

TDS

Le Machine Learning par la pratique

Le
Meilleur
Data Scientist
de France



Par Alexia Audevert & Reynald Rivière

Jeudi 20 Octobre 2016

QUI SOMMES- NOUS ?



Alexia AUDEVART

Data & Enthusiasm
@ekito



Reynald RIVIERE

DataScientist
@ Continental Automotive



@aaudevart



@RiviereReynald

Le Contexte

LE MEILLEUR DATA SCIENTIST DE FRANCE



French Data

1ère organisation de cette compétition

Organisé par French Data

250 Data Scientists

2h d'affrontement

10 Mai 2016 - Ecole 42 (Paris)

*Objectif: Mettre au point le meilleur modèle de machine learning
pour estimer combien coûte une boîte de médicaments*

mobiskill

engie

EY
Building a better
working world

etalab

datascience.net

AIR LIQUIDE
Creative Oxygen

PLATE-FORME DE CONCOURS DE DATA SCIENCE



- Création : Avril 2010
 - Plate-forme Internationale
 - + de 250 compétitions
 - + de 110 datasets
 - + de 50 000 Data Scientists
 - + de 3 Millions \$ de récompenses
- Création mi 2013 par la société Bluestone & GENES (Groupe des Écoles Nationales d'Économie et Statistique)
 - Plate-forme Francophone
 - + de 15 compétitions
 - + de 5 500 Data Scientists
 - + de 80 000 € de récompenses

