AI for house price prediction.

Applying artificial intelligence to real estate.

About me

David Jiménez

Data Scientist in

Predictiva.io

Index

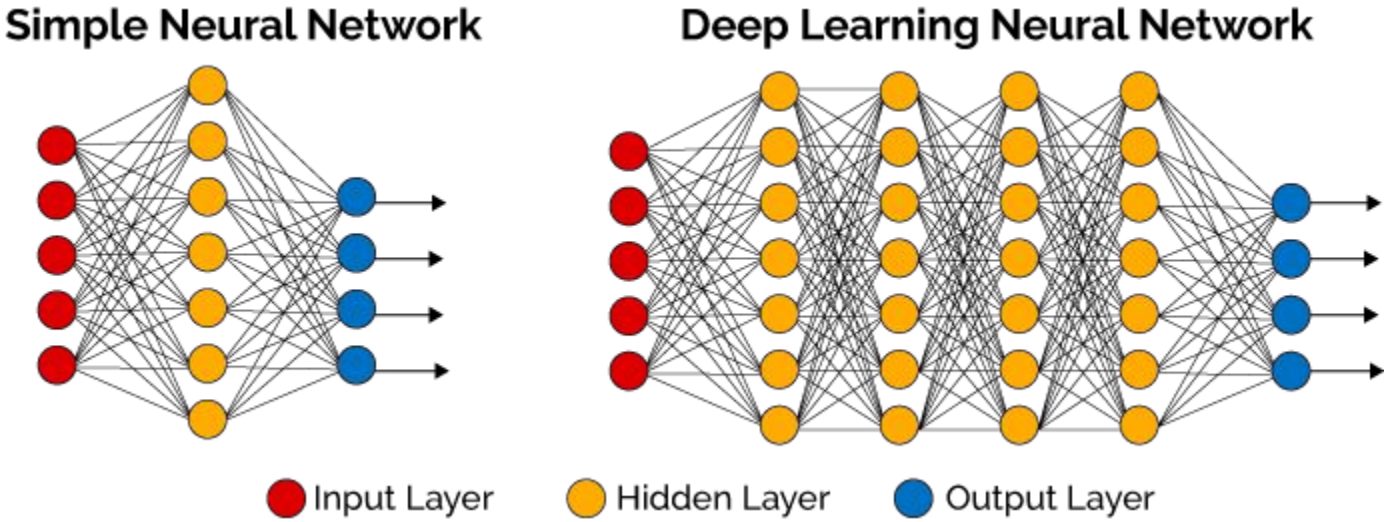
Introduction

Learn how to define a machine learning architecture capable of accepting multiple inputs, including numerical, categorical, and image data. We'll then train a single end-to-end network on this mixed data.

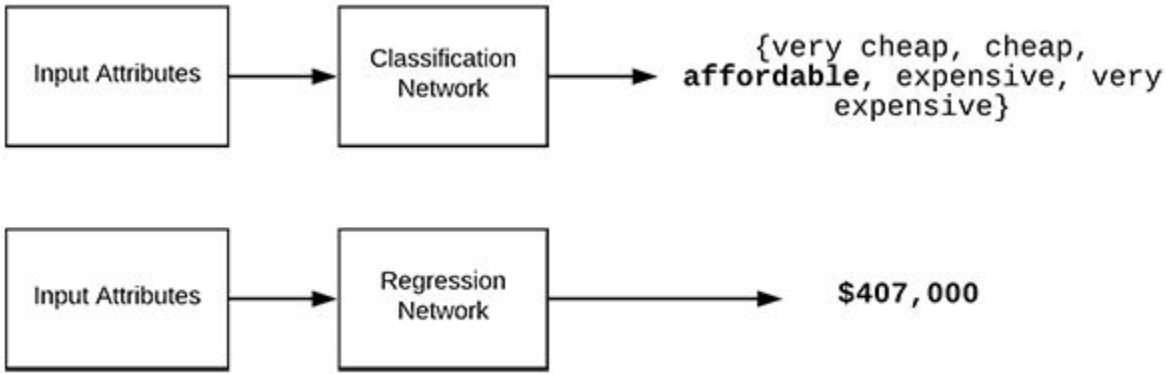
Main tasks

1. Basic regression ML model
2. Training CNN model for regression prediction
3. Multiple inputs and mixed data model

Machine Learning vs Deep Learning



Regression vs Classification



- **Classification** — predicting a label, e. g. to characterize the contents of an image.
- **Regression** — predict continuous values, e. g. task of house price prediction.

