CSC 215-01 Artificial Intelligence (Fall 2019)

Mini-Project 3:
House Price Estimation using Both Visual and Textual Data

Due at 5:30 pm, Wednesday, October 23, 2019

Team Members:

Ayushi Vadwala (220234041)

Jeet Shah (220267750)

Problem Statement:

Most existing automatic house price estimation systems rely only on some textual data like its neighborhood area and the number of rooms. In this project, we practice with algorithms to extract visual features from house photos and combining them with the house's textual information.

The goal of this project is to build a novel automatic house price estimation system by using both textual and visual inputs, other than only using textual information such as area, neighborhood, and number of bedrooms.

Methodology:

- Data Pre-processing like removing Outliers and mean values for all missing values.
- Encode categorical features.
- Normalize numeric features.
- Concatenate houses images into one image for each house.
- Split data into train and test data.
- Using a dense neural network for textual data and CNN model for image data predict "PRICE" of a house.

Step 1: Read the file with Dataset

```
filename_read = os.path.join("/content/gdrive/My Drive/Colab Notebooks/Houses Dataset/HousesInfo.txt")
```

```
cols=["Bedrooms","Bathrooms","area","zipcode","price"]
```

```
df = pd.read_csv(filename_read , sep=" ", header=None , names=cols)
```

Step 2: Remove Outliers from Textual data. Kept the houses with a price between 50K and 900K.

```
df = df.loc[(df['price'] > 50000) & (df['price'] < 900000)]
```

Step 3: For all missing values, inserted MEAN values using definition "missing_median".

```
missing_median(df,'Bedrooms')
missing_median(df,'Bathrooms')
missing_median(df,'area')
missing_median(df,'price')
```

Step 4: Encode the categorical features like 'Bedrooms', 'Bathrooms', 'Zipcode'.

```
encode_text_dummy(df, 'Bedrooms')
encode_text_dummy(df, 'Bathrooms')
encode_text_dummy(df,'zipcode')
```

Step 5 : Normalize numeric features using Z-Score like 'Area'.

```
encode_numeric_zscore(df, 'area')
```

Step 6: Handle Image dataset by Concatenating houses images into one image for each house.

Step 7: Remove Outliers from Image data. Kept images with a price between 50K and 900K.

```
img = img.drop(outlier)
```

```
img.shape
```

```
(452, 4)
```

Step 8 : Create Numpy array of images with a shape (452, 128, 128, 3).

```
img_arr=np.asarray(images_output)
```

```
img_arr.shape
```

```
(452, 128, 128, 3)
```

Step 9 : Split both textual and image dataset using Scikit-learn's "train_test_split" method.

```
x_train_NN, x_test_NN, y_train_NN, y_test_NN = train_test_split(X_NN, y_NN, test_size=0.3, random_state=42)
print("Shape of x_train: {}".format(x_train_NN.shape))
print("Shape of x_test: {}".format(x_test_NN.shape))
print("Shape of y_train: {}".format(y_train_NN.shape))
print("Shape of y_test: {}".format(y_test_NN.shape))
```

```
Shape of x_train: (316, 62)
Shape of x_test: (136, 62)
Shape of y_train: (316, 1)
Shape of y_test: (136, 1)
```

```
img_train, img_test = train_test_split(img_arr, test_size=0.3, random_state=42)
print("Shape of img_train: {}".format(img_train.shape))
print("Shape of img_test: {}".format(img_test.shape))
```

```
Shape of img_train: (316, 128, 128, 3)
Shape of img_test: (136, 128, 128, 3)
```

Using Keras-functional API, created DENSE and CNN models to extract features from text and image separately.

First input model

```
# first input model
model1 = Sequential()
model1.add(Dense(256, input_dim=x_train_NN.shape[1], activation="relu"))
model1.add(Dense(128, activation="relu"))
model1.add(Dense(64, activation="relu"))
model1.add(Dense(32, activation="relu"))
model1.add(Dense(1, activation="relu"))
```

