PyData

Bratislava

GAP DATA INSTITUTE

No free hunch: Experience from Kaggle competition

(PyData Bratislava Meetup #4, Nervosa)

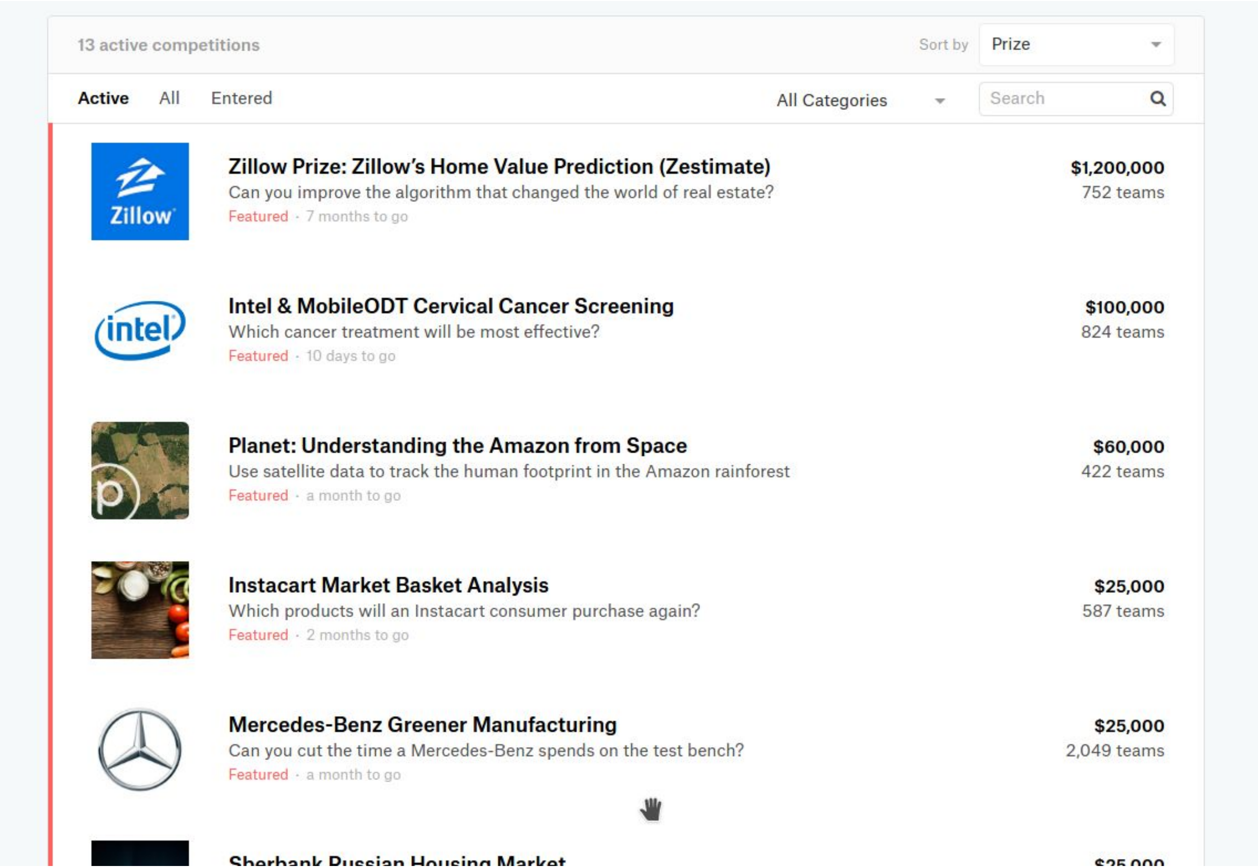
12. 6. 2017

Michal Šustr, #PyDataBA

No free Hunch: Insights from Kaggle competitions

Michal Šustr: http://lectures.ai

Site for ML competitions



Some info about Kaggle

- Founded in 2010
- Acquired by Google in March 2017 (joining Google Cloud)
- Over 536,000 registered users (May 2016)
- Very active community:
  4,000 forum posts per month,
  over 3,500 competition submissions per day

How it works

1. Host prepares data and description of the problem

2. Participants experiment with models and compete against each other. Submissions are scored immediately and summarized on a live leaderboard.

