Receipt Info Extraction Project

Presentor: Zhongbin Huang
Date:

Outline

The Problem

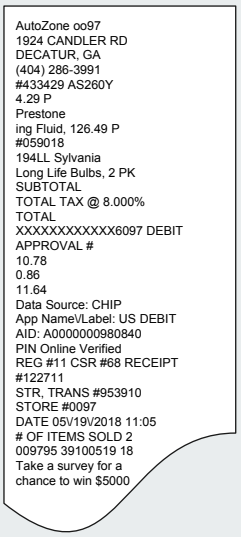
Goal

Minestones

Summary and Next Step

The Problem

Problem statement


AutoZone oo97
1924 CANDLER RD
DECATUR, GA
(404) 286-3991
#433429 AS260Y
4.29 P
Prestone
ing Fluid, 126.49 P
#059018
194LL Sylvania
Long Life Bulbs, 2 PK
SUBTOTAL
TOTAL TAX @ 8.000%
TOTAL
XXXXXXXXXXX6097 DEBIT
APPROVAL #
10.78
0.86
11.64
Data Source: CHIP
App NameVLabel: US DEBIT
AID: A0000000980840
PIN Online Verified
REG #11 CSR #68 RECEIPT
#122711
STR, TRANS #953910
STORE #0097
DATE 05V19V2018 11:05
OF ITEMS SOLD 2
009795 39100519 18
Take a survey for a
chance to win $5000

Extract useful information accurately from scanned receipt OCR records

Once the phone camera scans the receipt using the Apple/Android APP, detail information about the receipt will be returned in JSON format using Google OCR.

These JSON records have to be parsed and processed **SMARTLY** to get useful information.

Current status

Currently, rulebase method to return merchant name only has 29% recall score.

Recall score =

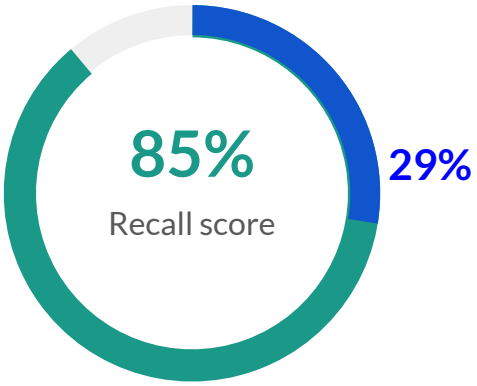
Merchant name correct

/ All return merchant name

Goal

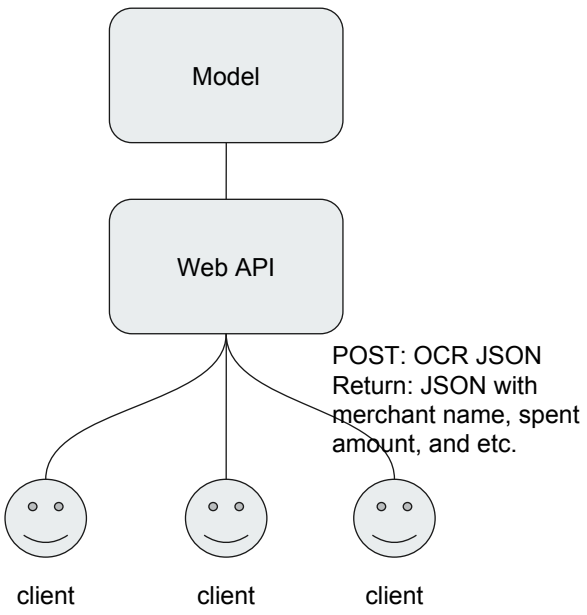
Our goal - Model performance

Increase the recall score of returning correct merchant name from 29% to 85%



Our goal - Model deployment

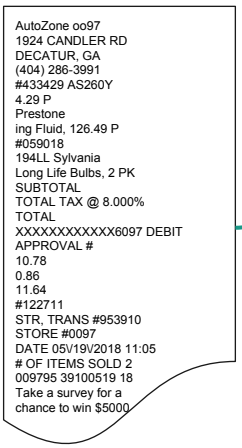
Build web server API that returns the model output in JSON response.



