**DATA SCIENCE PROJECT**

**Using Python Machine Learning Techniques and Power BI to Build a Semi-Supervised Model for Classifying and Predicting Employees Performance**

# **Business Problem:**

This notebook presents my analysis of an Employee Performance Analytics dataset that contains features on 930 employees on whether they will be a good fit on a project. The goal is to build semi-supervised model that uses these features to cluster and predict whether selected employee is best to be included in a project.

# **Business Objective:**

The objective of this project is to analyse and identify a combination of factors/variables which show the best fitted employee for the project. Clustering the data and identifying the best fit employee on the project will be the main objective of this analysis.

# **Project Requirements:**

Python 3.5 or above version

Python Machine Learning Library: Pandas, Numpy, Matplotlib, Seaborn, Scikit-Learn

Power BI version 2.88.621.0 or above version

# **Data Preparation:**

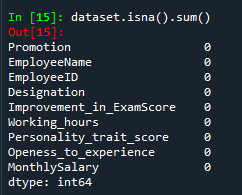
Now that we're set up, we must prep the data.

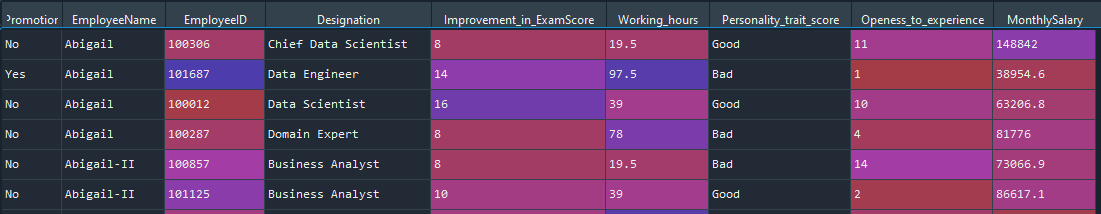
Let's take a quick look at the data.

**Input:**



**Output:**



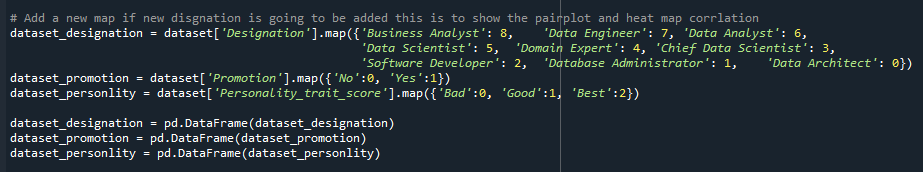


We then check for null values and take a closer look at the columns in our data set. We have three categorical independent varialble, Promotion, Designation and Personality\_trait\_score. The rest are numeric. We can also notice that there are no missing values on our dataset, therefore there is no need to do imputations for the data preparations.

# **Data Variable Preparation**

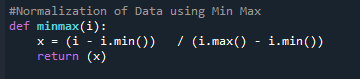
We then transform the categorical columns of data to its corresponding nominal values using label encoding technique of pandas library. We also removed the independent variable employee name, and employee ID as we will not be using it for data analysis.

**Input:**



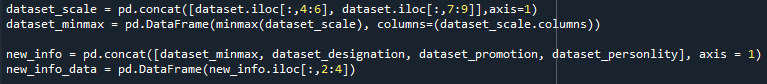
Defining a function for min max scaling.

**Input:**

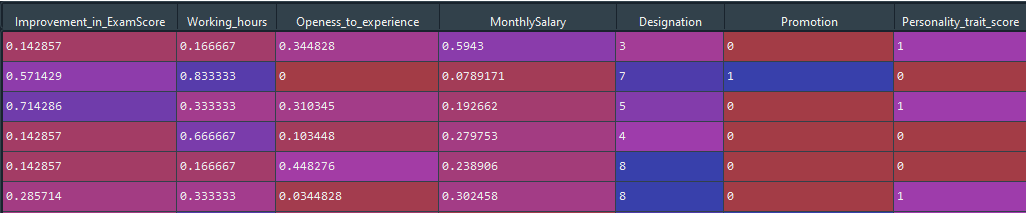


Applying normalization in the dataset that will be used for data analytics.

