1) Changing column names of the dataset that contain '.' To '_'

2) Load dataset into spark and do some exploration (EDA) on data schema and patterns

3) Make hypothesis (contacting a client previously encourages him to subscribe to the term deposit)

4) Do data cleaning:
    a. Changing column "y", changing "yes/no" to "0/1"
    b. Seeing what columns contain "unknown" and changing it to appropriate value
    c. Remove "default" column because it has no meaning
    d. Changed columns: ['job', 'marital', 'education', 'housing', 'loan']

At this initial stage, data is clearly "big" because it requires Spark, a distributed computing system designed to handle large-scale data processing.

5) Moving cleaned dataset to HDFS and doing the mapReduce that categorizes the people into 4 ages, 0(0-19) 1(20-39) 2(40-59) 3(60+) making the age categorical instead of continuous

6) Moving the MapReduced file back to Spark and making sure everything is okay

Using Hadoop and MapReduce signifies that the data is still large and complex. Hadoop's distributed processing is essential for handling such data.

7) Convert the Spark DataFrame to a Pandas DataFrame

The transition to using Pandas alongside Spark indicates that parts of the data are now manageable in-memory. Pandas operates best with data that fits into a single machine's memory. This stage suggests that the data size might be reducing, particularly the portion that's being visualized.

8) Saving data in MongoDB
    a. Initializing MongoDB client
    b. Create a new database and collection
    c. Convert DataFrame to a dictionary and insert it into MongoDB

Storing data in MongoDB suggests a more refined and structured form of data. MongoDB can handle large datasets, but the data stored for further analysis or querying is often smaller and more manageable.

9) Load data saved on MongoDB on a Sandbox environment to do some data manipulation and the machine learning hypothesis modeling.

By the time data reaches the sandbox environment for machine learning hypothesis modeling, it has typically been sampled, cleaned, and preprocessed. This data is usually small enough to fit into memory for efficient training and testing of ML models.

10) First start with LR model, I treated the previous column as categorical in order to see how each value contributes to the prediction, and extract the coefficients of the model to see this

11) One-hot encode the categorical features since ML models can't deal with categorical columns so we make them vectors of binary (0/1), separate features and target, split the data and check for NaNs

categorical features: ['job', 'marital', 'education', 'housing', 'loan', 'contact', 'month', 'day_of_week', 'poutcome', 'previous']

12) Create a pipeline to scale the data and then apply logistic regression.

Scaling the data ensures that all features have the same scale (mean-variance normalization). This is important because many machine-learning algorithms, including logistic regression, perform better when the input features are on a similar scale.

13) Confusion matrix and classification report to show performance and accuracy
14) Create Feature vs Coefficients graph making it easy to understand how number of contacts affect the outcome
15) Do the same procedure of categorical (without previous), one-hot and splitting
16) Train RF model and output report and confusion matrix
17) Compare both with ROC curves
18) Draw feature importance using RF
19) Draw Shap summary plot
20) Conclusion about the hypothesis (weak)