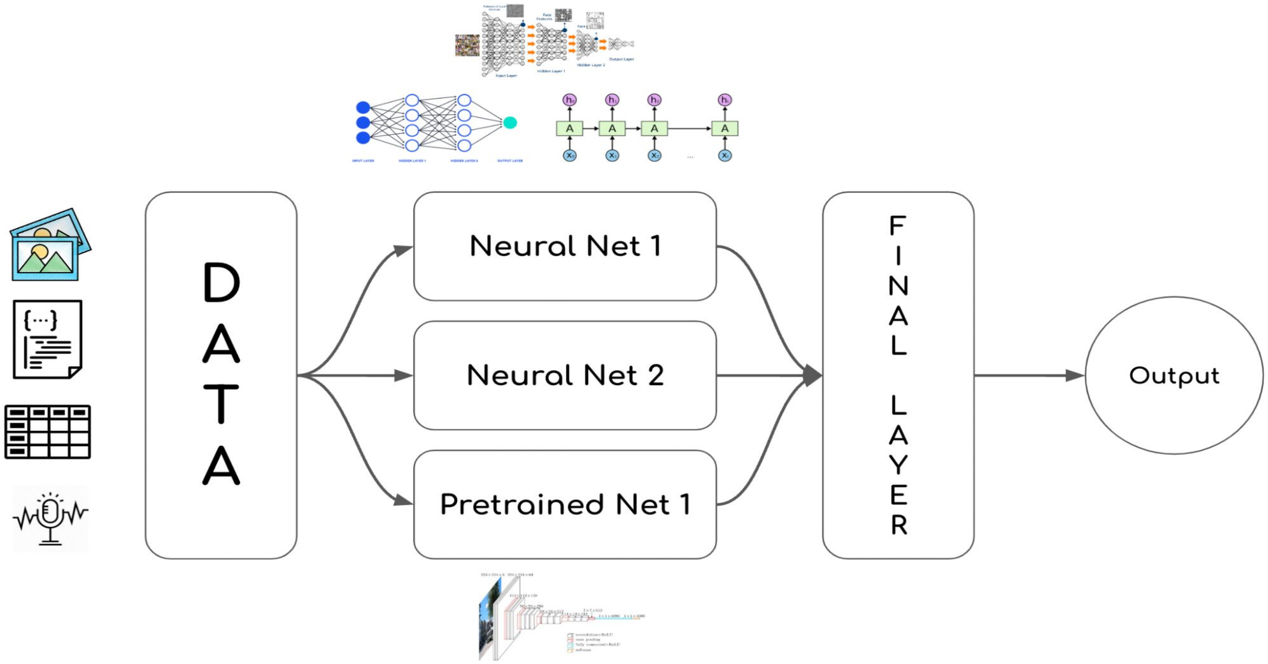
How to Stack Neural Networks together ?
Ideas and Applications >>>

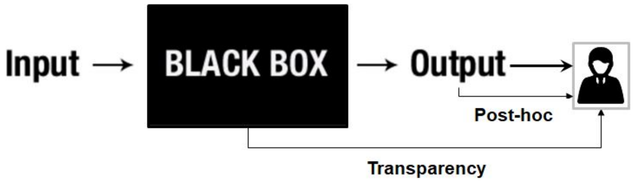


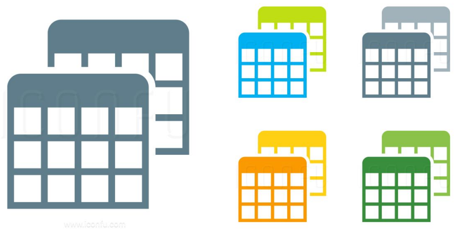
Hi, It's me!

