Final Project

1. **Introduction and Initial Exploratory Data Analysis**

Nowadays, fraud detection is one key aspect of many businesses and Machine Learning algorithmns are helping Data Scientist and Engineers tackle this problem. Banks, credit card providers, online shops and many other types of services have to face daily problems with attempts of fraud throughout the world. One of the most famous cases of corporate fraud in our recent history is the Enron scandal that happened in 2001. Enron Corporation was an American energy company with a market value of billions of dollars. The fraud occurred specially on the financial division of corporation, with key directors of the company falsifying numbers to investors and general public. Some directors and high position employees were involved in the fraud and were late charged with crimes. The goal of the current project is to access data generated out of employees that were part of the fraud inside Enron (labeled Person of Interest, a.k.a POI) and regular employees that were not prosecuted criminally. The dataset contains different types of numeric data like salary, number of email exchanged with POI, amount of stocks, etc and also some text data contained on the emails exchanged by employees. We will focus on the numeric data to do our exploratory data and model prediction, with the goal to predict the tested subjects as being a POI or not.

The data that were provided for the project consisted of a pickle file that could be read on a Python dictionary, with keys represented by the name of the employee, and values being a dicionary containing the features and respective values. In order to inspect the data in a more detailed way, the dictionary was converted into a Pandas DataFrame, with the names of each subject as index, and each feature as columns. By inspecting the Dataframe created with .info() it is noticed that it has 146 rows and 21 columns. One problem that is initially visualized is that the data type for most of the columns are non-null object type (the equivalent for string in Pandas), and that the missing values are coded as ‘NaN’ strings. The only exception to it is the ‘poi’ column which is a boolean type column. By converting the columns to numeric type, we can have a better notion of the missing values for each feature column.

<class 'pandas.core.frame.DataFrame'>

Index: 146 entries, ALLEN PHILLIP K to YEAP SOON

Data columns (total 21 columns):

salary 95 non-null float64

to\_messages 86 non-null float64

deferral\_payments 39 non-null float64

total\_payments 125 non-null float64

exercised\_stock\_options 102 non-null float64

bonus 82 non-null float64

restricted\_stock 110 non-null float64

shared\_receipt\_with\_poi 86 non-null float64

restricted\_stock\_deferred 18 non-null float64

total\_stock\_value 126 non-null float64

expenses 95 non-null float64

loan\_advances 4 non-null float64

from\_messages 86 non-null float64

other 93 non-null float64

from\_this\_person\_to\_poi 86 non-null float64

poi 146 non-null bool

director\_fees 17 non-null float64

deferred\_income 49 non-null float64

long\_term\_incentive 66 non-null float64

email\_address 0 non-null float64

from\_poi\_to\_this\_person 86 non-null float64

dtypes: bool(1), float64(20)

memory usage: 24.1+ KB

A traditional way to spot outliers is by using boxplots or scatter plots. But as our dataset has so many features, it becomes a daunting and complex task. We could use PCA to reduce the the number of dimensions of our data, and find samples that diverge 3 or more standard deviations from the mean, by using the z-score. But z-score is more suitable for data that follow a normal distribution, not our case. So in order to identify outlier we used a clustering approach with scikit-learn DBSCAN algorithm

As our database had many features, spotting outliers is not straight foward. One way to do it is by using PCA to reduce the number of dimensions and try visualizing it.

PCA plot

Also we can measure the z-score to find values that show 3 standard deviations more or less in relation to the mean. By doing this we spotted a row named ‘TOTAL’, which is probably an aggregation of all the other rows of the Dataframe, so it was removed.

2)