3. After the deadline passes, the competition host pays the prize money in exchange for the algorithm / software

Private vs public leaderboards

Quora · 3,394 teams · 5 days ago

Overview Data Kernels Discussion **Leaderboard** More | Submit Predictions

Public Leaderboard | Private Leaderboard

This leaderboard is calculated with approximately 35% of the test data.
The final results will be based on the other 65%, so the final standings may be different.

Raw Data Refresh

In the money | Gold | Silver | Bronze

#	△priv	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	DL guys			0.11277	263	5d
2	—	Depp Learning			0.11367	196	5d
3	—	Jared Turkewitz & sjv			0.11446	178	5d
4	—	YesOfCourse			0.11450	189	5d
5	—	Qingchen | KazAnova | Faron			0.11482	219	5d
6	—	LAMAA power			0.11532	406	5d
7	▾2	Unduplicated Duplicates		+4	0.11881	314	5d
8	—	NLPFakers		+3	0.11885	250	5d

Quora · 3,394 teams · 5 days ago

Overview Data Kernels Discussion **Leaderboard** More | Submit Predictions

Public Leaderboard | **Private Leaderboard**

The private leaderboard is calculated with approximately 65% of the test data.

Refresh

In the money | Gold | Silver | Bronze

#	△pub	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	DL guys			0.11580	263	5d
2	—	Depp Learning			0.11670	196	5d
3	—	Jared Turkewitz & sjv			0.11756	178	5d
4	—	YesOfCourse			0.11768	189	5d
5	—	Qingchen | KazAnova | Faron			0.11851	219	5d
6	—	LAMAA power			0.11887	406	5d
7	▴2	aphex34			0.12072	166	5d
8	—	NLPFakers		+3	0.12239	250	5d

Competition: Quora questions

Training set

```
df_train = pd.read_csv('../input/train.csv')
df_train.head()
```

	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0

Competition: Quora questions

Test Set

```
df_test = pd.read_csv('../input/test.csv')
df_test.head()
```

	test_id	question1	question2
0	0	How does the Surface Pro himself 4 compare wit...	Why did Microsoft choose core m3 and not core ...
1	1	Should I have a hair transplant at age 24? How...	How much cost does hair transplant require?
2	2	What but is the best way to send money from Ch...	What you send money to China?
3	3	Which food not emulsifiers?	What foods fibre?
4	4	How "aberystwyth" start reading?	How their can I start reading?

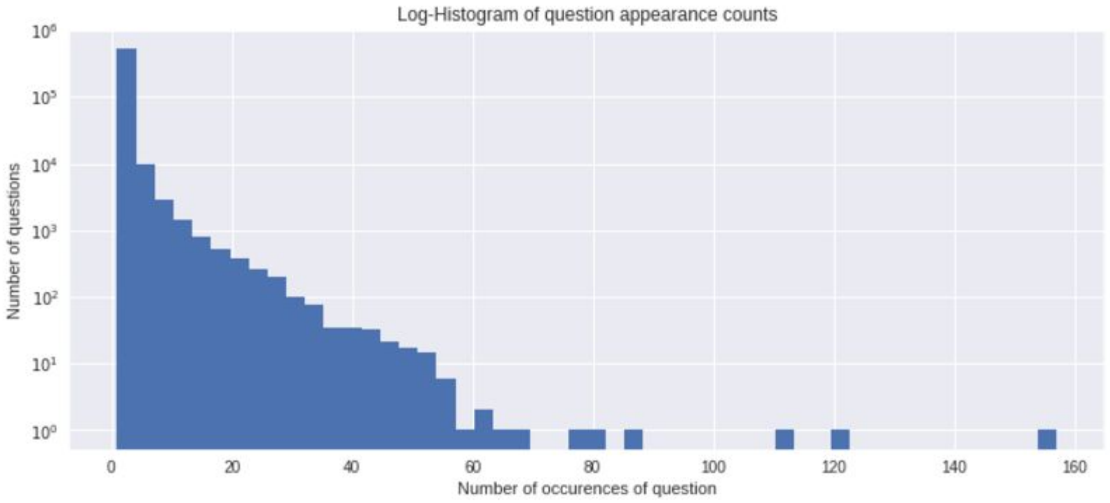
Score function:

$$logloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{i,j}\log(p_{i,j})$$

Typical first steps

Histogram of question counts

```
qids = pd.Series(df_train['qid1'].tolist() + df_train['qid2'].tolist())
plt.figure(figsize=(12, 5))
plt.hist(qids.value_counts(), bins=50)
```



Typical first steps

Character count of questions

Word count in questions

Semantic analysis - question marks, capital letters, full stops, numbers

Difference between train / test set (public leaderboard) - matters a lot!

