I used python 3 software in jupyter notebook environment. The libraries I used are

- Pandas-used for adding/updating DataFrame
- Numpy for maths and linear algebra
- Sklearn-A tool for machine learning
- Matplotlib and seaborn for visualization
- Tensorflow for deep learning

Preprocessing

The csv which was converted into data frame using pandas. First, I did label encoding of categorical data. I dropped these features 'Unnamed: 0','EASE-MENT','ADDRESS','APARTMENT NUMBER','SALE DATE'. These features were not related to target.
There were missing values in target which I removed (tried filling it with mean which didn't work, i.e. the distribution had many local mean). Then I imputed missing values of features(numerical) with its mean.
I explored data of sale price where I removed the outliers and brought the target into normal distribution.
I scaled all the features using min max scaling (categorical scaling- since it was label encoded not one hot encoded). I used PCA to reduce features into principal components of 12 while retaining 99% variance

Modelling

- The algorithmic method that I applied were boosting,Linear regression, deep learning.
- RMSE for gradient boosting and deep learning was lowest.
- The method of gradient boosting - fitting a weighted additive expansion of simple trees - represents a very general and powerful machine learning algorithm and is used a lot for prediction. Gradient bosting has a framework for iteratively improving any weak learner. Since the task was to reduce root mean squared error I used gradient boosting regressor algorithm from sklearn(a tool for machine learning). Parameter tuning using skearn's grid search method was done. With number of estimators of 130 and learning rate of 0.1, max depth of 5,min samples split of 800 and KFold CV of 5, I got RMSE of **338224.76**
- In my recent projects I have used deep learning algorithm and I wanted to give a try on this case study to see how it goes with house price prediction. In deep learning, the model consists of four hidden layers. The first layer contains 512 neurons. Subsequent hidden layers are 256, 128 and 64 neurons respectively. The activation function I used was rectified linear unit (ReLU) and the optimizer used to reduce mean squared error was adam optimizer. Number of epochs were 1000 with batch size of 256. I didn't used K fold cross validation(computationally very expensive). I got RMSE of **12652.54** using random shuffle of training data

Testing Data

For new test data we must preprocess features in the same way, i.e. cleaning, scaling, reducing dimensions etc. Using the weights of learned model we can easily predict the sale price