Minería de datos
Reto Kaggle

Integrantes: Mauricio Leiton
Denisse Orozco
Eunice Galvez

Objetivos

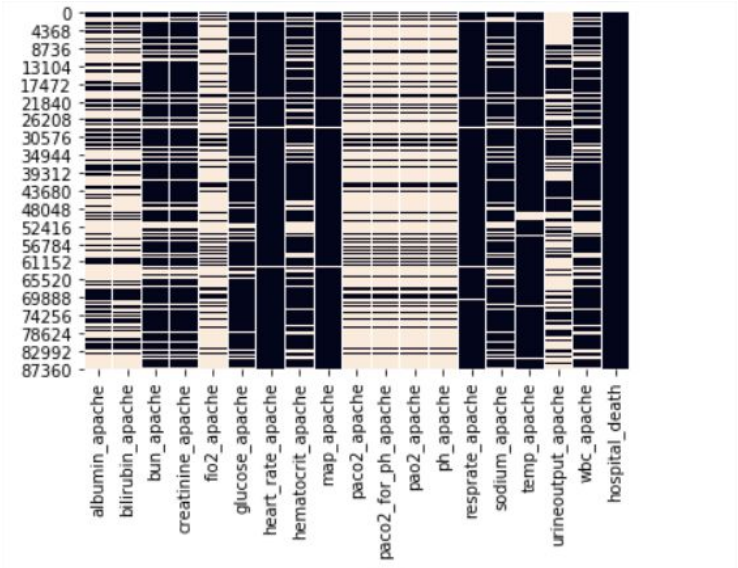
- Crear un modelo que usa la data de las de las primeras 24 horas de UCI (Unidad de Cuidados Intensivos) para predecir la supervivencia del paciente
- Aplicar y consolidar los conceptos aprendidos en el transcurso del curso minería de datos en casos reales
- Ayudar a los médicos a identificar patrones que puedan ocasionar la muerte de los pacientes
- Adquirir conocimientos que no están relacionados a nuestra área profesional.

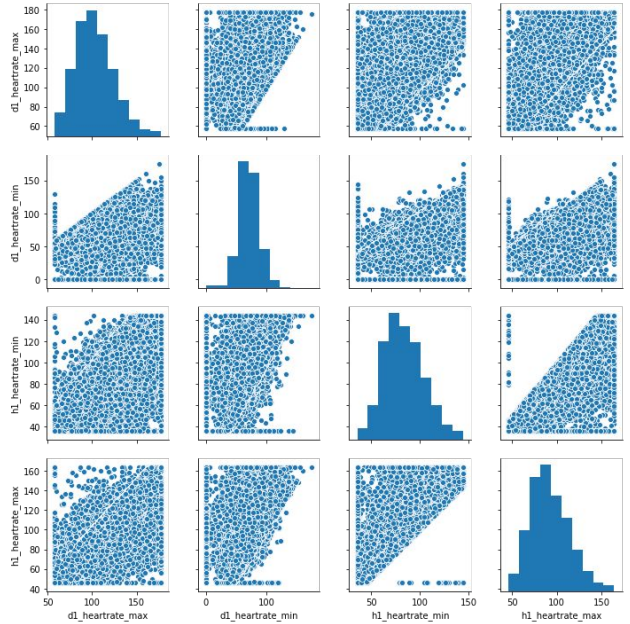
Dataset

- Instancias (pacientes) 91713 con 186 diferentes tipos de características
- Algunas de ellas son:
  - IDENTIFIER: Variables relacionadas al id del paciente y del hospital.
  - DEMOGRAPHIC: Datos de filiación y historia clínica del paciente.
  - APACHE covariate: Describe los síntomas y evolución del paciente durante las primeras 24 horas.
  - VITALS: Descripción de los signos vitales del paciente.

