Neha Reddy Vantari

2001081108

INFO-I535 MGMT ACCESS USE BIG DATA

Course Project: Big Data Concepts and Implementations

On

Human Resource Analytics using Python and MongoDB

Github link for this project: https://github.com/Neha-Reddy-08/INFO-I535.git

1. Introduction

Replacing an employee can be a major expense for companies and may cost them more than $100,000. As a result, most companies aim to retain their employees, but they often struggle to identify the factors that influence employee retention. Machine learning can be utilized to address this issue.

This project's objective is to analyze the employee database of a company using machine learning algorithms to identify which factors impact employee retention. The project will focus on querying the employee database, which is stored in MongoDB, and conducting data exploration to determine the primary features that lead employees to leave the company.

Furthermore, this project could help companies to predict which employees are most likely to leave in the future, allowing them to take appropriate measures to retain them. By doing so, companies can reduce the cost of replacing employees and create a more stable work environment. Additionally, identifying the factors that influence employee retention can assist companies in creating effective retention strategies that cater to their employees' needs and preferences.

2. Background

The problem of employee retention is critical for companies to address because the cost of replacing an employee can be substantial. In addition to the financial costs, losing valuable employees can lead to a decrease in productivity, loss of institutional knowledge, and reduced team morale. Furthermore, employee retention is a complex issue, and companies may struggle to identify the factors that lead to employee turnover. Therefore, leveraging machine learning algorithms to analyze employee data can provide insights that are difficult to obtain through traditional methods. The problem of employee retention is interesting because it affects not only the financial health of a company but also its overall performance. Losing

valuable employees can lead to a decrease in productivity, loss of institutional knowledge, and reduced team morale, which can all have a negative impact on a company's bottom line.

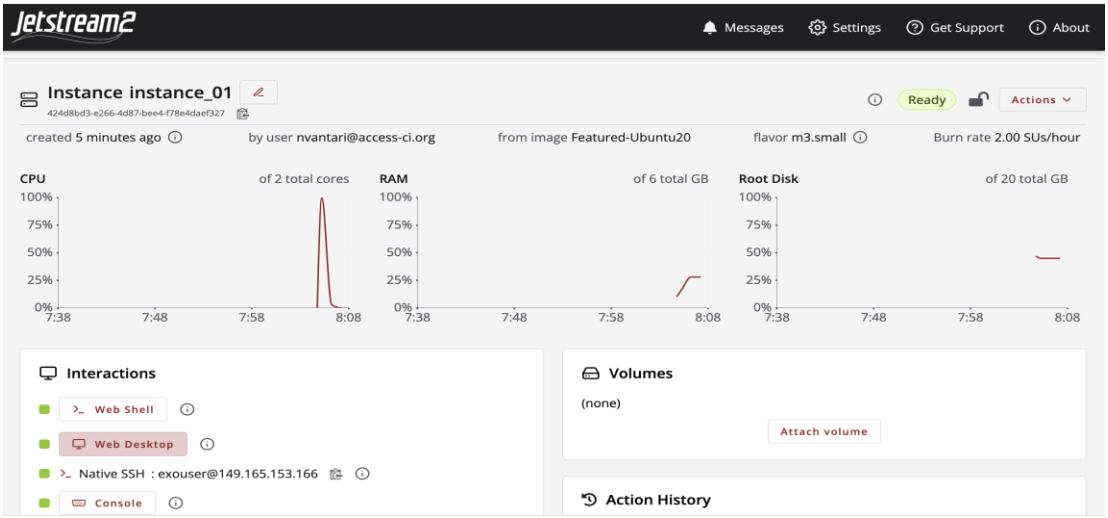
By identifying the key factors that contribute to employee retention, companies can create effective retention strategies that cater to their employees' needs and preferences. Such strategies could include offering competitive compensation and benefits packages, creating a positive work culture, providing opportunities for growth and development, and improving work-life balance.

Moreover, as the job market becomes increasingly competitive, companies need to ensure that they retain their top-performing employees to maintain a competitive edge. Therefore, addressing the issue of employee retention is not only important but also necessary for companies to remain successful in the long run.

3. Methodology

J2 instance:

I started this project by creating an instance in J2. JetStream2 provides a scalable, high-performance messaging system for cloud-native applications that require real-time data streaming and reliable message delivery.