Bridging gaps between Human and Machine Intelligence as an AI Scientist @ Polymerize.io

Building AI tools to enable faster material development cycles, accelerate R&D and add operational efficiency at every stage of the process

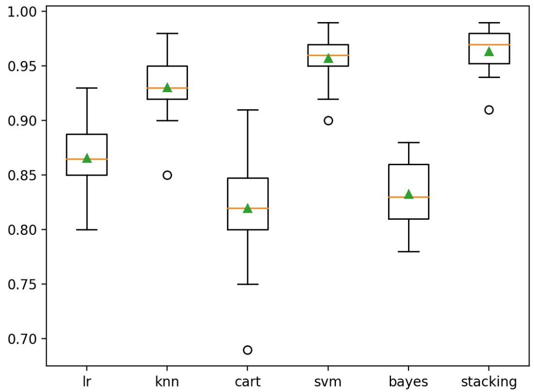
- Python !
- Data Science and Machine Learning to solve problems facilitating data-driven decision making
- Built an AI platform for Marketing Attribution, Planning and Optimisation enhancing ROI's to 1.5X
- Core areas
  - Explainable AI
  - Tabular Neural Nets - Issues, Ideas
  - Reinforcement Learning
  - Optimisation to exploit learned patterns (by models)
  - MLOps - Agile learning, distributed training and tuning



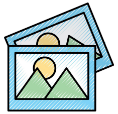


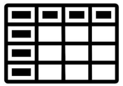
Stacking Models

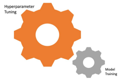
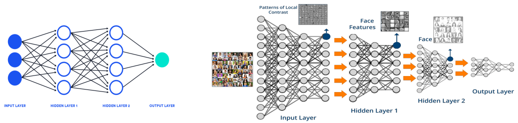
- **What is Stacking ?**
  - Ensemble technique with multiple independent (*vs.* **Boosting**), heterogeneous (*vs.* **Bagging**) models learning parallely with a meta-model / layer to combine outputs from each and work as a single model
- **Why Stack ?**
  - Access to capabilities of diverse models to make predictions with better performance and reduced error variance
- **Why Stack Neural Networks ?**
  - Enhances capability by incorporating multiple informative sources and types of data - Image, Text, Tabular, Speech etc.
  - Each model fine-tuned separately to drive additional value from specific datasets with added advantage of handling sparsity, weight and bias limitations, dimensionality, outliers and multitude of feature types - primary, secondary etc.
  - Explainable AI for every individual model, at a layer level for each data type
  - Applicability of Transfer Learning with pretrained models to push boundaries with smaller datasets



IC : machinelearningmastery.com

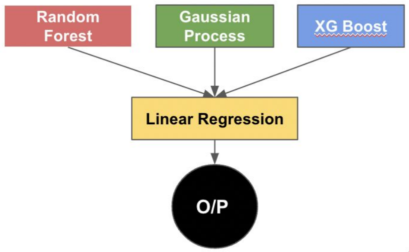




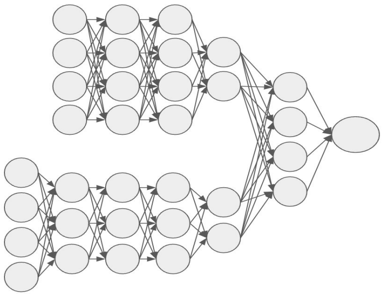



What we’re building today ?

