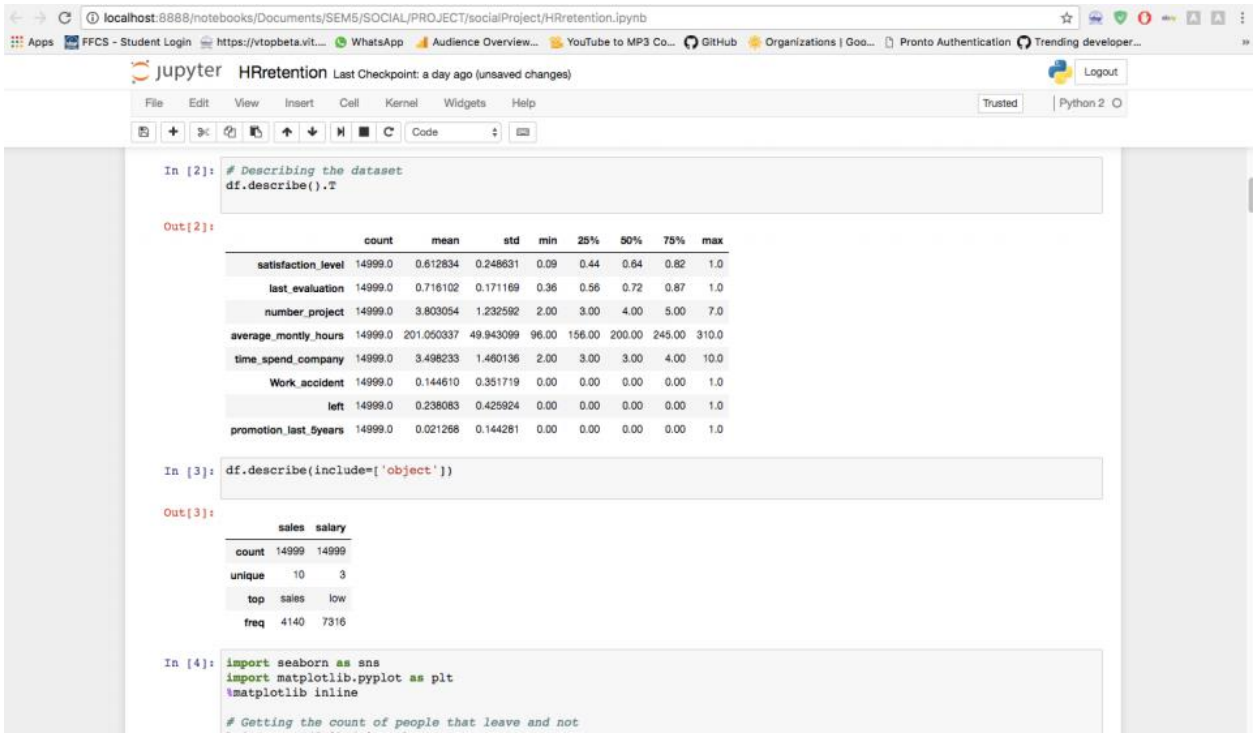
In [1]:
```
import numpy as np # linear algebra
import pandas as pd
df = pd.read_csv('HR_comma_sep.csv')
print df.head()
```

```
   satisfaction_level  last_evaluation  number_project  average_montly_hours  \
0                0.38             0.53               2                   157
1                0.80             0.86               5                   262
2                0.11             0.88               7                   272
3                0.72             0.87               5                   223
4                0.37             0.52               2                   159

   time_spend_company  Work_accident  left  promotion_last_5years  sales  \
0                   3              0     1                      0  sales
1                   6              0     1                      0  sales
2                   4              0     1                      0  sales
3                   5              0     1                      0  sales
4                   3              0     1                      0  sales

   salary
0     low
1  medium
2  medium
3     low
4     low
```

In [2]:
```
# Describing the dataset
df.describe().T
```

Out[2]:

	count	mean	std	min	25%	50%	75%	max
satisfaction_level	14999.0	0.612834	0.248631	0.09	0.44	0.64	0.82	1.0
last_evaluation	14999.0	0.716102	0.171169	0.36	0.56	0.72	0.87	1.0



localhost:8888/notebooks/Documents/SEM5/SOCIAL/PROJECT/socialProject/HRretention.ipynb

jupyter HRretention Last Checkpoint: a day ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 2