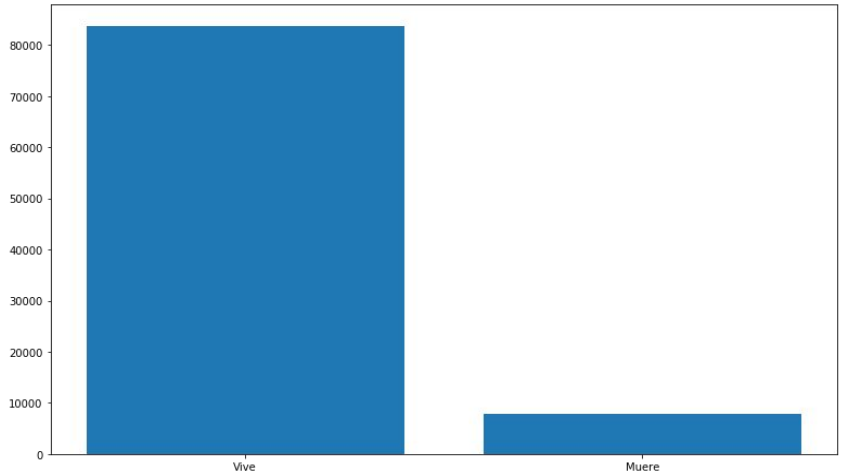
Análisis Exploratorio de datos

- 74 columnas con más del 30% missing values
- Gran cantidad de variables correlacionadas

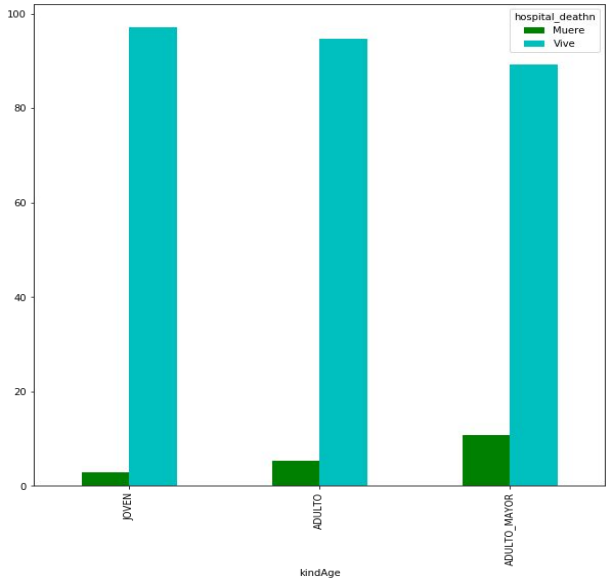




- La columna objetivo bastante desproporcionada 91.3698% del total de pacientes de nuestra data sobrevive y solo el 8.630% fallece



- Con respecto a la cantidad de pacientes que fallecieron, la frecuencia es mayor al incrementar la edad.



Preprocesamiento de datos

- Imputación de datos
- Selección de Features
  - Literatura[1]
  - Regularización

- Reducción de dimensionalidad
- Transformación de features

[1] Adapted from Knaus WA, Draper EA, Wagner DP, Zimmerman JE: APACHE II: A severity of disease classification system. Critical Care Medicine 13:818–829, 1985

Creación de los modelos

- Árboles de decisión (baseline)
- Naive Bayes
- **Cost-sensitive Regresión logística**
- **Cost-sensitive Random Forest**
- **Cost-sensitive SVM**
- Red neuronal profunda
- K Nearest Neighbors Classifier
- Stochastic Gradient Boosting
- Pila de clasificadores. (ensemble)

Ajuste de hiper parámetros / parámetros

Cost-sensitive RandomForest

Tuning	Max_features(parámetro)	n_estimadores (parámetro)	recall (score)
1	'sqrt' = 6	10	0.727455
2	'sqrt' = 6	100	0.736488
3	**'sqrt' = 6**	**1000**	**0.736909**
4	'log2' = 5	10	0.720296
5	'log2' = 5	100	0.730404
6	'log2' = 5	100	0.736394

Red neuronal

Tuning	n_neuronas	n_epoch	size_epoch	recall
1	**80**	**20**	**2000**	**0.1428**
2	60	20	2000	0.1334
3	80	20	1000	0.1245
4	60	20	1000	0.1123
5	80	10	2000	0.1399
6	60	10	2000	0.1366
7	80	10	1000	0.1253
8	60	10	1000	0.1233

