MatplotLib

Plot Scatter Plot:

```
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred_binary)
print("Accuracy:", accuracy)
```

Plot Line:

```
%matplotlib inline
plt.xlabel('area')
plt.ylabel('Prices')
plt.scatter(df.area,df.price, color='red', marker='+')
plt.plot(p_df.area, x_pred, color = 'blue')
```

Plot Predicted Value:

```
%matplotlib inline
plt.xlabel('Year')
plt.ylabel('Per Capita Income $')
plt.scatter(canada_df.year, canada_df.percap, color = 'red', marker =
"+")
plt.plot([[x_year]],y_pred , color = 'blue', marker = "+")
```

Pandas:

Rename fields:

```
canada_df = canada_df.rename(columns={'per capita income (US$)':
'percap'})
```

Drop fields:

```
final = merged_df.drop(['town', 'west windsor'], axis = 'columns')
```

Fill median values:

```
median_bedroom = math.floor(df.bedroom.median())
df.bedroom = df.bedroom.fillna(median_bedroom)
```

Replace values in a field:

```
df['Sex'].replace(
    {
        'female': 0,
        'male': 1
    },
    inplace=True)
df.head()
```

Checking for Nulls:

```
X_df_new.isna().sum()
```

Fill field with mean values:

```
df['Age'] = df['Age'].fillna(df['Age'].mean())
```

Feature Extraction:

`One Hot Encoding`

```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder

# Assuming X contains your data and you want to apply one-hot encoding
to the first column
# Replace [0] with the index or list of indices of the columns you want
to encode
column_transformer = ColumnTransformer([('encoder', OneHotEncoder(),
[0])], remainder='passthrough')
X_encoded = column_transformer.fit_transform(X)
X_encoded
```

`Dummies:`

```
dummies = pd.get_dummies(df.town).astype(int)
merged_df = pd.concat([df,dummies], axis = 'columns')
```

Pickle

```
with open('model_pickle','wb') as f:
    pickle.dump(reg,f)

with open('model_pickle', 'rb') as f:
    mp = pickle.load(f)
```

Model Selection

```python
# Split the dataset into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2,
random_state=42)
```