The configuration of the created VM is:

Featured-Ubuntu20

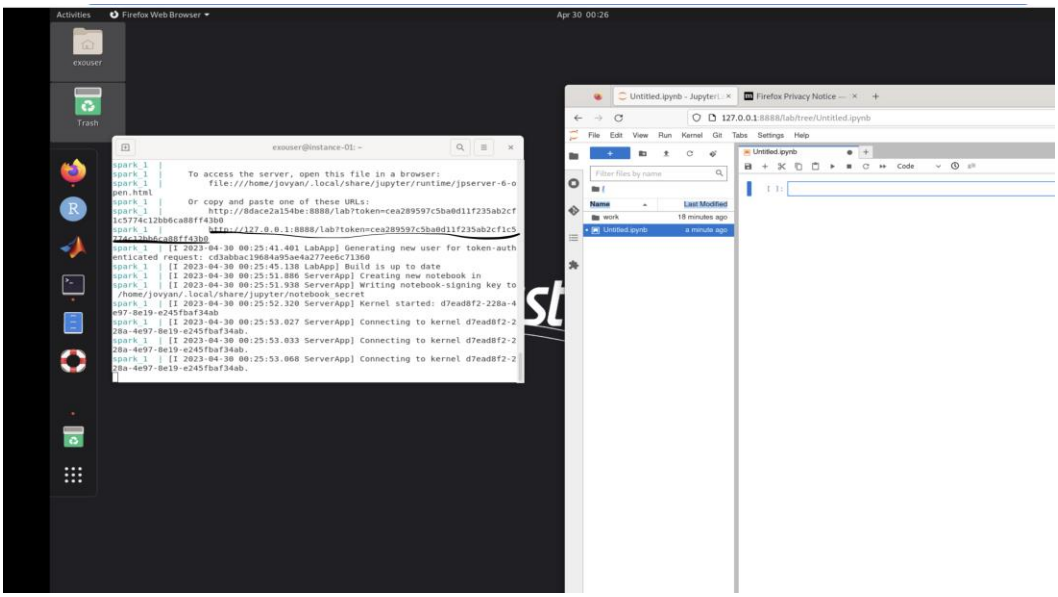
enabled Web Desktop

flavor m3.small: 2 CPU cores, 6 GB RAM, 20 GB Root Disk.

After creating the VM, now I want to open jupyter notebook in this instance and work on it. To do so, first, I checked if the docker is running and created a docker file named '**docker-**

compose.yaml' using nano editor and edited it by adding the contents like version and services as follows and started the container. Although I didn't use pyspark in this project, I have added this extension.

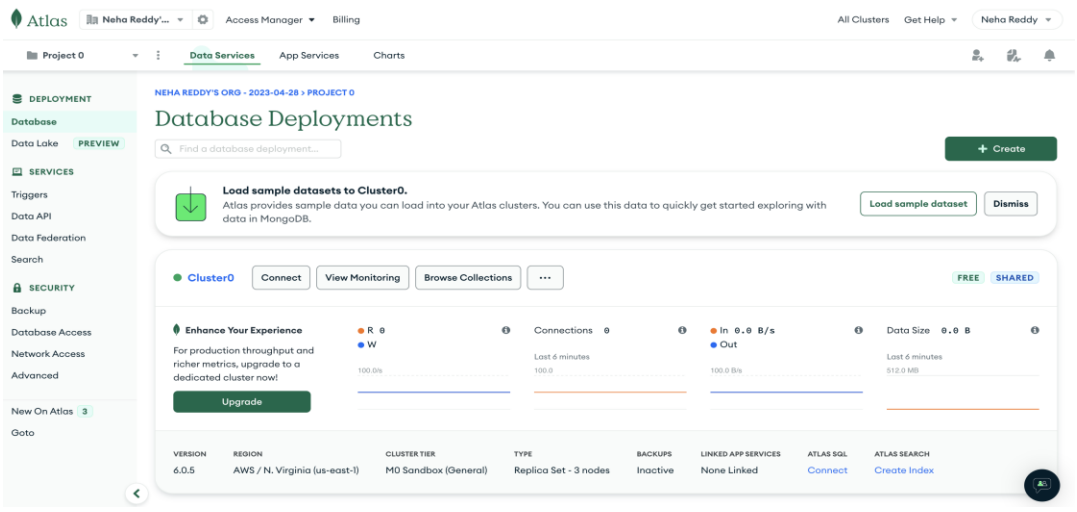
```
version: '3'
services:
 spark:
  image: jupyter /pyspark-notebook
  ports:
   - "8888:8888"
   - "4040-4080:4040-4080"
  volumes:
   - ./notebooks:/home/jovyan/work/notebooks/
```



I have copied the obtained URL that is underlined in the above image and pasted it in the browser which opened jupyter notebook application.

MongoDB Atlas:

I have created an account in MongoDB atlas.