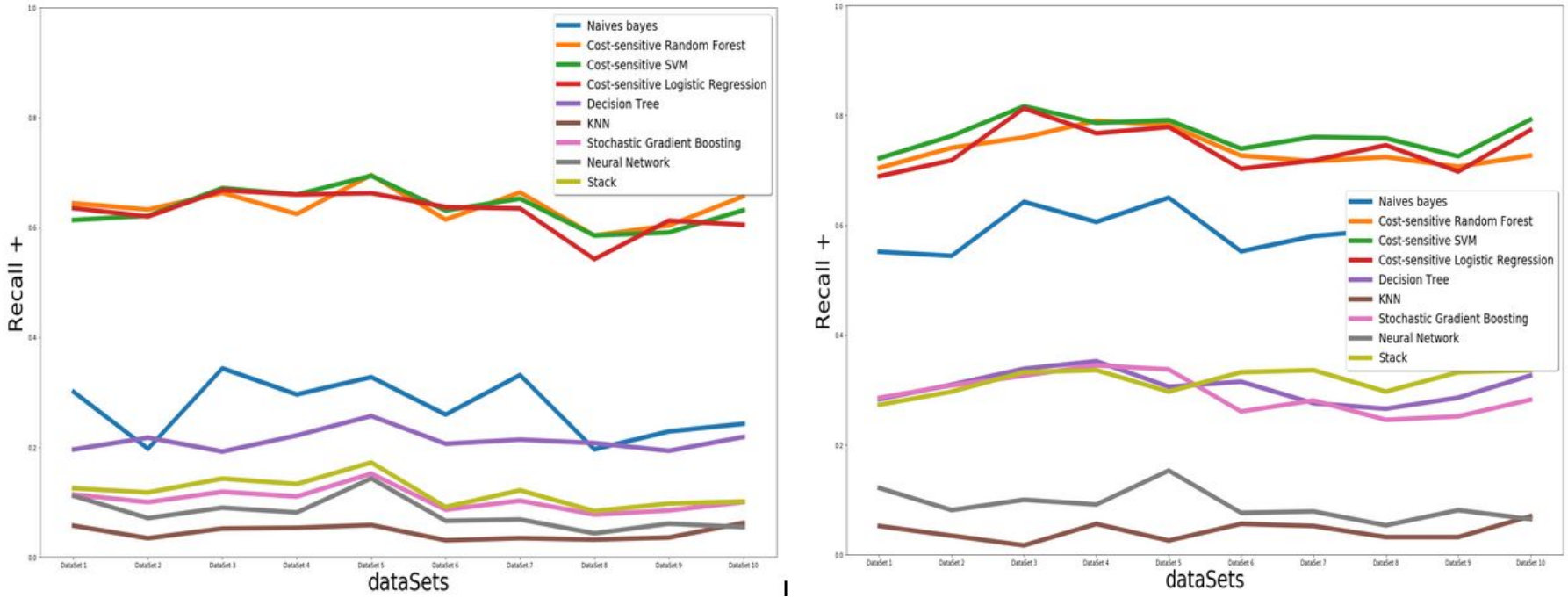
Análisis de Resultados

Modelo	Recall +	Recall -	AUC
Naive bayes	0.2721	0.9098	0.7309
Cost-sensitive Random Forest	**0.6382**	0.7278	0.7576
Cost-sensitive SVM	0.6351	0.7660	0.7319
Cost-sensitive Logistic regression	0.6276	0.7218	0.7188
Decision Tree	0.2121	0.9175	0.5652
KNN	0.0447	**0.9974**	0.7398
Stochastic Gradient Boosting	0.1043	0.9938	**0.7931**
Neuronal Network	0.0787	0.9159	0.7388
Stack(Ensemble)	0.1183	0.9923	0.7877

