Automated Machine Learning & Causality

Carlos Quintanilla

2023-08-31

Motivación

- Pero primero agradecimiento
- De qué hablar?
- Conversación con Hamzah Haji en Panamá
- Actualización del Curso de Data Mining
- Cómo ha cambiado en el último año ?

Contenidos

- Automated Machine Learning
- Causalidad y Machine Learning

La Idea Fundamental de AutoML

- Permitirle a alguien con *conocimientos básicos* de Estadística o de Data Mining correr modelos *altamente competitivos*
- Pueden pensar en esto como un proceso de *democratización* de Data Science
- Cualquiera puede usarla, no solo los expertos

La Idea Fundamental de AutoML

- Qué quiero decir por *conocimientos básicos*?
- Alguien que pueda correr una regresión lineal
- … en Python o en R.
- Qué quiere decir *altamente competitivos*?
- Un modelo con un error más bajo del que jamás podría lograr un profesor de INCAE (e.g. CQ)

Pequeña digresión

- Anécdota sobre uso de R en INCAE
- Circa 1999-2000

Pequeña digresión

Juramento personal

- No voy a volver a usar R jamás de los jamases
- Voy a usar StatTools/Excel, Gretl, Jamovi, RapidMiner
- y en Noviembre de 2022 ocurrió algo maravilloso en el mundo de la tecnología …

