Introduction to Statistical Learning with Applications

CM9: ML competitions, metrics, etc.

Pedro L. C. Rodrigues

The Netflix prize (2006 to 2009)

See more details at: https://www.thrillist.com/entertainment/nation/the-netflix-prize

The Netflix prize (2006 to 2009)

- 480k users
- 17k movies

$$M = \begin{bmatrix} \blacksquare & & & \blacksquare & & & & \blacksquare \\ & \blacksquare & & & & \blacksquare & & \\ & & & & \blacksquare & & & \\ & & \blacksquare & & & & \blacksquare & & \blacksquare \end{bmatrix}$$

Only 100M of the cells are known, i.e. we only know 1.2% of the ratings given by users

GOAL: Propose a recommandation algorithm that beats Netflix's by at least 10%

according to a score provided by the company

The Netflix prize (2006 to 2009)

The winning solution was good… but it was way too complicated to deploy

Nevertheless, it contained some interesting ideas that became standard in ML

- Data science competitions
- Performance metrics

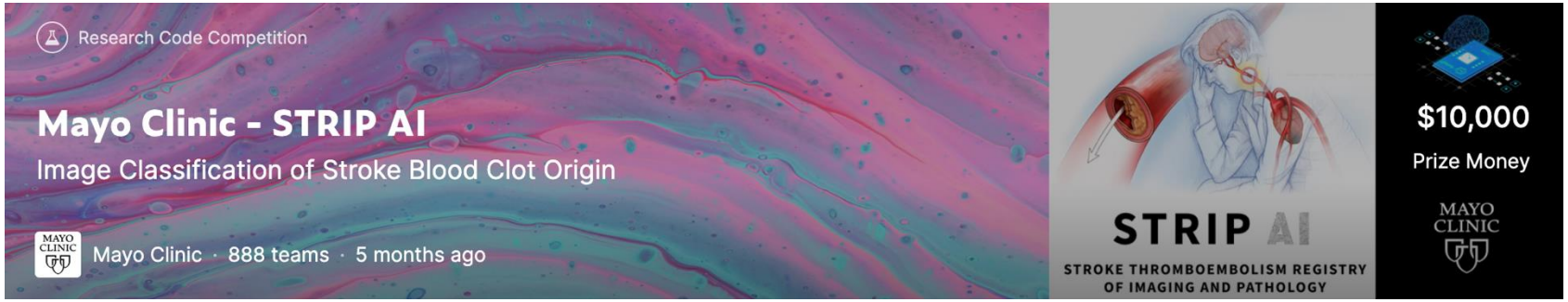
Data science competitions

- Data science competitions are always supervised problems
  (Why?)
- The goal is to obtain a function f for which every $\mathbf{x}$ yields prediction $f(\mathbf{x})$
- Such predictions should be close to the observations, i.e. $y \approx f(\mathbf{x})$

The data

Every challenge contains at least three files to download:

`x_train` `y_train`	the data on which participants will elaborate and train their model to learn the prediction function
`x_test`	the data used for evaluation



↳ https://www.kaggle.com/competitions/mayo-clinic-strip-ai/overview

↳ https://www.kaggle.com/competitions/amex-default-prediction

↳ https://www.kaggle.com/c/happy-whale-and-dolphin

Data science competitions

The score

The error between prediction $f(\mathbf{x})$ and observation y is evaluated with $L\big(f(\mathbf{x}), y\big)$

The goal is to minimize the average score on the inputs. The actual calculation of the score depends of the competition and is always explained in its presentation.

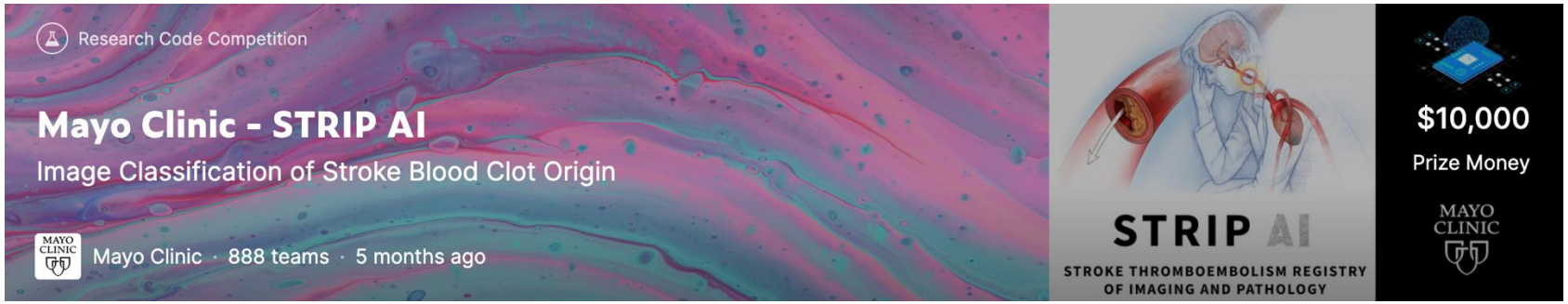
↳ These may be MSE, Cross Entropy, R2 score, AUROC, F1 score, other custom losses

Test set separation and overfitting

The test set is split in two halves, the public and private test sets.

- The Public score is computed on the public set. It is provided after each submission
- The Private score is obtained on the private set and is provided by the end of the competition.

