

Cell2Doc: ML Pipeline for Generating Documentation in Computational Notebooks

Tamal Mondal
IIT Hyderabad
cs21mtech12001@iith.ac.in

Scott Barnett
Deakin University
scott.barnett@deakin.edu.au

Akash Lal
Microsoft Research
akashl@microsoft.com

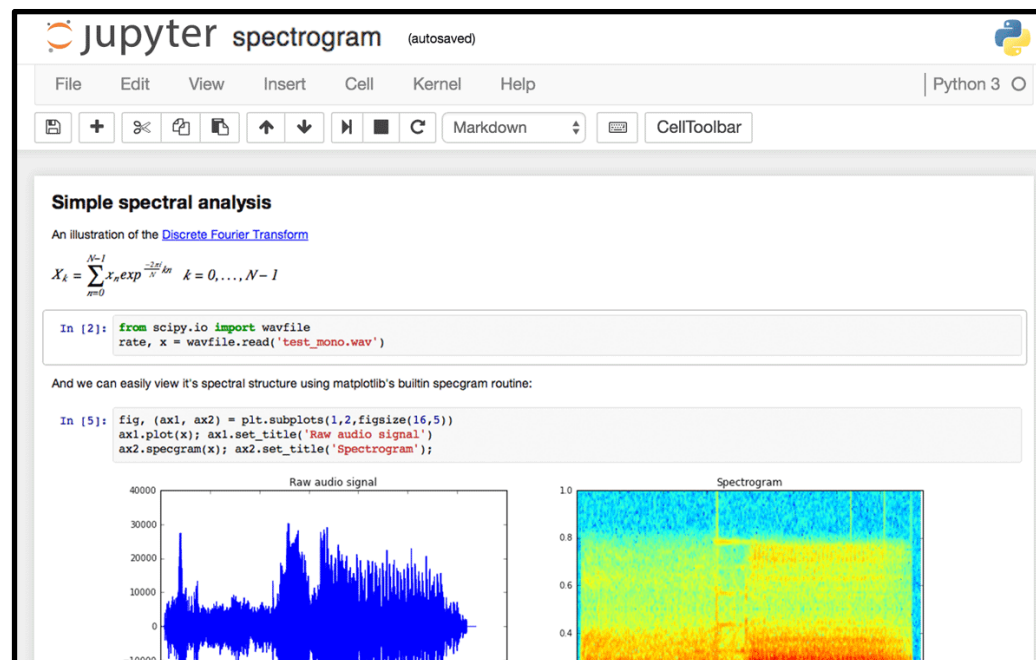
Jyothi Vedurada
IIT Hyderabad
jyothiv@cse.iith.ac.in

Proceedings of the 38th IEEE/ACM International Conference
on Automated Software Engineering (ASE2023)

発表者：井町 瑠貴

Computational Notebooks

ソースコード，ドキュメント，実行結果などを1つのファイルにまとめることができる



しかし，ドキュメントが不十分である場合が多い

- README等，他の情報源にアクセスする必要あり [36]
- ドキュメント自動生成技術で解決を目指す

[36] A. Rule, A. Tabard, and J. D. Hollan, “Exploration and explanation in computational notebooks,” in Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, ser. CHI ’18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1–12. [Online]. Available: <https://doi.org/10.1145/3173574.3173606>