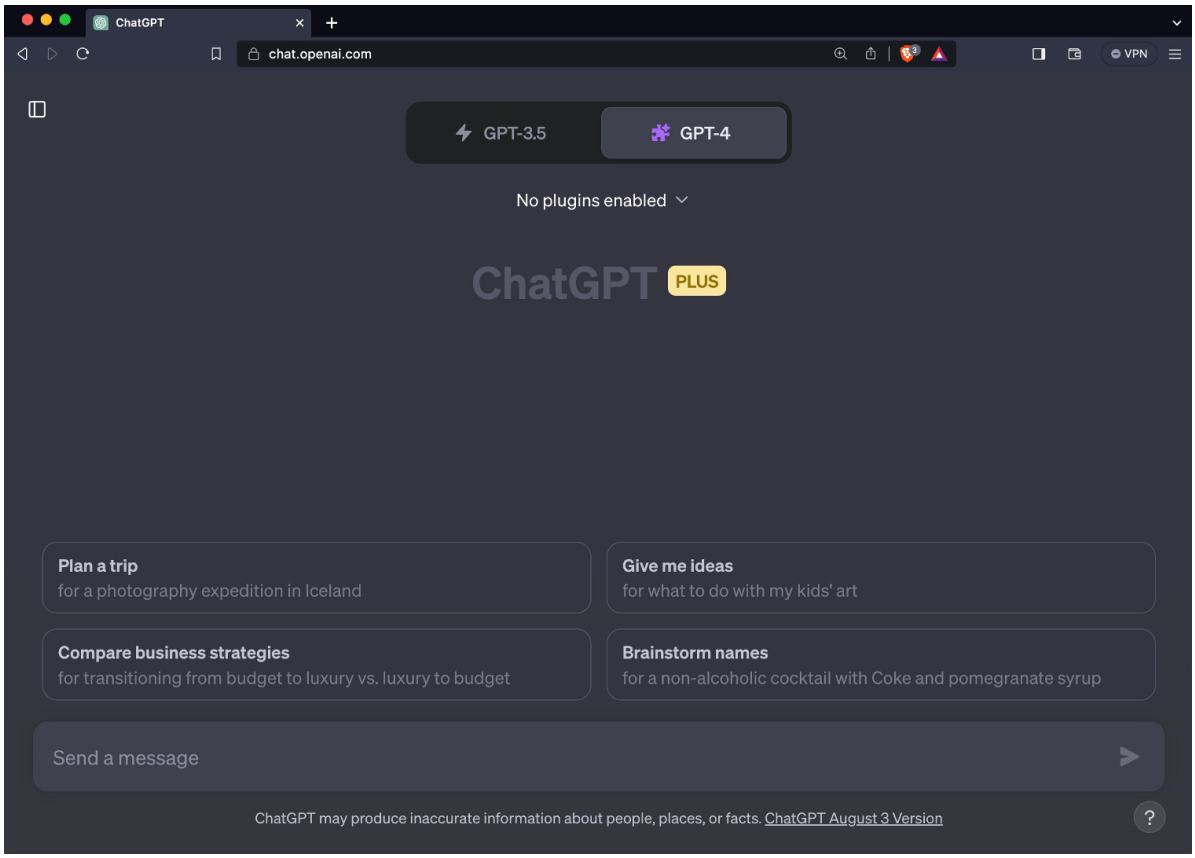
TidyModels Playlist

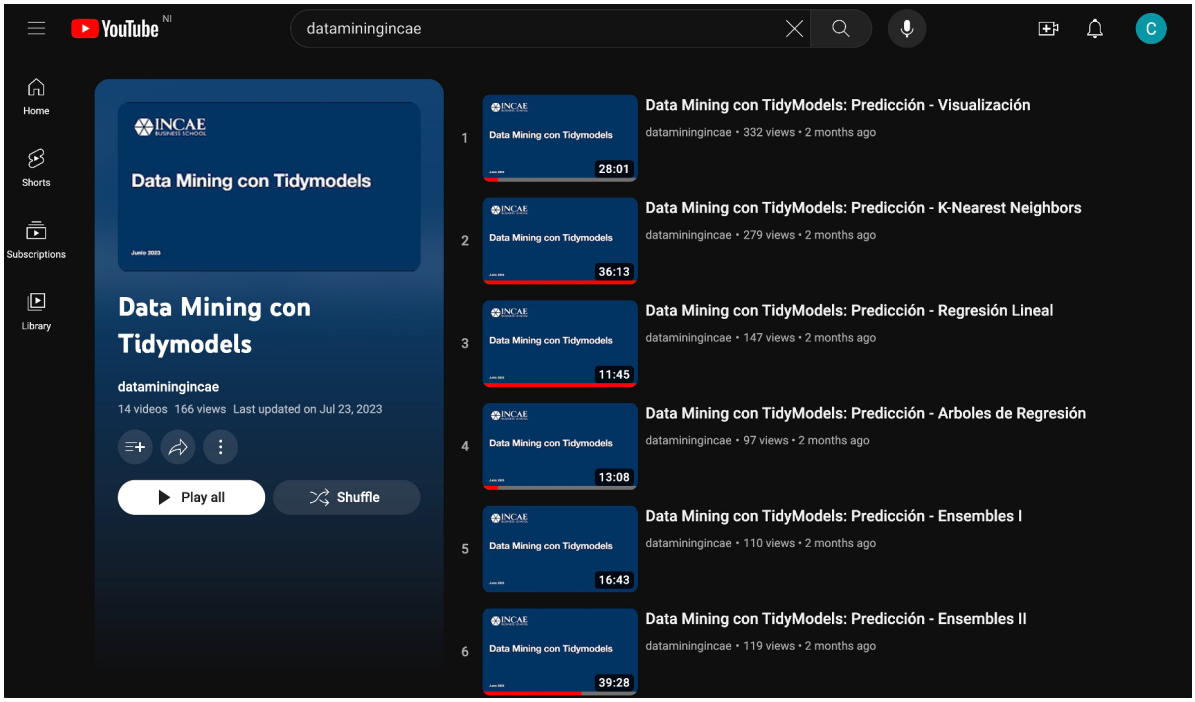
https://shorturl.at/pqIVZ

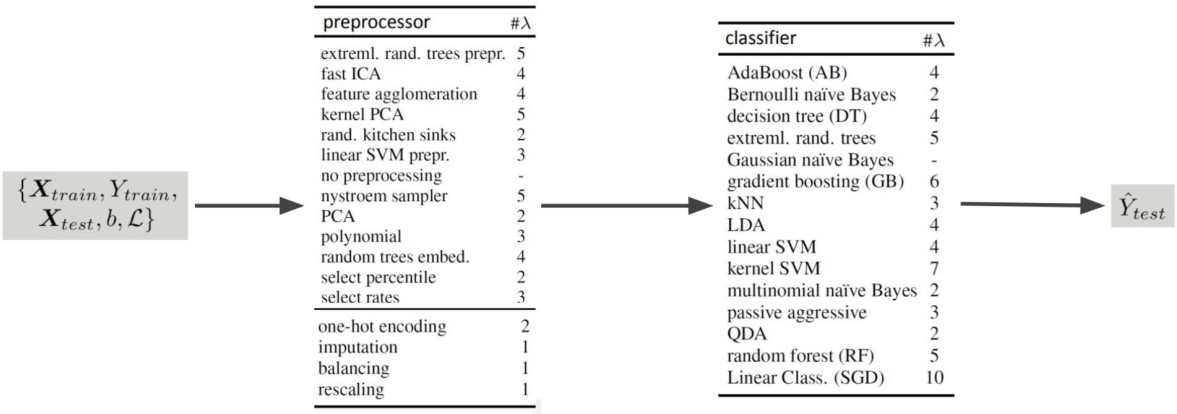
Auto-ML

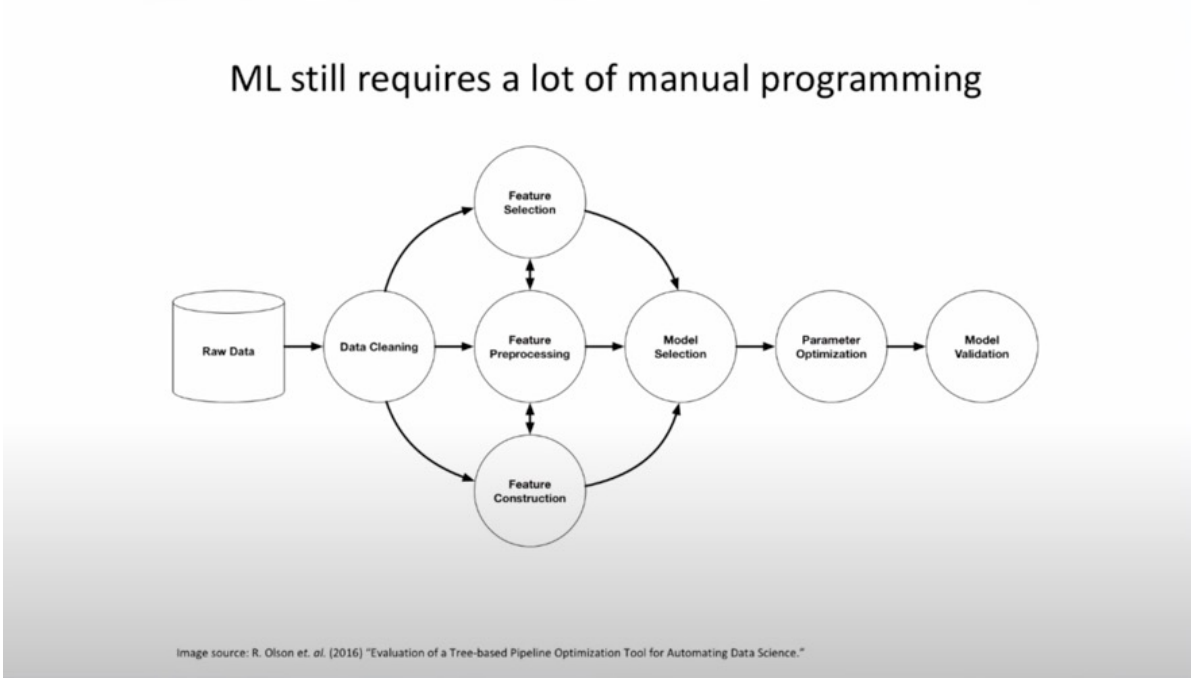
Auto-ML

- Definición: “Automated machine learning, also referred to as automated ML or AutoML, is the process of automating the time-consuming, iterative tasks of machine learning model development.”
- “It allows data scientists, analysts, and developers to build ML models with high scale, efficiency, and productivity all while sustaining model quality.”


ChatGPT
chat.openai.com
VPN
GPT-3.5
GPT-4
No plugins enabled
ChatGPT PLUS
Plan a trip
for a photography expedition in Iceland
Give me ideas
for what to do with my kids' art
Compare business strategies
for transitioning from budget to luxury vs. luxury to budget
Brainstorm names
for a non-alcoholic cocktail with Coke and pomegranate syrup
Send a message
ChatGPT may produce inaccurate information about people, places, or facts. ChatGPT August 3 Version
?


YouTube
datamineringincae
Home
Shorts
Subscriptions
Library
INCAE BUSINESS SCHOOL
Data Mining con Tidymodels
Junio 2023
Data Mining con Tidymodels
datamineringincae
14 videos 166 views Last updated on Jul 23, 2023
Play all
Shuffle
1 Data Mining con TidyModels: Predicción - Visualización
datamineringincae • 332 views • 2 months ago
28:01
2 Data Mining con TidyModels: Predicción - K-Nearest Neighbors
datamineringincae • 279 views • 2 months ago
36:13
3 Data Mining con TidyModels: Predicción - Regresión Lineal
datamineringincae • 147 views • 2 months ago
11:45
4 Data Mining con TidyModels: Predicción - Arboles de Regresión
datamineringincae • 97 views • 2 months ago
13:08
5 Data Mining con TidyModels: Predicción - Ensembles I
datamineringincae • 110 views • 2 months ago
16:43
6 Data Mining con TidyModels: Predicción - Ensembles II
datamineringincae • 119 views • 2 months ago
39:28


$\{X_{train}, Y_{train}, X_{test}, b, \mathcal{L}\}$
preprocessor #λ
extreml. rand. trees prepr. 5
fast ICA 4
feature agglomeration 4
kernel PCA 5
rand. kitchen sinks 2
linear SVM prepr. 3
no preprocessing -
nystroem sampler 5
PCA 2
polynomial 3
random trees embed. 4
select percentile 2
select rates 3
one-hot encoding 2
imputation 1
balancing 1
rescaling 1
classifier #λ
AdaBoost (AB) 4
Bernoulli naïve Bayes 2
decision tree (DT) 4
extreml. rand. trees 5
Gaussian naïve Bayes -
gradient boosting (GB) 6
kNN 3
LDA 4
linear SVM 4
kernel SVM 7
multinomial naïve Bayes 2
passive aggressive 3
QDA 2
random forest (RF) 5
Linear Class. (SGD) 10
$\hat{Y}_{test}$



Componentes

- Hyperparameter Optimization
- Meta-Learning (Warm-start)
- Ensembling

Hyparameter Optimization

- La mayoría de los algoritmos que usamos tienen muchos hyper-parametros que optimizar
- Randall Olson: "Los valores default típicamente no son óptimos"

HPO: Grids

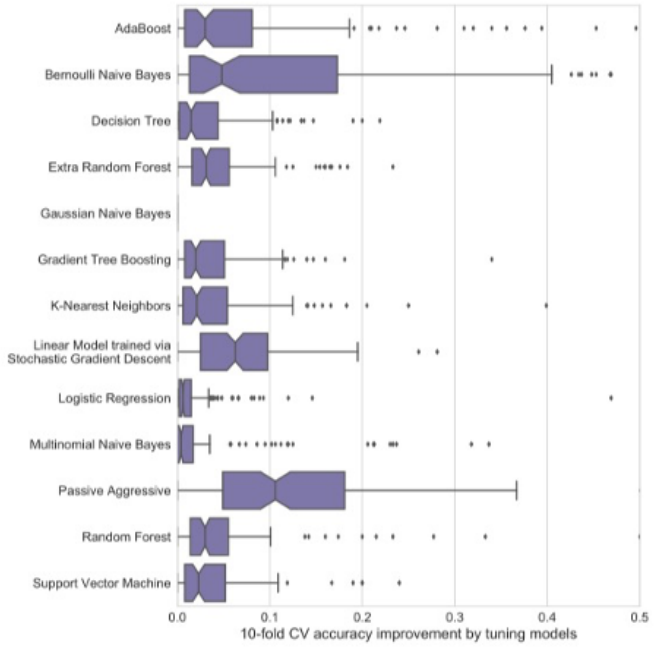
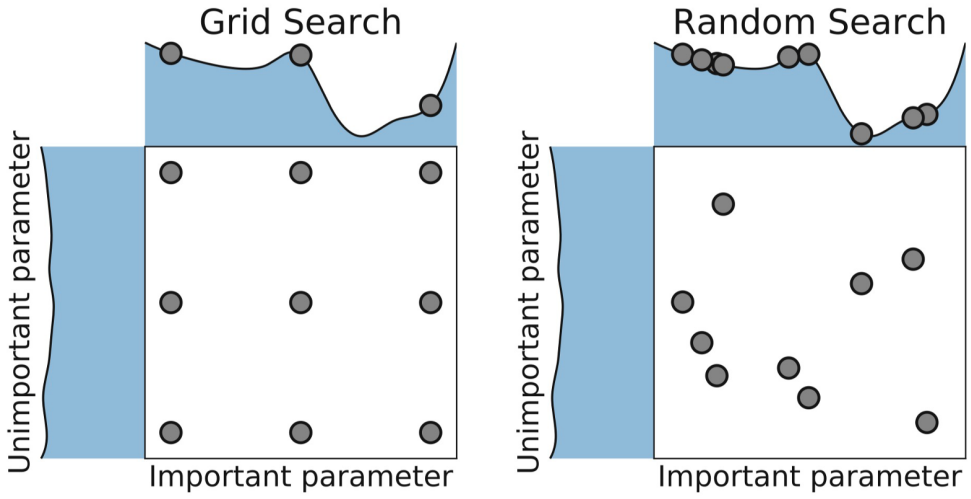


