Perceptron – Machine Learning Training Algorithms

Shared link:
https://colab.research.google.com/drive/16tZggyrnsJLwhyuKe_--Q4SZ4P6i8Ecv?usp=sharing

Introduction:

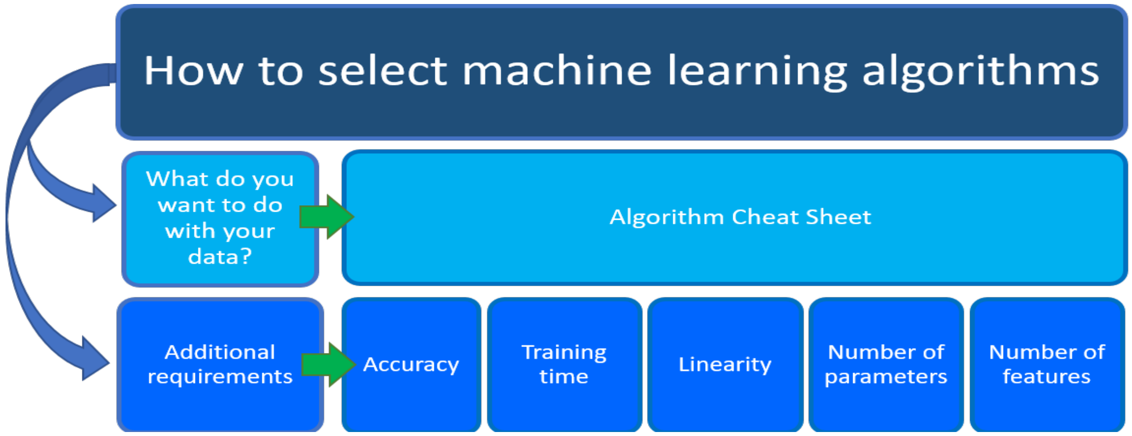
In this project, we use iris dataset to predict the housing prices for the classification task in machine learning. The data can be classified between two or more categories, in this dataset, the data is comma-separated, with fields: sepal_length, sepal_width, petal_length, petal_width. This are also group just the points by species/label so we can plot them. Our task in this project is to prepare the data into classification of perceptron. Using linear models, we can this algorithms to learn and study used for the perceptron training task. Implement of multi-layer perceptron using sklearn, numpy, and seaborn as the imported libraries.

In machine learning, the perceptron is an algorithm for supervised learning to predict values by estimating the relationship between two or more values (functions that can decide whether an independent input, represented by a vector of numbers, influence the dependent or not). It is a type of linear classifier, i.e. a classification algorithm that makes its predictions based on a linear predictor function combining a set of weights with the feature vector. The algorithm allows for online learning, in that it processes elements in the training set one at a time.

Choosing a dataset that needs to be learning from Scratch, we are going to implement a simple linear mode of algorithm using only built-in Python modules and numpy. We will also learn about the concept and the math behind this popular ML algorithm.

Linear Regression. In terms of usage, it does the same work what we used to do in cartesian plane by extending. All the scikit regression models implement two methods fit() and predict(). The data needs to be tested using at least one appropriate learning framework as instructed in the course.

We structured the outline of our report about the implementation: linear model introduction, interface, algorithm and data structure design, testing/evaluation, strategy and results. Comparison towards other available algorithms.



Source: Microsoft Docs website (https://docs.microsoft.com)

The simple linear which operated in neuron network is the *perceptron*, which approximates a single neuron with *n* binary inputs. It defines the following formulas to calculate *w*(weighted sum of its inputs) and to find the learning rate. Parameters and input are used, if that weighted sum is 0 or greater, it will tells us otherwise.

The larger the range of our learning rate, it will results to fast run in the machine learning model. The smaller the range of our learning rate, it will becomes slower to progress with less mistakes made.

Here are formulas below, which generate our aim to determine the "optimal" set of parameters and input for a binary linear classifier given the data.