Code

In [2]:
```
# Describing the dataset
df.describe().T
```

Out[2]:

	count	mean	std	min	25%	50%	75%	max
satisfaction_level	14999.0	0.612834	0.248631	0.09	0.44	0.64	0.82	1.0
last_evaluation	14999.0	0.716102	0.171169	0.36	0.56	0.72	0.87	1.0
number_project	14999.0	3.803054	1.232592	2.00	3.00	4.00	5.00	7.0
average_montly_hours	14999.0	201.050337	49.943099	96.00	156.00	200.00	245.00	310.0
time_spend_company	14999.0	3.498233	1.460136	2.00	3.00	3.00	4.00	10.0
Work_accident	14999.0	0.144610	0.351719	0.00	0.00	0.00	0.00	1.0
left	14999.0	0.238083	0.425924	0.00	0.00	0.00	0.00	1.0
promotion_last_5years	14999.0	0.021268	0.144281	0.00	0.00	0.00	0.00	1.0

In [3]:
```
df.describe(include=['object'])
```

Out[3]:

	sales	salary
count	14999	14999
unique	10	3
top	sales	low
freq	4140	7316

In [4]:
```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

# Getting the count of people that leave and not
```

Apps FFCS - Student Login https://vtopbeta.vit.... WhatsApp Audience Overview... YouTube to MP3 Co... GitHub Organizations | Goo... Pronto Authentication Trending developer...

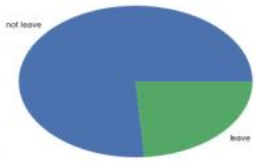
jupyter **HRretention** Last Checkpoint: a day ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 2

Code

In [4]:
```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

# Getting the count of people that leave and not
leftcounts=df['left'].value_counts()
print(leftcounts)

# Using matplotlib pie chart and label the pie chart
plt.pie(leftcounts,labels=['not leave','leave']);
```

```
0    11428
1     3571
Name: left, dtype: int64
```



In [5]:
```
leftdf = df[df['left']==1]
notleftdf =df[df['left']==0]
```

In [6]:
```
# Create a figure instance, and the two subplots
fig = plt.figure(figsize=(20,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)
```

PROJECT/socialProject/ HRretention Vinit

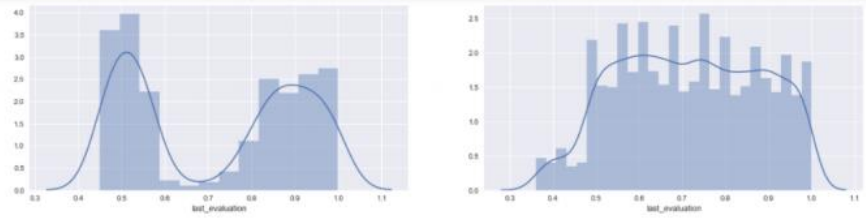
localhost:8888/notebooks/Documents/SEM5/SOCIAL/PROJECT/socialProject/HRretention.ipynb

Apps FFCS - Student Login https://vtopbeta.vit.... WhatsApp Audience Overview... YouTube to MP3 Co... GitHub Organizations | Goo... Pronto Authentication Trending developer...

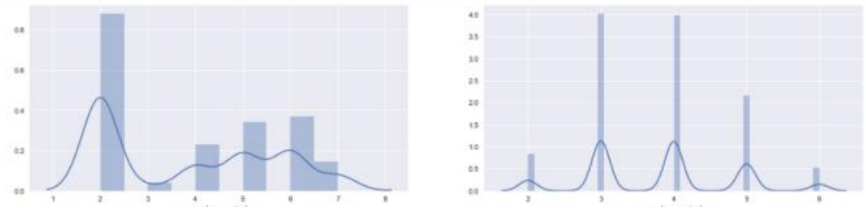
jupyter **HRretention** Last Checkpoint: a day ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 2

Code



In [8]:
```
# Create a figure instance, and the two subplots
fig = plt.figure(figsize=(20,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)

sns.distplot(leftdf['number_project'], kde=True,ax=ax1);

sns.distplot(notleftdf['number_project'], kde=True,ax=ax2);
```



localhost:8888/notebooks/Documents/SEM5/SOCIAL/PROJECT/socialProject/HRretention.ipynb