Minestones

Data exploration

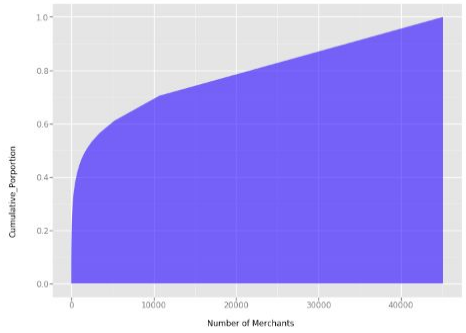
Google OCR




```
{"textAnnotations":[{"locale":"en","boundingPoly":{
"vertices":[{"x":44,"y":122},{"x":761,"y":122},{"x"
:761,"y":1257},{"x":44,"y":1257}]},"description":"C
RABBY JOE'S DONNTOWN\n276 DUNDAS ST\nLONDON, ON
N6B1T6\n5196454880\nSALE\nServer #: 000200\nMID:
5800203\nTID: 006\nBatch #: 417\n04\/29\/18\nAPPR
CODE: 04239\nVISA\nREF#:
00000032\n210141\nChip\n7932\nAMOUNT\nTIP\nTOTAL\n$
27.09\n$4.06\n$31.15\nAPPROVED\n"},{"boundingPoly":
{"vertices":[{"x":153,"y":122},{"x":315,"y":129},{"
x":313,"y":173},{"x":151,"y":166}]},"description":"
CRABBY"},{"boundingPoly":{"vertices":[{"x":345,"y":
133},{"x":445,"y":137},{"x":443,"y":170},{"x":344,"
y":166}]},"description":"JOE'S"},{"boundingPoly":{"
vertices":[{"x":470,"y":142},{"x":673,"y":151},{"x"
:671,"y":190},{"x":468,"y":181}]},"description":
```

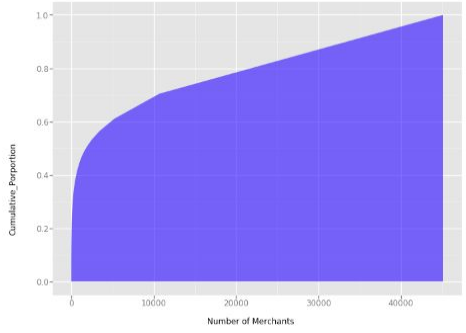

To be processed by our model

Data exploration



- 500 merchants will cover 50% of all the receipts !
- Now what's next?

Data exploration



- Construct a clean frequent merchant name list
- Fuzzy match the receipt words
- 50% accuracy WITHOUT machine learning model already !

Fuzzy match

Word in OCR JSON:

Word in merchant name list:

Mcdonald

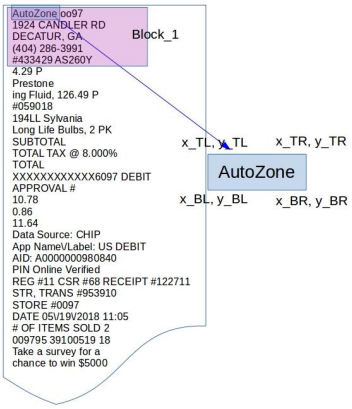
Mcdonald's

Pineapple

Mcdonald's

Some merchant names have two words or more -> Ngram

Feature Engineering



Features Parsed Directly from OCR File

- word_num
- x_TL, y_TL
- x_TR, y_TR
- x_BL, y_BR
- x_BR, y_BL
- block_num
- block_x_TL, block_y_TL

Additional Features Engineered

- Height, width
- Contains_digit
- Contains_alpha
- Has_2decimals

Generate Ngram word

- Why?

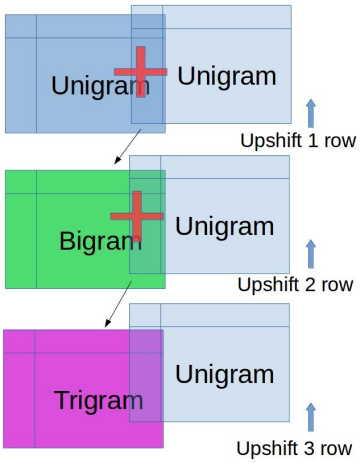
  Some merchant names have two words or more

  Want to keep features
- How?

  Shift Dataframe

  Combine

  Build features



Train machine learning model

- Model: Random Forest, XGBoost, Logistic Regression
- Oversampling: SMOTE
- Other techniques: Gridsearch, K-fold validation, Confusion matrix, AUC, Feature importance

Calculate recall score on receipt level

- Do training on all the receipt data

  BUT

- Only ONE merchant name will be returned for ONE receipt

- Some receipts have merchant name shows up more than one time

  BUT

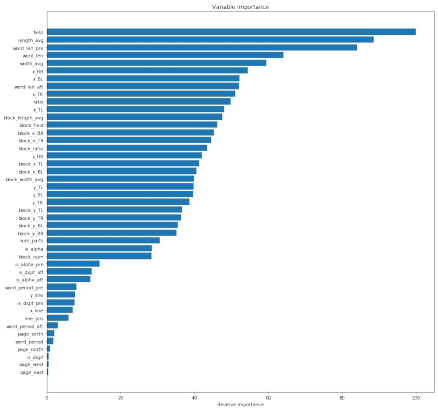
- We only consider the first one

Calculate recall score on receipt level

- Y_predict
- Use ***pred_proba*** to return the prediction probability of each Ngram words
- Group words by receipt ID
- For each receipt, ***y_predict=1*** only for the word that having the max probability

- Y_true
- only label (***y_true=1***) for the first merchant name that having max height.