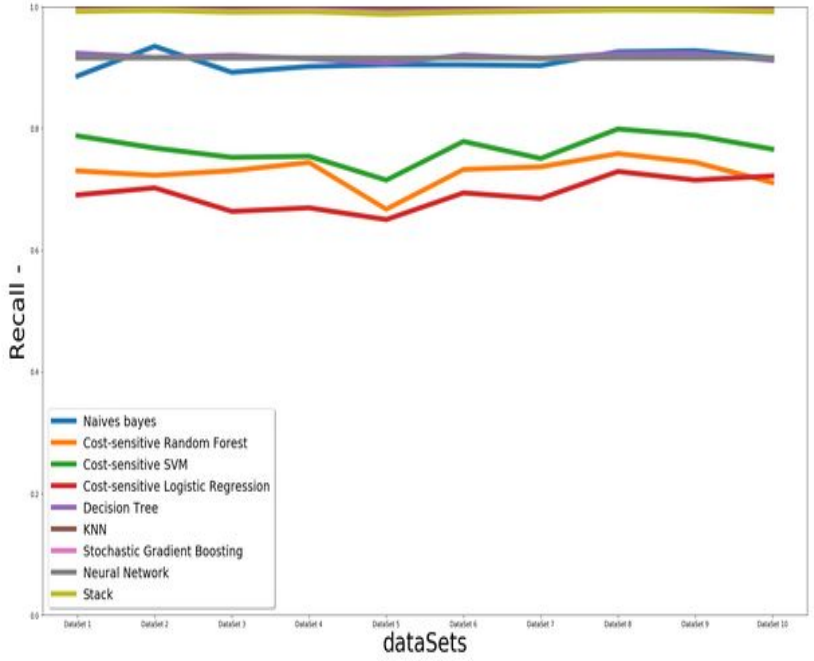
Tabla 1: Evaluación de los modelos con las variables obtenidas de la literatura[1]

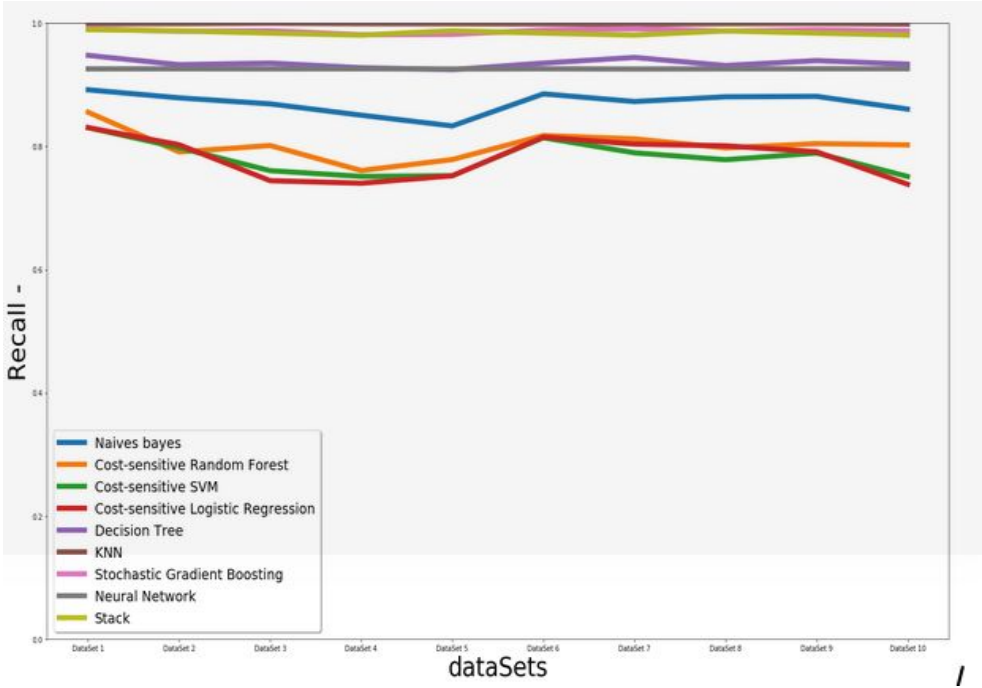
Modelo	Recall +	Recall -	AUC
Naive bayes	0.5860	0.8704	0.8368
Cost-sensitive Random Forest	0.7380	0.8022	0.8563
Cost-sensitive SVM	**0.7657**	0.7815	0.8368
Cost-sensitive Logistic regression	0.7406	0.7819	0.8448
Decision Tree	0.3056	0.9352	0.6204
KNN	0.0423	**0.9992**	0.7633
Stochastic Gradient Boosting	0.2920	0.9873	0.8803
Neuronal Network	0.0897	0.9259	0.7633
Stack(Ensemble)	0.3093	0.9855	**0.8804**