This split prevents participants from overfitting the whole test set, as they can't see how their solution behaves on the entire test set. (Why ?)



More often than not, assessing a method's score is one of the the most important parts in a competition…

IRL it would be data cleaning

↳ https://www.kaggle.com/competitions/mayo-clinic-strip-ai/overview

PUBLIC leaderboard

#	Team	Members	Score	Entries	Last	Solution
1	Hiroshi Sakiyama		0.33967	88	5mo	
2	Flype		0.34478	21	5mo	
3	hittie		0.35464	29	6mo	

PRIVATE leaderboard

#	△	Team	Members	Score	Entries	Last	Solution
1	▲ 470	khyeh		0.65993	6	6mo	
2	▲ 459	Ilya los		0.66421	7	5mo	
3	▲ 419	miyasaki		0.66432	16	5mo	

Data science competitions

Doing data competitions is a very practical skill that is hard to teach in class.

- Two very good places to start are:

  Titanic dataset: predicting whether a person died or not on the Titanic accident

  ↳ https://www.kaggle.com/competitions/titanic

  Houses dataset: predicting the price of a house given several features

  ↳ https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/overview

- Another good reference are the videos (in French) from S. Mallat's course @ CdF

  ↳ https://www.youtube.com/watch?v=8IAcJmP9bdU – Pierre Courtiol "S'attaquer à une competition de ML"

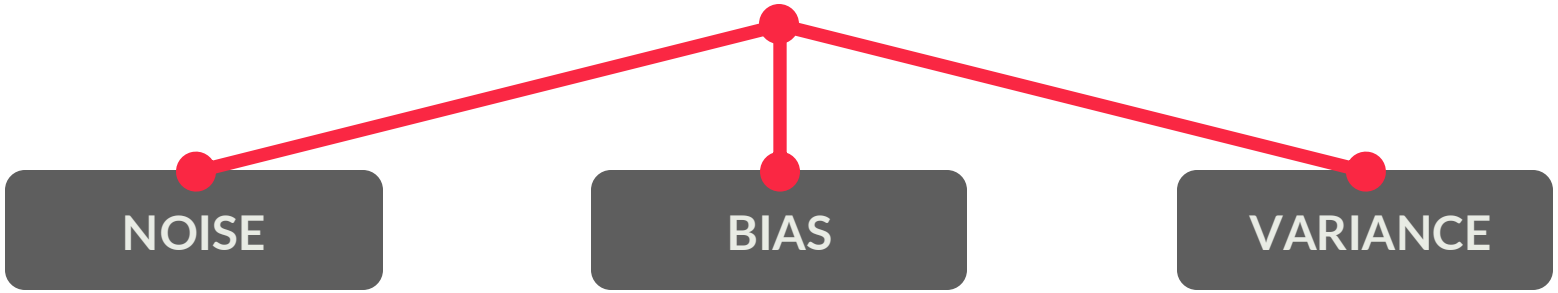
Hint: Most competitions are won using bagging, boosting, and regularization…

Data science competitions

Remember from CM3 that the generalization error of a regressor can be decomposed as

↳ It is possible to do the same decomposition for classifiers

$$\mathcal{L}\left(\hat{r}_{\mathcal{D}}(x)\right) = \mathbb{E}_{X,Y}\left[\left(Y - \hat{r}_{\mathcal{D}}(X)\right)^2 \mid X = x\right]$$



There are several techniques that people use to reduce the generalization error

- Regularization can increase bias but controls variance
- Bagging tries to reduce variance
- Boosting tries to reduce bias

- Data science competitions
- Performance metrics

Performance metrics

So far we have measured the quality of our predicted models based on

- The mean squared error (MSE) for regression
- Accuracy for classification

But in many situations these metrics may not be the most informative ones

Example Consider the default dataset from the James et al. (2022) book

The dataset has three predictors and binary labels

We can fit a logistic regression model and do

$$f_\beta(\mathbf{x}) \begin{cases} \geq 0.5 & \text{classify as YES} \\ < 0.5 & \text{classify as NO} \end{cases}$$

```
> head(df)
  default student   balance    income
1      No      No  729.5265 44361.625
2      No     Yes  817.1804 12106.135
3      No      No 1073.5492 31767.139
4      No      No  529.2506 35704.494
5      No      No  785.6559 38463.496
6      No     Yes  919.5885  7491.559
> dim(df)
[1] 10000     4
>
```

Performance metrics -- Classification

Example Consider the default dataset from the James et al. (2022) book

The classifier has 97.1% of accuracy 💪🏻

But the data set has 3.3% of labels YES and 96.7% of labels NO 🤡

		TRUE OBSERVATIONS	
		FALSE	TRUE
PREDICTED OBSERVATIONS	FALSE	9627	228
	TRUE	40	105

The confusion matrix gives all the information we need

Performance metrics -- Classification

Example Consider the default dataset from the James et al. (2022) book

The classifier has 97.1% of accuracy 💪

But the data set has 3.3% of labels YES and 96.7% of labels NO 🤡

TN = true negative
TP = true positive
FP = false negative
FN = false positive

PREDICTED OBSERVATIONS \ TRUE OBSERVATIONS	FALSE	TRUE
FALSE	**TN** 9627	**FN** 228
TRUE	**FP** 40	**TP** 105

$$\text{ACC} = \frac{TN + TP}{TN + TP + FN + FP}$$

The confusion matrix gives all the information we need

Performance metrics -- Classification

Example Consider the default dataset from the James et al. (2022) book

TN = true negative
TP = true positive
FP = false negative
FN = false positive

PREDICTED OBSERVATIONS \ TRUE OBSERVATIONS	FALSE	TRUE
FALSE	TN 9627	FN 228
TRUE	FP 40	TP 105

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- If we were interested in checking how well the classifier detected clients that were really in default, we would prefer to check the quantity called "True Positive Rate"
- Another quantity that might be of interest is how many clients the bank think will be in default when in fact they are not. This is what we call the "False Positive Rate"