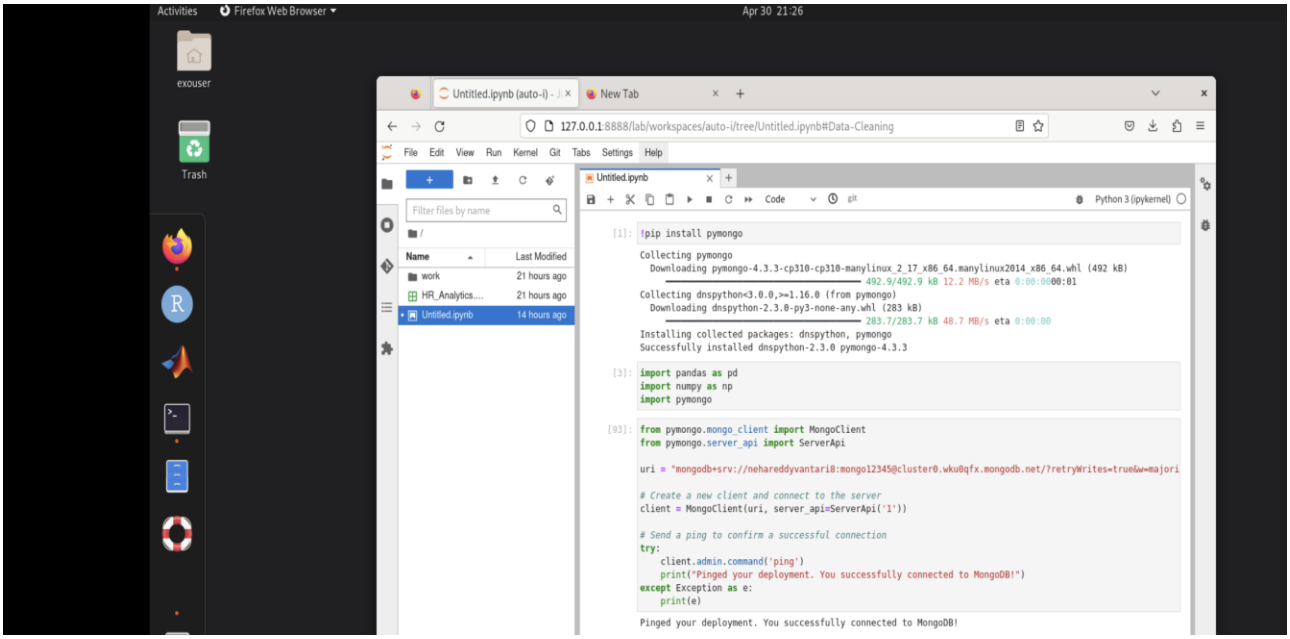
MongoDB is a popular open-source document-oriented NoSQL database system that uses a flexible JSON-like data model. It is designed to store unstructured or semi-structured data and offers a scalable, high-performance, and easy-to-use database solution.

MongoDB Atlas is a fully managed cloud database service that provides developers and organizations with an easy and secure way to deploy, operate, and scale their MongoDB databases on the cloud. It is a cloud-based version of the popular MongoDB database system that eliminates the need for businesses to manage their own database infrastructure.

Once, the account is created, I copied the connection string to connect to the account from J2 instance using python.

Connecting to MongoDB Atlas account from J2 instance:

To connect to MongoDB atlas, we can make use of pymongo library present in python as follows:



Dataset Description:

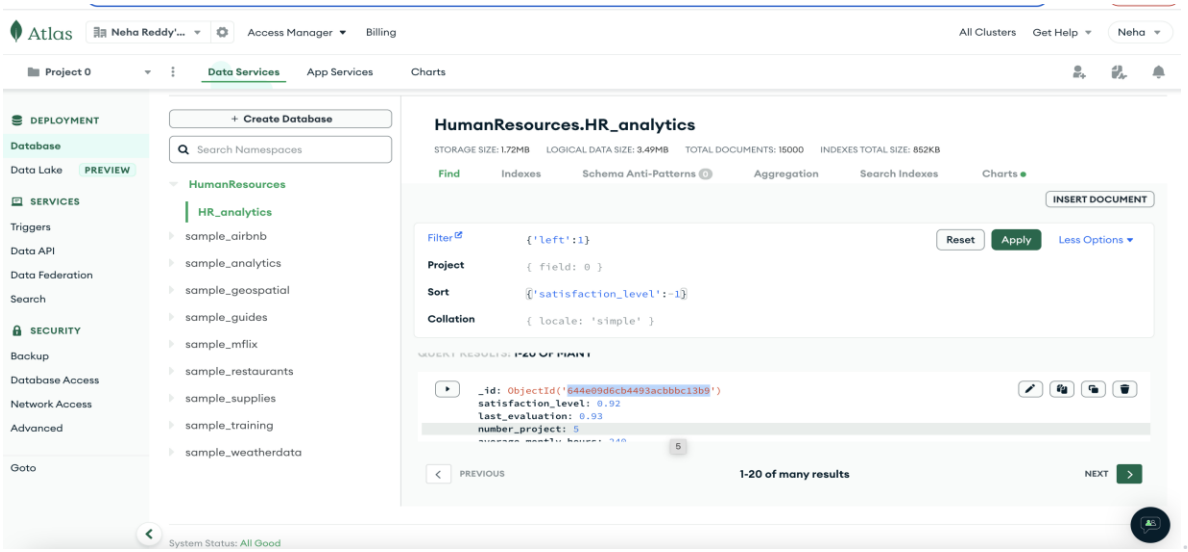
The data set I used for this project is a Human Resources dataset that contains information of various employes of a company. The data was collected and made available in data.world platform. This data set contains 14,999 rows and 10 different columns.