The House Prices Dataset


```
>>> import pandas as pd
>>> cols = ["bedrooms", "bathrooms", "area", "zipcode", "price"]
>>> inputPath = "HousesInfo.txt"
>>> df = pd.read_csv(inputPath, sep=" ", header=None, names=cols)
>>> df.head()
   bedrooms  bathrooms  area  zipcode     price
0         4        4.0  4053    85255  869500.0
1         4        3.0  3343    36372  865200.0
2         3        4.0  3923    85266  889000.0
3         5        5.0  4022    85262  910000.0
4         3        4.0  4116    85266  971226.0
>>>
```

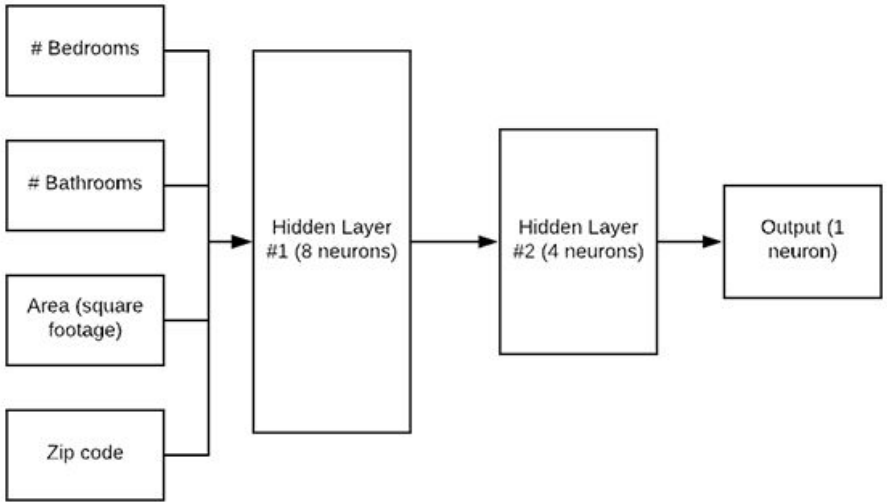
The dataset includes both *numerical/categorical* attributes along with *images* for 535 data points

The house dataset includes **four numerical and categorical attributes:**

1. Number of bedrooms
2. Number of bathrooms
3. Area (i.e., square footage)
4. Zip code

Development Environment

Implementing a Neural Network for Regression: Multilayer perceptron (MLP)



MLP Regression Results

The initial mean absolute percentage error (MAPE) starts off as high as 87% and then quickly drops to obtain a final value of **24.67%.**

What does this value mean?

Our final mean absolute percentage error implies, that on average, our network will be ~25% off in its house price predictions with a standard deviation of ~19%.

Limitations of the House Price Dataset

Being 24.67% off in a house price prediction is a good start but is certainly *not* the type of accuracy we are looking for.

That said, this prediction accuracy can also be seen as a *limitation* of the house price dataset itself.

Keep in mind that the dataset only includes four attributes:

1. Number of bedrooms
2. Number of bathrooms
3. Area (i.e., square footage)
4. Zip code

Most other house price datasets include *many* more attributes.

Limitations of the House Price Dataset

The fact that we are able to even obtain 24.67% mean absolute percentage error (MAPE) without the knowledge of an expert real estate agent is fairly reasonable given:

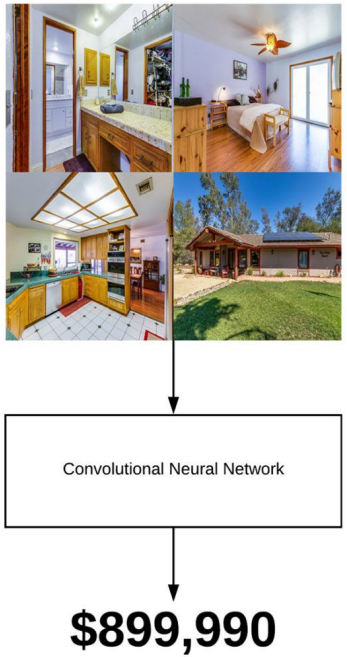
1. There are only 535 total houses in the dataset (we only used **362 total houses** for the purpose of this guide).
2. We only have **four attributes** to train our regression model on.
3. The attributes themselves, while important in describing the home itself, **do little to characterize the area surrounding the house.**
4. **The house prices are incredibly varied** with a mean of $533K and a standard deviation of $493K (based on our filtered dataset of 362 homes).

How can we better our house price prediction accuracy?

- What if we *leveraged images* for each house? Would that improve accuracy?
- Is there some way to *combine* both our categorical/numerical attributes with our image data?