Performance metrics -- Classification

Example Consider the default dataset from the James et al. (2022) book

TN = true negative
TP = true positive
FP = false negative
FN = false positive

PREDICTED OBSERVATIONS \ TRUE OBSERVATIONS	FALSE	TRUE
FALSE	TN 9627	FN 228
TRUE	FP 40	TP 105

$$\text{TPR} = \frac{TP}{TP + FN}$$

$$\text{FPR} = \frac{FP}{TN + FP}$$

- If we were interested in checking how well the classifier detected clients that were really in default, we would prefer to check the quantity called "True Positive Rate"
- Another quantity that might be of interest is how many clients the bank think will be in default when in fact they are not. This is what we call the "False Positive Rate"

Performance metrics -- Classification

In the end of the day, we want a large TPR and a small FPR

Note that depending of the context it might be preferable to invest on one or the other

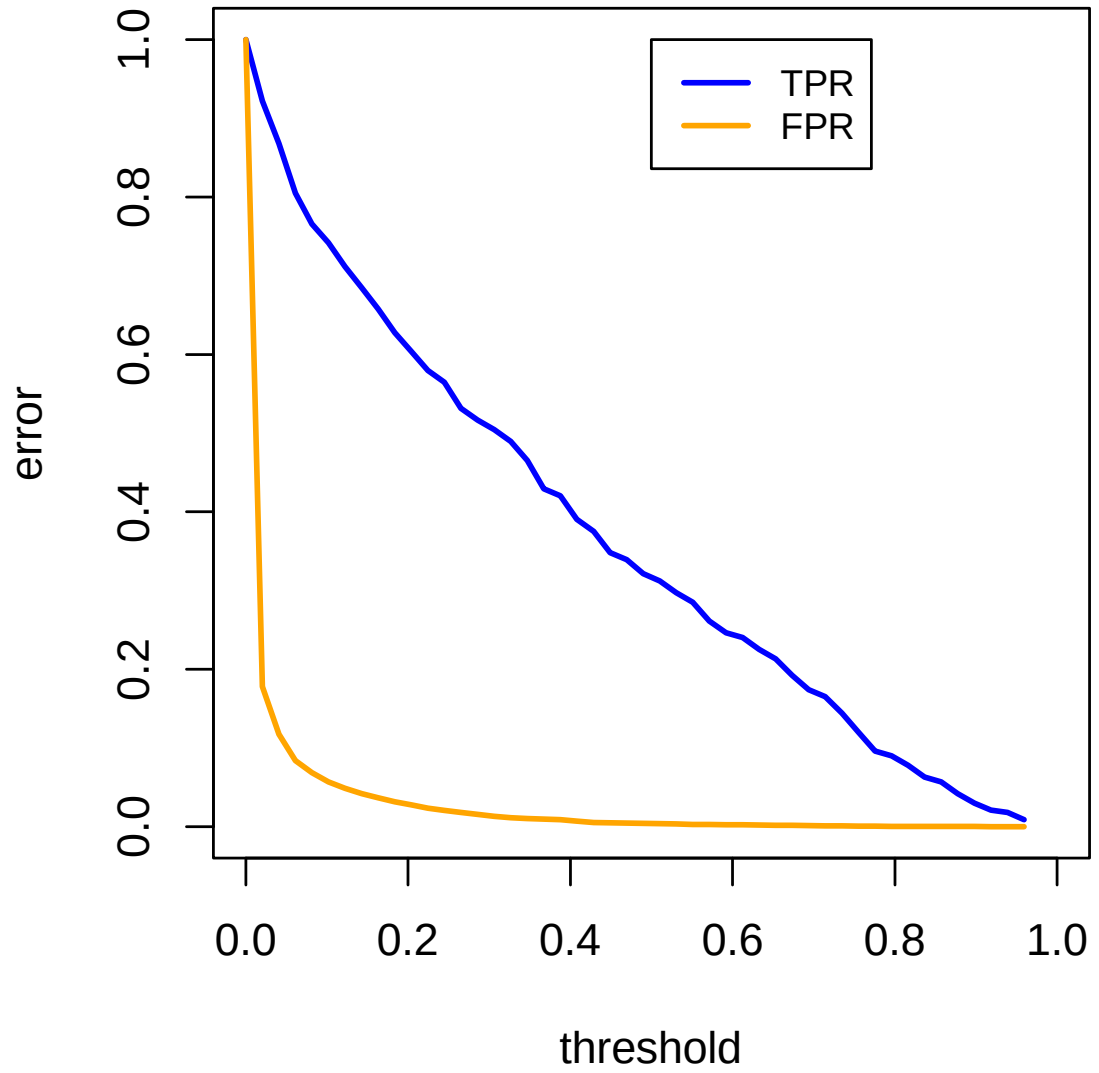
- For banks, it is more costly to have a weak true positive rate (TPR) than a slightly larger false positive rate
- In the case of radars in a cold war, a high false positive rate could have catastrophic consequences