Algorithms Formulas:

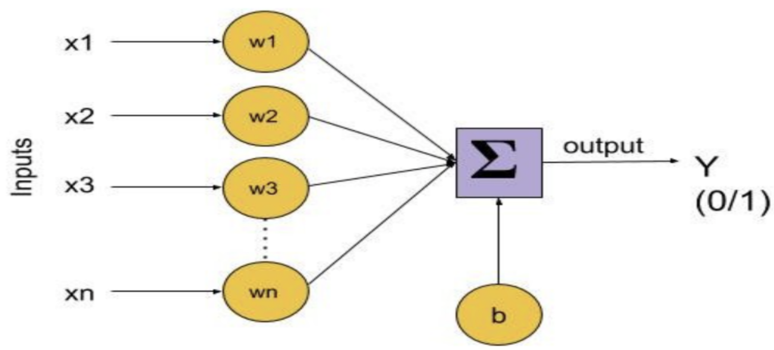
$$w = w + \text{learning rate} * (\text{expected} - \text{predicted}) * X$$

$$w \,.\, x = \textstyle\sum_i w_1 \,.\, x_1 + \ldots\ldots \sum_i w_i \,.\, x_i$$

Where **w** is weight being optimized, **learning rate** is a rate that you must configure (e.g. 0.01), the differences changes to the **expected and predicted** is the prediction error for the model on the training data attributed to the weight and **x** is the input value.

In this section we'll try to build a model that can predict the class (that is, the species) from the first four measurements.

To start with, after we load the data and explore the data. Our nearest neighbors function expects a so let's represent our data that way:



Schematic of Perceptron

Source: W4_Perceptron

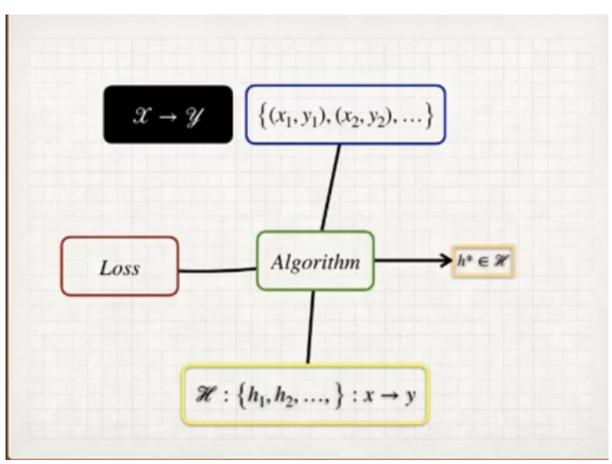
$$\bullet \quad Linear(x) = w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots w_k \cdot x_k + w_0 \cdot x_0$$

$$= w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots w_k \cdot x_k + b$$

The difference between Multi output perceptron and single layer neural network are their purpose and aims to predict between two categories using two-class averaged perceptron which is fast training and linear model. But if we are talking about prediction between several categories, it would be multi-class neural network, it is accurate and longer training times to run the model.

Modelling

Understanding the data, we would like to plot the measurements so we can see how they vary by species. Unfortunately, they are four-dimensional, which makes them tricky to plot. One thing we can do is look at the scatterplots for each of the six pairs of measurements.



Data Exploration & Preparation

Slicing the rows. Let’s split the data into a test set and a training set:

After cleaning we checked in the variable type: Check if all the variables have the correct variable type, based on the data dictionary. If not, then change them.

Machine learning is, more or less, a way for computers to learn things without being specifically programmed. In machine learning, our goal is either prediction after understanding through regression. Prediction is a process where, from a set of input variables, we estimate the value of an output variable.

The below table highlights the summary statistics from the three datasets received. Please let us know if the figures are not aligned with your understanding.

Notable data quality issues that were encountered and the methods used to mitigate the identified data inconsistencies are as follows. Furthermore, recommendations have been provided to avoid the re-occurrence of data quality issues and improve the accuracy of the underlying data used to drive business decisions.

```
1 df_NoDuplicate.describe()
```

	Rooms	Price	Distance	Postcode	Bedroom2	Bathroom	Car	Landsize	BuildingArea	YearBuilt	Lattitude	Longtitude	Propertycount
count	13580.000000	1.358000e+04	13580.000000	13580.000000	13580.000000	13580.000000	13518.000000	13580.000000	7130.000000	8205.000000	13580.000000	13580.000000	13580.000000
mean	2.937997	1.075684e+06	10.137776	3105.301915	2.914728	1.534242	1.610075	558.416127	151.967650	1964.684217	-37.809203	144.995216	7454.417378
std	0.955748	6.393107e+05	5.868725	90.676964	0.965921	0.691712	0.962634	3990.669241	541.014538	37.273762	0.079260	0.103916	4378.581772
min	1.000000	8.500000e+04	0.000000	3000.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1196.000000	-38.182550	144.431810	249.000000
25%	2.000000	6.500000e+05	6.100000	3044.000000	2.000000	1.000000	1.000000	177.000000	93.000000	1940.000000	-37.856822	144.929600	4380.000000
50%	3.000000	9.030000e+05	9.200000	3084.000000	3.000000	1.000000	2.000000	440.000000	126.000000	1970.000000	-37.802355	145.000100	6555.000000
75%	3.000000	1.330000e+06	13.000000	3148.000000	3.000000	2.000000	2.000000	651.000000	174.000000	1999.000000	-37.756400	145.058305	10331.000000
max	10.000000	9.000000e+06	48.100000	3977.000000	20.000000	8.000000	10.000000	433014.000000	44515.000000	2018.000000	-37.408530	145.526350	21650.000000