既存研究 | 事前学習済みモデル

コードを正確に説明することができない

- 長いスニペットには適していない
- アーキテクチャの構造的制限により情報ロスが発生

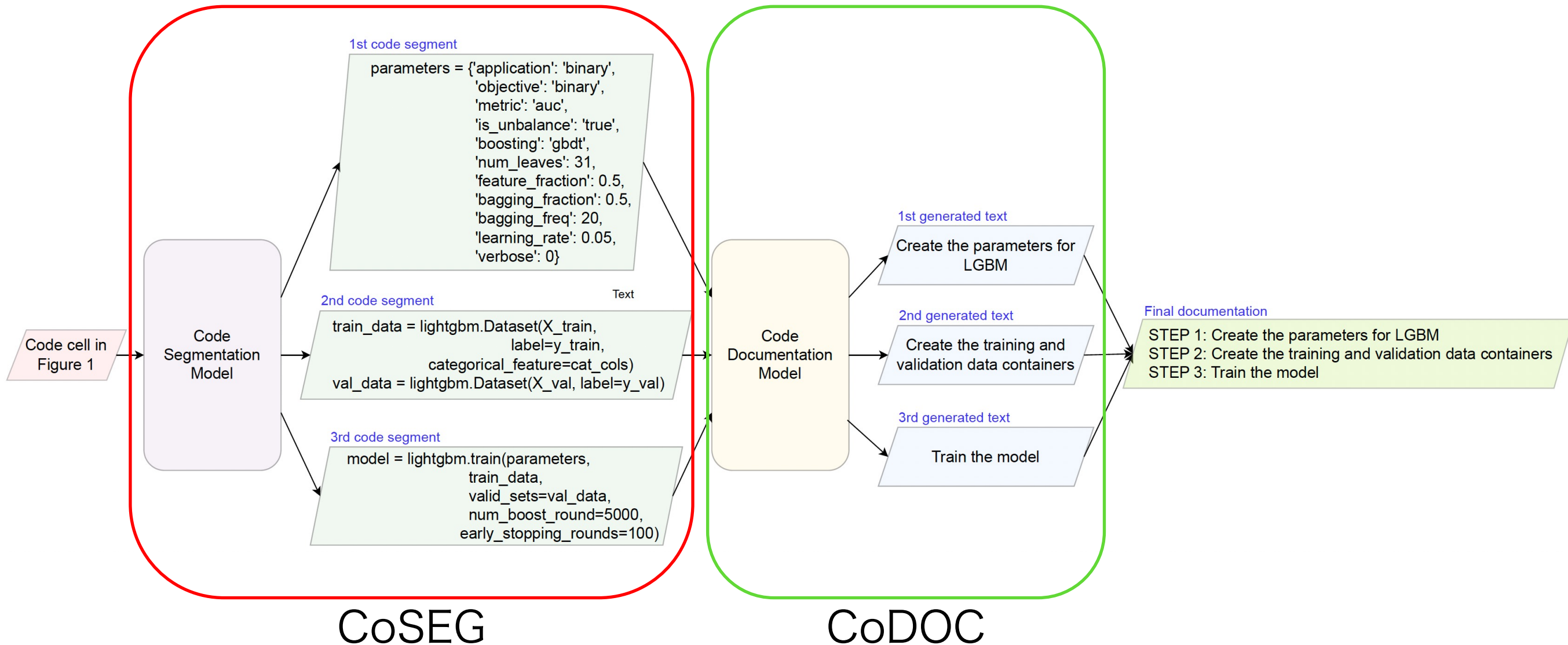
```
parameters = {  
    'application': 'binary', 'objective': 'binary',  
    'metric': 'auc', 'is_unbalance': 'true',  
    'boosting': 'gbdt', 'num_leaves': 31,  
    'feature_fraction': 0.5, 'bagging_fraction': 0.5,  
    'bagging_freq': 20, 'learning_rate': 0.05,  
    'verbose': 0  
}  
train_data = lightgbm.Dataset(X_train,  
    label=y_train, categorical_feature=cat_cols)  
val_data = lightgbm.Dataset(X_val, label=y_val)  
model = lightgbm.train(parameters,  
    train_data, valid_sets=val_data,  
    num_boost_round=5000, early_stopping_rounds=100)
```



Model	Generated Documentation
CodeT5	Now we will train the lightgbm model
PLBART	Train the model
CodeBERT	Light GBM Model
GraphCodeBERT	Train the model
UniXcoder	Train the model

Cell2Doc | 概要

コードセルをセグメントに分離し，それらを独立して文書化



データセット | 収集

Kaggleからノートブックを収集

- クラウドソーシング型のデータ分析プラットフォーム
- Github上のノートブックの品質は低い[36]
- ノートブックに”up vote”することができる
 - 多いものは十分に文書化されていると考えられる[41]

	Comprehensive data exploration with Python Updated 2y ago 1942 comments · House Prices - Advanced Regression Techniques	▲ 13837 Gold ...
	Titanic Tutorial Updated 1y ago 27599 comments · Titanic - Machine Learning from Disaster	▲ 11857 Gold ...
	Titanic Data Science Solutions Updated 5y ago 2492 comments · Titanic - Machine Learning from Disaster	▲ 10217 Gold ...
	Stacked Regressions : Top 4% on LeaderBoard Updated 5y ago	▲ 7041