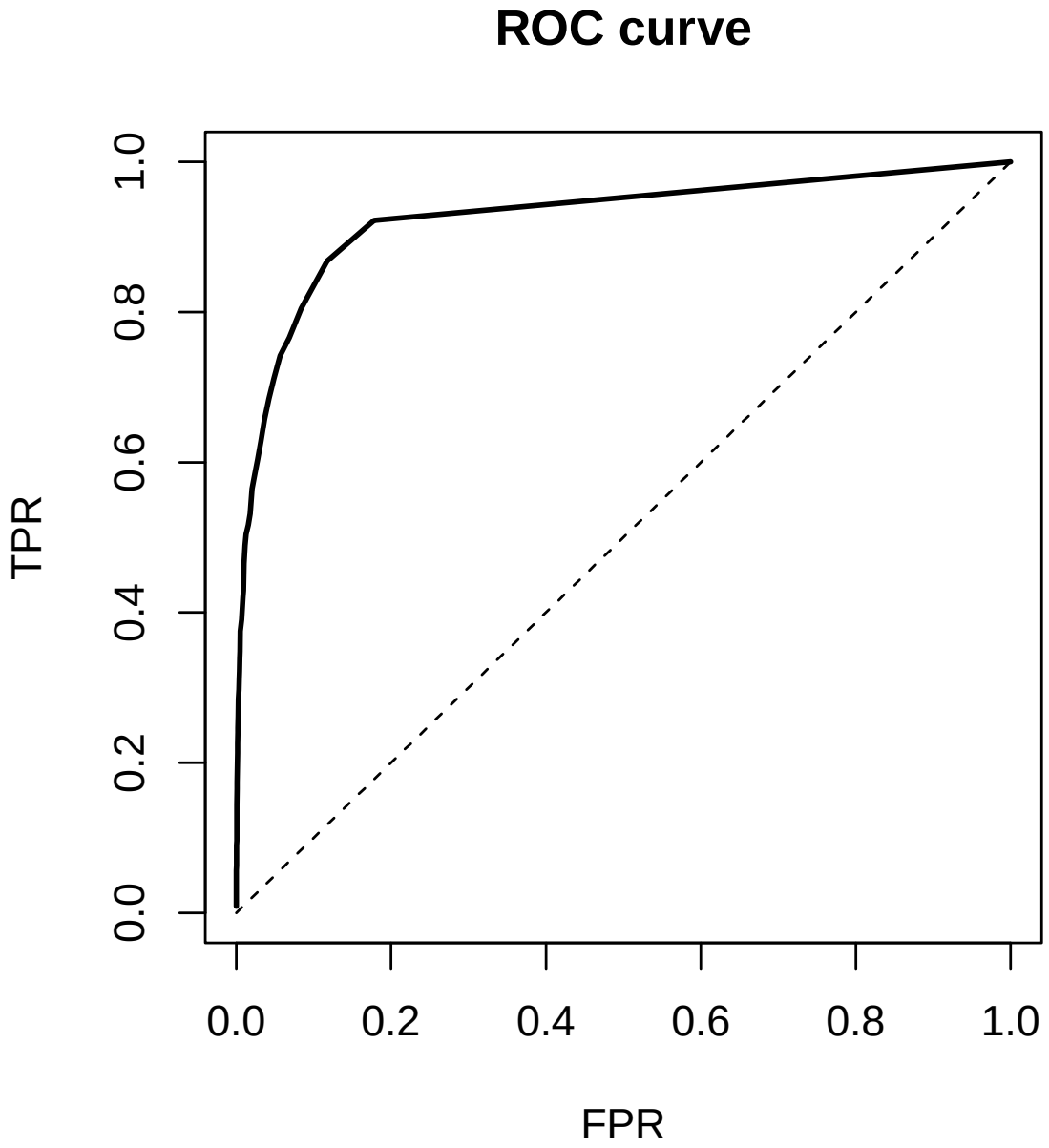
If we change our definition of the classifier, we can change the error rates

$$f_\beta(\mathbf{x}) \begin{cases} \geq \gamma & \text{classify as YES} \\ < \gamma & \text{classify as NO} \end{cases}$$

↳ For smaller values of gamma, the classifier will have more tendency to consider data points as from YES, meaning that we would more easily detect clients that defaulted. Thus, we would have a higher TPR. However, this comes at a cost: we also get a larger FPR. We need to find a balance.

Performance metrics -- Classification





Performance metrics -- Classification

In certain applications, it might be preferable to speak in terms of other quantities

CARACTÉRISTIQUES DE PERFORMANCE

Sensibilité, spécificité et précision clinique

Les performances du test rapide de détection de l'antigène du SRAS-CoV-2 ont été établies à partir de 605 écouvillons nasaux prélevés chez des personnes symptomatiques suspectes de COVID-19. Les résultats indiquent que la sensibilité relative et la spécificité relative sont les suivantes :

Performance clinique du test rapide de détection de l'antigène SRAS-CoV-2

Méthode		RT-PCR		Résultats totaux
Test rapide de détection de l'antigène SARS-CoV-2	**Résultats**	Négatif	Positif	
	Négatif	433	5	438
	Positif	2	165	167
Résultats totaux		435	170	605

Sensibilité relative : 97,1 % (93,1 %-98,9 %)*

Spécificité relative : 99,5% (98,2 %-99,9 %)*

Précision : 98,8 % (97,6 %-99,5 %)*

*95 % intervalles de confiance

La stratification des échantillons positifs après l'apparition des symptômes entre 0 et 3 jours a un pourcentage de concordance positive (PPA) de 98,8 % (n=81) et entre 4 et 7 jours a un PPA de 96,8 % (n=62).

