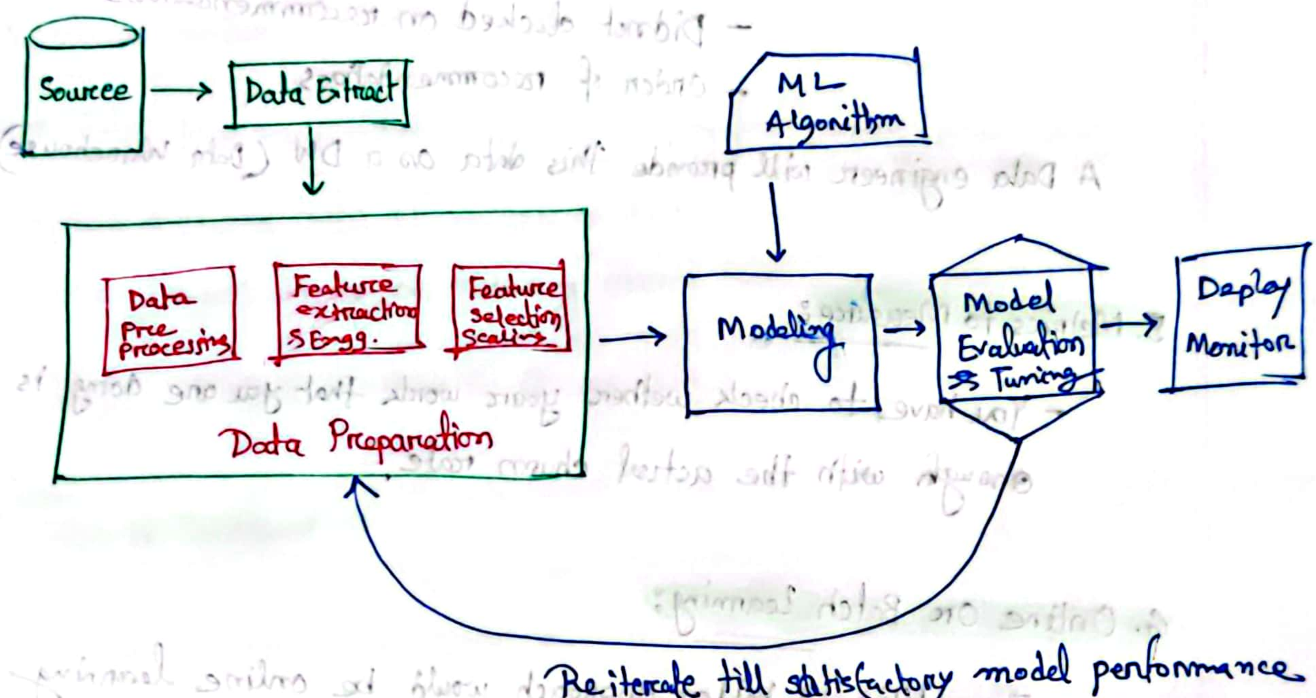
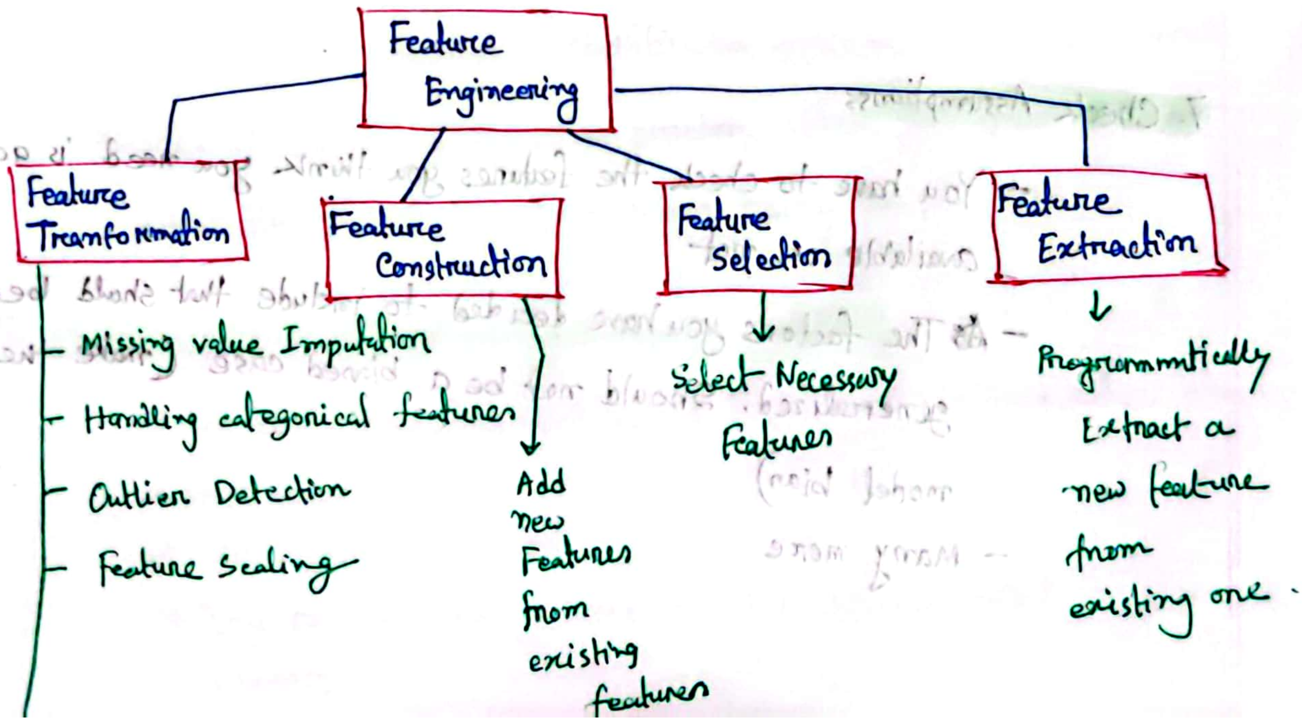
Data Understanding, Data fetching techniques, EDA Techniques are noted in Jupyter Notebook.