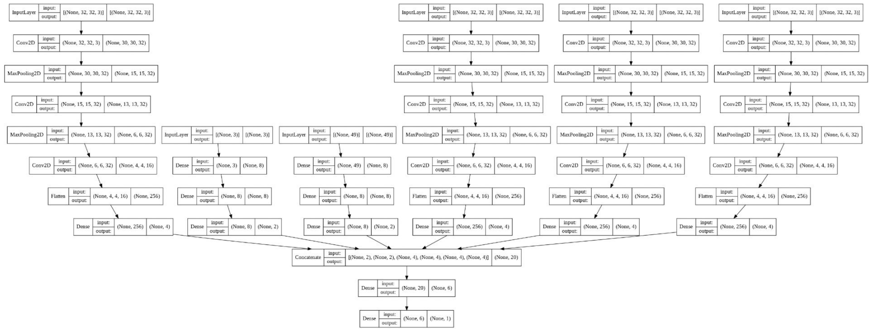
- **Stacking Classical ML Models**



- **Stacking 2 Deep Neural Nets together**

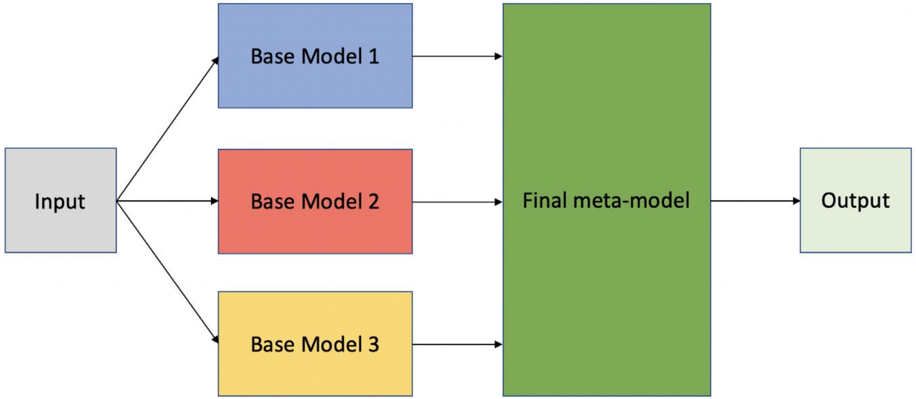


- **Stacking**
  - **Numerical DNN**
  - **Categorical DNN**
  - **4 CNN’s**



Stacking Classical ML Models

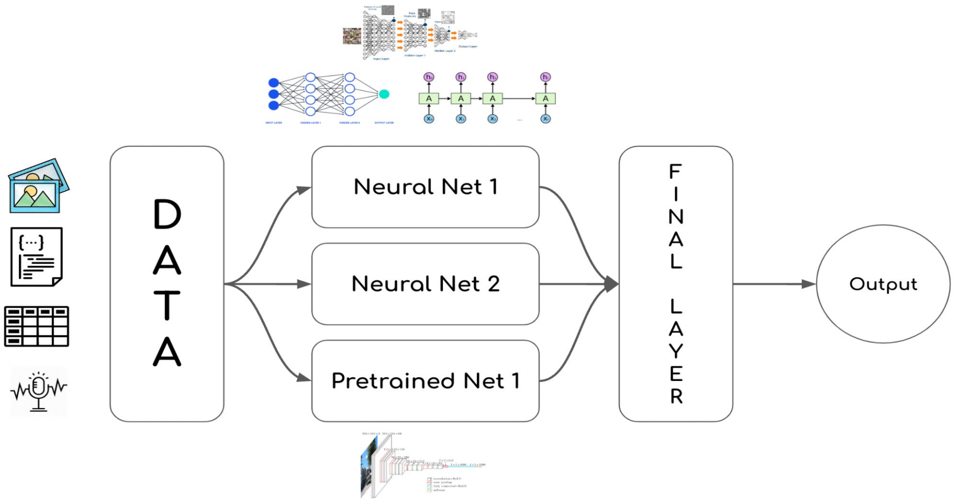
- Simple Implementation
  - sklearn.ensemble.StackingRegressor
  - sklearn.ensemble.StackingClassifier
- Base estimators / beta models
- Final estimator / meta model



Let's head over to the code to check out the implementation >>>

Fundamental Idea and Decisions

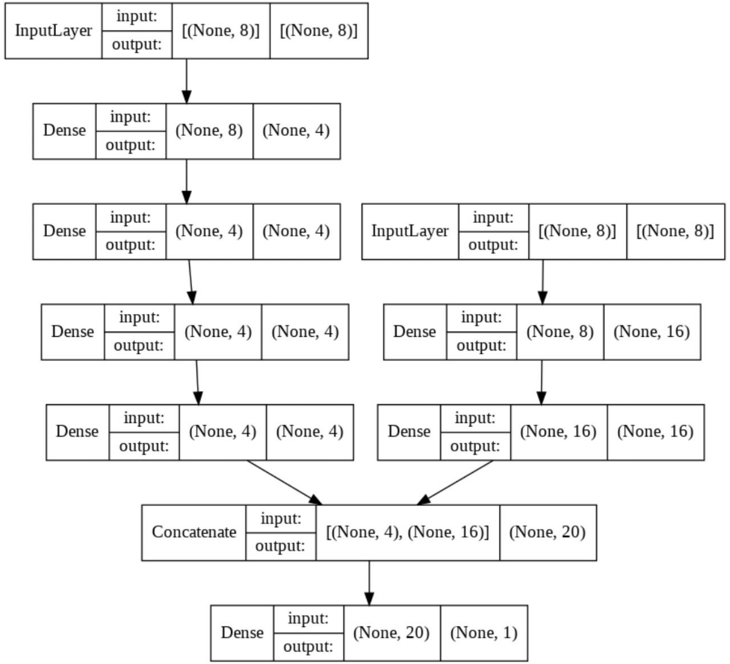
- Data based model selection
  - Same dataset different models / model types
  - Different dataset different models / model types
- Training Process
  - Individual (not trained with final layer)
  - Combined (trained together)
  - Individual + Combined (layer selective training)
- Final Layer dimensions and depth
  - Single
  - Multiple
- Hyperparameter Tuning
  - Individual
  - Combined
- Pre-trained Models
  - Layer selective training (model.trainable = False)
  - Use for high performance dimensionality reduction