Les échantillons positifs avec une valeur Ct ≤33 ont un pourcentage de concordance positive (PPA) plus élevé de 98,7 % (n=153).

Performance metrics -- Classification

In certain applications, it might be preferable to speak in terms of other quantities

- In medical tests, it is common to use

$$\text{sensitivity} = \frac{TP}{TP + FN}$$

$$\text{specificity} = \frac{TN}{TN + FP}$$

- In information retrieval, it is common to use

$$\text{precision} = \frac{TP}{TP + FP}$$

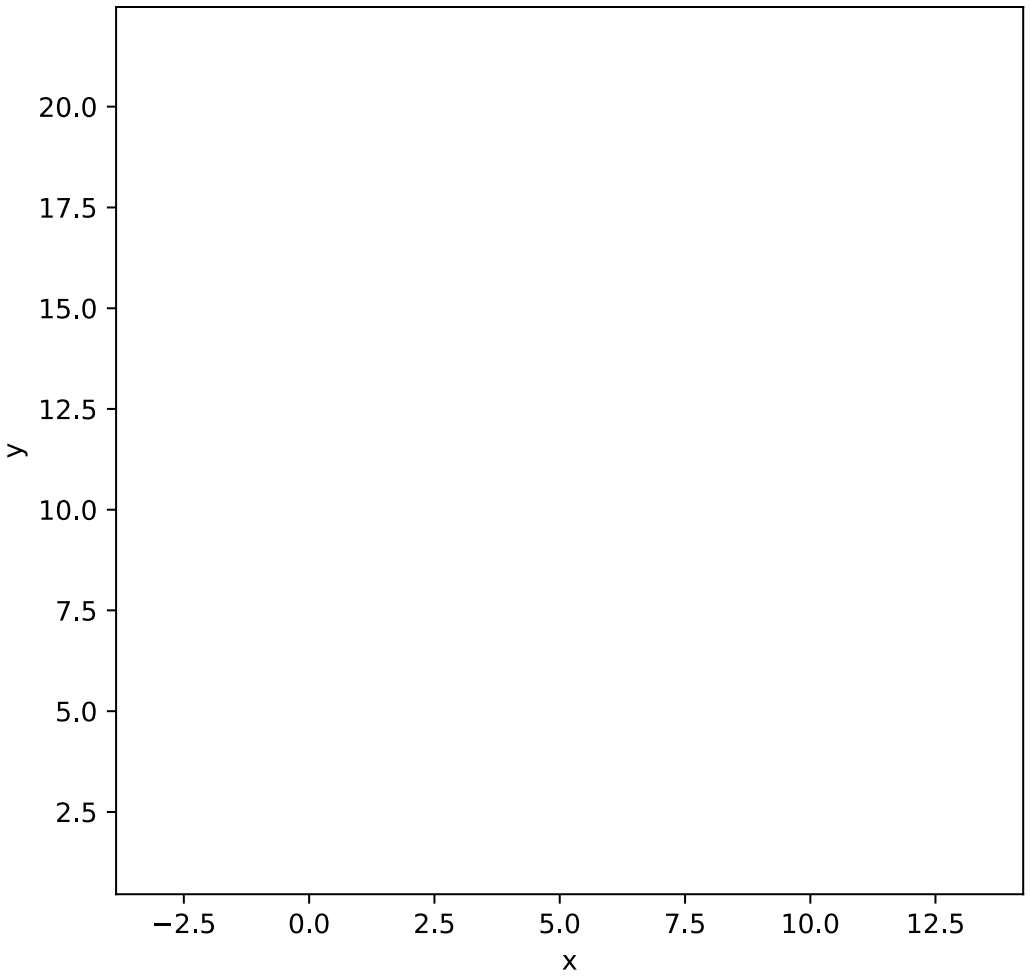
among all positive predictions from our classifier, how many of them were correct.

$$\text{recall} = \frac{TP}{TP + FN}$$

how many of the truly positive observations were indeed detected by our classifier