Typical first steps

Test/train class imbalance - big difference for logloss (different “prior”)

Score of 0.554 with a constant prediction at the training set mean of 0.369 (mean of labels on test set) => r ~ 0.174

$$r = \frac{logloss + log(1-p)}{log\left(\frac{1-p}{p}\right)}$$

Test has < ½ of positive labels than train!

Popular model - XGBoost

- optimized distributed gradient boosting library
- parallel tree boosting (also known as GBDT, GBM)
- runs on major distributed environment (Hadoop, SGE, MPI)
- Interfaces:
  - Command Line Interface (CLI).
  - C++ (the language in which the library is written).
  - Python interface as well as a model in scikit-learn.
  - R interface as well as a model in the caret package.
  - Julia.
  - Java and JVM languages like Scala and platforms like Hadoop.

A lot of params - https://github.com/dmlc/xgboost/blob/master/doc/parameter.md

- general parameters, booster parameters and task parameters

Feature engineering (tf-idf, char/word stats), resample, then xgboost - LB score 0.158

```python
def build_features(data, stops, weights):
    X = pd.DataFrame()
    f = functools.partial(word_match_share, stops=stops)
    X['word_match'] = data.apply(f, axis=1, raw=True) #1

    f = functools.partial(tfidf_word_match_share, weights=weights)
    X['tfidf_wm'] = data.apply(f, axis=1, raw=True) #2

    f = functools.partial(tfidf_word_match_share_stops, stops=stops, weights=weights)
    X['tfidf_wm_stops'] = data.apply(f, axis=1, raw=True) #3

    X['jaccard'] = data.apply(jaccard, axis=1, raw=True) #4
    X['wc_diff'] = data.apply(wc_diff, axis=1, raw=True) #5
    X['wc_ratio'] = data.apply(wc_ratio, axis=1, raw=True) #6
    X['wc_diff_unique'] = data.apply(wc_diff_unique, axis=1, raw=True) #7
    X['wc_ratio_unique'] = data.apply(wc_ratio_unique, axis=1, raw=True) #8

    f = functools.partial(wc_diff_unique_stop, stops=stops)
    X['wc_diff_unq_stop'] = data.apply(f, axis=1, raw=True) #9
    f = functools.partial(wc_ratio_unique_stop, stops=stops)
    X['wc_ratio_unique_stop'] = data.apply(f, axis=1, raw=True) #10

    X['same_start'] = data.apply(same_start_word, axis=1, raw=True) #11
    X['char_diff'] = data.apply(char_diff, axis=1, raw=True) #12

    f = functools.partial(char_diff_unique_stop, stops=stops)
    X['char_diff_unq_stop'] = data.apply(f, axis=1, raw=True) #13

#     X['common_words'] = data.apply(common_words, axis=1, raw=True)  #14
    X['total_unique_words'] = data.apply(total_unique_words, axis=1, raw=True)  #15

    f = functools.partial(total_unq_words_stop, stops=stops)
    X['total_unq_words_stop'] = data.apply(f, axis=1, raw=True)  #16

    X['char_ratio'] = data.apply(char_ratio, axis=1, raw=True) #17

    return X
```

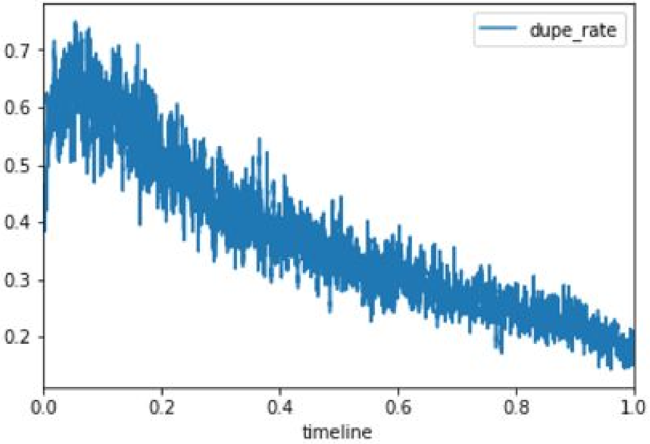
https://www.kaggle.com/act444/lb-0-158-xgb-handcrafted-leaky

“Magic” features (0.03 gain)

more frequent questions are more likely to be duplicates

https://www.kaggle.com/jturkewitz/magic-features-0-03-gain

(funny comments - “This feature is really powerful, maybe you haven't used it in the right way”



The above pattern, and the ~16.5% LB response rate reported by others, imply that the Public LB (and possibly Private LB) are potentially sourced from more recent data than the training set.