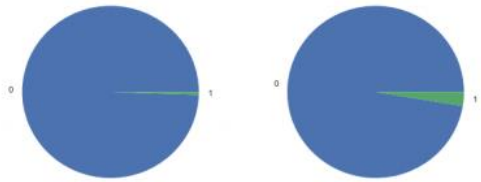
Apps FFCS - Student Login https://vtopbeta.vit.... WhatsApp Audience Overview... YouTube to MP3 Co... GitHub Organizations | Goo... Pronto Authentication Trending developer...

jupyter HRretention Last Checkpoint: a day ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 2

Code

In [12]:
```
# create a figure with two subplots
fig = plt.figure(figsize=(10,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)

leftpromotioncounts = leftdf['promotion_last_5years'].value_counts()
notleftpromotioncounts = notleftdf['promotion_last_5years'].value_counts()


# plot each pie chart in a separate subplot
ax1.pie(leftpromotioncounts,labels=leftpromotioncounts.index);
ax2.pie(notleftpromotioncounts,labels=notleftpromotioncounts.index);
```



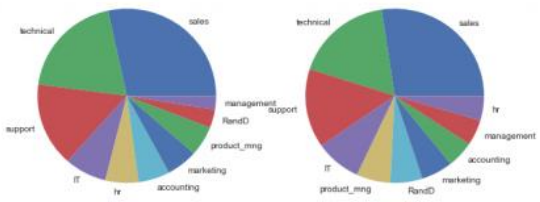
In [13]:
```
# create a figure with two subplots
fig = plt.figure(figsize=(10,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)

leftdepartmentcounts = leftdf['sales'].value_counts()
notleftdepartmentcounts = notleftdf['sales'].value_counts()
```

jupyter HRretention Last Checkpoint: a day ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 2

Code

In [13]:
```
# create a figure with two subplots
fig = plt.figure(figsize=(10,10))
ax1 = fig.add_subplot(221)
ax2 = fig.add_subplot(222)

leftdepartmentcounts = leftdf['sales'].value_counts()
notleftdepartmentcounts = notleftdf['sales'].value_counts()


# plot each pie chart in a separate subplot
ax1.pie(leftdepartmentcounts,labels=leftdepartmentcounts.index);
ax2.pie(notleftdepartmentcounts,labels=notleftdepartmentcounts.index);
```



Profile of the people who left

- satisfaction_level Not much pattern there
- last_evaluation High and low
- number_project Most have 2 but there are also overworked people (maybe depends on their job requirements)
- average_montly_hours It has either large or not so much average worked hours
- time_spend_company Spreaded out 2 to 5
- Work_accident Does not have much work accidents

localhost:8888/notebooks/Documents/SEM5/SOCIAL/PROJECT/socialProject/HRretention.ipynb

jupyter HRretention Last Checkpoint: a day ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 2

Code

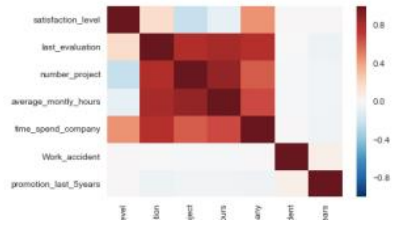
Profile of the people who left

- satisfaction_level Not much pattern there
- last_evaluation High and low
- number_project Most have 2 but there are also overworked people (maybe depends on their job requirements)
- average_montly_hours It has either large or not so much average worked hours
- time_spend_company Spreaded out 2 to 5
- Work_accident Does not have much work accidents
- promotion_last_5years Lot less not promoted
- sales Does not differ much
- salary Most of them are at the lower level salary (low)

Correlation Analysis¶

In [14]:
```
corr = leftdf.drop('left',axis=1).corr()
sns.heatmap(corr)
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x115b9cb90>
```



localhost:8888/notebooks/Documents/SEM5/SOCIAL/PROJECT/socialProject/HRretention.ipynb

jupyter HRretention Last Checkpoint: a day ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 2