Tabla 2: Evaluación de los modelos con las variables obtenidas de regularización

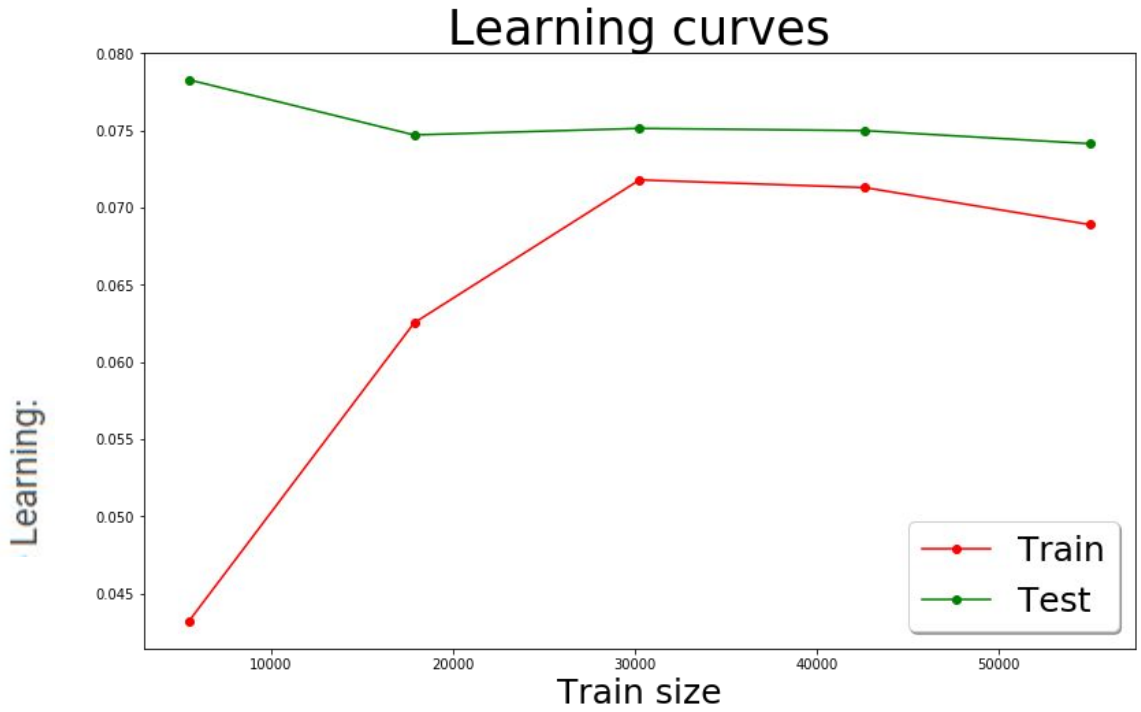


Gráfica: Rendimiento de los modelos usando features de [1](izquierda) y features obtenidos de la regularización(derecha)