1st place - large models

Embedding features - Word embeddings (Word2Vec), Sentence embeddings (Doc2Vec, Sent2Vec), Encoded question pair using dense layer from ESIM model trained on SNLI

Classical text mining features …

Structural features (graph of questions)

Models - Siamese and Attention Neural Networks

Rescaling - local subsamples of the data

Stacking -

Layer 1 : Around 300 models, Layer 2 : Around 150 models Layer 3 : 2 Linear models Layer 4 : Blend

https://www.kaggle.com/c/quora-question-pairs/discussion/34355

Ensembles (bagging, boosting, stacking)

1. **Bagging** (stands for **B**ootstrap **Agg**regation) is the way decrease the variance of your prediction by generating additional data for training from your original dataset using combinations with repetitions to produce multisets of the same cardinality/size as your original data. By increasing the size of your training set you can't improve the model predictive force, but just decrease the variance, narrowly tuning the prediction to expected outcome.
2. **Boosting** is a two-step approach, where one first uses subsets of the original data to produce a series of averagely performing models and then "boosts" their performance by combining them together using a particular cost function (=majority vote). Unlike bagging, in the classical boosting the subset creation is not random and depends upon the performance of the previous models: every new subsets contains the elements that were (likely to be) misclassified by previous models.
3. **Stacking** is a similar to boosting: you also apply several models to your original data. The difference here is, however, that you don't have just an empirical formula for your weight function, rather you introduce a meta-level and use another model/approach to estimate the input together with outputs of every model to estimate the weights or, in other words, to determine what models perform well and what badly given these input data.

https://stats.stackexchange.com/questions/18891/bagging-boosting-and-stacking-in-machine-learning

Ensembles (bagging, boosting, stacking)

	Bagging	Boosting	Stacking
Partitioning of the data into subsets	Random	Giving mis-classified samples higher preference	Various
Goal to achieve	Minimize variance	Increase predictive force	Both
Methods where this is used	Random subspace	Gradient descent	Blending
Function to combine single models	(Weighted) average	Weighted majority vote	Logistic regression

Kaggle Past Solutions

Sortable and searchable compilation of solutions to past Kaggle competitions - for inspiration:

http://ndres.me/kaggle-past-solutions/

Lung cancer detection

Remember public vs private leaderboard? “In CV we trust”

In the money | Gold | Silver | Bronze

#	△priv	Team Name	Kernel	Team Members	Score	Entries	Last
1	▾327	Neon			0.00000	9	2mo
2	▾339	Ikona-Diagnosys		+16	0.29019	3	2mo
3	▾349	生物医学工程		+8	0.32006	7	2mo
4	▾255	Bulbazaurs			0.32872	6	2mo
5	▾250	Chris Halfacre			0.35861	9	2mo
6	▾235	Zhen Li			0.36109	12	2mo
7	▾29	YT			0.36610	10	2mo
8	▾153	linbo_casia			0.37106	14	2mo
9	▾266	RS_group			0.37509	6	2mo
10	▾45	E.I.			0.37687	12	2mo

In the money | Gold | Silver | Bronze

#	△pub	Team Name	Kernel	Team Members	Score	Entries	Last
1	▴136	grt123			0.39975	2	2mo
2	▴87	Julian de Wit & Daniel Hammack			0.40117	2	2mo
3	▴23	Aidence			0.40127	2	2mo
4	▴157	qfpxfd		+5	0.40183	3	2mo
5	▴349	Pierre Fillard (Therapixel)			0.40410	8	2mo
6	▴206	MDai			0.41630	2	2mo
7	▴96	DL Munich			0.42752	2	2mo
8	▴78	Alex |Andre |Gilberto |Shize			0.43019	5	2mo
9	▴343	Deep Breath		+7	0.43872	2	2mo
10	▴262	Owkin Team			0.44068	9	2mo