Code

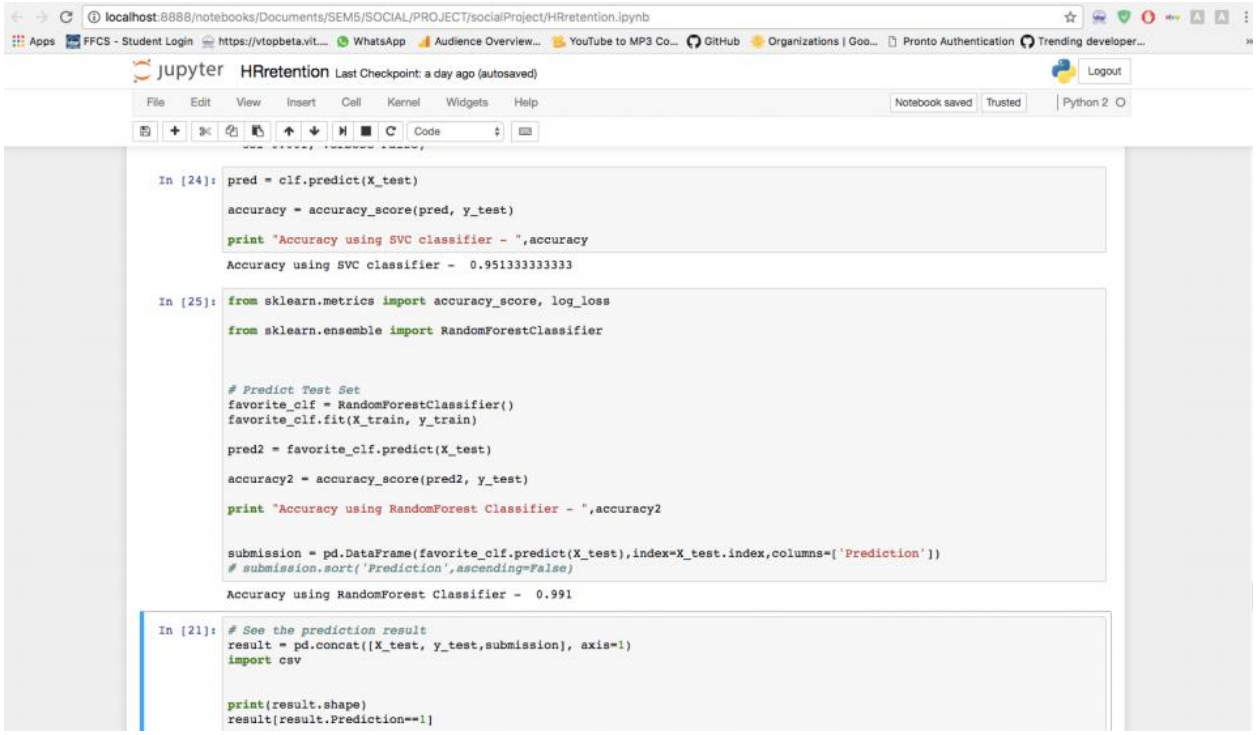
In [18]:
```
# initiate svm object
clf = svm.SVC()
# Fit the svm object with train data
clf.fit(X_train,y_train)
```

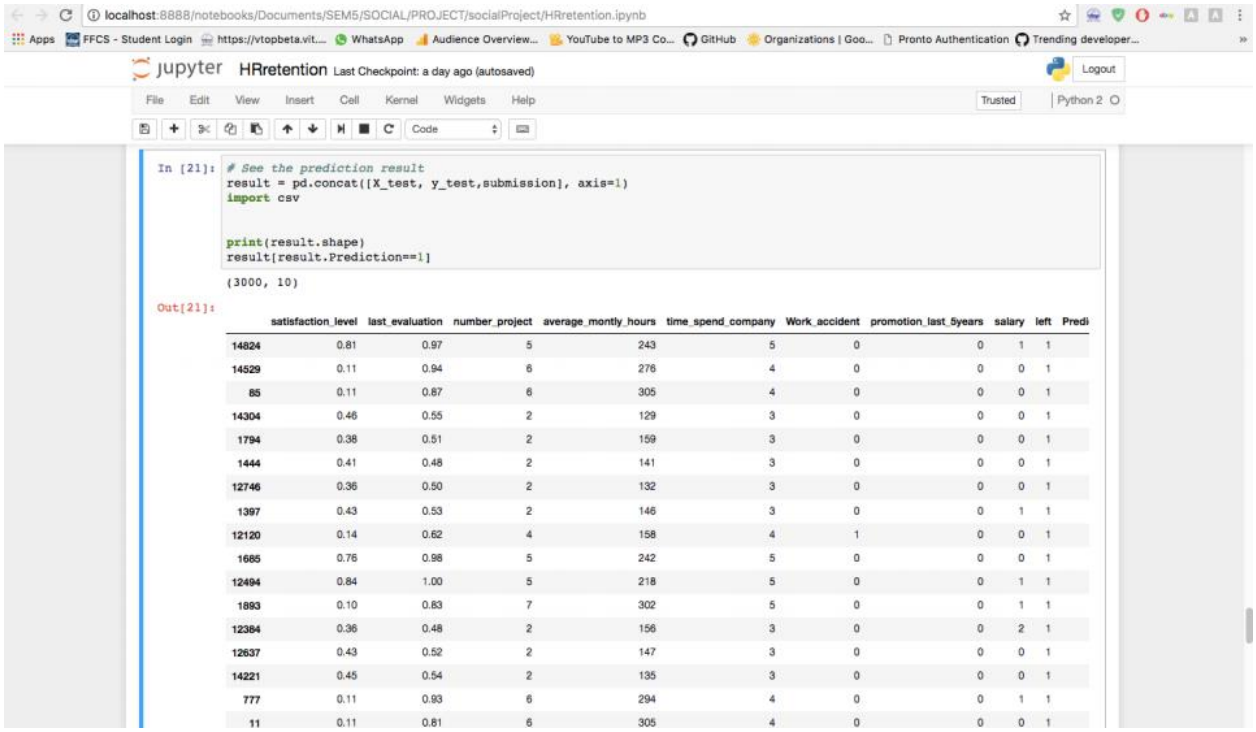
Out[18]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape=None, degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
```

In [19]:
```
pred = clf.predict(X_test)