Performance metrics -- Regression

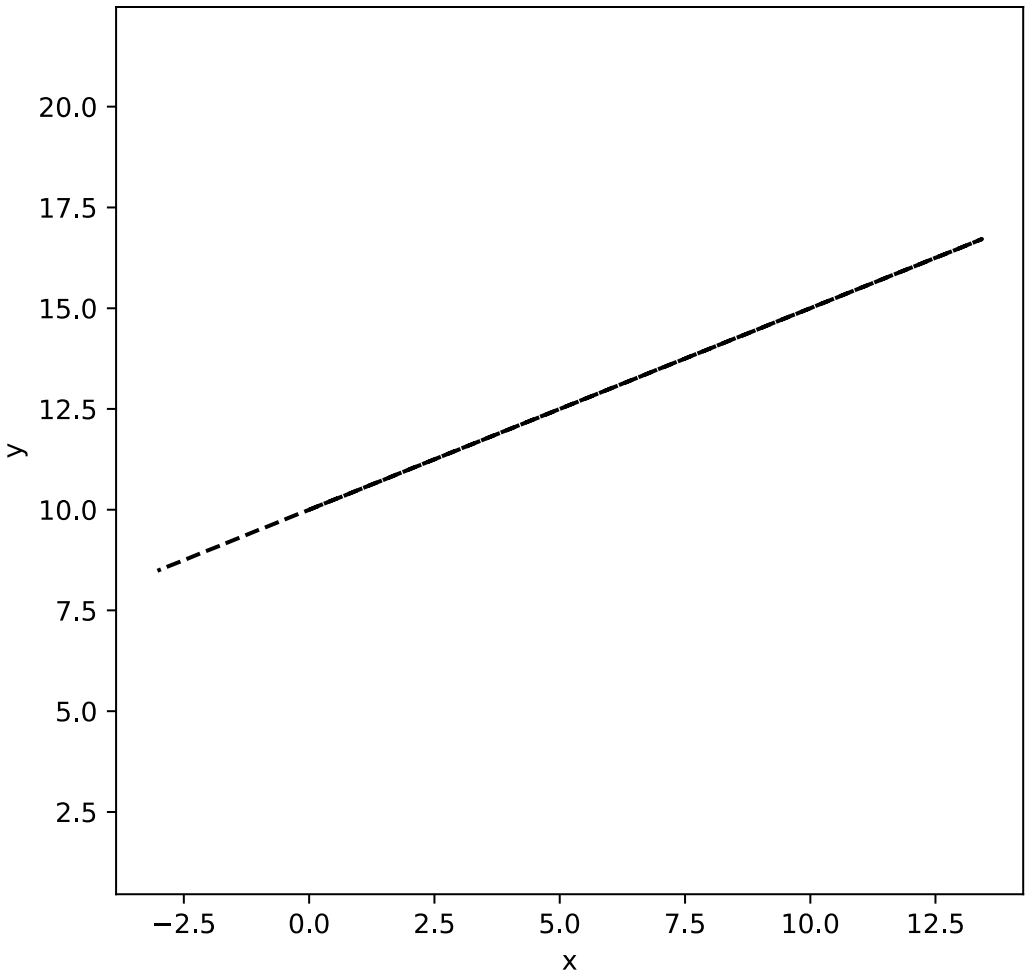
Consider now a simple regression with one predictor X and one observed variable Y



$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$$

Performance metrics -- Regression

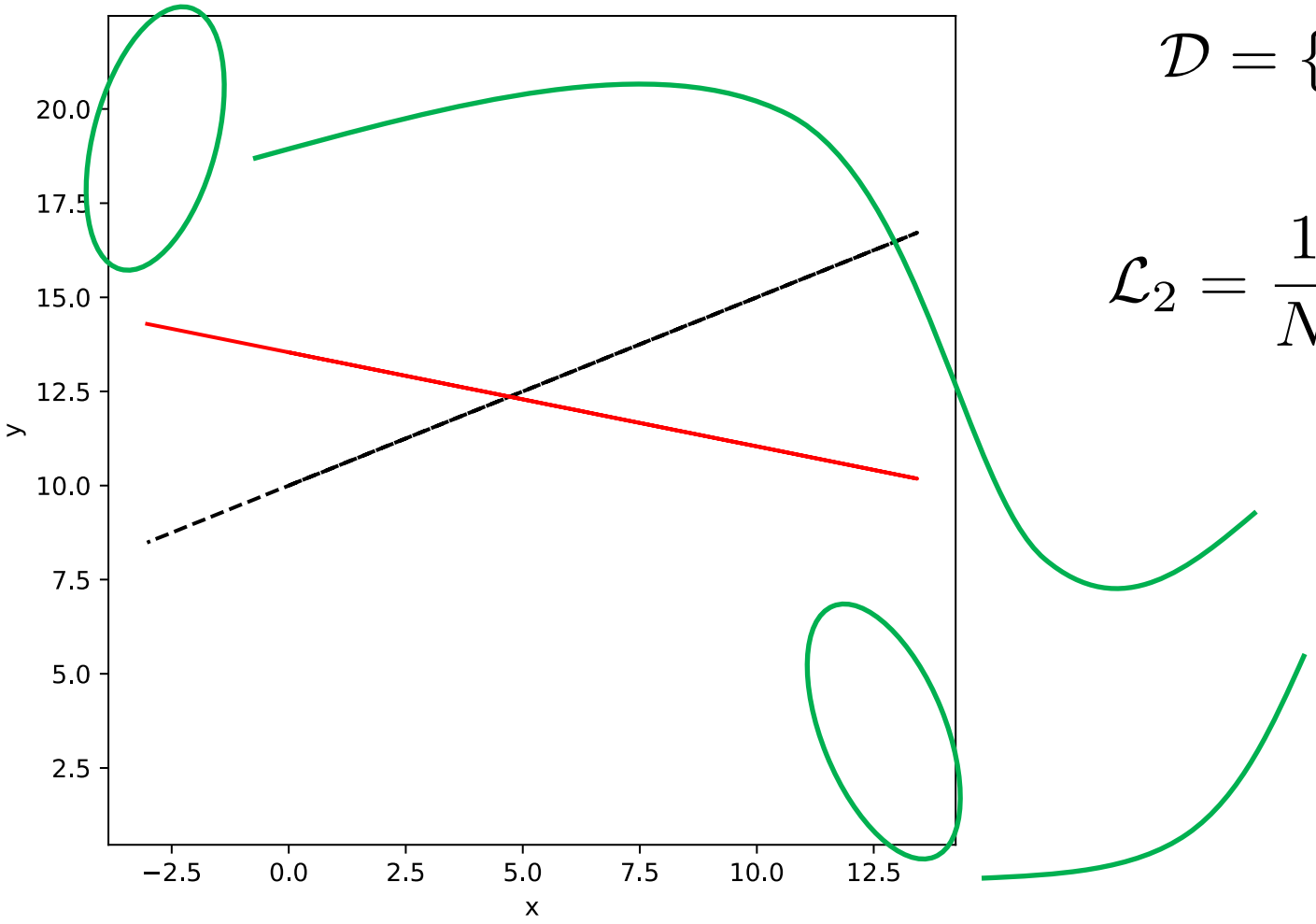
Consider now a simple regression with one predictor X and one observed variable Y