**Input:**



**Output:**



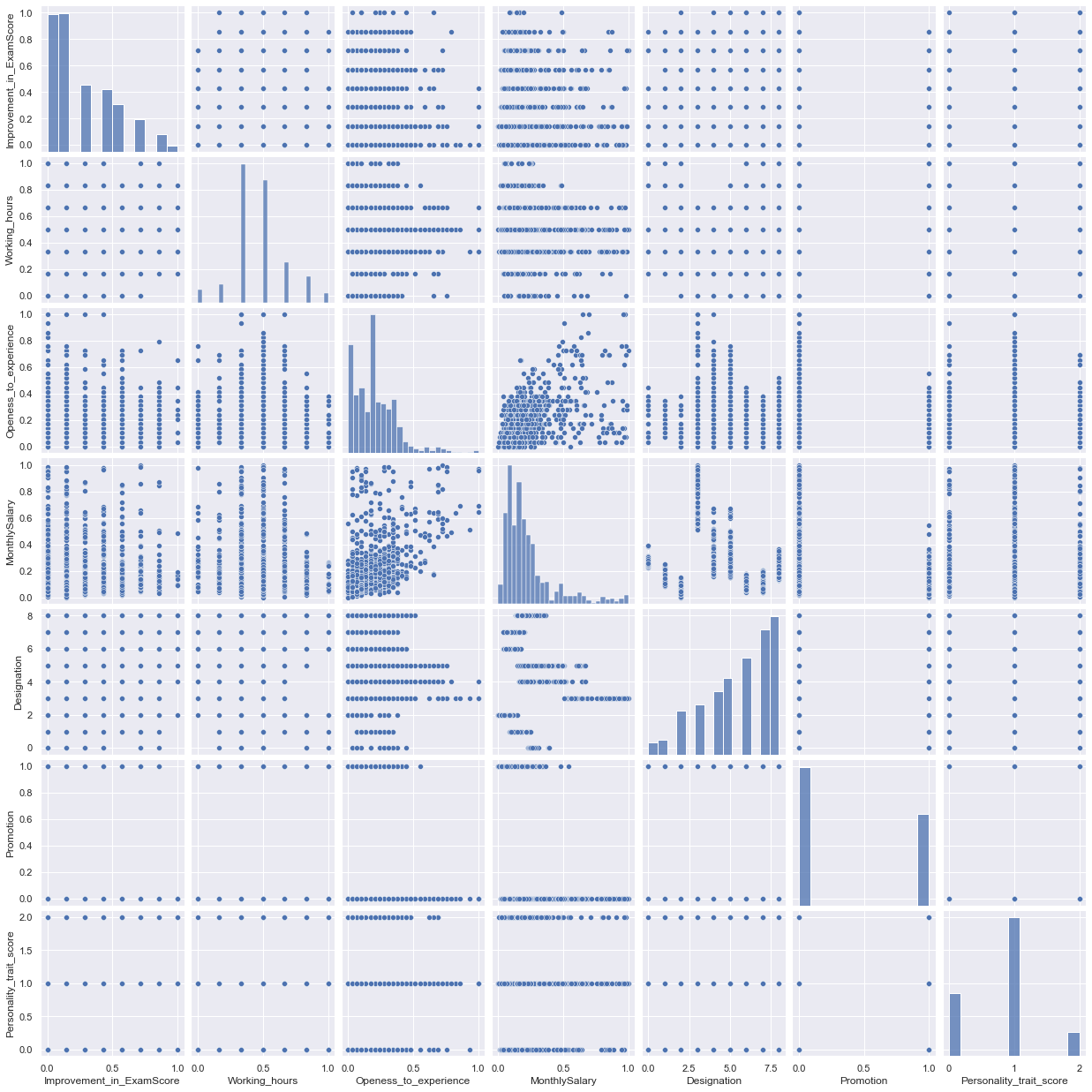
# **Exploratory Data Analysis**

This is the Most Important Chart in the perspective of the project manager, as It is very important to have some intuition and idea about the relation of the variables in selecting the best fitted employee for the project

**Input:**



**Output:**

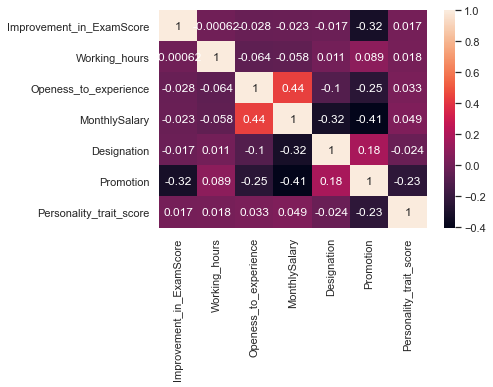


We also check the importance of the correlation of each variable to one another. As we can see in the image that only *Openess\_to\_experience* and *MonthlySalary* have a high correlation. Therefore, we can base our cluster based on this two-independent variable.