accuracy = accuracy_score(pred, y_test)

print accuracy
```

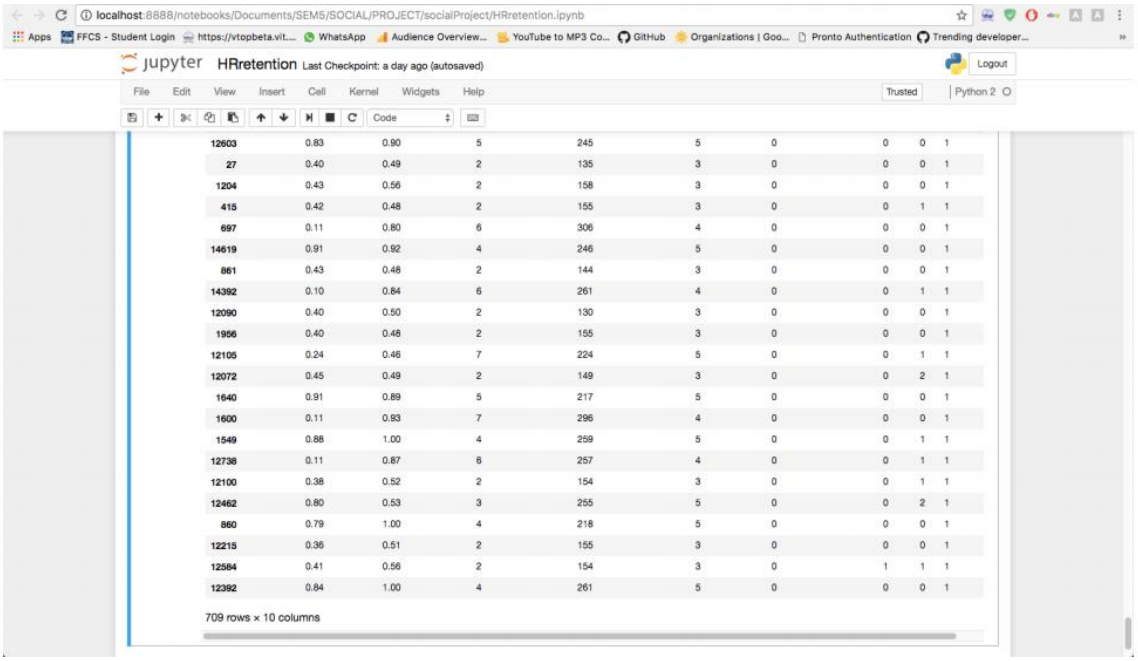
```
0.949333333333
```

In [20]:
```
from sklearn.metrics import accuracy_score, log_loss

from sklearn.ensemble import RandomForestClassifier



# Predict Test Set
favorite_clf = RandomForestClassifier()
favorite_clf.fit(X_train, y_train)

pred2 = favorite_clf.predict(X_test)

accuracy2 = accuracy_score(pred2, y_test)