$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$$

Performance metrics -- Regression

Consider now a simple regression with one predictor X and one observed variable Y



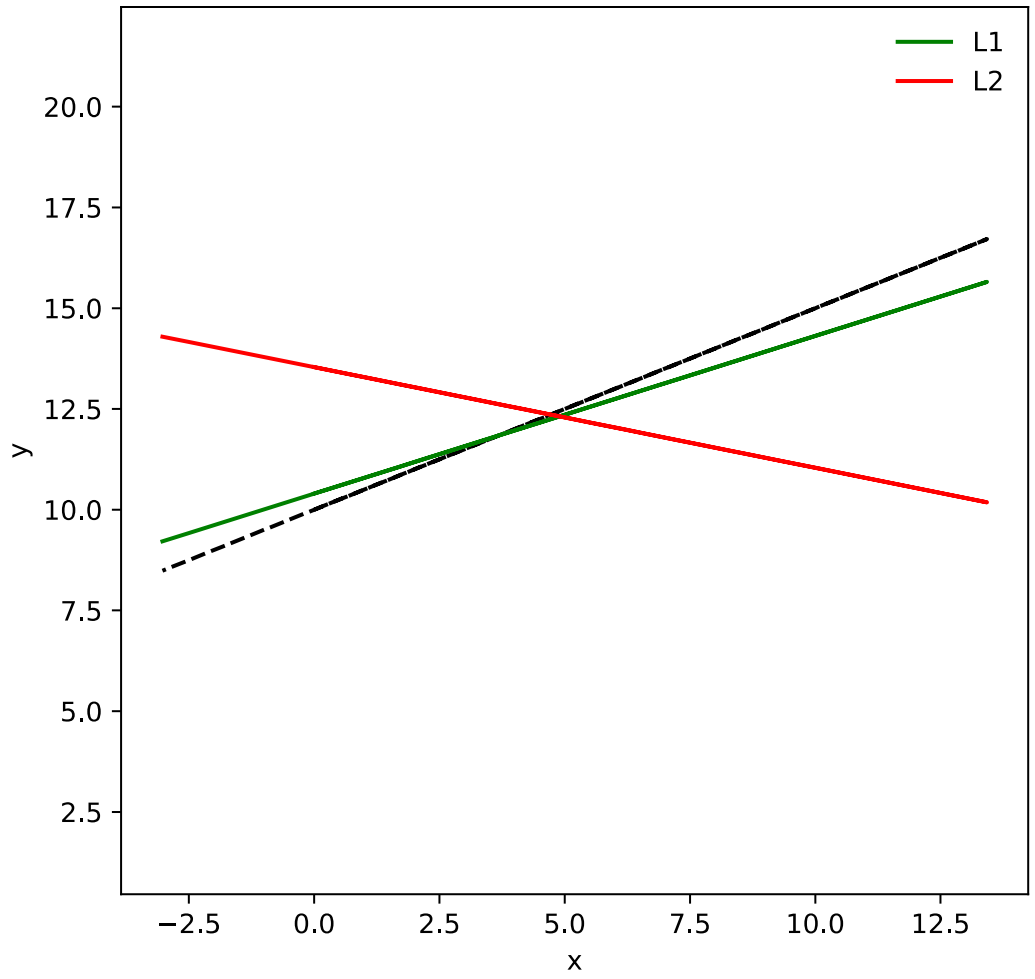
$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$$

$$\mathcal{L}_2 = \frac{1}{N} \sum_{i=1}^{N} \Big(y_i - (\beta_1 x_i + \beta_0) \Big)^2$$

The outliers contribute a lot to the minimization!