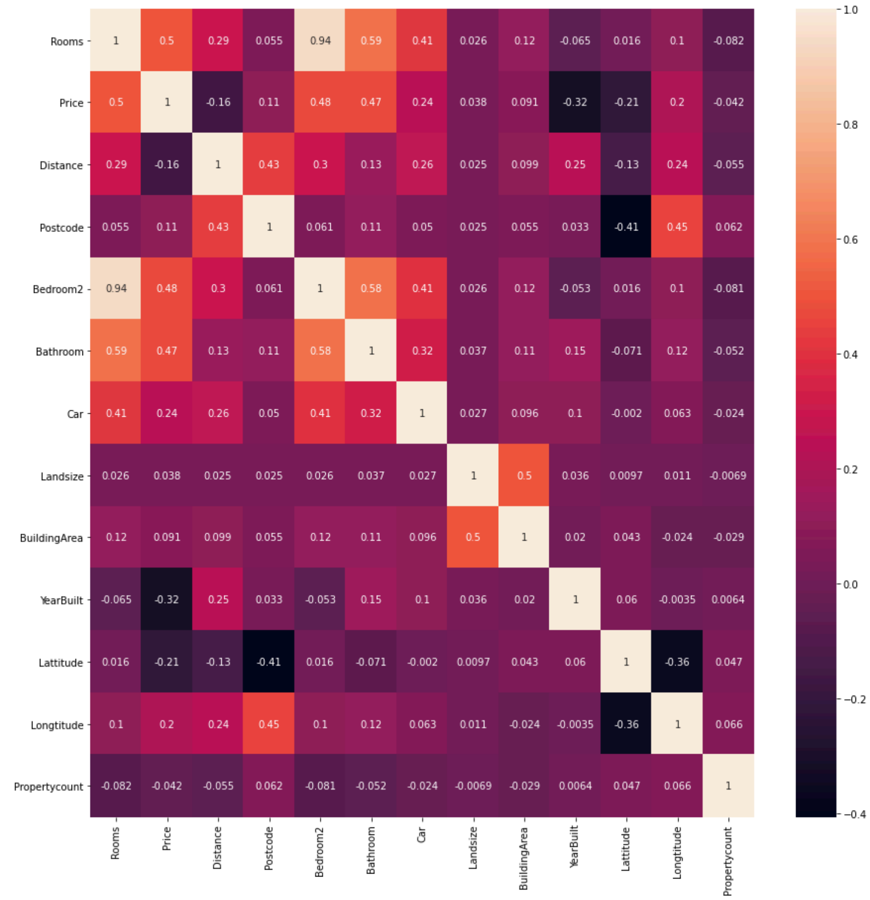
Summary of the housing prices in Melbourne, Australia

Evaluation

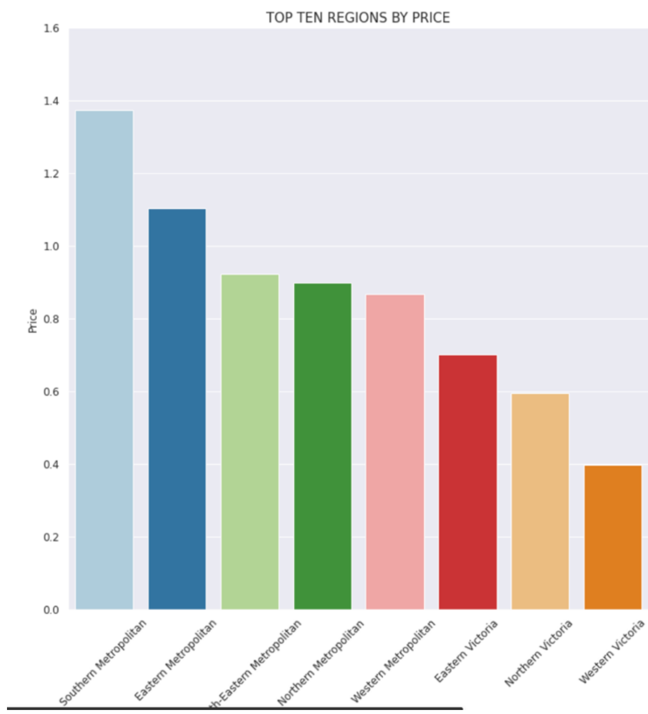
After the data exploration, we want to see the relationship between price and other dependent factors. In this case, say if the number of rooms, larger the size of the land, and shorter distance to Melbourne CBD, even region of each suburbs located. All of this determinants can determine the pricing. Therefore, we create a linear model.