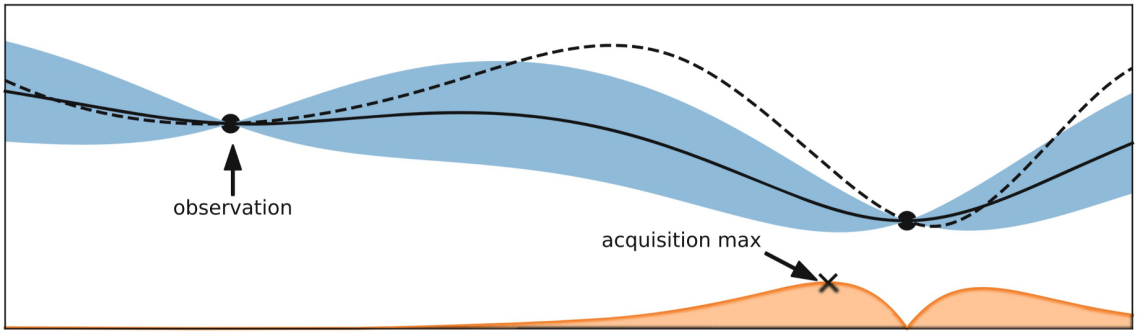
Fig. 3. Improvement in 10-fold CV accuracy by tuning each ML algorithm's parameters instead of using the default parameters from scikit-learn.



Xgboost

Hyperparameter	Lower bound	Upper bound	Result
(Probst et al. 2019a), several applications, 38 data sets			
`nrounds`	1	5000	920.7 to 4847.15 *
`eta`	2^{-10}	2^{0}	0.002 to 0.445 *
`subsample`	0.1	1	0.545 to 0.964 *
`maxdepth` x	1	15	2.6 to 14 *
`min_child_weight`	2^{0}	2^{7}	1.061 to 7.502 *
`colsample_bytree`	0	1	0.334 to 0.922 *
`lambda`	2^{-10}	2^{10}	0.004 to 29.755 *
`alpha`	2^{-10}	2^{10}	0.002 to 6.105 *

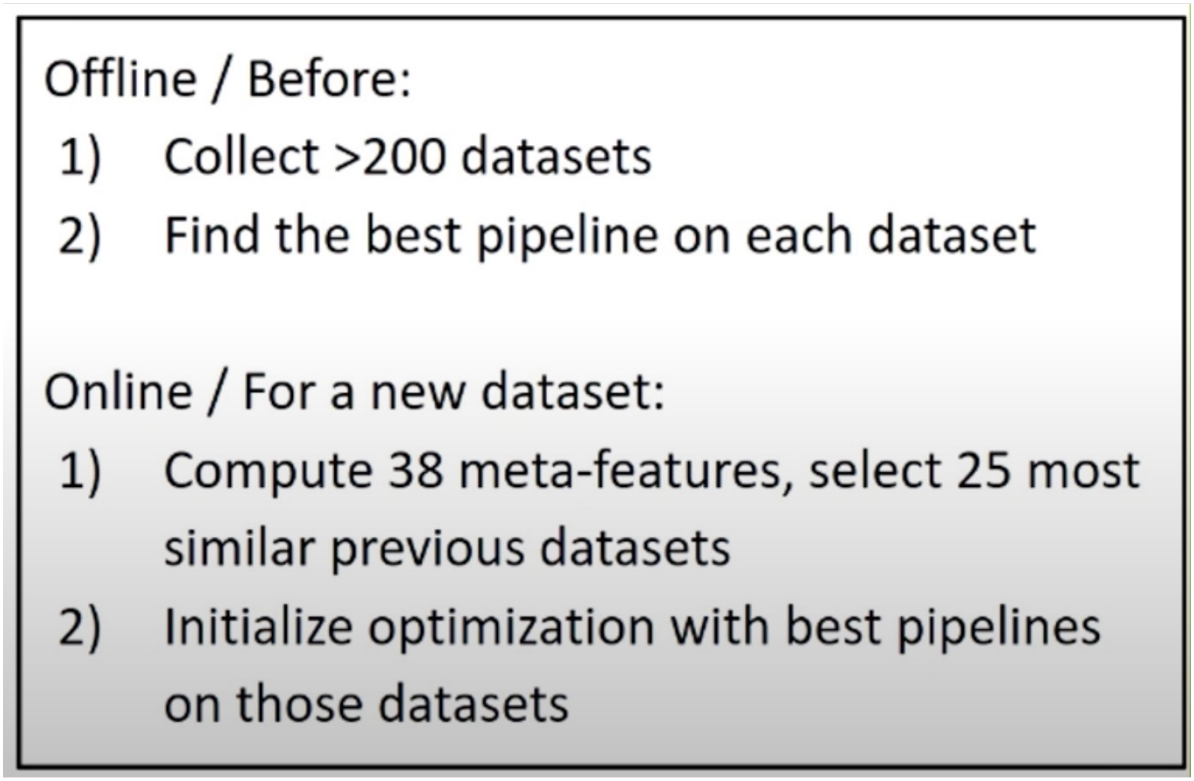
HPO: Bayesian Optimization



Meta-Learning

- La idea aquí es usar la experiencia que tenemos para optimizar distintos algoritmos
- Basandonos en las características del problema que queremos resolver, de la hoja de datos a mano
- No partir de cero

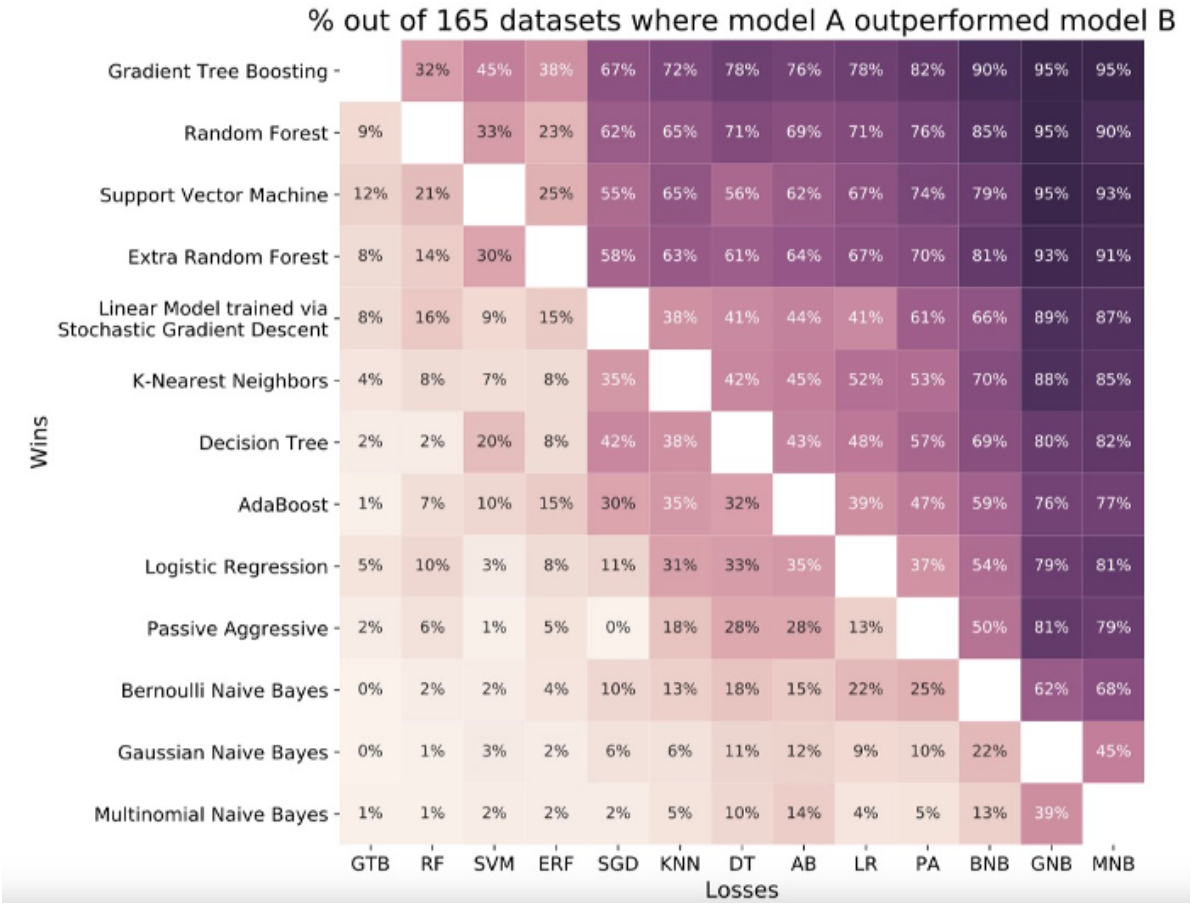
Meta-Learning (AutoSklearn)



