Receipt Categorizer – Machine Learning Model

# Problem Statement :

Given an image of a receipt, identify and categorize the following elements

1. Vendor Name
2. Vendor Address
3. Item Name
4. Item Price
5. Total Label
6. Total Price
7. Other

# Machine Learning Model:

We are going to create a supervised machine learning model to help identify and categorize the receipt elements.

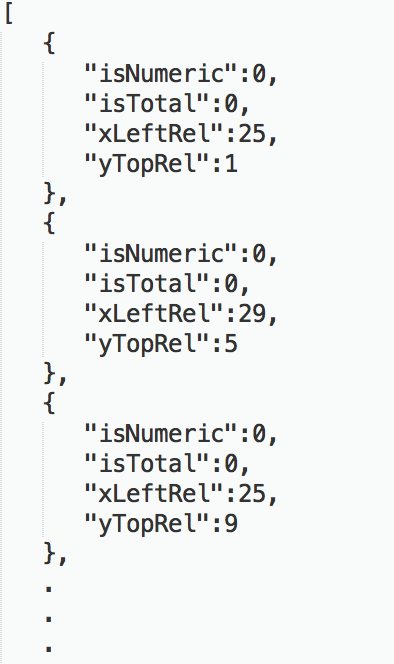
## Input/output of the Model:

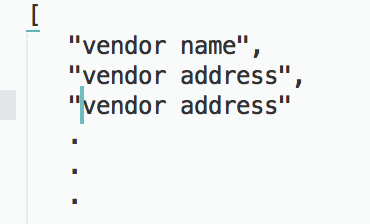
The Android App we created in Step 1, takes the image of a receipt as input and recognizes the text in images.

For each block of text, the above properties are calculated and posted to the service in JSON format.

The Machine Learning Model that we create in this step, will categorize each block of text in the receipt, based on these properties.



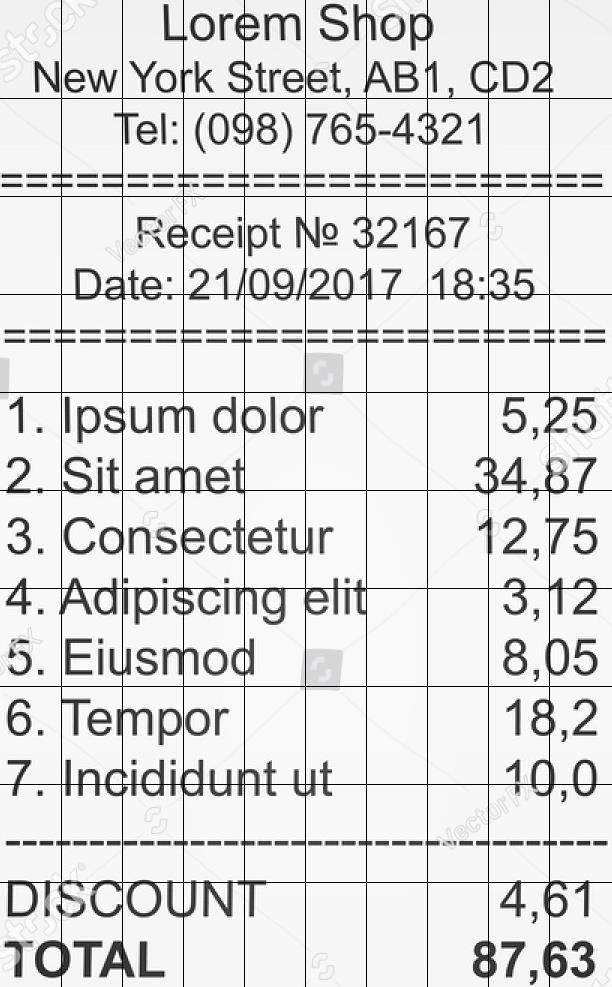




Model

## Feature Engineering:

### Features - A feature is an input variable describing our data, it is the way that we represent our data it to the machine learning system — the x variable in simple linear regression. A simple machine learning project might use a single feature, while a more sophisticated machine learning project could use millions of features

You must create a **representation** of the data to provide the model with a useful vantage point into the data's key qualities. That is, in order to train a model, you must choose the set of features that best represent the data.