print "Accuracy - ",accuracy2


submission = pd.DataFrame(favorite_clf.predict(X_test),index=X_test.index,columns=['Prediction'])
```





	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	promotion_last_5years	salary	left	Predi
14824	0.81	0.97	5	243	5	0	0	1	1	
14529	0.11	0.94	6	276	4	0	0	0	1	
85	0.11	0.87	6	305	4	0	0	0	1	
14304	0.46	0.55	2	129	3	0	0	0	1	
1794	0.38	0.51	2	159	3	0	0	0	1	
1444	0.41	0.48	2	141	3	0	0	0	1	
12746	0.36	0.50	2	132	3	0	0	0	1	
1397	0.43	0.53	2	146	3	0	0	1	1	
12120	0.14	0.62	4	158	4	1	0	0	1	
1685	0.76	0.98	5	242	5	0	0	0	1	
12494	0.84	1.00	5	218	5	0	0	1	1	
1893	0.10	0.83	7	302	5	0	0	1	1	
12384	0.36	0.48	2	156	3	0	0	2	1	
12637	0.43	0.52	2	147	3	0	0	0	1	
14221	0.45	0.54	2	135	3	0	0	0	1	
777	0.11	0.93	6	294	4	0	0	1	1	
11	0.11	0.81	6	305	4	0	0	0	1	



12603	0.83	0.90	5	245	5	0	0	0	1
27	0.40	0.49	2	135	3	0	0	0	1
1204	0.43	0.56	2	158	3	0	0	0	1
415	0.42	0.48	2	155	3	0	0	1	1
697	0.11	0.80	6	306	4	0	0	0	1
14619	0.91	0.92	4	246	5	0	0	0	1
861	0.43	0.48	2	144	3	0	0	0	1
14392	0.10	0.84	6	261	4	0	0	1	1
12090	0.40	0.50	2	130	3	0	0	0	1
1956	0.40	0.48	2	155	3	0	0	0	1
12105	0.24	0.46	7	224	5	0	0	1	1
12072	0.45	0.49	2	149	3	0	0	2	1
1640	0.91	0.89	5	217	5	0	0	0	1
1600	0.11	0.93	7	296	4	0	0	0	1
1549	0.88	1.00	4	259	5	0	0	1	1
12738	0.11	0.87	6	257	4	0	0	1	1
12100	0.38	0.52	2	154	3	0	0	1	1
12462	0.80	0.53	3	255	5	0	0	2	1
860	0.79	1.00	4	218	5	0	0	0	1
12215	0.36	0.51	2	155	3	0	0	0	1
12584	0.41	0.56	2	154	3	0	1	1	1
12392	0.84	1.00	4	261	5	0	0	0	1

709 rows × 10 columns

In [5]:
```
from keras.models import Sequential
from keras.layers import Dense
import numpy
# fix random seed for reproducibility
numpy.random.seed(7)
```

```
Using TensorFlow backend.
```

In [7]:
```
dataset = numpy.loadtxt("data/HR_comma.csv", delimiter=",")
# split into input (X) and output (Y) variables
X = dataset[:,0:8]
Y = dataset[:,8]
```

In [8]:
```
model = Sequential()
model.add(Dense(32, input_dim=8, activation='sigmoid'))
model.add(Dense(8, activation='sigmoid'))
model.add(Dense(1, activation='sigmoid'))
# Compile model
model.compile(loss='binary_crossentropy', optimizer='rmsprop', metrics=['accuracy'])
# Fit the model
model.fit(X, Y, epochs=150, batch_size=10)
```

```
14999/14999 [==============================] - 10s 644us/step - loss: 0.1623 - acc: 0.9507
Epoch 145/150
14999/14999 [==============================] - 9s 589us/step - loss: 0.1603 - acc: 0.9501
Epoch 146/150
14999/14999 [==============================] - 8s 566us/step - loss: 0.1601 - acc: 0.9515
Epoch 147/150
14999/14999 [==============================] - 8s 559us/step - loss: 0.1611 - acc: 0.9503
Epoch 148/150
14999/14999 [==============================] - 9s 572us/step - loss: 0.1611 - acc: 0.9516
Epoch 149/150
14999/14999 [==============================] - 8s 565us/step - loss: 0.1570 - acc: 0.9513
Epoch 150/150
14999/14999 [==============================] - 8s 558us/step - loss: 0.1588 - acc: 0.9509
```

Out[8]:
```
<keras.callbacks.History at 0x7fb4d7a93450>
```

In [10]:
```
scores = model.evaluate(X, Y)
print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
```