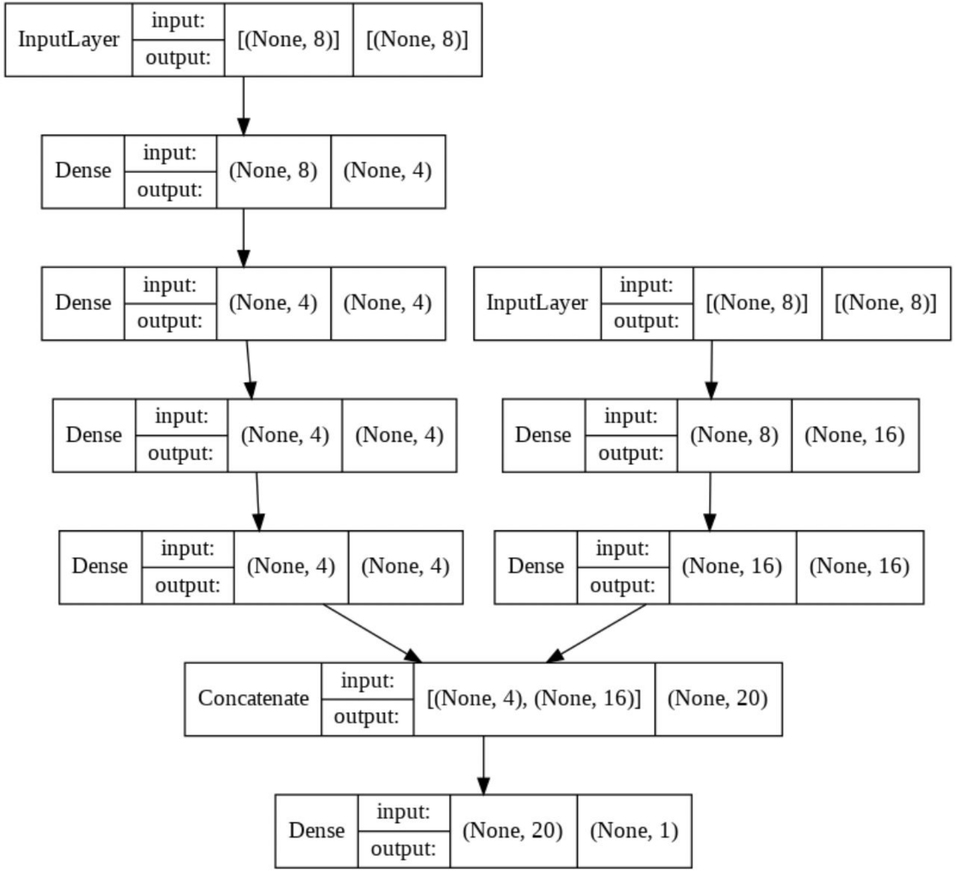
Stacking 2 Deep Neural Nets together

- **Steps to build a simple Stacked DNN**

  - build network1 - any architecture
  - build network2 - any architecture
  - Use ***tf.keras.layers.concatenate*** to concat the *network1.output* and *network2.output*
  - Finally, create a single/multiple Dense layers with required dimensions
  - *use tensorflow functional API to connect the concatenated output of the stacked networks as input to the final dense layer*
  - use ***tf.keras.models.Model*** to combine all of them together with inputs from the individual layers and output as the final dense layer
  - define optimiser and loss functions and finally compile the model
  - model.predict() with the right Input shape



Let’s code this !