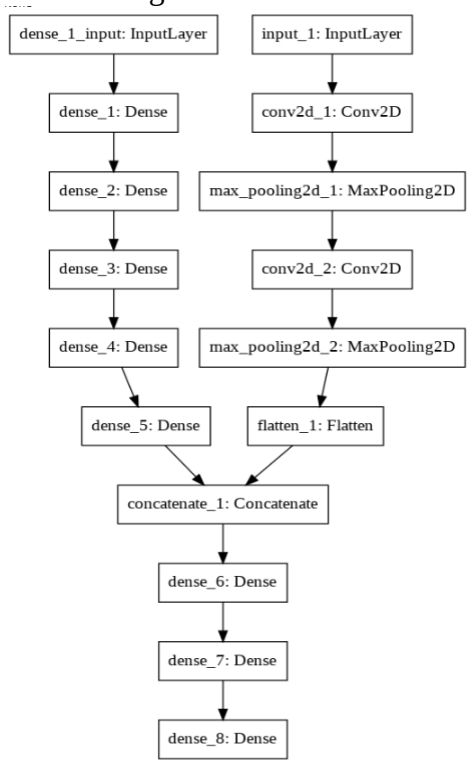
Second input model

```
# second input model
visible2 = Input(shape=(128,128,3))
conv21 = Conv2D(32, kernel_size=4, activation='relu')(visible2)
pool21 = MaxPooling2D(pool_size=(2, 2))(conv21)
conv22 = Conv2D(16, kernel_size=4, activation='relu')(pool21)
pool22 = MaxPooling2D(pool_size=(2, 2))(conv22)
flat2 = Flatten()(pool22)
model2 = Model(visible2, flat2)
```

Concate both input model

```
# merge input models
merge = concatenate([model1.output, model2.output])

# interpretation model
hidden1 = Dense(50, activation='relu')(merge)
hidden2 = Dense(40, activation='relu')(hidden1)
output = Dense(1, activation='relu')(hidden2)

model = Model(inputs=[model1.input, model2.input], outputs=output)
```

Model Design



Model training using EarlyStopping. Took input of both textual and image data and output as "Price" data.

```
model.compile(loss='mean_squared_error', optimizer='adam')
monitor = EarlyStopping(monitor='val_loss', min_delta=1e-4, patience=5, verbose=1, mode='auto')
model.fit([x_train_NN, img_train], y_train_NN,batch_size=64,validation_data=([x_test_NN, img_test], y_test_NN),callbacks=[monitor],verbose=2,epochs=1000)
```

Experimental Results and Analysis

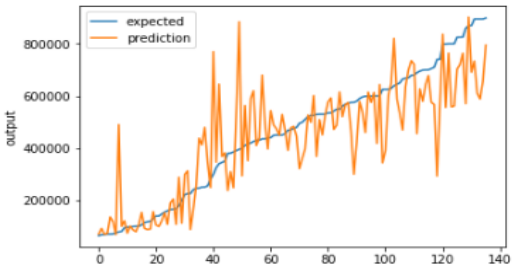
RMSE and R2 Score of a model.

```
from sklearn import metrics

model_pred = model.predict([x_test_NN,img_test])

score = np.sqrt(metrics.mean_squared_error(y_test_NN,model_pred))

print("Score (RMSE) : {}".format(score))
print("R2 score     :",metrics.r2_score(y_test_NN,model_pred))
print("MSE          :", metrics.mean_squared_error(y_test_NN, model_pred))
```

```
Score (RMSE) : 139922.42029878148
R2 score     : 0.6648726642389936
MSE          : 19578283702.268852
```

Regression Lift Chart.

```
chart_regression(model_pred.flatten(),y_test_NN)
```



Task Division and Project Reflection

Name: Ayushi Vadwala

Tasks performed:

Data Preprocessing
Feature Normalization
Encode Categorical Features
Handle Image dataset by concatenating houses images into one image for each house.
Implemented model using Keras-Functional API to extract features from text and image separately.
Prediction for the Test data and compared actual and predicted result.

Name: Jeet Shah

Tasks performed:

Remove Outliers
Fill Mean values for all missing values
Split the data into train and test data.
Implemented model using Keras-Functional API to extract features from text and image separately.
Plotted Regression Lift Chart.

Learnings:

- Analyzing what are categorical features and what are numeric features.
- How to handle image dataset by combining separate images for same house.
- How to apply Keras functional API for house prediction dataset.
- How to Combine Textual and image dataset to use as an input.
- Applying the models and comparing their performance.
- How to implement Deep Neural Network and Convolution Neural Networks together as one model.