*CNN: Convolutional Neural Network

Train a CNN model for regression prediction



Four images of each house are also provided:

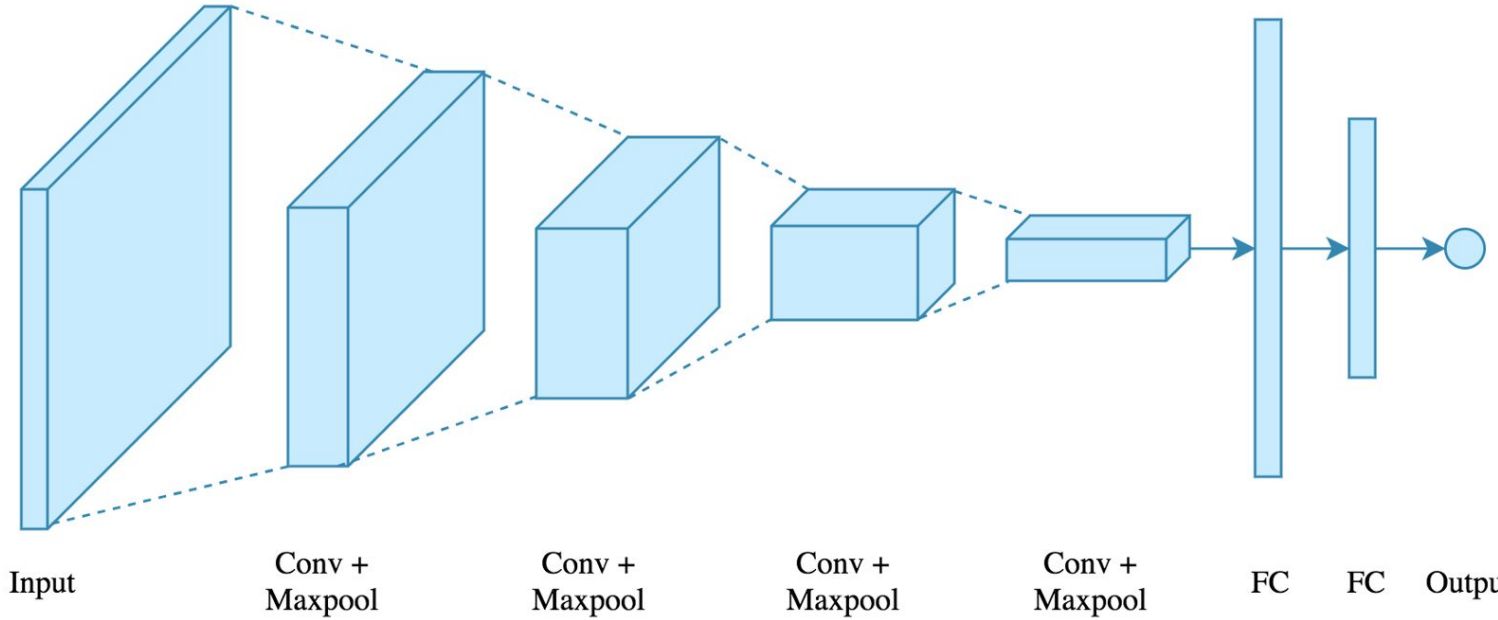
1. Bedroom
2. Bathroom
3. Kitchen
4. Frontal view of the house

A total of 535 houses are included in the dataset, therefore there are *535 x 4 = 2,140* total images in the dataset.

We'll be pruning that number down to 362 houses (1,448 images) during our data cleaning.

*CNN: Convolutional Neural Network

Train a CNN model for regression prediction



How are we going to use these images to train our CNN?

We have three options:

1. Pass the images *one at a time* through the CNN and use the price of the house as the target value for each image
2. Utilize multiple inputs with Keras and have four independent CNN-like branches that eventually merge into a single output
3. Create a montage that *combines/tiles* all four images into a single image and then pass the montage through the CNN

How are we going to use these images to train our CNN?

The first option is a poor choice — we'll have multiple images with the same target price.

If anything we're just going to end up "confusing" our CNN, making it impossible for the network to learn how to correlate the prices with the input images.

The second option is also not a good idea — the network will be computationally wasteful and harder to train with four independent tensors as inputs. Each branch will then have its own set of CONV layers that will eventually need to be merged into a single output.

Instead, we should choose the third option where we *combine* all four images into a *single image* and then pass *that image* through the CNN (as depicted in figure).

Regression CNN Results

Our mean absolute percentage error (MAPE) starts off extremely high, in the order of 500-4,000% in the first ten epochs; however, by the time training is complete we are at a much lower training loss of 30%.

The problem though is that we've clearly overfit.

While our training loss is 30% our validation loss is at **55.87%**, implying that, on average, our network will be ~56% off in its house price predictions.