PLATE-FORME DE CONCOURS DE DATA SCIENCE

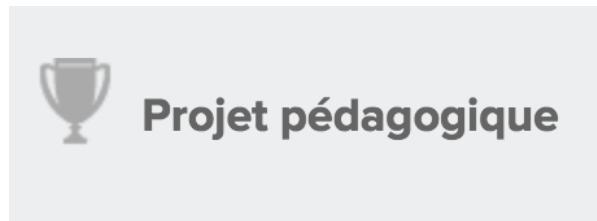
KAGGLE - DATA SCIENCE.NET



Challenge ouvert à tous sur la plate-forme



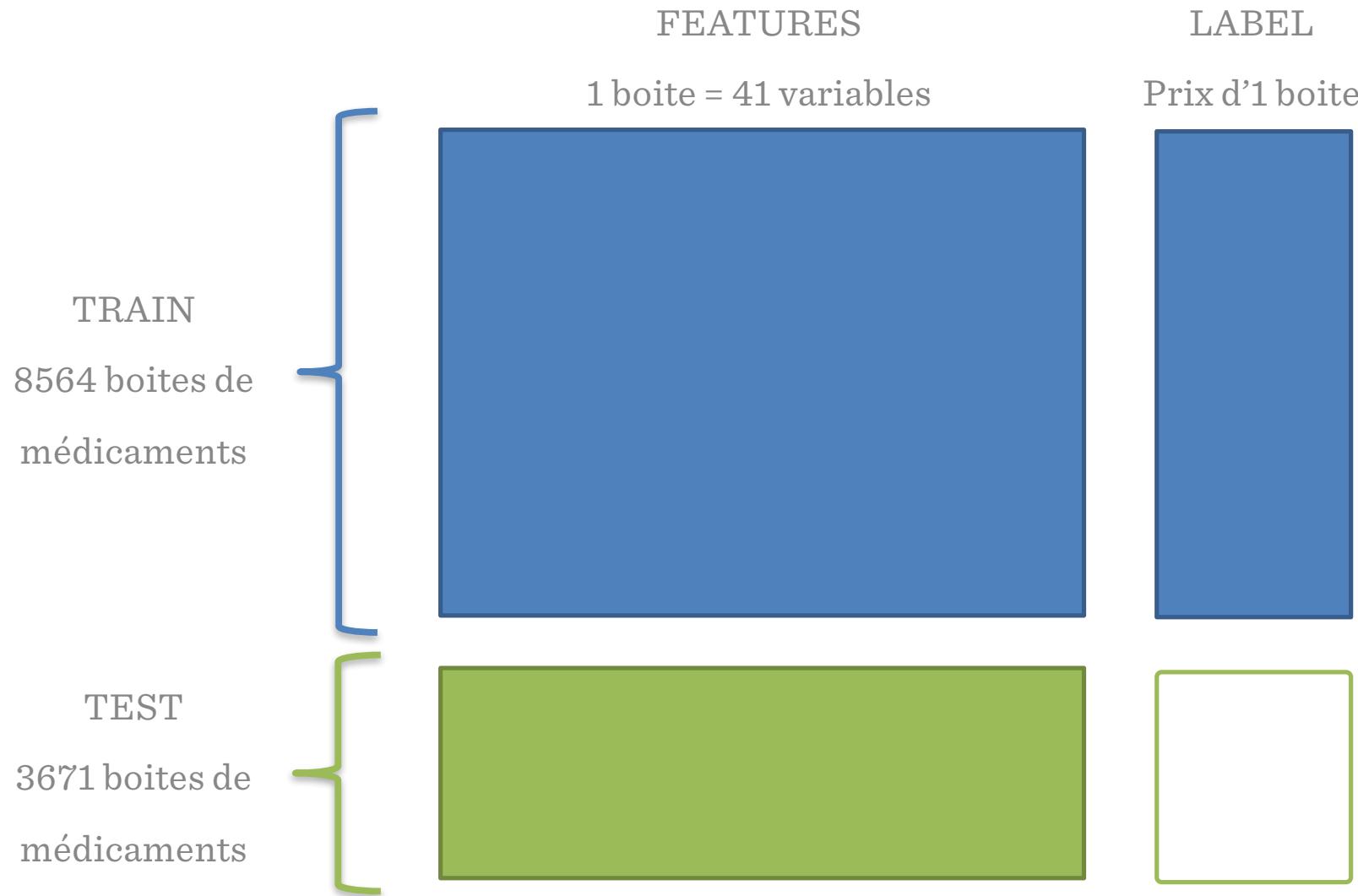
11 Mai 2016 – 31 Octobre 2016



Les Données

Connaitre les données pour comprendre le problème

LES DONNEES



METRIQUE

MAPE: Mean Absolute Percentage Error

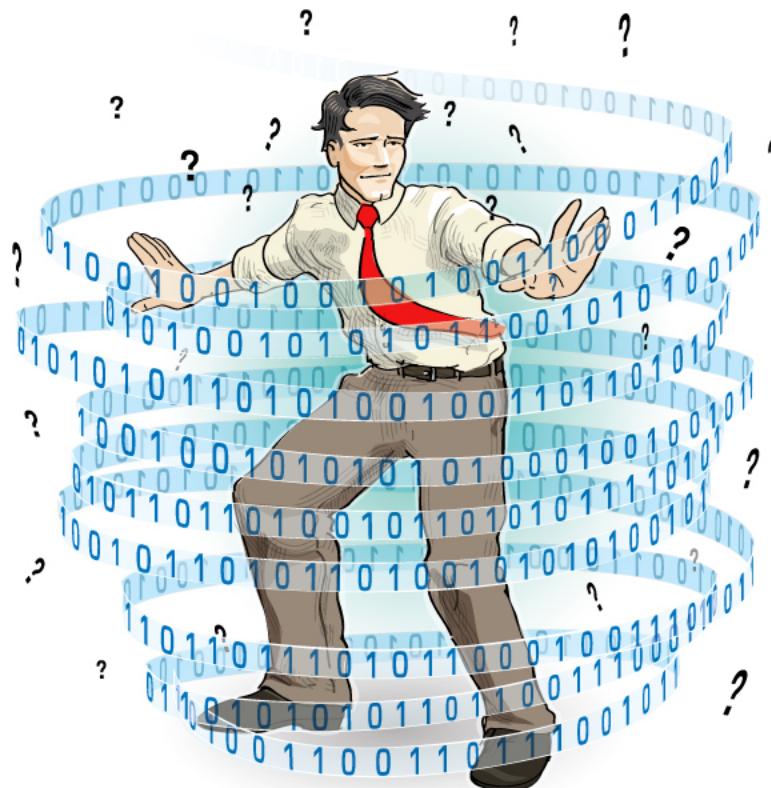
=> Erreur en pourcentage absolu de la moyenne

Moyenne des écarts en valeur absolue par rapport aux valeurs observées

Exprime l'exactitude sous forme de pourcentage (indicateur pratique de comparaison)

$$MAPE = \frac{100}{n} \sum_{k=1}^n \left| \frac{\text{Valeur Observée}_k - \text{Valeur Prédite}_k}{\text{Valeur Observée}_k} \right|$$

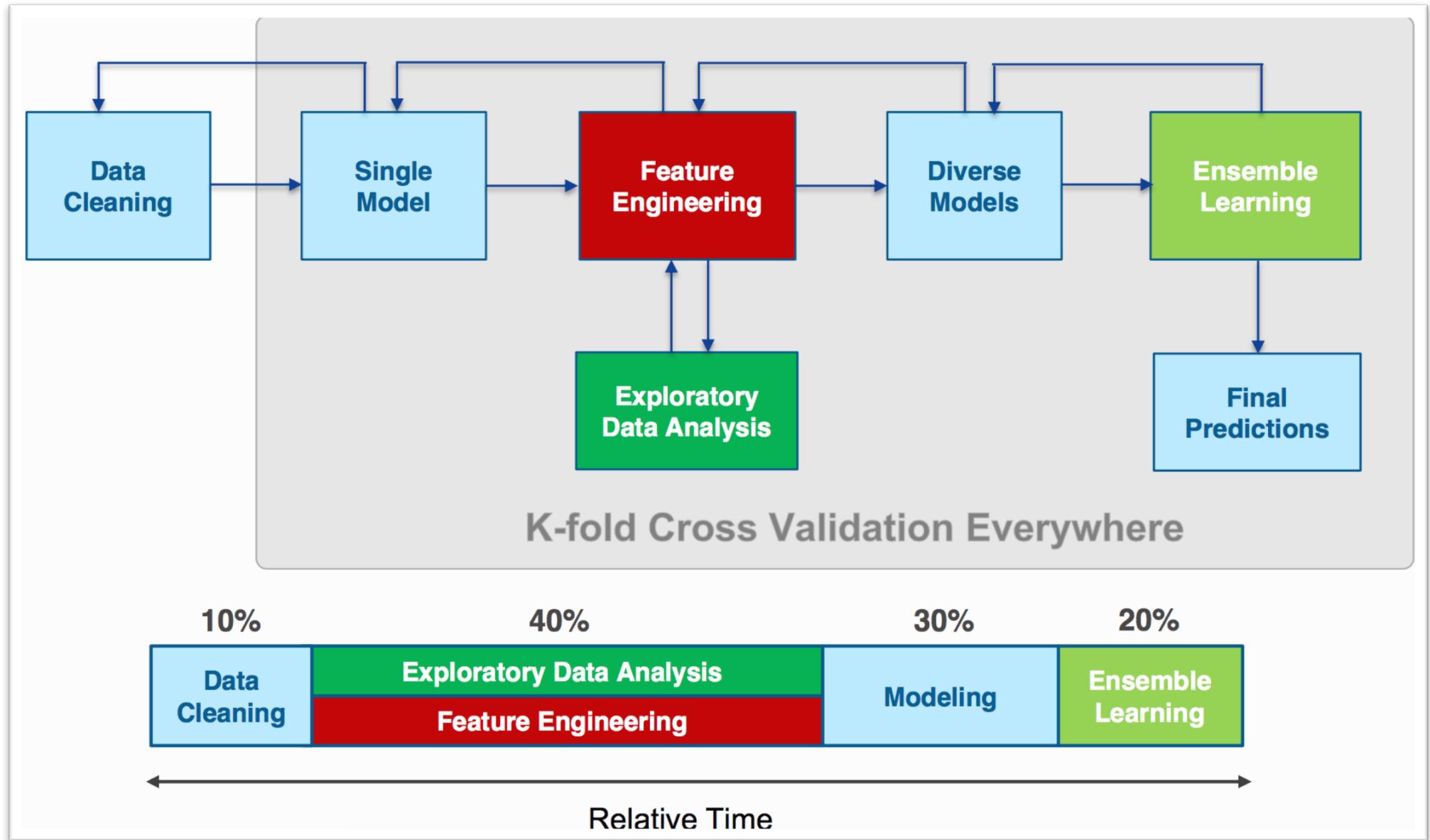
En route vers le code...