```
14999/14999 [==============================] - 1s 53us/step

acc: 95.61%
```

In [25]:
```
# Backprop on the Seeds Dataset
from random import seed
from random import randrange
from random import random
from csv import reader
from math import exp

network = list()
```

In [26]:
```
# Load a CSV file
def load_csv(filename):
	dataset = list()
	with open(filename, 'r') as file:
		csv_reader = reader(file)
		for row in csv_reader:
			if not row:
				continue
			dataset.append(row)
	return dataset
```

In [27]:
```
# Convert string column to float
def str_column_to_float(dataset, column):
	for row in dataset:
		row[column] = float(row[column].strip())
```

In [28]:
```
# Convert string column to integer
def str_column_to_int(dataset, column):
	class_values = [row[column] for row in dataset]
	unique = set(class_values)
	lookup = dict()
	for i, value in enumerate(unique):
		lookup[value] = i
	for row in dataset:
		row[column] = lookup[row[column]]
	return lookup
```

In [29]:
```
# Find the min and max values for each column
def dataset_minmax(dataset):
	minmax = list()
	stats = [[min(column), max(column)] for column in zip(*dataset)]
	return stats
```

In [30]:
```
# Rescale dataset columns to the range 0-1
def normalize_dataset(dataset, minmax):
	for row in dataset:
		for i in range(len(row)-1):
			row[i] = (row[i] - minmax[i][0]) / (minmax[i][1] - minmax[i][0])
```

In [44]:
```
# Split a dataset into k folds
def cross_validation_split(dataset, n_folds):
	dataset_split = list()
	dataset_copy = list(dataset)
	fold_size = int(len(dataset) / n_folds)
	for i in range(n_folds):
		fold = list()
		while len(fold) < fold_size:
			index = randrange(len(dataset_copy))
			fold.append(dataset_copy.pop(index))
		dataset_split.append(fold)
	return dataset_split
```

In [31]:
```
# Calculate accuracy percentage
def accuracy_metric(actual, predicted):
	correct = 0
	for i in range(len(actual)):
		if actual[i] == predicted[i]:
			correct += 1
	return correct / float(len(actual)) * 100.0
```

In [43]:
```
# Evaluate an algorithm using a cross validation split
def evaluate_algorithm(dataset, algorithm, n_folds, *args):
	folds = cross_validation_split(dataset, n_folds)
	scores = list()
	for fold in folds:
		train_set = list(folds)
		train_set.remove(fold)
		train_set = sum(train_set, [])
		test_set = list()
		for row in fold:
			row_copy = list(row)
			test_set.append(row_copy)
			row_copy[-1] = None
		predicted = algorithm(train_set, test_set, *args)
		actual = [row[-1] for row in fold]
		accuracy = accuracy_metric(actual, predicted)
		print accuracy
		scores.append(accuracy)
	return scores
```

In [32]:
```
# Calculate neuron activation for an input
def activate(weights, inputs):
	activation = weights[-1]
	for i in range(len(weights)-1):
		activation += weights[i] * inputs[i]
	return activation
```

In [33]:
```
# Transfer neuron activation
def transfer(activation):
	return 1.0 / (1.0 + exp(-activation))
```

In [34]:
```
# Forward propagate input to a network output
def forward_propagate(network, row):
        inputs = row
        for layer in network:
                new_inputs = []
                for neuron in layer:
                        activation = activate(neuron['weights'], inputs)
                        neuron['output'] = transfer(activation)
                        new_inputs.append(neuron['output'])
                inputs = new_inputs
        return inputs
```

In [35]:
```
# Calculate the derivative of an neuron output
def transfer_derivative(output):
        return output * (1.0 - output)
```

In [36]:
```
# Backpropagate error and store in neurons
def backward_propagate_error(network, expected):
        for i in reversed(range(len(network))):
                layer = network[i]
                errors = list()
                if i != len(network)-1:
                        for j in range(len(layer)):
                                error = 0.0
                                for neuron in network[i + 1]:
                                        error += (neuron['weights'][j] * neuron['delta'])
                                errors.append(error)
                else:
                        for j in range(len(layer)):
                                neuron = layer[j]
                                errors.append(expected[j] - neuron['output'])
                for j in range(len(layer)):
                        neuron = layer[j]
                        neuron['delta'] = errors[j] * transfer_derivative(neuron['output'])
```

In [37]:
```
# Update network weights with error
def update_weights(network, row, l_rate):
        for i in range(len(network)):
                inputs = row[:-1]
                if i != 0:
                        inputs = [neuron['output'] for neuron in network[i - 1]]
                for neuron in network[i]:
                        for j in range(len(inputs)):
                                neuron['weights'][j] += l_rate * neuron['delta'] * inputs[j]
                        neuron['weights'][-1] += l_rate * neuron['delta']
```

In [38]:
```
# Train a network for a fixed number of epochs
def train_network(network, train, l_rate, n_epoch, n_outputs):
        for epoch in range(n_epoch):
                for row in train:
                        outputs = forward_propagate(network, row)
                        expected = [0 for i in range(n_outputs)]
                        expected[row[-1]] = 1
                        backward_propagate_error(network, expected)
                        update_weights(network, row, l_rate)
```

In [39]:
```
# Initialize a network
def initialize_network(n_inputs, n_hidden, n_outputs):
    global network
    network = list()
    hidden_layer = [{'weights':[random() for i in range(n_inputs + 1)]} for i in range(n_hidden)]
    network.append(hidden_layer)
    output_layer = [{'weights':[random() for i in range(n_hidden + 1)]} for i in range(n_outputs)]
    network.append(output_layer)
    return network
```

In [40]:
```
# Make a prediction with a network
def predict(network, row):
        outputs = forward_propagate(network, row)
        return outputs.index(max(outputs))
```

In [45]:
```
def back_propagation(train, test, l_rate, n_epoch, n_hidden):
     n_inputs = len(train[0]) - 1
     n_outputs = len(set([row[-1] for row in train]))
     network = initialize_network(n_inputs, n_hidden, n_outputs)
     train_network(network, train, l_rate, n_epoch, n_outputs)
     predictions = list()
     for row in test:
         prediction = predict(network, row)
         predictions.append(prediction)