Feature Engineering:





Feature Scaling (Standardization) : Already Noted Before.

Lab instructions:

- ✓ Import libraries
- ✓ Import social Network dataset and take three features from it (Age, Estimated Sal, Purchased)
- ✓ Train, test, split
- ✓ Standard Scaler
- ✓ Convert scaled data to DF again
- ✓ check df.describe(). on both X_train and scaled dataset
- ✓ Plot 2 subplots of before and after scaling data to check what changes happen
- ✓ Plot KDE Plot in the same way
- ✓ Check individual Distributions of Age and Salary using KDE plot (Before and after)
- ✓ Apply logistic Regression on both scaled and unscaled data to check who performs better. If scaling actually matters or not?
- ✓ Apply Decision Trees in the same way to understand that not ~~algo~~ all algorithm will work better on scaling data.

Encoders: Most popular encoders are →

1) Ordinal Encoder
2) Label Encoder [~~Att~~ Basic workings
3) One Hot encoder already noted]

When to use what?

Ordinal encoder → Use it in ordinal category feature where you need to provide orders. But use it only for Independent Features

Label encoder → It has been used for to encode Target features which are categorical

One hot encoder → It is used in Nominal category features who are not an ordered feature (Means they can't be ordered)

Dummy Variable Trap:

After doing one hot encoding, for a categorical feature which has n unique categories, for that n new colum's get created. But for n unique categories, from the n created new columns, we have to delete 1 column To solve the problem of "multi Collinearity". (Generally it has to be the 1st column which needs to be deleted). So for n categories (n-1) columns will be kept after One hot encoding.

In a machine learning model, the input (independent) features should be ~~not~~ independent to each other means they should not have any mathematical relationship with each other. So, after one hot encoding as n new columns (features) get created, they show a mathematical relationship with each other for which it becomes a problem for some linear models like (Linear Regression, Logistic Regression). So to solve the problem we need to remove 1 column. After removing 1 column with (n-1) columns, n categories still can be represent. So that won't be a issue.

One hot encoding using most frequent Categories:

Sometimes a categorical column can have so much columns that it won't ~~we~~ be wise to create n new columns with that altere one hot encoding. In those cases, we do a little bit modified one hot encoding. Suppose, we create new columns for those categories which are more frequent in the data column, and for remaining categories, we make a single feature named "Others" and encode all the lower frequent categories with that column.

The threshold of shifting categories to "others" can be decided by person.