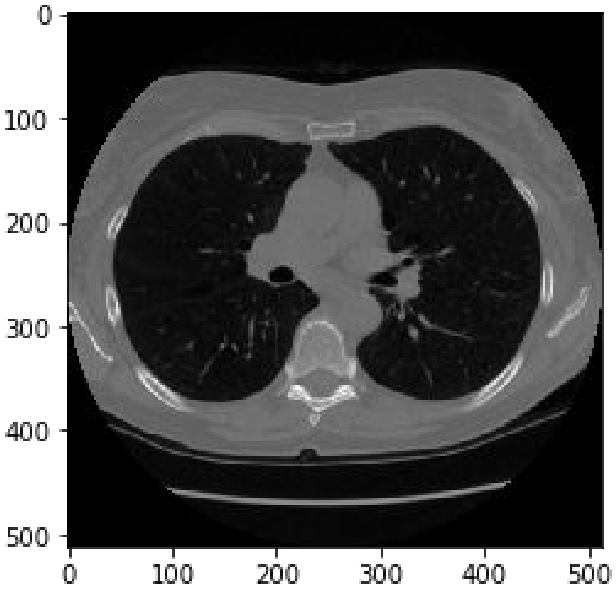
Heavily imbalanced - 5000:500000

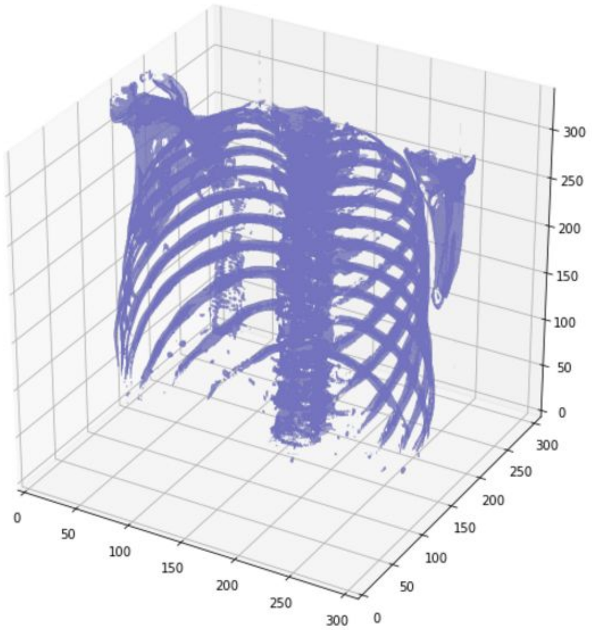
Lung cancer detection

Three most voted kernels (tutorials):

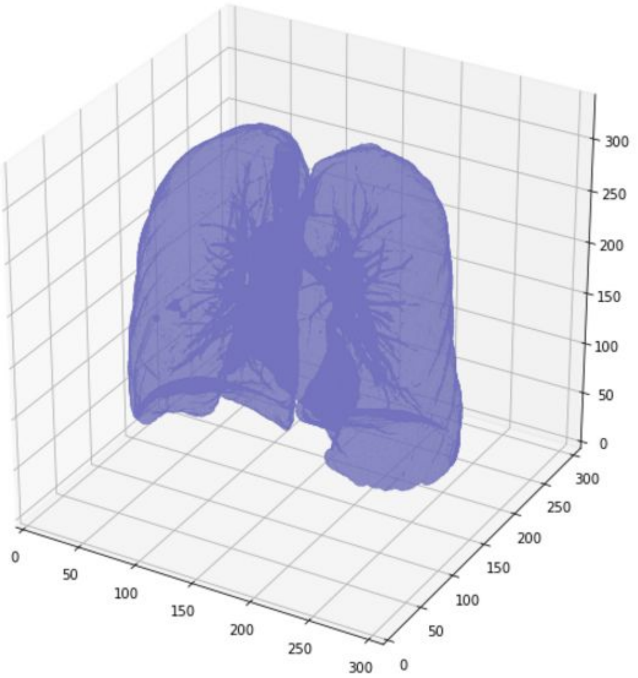
- https://www.kaggle.com/gzuidhof/full-preprocessing-tutorial
- https://www.kaggle.com/sentdex/first-pass-through-data-w-3d-convnet
- https://www.kaggle.com/arnavkj95/candidate-generation-and-luna16-preprocessing