     return(predictions)
```

In [46]:
```
# Test Backprop on Seeds dataset
seed(1)
# load and prepare data
filename = 'data/HR_dataset.csv'
dataset = load_csv(filename)
for i in range(len(dataset[0])-1):
        str_column_to_float(dataset, i)

# convert class column to integers
str_column_to_int(dataset, len(dataset[0])-1)
# normalize input variables
minmax = dataset_minmax(dataset)
normalize_dataset(dataset, minmax)
# evaluate algorithm
n_folds = 5
l_rate = 0.3
n_epoch = 100
n_hidden = 5
scores = evaluate_algorithm(dataset, back_propagation, n_folds, l_rate, n_epoch, n_hidden)
print('Scores: %s' % scores)
print('Mean Accuracy: %.3f%%' % (sum(scores)/float(len(scores))))
```

```
95.9986662221
97.2657552518
96.8989663221
96.198732911
96.432144048
Scores: [95.99866622207402, 97.26575525175059, 96.89896632210737, 96.19873291097032, 96.43214404801601]
Mean Accuracy: 96.559%
```

In [22]:
```
satisfaction_level = float(input("satisfaction_level : "))
last_evaluation = float(input("last_evaluation : "))
number_project = int(input("number_project : "))
average_montly_hours = int(input("average_montly_hours : "))
time_spend_company = int(input("time_spend_company : "))
Work_accident = int(input("Work_accident : "))
promotion_last_5years = int(input("promotion_last_5years : "))

row = [satisfaction_level,last_evaluation,number_project,average_montly_hours,time_spend_company,Work_accident,promotion_last_5y
ears,None]

for i in range(len(row)-1):
    row[i] = (row[i] - minmax[i][0]) / (minmax[i][1] - minmax[i][0])


prediction = predict(network,row)

if(prediction ==1):
    print "Employee will not leave"
else :
    print "Employee will leave"
```

```
satisfaction_level : 0.84
last_evaluation : 0.64
number_project : 2
average_montly_hours : 211
time_spend_company : 3
Work_accident : 0
promotion_last_5years : 0
Employee will not leave
```

References

i) Serrat, Olivier. "Employee performance analysis." Knowledge solutions . Springer Singapore, 2017. 39-43.

ii) Marcus, S., Melanie Moy, and Thayne Coffman. "Employee performance analysis." Mining graph data (2007): 443-467.

iii) McGloin, Jean Marie, and David S. Kirk. "HR Retension analysis." Handbook of quantitative criminology . Springer New York, 2010. 209-224.

iv) https://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/

v) https://medium.freecodecamp.org/building-a-3-layer-neural-network-from-scratch-99239c4af5d3