[41] A. Y. Wang, D. Wang, J. Drozdal, X. Liu, S. Park, S. Oney, and C. Brooks, “What makes a well-documented notebook? a case study of data scientists’ documentation practices in kaggle,” in Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems, ser. CHI EA ’21. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: <https://doi.org/10.1145/3411763.3451617>

データセット | 収集

| KGTorrent^[35]からノートブックを収集

- 2020年10月27日時点
- 約250kのノートブック

| 以下の条件でフィルタリング

- 10以上のup voteを得ている
- チュートリアルやブログ用ではない

[35] L. Quaranta, F. Calefato, and F. Lanubile, “KGTorrent: A dataset of python jupyter notebooks from kaggle,” in 2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR). IEEE, may 2021. [Online]. Available: <https://doi.org/10.1109%2Fmsr52588.2021.00072>

データセット | 収集

フィルタリングの結果, 5428のノートブックが得られた

- 15kのコード・ドキュメントペア
- 21kのコード・コメントペア

Property	Max	Avg	Min	75%
Code tokens for code-markdown pairs	99	53	2	72
Documentation tokens for code-markdown pairs	1131	28	3	32
Code tokens for code-comment pairs	99	52	3	69
Documentation tokens for code-comment pairs	810	10	3	11

データセット | 前処理

Markdownセルは真のドキュメントとして使用することはできない

- コードの要約以外の情報も含まれる

NLPベースのテキスト要約を用いて、ノイズの少ないドキュメントを得る

- 抽出的要約：Spacy
- 抽象的要約：BERT

データセット | 前処理

前処理の結果, 以下のようなデータを得た

Property	Training	Validation	Test
Number of notebooks	3073	357	377
Code-Documentation pairs	14802	1821	1755
Min. # tokens in Documentation	3	3	3
Avg. # tokens in Documentation	6	6	3
Max. # tokens in Documentation	13	13	13
Min. # tokens in Code	3	5	3
Avg. # tokens in Code	52	51	52
Max. # tokens in Code	99	99	99