- Sattisfaction_level: lies in the range of 0 to 1. indicates how satisfied an employee is with his current job.
- Number_project: Number of projects the employee has worked on.
- Average_monthly_hours: average number of hours an employee is working in a month.

- Time_spend_company: Number of years the employee has been working in the company.
- Work_accident: number of work accidents if any
- Promotion_last_5years: if an employee has been promoted in the last 5 years. 1 indicates yes and 0 indicates no.
- Departments: the department to which an employee belongs.
- Salary: if the salary of an employee is ‘low’, high’ or ‘medium’
- Left: 1 if the employee is willing to leave, 0 otherwise

Here, ‘left’ is a target variable while all others are feature variables. With the help of the feature variables, we can determine the willingness of an employee to leave the company.

The next step is to import the dataset to MongoDB. I have created a database named HumanResources and collection named HR_analytics in my MongoDB account and imported the dataset.



Why is MongoDB used in this project?

This project chooses to use MongoDB for several reasons. Firstly, MongoDB is easy to use and manage, allowing for efficient storage and querying of data. As shown in the above image, data can be easily queried and analyzed to identify patterns and trends. The query in the image retrieved all the documents of employees who are willing to leave in decreasing order of their satisfaction level. Likewise, we can analyze the data by querying the data set. By analyzing the data in this way, we can decide which metrics are majorly affecting the employees to leave or stay in the company.

In addition to ease of use, MongoDB also offers flexibility and scalability. For example, I want to add new employee data to the dataset in the middle of the project. I can do this using Mongo DB easily by performing insertion operation without any disruption to the project. This is in contrast to traditional SQL databases where manual updates and database restarts are often required. The way of inserting, updating and deleting new documents (employee's data) are explained clearly in the following sections with python code.

Furthermore, MongoDB offers a rich set of APIs and tools, which can be used to manipulate and manage data within the database. This makes it easier to develop and deploy applications that interact with the database.

Overall, MongoDB has a significant impact in this project by making several tasks easy such as:

Storing the data.

Querying and analyzing the data.

Retrieving the required data.

Scaling the database.

Providing flexible schema.

Why J2 instance is used in this project?

Firstly, the J2 instance has better computing power than my personal computer, which allows for faster data processing and analysis. This is particularly important as I am working with a large dataset.

Secondly, the J2 instance is a cloud-based virtual machine, which means that I can access it from anywhere, as long as I have an internet connection.

Thirdly, the J2 instance is pre-configured with many software tools commonly used in data analysis and machine learning, including Python, Jupyter notebook, and MongoDB, which made it easier to set up the environment and helped me to start working on my project right away.

Finally, using J2 instance's Jupyter notebook for connecting to my MongoDB account provides better security for your data, as the data is stored in the cloud

Data Cleaning and Preprocessing:

The next step is to find the dataset from MongoDB collection and convert it to data frame for cleaning and preprocessing.

```
[98]: # Retrieve data from MongoDB
      employees = collection.find({})

[99]: employees=pd.DataFrame(employees)
      employees.head()
```

[99]:

ıber_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years	Departments	salary
2	157	3	0	1	0	sales	low
5	262	6	0	1	0	sales	medium
7	272	4	0	1	0	sales	medium
5	223	5	0	1	0	sales	low
2	159	3	0	1	0	sales	low

```
[100]: employees.info()
```

The data cleaning and preprocessing in this project includes removing null values if any, handling missing values, dealing with the categorical variables, removing outliers and removing duplicates.