Ensembling

- No hay algoritmo perfecto

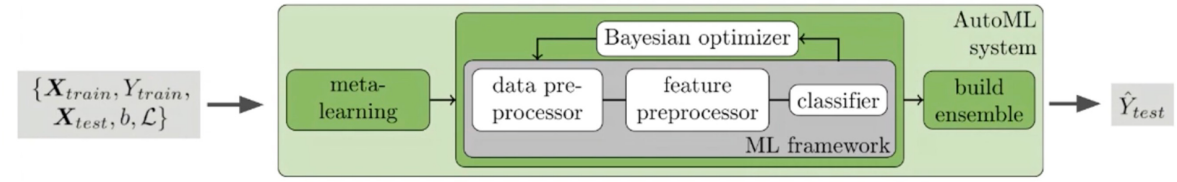
No-Free Lunch


% out of 165 datasets where model A outperformed model B
Wins
Losses
Gradient Tree Boosting 32% 45% 38% 67% 72% 78% 76% 78% 82% 90% 95% 95%
Random Forest 9% 33% 23% 62% 65% 71% 69% 71% 76% 85% 95% 90%
Support Vector Machine 12% 21% 25% 55% 65% 56% 62% 67% 74% 79% 95% 93%
Extra Random Forest 8% 14% 30% 58% 63% 61% 64% 67% 70% 81% 93% 91%
Linear Model trained via Stochastic Gradient Descent 8% 16% 9% 15% 38% 41% 44% 41% 61% 66% 89% 87%
K-Nearest Neighbors 4% 8% 7% 8% 35% 42% 45% 52% 53% 70% 88% 85%
Decision Tree 2% 2% 20% 8% 42% 38% 43% 48% 57% 69% 80% 82%
AdaBoost 1% 7% 10% 15% 30% 35% 32% 39% 47% 59% 76% 77%
Logistic Regression 5% 10% 3% 8% 11% 31% 33% 35% 37% 54% 79% 81%
Passive Aggressive 2% 6% 1% 5% 0% 18% 28% 28% 13% 50% 81% 79%
Bernoulli Naive Bayes 0% 2% 2% 4% 10% 13% 18% 15% 22% 25% 62% 68%
Gaussian Naive Bayes 0% 1% 3% 2% 6% 6% 11% 12% 9% 10% 22% 45%
Multinomial Naive Bayes 1% 1% 2% 2% 2% 5% 10% 14% 4% 5% 13% 39%
GTB RF SVM ERF SGD KNN DT AB LR PA BNB GNB MNB

Ensembling

- Después de optimizar los hyperparametros, combinamos todos los modelos
- Puede ser algo tan trivial como una combinación lineal obtenida con una regresión lineal
- Puede ser algo mas avanzado como Lasso

AutoSklearn 1.0



Ventajas de Auto-ML (Olson et al, 2018)

- Mejora notable del desempeño de diferentes algoritmos una vez que los ajustamos correctamente
- En manos de un novato, como ustedes o como yo mismo a veces, super-performers como SVM y Xgboost tienen un desempeño mediocre

---

Ventajas

- No Free-Lunch Theorem (hay que probarlos todos)
- Tedioso, error-prone (ganancias en eficiencia extraordinarios)
- Transparencia/Reproducibilidad

Table 1: Problem characteristics and performance comparisons.

Applications	#training data	#testing data	#features	#classes	Accuracy by users	Accuracy by our procedure
Astroparticle[1]	3,089	4,000	4	2	75.2%	96.9%
Bioinformatics[2]	391	0[4]	20	3	36%	85.2%
Vehicle[3]	1,243	41	21	2	4.88%	87.8%

Ejemplos Auto-ML

- H2o automl (R/Python - Open Source/Commercial)
- Amazon's AutoGluon (Python - Open Source)
- Microsoft's Flaml (Python - Open Source)
- Uber's Ludwig (Python - Open Source)
- JADBio (Just Add Data - Comercial)
- DataRobot (Comercial)

---

https://openml.github.io/automlbenchmark/results.html

Benchmarks

https://openml.github.io/automlbenchmark/frameworks.html

Benchmarks

- El ganador fue AutoGluon, pero casi todos los demas muy cerca
- Dejen mostrales un par de ejemplos usando AutoGluon
- Recuerden la promesa de AutoML: Permitirle a alguien con *conocimientos básicos* de Estadística o de Data Mining correr modelos *altamente competitivos*

Journal of Machine Learning Research 1 (2000) 1-48 Submitted 4/00; Published 10/00

AMLB: an AutoML Benchmark

Pieter Gijsbers[1] P.GIJSBERS@TUE.NL
Marcos L. P. Bueno[1] M.L.DE.PAULA.BUENO@TUE.NL
Stefan Coors[2] STEFAN.COORS@STAT.UNI-MUENCHEN.DE
Erin LeDell[3] ERIN@H2O.AI
Sébastien Poirier[3] SEBASTIEN@H2O.AI
Janek Thomas[2] JANEK.THOMAS@STAT.UNI-MUENCHEN.DE
Bernd Bischl[2] BERND.BISCHL@STAT.UNI-MUENCHEN.DE
Joaquin Vanschoren[1] J.VANSCHOREN@TUE.NL

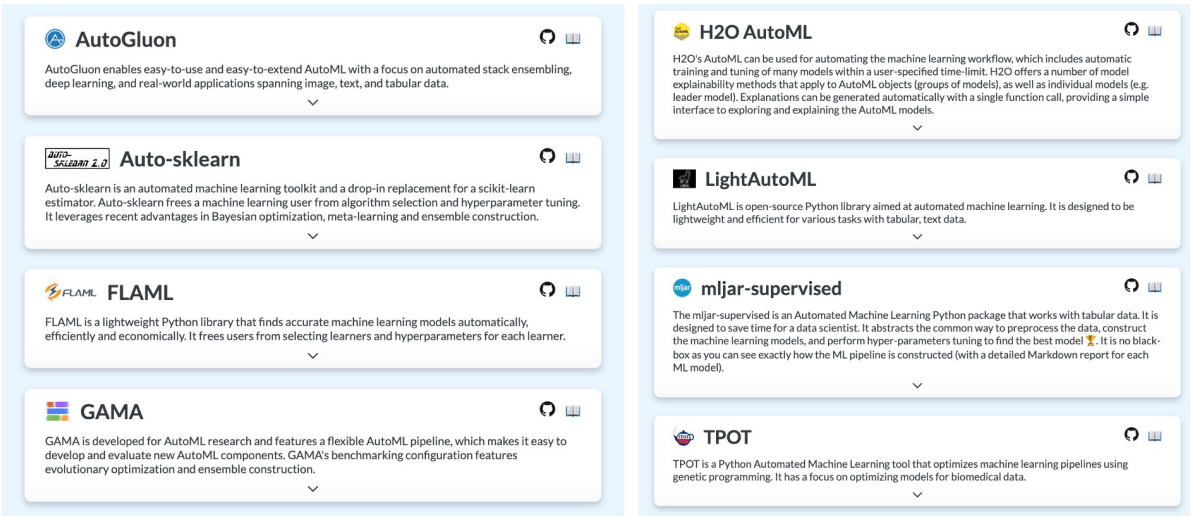
[1] Eindhoven University of Technology, Eindhoven, The Netherlands
[2] Ludwig Maximilian University of Munich, Munich, Germany
[3] H2O.ai, Mountain View, CA, United States

Editor: TBD

Abstract

Comparing different AutoML frameworks is notoriously challenging and often done incorrectly. We introduce an open and extensible benchmark that follows best practices and avoids common mistakes when comparing AutoML frameworks. We conduct a thorough comparison of 9 well-known AutoML frameworks across 71 classification and 33 regression tasks. The differences between the AutoML frameworks are explored with a multi-faceted analysis, evaluating model accuracy, its trade-offs with inference time, and framework failures. We also use Bradley-Terry trees to discover subsets of tasks where the relative AutoML framework rankings differ. The benchmark comes with an open-source tool that integrates with many AutoML frameworks and automates the empirical evaluation process end-to-end: from framework installation and resource allocation to in-depth evaluation. The benchmark uses public data sets, can be easily extended with other AutoML frameworks and tasks, and has a website with up-to-date results.