© behance : <https://www.behance.net/gallery/5958295/Data-Hero-Oya-Group>

Les étapes du Machine Learning

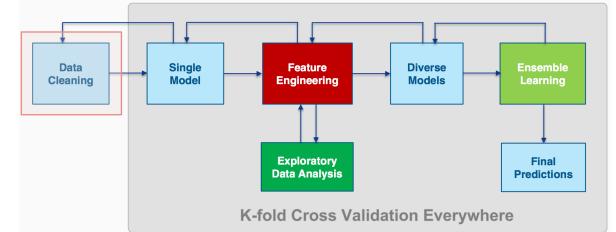
RECOMMENDED DATA SCIENCE PROCESS (IMHO)



DATA Cleaning

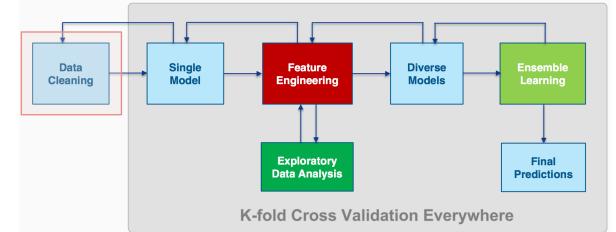
Permettre l'analyse de données

DATA CLEANING TECHNIQUES



- *Supprimer les données en doublon, les constantes, les variables trop bruitées*
 - *Python: VarianceThreshold in scikit-learn package, scipy.stats.pearsonr*
- *Réduire le nombre de dimensions* du dataset des données inutiles peut accélérer la phase de training sans diminuer la performance,
- Attention au risque de “perturber” le dataset !
Il faut donc être sûr de ce que l'on fait avec une cross validation.

DATA CLEANING TECHNIQUES

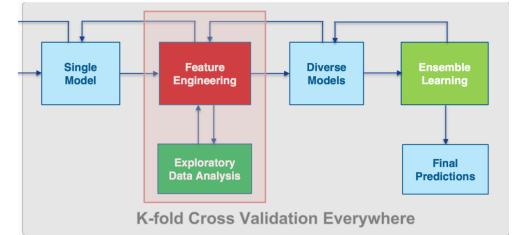


- *Traiter les données manquantes (NaN) :*
 - Par **interpolation** : moyenne, médiane, valeur la + fréquente... voire un autre classifier.
 - *Python: Imputer(strategy='mean', axis=0) in scikit-learn package*
 - Par **suppression** des lignes ou colonnes avec trop de valeur manquantes, etc.,
 - Par **transformation**
 - binaires : -1 si $v < 0$, 1 si $v > 0$, 0 pour les NaN,
 - catégorielles : créer une catégorie à part comme "inconnue"
 - numériques : le valoriser avec une valeur aberrante (-99999999)
- L'absence de données peut être une information (à ne pas supprimer donc)

Feature Engineering

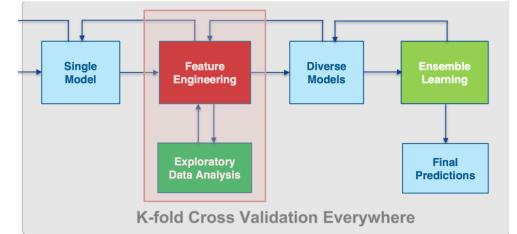
Une des clés du succès en ML

FEATURE ENGINEERING



- Trouver la "*gold feature*"
- *Etapes clés :*
 - Data Transformation
 - Feature Extraction
 - Sélection
- *Transformer la target lors de l'apprentissage*
 - Atténue l'influence des valeurs aberrantes
 - ‘Linéarise’ le problème,
 - $\log(x+1)$, \sqrt{x} , etc.

FEATURE ENGINEERING



- *Data transformation*

- Evite qu'une feature ayant beaucoup de grandes valeurs ne domine les features ayant de faible.

