Approach to solve the problem

1. Brief on the approach

We have loaded the data with the help of panda Library we have cleaned the data by using some techniques, later we applied feature engineering techniques so that we can know which column contribute to the target feature .once the pre-processing part is done we moved on to creating our model be applied various algorithm then the data set and test data set Until the model was trained and tested to its ideal accuracy .

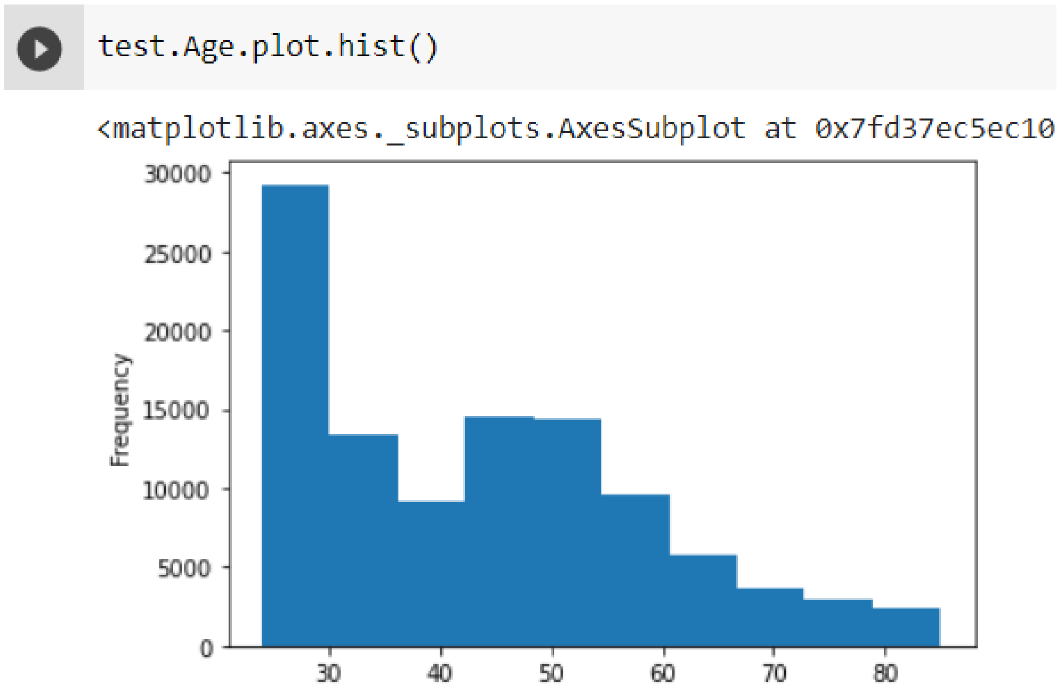
2.What data-preprocessing / feature engineering ideas really worked? How did you discover them?

We have applied various data preprocessing Like initially described the data you know it's statistical values like percentile mean Max count.
To deal with missing values initially went with filling the null values with the help of **dataset.isnull().any()** It Checks whether any column has a null or a missing value.
Then dataset.fillna()
But since this approach wasn't giving us the desire accuracy score for our training model and we decided to simply drop the null values by
Dataset.dropna()
We moved on to feature detection. What we actually need to do is to find the contribution of each column in the data set towards the targeted column. For example in this data set the **targeted column is is_lead** so basically we need to find all other columns' contribution and its distribution towards the **is_lead column** which is the targeted column.So with the help of Sklearn library we have imported feature detection method
Example is given below.
For instance, we can perform a $\chi2$ test to the samples to retrieve only the two best features as follows:

```
>>> from sklearn.datasets import load_iris

>>> from sklearn.feature_selection import SelectKBest

>>> from sklearn.feature_selection import chi2

>>> X, y = load_iris(return_X_y=True)
>>> X.shape
output>>>>(150, 4)
```

```
>>> X_new = SelectKBest(chi2, k=2).fit_transform(X, y)
>>> X_new.shape
output>>>>(150, 2)
```

Then with the help of **Data visualisation** important libraries like matplotlib seaborn we have plotted some data visualisation graphs.

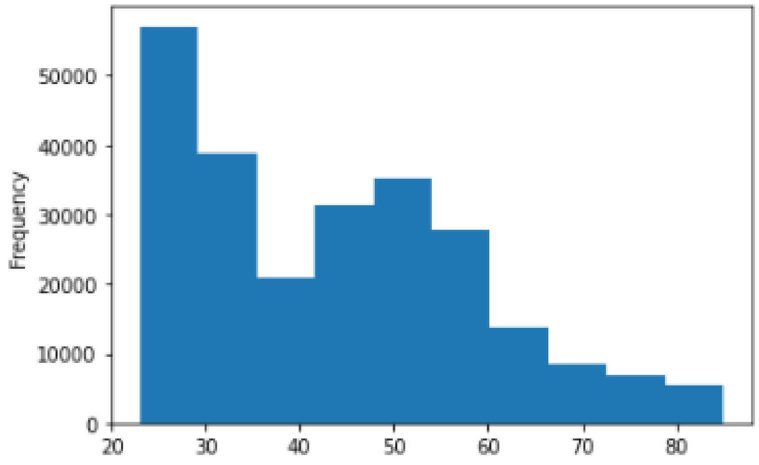


```
test.Age.plot.hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd37ec5ec10
```