Consider the above receipts - which vary in dimensions. If we divide each receipt into a 10 X 10 grid the following patterns can be observed (which will hold true for majority of receipts) :

1. The Vendor Name appears in Row 1
2. The Total appears in Row 10
3. All prices are numeric
4. All the Item Names are aligned with the ‘Total’ label and appear around Column 1.
5. All the Item Prices are aligned with the Total Price and appear around Column 9.

Keeping in mind these observations, the following will be the features in our model

1. xLeftRel – The relative x-axis position of the top-left corner of each block of text in the receipt

xLeftRel = ( 100 \* TopLeft.x ) / Image Width

1. yTopRel – The relative y-axis position of the top-left corner of each block of text in the receipt

yTopRel = ( 100 \* TopRight.y ) / Image Height

1. IsNumeric – Set to true if the block if text is Numeric
2. IsTotal – Set to true if the block of text is the ‘Total’ label in the receipt.

**Do It Yourself:** In real life Data Scientists spend a significant amount of time in Feature Engineering as it is one of the most important aspects of machine learning.

*What other features can you think of to improve the accuracy of the model in the given use case?*

## Important Concepts:

### Labels

A label is the thing we're predicting—the y variable in simple linear regression.

In our case it is the classification of receipt elements

0 – vendor name

1 – vendor address

2 – other

3 – item name

4 – item price

5 – total

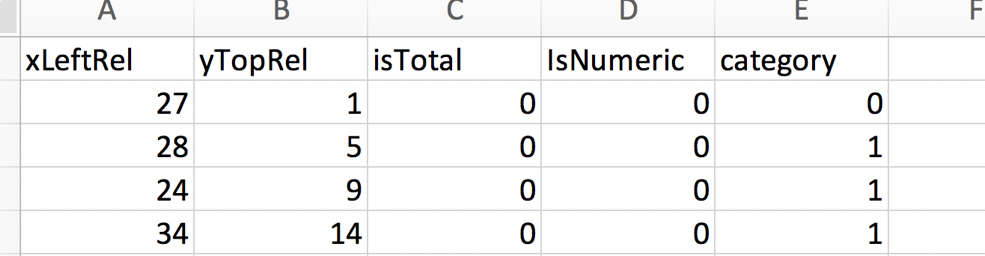
6 – total price

Example – One piece of data

### Labeled example

An example in which the feature and label are both defined. Used to train and test the model.

In our case **refer Test\_Dataset.csv, Training\_Dataset.csv**. The csv contain features for blocks of text from receipts with the categories already defined



### Unlabeled example

An example where only the feature is defined, not the label. Once we've trained our model with labeled examples, we use that model to predict the label on unlabeled examples.

In our case, the data extracted from the Android App for which we need to predict the categories.

### Model

A model defines the relationship between features and label**, it is the thing that is doing the predicting**. It's something that we're going to create in this step.

TensorFlow is a platform that can be used to build machine learning models. For the full list of low-level TensorFlow operations that are possible to use in your code, check out the API page under TensorFlow.org. You'll be interacting the most with the TensorFlow estimators API, that make it really easy to build neural-network models.

## tf.estimator.DNNClassifier

A classifier for TensorFlow DNN models.

hidden\_units: Iterable of number hidden units per layer. All layers are fully connected. Ex. [64, 32] means first layer has 64 nodes and second one has 32.

feature\_columns: An iterable containing all the feature columns used by the model. All items in the set should be instances of classes derived from \_FeatureColumn.

n\_classes: Number of label classes. Defaults to 2, namely binary classification. Must be > 1.

### Train: Trains a model given training data input\_fn.

input\_fn: A function that provides input data for training as minibatches. The function should construct and return [tf.data.Dataset](https://www.tensorflow.org/api_docs/python/tf/data/Dataset)

steps: Number of steps for which to train the model.

### Evaluate Evaluates the model given evaluation data input\_fn.

### Predict Yields predictions for given features.

**Do It Yourself:** Hyperparameters are the knobs that programmers tweak in machine learning algorithms. *Change the steps and batch size in out model and see the effects on accuracy. What are the optimal values?*

## Useful Links :

Tensorflow - <https://www.tensorflow.org/tutorials/>

Estimator - <https://www.tensorflow.org/api_docs/python/tf/estimator>

DNNClassifier - <https://www.tensorflow.org/api_docs/python/tf/estimator/DNNClassifier>

Git Repo -