Keywords: open source, benchmark, automated machine learning, automl



Ejemplo 1 : Diamantes

- Vamos a pronosticar el precio de un diamantes como función de sus caracteristicas
- Esta es una hoja de datos benchmark en Data Mining
- Todo el mundo la conoce

Linear Regression

```
# Common Libraries
import numpy as np
import pandas as pd
import sklearn.metrics
from sklearn.model_selection import train_test_split

# Framework library
from sklearn.linear_model import LinearRegression

# Reading data
data = pd.read_csv('diamantes.csv')
target = data['price']
data = data.drop(['price'],axis=1)
```

```
# Specific preprocessing
data = pd.get_dummies(data)

# Divide the data into train and test
X_train, X_test, y_train, y_test = train_test_split(data, target, test_size= .2,random_sta

# Setup framework
modelo = LinearRegression()


# Fit model
modelo.fit(X_train, y_train)

# Get the predictions of the final ensemble
train_predictions = modelo.predict(X_train)
test_predictions = modelo.predict(X_test)

# Evaluate the predictions : R2 & RMSE
print("Train R2 score:", sklearn.metrics.r2_score(y_train, train_predictions))
print("Train RMSE score:", sklearn.metrics.mean_squared_error(y_train, train_predictions,
print("Test R2 score:", sklearn.metrics.r2_score(y_test, test_predictions))
print("Test RMSE score:", sklearn.metrics.mean_squared_error(y_test, test_predictions, squ
```

Linear Regression

```
# Common Libraries
import numpy as np
import pandas as pd
import sklearn.metrics
from sklearn.model_selection import train_test_split

# Framework library
from sklearn.linear_model import LinearRegression

# Reading data
data = pd.read_csv('diamantes.csv')
target = data['price']
```

```
data = data.drop(['price'],axis=1)

# Specific preprocessing
data = pd.get_dummies(data)

# Divide the data into train and test
X_train, X_test, y_train, y_test = train_test_split(data, target, test_size= .2,random_sta

# Setup framework
modelo = LinearRegression()

# Fit model
modelo.fit(X_train, y_train)

# Get the predictions of the final ensemble
train_predictions = modelo.predict(X_train)
test_predictions = modelo.predict(X_test)

# Evaluate the predictions : R2 & RMSE
print("Train R2 score:", sklearn.metrics.r2_score(y_train, train_predictions))
print("Train RMSE score:", sklearn.metrics.mean_squared_error(y_train, train_predictions,
print("Test R2 score:", sklearn.metrics.r2_score(y_test, test_predictions))
print("Test RMSE score:", sklearn.metrics.mean_squared_error(y_test, test_predictions, squ
```

AutoGluon

```
# Common libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
import sklearn.metrics

# Framework library
from autogluon.tabular import TabularDataset, TabularPredictor

# Reading data
data = pd.read_csv('/Users/carlosq/Desktop/automl/diamantes.csv')
```

```
#  Divide the data into train and test (including dependent variable; no separation)
train_data, test_data = train_test_split(data, test_size= .2,random_state=25)

# Further process dataset
train_data = TabularDataset(train_data)
test_data = TabularDataset(test_data)
y_train = train_data['price']
y_test = test_data['price']

# Set up framework
automl = TabularPredictor(label='price', path='agModels')

# Fit the model
automl.fit(train_data,time_limit=120,num_cpus=8)

# View the models found by auto-sklearn
automl.leaderboard(test_data, silent=True)

# Get the predictions of the final ensemble
train_predictions = automl.predict(train_data)
test_predictions = automl.predict(test_data)

# Evaluate the predictions : R2 & RMSE
print("Train R2 score:", sklearn.metrics.r2_score(y_train, train_predictions))
print("Train RMSE score:", sklearn.metrics.mean_squared_error(y_train, train_predictions,
print("Test R2 score:", sklearn.metrics.r2_score(y_test, test_predictions))
print("Test RMSE score:", sklearn.metrics.mean_squared_error(y_test, test_predictions, squ
```

AutoGluon

```
# Common libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
import sklearn.metrics

# Framework library
from autogluon.tabular import TabularDataset, TabularPredictor
```

```
9
10 # Reading data
11 data = pd.read_csv('/Users/carlosq/Desktop/automl/diamantes.csv')
12
13 #  Divide the data into train and test (including dependent variable; no separation)
14 train_data, test_data = train_test_split(data, test_size= .2,random_state=25)
15
16 # Further process dataset
17 train_data = TabularDataset(train_data)
18 test_data = TabularDataset(test_data)
19 y_train = train_data['price']
20 y_test = test_data['price']
21
22 # Set up framework
23 automl = TabularPredictor(label='price', path='agModels')
24
25 # Fit the model
26 automl.fit(train_data,time_limit=120,num_cpus=8)
27
28 # View the models found by auto-sklearn
29 automl.leaderboard(test_data, silent=True)
30
31 # Get the predictions of the final ensemble
32 train_predictions = automl.predict(train_data)
33 test_predictions = automl.predict(test_data)
34
35 # Evaluate the predictions : R2 & RMSE
36 print("Train R2 score:", sklearn.metrics.r2_score(y_train, train_predictions))
37 print("Train RMSE score:", sklearn.metrics.mean_squared_error(y_train, train_predictions,
38 print("Test R2 score:", sklearn.metrics.r2_score(y_test, test_predictions))
39 print("Test RMSE score:", sklearn.metrics.mean_squared_error(y_test, test_predictions, squ
```

Live demo

Ejemplo # 2: Colon Cancer

- Quería escoger un ejemplo que ilustrara como un novato podía usar AutoML y obtener resultados sorprendentes ...
- ... aunque no entendiera como los obtuvo
- Yo no se nada, pero ayudenme a decir NADA sobre Machine Learning aplicada a imagenes
- Neural Networks (Deep Learning) es el *go-to-guy*
- Yo he usado muy poco Neural Networks

Lung and Colon Cancer Histopathological Image Dataset (LC25000)

Andrew A. Borkowski, MD*[1,2], Marilyn M. Bui, MD, PhD[2,3], L. Brannon Thomas, MD, PhD[1,2], Catherine P. Wilson, MT[1], Lauren A. DeLand, RN[1], Stephen M. Mastorides, MD[1,2]

[1] Pathology and Laboratory Service, James A. Haley Veterans' Hospital, Tampa, Florida, USA
[2] Department of Pathology and Cell Biology, University of South Florida, Tampa, Florida, USA
[3] Department of Pathology and Analytic Microscope Core, Moffitt Cancer Center, Tampa, Florida, USA

*E-mail: andrew@usf.edu

Abstract

The field of Machine Learning, a subset of Artificial Intelligence, has led to remarkable advancements in many areas, including medicine. Machine Learning algorithms require large datasets to train computer models successfully. Although there are medical image datasets available, more image datasets are needed from a variety of medical entities, especially cancer pathology. Even more scarce are ML-ready image datasets. To address this need, we created an image dataset (LC25000) with 25,000 color images in 5 classes. Each class contains 5,000 images of the following histologic entities: colon adenocarcinoma, benign colonic tissue, lung adenocarcinoma, lung squamous cell carcinoma, and benign lung tissue. All images are de-identified, HIPAA compliant, validated, and freely available for download to AI researchers.

Keywords: LC25000, image dataset, machine learning, deep learning, medical imaging, cancer pathology