Columns like “Departments” and “sales” are categorical. We have to encode them to numerical values. I have performed label encoding using scikit learn library as shown below:

```
[102]: from sklearn.preprocessing import LabelEncoder
       cols_to_encode = ['Departments ', 'salary']
       # Create a label encoder object
       label_encoder = LabelEncoder()

       # Apply the label encoder to the selected columns
       for col in cols_to_encode:
           employees[col] = label_encoder.fit_transform(employees[col])

       # Print the encoded DataFrame
       employees.head()
```

[102]:

mber_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years	Departments	salary
2	157	3	0	1	0	7	1
5	262	6	0	1	0	7	2
7	272	4	0	1	0	7	2
5	223	5	0	1	0	7	1
2	159	3	0	1	0	7	1

To handle missing values, first identify them using isnull() method in pandas and fill the missing values with the average value or median of that column.

To drop the duplicate values, I used drop_duplicates() method present in pandas.

Insertion, Deletion and Updation of documents in MongoDB database from J2 instance:

As discussed in the previous section ‘why MongoDB is used in this project?’, insertion, deletion and updation operations can be performed on the MongoDB providing advantages like scalability and flexibility of database.

The insertion operation is performed as follows:

inserting a new employee data into the dataset

```
[103]: new_employee_1={
           'satisfaction_level':0.88,
           'Work_accident':0,
           'left':0,
           'promotion_last_5years':1,
           'last_evaluation':0.63,
           'number_project':1,
           'average_montly_hours':160,
           'time_spend_company':2,
           'Departments ':"sales",
           'salary':"high"
       }
       new_employee_2={
           'satisfaction_level':0.98,
           'Work_accident':0,
           'left':0,
           'promotion_last_5years':0,
           'last_evaluation':0.73,
           'number_project':3,
           'average_montly_hours':170,
           'time_spend_company':2,
           'Departments ':"sales",
           'salary':"high"
       }
       # Insert data into collection
       collection.insert_one(new_employee_1)
       collection.insert_one(new_employee_2)

[103]: <pymongo.results.InsertOneResult at 0x7fafdf1b2320>
```

To ensure if the insertion operation is successful, we can retrieve these documents:

```
[105]: #Checking the inserted document by reading the document
       result = collection.find_one({'satisfaction_level':0.88,
           'Work_accident':0,
           'left':0,
           'promotion_last_5years':1,
           'last_evaluation':0.63,
           'number_project':1,
           'average_montly_hours':160,
           'time_spend_company':2,
           'Departments ':"sales",
           'salary':"high"})
       print(result)

{'_id': ObjectId('644e0a13cb4493acbbbc4cbd'), 'satisfaction_level': 0.88, 'Work_accident': 0, 'left': 0,
'promotion_last_5years': 1, 'last_evaluation': 0.63, 'number_project': 1, 'average_montly_hours': 160, 't
ime_spend_company': 2, 'Departments ': 'sales', 'salary': 'high'}
```

This document now has an object id which is created by MongoDB indicates that the document has been inserted into the dataset

We can also update the value of a particular attribute in a document and also delete a document as shown below:

Updating a document in the database

```
[106]: result = collection.update_one(
           {'satisfaction_level':0.88},
           {'$set': {'satisfaction_level': 0.92}}
       )
       print(result.modified_count)

1
```

Deleting a document from the database

```
[107]: result = collection.delete_one({'satisfaction_level': 0.92})
       print(result.deleted_count)

1
```

Data Exploration:

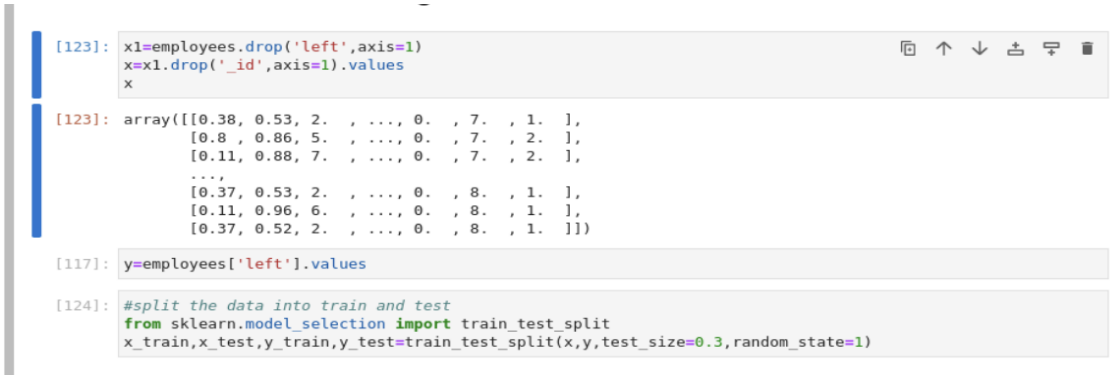
Data exploration is the process of examining and analyzing a dataset to gain a better understanding of its structure, patterns, and relationships between variables. It involves summarizing and visualizing the data, identifying any trends or outliers, and uncovering any potential issues or biases that may affect the quality of the data.