Here are some understanding observing from the influence factors of one attributes to another. In this case, rooms and number of bedroom is highly related amounted to 0.94 close to 1.

```
<matplotlib.axes._subplots.AxesSubplot at 0x7faf08900b70>
```



We also analyse the nominal attributes of Melbourne regions to price. Using pearson correlation's matrix, we will correlated the region as another features to the housing pricing.



Some hypotheses need to be analyse and as discussed to classify which is most accurate to predict the factors to determine the prices of houses in the real estate industry.

Hypothesis 1: For a 2BR housing in Melbourne, it is still affordable to predict and purchase the price of $150,000 or more than that!

Hypothesis 2:
Class 1: MyPerceptron

defines with a function in this class method. For first parameter, initialise the self-points to these class object. It is implicit.

```
- y_ = np.array([1 if i > 0 else 0 for i in y])
```

```
Type: br - bedroom(s); h - house,cottage,villa, semi,terrace; u - unit,
duplex; t - townhouse.
```

Class 2: Accuracy

Function for entropy , a decision tree class for initialising , fitting , predicting , building the tree by defining stopping criteria, selecting the best criteria for splitting, finding information gain, defining the split, traversing and defining most common label. A function for determining accuracy.

From this prediction between classification that can be used to learning algorithm, very widely used, easy to implement,

- Additive updates to weights
- Geometric interpretation
- Mistake bound
- Practical variants abound
- You should be able to implement the Perceptron algorithm

https://www.cs.utah.edu/~zhe/pdf/lec-10-perceptron-upload.pdf
http://users.stat.ufl.edu/~winner/sta6208/notes1.pdf

Analysis on how to improve such aspects of your implementation, e.g. by comparing with widely used machine learning libraries.