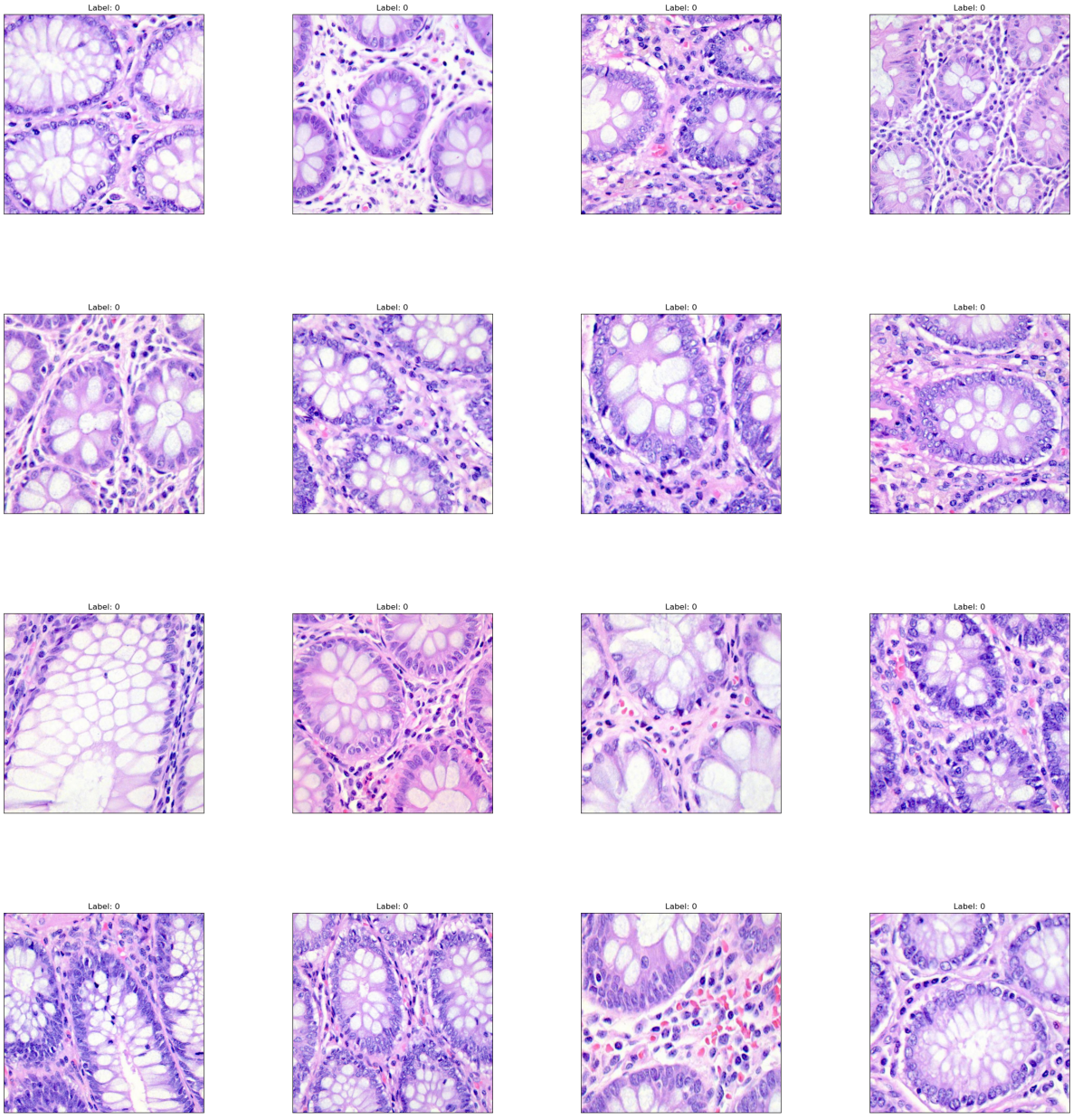
https://arxiv.org/abs/1912.12142

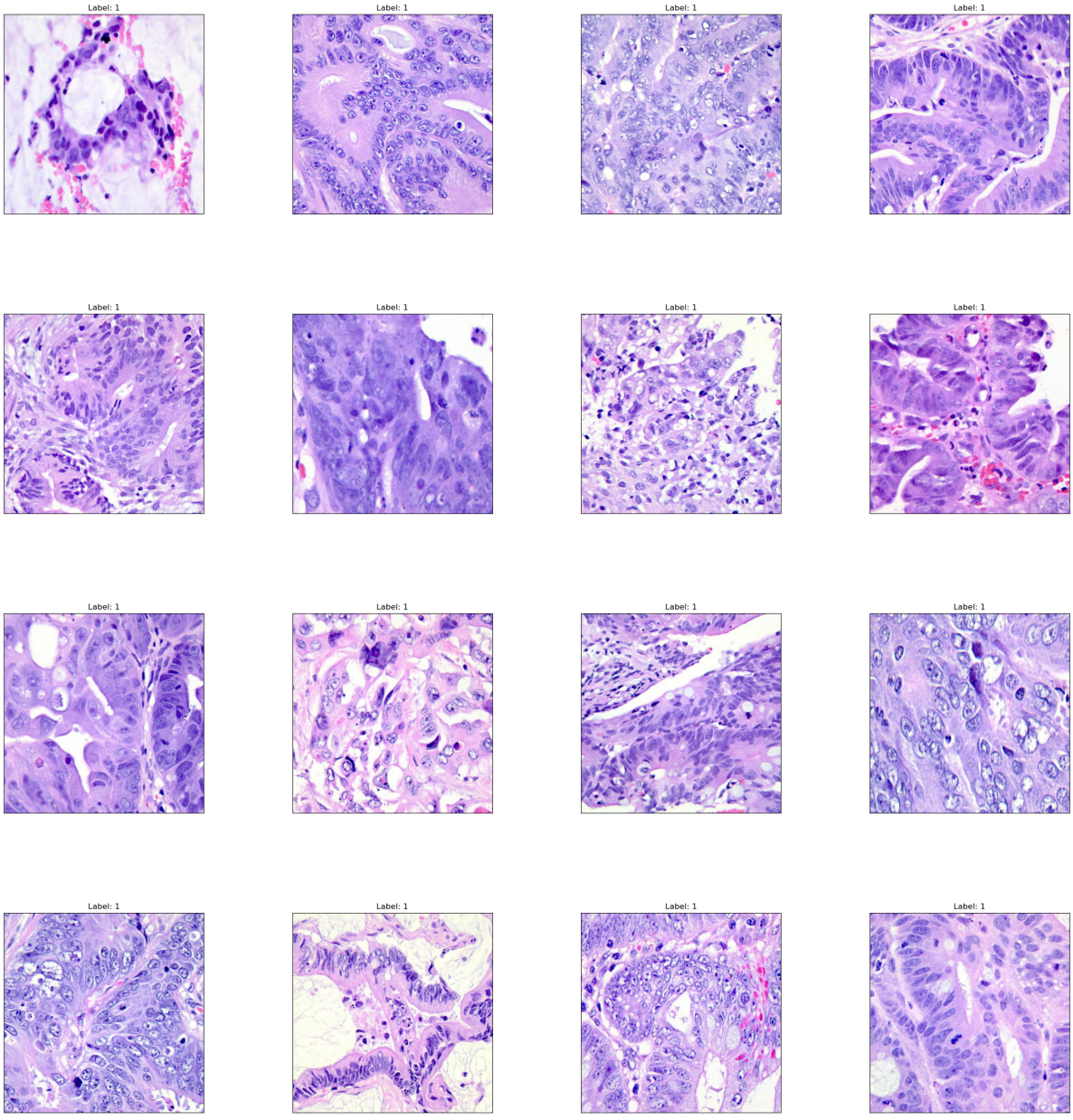
Benign

Cancer

AutoGluon

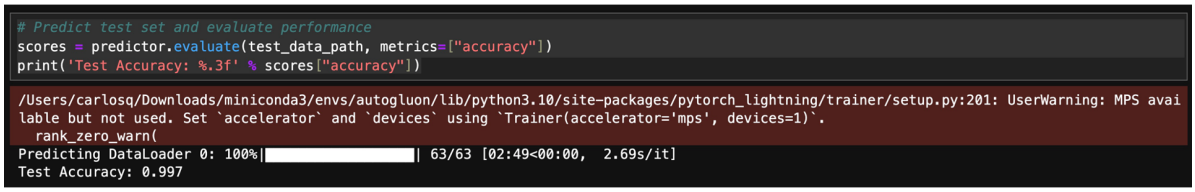
```
# Import relevant libraries

import pandas as pd
from sklearn.model_selection import train_test_split
from autogluon.multimodal import MultiModalPredictor

# Import 'dataset' (list of addresses for images and labels)

images = pd.read_csv('/Users/carlosq/Downloads/coloncancer/lista_imagenes.csv')

# Split data intro train and test sets

train_data_path, test_data_path = train_test_split(images,train_size=0.8,random_state=1234

# Set up model

predictor = MultiModalPredictor(
    label="label",
    path="/Users/carlosq/Downloads/coloncancer/resultados"
    )

# Fit the models. Time limit = 600 seconds

predictor.fit(train_data=train_data_path, time_limit=600)

# Predict test set and evaluate performance

scores = predictor.evaluate(test_data_path, metrics=["accuracy"])
```


Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0
Label: 0


Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1
Label: 1

```
print('Test Accuracy: %.3f' % scores["accuracy"])

# Predict test set and evaluate performance

scores = predictor.evaluate(test_data_path, metrics=["accuracy"])
print('Test Accuracy: %.3f' % scores["accuracy"])
```

El resultado

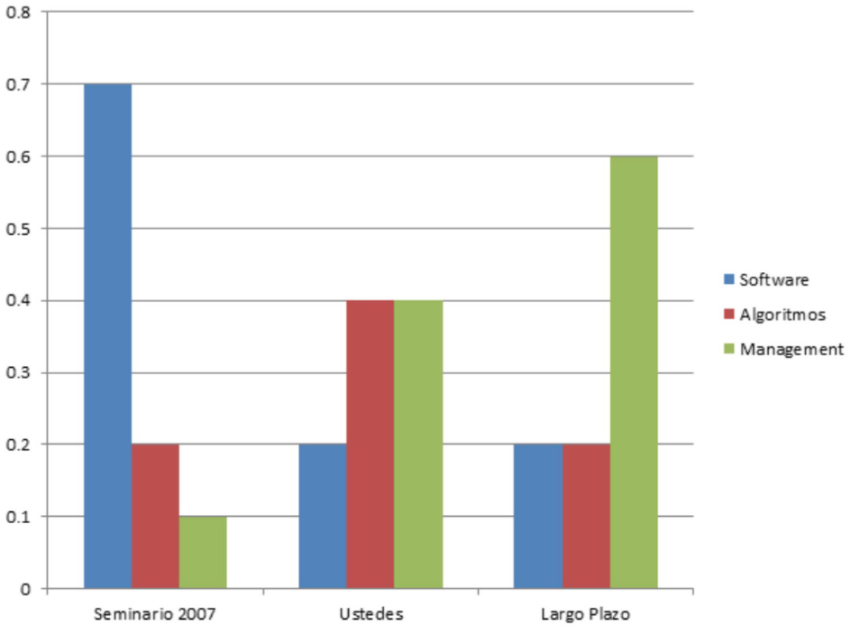


Dilema

- Estos resultados plantean un dilema importante para mi clase
- Qué hago con ella?
- No necesito 18 sesiones para enseñarles a los estudiantes como escribir esos dos programas que les he mostrado
- Dos consideraciones

Consideración # 1

- Se vale usar un algoritmo sin entenderlo
- Integral definida



Consideración # 2

Qué hice?