- *Rescaling*

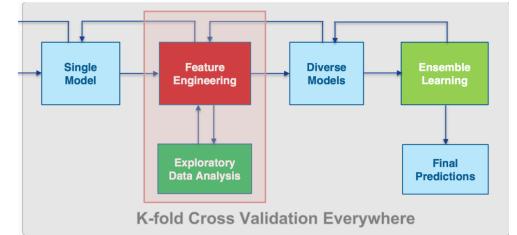
- Distribue les variables dans un intervalle donné (e.g., [-1,+1], [0,1], ...)
- *Python scikit-learn: MinMaxScaler*

- *Standardization*

- Centre les valeurs ($\mu = 0$) and fixe la variance ($\sigma = 1$)
- *Python scikit-learn: StandardScaler*

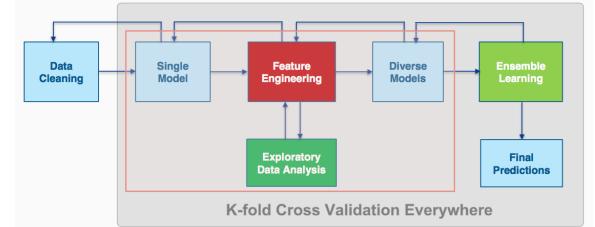
- Ces transformations sont critiques pour certains classifieurs

FEATURE ENGINEERING



- *Data transformation*
 - *Transformation des variables non numériques en numériques*
 - *Labeled Encoding*
 - Transforme la variable catégorielle ‘n classes’
 - *Python scikit-learn:LabelEncoder*
 - *One Hot Encoding ou dummification*
 - Transforme la variable catégorielle en n variables binaires
 - *Python scikit-learn: OneHotEncoder ou get_dummies*
 - Effet de bords : attention au fléau de la dimension !

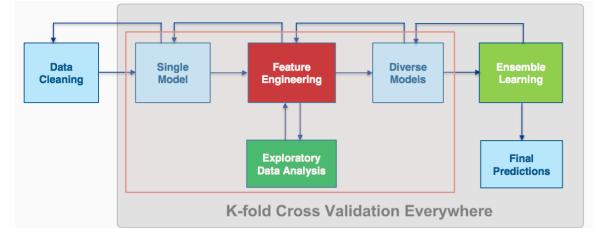
FEATURE ENGINEERING



- *Feature Extraction*

- Extraire *l'information* contenue dans un *texte* par exemple
- Possibilité d'extraire des *compteurs d'occurrences* comme feature
- Quelques exemples :
 - **Location :**
=> Adresse, ville, code postale....
 - **Time :**
=> Année, mois, jour, heure, minute, intervalle, semaine de l'année, weekend, matin, midi, soir, etc.

FEATURE ENGINEERING



- *Feature Extraction*

- Les fichiers **HTML** sont régulièrement utilisés en classification
 - Exemple de feature :
 - html attributes (id, class, href, src,)
 - tag names, inner content
 - Le outils d'analyse de texte peuvent générer un très (trop) grand de features qui être peuvent être réduites avec des outils comme **PCA** ou **truncated SVD**
 - Avec le risque de perdre certaines informations ! La cross validation aide à trouver un l'équilibre
 - Outils :
 - *Python scikit-learn: TF-IDF, PCA, etc.*
 - *Python : BeautifulSoup pour le traitement de fichiers HTML, etc.*

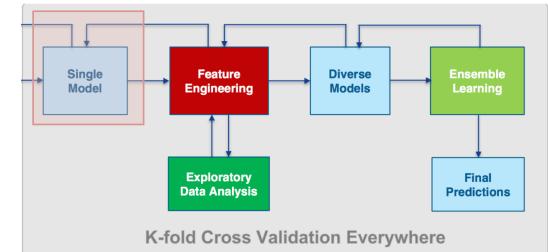
En route vers le code...



© behance : <https://www.behance.net/gallery/5958295/Data-Hero-Oya-Group>

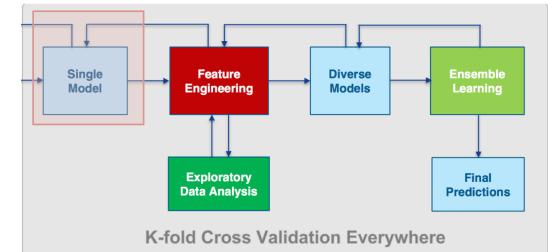
Single Model

SINGLE MODEL



Model Type	Name	R	Python
Regression	Linear Regression	• <code>glm</code> , <code>glmnet</code>	• <code>sklearn.linear_model.LinearRegression</code>
	Ridge Regression	• <code>glmnet</code>	• <code>sklearn.linear_model.Ridge</code>
	Lasso Regression	• <code>glmnet</code>	• <code>sklearn.linear_model.Lasso</code>
Instance-based	K-nearest Neighbor (KNN)	• <code>knn</code>	• <code>sklearn.neighbors.KNeighborsClassifier</code>
	Support Vector Machines (SVM)	• <code>svm {e1071}</code> • <code>LiblinearR</code>	• <code>sklearn.svm.SVC</code> , <code>sklearn.svm.SVR</code> • <code>sklearn.svm.LinearSVC</code> , <code>sklearn.svm.LinearSVR</code>
Hyperplane-based	Naive Bayes	• <code>naiveBayes {e1071}</code>	• <code>sklearn.naive_bayes.GaussianNB</code> • <code>sklearn.naive_bayes.MultinomialNB</code> • <code>sklearn.naive_bayes.BernoulliNB</code>
	Logistic Regression	• <code>glm</code> , <code>glmnet</code> • <code>LiblinearR</code>	• <code>sklearn.linear_model.LogisticRegression</code>