== compare the list in the array

multiple regression equation I've mentioned earlier

For each training sample $X_{i:}$

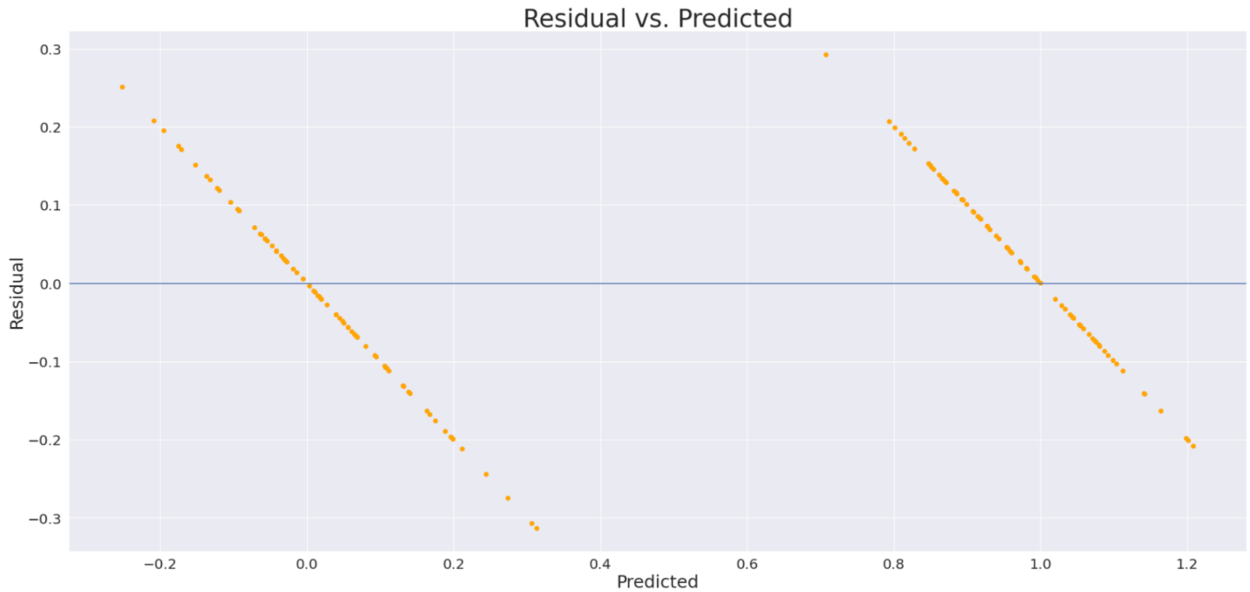
$w: = w + \Delta w$

$\Delta w := \partial - (y_i$

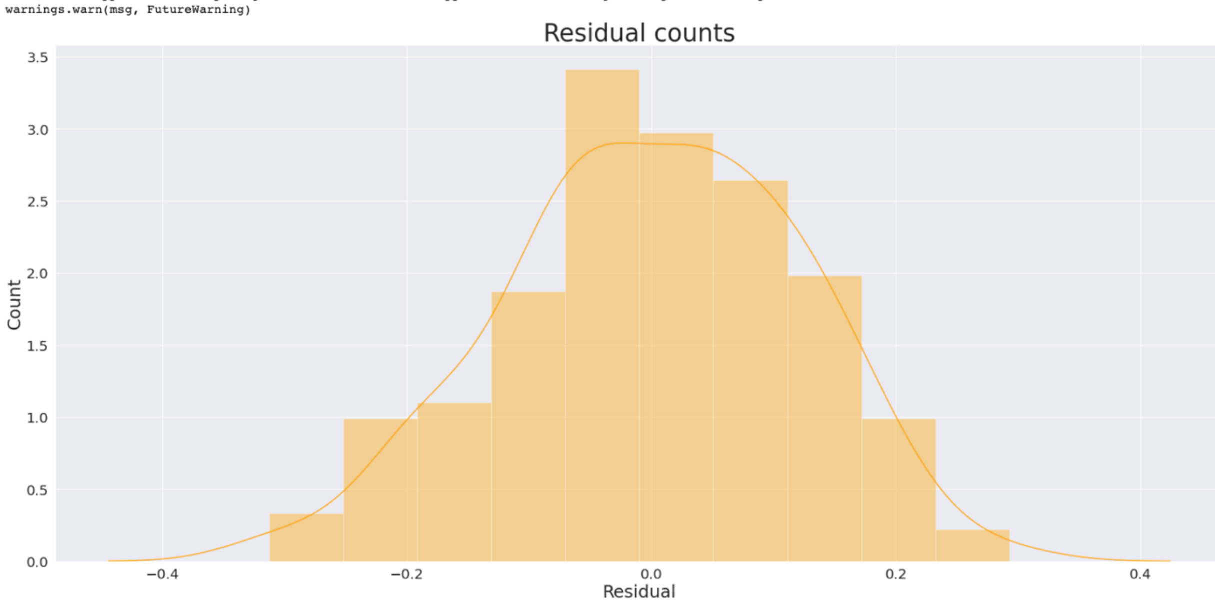
Results

From the correlation to determine the price and impacting the houses' prices. One determinant rank could be as follows on the rooms, bedrooms, and bathrooms, based on the numbers of rooms and on this factors alone, we notice on how the factors like distance from the housing, number of rooms, how large the landsize are, and car spot are the greatest determinants on the housing prices.

Actual (residual) vs predicted with test and train data




```
/usr/local/lib/python3.6/dist-packages/seaborn/distributions.py:2551: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Plea
  warnings.warn(msg, FutureWarning)
```



Correct implementation

Regressor or decision tree could be used to split the

```
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,y_train))
print(classification_report(y_test,y_train))
confusion_matrix(y_test, y_train)
```

Experiment design and evaluation

Machine Learning. Start with evaluate, cross-validation model, evaluate model, and evaluate recommender

Iniatialize the model

Anomaly detection

To explain perceptron classification, a machine learning technique that can be used for predicting if a person is male or female based on numeric predictors such as age, height, weight, and so on. It's mostly useful to provide a baseline result for comparison with more powerful ML techniques such as logistic regression and k-nearest neighbors.

Conclusion

Moving forward, the team will continue with the data cleaning, standardisation and transformation process for the purpose of model analysis. Questions will be raised along the way and assumptions documented. After we have completed this, it would be great to spend some time with your data SME to ensure that all assumptions are aligned with Sprocket Central's understanding. For future coding work, we will try on