- Comenzando con el último curso que enseñé estructure el curso en tres partes:
- Modelos Básicos (lineal, knn, arboles, naive bayes)
- Modelos Avanzados (random forest, xgboosting, superlearners, auto-ml)
- Interpretabilidad. Fairness. Otras consideraciones éticas.

Causal Machine Learning

Otra digresión

- Cómo comence a enseñar causalidad en INCAE?
- Siempre tuve un par de sesiones en Métodos Cuantitativos
- Pero el esfuerzo en serio comenzo durante la pandemia

Otra digresión

Transición: Trabajo de Grupo # 4 (Julio 2023)

Instrucciones Trabajo de Grupo "Rate Optimization"

Ustedes trabajan para una institución financiera que concede préstamos en EE.UU. Todas las aplicaciones de los clientes se hacen en línea. Un ejemplo de esta industria en EE.UU. son los llamados **auto-loans.** El cliente llena un formulario online y recibe una respuesta de la institución en un tiempo breve (menos de 24 horas en promedio).

La institución debe decidir si acepta la aplicación y de aceptarla qué tasa de interés ofrecer al cliente. Una vez que el cliente recibe esta oferta decide si la acepta o no. En la hoja de datos (**rate0.csv**) que recibirán ustedes saben ya que ocurrió con todos los clientes.

Transición: Trabajo de Grupo # 4 (Julio 2023)

Transición: Trabajo de Grupo # 4 (Julio 2023)

Transición: Trabajo de Grupo # 4 (Julio 2023)

- Esta fue una asignación 'injusta' porque yo no les enseñe a los estudiantes cómo resolver un problema semejante
- Durante el curso les enseñe a pronosticar una variable (numérica o binomial) como función de una batería de variables mixtas
- Este trabajo les pide algo fundamentalmente **nuevo**
- Les pide analizar los efectos de un cambio en una variable sobre la que tenemos pleno control, la tasa de interés
- Es el análisis de una *intervención* de marketing
- Este no es un problema de predicción. Es un problema de causalidad.

Caso Clásico

- $$Y = T * \theta + \epsilon$$

- id: identificador único
- fico : un score crediticio en EE.UU. A mayor su valor, mejor el record crediticio.
- gender: genero del cliente hombre/mujer
- birthdate: fecha de nacimiento del cliente
- income: ingreso anual del cliente en U.S. $.
- zipcode: zipcode donde vive el cliente.
- state: Estado en EE.UU. donde vive el cliente.
- amount: monto del préstamo en U.S. $.
- term: plazo del préstamo en meses.
- datedue: fecha en que el préstamo debía pagarse.
- rate: tasa ofrecida al cliente
- accept: variable binaria igual a 1 si el cliente aceptó la oferta.
- default: variable binaria igual a 1 si el cliente no pago el préstamo (defaulted).

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	id	fico	gender	birthdate	income	zipcode	state	amount	term	datedue	rate	accept	default
2	173604	734	female	20/6/67	30900	90201	CA	20000	12	25/3/23	4.61223034	1	0
3	197081	765	female	21/8/72	26800	33606	FL	20000	12	18/10/22	7.31876414	1	0
4	197084	827	female	17/11/83	33900	10314	NY	20000	12	2/2/23	9.6058689	0	0
5	119789	638	female	14/11/74	19900	11373	NY	20000	12	23/4/23	9.44781899	1	0
6	122129	806	female	27/5/67	28900	33909	FL	20000	12	15/2/23	4.54123374	1	0
7	142546	792	male	14/11/86	33200	33909	FL	20000	12	2/3/23	18.1297313	0	0
8	183385	661	female	3/7/65	20500	75034	TX	20000	12	21/3/23	11.0319995	1	0
9	135964	595	female	23/11/93	18100	91331	CA	20000	12	30/1/23	10.4871656	1	0
10	192737	596	female	3/10/86	18400	77494	TX	20000	12	26/1/23	1.32075518	1	0

Para efectos prácticos ustedes pueden suponer que la hoja **rate0.csv** fue el resultado de un experimento conducido por el banco en el que la tasa de interés para cada cliente fue escogida al azar con el propósito de aprender como responden los clientes (de diversas características) a distintas ofertas.

Su Misión en este Trabajo

Todo lo que quiero de ustedes es una oferta de tasa de interés para cada cliente en las hojas **rate1.csv** y **rate2.csv** que corresponden a los clientes de las dos hojas de respuesta de Excel.

Si NO desean aceptar la aplicación del cliente "ofrezcan" una tasa de interés de -99. Las leyes contra usura recientemente aprobadas en los Estados donde opera la institución prohiben que las tasas de interés en este tipo de préstamos excedan 20%. La institución también tiene como política no ofrecer una tasa por debajo de 2%. Dentro de este rango, ustedes tienen plena libertad para seleccionar la tasa de interés. Puede ser uniforme para todos los clientes. Puede ser diferente para cada cliente.

- **Y** es la variable que nos interesa. Por ejemplo, compras.
- **T** es la intervención. Podría ser un cupón, un descuento, una llamada
- ϵ es un error.
- θ es el efecto. Aquí se supone constante.
- Este es el caso de un experimento aleatorio bien diseñado. El tratamiento, **T**, ha sido asignado completamente al azar.

- No necesitamos Machine Learning. Estadística 101 basta.

Variación Caso Clásico

- $$Y = T * \theta + g(X) + \epsilon$$
- **T** todavía es asignado al azar, pero
- La variable **Y** depende de la intervención, pero también de otras variables **X**
- **X** podría ser ingreso por ejemplo
- Típicamente la función *g( )* se asume lineal
- Podríamos seguir utilizando estadística básica para estimar el efecto θ
- Pero hay ganancias en eficiencia (menor variabilidad) de tomar en cuenta **X**

Unconfoundness

- $$Y = T * \theta + g(X) + \epsilon$$
- $$T = m(X) + \eta$$
- El tratamiento, T, ya no es asignado al azar
- Ahora depende de la variable **X**. El cupón se le envía con más frecuencia a los clientes de niveles bajos de ingreso (o viceversa)
- La segunda ecuación se conoce como *Propensity Score*. Mide que tan probable es que un cliente reciba la intervención (el tratamiento)

Unconfoundness

- Usualmente las funciones $g(\ )$ y $m(\ )$ se asumen lineales. La primera es una regresión lineal. La segunda una regresión logística.
- Y aquí esta potencialmente la ventaja de usar Machine Learning.
- Nada en la teoría nos dice que las funciones son lineales.
- Pero modelar $g(\ )$ y $m(\ )$ es un problema de predicción. Machine Learning destaca en ese tipo de tareas.
- Este tipo de modelos se llaman *Double Machine Learning Models*

Variación Unconfoundness

- $$Y = T * \theta(X) + g(X) + \epsilon$$
- $$T = m(X) + \eta$$
- Ahora el efecto depende de $\mathbf{X}$
- No todas los clientes responden de igual forma a una intervención. Los clientes con ingreso alto, por ejemplo, podrían ser poco susceptible a la intervención, pero los de bajo ingreso no.
- Esto se conoce como *efectos heterogeneos* y da lugar a la posiblidad de intervenciones *personalizadas*

EconML

```
# Load relevant libraries
import pandas as pd
from econml.dml import CausalForestDML

# Read dataset
datos = pd.read_csv('/Users/carlosq/Desktop/automl/masterminds/het1.csv')
y = datos['y']
T = datos['t']
X = datos.drop(['y','t'],axis=1)
```

```
# Set up the model and estimate
est = CausalForestDML(discrete_treatment=True)
est.fit(y, T, X=X, W=None)

# Average Treatment Effect
est.ate_

# Variable importance (to explain heterogeneity)
est.feature_importances_

# What is Theta for each individual in a test set
Xtest = pd.read_csv('/Users/carlosq/Desktop/automl/masterminds/het2.csv')
theta_test=est.effect(Xtest)
```