**Input:**



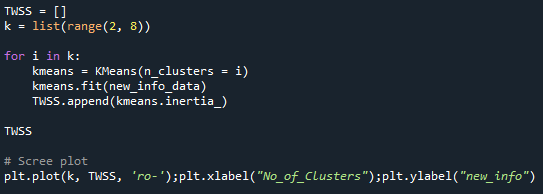
**Output:**



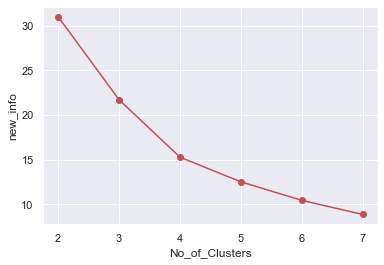
# **Data Mining Preparation for Unsupervised Learning Model**

Executing the function for scree plot to identify what is the best number of clusters we can use in the data. As we can observed in the data below that the best number of clusters we can use is 3 – as there is where the sudden change in our data happened. Therefore, we can use 3 as number of cluster in or model.

**Input:**

****

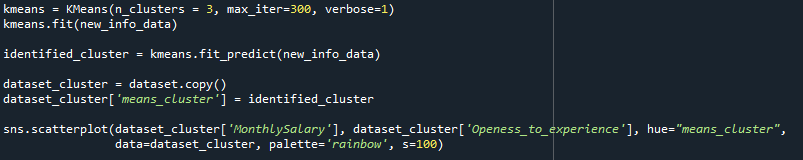
**Output:**



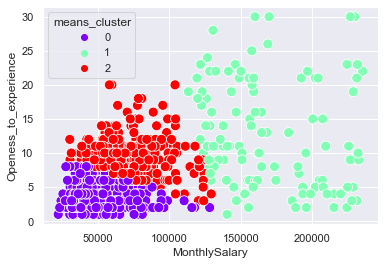
# **Application of K-Means Clustering on the data**

Using K-Means to cluster the data with 3 clusters and maximum iteration of 300 we have arrived to our scatter plot as seen on the image below. The data is been clustered to those with *low Openness\_to\_experience, and MonthlySalary, medium Openness\_to\_experience and MonthlySalary, and high Openness\_to\_experience, and MonthlySalary.*

**Input:**

****

**Output:**



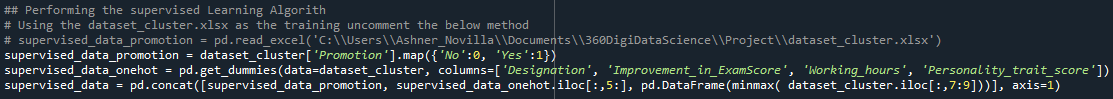
# **Application of Semi-supervised Learning to the Clustered Data**

After performing the unsupervised model, we will move to supervised model – our objective is to make a semi – supervised model. From the *“dataset\_cluster”* from the previous model we will start creating a supervised model

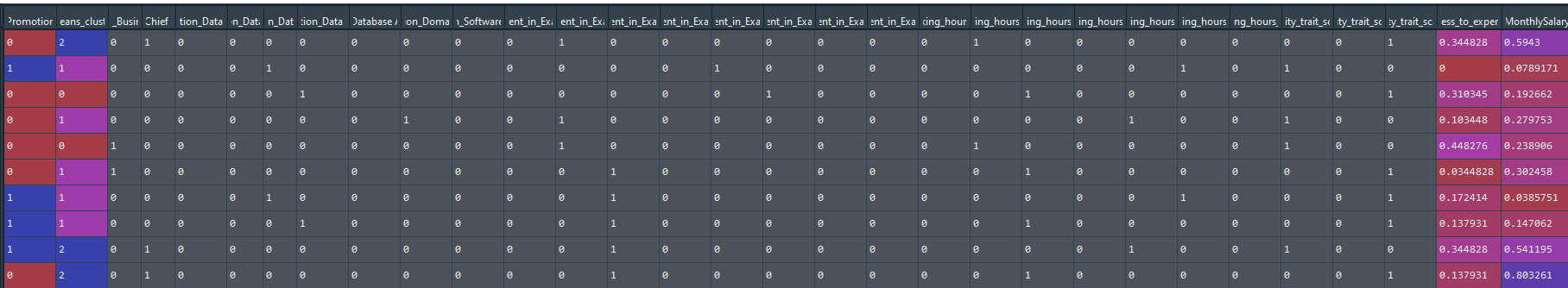
# **Data Preparation for Supervised Model**