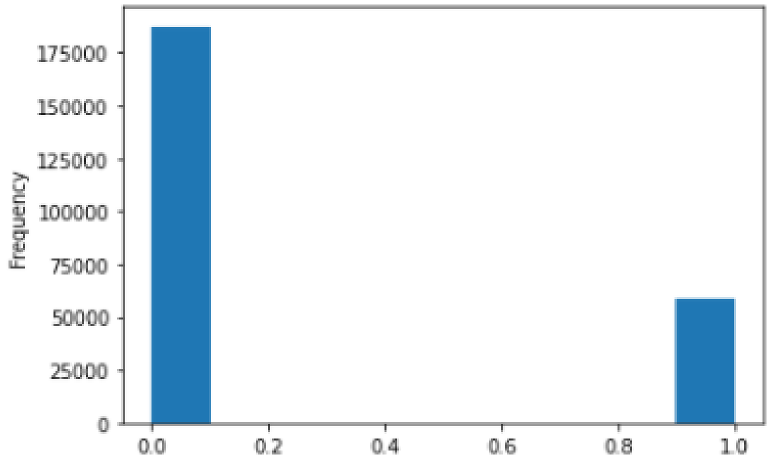
```
train.Age.plot.hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd37eb38b10>
```



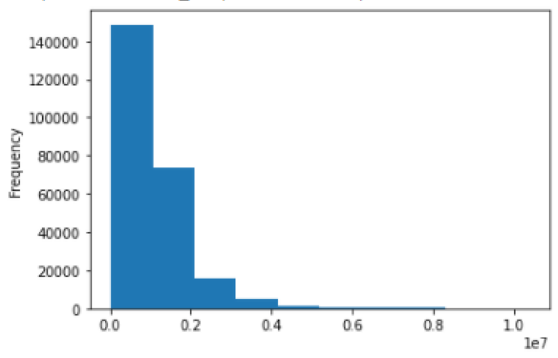
```
[5]  train.Is_Lead.plot.hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd38136cfd0>
```



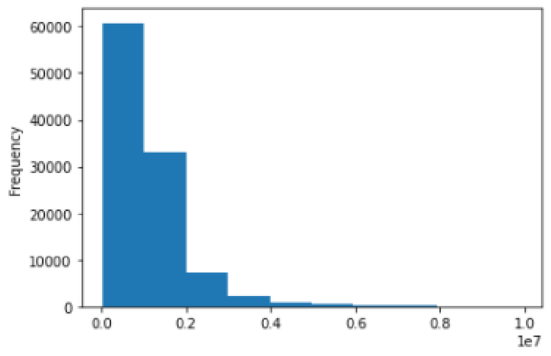
```
train.Avg_Account_Balance.plot.hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd37dea5990>
```



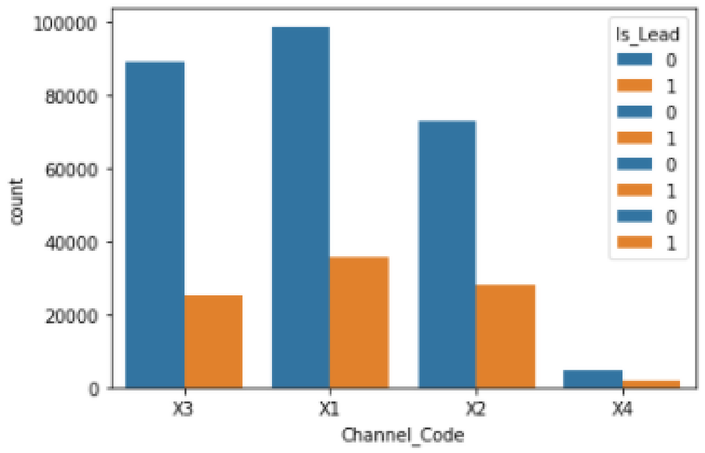
```
[11] test.Avg_Account_Balance.plot.hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd37de81e50>
```



```
14] sns.countplot(data=train,x='Region_Code', hue='Is_Lead')
    sns.countplot(data=train,x='Gender', hue='Is_Lead')
    sns.countplot(data=train,x='Occupation', hue='Is_Lead')
    sns.countplot(data=train,x='Channel_Code', hue='Is_Lead')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd37cfc4e10>
```



3.What does your final model look like? How did you reach it?

After getting the desired columns which can contribute to its target column being moved onto building our model by using various algorithms with help of Sklearn library .We have splitted the data set into train and test then We trained the data set .We have trained our model with several algorithms like

- Catboostregressor
- XGBclassifier

We had a great competition between the **Xgboostclassifier and catboostregressor**

With accuracy of 0.8707 , 0.87198 respectively

Again we went to the procedure by cleaning some data by filtering some data and finally we got our desired accuracy with **catboostclassifier**