SINGLE MODEL

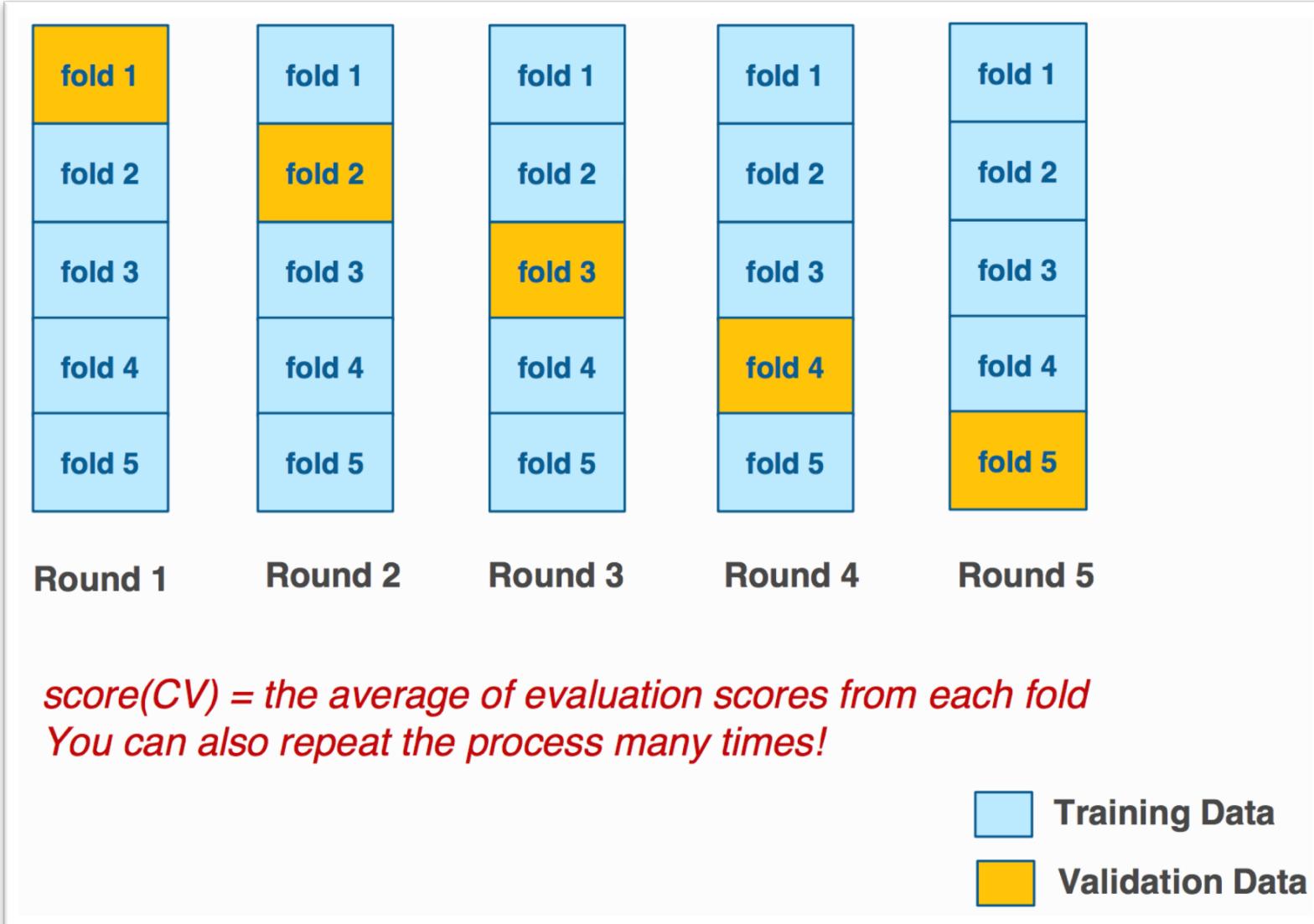


Model Type	Name	R	Python
Ensemble Trees	Extremely Randomized Trees	<ul style="list-style-type: none"> • <code>extraTrees</code> 	<ul style="list-style-type: none"> • <code>sklearn.ensemble.ExtraTreesClassifier</code> • <code>sklearn.ensemble.ExtraTreesRegressor</code>
	Gradient Boosting Machines (GBM)	<ul style="list-style-type: none"> • <code>gbm</code> • <code>xgboost</code> 	<ul style="list-style-type: none"> • <code>sklearn.ensemble.GradientBoostingClassifier</code> • <code>sklearn.ensemble.GradientBoostingRegressor</code> • <code>xgboost</code>
Neural Network	Multi-layer Neural Network	<ul style="list-style-type: none"> • <code>nnet</code> • <code>neuralnet</code> 	<ul style="list-style-type: none"> • PyBrain • Theano
Recommendation	Matrix Factorization	<ul style="list-style-type: none"> • <code>NMF</code> 	<ul style="list-style-type: none"> • nimfa
	Factorization machines		<ul style="list-style-type: none"> • pyFM
Clustering	K-means	<ul style="list-style-type: none"> • <code>kmeans</code> 	<ul style="list-style-type: none"> • <code>sklearn.cluster.KMeans</code>
	t-SNE	<ul style="list-style-type: none"> • <code>Rtsne</code> 	<ul style="list-style-type: none"> • <code>sklearn.manifold.TSNE</code>