As we can see from our pair plot during the data clustering – its been observed that *‘Designation’, Improvement\_in\_ExamScore’, ‘Working\_hours’, ‘Personality\_trait\_scrore’, and ‘Promotion’* are actually discrete data. Therefore, using one-hot encoding we can get a better accuracy of our target cluster.

**Input:**

****

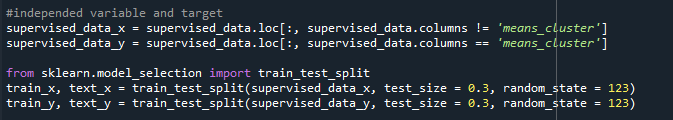
**Output:**

****

# **Data Mining for Supervised Model**

We then proceed to splitting the data to train and test using sklearn.preprocessing. the test\_size = 30% our whole data while train\_data = 70% with a random\_state = 123.

**Input:**



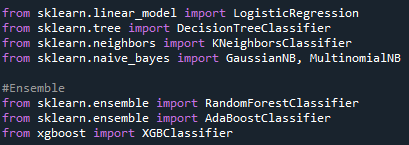
# **Data Modeling for Supervised Model**

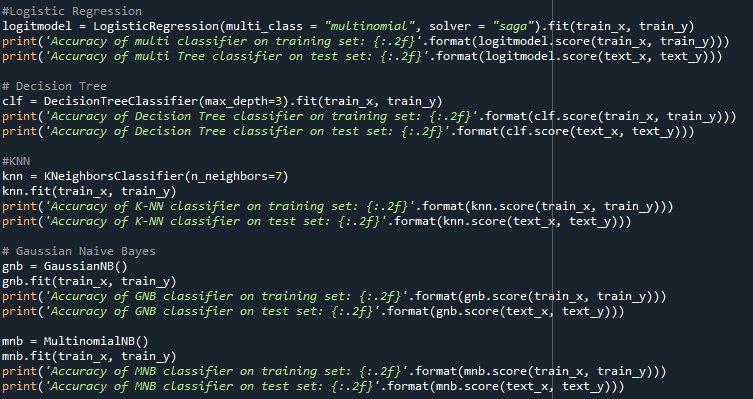
We now must decide on the modelling algorithms that we want to apply. The goal of our model is to use a set of employee characteristics to label him/her on what cluster he/she belongs that will help to identify if he/she is a good fit for the project. Essentially, we are looking for supervised learning algorithms that perform "classification". The following are a few algorithms that meet these criteria:

* Logistic Regression
* Decision Tree Classifier
* KNeighbors Classifier
* Naive Bayes Classifier using GaussianNB and MultinomialNB
* RandomForestClassifier
* AdaBoostClassifier
* XGBClassifier

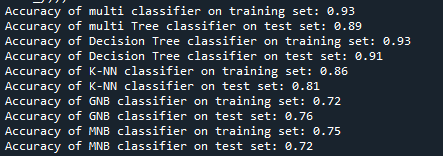
I ultimately selected these techniques because they are relatively simple to understand.