Regression CNN Results

CNN method obtain a mean absolute error (MAPE) of **55.87%**, that's a pretty poor result given that our first model where we obtained a mean absolute error of **24.67%, *far better* than CNN model**.

So, what does this mean?

All it means is that the interior of a home doesn't necessarily correlate with the price of a home.
For example, let's suppose there is an ultra luxurious celebrity home in Beverly Hills, CA that is valued at $10,000,000.

Now, let's take that *same home* and transplant it to Forest Park, one of the *worst* areas of Detroit.
In this neighborhood the median home price is $13,000 — do you think that gorgeous celebrity house with the decked out interior is still going to be worth $10,000,000?

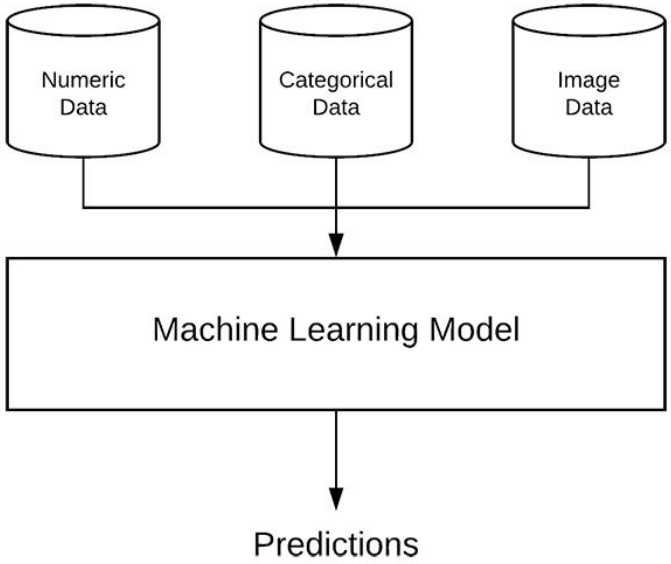
Regression CNN Results

There is more to the price of a home than *just* the interior. We also have to factor in the local real estate market itself.

There are a *huge number of factors* that go into the price of a home but by in large, one of the *most important attributes* is the locale itself.

Therefore, it shouldn't be much of a surprise that our CNN trained on house images didn't perform as well as the simple MLP trained on the numerical and categorical attributes.

Multiple Inputs and Mixed Data



In machine learning, mixed data refers to the concept of having multiple types of independent data.

We would have multiple types of input data, including:

1. **Numeric/continuous values,** such as age, heart rate, blood pressure...
2. **Categorical values,** including gender and ethnicity
3. **Image data,** such as any MRI, X-ray, etc.

All of these values constitute different data types; however, our machine learning model must be able to ingest this “mixed data” and make (accurate) predictions on it.

Mixed Dataset

We include both **numerical/categorical data** along with **images data** for each of the 535 example houses in the dataset.

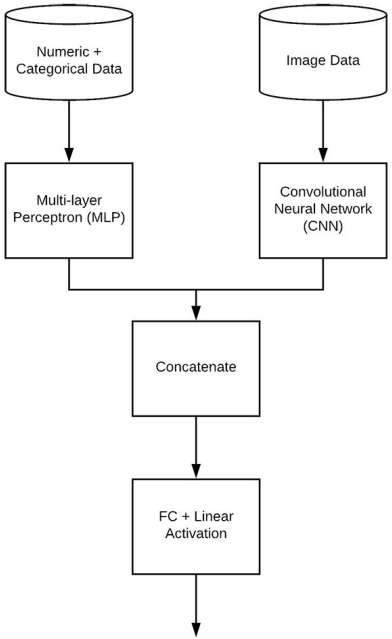
The numerical and categorical attributes include:

- Number of bedrooms
- Area (i.e., square footage)
- Number of bathrooms
- Zip code

A total of four images are provided for each house as well:

- Bedroom
- Kitchen
- Bathroom
- Frontal view of the house

Defining our Multi-layer Perceptron (MLP) and Convolutional Neural Network (CNN)



Multi-input and mixed data results

By the end of training, we are obtaining of **21.36%** mean absolute percentage error (MAPE) on our testing set, implying that, on average, our network will be ~21% off in its house price predictions.

Let's compare this result to our previous two posts in the series:

1. Using *just* an MLP on the numerical/categorical data: **24.67%**
2. Using *just* a CNN on the image data: **55.87%**

Results

As you can see, working with mixed data by:

1. Combining our numerical/categorical data along with image data
2. And training a multi-input model on the mixed data…

…has led to a better performing model!

MLP	CNN	MLP + CNN
24,67	55,87	21,36

MAPE results, low is better.

Thank you.