I have used tools like seaborn, matplot and techniques like histogram, barplot, scatterplot, distplot and correlation matrix to gain insights into the data. The plots are explained in the next section.

Predictive Modelling:

Predictive modeling is the process of using statistical algorithms and machine learning techniques to analyze data and make predictions about future events or outcomes. Here, in this project, I am trying to predict if an employee is willing to leave the company or happy with his job using Machine Learning algorithms like Logistic Regression, Decision Trees and Random Forest.

First, I have separated the feature variables and target variable('left') and then divided the data into train and test sets using train_test_split method in scikit learn library as follows:



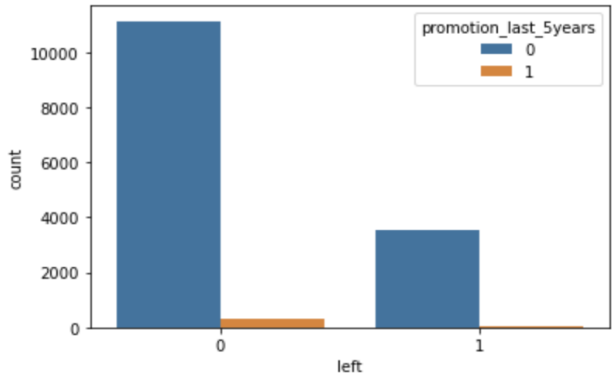
```
[123]: x1=employees.drop('left',axis=1)
       x=x1.drop('_id',axis=1).values
       x

[123]: array([[0.38, 0.53, 2.  , ..., 0.  , 7.  , 1.  ],
              [0.8 , 0.86, 5.  , ..., 0.  , 7.  , 2.  ],
              [0.11, 0.88, 7.  , ..., 0.  , 7.  , 2.  ],
              ...,
              [0.37, 0.53, 2.  , ..., 0.  , 8.  , 1.  ],
              [0.11, 0.96, 6.  , ..., 0.  , 8.  , 1.  ],
              [0.37, 0.52, 2.  , ..., 0.  , 8.  , 1.  ]])

[117]: y=employees['left'].values

[124]: #split the data into train and test
       from sklearn.model_selection import train_test_split
       x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=1)
```

Then, I fit each of the model to the training data and predicted the output using testing data. I have calculated accuracy and confusion matrix for each algorithm. The results are discussed in the next section.

4. Results

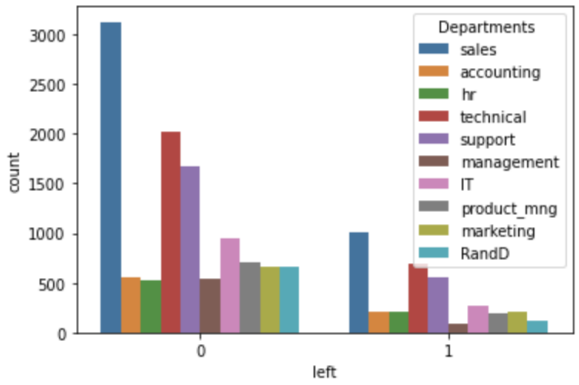
Data Exploration Results:

```
In [12]: sns.countplot(x='left', hue = 'promotion_last_5years', data=dataset)
Out[12]: <AxesSubplot:xlabel='left', ylabel='count'>
```



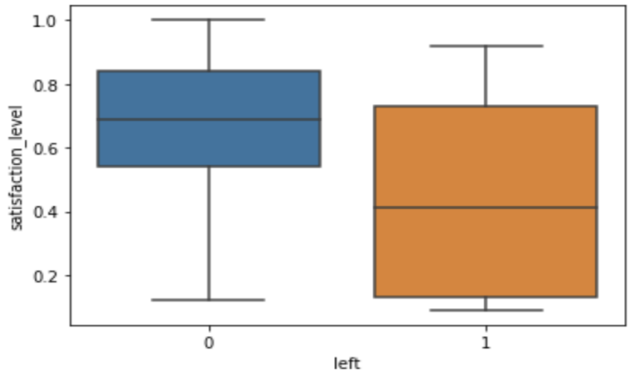
From the above graph, we can say that almost all the employees who were promoted in the last 5 years are willing to stay in the company while those who weren't promoted are willing to leave.

```
In [9]: sns.countplot(x='left',hue='Departments ',data=dataset)
Out[9]: <AxesSubplot:xlabel='left', ylabel='count'>
```