Performance metrics -- Regression

Consider now a simple regression with one predictor X and one observed variable Y



$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$$

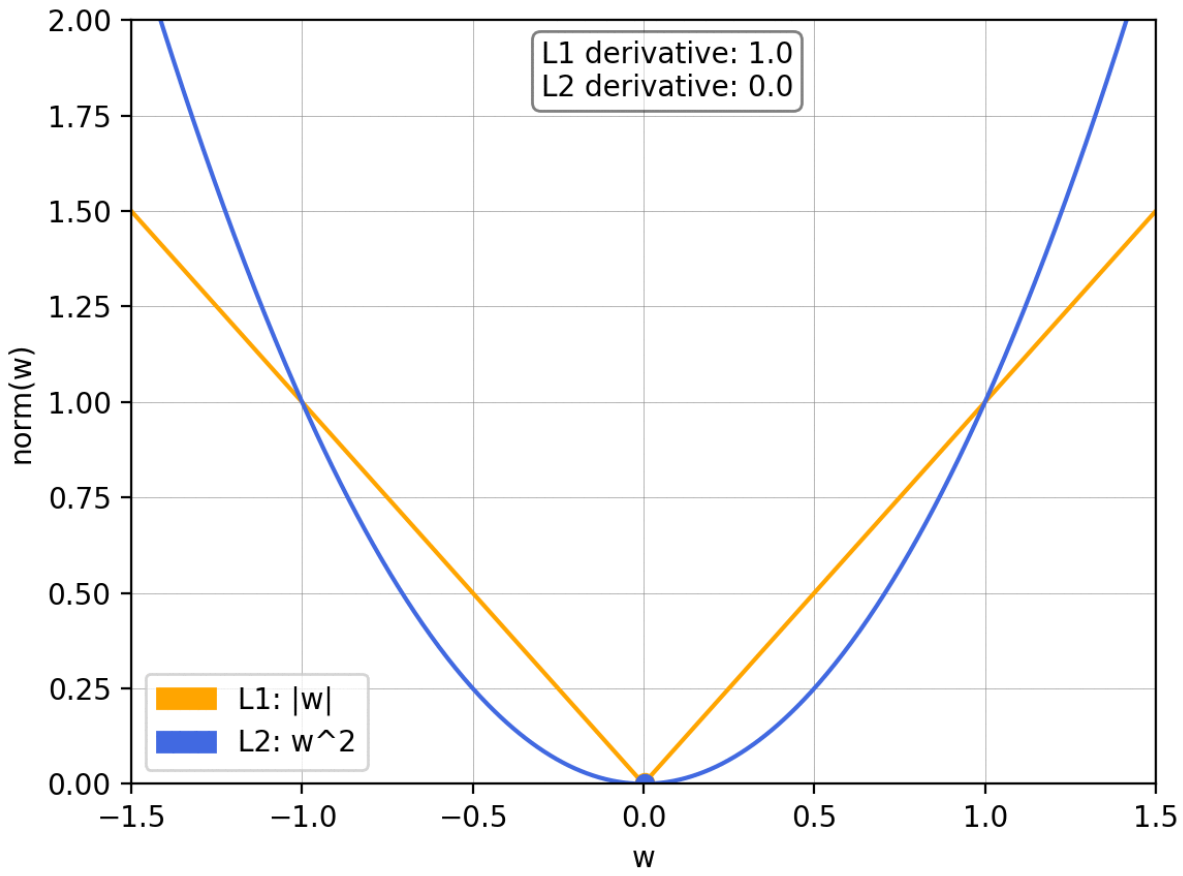
$$\mathcal{L}_2 = \frac{1}{N} \sum_{i=1}^{N} \Big(y_i - (\beta_1 x_i + \beta_0) \Big)^2$$

$$\downarrow$$

$$\mathcal{L}_1 = \frac{1}{N} \sum_{i=1}^{N} \Big| y_i - (\beta_1 x_i + \beta_0) \Big|$$

Performance metrics -- Regression

Consider now a simple regression with one predictor X and one observed variable Y



$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$$

$$\mathcal{L}_2 = \frac{1}{N} \sum_{i=1}^{N} \Big(y_i - (\beta_1 x_i + \beta_0) \Big)^2$$

$$\downarrow$$

$$\mathcal{L}_1 = \frac{1}{N} \sum_{i=1}^{N} \Big| y_i - (\beta_1 x_i + \beta_0) \Big|$$

Ref: https://medium.com/data-science

Performance metrics -- Regression

Consider now a simple regression with one predictor X and one observed variable Y

$$\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$$

$$\mathcal{L}_2 = \frac{1}{N} \sum_{i=1}^{N} \Big(y_i - (\beta_1 x_i + \beta_0) \Big)^2$$

```
model = LinearRegression()
model.fit(X, y)
y_pred_L2 = model.predict(X)
```

$$\mathcal{L}_1 = \frac{1}{N} \sum_{i=1}^{N} \Big| y_i - (\beta_1 x_i + \beta_0) \Big|$$

```
model = QuantileRegressor(quantile=0.5, alpha=0.0)
model.fit(X, y)
y_pred_L1 = model.predict(X)
```

Why take the QuantileRegressor ?

Performance metrics -- Regression

There are many other metrics for regression, such as

$$\mathcal{L}_2 = \frac{1}{N}\sum_{i=1}^{N}\Big(y_i - (\beta_1 x_i + \beta_0)\Big)^2$$

(ensure a scale-free property to the metric)

$$\mathcal{L}_1 = \frac{1}{N}\sum_{i=1}^{N}\Big|y_i - (\beta_1 x_i + \beta_0)\Big| \longrightarrow \mathcal{L}_{\text{MAPE}} = \frac{1}{N}\sum_{i=1}^{N}\Big|y_i - (\beta_1 x_i + \beta_0)\Big| \times \frac{1}{|y_i|}$$

$$\mathcal{L}_{\text{MSLE}} = \frac{1}{N}\sum_{i=1}^{N}\Big(\log(1+y_i) - \log(1+\beta_0+\beta_1 x_i)\Big)^2$$

(this metric penalizes an under-prediction <u>more</u> than an over-prediction)