Example 1 : Real Estate Price Prediction
[Property Data + Property Images]

- A great read : https://arxiv.org/pdf/1609.08399.pdf

  https://github.com/emanhamed/Houses-dataset

- Numerical + Categorical Data
- Images of sections of the house
  - Bedroom
  - Bathroom
  - Kitchen
  - Frontal View

```
df.head(5)
```

	number of bedrooms	number of bathrooms	area of the house	zipcode	price
0	4	4.0	4053	85255	869500
1	4	3.0	3343	36372	865200
2	3	4.0	3923	85266	889000
3	5	5.0	4022	85262	910000
4	3	4.0	4116	85266	971226

Houses Dataset
- 100_bathroom.jpg
- 100_bedroom.jpg
- 100_frontal.jpg
- 100_kitchen.jpg

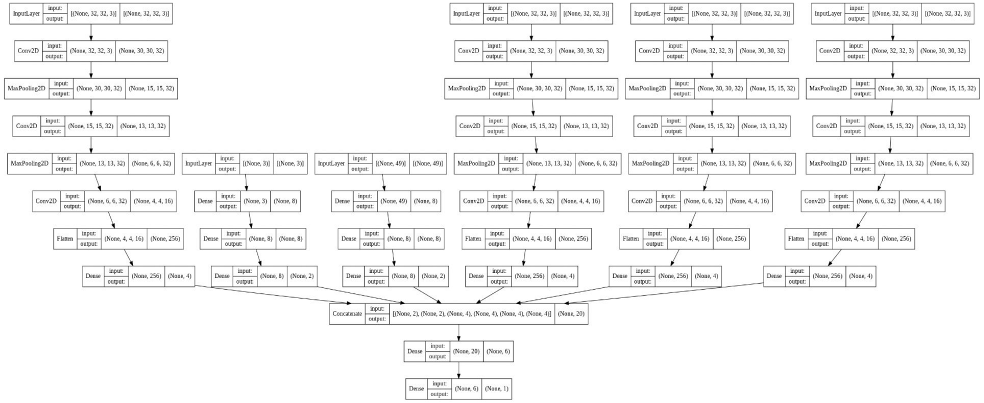
The dataset folder contains 2140 images, 4 images for each house (535 houses)

It contains a text file that contains the metadata of the dataset.

Each row in the file represents the house_number in order. The data has number of bedrooms, number of bathrooms, area of the house, zip code and the price (target)

Stacking - Numerical DNN + Categorical DNN + 4 CNN’s

- Process the data and prepare for building networks
- Build a DNN on Numerical Data
- Build a DNN on Categorical (Label Binarised) Data
- Build 4 CNN’s for each part of the property
    - Bathroom
    - Bedroom
    - Kitchen
    - Frontal View
- Concatenate all of them together
- Add a dense layer post concatenation
- Add a final layer to predict the Property Price
- Train the model
- Make a Prediction



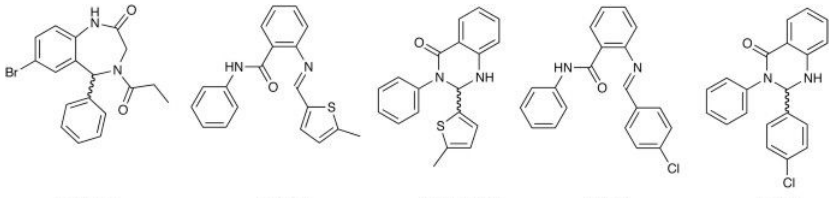
Let’s head to the code one more time!

Other Ideas and Applications

- **Materials Science :** Experiment Data + Compound Structure

Experiment	Ingredient 1	Ingredient 2	Ingredient 3	Ingredient 4	Processing 1	Processing 2	Processing 3	Property 1	Property 2	Property 3
1	64	42	50	73	19	16	21	42	23	55
2	36	66	32	24	34	44	76	72	31	19
3	50	94	74	42	68	43	95	16	29	45

+



- **Customer Retention :** Feedback Text + Call Records

Bennett was very helpful, courteous and punctual for this assignment, which was our company's first foray into Upwork. The completed assignment was well-written, with clear explanations, and no spelling errors, etc. However, the final document was a touch more simplistic/top-level than we expected, given the sum allotted for payment. Overall, I would recommend Bennett for fast-turn jobs. He is very good at providing a solid deliverable to work with and at synthesizing complex information into easily understood copy.