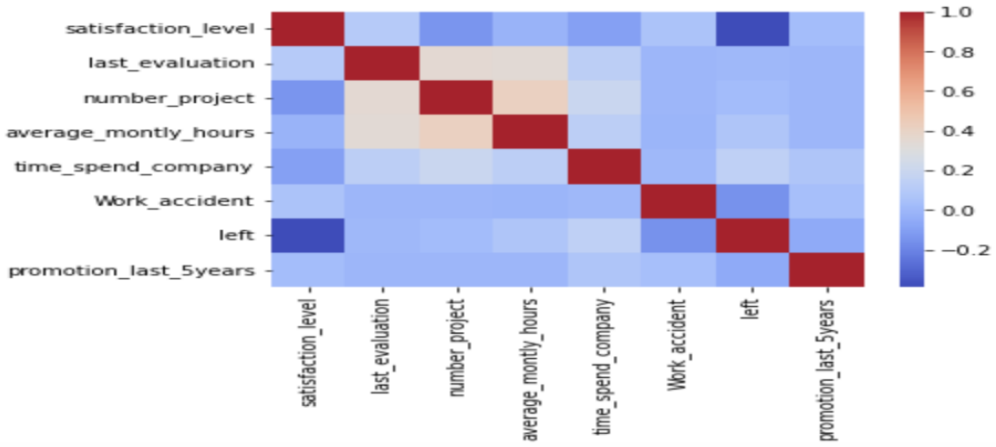
This graph indicates that most of the employees who are willing to leave are from sales and technical department which means that the company should develop stategies specifically to retain employees of these departments.

```
In [18]: sns.boxplot(x='left',y='satisfaction_level',data=dataset)
Out[18]: <AxesSubplot:xlabel='left', ylabel='satisfaction_level'>
```

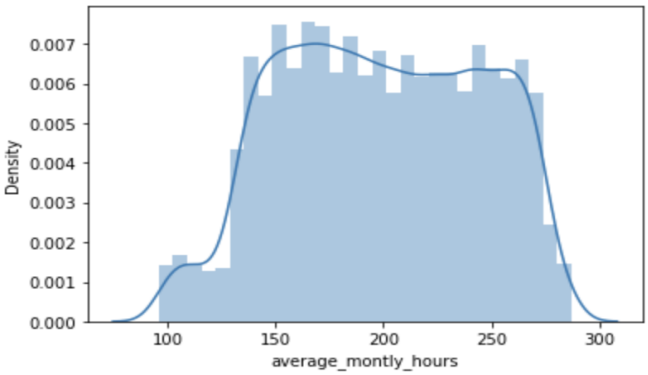


The above plot depicts how satisfaction level affect employees. Satisfaction level is inversely proportional to employee willingness to leave the company.

```
# Create a heatmap using seaborn
sns.heatmap(corr_matrix, cmap='coolwarm')

# Show the plot
plt.show()
```



```
In [21]: sns.distplot(notleft['average_montly_hours'])
Out[21]: <AxesSubplot:xlabel='average_montly_hours', ylabel='Density'>
```



This correlation plot and distplot suggests that our target variable is least correlated with satisfaction level and highly correlated with average monthly hours which means, if an employee is working for more hours, his satisfaction level is low and is willing to quit. This insight can help a company to retain its employees.

Predictive Modelling Results:

(a) Logistic Regression

Logistic regression

```
[125]: from sklearn.linear_model import LogisticRegression
       reg=LogisticRegression()
       reg.fit(x_train,y_train)

/opt/conda/lib/python3.10/site-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs
failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(

[125]: ▾ LogisticRegression
       LogisticRegression()

[126]: model = LogisticRegression(solver='liblinear', random_state=0).fit(x, y)
       model.predict(x)

[126]: array([0, 0, 1, ..., 0, 1, 0])

[127]: y_pred=reg.predict(x_test)

[128]: #accuracy of algorithm and confusion matrix
       from sklearn.metrics import confusion_matrix,accuracy_score
       cm=confusion_matrix(y_test,y_pred)

[129]: cm

[129]: array([[3128,  288],
              [ 777,  307]])

[130]: accuracy_score(y_test,y_pred)

[130]: 0.7633333333333333
```