アプローチ | CoSEG

コードセル内の異なるタスクを識別する
(code segmentation) モデルを構築

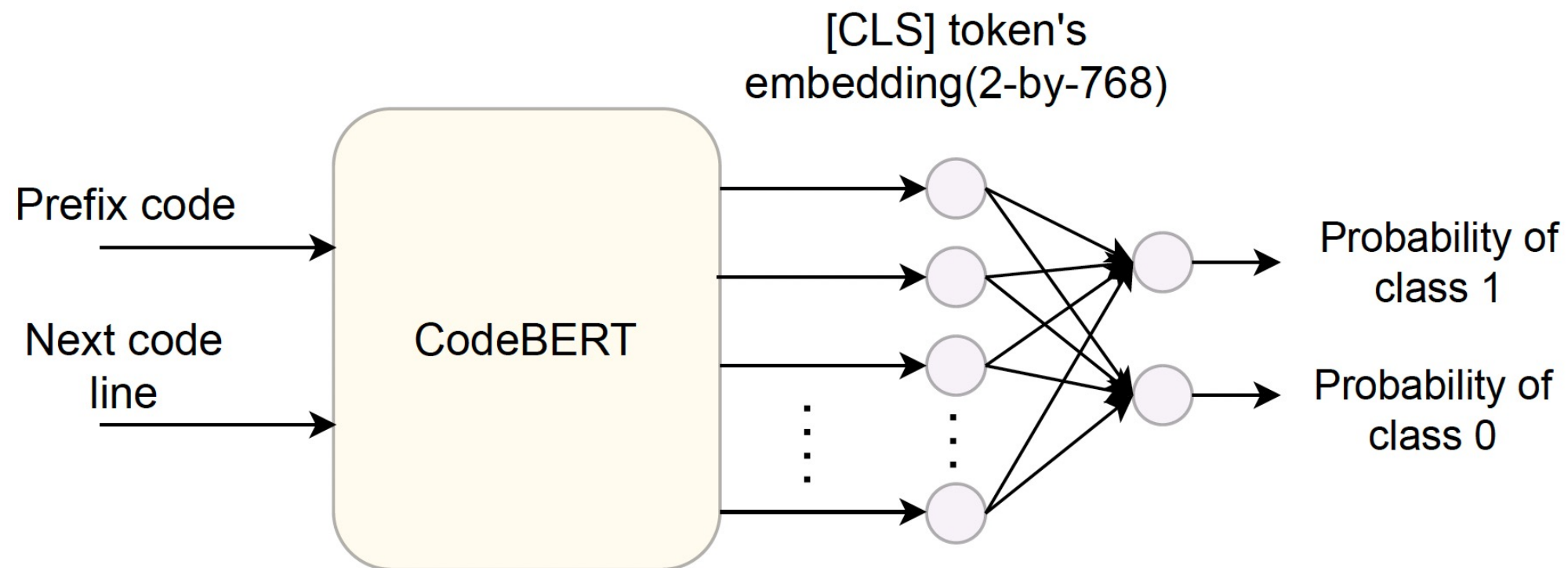
- コードコメントの位置
- コードセルに存在するAST

事前学習されたCodeBERTモデルをファイン
チューニングして、対象行でタスクが変化してい
るかどうかを接頭辞コードから判断 (2値分類)

- 入力
 - [CLS] $pc1, pc2, pc3, \dots$ [EOS] [CLS] $c1, c2, c3, \dots$ [EOS]

アプローチ | CoSEG

対象行で新しいタスクが始まれば1,
そうでなければ0を出力する



アプローチ | CoSEG

各行で CoSEGに問い合わせし，対象行が以前までと同じタスクか，新しいタスクに属するかを判断

Input: C: Code as $[c_1, c_2, \dots, c_n]$, where c_i is a code line; M: Code segmentation model

Output: S: Segmented code as $[s_1, s_2, \dots, s_m]$, where s_j is a code segment

```
1 Procedure segmentation( $C, M$ )
2    $S \leftarrow [[c_1]]$ ;
3   for  $i \leftarrow 2$  to  $n$  do
4      $prediction \leftarrow M(S.last, c_i)$ ;
5     if  $prediction = 1$  then
6        $S = S \oplus [[c_i]]$ ;           // Start a new segment
7     else
8        $S.last = S.last \oplus [c_i]$ ; // Add to prev. segment
```

アプローチ | CoSEG用のデータセット

コードセグメンテーション用の既存のラベルはないため、訓練用に作成

- 目的：異なるタスクを分割するコード行の識別
- 以下の2つの基準を用いる
 - コードコメントの位置
 - コメントの前後で異なるタスクが行われている
 - ASTを用いたコード構造
 - def, for, while, if, class, tryなどのAST構造要素

アプローチ | CoSEG用のデータセット

各データポイントは以下のような構成

- (プレフィックスコード, 対象行, ラベル)
 - プレフィックスコード: 前のセグメントのコード

Split	Notebooks	Total data points	Class 1	Class 0
Training set	4309	275106	116511	158595
Validation set	539	34319	14428	19891
Test set	536	31032	12880	18152

| 以下の5つの事前学習済みモデルを利用

- CodeT5
- PLBART
- CodeBERT
- GraphCodeBERT
- UniXcoder

アプローチ | CoDocの入力表現

コードとドキュメントをペアとして入力

- 以下の4種類の表現を検討
 - Code Markdown
 - コード：そのまま, ドキュメント：セル全体
 - Code Summarized Markdown
 - コード：そのまま, ドキュメント：要約
 - English Code tokens Summarized Markdown
 - コード：文字を含むトークンを除外
 - ドキュメント：要約
 - Split Code Summarized Comment and Markdown
 - コード：そのまま, ドキュメント：要約
 - コメント：そのまま