Cross validation

K-FOLD CROSS VALIDATION

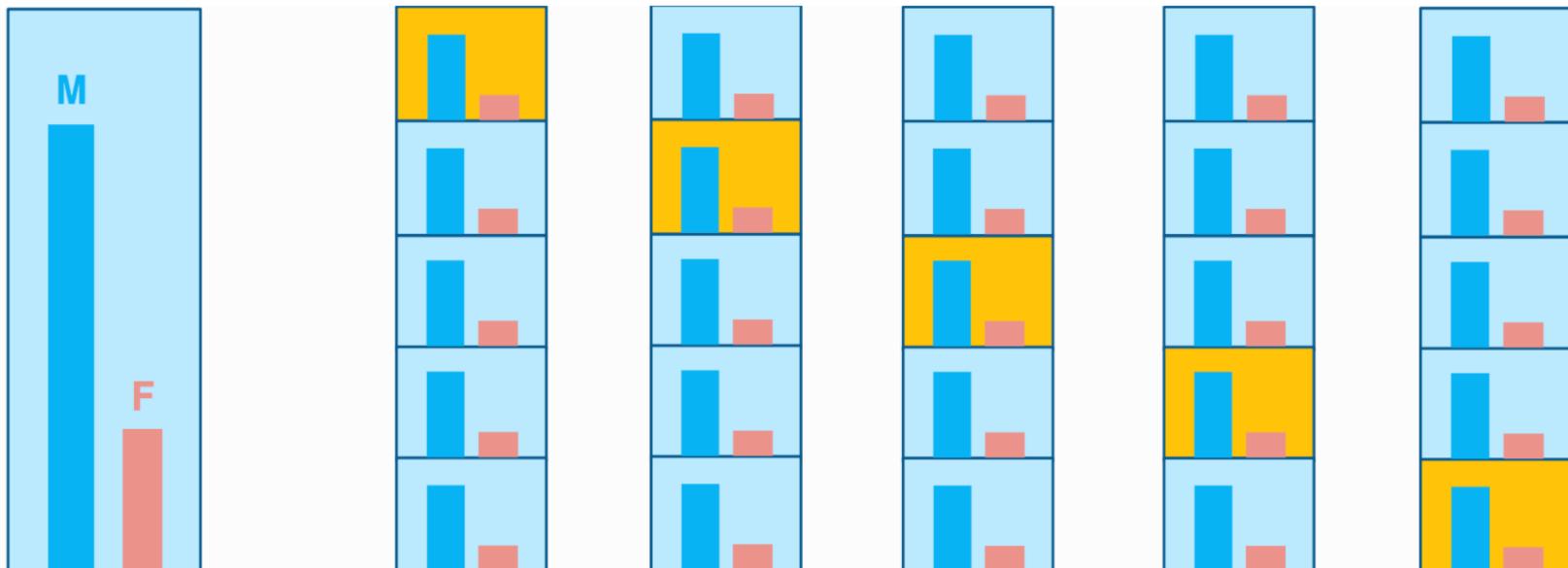
K=5



$score(CV) = \text{the average of evaluation scores from each fold}$
You can also repeat the process many times!

- Training Data
- Validation Data

STRATIFIED k-FOLD CROSS VALIDATION K=5



Keep the distribution of classes in each fold

- Training Data (Blue square)
- Validation Data (Orange square)

En route vers le code...