(b) Decision Tree

Decision Tree

```
[138]: from sklearn.tree import DecisionTreeClassifier,plot_tree
       dc=DecisionTreeClassifier(random_state=0)
       dc.fit(x_train,y_train)

[138]: DecisionTreeClassifier
       DecisionTreeClassifier(random_state=0)

[139]: y_pred2 = dc.predict(x_test)
       y_pred2

[139]: array([0, 0, 0, ..., 0, 0, 1])

[140]: from sklearn.metrics import confusion_matrix, accuracy_score
       cm_dt=confusion_matrix(y_test, y_pred2)
       print("Confusion matrix:\n", cm_dt)
       print("Accuracy score: ", accuracy_score(y_test, y_pred2))

       Confusion matrix:
        [[3339   77]
        [  42 1042]]
       Accuracy score:  0.9735555555555555
```

(c) Random Forest

Random Forest

```
[141]: from sklearn.ensemble import RandomForestClassifier
       classifier = RandomForestClassifier(n_estimators = 20, random_state = 0)
       classifier.fit(x_train, y_train)

[141]: RandomForestClassifier
       RandomForestClassifier(n_estimators=20, random_state=0)

[142]: y_pred = classifier.predict(x_test)

[143]: y_pred

[143]: array([0, 0, 0, ..., 0, 0, 1])

[144]: cmr=confusion_matrix(y_test,y_pred)
       accuracy_score(y_test,y_pred)
       print("confusionmatrix:\n",cmr)
       print("Accuracy score: ", accuracy_score(y_test, y_pred))

       confusionmatrix:
        [[3408    8]
        [  42 1042]]
       Accuracy score:  0.9888888888888889
```

Among these three algorithms, Random Forest performed better with an accuracy of 98%. The use of such models in real-time applications can greatly benefit companies by enabling them to automatically determine the likelihood of an employee staying in the company. With this automated analysis, companies can avoid the need to conduct exhaustive analyses on all employees regularly, as these models can handle the task efficiently.

By utilizing these models, companies can easily determine whether an employee is likely to leave their job based on the given data. This information can be used by the company to develop and implement appropriate strategies to retain valuable employees. Thus, the predictive power of these models can greatly benefit companies in their efforts to maintain a productive and stable workforce.

5. Discussion

Human Resource Analytics project focuses on employee retention which is important for companies to manage financial costs, productivity and loss of institutional knowledge. The obtained results suggest that employee satisfaction level is major criteria and to satisfy employees, the average work hours should be reduced. In addition, the management should acknowledge employees by promoting them or giving them a hike. Moreover, the results also suggest that by using Random Forest model, the companies can ease the task of analyzing the likelihood of employees.

Due to the large size of the dataset being used in this project, as well as the typically massive employee databases encountered in real-world scenarios, it is crucial to employ a virtual machine that can support the necessary RAM and CPU requirements. The J2 instance has been a particularly valuable asset in this regard, as it has proven capable of running the necessary code quickly and efficiently. This has helped to minimize the time required for data analysis and model development, ultimately facilitating the development of more effective solutions for employee retention and management.

During the course of this project, it has become apparent that employee databases are subject to frequent change in real-world scenarios. This raises the question of how to update the database without requiring a restart each time. Fortunately, MongoDB has emerged as a valuable solution in this context, with its ability to support insertion and deletion operations, as well as its inherent flexibility as a database platform. With MongoDB, it is possible to update the database quickly and easily, without requiring extensive manual updates or downtime. This can help to ensure that employee data remains accurate and up-to-date, ultimately facilitating more effective decision-making and analysis.

6. Conclusion

This project aimed to explore the employee database of a company by utilizing MongoDB and J2 VM and identify the willingness of an employee to stay within the company using machine learning algorithms. The dataset used in the project was analyzed using various techniques, including data cleaning, exploration, modeling, and prediction. Three machine learning algorithms, Logistic Regression, Decision Tree, and Random Forest were trained on the dataset to predict the likelihood of an employee leaving the company. Among these, Random Forest performed better with an accuracy of 98%

Although the project achieved good results, it is important to note that these results are only applicable to the specific dataset used in this project. It is possible that different datasets may produce different accuracy values, as machine learning algorithms are generally more effective when trained with diverse datasets. Furthermore, the dataset used in this project has some degree of skewness, which could limit the generalizability of the results to real-world scenarios. In order to obtain more accurate and reliable results, it is recommended to train the model on a more diverse dataset that is representative of the real-world population.

7. References:

- https://core.ac.uk/download/pdf/229656351.pdf
- https://nafeea3000.medium.com/human-resource-analytics-using-machine-learning-6a32392f6ec1
- https://medium.com/analytics-vidhya/import-csv-file-into-mongodb-9b9b86582f34
- https://www.mongodb.com/community/forums/t/import-csv-file-data-into-mongodb-using-python/15759
- https://docs.jetstream-cloud.org/general/vmsizes/
- https://www.researchgate.net/publication/362834732_Identification_of_human_resource_analytics_using_machine_learning_algorithms