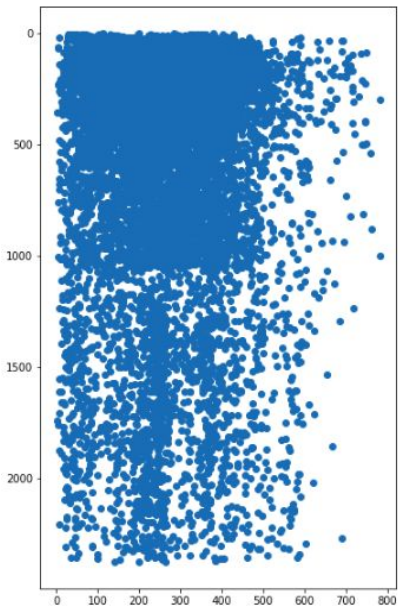
Feature importance



- Block number, coordinate are important features

Merchant location

- most of the merchant names are located on the top left of the receipts



Three Levels of Merchant Name Extraction

Level 1	Level 2	Level 3
Logo Annotation	Fuzzy matching	Machine learning model

Three Levels of Merchant Name Extraction

Level 1
Logo Annotation

- Google OCR will catch the merchant logo.
- Approximately 20% of the OCR outputs have this information
- Use this merchant name directly if it exists

Three Levels of Merchant Name Extraction

Level 2

Fuzzy matching

- We can assume 99% of the time the matching word is the correct merchant name
- About 50% receipts can be matched
- Overlap with the ones having logo

Three Levels of Merchant Name Extraction

Level 3

Machine learning model

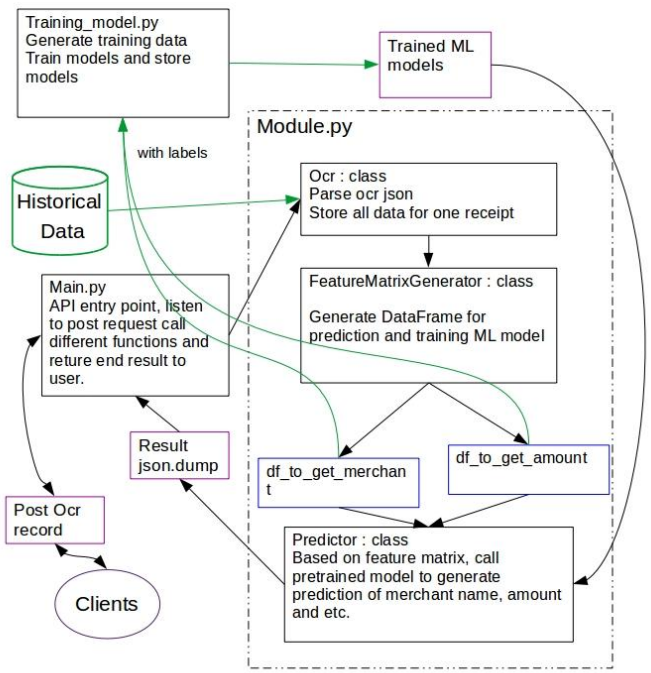
- the ngram word that having the highest probability being merchant name within a receipt will be returned
- From our model evaluation and tuning process, the best recall score is 0.6

Three Levels of Merchant Name Extraction

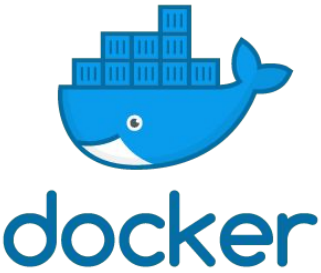
Level 1	Level 2		Level 3
Logo Annotation	Fuzzy matching	Overlap	Machine learning model

Accuracy: 20% + 50% - 15% + 0.6* 45% = 82%

API



Deployment

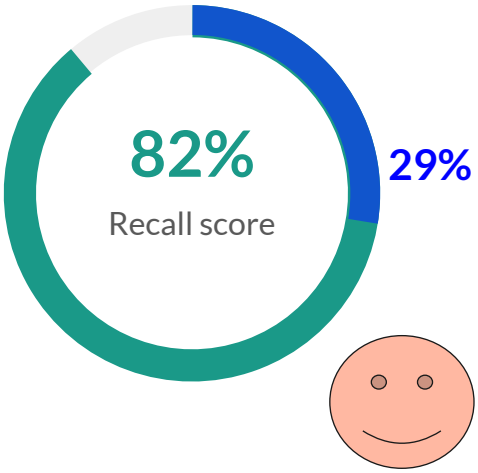



```
FROM tiangolo/uwsgi-nginx-flask:python3.6
EXPOSE 80
COPY ./app /app
RUN pip install -r requirements.txt
```

Summary and next step

Achievement

Model accuracy



Achievement

Model deployment

Ready to

Deploy!

What's Next?

1. Wrap the google OCR API in the server side.
2. Balance server load.
3. Optimize cost.
4. Schedule to retrain machine learning model.
5. Update frequent merchant list automatically.

Questions?