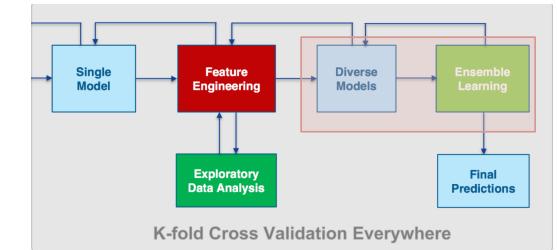


© behance : <https://www.behance.net/gallery/5958295/Data-Hero-Oya-Group>

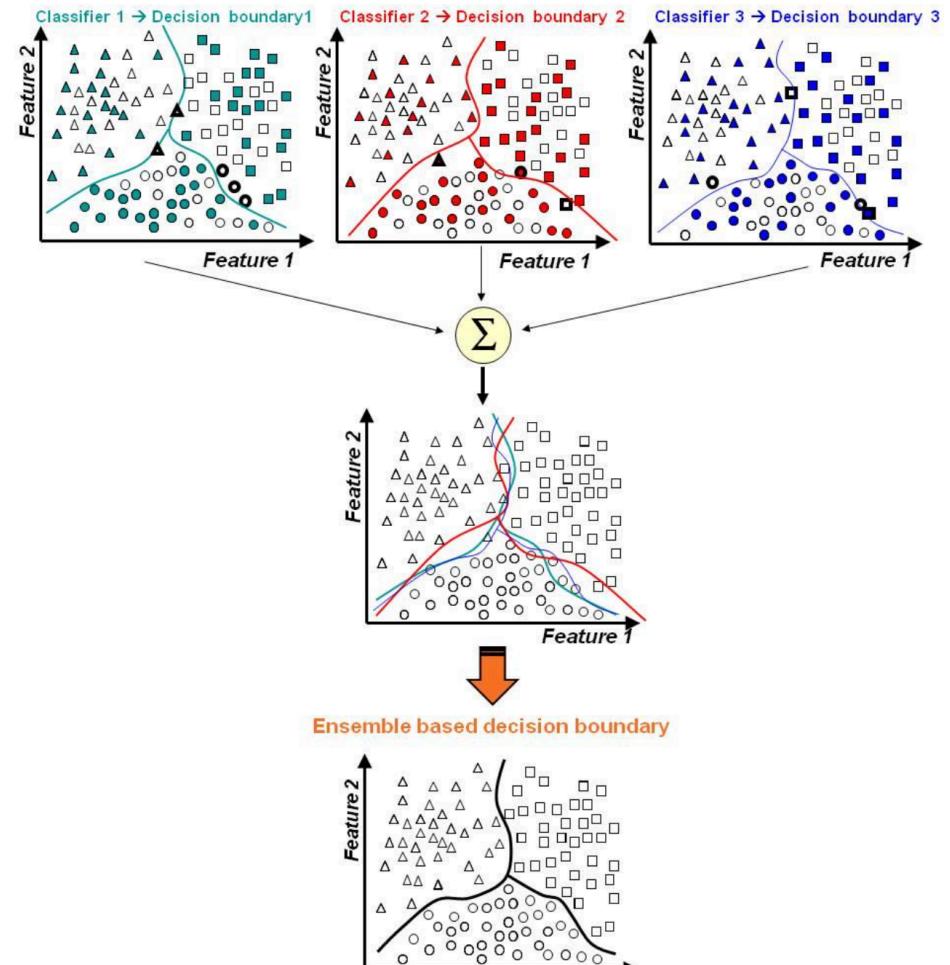
Ensemble Learning

Pour dépasser l'approche mono modèle

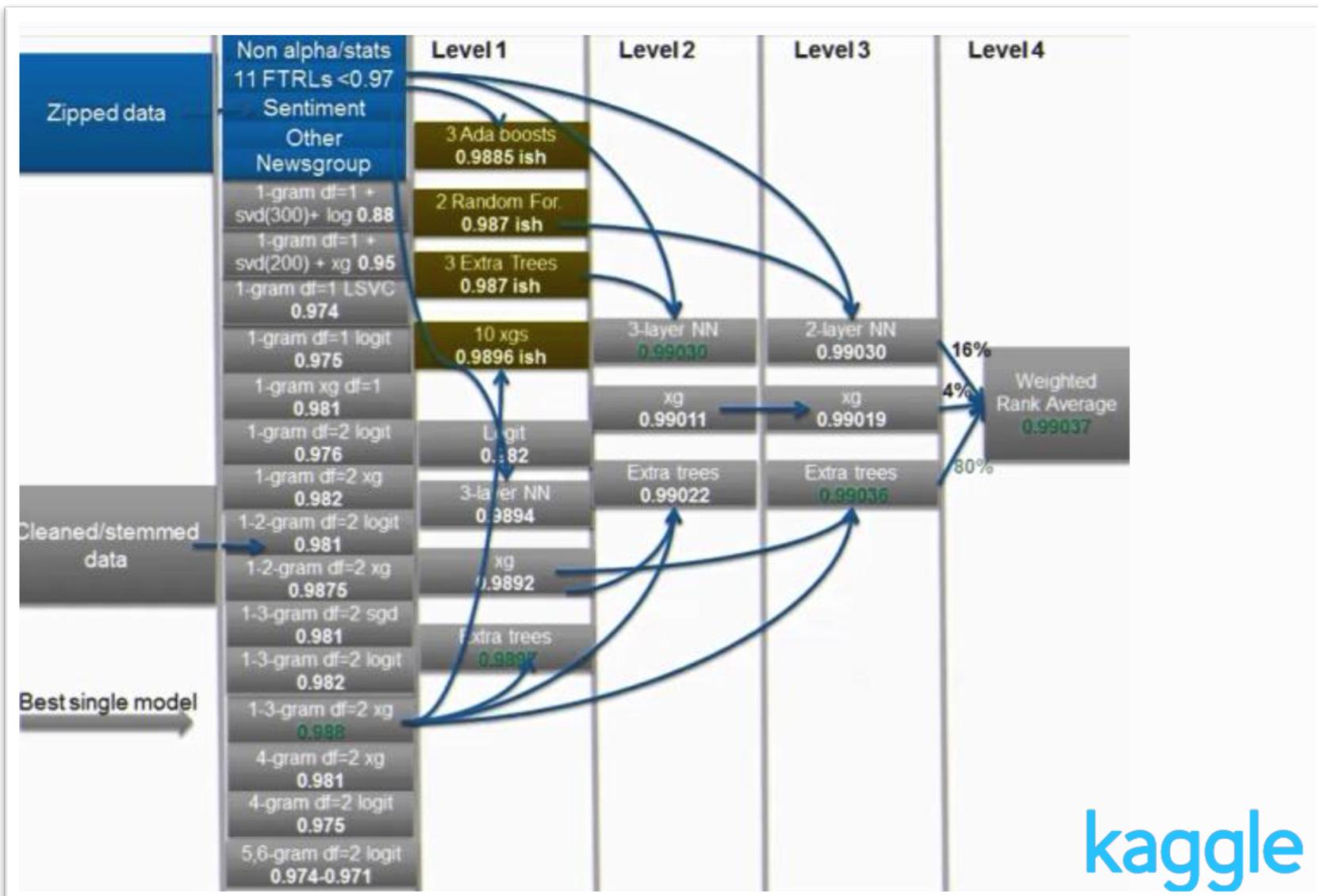
ENSEMBLE LEARNING



- Les *differents classifieurs* “captent” *differents aspects* d'un même dataset
- *Approche simple* : moyenner (avec ou sans pondération) les prédictions obtenues
- *Limite* le risque *d'overfitting*,
- *Améliore* la *capacité de généralisation*



ENSEMBLE LEARNING GENERALIZED STACKING



Le Machine Learning par la pratique

Par Alexia Audevert & Reynald Rivière

MERCI POUR VOTRE ATTENTION



QUESTIONS ?