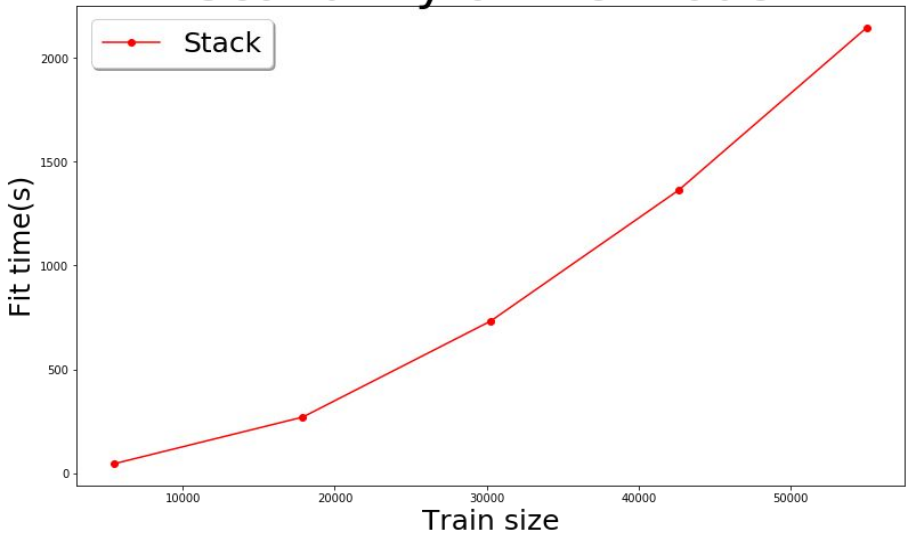




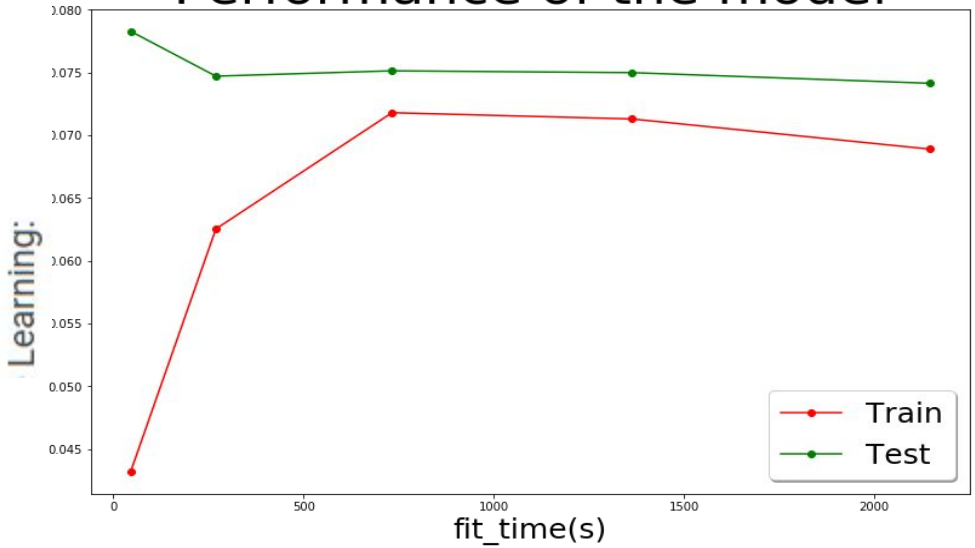
Gráfica: Rendimiento de los modelos usando features de [1](izquierda) y features obtenidos de la regularización(derecha)


Learning curves
Learning:
0.080
0.075
0.070
0.065
0.060
0.055
0.050
0.045
10000
20000
30000
40000
50000
Train size
Train
Test

Scalibitity of the model



Performance of the model



Conclusiones

- La regularización permitió incrementar significativamente nuestro métrica objetivo que era el recall negativo proveyéndonos de features más eficientes que aquellas obtenidas a través de la literatura.
- Aquellos modelos que fueron implementados con Cost-Sensitive tuvieron mejor rendimiento debido a que esto está diseñado para modelos de clasificación con la distribución del target no balanceada.
- El algoritmo de ensamble de clasificadores mejorar el recall positivo y cambio de decrementar el recall negativo, pero al final obtuvo el mejor equilibrio entre ambas métricas.

Recomendaciones

- Descartar samples que tengan mucho porcentaje de missing values. Ya que lo nuestros modelos no estuvieran aprendiendo de ellos valores reales sino más bien datos interpolados que más bien le puede causar ruido.
- Se podría buscar métodos de imputación de datos más avanzados que más cercanos al comportamiento de nuestra data.

¿ Preguntas?