EconML

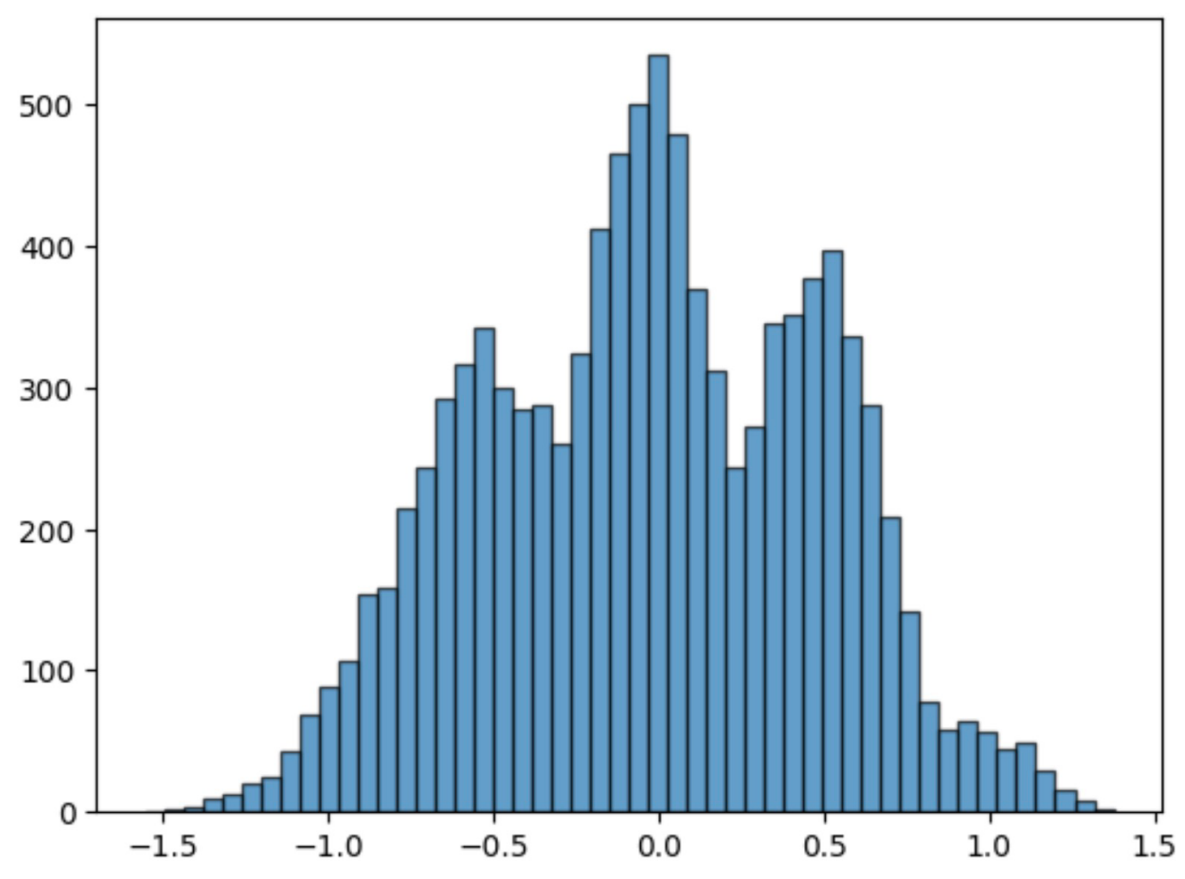
```
# Load relevant libraries
import pandas as pd
from econml.dml import CausalForestDML

# Read dataset
datos = pd.read_csv('/Users/carlosq/Desktop/automl/masterminds/het1.csv')
y = datos['y']
T = datos['t']
X = datos.drop(['y','t'],axis=1)

# Set up the model and estimate
est = CausalForestDML(discrete_treatment=True)
est.fit(y, T, X=X, W=None)

# Average Treatment Effect
est.ate_

# Variable importance (to explain heterogeneity)
est.feature_importances_

# What is Theta for each individual in a test set
Xtest = pd.read_csv('/Users/carlosq/Desktop/automl/masterminds/het2.csv')
```

```
23 theta_test=est.effect(Xtest)
```

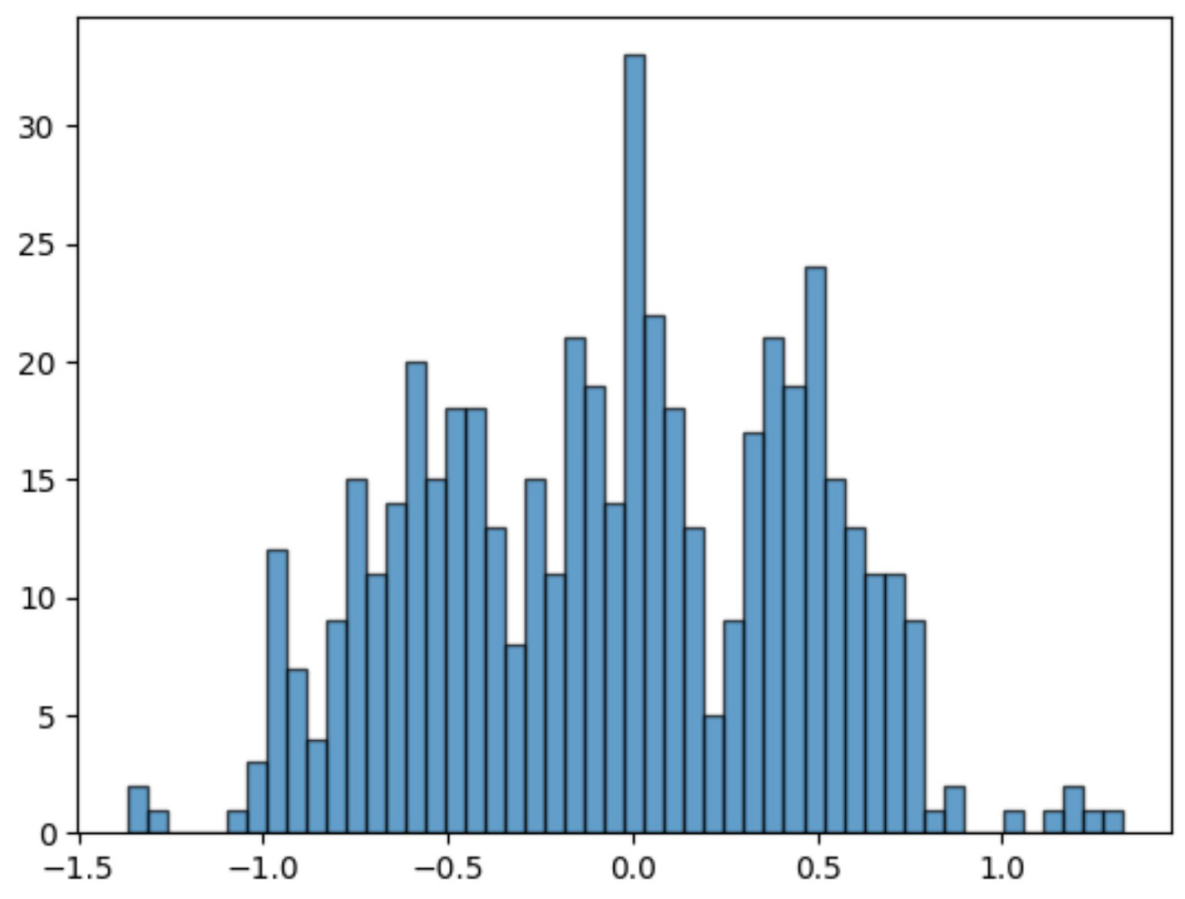
Average Treatment Effect

- est.ate_ = -0.00062109
- Efecto promedio es 0, pero queda la duda sobre la posibilidad de efectos heterogeneos
- est.feature_importances_
- [0.03 , 0.94, 0.00, 0.00, 0.01]

Efectos Heterogeneos (Training Set)



Efectos Heterogeneos (Test Set)



Algunas Referencias

- Automated Machine Learning. Frank Hutter et al.
- Hyperparameter Tuning for Machine and Deep Learning with R. Eva Bartz et al.
- AutoML - Automated Machine Learning. An Online Course. https://ki-campus.org/courses/automl-luh2021
- Machine Learning-based Causal Inference. https://bookdown.org/stanfordgsbsilab/ml-ci-tutorial/

Muchas gracias!