**Input:**

****

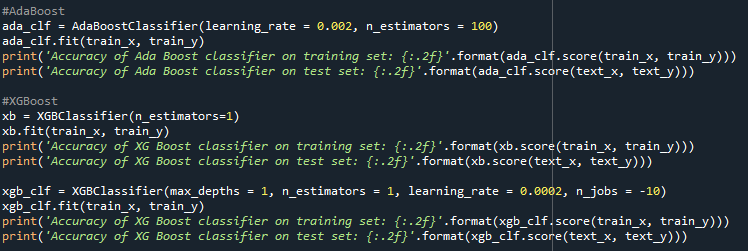
****

**Output:**

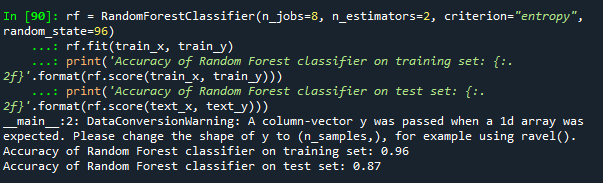
****

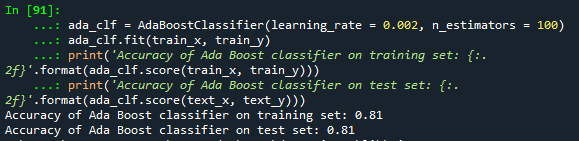
**Input:**

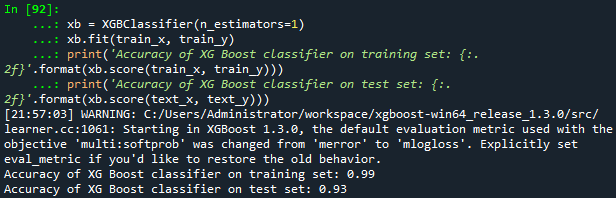
Using Ensemble Techniques to check if it will show a better accuracy



**Output:**

****

****

****

# **Model Evaluation**

As we observed in our data modelling, only XGBoost Classifier with train accuracy of 99% and test accuracy of 93% and Decision Tree Classifier with a train accuracy of 93% and test accuracy of 91% - gave a high accuracy. If we look closely, we can see that XGBoost Classifier is overfitting which will be a problem in resolving future analysis. Therefore, it’s been decided that we will be using Decision Tree Classifier.

|  |  |  |
| --- | --- | --- |
| **Model** | **Train Score** | **Test Score** |
| LogisticRegression | 0.93 | 0.89 |
| DecisionTreeClassifier | 0.93 | 0.91 |
| KNeighborsClassifier | 0.86 | 0.81 |
| NaïveBayes GaussianNB | 0.72 | 0.76 |
| NaïveBayes MultinomialNB | 0.75 | 0.72 |
| RandomForestClassifier | 0.96 | 0.87 |
| AdaBoostClassifier | 0.81 | 0.81 |
| XGBClassifier | 0.99 | 0.93 |

# **Data Visualization using Power BI and Python**

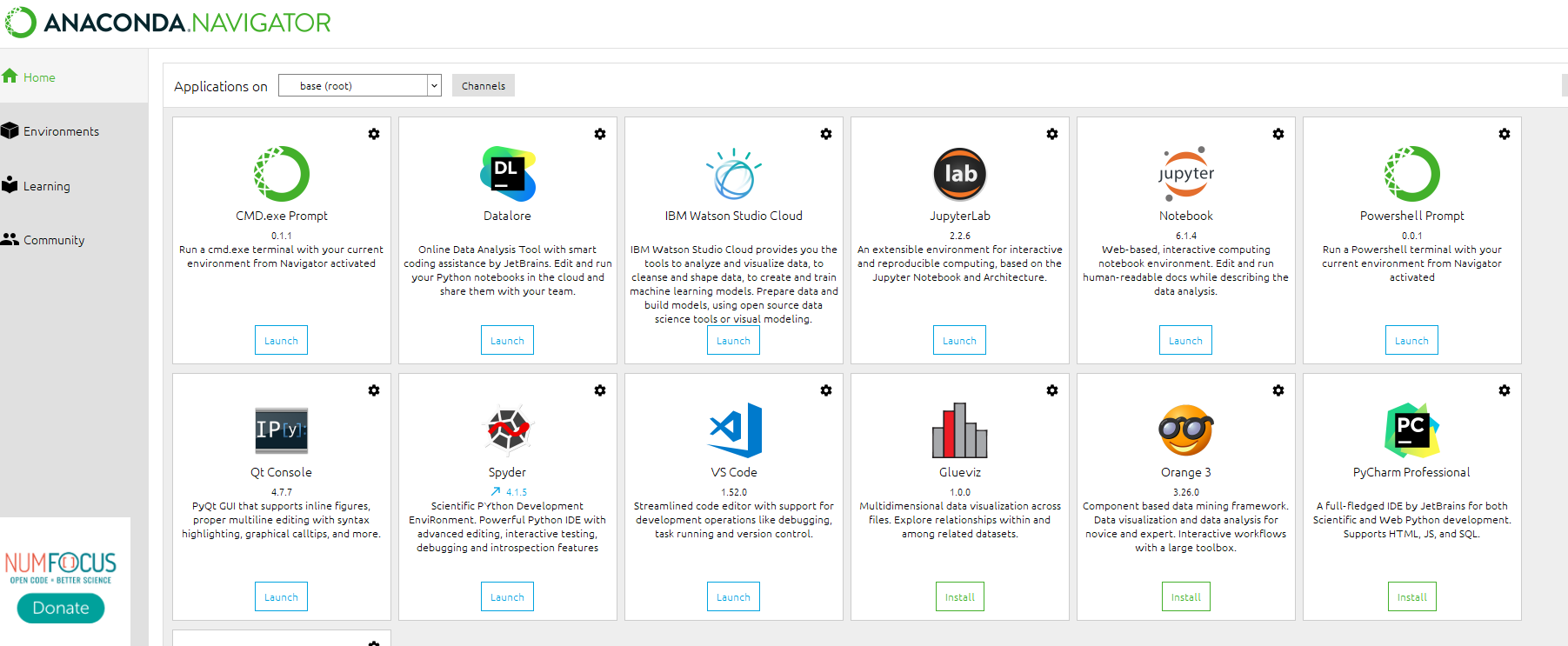
We need to install Power BI and install a python to our PC. In the student side we installed anaconda in our system to enable a fully established environment for data analysis.