+

+

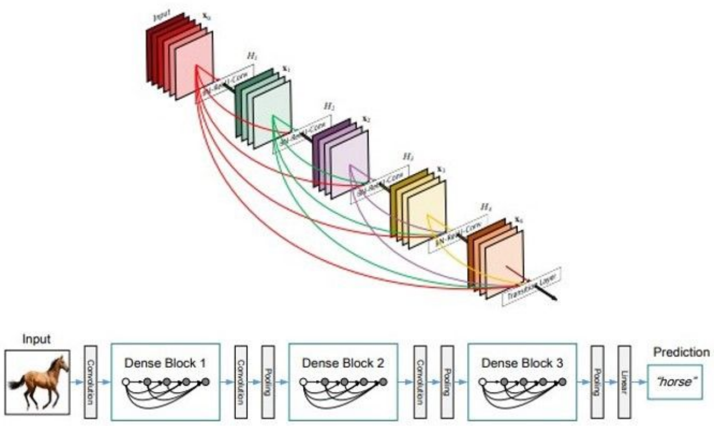
Industry	Customer Start Date	Close Date	Stage	Fiscal Period	Age	Opportunity Ow
Building Materials	12/9/2015	12/8/2015	Closed Won	Q4-2015	5	Zach Patterson
Computer & Network Security	11/1/2015	10/15/2015	Closed Won	Q4-2015	58	Tonni Bennett
Computer Software	8/14/2015	8/14/2015	Closed Won	Q3-2015	84	Tonni Bennett
Computer Software	11/2/2015	10/30/2015	Closed Won	Q4-2015	93	Tonni Bennett
Computer Software	8/31/2015	8/31/2015	Closed Won	Q3-2015	33	Stuart Maron
Computer Software	12/17/2015	12/16/2015	Closed Won	Q4-2015	63	Tonni Bennett
Computer Software	10/20/2015	9/30/2015	Closed Won	Q3-2015	71	Stuart Maron
Computer Software	9/14/2015	9/14/2015	Closed Won	Q3-2015	10	Stuart Maron
Marketing and Advertising	9/29/2015	9/28/2015	Closed Won	Q3-2015	56	Tonni Bennett
Marketing and Advertising	10/27/2015	10/26/2015	Closed Won	Q4-2015	47	Stuart Maron
Marketing and Advertising	10/1/2015	9/30/2015	Closed Won	Q3-2015	40	Stuart Maron
Marketing and Advertising	1/1/2016	12/14/2015	Closed Won	Q4-2015	11	Zach Patterson
Internet	10/1/2015	9/24/2015	Closed Won	Q3-2015	112	Stuart Maron
Internet	8/21/2015	8/20/2015	Closed Won	Q3-2015	6	Stuart Maron
Internet	1/1/2016	12/14/2015	Closed Won	Q4-2015	126	Tonni Bennett
Software Development & Design	10/14/2015	10/13/2015	Closed Won	Q4-2015	42	Stuart Maron
Software Development & Design	9/30/2015	9/30/2015	Closed Won	Q3-2015	41	Stuart Maron

Hyperparameter Tuning Stacked Models

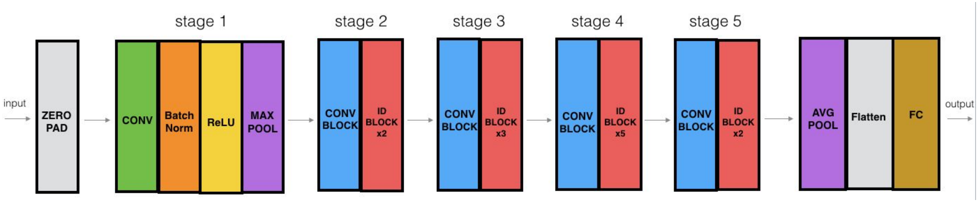
Bonus!

Let’s code a tuner with Ray!

Working with Pretrained Models



Densenet 121



Resent 50

Thanks! Cheers to the love for Python and Data!