RQ | 提案手法の評価

RQ1 : CoSEGはコードセルの分割にどの程度有効か

RQ2 : 人間はCell2Docの出力をどう評価するか

RQ3 : CoDocはさまざまな入力表現に対して、
どの程度有効か

RQ4 : Cell2Docは異なる事前学習モデルに対して、
どのような結果が得られるか

RQ5 : 生成モデルをCoDocとして利用できるか

まとめ

Computational Notebooksにおける自動ドキュメント化を目的としたCell2Docを提案

- 同一コードセル内のコードをタスクごとに分割することで高い性能を発揮

所感

ノートブックにおけるドキュメントという着眼点が自分にはなかった観点だったので興味深かった

- 手軽さゆえに、一般的なドキュメントよりも運用は難しそう

一般的なドキュメントよりも生成時間に重要ではないか？

- よりリアルタイム性が必要なイメージがある

複数セルだとどうなる？

- 記述の順番にルールはないため、難しい？
- 実行時間でソートすればもっと大きな塊でできる？

補足

補足 | ドキュメント要約の例

Ex.	Code Snippet	Original Documentation	Spacy Summarization	BART Summarization
1	<pre>def objective(values): params = { 'max_depth': values[0], 'num_leaves': values[1], 'min_child_samples': values[2], 'scale_pos_weight': values[3], 'subsample': values[4], 'colsample_bytree': values[5], 'metric': 'auc', 'nthread': 8, 'boosting_type': 'gbdt', 'objective': 'binary', 'learning_rate': 0.15, } ...</pre>	Below is the fun part. The function gp_minimize requires an objective function and what the function all needs is basically a metric we want to minimize. Of course, we can just use whatever training setup we have been using but just tweak it to return a AUC to minimize..(negative AUC)	The function gp_minimize requires an objective function and what the function all needs is basically a metric we want to minimize.	The function gp_minimize requires an objective function. What the function all needs is basically a metric we want to minimize.
2	<pre>target = train['trip_duration'] print("Longest trip duration {} or {} minutes: ".format(np.max(target.values), np.max(target.values)//60)) print("Smallest trip duration {} or {} minutes: ".format(np.min(target.values), np.min(target.values)//60)) print("Average trip duration : {} or {} minutes ".format(np.mean(target.values), np.mean(target.values)//60))</pre>	No overlap in train and test ids. Hence we can drop the id column for now until unless we figure out that there actually was some overlap(by some magic of course!)	Hence we can drop the id column for now until unless we figure out that there actually was some overlap(by some magic of course!)	No overlap in train and test ids. Hence we can drop the id column for now until we figure out that there actually was some overlap.
3	<pre>from matplotlib import pyplot from keras.preprocessing.image import ImageDataGenerator from keras.models import Sequential from keras.layers import Conv2D, MaxPooling2D, Dense, Dropout, Input, Flatten, Activation ...</pre>	Get back to building a CNN using Keras. Much better frameworks then others. You will enjoy for sure.	Get back to building a CNN using Keras.	Get back to building a CNN using Keras. Much better frameworks then others.

補足 | 入力表現の例

Code	Markdown	Summarized Markdown
<pre> 1 TRAINING_SIZE = 300000 2 TEST_SIZE = 50000 3 # Load data 4 train = pd.read_csv('../input/train.csv', 5 skiprows=range(1,184903891- 6 TRAINING_SIZE-TEST_SIZE), 7 nrows=TRAINING_SIZE, 8 parse_dates=['click_time']) 9 val = pd.read_csv('../input/train.csv', 10 skiprows=range(1,184903891-TEST_SIZE), 11 nrows=TEST_SIZE, 12 parse_dates=['click_time']) 13 # Split into X and y 14 y_train = train['is_attributed'] 15 y_val = val['is_attributed'] </pre>	Instead of doing cross-validation, in this notebook I'm going to use a simple fixed training and testing set. Just for illustration purpose.	Instead of doing cross-validation, in this notebook I'm going to use a simple fixed training and testing set.

English Code Tokens
<pre> ['TRAINING_SIZE', 'TEST_SIZE', 'train', 'pd', 'read_csv', '../input/train.csv', 'skiprows', 'range', 'TRAINING_SIZE', 'TEST_SIZE', 'nrows', 'TRAINING_SIZE', 'parse_dates', 'click_time', 'val', 'pd', 'read_csv', '../input/train.csv', 'skiprows', 'range', 'TEST_SIZE', 'nrows', 'TEST_SIZE', 'parse_dates', 'click_time', 'y_train', 'train', 'is_attributed', 'y_val', 'val', 'is_attributed'] </pre>