Install the system as seen on the image below:

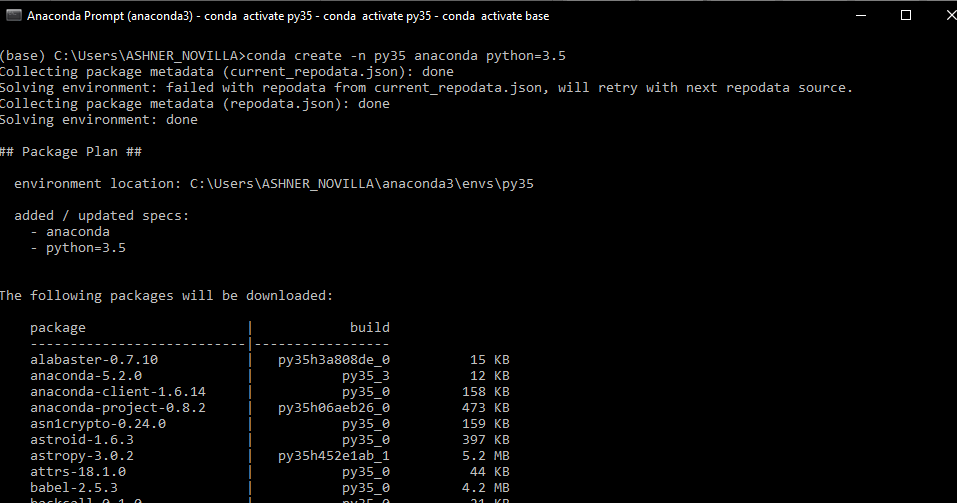


Install Anaconda Navigator as seen on the image below:

Link for the installation: [Installing on Windows — Anaconda documentation](https://docs.anaconda.com/anaconda/install/windows/)



Launch the Anaconda Prompt and start making an environment that will be used by Power BI as seen in the image below:



Install the needed library for Data Analytics

**pip install pandas**

**pip install matplotlib**

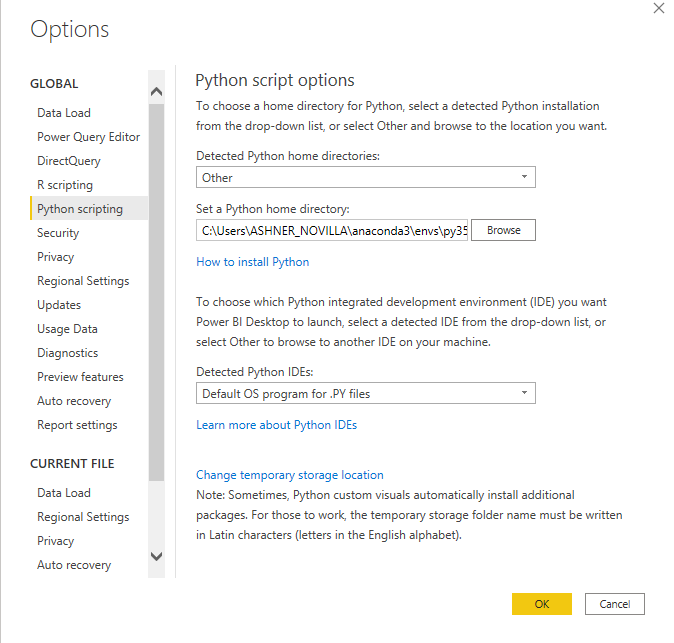
**pip install sklearn**

**pip install numpy**

**pip install xgboost**

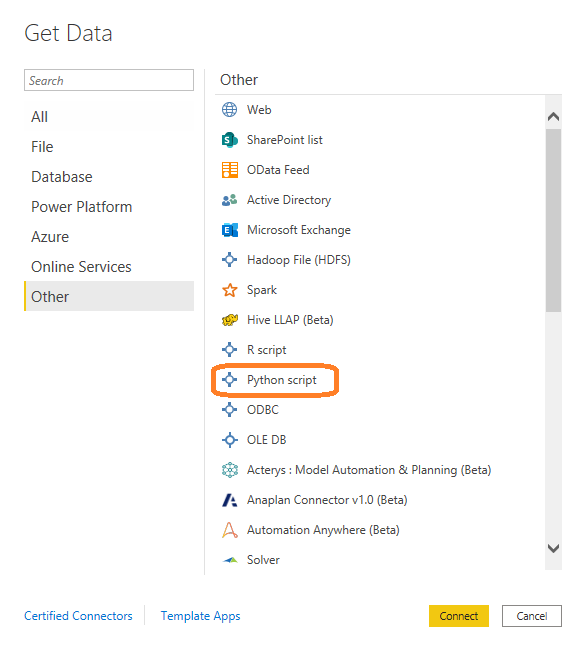
**pip install seaborn**

Launch the Power BI and go to File > options and settings > options > python scripting. Set the Python home directory to the environment that been created.

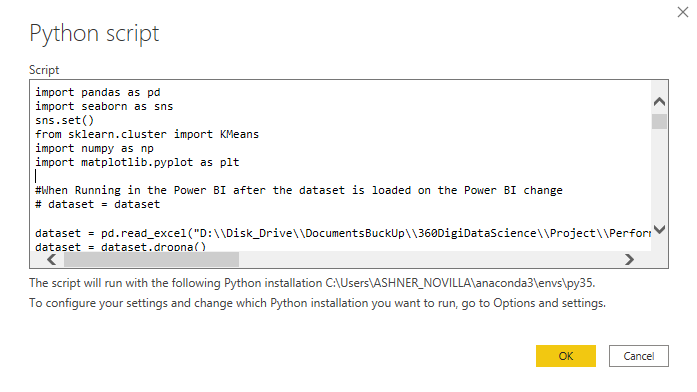


# **Importing the Data Model to Power BI using Python**

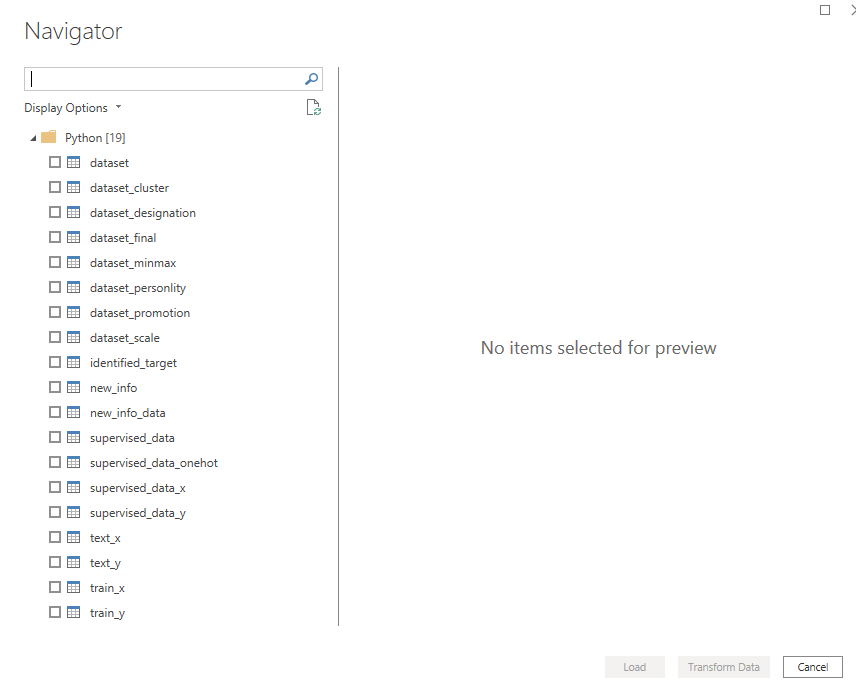
From Power BI click File > Get Data > Get Data to get Started > Other > Python Script. Then we can copy and then paste the code we have designed**. Load the dataset\_final** and perform data visualization.



