MVP

Project :California Houses Prices Prediction

Dataset:
https://www.kaggle.com/camnugent/california-housing-prices

Project Description:

I want to use California datasets to predict the median houses prices in any area in California State. I done some exploring for the data to get better understanding as well as some preprocessing for the data to build the machine learning model which is a linear regression model.

- Importing libraries and printing the head of the datasets

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
import seaborn as sns
from sklearn.metrics import mean_squared_error
from pandas.plotting import scatter_matrix
housing = pd.read_csv("C:/Users/IT676/Downloads/housing.csv")
housing.head()
```

Out[141…

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	NEAR BAY
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	NEAR BAY
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	NEAR BAY
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	NEAR BAY
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	NEAR BAY

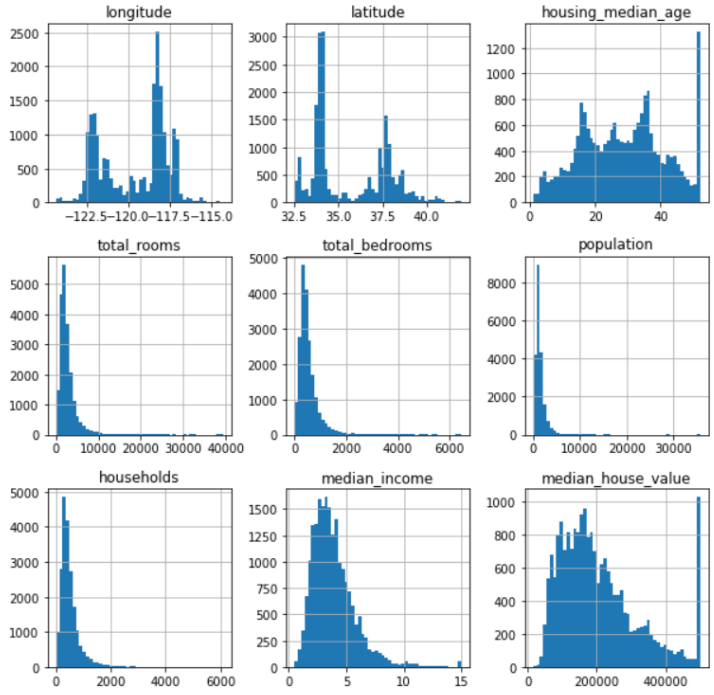
- Information about the dataset

```
housing.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   longitude           20640 non-null  float64
 1   latitude            20640 non-null  float64
 2   housing_median_age  20640 non-null  float64
 3   total_rooms         20640 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20640 non-null  float64
 6   households          20640 non-null  float64
 7   median_income       20640 non-null  float64
 8   median_house_value  20640 non-null  float64
 9   ocean_proximity     20640 non-null  object
dtypes: float64(9), object(1)
memory usage: 1.6+ MB
```

- Plotting the histogram of the features(columns)

```
import matplotlib.pyplot as plt
housing.hist(bins=50, figsize=(10, 10))
plt.show()
```



- Splitting data and adding extra features

```
y = housing['median_house_value']
x = housing.drop('median_house_value', axis = 1)
x['total_bedrooms'].fillna(x.total_bedrooms.median(), inplace = True )
```

```
x['rooms_per_house'] = x['total_rooms']/x['households']
x['bedrooms_per_room'] = x['total_bedrooms']/x['total_rooms']
x['population_per_houshold'] = x['population']/x['households']
x_train, x_test, y_train, y_test = train_test_split(x, y)
```

```
len(x_train)
```

```
15480
```

- Correlation matrix for training data

```
train = x_train.join(y_train)
corr_mat = train.corr()
```

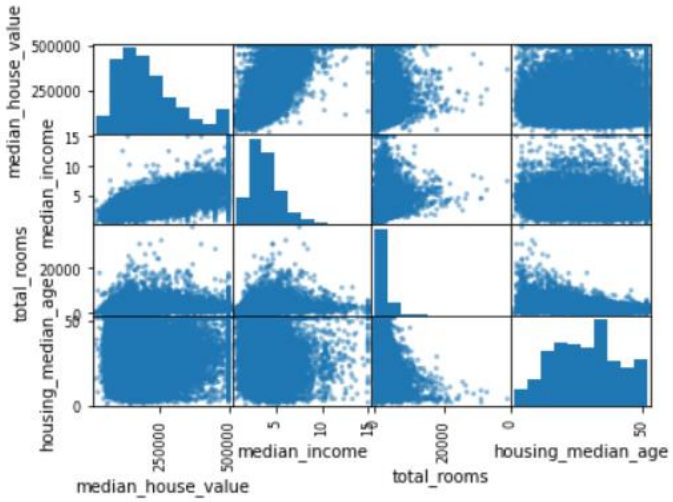
```
corr_mat['median_house_value'].sort_values(ascending = False
```

```
median_house_value          1.000000
median_income               0.689846
rooms_per_house             0.147699
total_rooms                 0.135577
housing_median_age          0.101256
households                  0.067721
total_bedrooms              0.049218
population                 -0.021525
population_per_houshold    -0.021730
longitude                  -0.046076
latitude                   -0.144464
bedrooms_per_room          -0.227088
Name: median_house_value, dtype: float64
```

- Plotting the scatter plot for matrix

```
housing_cols = train[['median_house_value','median_income','total_rooms','housing_median_age']]
```

```
scatter_matrix(housing_cols);
```



- Building and trining machine learning model (linear regression)

```
lm = LinearRegression()
x = train['median_income'].values.reshape(-1,1)
y = train['median_house_value']
lm.fit(x,y)
predict = lm.predict(x)
```

- Prediction values:

```
predict
```

```
array([270326.48920241, 498076.78368883, 215866.29098622, ...,
       440546.15415523, 253981.70002508, 319238.88069577])
```

- Accuracy and mean square errors scores:

```
print("Training Accuracy :", lm.score(x, y))
print("Testing Accuracy :", lm.score(x, y))
```

```
Training Accuracy : 0.4758880496560711
Testing Accuracy : 0.4758880496560711
```

```
mse = mean_squared_error(predict, y)
np.sqrt(mse)
```

```
83327.43137737601
```

- Plotting model Prediction

```
plt.plot(y, predict,'.')

# plot a line, a perfit predict would all fall on this line
x = np.linspace(0, 100,10)
y = x
plt.plot(x, y)
plt.show()
```