QUELQUES LIENS

- Livre FR – Data Science : fondamentaux et études de cas
- MOOC Machine Learning Coursera – Andrew NG
 - <https://www.coursera.org/learn/machine-learning>
- General Tips for participating Kaggle competitions
 - <http://fr.slideshare.net/markpeng/general-tips-for-participating-kaggle-competitions>
- Ensemble Learning
 - http://www.scholarpedia.org/article/Ensemble_learning
- NoteBook du code + slides
 - <https://github.com/aaudevart/MDF-TDS>



ANNEXES

BIG SHAKE UP ON KAGGLE PRIVATE LB!

 TAB FOOD INVESTMENTS

Completed • \$30,000 • 2,257 teams

Restaurant Revenue Prediction

Mon 23 Mar 2015 – Mon 4 May 2015 (7 months ago)

[Dashboard](#) ▾

Private Leaderboard - Restaurant Revenue Prediction

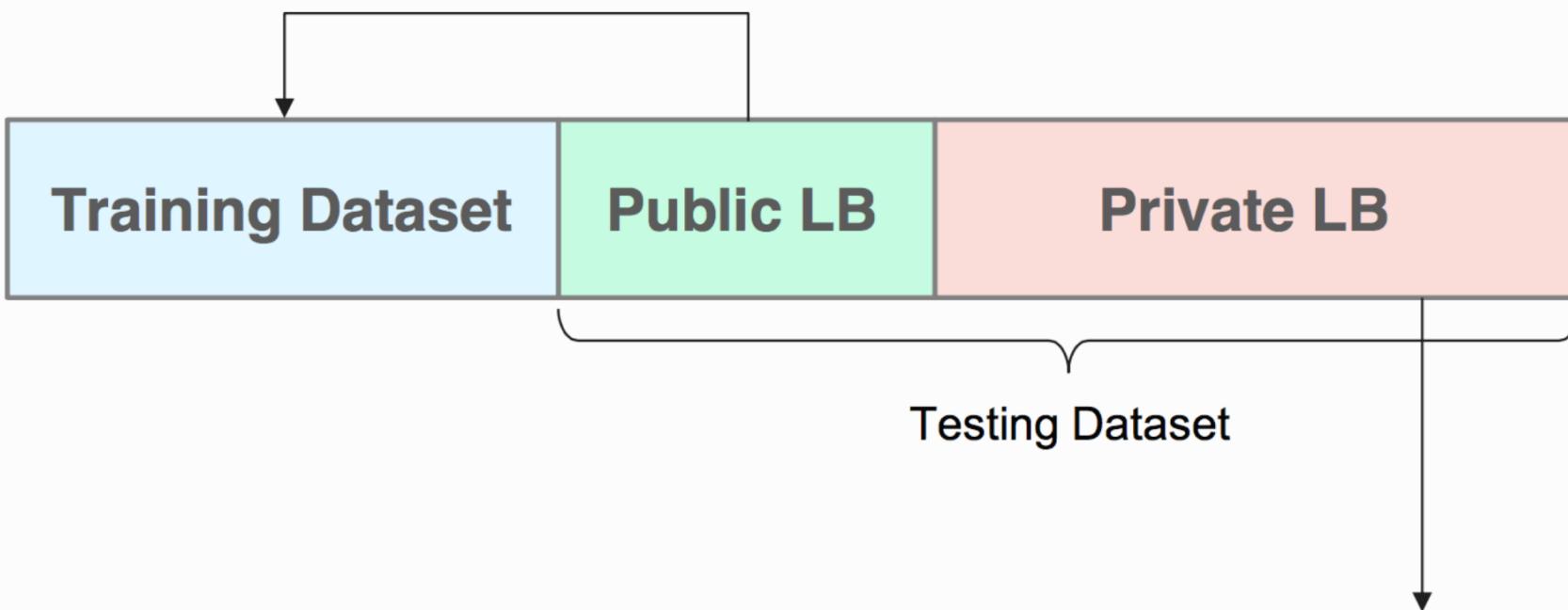
This competition has completed. This leaderboard reflects the final standings.

See someone using multiple accounts? [Let us know.](#)

#	Rank	Team Name	model uploaded * in the money	Score	Entries	Last Submission UTC (Best – Last Submission)
1	↑205	Arsenal	‡ *	1727811.48554	21	Fri, 24 Apr 2015 03:14:27 (-23.9d)
2	↑42	Climent_Josep	‡ *	1729265.36129	27	Mon, 04 May 2015 06:32:55 (-28.2h)
3	↑338	OldSchool	‡ *	1732292.38960	72	Tue, 28 Apr 2015 08:53:55 (-3.9d)
4	↑170	McData	‡	1734366.17757	53	Fri, 01 May 2015 06:01:48
5	↑294	Statsfreak		687 ↓125 Yinghao		1837568.63872
6	↑61	Manuel Díaz (TFI)		688 ↓683 Analytic Bastard		1837732.75137
7	↑482	mj_coder		689 ↓535 david li		1838065.25023
8	↑345	Rynsin		690 ↑752 Caner		1838149.98065
9	↑1058	Chapulist		691 ↓246 Nithin		1838160.15572
10	↑48	Pete_Ter		692 ↓399 kalyan_TFI		1838249.27788
				693 ↑16 Axel		1838258.11833
				694 ↓268 Nilesh Kadam		1838303.64440

HOW KAGGLE PREVENT OVERFITTING ?

Validation feedback but sometimes misleading



Might be different from public LB
(used to determine final prize winners!)