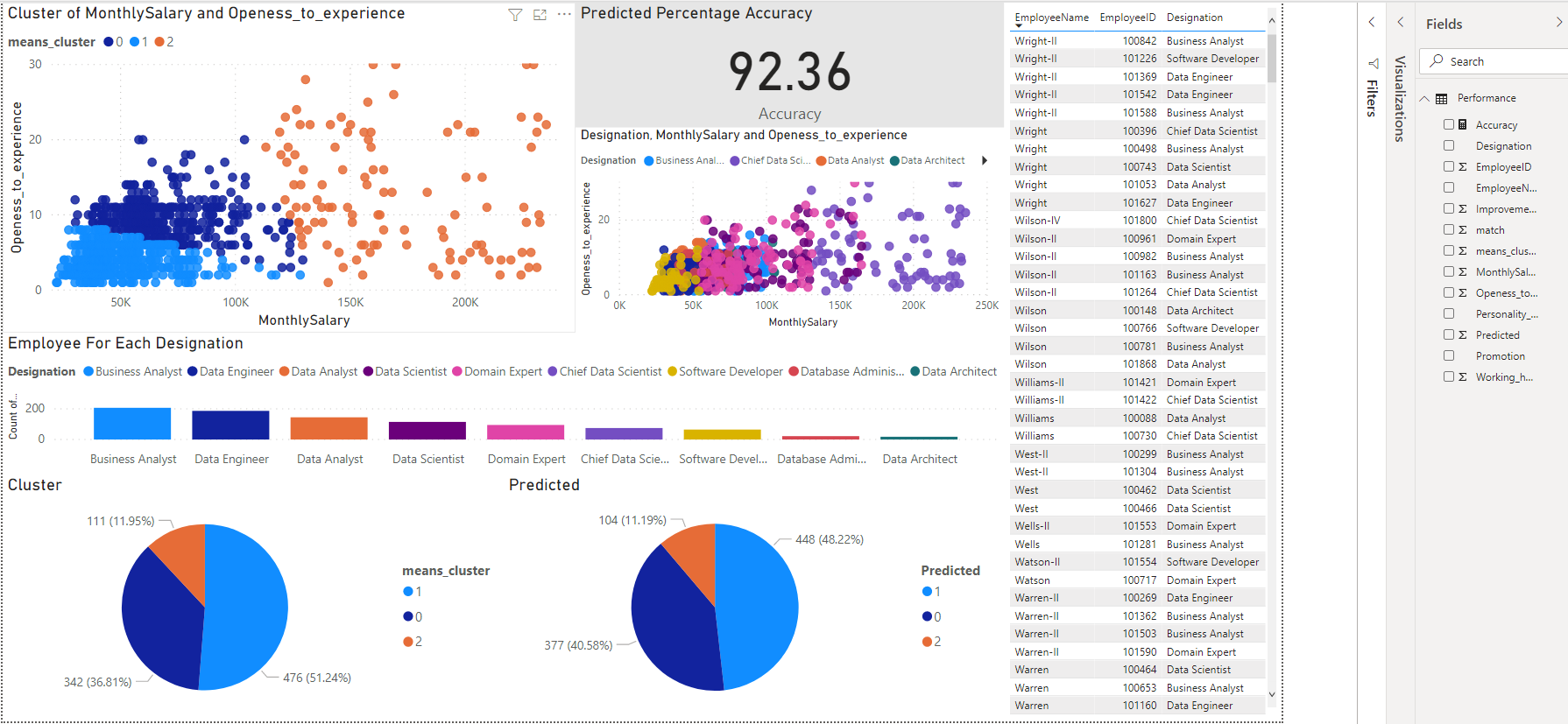
**Input:**

****

**Output:**

****

# **Final Output for Data Analysis using Python for Data Analysis and Power BI for Data Visualization**

****