Lung cancer detection

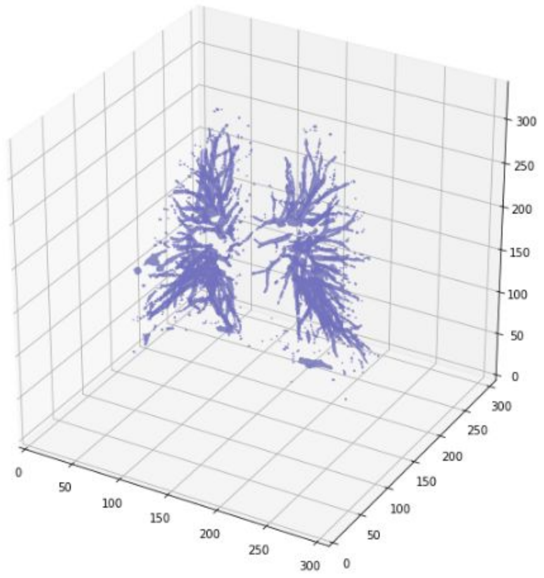




Lung segmentation




```
plot_3d(segmented_lungs_fill - segmented_lungs, 0)
```



2nd place solution

3 Our Approach

There are 4 major steps in our solution:

1. Normalize CT scan
2. Find regions likely to have nodules
3. Predict nodule attributes
4. Aggregate nodule attribute predictions into a global patient-level diagnosis forecast

Ultimately our solution combines 17 3D convolutional neural network models and consists of two ensembles. The models in each ensemble were built using different architectures, training schedules, objectives, subsampled data, and activation functions. This added diversity makes combining the models more effective.

https://www.kaggle.com/c/data-science-bowl-2017/discussion/32544

2nd place solution

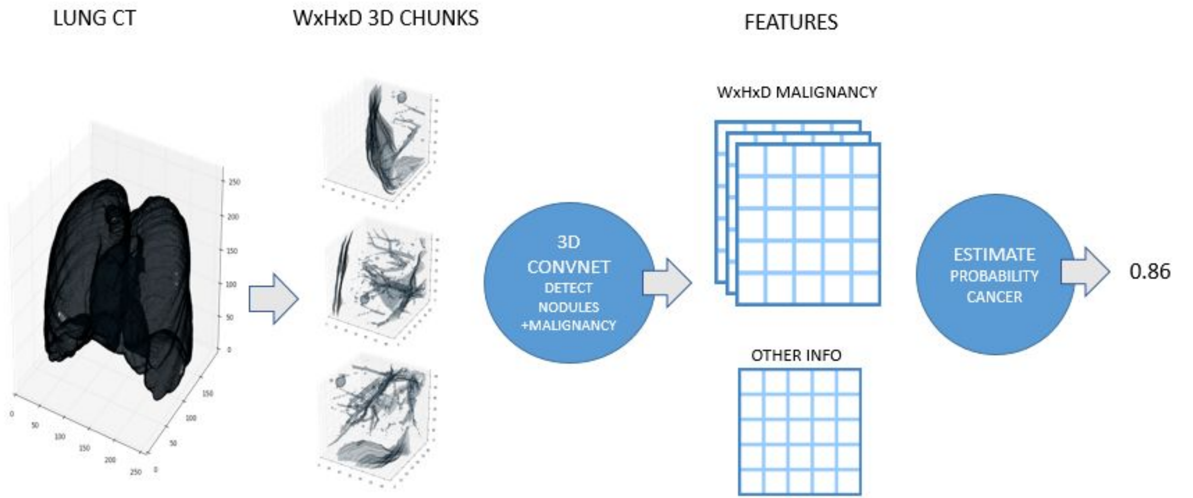


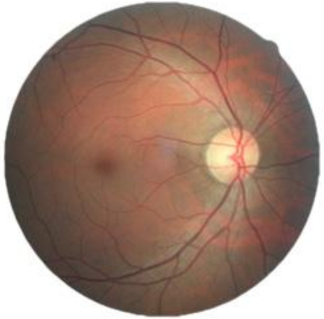
Figure 1. High level description of the approach

https://www.kaggle.com/c/data-science-bowl-2017/discussion/32544

Public sharing controversy

- A lot of useful code is shared in Kernels
- Downside - newcomers have easier time

Competitions move the field



Netflix competition - SVD decomposition for recommender systems

Medical competitions - Diabetic Retinopathy

Currently - Cervical Cancer Screening (7th most frequent deadly illness)

Thanks for your attention!